APPLIED PARALLEL PROGRAMMING ON GPU

Project Tutorial

Dr. Alptekin Temizel

Spring 2018-2019

Term Project Guideline

1. Term Project Description

In the course, you will prepare a project by implementing a problem on GPU using CUDA or OpenCL. You can develop your own algorithm or modify the existing algorithms for GPU platforms. In either case you are expected to benchmark the algorithms.

The project could be done by a single student or by a group (2 students). The project complexity is expected to be higher when it is a group project.

Phase 1:

In the first stage of project,

- You will define your own problem. The problem shouldn't be too big or too small. Remember that this is a ~2 months project.
- Then, you will do a literature survey on the chosen problem and the topic. Your literature survey will define the related works on the problem, the software and computing environments for the algorithm implementations and your approach for the solution. It is encouraged that you choose a project related to your thesis or related to a project in your workplace. Literature survey should give you an insight about how new is your project. You need to discuss if the proposed project is suitable for GPU implementation.
- You need to upload your **proposal document** and the **presentation** to METU-Online by the deadline.

The deadline is 25th March 2019 @23:55h

There is no late delivery as you are going to make presentation in the class the next day!

You are expected to make a **10-minute presentation** on 26th March 2019 Monday (during the course hours). **We need to preload the presentations to the PC in the class to keep the time so please upload your presentation by the deadline. We have less than 8 minutes per project so your presentation should be short and concise (4-5 slides).**

This part corresponds to 10% of your overall course grade.

Phase 2:

In this phase you are expected to prepare a design document where you outline how you are going to parallelize your problem and how you are going to use memory.

In this phase you are expected to:

- Have decided on the algorithm
- Have understood the algorithm

Even though these could change, you should have:

- Decided on how you are going to **parallelize** your problem
- Decided on how you are going to use **memory**
- Decided on what **parameters** to investigate for performance
- Started to work on CPU implementation

From now on it is your work so you should:

- **Use your own words and don't copy and paste any text**
- Write equations yourself
- Draw the graphs and figures yourself

Draw a block diagram of the overview of the project, include any preliminary flow charts of your algorithms and discuss where you believe the parallelism in this algorithm is. Break down your project into components you plan to have assigned to host and device/kernels, list them and describe the interface, inputs and outputs of each module and what it accomplishes.

The evaluation will be based on the following criteria:

Originality: 20%
Algorithm Description: 15%
Parallelization Design: 25%
Memory Design: 20%
Performance Analysis Parameters: 20%

This part corresponds to 15% of your overall course grade.

The deadline: 23rd April 2019 @23:55h

Phase 3:

This is the implementation phase. You will implement the algorithms on GPU using CUDA or OpenCL. You are expected to provide a working project. The project must be fully functional and bug free. Parallelization is an important concept in GPU programming and you should have a parallel implementation and you should discuss **why** you chosen this method and **how** you designed.

You need to **analyze your results and provide objective analysis of performance gain**. For the comparison, you'll also need a CPU implementation. You should test your algorithm with various parameters (different data sizes etc.) and critically discuss the results and draw conclusions.

Then, you will introduce your results together with your literature survey in IEEE journal format;

(**http://www.ieee.org/web/publications/authors/transjnl/index.html**).

You can also refer to the Term Paper Guideline, which is presented in the next section.

You are also expected to prepare a poster (no need to print out, electronic copy will be sufficient) according to the guidelines:

http://www.gputechconf.com/page/call-for-posters.html

You can also find an example poster together with this document.

This part corresponds to 25% of your overall course grade.

Deliverables (only in electronic form):

1. Working code and all the related files
2. Project report document
3. Project poster presentation document

Delivery deadline is: 27th May 2019

You will be presenting your project on 28th May 2019. I will assign you a 15 minute slot.

Note: You don't need a Powerpoint presentation, you are expected to explain your project and your results on your poster displayed on computer screen. Note that a good poster presentation is very visual and has very little text and have illustrative figures, results etc.

***Policy for Copying**: Passing the work of others (either from another student or a code on internet etc.) off as your own is a breach of academic ethics and also of the University's disciplinary rules. When you submit a work it automatically implies that you claim the ownership of the work.*

Note that METU is subscribed to some tools which allow cross checking of submitted works as well as checking with any work on internet or any university subscribed to the system. No exceptions will be allowed and any work found to be copied will result in failing the course.

2. Term Paper Guideline

In preparing your project and writing your term paper, you should consider some important points. You can find several useful guidelines on the internet such as:

- http://www.lhup.edu/~dsimanek/termpapr.htm
- http://www.hkbu.edu.hk/~gis/termpap.htm
- http://www.stanford.edu/~armin/courses.guidelines.html

There is also a comprehensive summary for project preparation which is introduced in KDD 2009 conference (Research Track Paper Preparation and Submission Guidelines) which refer to the work of Prof. George Heilmeier:

In writing your paper, we suggest you try to address the following questions, credited to George Heilmeier:

- What are you trying to do? Articulate your objectives using absolutely no jargon.
- How is it done today, and what are the limits of current practice?
- What's new in your approach and why do you think it will be successful?
- Who cares?
- If you're successful, what difference will it make?
- What are the risks and the payoffs? (in other words, what are the limitations and strengths of your work)
- What are the midterm and final exams to check for success? (in other words, what are the measures of evaluation and evidence of success)

In light of the above principles, we *suggest* the following guidelines for the paper content. Note that the headings and the structure below are meant to be general categories; please exercise your discretion and creativity to make the paper as comprehensible as possible to the readers and reviewers.

i. Abstract

Try to include the following:

- Motivation: one or two sentences on the problem and it significance;
- Results: a short paragraph on approach and results;
- Availability: a link to code, data, and supplementary materials, or a statement why this is not possible.

ii. Motivation & Significance

What is the problem and why is it important or significant?

iii. Problem Statement

Formal definition of the problem with any preliminary concepts.

iv. Prior Work & Limitations

What are the existing approaches, and their limitations?

v. Theory/Algorithm

- Discuss the main theoretical or algorithmic ideas of the paper;
- Mention the main theorems (if any), the intuition behind those, and their practical application. Move the proofs to the appendix, unless the proof itself is the main contribution;
- Discuss your algorithmic solution (if any) at the conceptual level with pseudo-code, to convey the main ideas. Move minute (but practically important) implementation details to the appendix;
- Discuss why you chose certain paths, and discuss unfruitful paths that you discarded. In other words, give both the theoretical and/or algorithmic insights into your work.

vi. Experiments or other Evidence of Success

- Complete parameter settings and data descriptions should be provided (including any links to public resources);
- Clearly specify the experimental procedure, including evaluation measures;
- Compare to prior solutions, or at least to strawman solutions;
- Clearly discuss the results and what they mean;
- Only include the most relevant experiments here, using the appendix to provide any additional results (say on minor parameter tuning of your method, etc).

vii. Discussion and Future Work

Describe insights you gained, the limitations and applicability of your work, and directions for future research. Every solution has limitations, which should be explicitly mentioned.

viii. References

Include the most relevant works, making sure all citations are complete (including editors, publishers, page numbers, etc.).

ix. APPENDIX

You should use the appendix for supporting details. For example, you may use it to convey detailed technical/practical aspects of your implementation. You may use the appendix for theorem proofs, or for additional experimental results. Include include pointers in the main paper to relevant sections in the appendix.

The appendix *is* an integral part of the paper, since it will provide details that are important for a proper appreciation of your work (e.g., for replicating or extending it, or for comparison). However, it should be possible on a first read-through to get a good understanding of the paper's contribution from the main part alone. Structuring the paper in this way provides a service to the reader, by separating main ideas from technical details.