GPU Computing
Projects

E. Carlinet, J. Chazalon {firstname.lastname@lrde.epita.fr}

April 2020

EPITA Research & Development Laboratory (LRDE)

Slides generated on April 17, 2020

Instructions for the Project

Objectives

The goals of the project are to:

- apply data-parallelism concepts
- practice with CUDA
- set up a benchmark with a sound evaluation procedure
- present your results in a clear and convincing way

Possible Subjects

Standard Option

We propose 1 subject, that most of you should work on:
Implementation and performance analysis of a Graph Cut algorithm in CUDA

This is an important image processing algorithm, and will be described briefly in later slides.

Special Option

For students who are at ease with CUDA, and want to investigate a particular question:
Implementation and performance analysis of *SOME INTERESTING* algorithm in *YOUR PARALLEL PROGRAMMING TECHNOLOGY OF CHOICE*

If you choose this way, you must **validate your subject with us before April 24th.**
Contact us by email.
We expect:

1. a clear statement of the problem you want to work on;
2. a detailed work plan.

Our Expectations

We expect your implementation to be:

- running on GPU;
- correct, ie to produce an acceptable result.

Do not try to make it fast at first, just make it work.

Then, try to apply NVidia's Assess, Parallelize, Optimize, Deploy (APOD) design cycle as described in their CUDA C++ Best Practices Guide:

1. identify the part of the code which is responsible for the bulk of the execution time;
2. use all available weapons (CUDA API, libraries, research papers) to obtain a parallel version of the code (assumed to be sequential at first);
3. use all available weapons (CUDA API, libraries, research papers) to optimize the performance of the parallel code;
4. measure the performance of the new code.

Project Outline for Standard Performance Analysis

Broad Outline	Concrete Example
Choose an application	Mandelbrot
Determine the most time-consuming part of the app	
Determine one or more data-parallel approaches to solving the problem	Tiling...
Create multiple implementations of the approach	One naïve version, one version with shared memory...
Benchmark the implementations	Record memory transfer time, kernel time, utilization, FLOPS, etc.
Relate results to course concepts	Identify the cause of the bottleneck (memory or compute bounding)

Teams

Teams of 3 (plus one team of 4).

~~**Everyone** must **register** on the course in **Moodle**.~~

1 team member must **submit information about your team before April 24th**.

Each team participant will to **describe his/her work in the project**.

Final Deliverables (1/3)

1. Implementation

- Source code for C++ CPU reference
- Source code for CUDA implementation(s)
- Source code for benchmark tools
- Build scripts (GNU Make, CMake…)

Final Deliverables (2/3)

2. Report

- Description of the problem
  - Detailed if custom subject
  - Quick summary otherwise
- Quick description of the baseline CPU implementation (paper reference, parallel or not, etc.)
- Quick description of the baseline GPU implementation (same as CPU baseline)
- **Justification of the performance indicators you have used**
- **Analysis of performance bottlenecks** (with measured indicators, graphs, etc.)
- **For each improvement over the GPU baseline (implementations):**
  - justification of this work regarding performance analysis
  - description of the improvement (ex: used output privatization instead of global atomics)
  - comparison of the performance with and without this implementation
- Table with summary of the benchmark
- Summary of who did what (contribution of each team member)

Final Deliverables (3/3)

3. A live lecture / defense

- 15' presentation
- 10' discussion

Submit 1 & 2 on Moodle before June 30th.

Grade Sheet Used for Last Session

	Travail effectif	Présentation	Note individuelle
Total	**10**	**5**	**5**
Critères	Difficulté technique (bonus/malus) Qualité de l'implémentation (au regard des notions vues en cours) Qualité de l'évaluation, du benchmark (2 implém. CPU, 1+ GPU…) Prise de recul, analyse (identification des goulots d'étranglement…)	Qualité du support Clarté de l'exposé – Pédagogie Mise en contexte Positionnement et état de l'art	Prise de parole Réponse aux questions Implication dans le groupe
Repères			
A [16-20]	Benchmarks cohérents Au moins 3 implémentations (dont 2 variantes GPU) Utilisation de nvprof et/ou nsight (présentation graphique) Description des techniques utilisées Explication des différences de performance Idenfification des pistes d'amélioration	Oral de qualité, apprécié de l'auditoire Structure claire Illustrations claires Bonne maîtrise du temps	Maîtrise globale du sujet Réponses pertinentes aux questions Leader de groupe (A+)
B [12-16[	Bonne implémentation (versions CPU et GPU avec gain) Au moins un benchmark entre les 2 implémentations Identification d'au moins un facteur de ralentissement (mais pas forcément de solutions) MAIS une seule version GPU	Présentation retenant l'attention (B+) Présentation orale honnête (B-)	Participation honnête au projet Capable de faire appel aux notions de cours
C [8-12[	Minimum attendu Version GPU fonctionnelle MAIS manque de prise de recul sur les gains possibles MAIS manque d'analyse de la performance OU pas de version CPU	Présentation passable (C+) Présentation décevante (C-)	Présentation confuse Passif lors des questions Maîtrise incomplète du sujet
D [5-8[	Clairement en dessous du travail attendu Implémentation GPU non fonctionnelle	Mauvaise présentation orale	Travail très faible Mauvaise compréhension du sujet
E [0-5[	Pas de travail significatif Pas d'implémentation GPU	Inacceptable / absent / …	Inacceptable / absent / …

Defenses

Defenses will be held in the **beginning of July** (exact date TBA).

We will use *Teams* to meet.

The participation of all team members is required.

Moodle Links

Course page:
`https://moodle.cri.epita.fr/course/view.php?id=196`

Submit team composition + subject:
`https://moodle.cri.epita.fr/mod/assign/view.php?id=2350`

Course feedback:
`https://moodle.cri.epita.fr/mod/feedback/view.php?id=2351`

Final project submission:
`https://moodle.cri.epita.fr/mod/assign/view.php?id=2352`

Summary of Tasks and Deadlines

What	Deadline	Who
~~Register on Moodle~~	~~April 24th~~	~~Everyone~~
Submit team composition + subject	April 24th	1 person/team
Submit extra subject justification if needed	April 24th	1 person/team
(opt.) Complete the feedback form	April 30th	Everyone
... Work on your project...		
Submit code + report + presentation slides	June 30th	1 person/team
Defend your project (live presentation)	Beginning of July	Everyone

Focus on the Benchmark

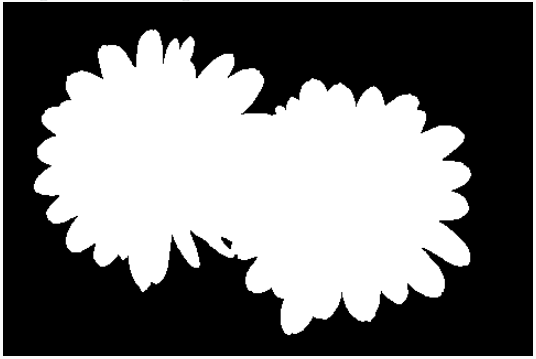
Dataset

We provide you with a **dataset of 15 images** to test the performance of your approach(es).

How to get it:
`https://www.lrde.epita.fr/~jchazalo/SHARE/segmentation_dataset.tar.gz`

Sample input

Expected output



You Need to Complete This Dataset

1. For each input, **generate sample input interactions** to select foreground and background pixels.
   - Save them to a fake interaction image.
2. Use this new image to initialize your algorithm along with the input.
3. Check the results are acceptable (not totally different from ground truth).

Use It!

Your algorithm / code section being benchmarked should be a function:

- taking an image, a set of foreground and a set of background pixels
- returning an image of foreground and background pixels

We expect you to:

- report performance indicators using this dataset;
- illustrate your report and presentation with samples from this dataset.

About Graph Cuts

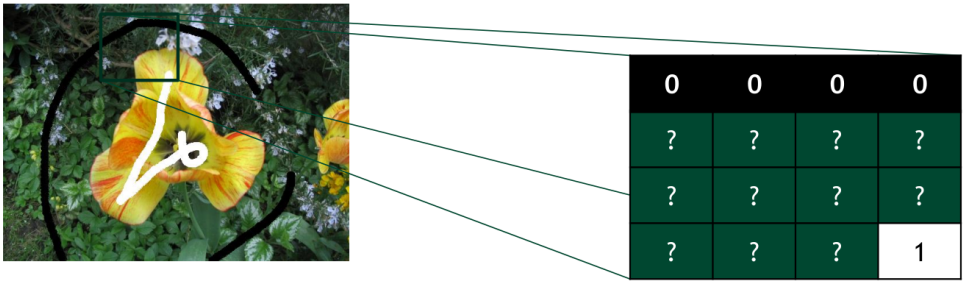
Overview

A graph algorithm.

Used in (interactive) image processing:

1. manually select some pixels belonging to an object (“foreground”)
2. manually select some pixels not belonging to the object (“background”)
3. **automatically find the best frontier between foreground and background (global optimum)**
4. *loop to 1 if result needs improvements*

Illustration



User marks ***some*** pixels as Background and Foreground

Compute ***for all*** pixels if they are Background or Foreground

Recommended Readings

- Timo Stich, “Graph Cuts with CUDA”, Presentation at the NVidia GPU Technology Conference, San Jose, 2009.
- Wikipedia *push-relabel max flow algorithm*
- Goldberg, Andrew V.; Tarjan, Robert E. (1988). “A new approach to the maximum-flow problem”. Journal of the ACM. 35 (4): 921.

Implementations Hints (Final Reminders)

- Have a working (*slow*) C++ reference implementation first (and keep it forever)
- Tag (*git tag*) the versions of your program before any optimization (useful to track and benchmark ideas)
- Try optimizations step by step so that you can tell which ones are the most important