Problem Collection

Programming and Architecture of Computing Systems

January 16, 2021

Contents

Preliminary Notes

This brief collection of problems is divided in three parts. The first part covers small questions, the second part includes test questions, and the third part refers to some longer exercises.

To report erratas, typos... please mail either alvabre@unizar.es, rgran@unizar.es or dario@unizar.es.

Small Questions

1. Please briefly respond to the following questions: ¿Is a concurrent application always parallel? ¿Is a parallel application always concurrent?
2. According to Amdahl's Law, for a program where the sequential part represents the 15% of the total, what would be the potential speed-up for a 16-core machine.
3. Can a processor execute instructions from two different instruction sets?
4. Enumerate what are the key design features of GPUs to allow a very fast context switching of wavefronts.
5. Explain what the conditional branching problem is on GPUs and how it is solved.
6. Make comparative analysis between a GPU and an ASIC.

Test Questions

1. The Local Data Share (LDS) cache on a GPU is used to:
   a) Amplify the regular cache bandwidth
   b) Execute atomic instructions
   c) Synchronization of wavefronts
   d) All of the above
   e) None of the above
2. In OpenCL, Local Memory is shared between:
   a) All the workitems of a global work domain
   b) Workitems in the same kernel launch

c) Local Memory is an abstraction not present in OpenCL

d) Workitems in the same workgroup

e) None of the previous ones

3. In OpenCL, workitems that access global shared variable must explicitly assure memory order in order to avoid race conditions

a) Always

b) Just in case they do not belong to the same workgroup

c) Just in case they do belong to the same workgroup

d) Never

Exercises

1. The dot product algorithm takes two vectors of the same length and returns a single number. The number is the sum of the products of the corresponding entries in the input victors.

   In C++, the algorithm can be coded as follows:

```
template<typename T>
T doc_product(const std::vector<T> &a, const std::vector<T> &b>
{

if(a.lenght() != b.lenght()) {
    error(...);
}

// initialize to 0 regardless the type
T dot_p{}; // Also T dot_p = T();

for(size_t i = 0; i < a.length(); ++i) {
  dot_p+=(a[i]*b[i]);
}
```

   Please answer the following questions:

   a. Implement the doc product using threads and static partitioning.

   b. Implement the doc product assuming you have the thread pool and the thread-safe queue from Laboratory 4.

   c. For the thread-pool version, would all tasks perform the same ammount of work?

2. Given an `std::vector<int>` array, could you please write a parallel algorithm that finds the minimum and maximum values of the array.

3. See Exercise 2 from the collection of exercises referring to metrics.

4. Write an OpenCL program that calculates the dot product of to integer arrays. Additionally to the kernel code, in the host side of the program, just focus on the buffer management, command-queue management and kernel launch.

   a. Please, analytically model the execution time of this work assuming the computational device has the following characteristics: 8 compute units, each compute unit has 128 parallel cores, each core has two floating-point arithmetic units and, frequency of the computational device is 1.5GHz