Prof. Dr. C. Stamm christoph.stamm@fhnw.ch Tel.: 056 202 78 32

Exercise 3

1 Cost-Optimality and Minimum Execution Times

Consider a parallel system with *p* processing elements solving a problem consisting of *W* units of work in time $T_p = \frac{\sqrt{W}}{2} + \frac{W}{p}$.

a) Compute the cost, overhead, speedup, efficiency, and isoefficiency for the parallel system.
b) Is the parallel system scalable?
c) Can the problem be solved cost-optimally for $p = \Theta(W)$?
d) Assume $p < \Theta(W)$, can now the problem be solved cost-optimally? If yes, what is a valid upper bound for *p*?
e) Let $C(W) = ½W$ be the degree of concurrency. Compute the minimum execution time.
f) Compute the lower bound on the parallel runtime for solving the problem cost-optimally.

2 Cost-Optimality and Isoefficiency

Consider a parallel system with *p* processing elements solving a problem consisting of *W* units of work. Prove that if the isoefficiency function of the system is worse (greater) than $\Theta(p)$, then the problem cannot be solved cost-optimally with $p = \Theta(W)$. Also prove the converse that if the problem can be solved cost-optimally only for $p < \Theta(W)$, then the isoefficiency function of the parallel system is worse than linear.

3 Vector Operations

In Exercise 2 you programmed an OpenCL kernel `edges(…)`. Because the edge detection is concurrently computed on three image channels, this kernel offers several possibilities to use built-in math and vector operations. Rewrite the kernel code by using built-in math and vector operations.

4 GPU Programming with "Single Source" Approach

There are different technologies available to program GPUs and all of them produce different GPU code. Hence, it is helpful to use another technology for the same problem and to compare the results, the performance, and the implementation cost.

Windows and Visual Studio users should try to use C++ AMP (Accelerated Massive Parallelism). AMP is fully integrated in Visual Studio. A template file "amp.cpp" is already included in the given source code. Make sure that C++ Language Confirmation Mode isn't set to permissive and use Release Configuration!

Linux users should try to use OpenACC. The OpenACC API describes a collection of compiler directives (like OpenMP) to specify loops and regions of code in standard C/C++ to be offloaded from a host CPU to an attached accelerator, providing portability across operating systems, host CPUs and accelerators. More information about this approach is on the following web site. Please note, that OpenACC compilers are only free for thirty days: http://www.openacc-standard.org. A template file "acc.cpp" is already included in the given source code.

Another approach you can try is SYCL. Some SYCL implementations make use of OpenCL and therefore you need a working OpenCL installation on your system:

- Codeplay
  - ComputeCpp
  - Playground (online programming and testing)
  - Playground (online programming and testing)
- Intel oneAPI
  - Base Toolkit (includes SYCL)
  - DevCloud (virtual development sandbox to learn program oneAPI cross-architecture)