O título vem aqui

**E o subtítulo aqui**

André Pereira pg19828, Rafael Silva pg20202

UCE de Computação Gráfica

Departamento de Informática

Universidade do Minho

# Abstract

Two parallelization techniques were adopted to parallelize a scientific application for reconstructing particle collisions. One, for a multicore shared memory environment, using OpenMP, and other for a heterogeneous parallel system with a GPU, using CUDA. A naïve implementation for both environments was made and presented in a previous report. In this report, the focus will be on analyzing the parallel region and identifying its bottlenecks, only for the multicore shared memory implementation. Two approaches to overcome these bottlenecks and a final performance analysis of the implementation are presented.

# Introdução

**The Problem**

The problem Most of the human activities produce waste, which we commonly call trash. Trash is one of the problems of the global community, is usually taken to sanitary landfills where they are accumulated, polluting the environment. Aiming to reduce the environmental impact created the policy of the 3R's, which proposes that everyone should reduce the amount of waste it produces (e.g. use of products with bigger packages to minimize the amount of packaging waste), Reusing resources (e. g. reusing supermarket bags for packaging trash), and lastly and most importantly in the context of this project: Recycling, or, reuse a material as feedstock to another. For example, in the city of Porto and metropolitan area 488 216.35 tons of waste were produced in 2011, of those 53796.73 tons were recycled [3], saving the environment.

For recycling to be viable it is necessary that citizens separate recyclable trash from undifferentiated trash, we propose a game that promotes recycling as something positive by using augmented reality.

**The Application**

Using the library ARToolKit, we propose the development of a game in which players use patterns to move gloves that catch the trash. Each glove is a type of recyclable waste (eg packaging, paper and cardboard, glass and undifferentiated trash).

The goal is that the player can correctly associate the trash to its type, to deposit it in the correct container.

The game will be targeted to children, because they are susceptible to the game market, and they are more influenced by games.

# Dificuldades de implementação

Como prova de conceito desenvolvemos uma aplicação com as especificações indicadas, utilizando a biblioteca ARToolkKit.

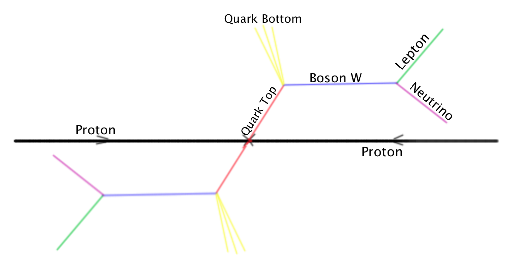


Figure 3.1 – Resulting particles from a head-on protons collision.

Since this system obeys a set of physic properties, the characteristics of the neutrino can be obtained by resolving a system of equations, based on the other particles characteristics, since its momentum and energy is conserved during the collision. The dilep routine solves this system using analytical methods. The objective is to run the dilep as many times as possible so that, within each event, a variance to the characteristics of the detected particles can be applied. This would help to take into account the experimental resolution of the detector and produce more physically accurate results for the neutrino, and system, characteristics.

# Testing Methodology

**Testing Environment**

To obtain the necessary data for the information presented in the next sections the Performance API (PAPI) was used to access the CPU hardware counters. The time measurements were made using the function clock from GLIBC, the GNU C library , with millisecond precision (since each execution of the application was in the order of minutes, further precision is not required), as the system used for these measurements did not have the PAPI library available.

|  |  |  |
| --- | --- | --- |
| **CPUs** | **L1 Cache** | **L2 Cache** |
| Intel Xeon 5130 | 32 KB I.  32KB D. per Core | 4 MB shared |
| ***# Cores*** | ***# Threads*** | ***Clock Freq.*** |
| 2 | 2 | 2.0 GHz |

Table 4.1 – CPU on the multicore PAPI test system, compute-201-1.

All tests were executed in a linux Machine, the distribution was Cent OS 5.3, the linux kernel version was 2.6.32-71.29.1.el6.x86\_64 and the code was compiled with gcc 4.4.6.

**Testing Methodology**

The tests are divided between two systems because the scaling of the parallel region cannot be properly measured in the compute-201-1 system, as it only has 4 cores (with no multithreading) and its technology is outdated. The compute-611-2 system offers a better environment to study the parallelization scaling, up to 24 threads with multithreading.

# Conclusion

We were presented with a scientific application, which the objective was to improve its performance. Instead of determining in which region of the code the application was spending the most computational time and resources, the goal was to increase the number of executions of a specific function, dilep, which, in the original application, was not the bottleneck. However, when the number of dilep executions was increased, as presented in the previous report, it quickly became the routine spending most of the time in the application.

References

1. LIPOR – Serviço Intermunicipalizado de Gestão de Resíduos do Grande Porto, *Relatório de Análise Estatística LIPOR*, [2012?], retirado em 10-06-2012.
2. .
3. .