A Target-Aware Compiler to Translate MATLAB into C and OpenCL*

Luís Reis, João Bispo, João M.P. Cardoso
Faculdade de Engenharia (FEUP), Universidade do Porto, Portugal, and
INESC-TEC, Porto, Portugal
{luiscubal, jbispo, jmpc}@fe.up.pt

Keywords—MATLAB, Source-to-Source Compilers, OpenCL, High-Level Synthesis

MATISSE [1] is a MATLAB source-to-source compilation framework that is capable of generating C and OpenCL code. Generating OpenCL allows us to cover a wide range of devices, including multi-core CPUs, GPUs and FPGAs. However, OpenCL programs are typically not performance-portable. In particular, OpenCL programs require highly specific tuning to exhibit acceptable performance on FPGAs.

MATISSE is capable of compiling a non-trivial subset of MATLAB and generates efficient C code in many cases. We support specializing code generation for specific platforms and devices (*target-aware* compilation), so we are able to generate code for a wide range of devices, provided we can identify the patterns that are ideal for each platform and know how to transform the code to fit those patterns.

Some of the properties that are important to determine for each device include:

(1) Whether to vectorize the code explicitly, as some OpenCL implementations already do so and others don't benefit from it at all;
(2) How to distribute tasks across threads, such as whether to run as many work-items as there are tasks or run a fixed number of work-items and distribute tasks among them;
(3) Target-specific extensions and how to structure the code to use these extensions. For instance, to achieve good performance on FPGAs, it is important to use the pipelining attributes. However, the implementation may further require dividing the kernel into multiple stages (e.g., load data to BRAM, compute, store results) for pipelining to work;
(4) Target different OpenCL versions (e.g., to determine whether `work_group_reduce_add` can be used);
(5) Target different hardware/driver capabilities, such as Shared Virtual Memory (SVM);
(6) Whether to pass the source code to the OpenCL API or run it through an OpenCL compiler and use the resulting binary.

In this presentation, we demonstrate the capabilities and challenges of the OpenCL backend of MATISSE, including general and target-aware optimizations. We will also explore the architecture of our compiler framework.

The MATISSE website demonstrates our compilation capabilities and will be used in this presentation. It can be accessed at http://specs.fe.up.pt/tools/matisse/.

```
%!parallel
function Z = vectoradd(X, Y)
    Z = X + Y;
end
```

Figure 1 Vector addition with MATISSE parallelization directive.

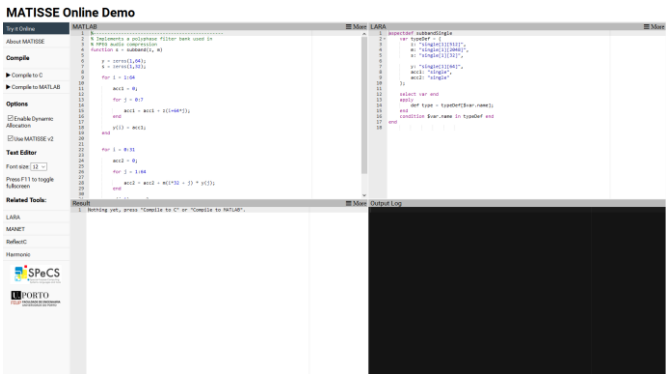

Figure 2 MATISSE Demonstration website

```
kernel void vectoradd(global int* X_data, global int* Y_data, global int* Z_data,
int N) {
    size_t global_id0 = get_global_id(0);
    if (global_id0 < N) {
        int i = global_id0 + 1;
        Z_data[i - 1] = X_data[i - 1] + Y_data[i - 1];
    }
}
```

Figure 3 Simplified generated OpenCL code for example in Figure 1.

REFERENCES

[1] L. Reis, J. Bispo, and J.M.P. Cardoso, SSA-based MATLAB-to-C Compilation and Optimization. In *Proceedings of the 3rd ACM SIGPLAN International Workshop on Libraries, Languages, and Compilers for Array Programming (ARRAY 2016)*, Santa Barbara, CA, USA, 2016.

This work has been partially supported by the TEC4Growth project, "NORTE-01-0145-FEDER-000020", financed by the North Portugal Regional Operational Programme (NORTE 2020), under the PORTUGAL 2020 Partnership Agreement, and through the European Regional Development Fund (ERDF). It has also been partially supported by the ANTAREX project, financed by the EU H2020 FET-HPC program under gran 671623. Reis and Bispo acknowledge the support of the Portuguese Fundação para a Ciência e Tecnologia (FCT), through the PhD scholarship PD/BD/105804/2014 and post-doctoral scholarship SFRH/BPD/118211/2016, respectively.