A Domain-Specific Aspect Language for MATLAB and its Strategic Programming Weaver

Ricardo Nobre
Universidade do Porto
Faculdade de Engenharia (FEUP)
Porto, Portugal
INESC-ID,
Lisboa, Portugal
ricardo.nobre@fe.up.pt

Tiago Carvalho, João M. P. Cardoso
Universidade do Porto
Faculdade de Engenharia (FEUP)
Porto, Portugal
tdrc@fe.up.pt, jmpc@acm.org

Pedro C. Diniz
INESC-ID,
Lisboa, Portugal
pedro@esda.inesc-id.pt

Abstract: The MATLAB programming language has been widely adopted in many embedded system projects, from control to digital/image processing systems, as a vehicle to model and validate through simulation main system functionalities. While MATLAB offers a high-level of abstraction and validation, C is still the de facto standard programming language for embedded systems, as MATLAB’s interpreted execution paradigm with its highly dynamic type implementations, still substantially hampers its execution performance. The current practice is, thus, for designers to manually translate and further validate MATLAB codes to C. This work explores an alternative approach that relies on augmenting MATLAB codes with type, array shape information and dynamic properties, which are key to support an automatic and efficient translation of these codes to C. Developers could thus benefit from the expressive power of MATLAB while relying on traditional compilation and optimization techniques for performance with guaranteed translation correctness.

This poster presents the main concepts of a domain-specific aspect language for specifying data types and shapes and code to be injected in MATLAB code. The aspect language allows the programmer to define new versions of a base MATLAB program without invasive modifications to the original MATLAB program. We present an elegant and efficient strategic-based weaver for MATLAB programs using aspects. Moreover, we analyze several applications using our approach, namely by introducing logging information and type specialization in MATLAB programs. Finally, we present the experimental results of performing type specialization via aspects considering the translation to C of two MATLAB codes used in real industrial codes.

Keywords: MATLAB, Compiler optimizations, synthesis, hardware/software cores
A Domain-Specific Aspect Language for MATLAB and its Strategic Programming Weaver

Ricardo Nobre\textsuperscript{1,2}, Tiago Carvalho\textsuperscript{1}, João M. P. Cardoso\textsuperscript{1} and Pedro Diniz\textsuperscript{2}
\textsuperscript{1}Universidade do Porto, Faculdade de Engenharia (UPORTO), Porto, Portugal \textsuperscript{2}INESC-ID, Lisboa, Portugal
ricardo.nobre@fe.up.pt, tdc@fe.up.pt, jmpc@acm.org, pedro.esda.inesc-id.pt

MATLAB in Embedded Systems

- MATLAB is widely adopted in many embedded system projects, in the domain control and digital/image signal processing, in early solution prototyping and validation via simulation.
- MATLAB’s high-level of abstraction assures it is easy to use but highly dynamic type implementations still hamper substantially its execution performance.
- Effective use as an efficient embedded software solution requires MATLAB abstractions such as data types, array shapes, and dynamic properties to be resolved and refined at compile time.
- Our approach uses a domain-specific aspect-oriented language (LARA) to convey to MATLAB secondary concerns and data type/shape specialization.
- Enables a compiler to generate C code compliant to target hardware synthesis tools and generate C code to be compiled for the target hardware architecture.

MATLAB Compiler Infrastructure

Weaving actions and code generation engines use strategic programming techniques.
Tom (http://tom.loria.fr/), is used to program transformations.

MATLAB Front-End

Weaving Actions: Type/Shape Assignment

Example of an aspect to assign a fixed-point data type to variables.

\begin{verbatim}
aspectdef IntegerTypeDef
    vars: select var(*) end //select all variables
    apply to vars def/line type="fixed(20,16)"
end
\end{verbatim}

The weaver transforms the input code according to the above aspect.

Experimental Results

Conclusions

- Aspects are used to describe secondary concerns. They are used to define types and shapes, properties, and to insert code (e.g., for monitoring, for producing different code versions).
- Aspects promote efficient translation of MATLAB to C code, which is of paramount importance in the embedded computing domain.
- Aspects promote clean and unpolluted MATLAB code ready to be translated to C code.
- Aspects allow separation of concerns, e.g., debug code separated from production code.

DATE’12 Friday Workshop: Fourth Friday Workshop on Designing for Embedded Parallel Computing Platforms: Architectures, Design Tools, and Applications
http://conferenze.dei.polimi.it/depcp/
March 16, 2012, Dresden, Germany