Just-in-time adaptive decoder engine: a universal video decoder based on MPEG RVC

Jérôme Gorin, Hervé Yviquel, Françoise Prêteux, Mickaël Raulet

Just-In-Time Adaptive Decoder Engine: A Universal Video Decoder based on MPEG RVC

Jérôme Gorin
ARTEMIS, Institut
Telecom/Telecom SudParis
UMR 8145, SudParis, France
jerome.gorin@etu.upmc.fr

Hervé Yviquel
IRISA, University of Rennes 1
6 rue de Kerampont
22300 Lannion, France
herve.yviquel@irisa.fr

Françoise Prêteux
MINES ParisTech
60 Boulevard Saint-Michel
75272 Paris, France
Francoise.Preteux@mines-paristech.fr

Mickaël Raulet
IETR, INSA Rennes
F-35000,
mickael.raulet@insa-rennes.fr

ABSTRACT

In this paper, we introduce the Just-In-Time Adaptive Decoder Engine (Jade) project, which is shipped as part of the Open RVC-CAL Compiler (Orcc) project. Orcc provides a set of open-source software tools for managing decoders standardized within MPEG by the Reconfigurable Video Coding (RVC) experts. In this framework, Jade acts as a Virtual Machine for any decoder description that uses the MPEG RVC paradigm. Jade dynamically generates a *native* decoder representation suitable for X86, ARM and CELL platforms with a possibility of exploiting multi-core CPUs. Thus, according to the MPEG RVC decoder description coupled with a video coded stream, Jade can create, configure and re-configure video decompression algorithms adapting to the video content.

Categories and Subject Descriptors

4 [**Multimedia systems and middleware**]: Miscellaneous; 3 [**Scalability in media processing, analysis, and applications**]: Miscellaneous

General Terms: Algorithms, Languages, Performance, Standardization.

Keywords: Adaptive decoding, multimedia application, dataflow programs, decoder reconfiguration, scalable execution, MPEG Reconfigurable Video Coding (RVC).

1. INTRODUCTION

MPEG Reconfigurable Video Coding (RVC) has been chosen by MPEG to be an alternative paradigm for codec deployment. The goal of MPEG RVC is to provide dataflow representations of existing reference software decoders at component level. Thus MPEG RVC enables arbitrary decoder representations of any combination of fundamental algorithms, without additional standardization steps. By adding the side-information of the decoder description alongside the video coded bitstream, MPEG RVC defines the new concept of an RVC decoder. As shown on Fig. 1, an RVC decoder can create and handle several decoder descriptions on-the-fly, either by using coding tools standardized in MPEG, or proprietary implementations of coding tools, or other hybrid versions composed from proprietary and standardized implementations.

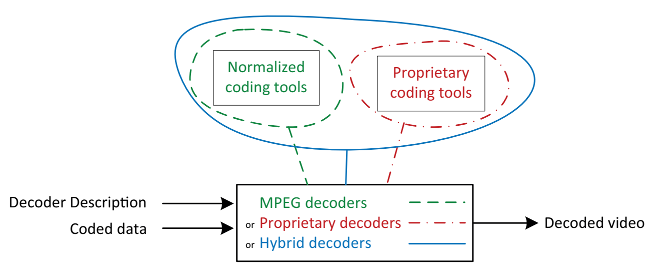

Figure 1: Representation of a dynamic MPEG RVC decoder.

In this paper, we introduce the Just-In-Time Adaptive Decoder Engine (Jade) as the first *portable* decoder engine capable of managing *on-the-fly* MPEG RVC decoder descriptions. The goal of Jade is to exploit the generic properties of a dataflow representation to produce an efficient *native* decoder representation suited to the architecture of the underlying platform. Jade can create, configure and re-configure the video decompression algorithms adaptively to the coded video bitstream, avoiding incompatibility issues between decoders and coded bitstreams. This concept has significant advantages compared to a "statically" compiled decoder: a Just-In-Time decoding platform does not need decoder software updates. Moreover, RVC descriptions are generic and provide sufficient information to produce an efficient implementation suited for a wide range of platforms.

In the following, section 2 gives an overview of the MPEG

MM'11, November 28–December 1, 2011, Scottsdale, Arizona, USA.

RVC standard and the property of its description. In section 3, we provide a presentation of the main concept of Jade and its implemented features. Finally, we give use procedure in section 4.

2. MPEG RVC OVERVIEW

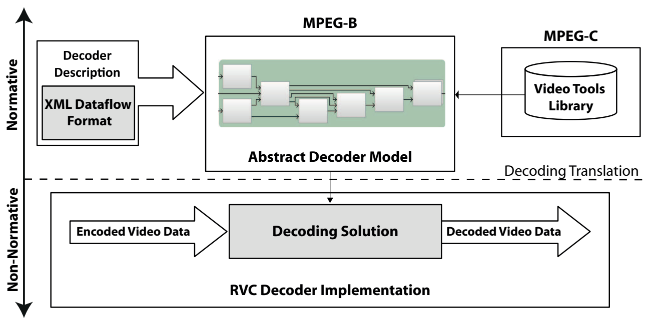

Figure 2: A typical use of the MPEG RVC Framework.

The MPEG Reconfigurable Video Coding (RVC) [16] framework is a new, still evolving ISO standard from MPEG. It aims at providing video codec specifications based on a high-level library of components. The main goal of MPEG RVC is to produce new *Abstract Decoder Model* (ADM) of already existing MPEG standard which will be suitable for both hardware and software implementation [16]. An ADM is a generic representation of a decoder, built as a dataflow diagram of coding tools encapsulated into Functional Units (FUs). Dataflow representations are expressed with the *XML Dataflow Format* (XDF), an XML dialect standardized in MPEG-B part 4 [16] (*Codec configuration representation*) that describes the connections between FUs. Each FU is described in RVC-CAL Actor Language (RVC-CAL) and defines a processing entity of a decoder.

Figure 2 shows a typical use of a normative ADM description to produce a non-normative decoding solution that can target either software or hardware platform. MPEG RVC provides both a normative standard library of FUs standardized in MPEG-C part 4 [16], the *Video Tools Library* (VTL), and a set of decoder descriptions/configurations expressed as networks of FUs. Such a representation is modular and helps the reconfiguration of a decoder by modifying the topology of the network. Adding new coding technologies in an existing standard is a particularly sensitive part of any standardization process. As a consequence, MPEG RVC mainly focuses on the reusability of the standardized coding tools by allowing different decoder descriptions to instantiate common FUs across standards.

3. JUST-IN-TIME ADAPTIVE DECODER ENGINE FEATURES

Jade [13, 12], represented in Fig. 3, extends a Virtual Machine to handle a RVC decoder description. Its configuration engine (Fig. 4) has two inputs, a decoder configuration and a representation of the Video Tools Library (VTL) [11] standardized in MPEG-C. It outputs a complete dataflow representation of the decoder as a set of interconnected functional processing units in byte code format. This decoder representation can then be compiled or interpreted by a specific

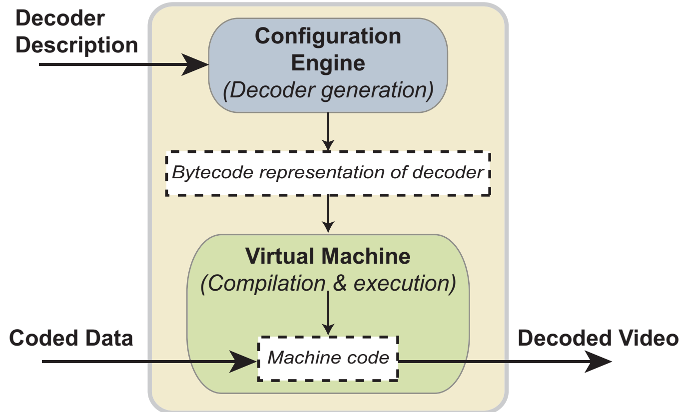

Figure 3: Structure of the Just-in-time Adaptive Decoder Engine.

Virtual Machine (VM). We chose to base Jade on the open source LLVM infrastructure [15]. This VM provides efficient Just-In-Time compilation and multicore execution for a wide range of platforms (X86, X86-64, PPC, ARM, etc.).

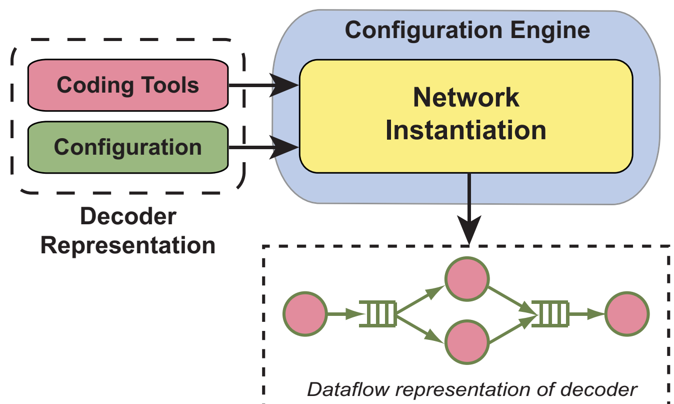

Figure 4: Configuration of an RVC description by the configuration engine.

3.1 On-the-fly reconfiguration of decoders

The configuration engine of Jade contains several mechanisms to switch between different decoder representations during the decoding process. The dataflow representation of the coding tools provided by MPEG RVC gives the ability to incrementally and partially re-program a decoder when receiving new configurations from a bitstream.

The configuration and the reconfiguration of decoder are illustrated in Table 1. Two representations of decoders are standardized by MPEG RVC; the *Simple Profile* (SP) [17, 14] from the MPEG-4 part 2 standards and the *Constrained Baseline Profile* (CBP) [10] of the Advanced Video Coding (AVC)/H.264 decoder from the MPEG-4 part 10. Reconfiguration is done by switching to a proprietary configuration of these decoders. The first configuration is an optimized configuration of the same decoder provided by Ericsson. The second reconfiguration is a configuration developed by the IETR laboratory which represents the *Fidelity Range Extensions* (FRExt) of MPEG-4 AVC. The benchmarks are realized on an Intel E6600 Core2 Duo processor at 2.40 GHz and the entire decoder configurations are available at *http://orc-apps.sourceforge.net*.

Table 1: Configuration and reconfiguration times between implementations of MPEG-4 part 2 *Simple Profile* (SP) and MPEG-4 part10 Advanced Video Coding.

Std.	Configurations	conf.	reconf.
SP	RVC -> Ericsson	1141 ms	380 ms
AVC	CBP -> FRExt	3313 ms	1610 ms

3.2 Clustering and spreading execution on processing units

Jade maximizes the use of the computing resources of any target platform by taking advantage of the inherent parallelism present in an MPEG RVC decoder. The *configuration* of a decoder gives information about the interconnection between coding tools (algorithms) that compose a decoder without carrying any implementation details for a specific platform. Therefore, Jade can execute the adaptive decoder according to the features (e.g. multiple cores) of the underlying platform.

Two optimizations algorithms based on execution models were incorporated in Jade to utilize the concurrency of a decoder configuration, depending on the number of cores in the underlying platforms. The first optimization analyzes a configuration and removes concurrency between tools to find an efficient execution on a same core. The second optimization applies an efficient distribution of independent coding tools onto separate cores. Table 2 shows the ability of Jade for exploiting multi-core CPUs. Each decoder configuration is executed on both cores of an Intel E6600 Core2 Duo processor at 2.40 GHz.

3.3 Easy edition and creation of decoders

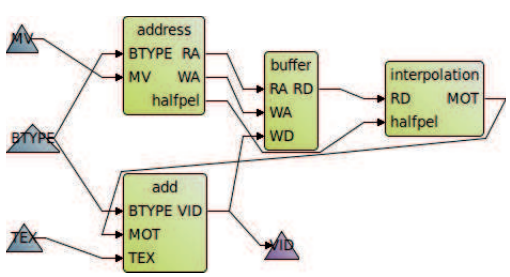

Figure 5: Graphiti edition of the motion compensation for the MPEG-4 part 2 decoder.

Jade is delivered with a complete framework, called Open RVC-CAL Compiler (Orcc), which permits exploring and editing decoder configurations. The configuration of a decoder is described using the XML Dataflow Format (XDF) and can be edited using Graphiti[1], a generic graph editor delivered as an Eclipse plug-in. Orcc also features a complete IDE based on XText for editing the Video Tool Library. It allows developers to add proprietary coding tools to VTLs and to enhance their decoder implementation with modified or completely new versions of decoder configuration.

Figure 5 shows a screenshot of the motion compensation part of the MPEG-4 part 2 *Simple Profile* decoder as a block diagram. The squares represent the coding tools of the configuration and the wires the communications between the coding tools. The triangles represent the inputs and outputs of the given network, allowing the use of a hierarchical description.

Table 2: Decoding performance of the two RVC implementations of MPEG-4 part 2 (SP) and MPEG-4 part 10 (AVC) running on 1 and 2 cores.

Decoder	One core	Two cores	Speedup
SP (RVC)	33 fps	61 fps	1.86
AVC (RVC)	36 fps	79 fps	2.19

4. USING JADE

The binaries and sources code of the first release version (1.0) of Jade can be downloaded at [3]. The source code package is organized in three different directories: the library sources (src), doxygen documentation (doc) and the VTL folder (VTL). Jade necessarily requires a version of the VTL as standardized in MPEG-C to be set in the VTL folder. One can be downloaded at [3] or be generated by the Orcc Framework. The installation documentation of the Orcc framework is available at [6]. The user can also find more open-source applications that are directly useable in Jade such as a JPEG encoder, a GZIP compressor and a cryptographic encoder (CTL) at [4].

The compilation of Jade also requires the use of three other open-source projects, selected with portability in mind. CMake [1] provides the ability of an easy cross-compilation onto multiple operating systems and multiple processor architectures. The SDL libraries [8] permit a portable display and LLVM [5] delivers the necessary libraries for an efficient Just-In-Time compilation for mixed platforms. For Windows users, a library of POSIX thread management [7] is necessary.

Jade provides several interfaces to test its different capabilities. It can be integrated into the Eclipse framework, used from the command line or directly in a player with GPAC [2]. The reader can refer to [6] to find more information about compilation, installation and use of Jade.

4.1 Interfaces of Jade

Jade provides a command line and a simulator interface to demonstrate its configurability. It also provides two other test interfaces : the *scenario* or *console* modes that provides dynamic interaction with the engine.

1. *Command line*: The simplest way to test Jade is to use the command line. The option *–help* gives a full list of all options available for tuning Jade performance depending on the underlying platform. The mandatory options to launch Jade are an XDF network (*-xdf*), an encoded video (-*i*), and the location of your Video Tool Library folder (-*L*).

2. *Console or scenario mode*: These interfaces enable testing the dynamic behavior of Jade. Series of commands can be applied to Jade manager for on-the-fly configuring and reconfiguring of multiple decoders.

[1]Graphiti is available at : http://graphiti-editor.sf.net

The list of commands is explained in detail in a README file provided with the project package.

3. *Orcc simulator*: Jade can be shipped as an eclipse plug-in for helping decoder developers to efficiently test their application. First, the Orcc development and the Graphiti editor must be correctly installed to the eclipse framework. [6] gives instructions to install the complete framework for creating and editing a decoder configuration. Once installed, Jade appears as a new simulator on the Orcc Framework.

4.2 Integration of Jade in a GPAC player

GPAC [2] is currently following ongoing standardization efforts such as HTTP streaming [18] or an MPEG-4 SVC decoder [9]. GPAC is also capable of encapsulating/de-encapsulating RVC decoder description into/from *mp4* video files. Jade is natively included in this project as a GPAC plug-in that allows the player to intantiate on-the-fly an RVC description from an mp4 files and to decode the associated embedded rawstream.

- Encapsulating RVC description: Use MP4Box -add *rawstream.264*:rvc=*config.xml* *rvc_video.mp4*. *rawstream.264* is an encoded bitstream filename, *config.xml* is an XDF network and *rvc_video.mp4* is the resulting mp4 files.

- De-encapsulating RVC description: Use MP4Client *rvc_video.mp4* as command line or Osmo4 as a GUI.

5. CONCLUSIONS

Orcc is a two year old framework now used by industry and research laboratories worldwide. In only one year, Jade has evolved from its experimental origins to become a real proof of concept of the MPEG RVC paradigm. It provides the first Virtual Machine which is able to dynamically instantiate RVC decoder descriptions avoiding inconsistency between a coded bitstream and a decoding platform. An interesting aspect of Jade and MPEG RVC is also its potential usage in the academic and research world. Students, developers and researchers can mix and modify on-the-fly video coding algorithms in a particular decoder to get the best trade-off between coding efficiency and computational cost.

Although the performance is lower than a manually optimized decoder on a single-core system, the property of the MPEG RVC decoder representation is suitable for the next generation of multi-core processor. Moreover, remembering that dataflow representation of decoder provides a considerable potential parallelism, we have already planned to integrate GPU management based on the OpenCL framework to the Jade's Virtual Machine.

6. ACKNOWLEDGMENTS

The authors would like to thank people having contributed directly or indirectly to the development of Jade. First, we cannot avoid offering our deepest respect to Matthieu Wipliez who has led the Orcc project to maturity and his genius for finding easy solutions to complex problems. We also would like to take this opportunity to extend warm thanks to Jean Lefeuvre and his team for their help and to Olivier Labois for his work on the integration of Jade to GPAC.

7. REFERENCES

[1] CMake. `http://www.cmake.org/`.

[2] GPAC. `http://gpac.wp.institut-telecom.fr`.

[3] Jade. `http://sourceforge.net/projects/orcc/files/Jade/`.

[4] Jade applications. `http://orc-apps.sourceforge.net/`.

[5] LLVM. `http://llvm.org`.

[6] Orcc. `http://orcc.sourceforge.net`.

[7] Pthreads-w32. `http://sourceware.org/pthreads-win32/`.

[8] Simple Direct Media Layer. `http://www.libsdl.org/`.

[9] M. Blestel and M. Raulet. Open SVC decoder: a flexible SVC library. In *Proceedings of the international conference on Multimedia*, MM '10, pages 1463–1466, New York, NY, USA, 2010. ACM.

[10] J. Gorin, M. Raulet, Y. L. Cheng, H. Y. Lin, N. Siret, K. Sugimoto, and G. G. Lee. An RVC dataflow description of the AVC Constrained Baseline Profile decoder. In *Image Processing (ICIP), 2009 16th IEEE International Conference on*, pages 753–756, 2010.

[11] J. Gorin, M. Wipliez, J. Piat, M. Raulet, and F. Preteux. A portable Video Tools Library for MPEG Reconfigurable Video Coding using LLVM representation. In *Design and Architectures for Signal and Image Processing (DASIP 2010)*, pages 281–286, 2008.

[12] J. Gorin, M. Wipliez, F. Preteux, and M. Raulet. LLVM-based and scalable MPEG-RVC decoder. *Journal of Real-Time Image Processing*, pages 1–12.

[13] J. Gorin, M. Wipliez, M. Raulet, and F. Preteux. An LLVM-based decoder for MPEG Reconfigurable Video Coding. In *IEEE Workshop on Signal Processing Systems (SiPS 2010), Washington, D.C., USA*, pages 281–286, 2008.

[14] J. W. Janneck, I. Miller, D. Parlour, G. Roquier, M. Wipliez, and M. Raulet. Synthesizing Hardware from Dataflow Programs: An MPEG-4 Simple Profile Decoder Case Study. *Journal of Signal Processing Systems*, 63(2):241–249, May 2011.

[15] C. Lattner and V. Adve. LLVM: A compilation framework for lifelong program analysis & transformation. In *Proceedings of the international symposium on Code generation and optimization: feedback-directed and runtime optimization*, page 75. IEEE Computer Society, 2004.

[16] M. Mattavelli, I. Amer, and M. Raulet. The Reconfigurable Video Coding Standard [Standards in a Nutshell]. *Signal Processing Magazine, IEEE*, 27(3):159 –167, May 2010.

[17] G. Roquier, M. Wipliez, M. Raulet, J. Janneck, I. Miller, and D. Parlour. Automatic software synthesis of dataflow program: An MPEG-4 simple profile decoder case study. In *Signal Processing Systems, 2008. SiPS 2008. IEEE Workshop on*, pages 281–286. IEEE, 2008.

[18] I. Sodagar. Overview of Dynamic Adaptive Streaming over HTTP (DASH). In *ISO/IEC JTC1/SC29/WG11 N11964*, March 2011, Geneva, Switzerland.