Applied Soft Computing 38 (2016) 933–941

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Evolutionary circuit design for fast FPGA-based classification of network application protocols☆

D. Grochol, L. Sekanina*, M. Zadnik, J. Korenek, V. Kosar

Brno University of Technology, Faculty of Information Technology, IT4Innovations Centre of Excellence, Bozetechova 2, 61266 Brno, Czech Republic

ARTICLE INFO

Article history:
Received 10 July 2015
Received in revised form 1 September 2015
Accepted 24 September 2015
Available online 28 October 2015

Keywords:
Application protocol
Classifier
Cartesian genetic programming
Field programmable gate array

ABSTRACT

The evolutionary design can produce fast and efficient implementations of digital circuits. It is shown in this paper how evolved circuits, optimized for the latency and area, can increase the throughput of a manually designed classifier of application protocols. The classifier is intended for high speed networks operating at 100 Gbps. Because a very low latency is the main design constraint, the classifier is constructed as a combinational circuit in a field programmable gate array (FPGA). The classification is performed using the first packet carrying the application payload. The improvements in latency (and area) obtained by Cartesian genetic programming are validated using a professional FPGA design tool. The quality of classification is evaluated by means of real network data. All results are compared with commonly used classifiers based on regular expressions describing application protocols.

1. Introduction

Evolutionary algorithms (EAs) are traditionally used in the circuit design community mainly as efficient optimization techniques. In recent years, significant developments and progress in evolutionary circuit design have been witnessed. In many cases these techniques were capable of delivering efficient circuit designs in terms of an on-chip area minimization (e.g. [1]), adaptation (e.g. [2]), fabrication variability compensation (e.g. [3]), and many other properties (see, for example, many requirements on synthetic benchmark circuits in [4]). In this paper, it is exploited that the evolutionary design can produce fast and efficient circuit implementations. One of the targets is the circuit latency which is a crucial parameter in high performance computing and other applications such as security monitoring of high speed computer networks or high frequency trading. The objective of this work is to minimize the latency and area of key circuits needed in a hardware accelerator intended for classification of application protocols in high speed networks. The classifier is embedded into a software defined monitoring (SDM) platform (see details in Section 2) which is accelerated in a field programmable gate array (FPGA) [5].

In order to identify the application (or the application protocol) the network traffic belongs to, one has to inspect one or several packets with a payload. The main difficulty is that the time to process one packet is less than 7 ns in the case of modern 100 Gbps link. Hence this task has to be performed by specialized hardware. In previous work of the authors [6], key circuit components were developed for an FPGA-based application protocol classifier in which the area and latency were optimized by means of Cartesian genetic programming (CGP). The resulting circuit enabled to classify three application protocols (HTTP, SMTP, SSH) using the first packet carrying the application payload. This circuit, in fact, implemented a deterministic parallel combinational signature matching algorithm in the FPGA.

A more significant latency and area reduction, which will be crucial for classifiers supporting throughputs beyond 100 Gbps, is possible either by using advanced (faster) hardware or changing the packet processing scenario. In this paper, a new approach is proposed with respect to [6] in which small errors in the hardware protocol classification are tolerated assuming that latency and area of the classifier are significantly reduced. This concept is supported by SDM because the traffic unclassified in the hardware can be sent to the software for detailed processing.

Within this scope, the proposed work focuses on a design and optimization of three proprietary circuits, operating as application protocol classifiers, which differ in the quality of classification,

☆ This paper is an extended, improved version of the paper A Fast FPGA-Based Classification of Application Protocols Optimized Using Cartesian GP presented at EvoComNet2015 and published in: Applications of Evolutionary Computing, Proceedings of 18th European Conference, EvoApplications 2015, Copenhagen, Denmark, April 8–10, 2015, LNCS 9028, pp. 67–78, Springer, 2015.

* Corresponding author. Tel.: +420 541141215.
E-mail addresses: igrochol@fit.vutbr.cz (D. Grochol), sekanina@fit.vutbr.cz (L. Sekanina), izadnik@fit.vutbr.cz (M. Zadnik), korenek@fit.vutbr.cz (J. Korenek), ikosar@fit.vutbr.cz (V. Kosar).

http://dx.doi.org/10.1016/j.asoc.2015.09.046
1568-4946/© 2015 Elsevier B.V. All rights reserved.

latency and area. Classifier CL-acc (accuracy) is implemented according to [6] with the goal to minimize the classification error. While classifier CL-cmp (compromise) provides a moderate compromise between the latency, area and classification accuracy, classifier CL-lat (latency) is highly optimized for a low latency. Each classifier is evaluated in the task of classification of four protocols (HTTP, SMTP, SSH, and SIP) we deem most crucial from the perspective of network monitoring. It should be noted that SIP has not been considered in the initial study [6].

The main contribution of this paper is to show that these circuit classifiers can be optimized by CGP in order to significantly reduce their latency and resources requirements. The classification algorithm is not optimized by CGP. The improvements in latency (and area) obtained by CGP are validated using a professional FPGA design tool. The quality of classification is evaluated by means of real network data. All results are compared with commonly used classifiers based on regular expressions describing application protocols. Contrasted to [6], in which only key components of one classifier were implemented and optimized, complete FPGA implementations of three classifiers are evaluated.

The rest of the paper is organized as follows. Section 2 briefly surveys the field of traffic analysis in high speed networks, accelerated network technologies using FPGAs and evolutionary circuit design. Section 3 provides a specification of the classifier and network data used for the evaluation. In Section 4, the proposed hardware classifier and its approximations are introduced. Cartesian genetic programming is presented as a digital circuit design and optimization method in Section 5. Section 6 describes the implementation steps taken and the results in terms of area and latency in the FPGA. Finally, the quality of classification is assessed in terms of precision and recall. Conclusions are given in Section 7.

2. Relevant work

This paper deals with several different research areas – network traffic analysis in high speed networks, FPGA technology, fast pattern matching and evolutionary circuit design. The purpose of this section is to provide an appropriate introduction to them and to their intersections which are relevant for the target application.

2.1. Traffic analysis in high speed networks

An abstract yet detailed network traffic visibility is a key prerequisite to network management, including tasks such as traffic engineering, application performance monitoring and network security monitoring. In recent years the diversity and complexity of network applications and network threats have grown significantly. This trend has rendered monitoring of network and transport layer insufficient and it has become important to extend the visibility into the application layer, primarily to identify the application (or the application protocol) the traffic belongs to. The port numbers are no longer reliable application differentiators due to new emerging applications utilizing ports dynamicaly or to applications evading the firewalls by hiding behind well-known port numbers or utilizing port numbers defined by users [7].

The research in the area of application identification has come up with distinct approaches to identify applications carried in the traffic. These approaches differ in the level of detail that is utilized in the identification method. The most abstract one is behavioral analysis [8,9]. Its idea is to observe only the port number and destination of the connections per each host and then to deduce the application running on the host by its typical connection signature. If more details per connection are available, statistical fingerprinting [10] comes into play. In this case, a feature set is collected per each flow and the assumption is that the values of the feature set vary across applications and hence they leave a unique fingerprint. Behavioral and statistical fingerprinting generally classifies traffic to application classes rather than to particular applications. The reason is that different applications performing the same task exhibit similar behavior. For instance, application protocols such as Oscar (ICQ), MSN, XMPP (Jabber) transport interactive chat communications and hence exhibit a similar behavior, which makes it very hard to differentiate between them. The inability to distinguish applications within the same class is seen as a drawback in some situations when, for example, it is necessary to block a particular application while allowing others in the same class. The approach utilizing the greatest level of detail is a deep packet inspection. It identifies applications based on the packet payload. The payload is matched with known patterns (defined, for example, by regular expressions) derived for each application [11].

The application identification poses several on-going challenges. The identification process is bound to keep pace with ever increasing link speeds (for example, the time to process each packet is less than 7 ns in the case of a 100 Gbps link). Another challenge is represented by the growing number of protocols (i.e., the application identification must address trends such as new emerging mobile applications or applications moving into the network cloud [12]). Some deployments of application identification also require prompt (near real-time) identification to enable implementation of traffic engineering or application blocking [13].

Hardware acceleration (e.g. utilizing an FPGA) is often employed to speed up network processing [14,15], including the application identification directly on the network card. An FPGA renders it possible to utilize various pattern matching algorithms to identify applications. However, pattern matching may exhibit several constraints, that is, the high cost to process wide data inputs (which is the case for high throughput buses in FPGA) and the high complexity and overhead of a pattern matching algorithm which consumes valuable hardware resources or constrains the achievable frequency.

These drawbacks are addressed by alternative methods which look for constants and fixed-length strings (for brevity they are called the *signatures* in the paper) rather than regular expressions (e.g. [16]). This paper builds upon this strategy and envisions a hardware-software codesign approach in which a simple circuit labels the traffic belonging to applications of interest with some probability of false positives while software can subsequently handle and check the labeled traffic with a more complex algorithm effectively. This approach is supported by the software defined monitoring concept [5]. Software defined monitoring employs sophisticated processes running in the software to subsequently install rules in the hardware (network card). While it is not possible (or at a very high cost) to process all traffic in the software, the application identification is offloaded into the hardware. The offload not only reduces the host memory and processor load but it also increases the expressive strength of the SDM rules. The target applications range from application-specific forwarding and traffic shaping to traffic monitoring and blocking.

2.2. FPGAs in network applications

Performance requirements are growing due to the increasing volume and rates of network traffic. Paxson et al. [17] argue that these performance requirements should be met by leveraging a high degree of possible parallelism that is inherent to network traffic monitoring. FPGAs as well as ASICs may deliver such a vast support of parallelism. However, only FPGAs render it possible to prototype and implement critical application components for various network applications at the highest speeds while the optimized ASICs follow after broad deployment a few years later on. FPGAs are thus extensively used in the so-called hardware-accelerated