Microprocessors and Microsystems 000 (2016) 1–17

Contents lists available at ScienceDirect

Microprocessors and Microsystems

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

Co-Processor for evolutionary full decision tree induction

Bogdan Z. Vukobratović*, Rastislav J.R. Struharik

Faculty of Technical Sciences, University of Novi Sad, Trg Dositeja Obradovića 6, Novi Sad, 21000, Serbia

ARTICLE INFO

Article history:
Received 8 June 2015
Revised 4 March 2016
Accepted 23 May 2016
Available online xxx

Keywords:
Data mining
Machine learning
Hardware-software co-design
Decision trees
Evolutionary algorithms
Hardware acceleration
FPGA
Co-processor

ABSTRACT

In this paper a co-processor for the hardware aided decision tree induction using evolutionary approach (EFTIP) is proposed. EFTIP is used for hardware acceleration of the fitness evaluation task since this task is proven in the paper to be the execution time bottleneck. The EFTIP co-processor can significantly improve the execution time of a novel algorithm for the full decision tree induction using evolutionary approach (EFTI) when used to accelerate the fitness evaluation task. The comparison of the HW/SW EFTI implementation with the pure software implementation suggests that the proposed HW/SW architecture offers substantial DT induction time speedups for the selected benchmark datasets from the standard UCI machine learning repository database.

1. Introduction

As a branch of artificial intelligence, machine learning [1,2] comprises a set of procedures/algorithms for the construction of the systems that adapt their behavior to the input data, i.e. by "learning" from the data. An important feature of the machine learning systems is that they can be built with little knowledge of input data and can perform well on previously unseen data instances (generalization property).

In the open literature, a range of machine learning systems have been introduced, including decision trees (DTs) [3,4], support vector machines (SVMs) [5] and artificial neural networks (ANNs) [6]. Data mining is a field where machine learning systems have been widely used [7], among which DTs, ANNs and SVMs are the most popular [3,8,9].

The machine learning systems can be constructed using supervised learning, unsupervised learning or any combination of the two techniques [1,2]. Supervised learning implies using the desired responses to various input data to construct the system, while unsupervised learning implies constructing the system based on the input data only. When the supervised learning is used, the lifetime of a machine learning system usually comprises two phases: the training (induction or learning) and the deployment. During the training phase, a training set is used to build the system. The training set comprises input data and the desired system responses to that data. Once constructed, the system is ready to be used, where new, previously unseen data, will arrive and the system must provide the responses using the knowledge extracted from the training set.

The machine learning systems can perform various tasks, such as classification, regression, clustering, etc. The classification implies categorizing objects given the list of their attributes. Widely used to represent classification models is a DT classifier, which can be depicted in a flowchart-like tree structure. Due to their comprehensible nature, that resembles the human reasoning, DTs have been widely used to represent classification models. Amongst other machine learning algorithms DTs have several advantages, such as the robustness to noise, the ability to deal with redundant or missing attributes, the ability to handle both numerical and categorical data and the facility of understanding the computation process.

This paper focuses on oblique binary classification DTs. The leaves of the DT represent the classes of the problem. The non-leaves contain the tests which are performed on the problem instances in order to determine their path through the DT until they reach DT leaves. Each instance of the problem is defined by its attribute vector - **A**. The tests performed by oblique DT in each node have the following form:

$$\mathbf{a} \cdot \mathbf{A} = \sum_{i=1}^{N_A} a_i \cdot A_i < thr, \quad (1)$$

* Corresponding author.
E-mail addresses: bogdan.vukobratovic@gmail.com (B.Z. Vukobratović), rasti@uns.ac.rs (R.J.R. Struharik).

http://dx.doi.org/10.1016/j.micpro.2016.05.013
0141-9331/

Please cite this article as: B.Z. Vukobratović, R.J.R. Struharik, Co-Processor for evolutionary full decision tree induction, Microprocessors and Microsystems (2016), http://dx.doi.org/10.1016/j.micpro.2016.05.013

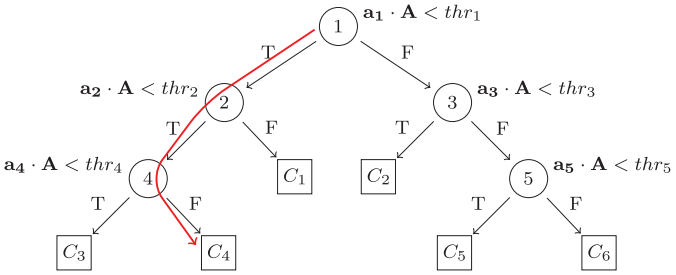

Fig. 1. An example of the oblique binary DT with one possible traversal path shown in red. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

where **a** represents the coefficient vector, N_A equals the size of the attribute and the coefficient vectors and *thr* models the afine part of the test. The Fig. 1 shows an example of the oblique binary DT.

Each instance starts from the DT root node and traverses the DT in order to be assigned a class. If the test condition given by the Eq. (1) is **true**, the DT traversal is continued via the left child, otherwise it is continued via the right child. Depending on the leaf in which the instance finishes after DT traversal, it is classified into the class assigned to that leaf. One possible traversal path is shown in the Fig. 1 in red. After the traversal, the instance will be classified into the class C_4.

There are two general approaches to DT induction: incremental (node-by-node) and full tree induction. Furthermore, the process of finding the optimal oblique DT is a hard algorithmic problem [10], therefore most of the oblique DT induction algorithms use some kind of heuristic for the optimization process, which is often some sort of evolutionary algorithm (EA). The Fig. 2 shows the taxonomy of EAs for the DT induction as presented in [11]. Computationally least demanding approach for the DT induction is a greedy top-down recursive partitioning strategy for the tree growth, hence most of the DT induction algorithms use this approach. Naturally, this approach suffers from the inability of escaping the local optima. Better results, especially if the DT size is considered, could be obtained by the inducers that work on full DT, with cost of the higher computational complexity [12].

The DT induction phase can be very computationally demanding and can last for hours or even days for practical problems. This is certainly true for the full DT inference algorithms. By accelerating this task, the machine learning systems could be trained faster, allowing for shorter design cycles, or could process large amounts of data, which is of particular interest if the DTs are used in the data mining applications [7]. This might also allow the DT learning systems to be rebuilt in real-time, for the applications that require such rapid adapting, such as: machine vision [13,14], bioinformatics [15,16], web mining [17,18], text mining [19,20], etc.

In order to accelerate the DT induction phase, two general approaches can be used. First approach focuses on developing new algorithmic frameworks or new software tools, and is the dominant way of meeting this requirement [21,22]. Second approach focuses on the hardware acceleration of machine learning algorithms, by developing new hardware architectures optimized for accelerating the selected machine learning systems.

Proposed co-processor is used for the acceleration of a new DT induction algorithm, called *EFTI*. *EFTI* (Evolutionary Full Tree Induction) is an algorithm for full oblique classification DT induction using EA. In the remaining of the paper, the proposed co-processor will be called *EFTIP* (Evolutionary Full Tree Induction co-Processor).

The hardware acceleration of the machine learning algorithms receives a significant attention in the scientific community. A wide range of solutions have been suggested in the open literature for various predictive models. The authors are aware of the work that has been done on accelerating SVMs and ANNs, where hardware architectures for the acceleration of both learning and deployment phases have been proposed. The architectures for the hardware acceleration of SVM learning algorithms have been proposed in [23], while the architectures for the acceleration of previously created SVMs have been proposed in [24–27]. The research in the hardware acceleration of ANNs has been particularly intensive. Numerous hardware architectures for the acceleration of already learned ANNs have been proposed [28–30]. Also, a large number of hardware architectures capable of implementing ANN learning algorithms in hardware have been proposed [31–33]. However, in the field of hardware acceleration of the DTs, the majority of the papers focus on the acceleration of already created DTs [34–36]. Hardware acceleration of DT induction phase is scarcely covered. The authors are currently aware of only two papers on the topic of hardware acceleration of the DT induction algorithms [37,38]. However, both of these results focus on accelerating greedy top-down DT induction approaches. In [37] the incremental DT induction algorithm, where EA is used to calculate the optimal coefficient vector one node at a time, is completely accelerated in hardware. In [38] a HW/SW approach was used to accelerate the computationally most demanding part of the well known CART incremental DT induction algorithm.

This paper is concerned with the hardware acceleration of a novel full DT evolutionary induction algorithm, called *EFTI*. *EFTI* is an algorithm for full oblique classification DT induction using EA [efti]. As mentioned earlier, full DT induction algorithms typically build better DTs (smaller and more accurate) when compared with the incremental DT induction algorithms. However, full DT induction algorithms are more computationally demanding, requiring much more time to build a DT. This is one of the reasons why incremental DT induction algorithms are currently dominating the DT field. Developing a hardware accelerator for full DT induction algorithm should significantly decrease the DT inference time, and therefore make it more attractive. As far as the authors are aware, this is the first paper concerned with the hardware acceleration of full DT induction algorithm.

The *EFTI* algorithm was chosen to be accelerated by hardware, since it does not use the population of individuals as most of EA-based DT algorithms do [39–42]. As far as authors are aware, this is the first full DT building algorithm that operates on a single-individual population. This makes the *EFTI* algorithm particularly interesting to be used in embedded applications, where memory and processing resources are tightly constrained. The *EFTI* algorithm proved to provide smaller DTs with similar or better classification accuracy than other well-known DT inference algorithms, both incremental and full DT [43]. Being that the EAs are iterative by nature and extensively perform simple computations on the data, the *EFTI* algorithm should benefit from the hardware acceleration, as would any other DT induction algorithm based on the EAs. This paper proposes *EFTIP* co-processor to accelerate only the most computationally intensive part of the *EFTI* algorithm, leaving the remaining parts of the algorithm in software. In the paper, it is shown that the most critical part of the *EFTI* algorithm is the training set classification step from the fitness evaluation phase. *EFTIP* has been designed to accelerate this step in hardware. Another advantage of this HW/SW co-design approach is that the proposed *EFTIP* co-processor can be used with a wide variety of other EA-based DT induction algorithms [11,39–42] to accelerate the training set classification step that is always present during the fitness evaluation phase.

2. *EFTI* algorithm

This section describes the *EFTI* algorithm for full DT induction based on EA. This algorithm requires only one individual for DT

Please cite this article as: B.Z. Vukobratović, R.J.R. Struharik, Co-Processor for evolutionary full decision tree induction, Microprocessors and Microsystems (2016), http://dx.doi.org/10.1016/j.micpro.2016.05.013