

Real-Time Reconfiguration Approach Based on Efficient Classification and Diagnosis of Embedded Systems

Eurípedes G. O. Nóbrega¹, Denis S. Loubach¹, Ingo Sander², Osamu Saotome³, Ingemar Söderquist⁴
{*egon, dloubach*}@fem.unicamp.br; *ingo@kth.se, ingemar.soderquist@saabgroup.com, osaotome@ita.br*

¹Advanced Computing, Control & Embedded Systems Lab, University of Campinas - UNICAMP,
13083-860, Campinas, SP, Brazil

²Electronics and Embedded Systems Department, KTH Royal Institute of Technology, Stockholm, Sweden

³Electronics Engineering Division, Aeronautics Institute of Technology - ITA, São José dos Campos, SP, Brazil

⁴HMI & Avionics Department, Aeronautics, Saab AB, Linköping, Sweden

Abstract

As fast as digital embedded systems has been evolving, there remains a lot of subjects yet to challenge hard real-time applications, with aircraft and spacecraft avionics as a very important example. Performance and applicability are two fundamental open issues. Fault-tolerant systems are also mandatory, when there are high costs involved, and definitively when human lives are at risk. Considering present technology level, FPGA-based System-on-a-Chip (SoC) is the main choice to develop real-time critical complex embedded systems, with fault-tolerant and reconfigurability capacities. But to face the expected goals, precise diagnosis is essential to drive the reconfiguration efforts in the right directions. An intelligent and global monitoring system is necessary to achieve this objective, presenting the ability to observe and to assess every action, event or communication that relate or occur in or to the system. Clearly, this represents a significant new player to share the resources, however without any unintended effect on the hosted applications or functions. It is expected that methodology and architecture development shall conduct to acceptable overhead levels, in return for the security increase. A Fault Detection and Diagnosis (FDD) system development is here proposed, based on application of a system state estimation and assessment, to be achieved by classification methods. Some classification techniques are briefly described, including Convolution Neural Networks (CNN), to process the significant amount of data generated with this overall monitoring approach, adopting machine learning techniques to be implemented also with FPGA. CNN have been most successfully used in several big data analysis, focusing in pattern recognition and computer vision, but also on signal interpretation like audio, voice and EEG processing. The FDD proposed architecture provides a parallel framework that can achieve partial reconfiguration on real-time, and that shall permit to adopt multiple classifiers implementation, including rule-based, support vector machines and CNN. Preliminary tests permit to expose the architecture and assess the potential of acceleration that may be achieved for the parallel implementation of the classifiers that are going to compose the FDD system.

INTRODUCTION

Digital embedded systems has been attracting renewed attention due to the introduction of the concepts related to the Internet of Things (IoT), but there yet remains important challenges in order to turn most promises into reality. To become the ubiquitous computational presence that IoT proposes, not only the processing hardware has to evolve, which has been occurring, but also some new concepts about distributed computation power are need to extend the embedded system applications and functions. Aircraft and spacecraft avionics are a very important example of embedded systems that may greatly benefit from these new ideas that are going to bring a new wave of computation services. Among these, hardware and software reconfiguration have been intensively researched recently [Loubach, 2016], demanding significant performance increase in order to satisfy real-time applicability of the new functions. Fault-tolerant systems (FTS) concepts have been studied for some time but this area can benefit as well from the new frameworks that are being proposed. FTS are easily associated to reconfiguration, considering its complementarities. Specifically, avionic reconfigurable systems based on condition assessment and fault diagnosis, create a foundation to FTS, which certainly brings significant operational cost reduction and increases human security.

Recent advances in the digital computer area have lately been directed to multicore system development, considering that microelectronics fabrication methods are near the physical limits. The most significant innovation at the huge data centers nowadays is the increasing use of GPU (Graphic Processing Units) to implement parallel processing, which begins to be followed by the use of FPGAs (Field Programmable Gate Arrays). The main common characteristic between these two architectures is the flexibility, meaning the possibility of rapid reconfiguration of how to use the elementary processing cores, to face the changing dynamics of intensive data processing. On the other hand, the main difference is the electric power consuming, with the FPGA presently reaching very low levels. Flexibility and low electric consumption make the FPGA the natural choice for distributed high computational power for embedded systems.

An important subject to require research and development is, therefore, the architectural studies to provide parallelism to hardware implementation of known algorithms, in order to achieve significant performance acceleration when compared with software solutions. Adopting these new architectures for embedded systems can bring over computation power, flexibility, low electrical consume and low overall costs. That seems indeed the main road for embedded systems development in the near future, in order to be able to execute all the new functionalities. Additionally, to convert known algorithms to parallelism, considering hardware concurrency and depending on the respective time and memory constraints, can lead to new algorithms, which may also need adaptation to the available resources in the specific adopted FPGA. It ends up to how to develop and implement new parallel fast algorithms to get powerful computation in small systems with low electrical consumption [Farabet et al., 2009].

As already stated, one possible approach to FTS is based on reconfiguration, implying two different frameworks to be adopted. The first framework is for designing fault-tolerant controllers (FTC), where the traditional sensor-based feedback loop provides the set-point error $e(t)$ that is processed by the controller to apply the control signal $u(t)$ to the actuator. Figure 1 shows an FTC block diagram including the new blocks to provide the fault-tolerance, respectively the fault detection and diagnosis (FDD) and the reconfiguration mechanism module (RMM). FDD output signal vector $f(t)$ must characterize the fault in the system, based on performance measurements and estimations, which is used to reconfigure the controller parameters by the RMM, according to some specific law. Often the sensors and actuators have analog signals, and its faults are then diagnosed in order to reconfigure the controller. Faults in the plant components or structure can be also monitored and assessed in this configuration.

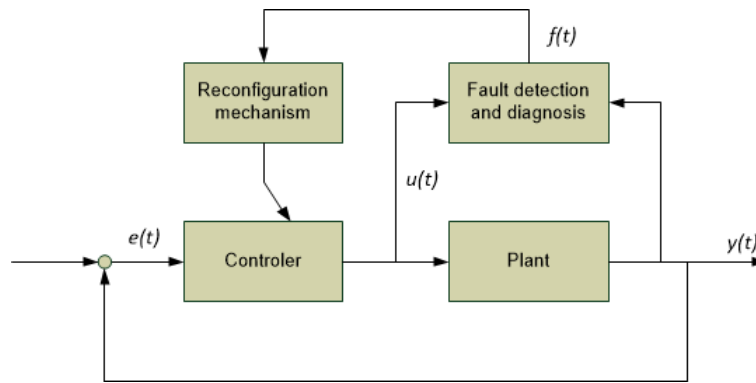


Figure 1: Fault-tolerant controller reconfiguration block diagram

However, the above presented approach does not take into account the possibility of faults in the controller itself, i.e. faults in the digital computation system that are in charge of the controller algorithm. The second framework is mainly concerned with digital system faults, which may be caused by several circumstances, including aging, radiation exposure, vibration, short or open circuits, and so on. There are diverse low-level faults and failures, which require some techniques to detect and diagnose. However, considering a complex embedded system supporting several functions and services, involving multiples digital only subsystems and also digital controllers processing analog signals, an intelligent and global monitoring system is necessary, mixing the two frameworks, presenting the ability to observe and to assess every action, event or communication that may occur in or to the system. This constitutes what is today called a big data problem about a system of systems, which points to a sophisticated classification solution.

Considering an actual avionic system, this monitoring system represents a new important global function to share the resources, however the respective architecture development shall conduct to acceptable overhead levels in return for the security increase. This means that high efficiency and significant results are essential for this FTS to justify the concession of the respective embedded computational space. A key point to consider is how to analyze this big data problem in order to respond to these requirements. A possible architecture solution is to adopt a distributed intelligent system, reproducing the main avionic structure, where each subsystem incorporates a local signal analysis to assess the state of each submodule. Therefore, for each substructure of the system there is a correspondent structure of the monitoring and diagnosis system, configuring an equivalent structural tree where the top is the high level module that maintains the dialog with the main system. The objective of the FTS subsystem is to keep the whole system working in the better possible way, including itself.

In a FDD system, the classifier to detect repeating patterns in data usually represents the most computational demanding part. There are several classification methods that may be used to continuously detect, interpret and assess the analog and digital signals. These methods have been successfully used to provide solutions to several problems in the last two decades, but the focus is today on machine learning techniques considering 2D applications like the computer vision area, or 1D applications like natural language audio processing [LeCun et al., 2015].

The first action from a FDD system is to ensure that there is no fault crossing the first alarm threshold, considering all the signals that are under scrutiny. However, when some fault is detected, a diagnostic report must be emitted with sufficient information to permit the central controller to decide exactly what to do in order to circumvent the problem and, when applicable, to delimit the reconfigu-

ration that shall be implemented. Formal SoS, Model of Computation (MoC) [Jantsch, 2005, Jantsch, 2003, Jantsch and Sander, 2005, Lee and Sangiovanni-Vincentelli, 1998] and languages are the adopted way to provide the structural knowledge to decide the extension of the needed reconfiguration. However, this subject demands a lot of space to be approached, and is not treated here.

This paper proposes an architecture for an FDD high-level system to include and monitor the whole avionic system, incorporating hardware and software modules based on a FPGA implementation. Taking this FDD system as a basis, a fault-tolerant avionics can implement regular or critical reconfiguration or complete module regeneration, depending on the gravity of the assessment of the state of each subsystem. A classification system is the core of the proposed diagnosis system.

CLASSIFICATION METHODS

Brief descriptions of some classifiers are presented in this section due to its importance considering big data analysis. Currently, classifiers are seen as an application of machine learning methods, which corresponds to an intelligent algorithm that automatically learns how to discriminate the several classes of a given problem through training with examples. Training results from exposure to a set of data samples, used to generate indicators that separate groups with data similarities, which constitute the classes. The training set may be already classified, with the respective class of each sample previously known, which is commonly called the sample label. A supervised learning represents the cases when the known sample labels are used during the training. Unsupervised learning represents the techniques that do not need a labeled training set. In this case, the algorithm has to devise interesting patterns, in a prescribed sense, that are present in the samples. It is possible also to have a problem that mixes the two training methods. Classification is a two steps process, comprising first the training and, after that the attribution of a class to a specific unknown sample. However, incrementing learning is frequently necessary, when the classifier needs to include new classes after already trained [Wen and Lu, 2007].

Simple statistical class discrimination is the techniques that uses Bayes rule to find the probability of a given sample to belong to each predefined class. The idea is basically to consider hyperspace clusters (whose coordinates are known as features) that concentrate groups of data points with similarities constituting the classes, and to find a surface that separates these clusters. Then a simple distance measure of each sample point in this hyperspace to the density center of each cluster can classify an unlabeled sample. This approach is simple and effective, but only for cases where there are good known statistical properties for the classes and requires statistical independency between the features, two common but hard to find properties. Moreover, the values of all features need to be known to achieve a classification. A linear classifier is one that uses a linear function describing the separating surfaces. It is called nonlinear classifiers otherwise. The concept of discriminant surfaces between the classes is common among the classifiers in general. To face the statistical difficulties, several classifiers have been proposed, with different levels of success depending on the application [Britto Jr et al., 2014]. Some of them are described in the following.

Rule-based Method

Rule-based expert systems adopt knowledge-based rules of the type “*if* antecedents *are* true *then* consequents *are also* true”, subject to an algorithm that provokes inference through a sequence of a dynamic chain of rules. Antecedents are a logical expression of several variables that may be binary-valued or obey to some fuzzy pertinence function. Antecedents are the actions to be taken in case of a

true assessment of the antecedents, or some logical elements to be declared true in an inference procedure [Soe and Zaw, 2008]. These methods originated from the pioneer work of [Buchanan and Shortli, 1984] and evolved along the decades in the sense of how to acquire efficiently a reliable knowledge base to generate the rules. The first approach was to get the rules based on human expertise, but despite good results, this revealed to be of long gestation with slowly growing of the knowledge. A statistical approach to examine a given training set to infer the rules using what is known as Bayesian networks [A. Oniésko, 2001] was an obvious step in the development of rule-based systems (RBS). However, these techniques present all the requirements that make it difficult to get general results, as already mentioned about the classifiers. Additionally, today availability of immense databases requires efficient methods that can extract correlations and significance directly from the data, which involves *data mining* techniques [Mutter et al., 2004, Zaiane et al., 2002]. This conducted to the concept of association rule mining (ARM), where the goal is to find groups of data that present interesting relationships in a database. However, there are clearly very high computational costs involved in this associative rule finding, even for moderate size databases, which required new specific methods to solve the problem [Mutter et al., 2004]. The proposal of the Apriori algorithm [Agrawal and R., 1994] about two decades ago opened the research that conducted to several improvements and several associative rule-base diagnostic systems have been published since. However, the size of databases and number of features can grow very fast, which conducted to parallel techniques [B. Liu and Ma, 2000].

Support Vector Machine Classifier

Support Vector Machine (SVM) techniques are used to classify data that require linear discriminant functions. It became known two decades ago due to the work of [Cortes and Vapnik, 1995] and is probably the most successful statistical classifier with many different applications [Lotte et al., 2007], including machine vision for autonomous vehicle.

Considering a linear surface separating two classes, the support vectors are those samples that define the minimal respective distance, providing the bigger margin for the classification. The support vectors are formed by the samples that define the frontier of each class region. This way, the two support vectors between two classes define a hypersurface with an optimal separator exactly in the middle of those two surfaces. If there is not such a linear hypersurface, a nonlinear transformation, called kernel, must be found such that in the new hyperspace a linear discriminant may be found. Because this transformation function is difficult to find, some kernel functions are commonly used, such as hyperbolic tangent, sigmoid or radial basis functions. Training of SVM classifiers is to find the hyperplane that maximizes the classification margin, solved as an optimization problem [JAKKULA, 2006]. Several different applications were successfully developed [Sun and Huang, 2011, Moguerza and Muñoz, 2005], including mixing with deep learning and SVM [Tang, 2013].

Convolutional Neural Networks

Perceptron neural networks have been used for classification for more than three decades now, using mostly the basic configuration of an input data layer, a nonlinear hidden layer and an output linear layer. A hidden layer is called one that is not the input or the output layer. This network is trained using the well known backpropagation algorithm, and corresponds to an universal function approximator, which can easily find solutions to discriminate the classes [Kotsiantis et al., 2006]. It can deal well with signal time complexities and also ignore irrelevant features in the training. However, instead of treating raw data to separate similar objects, this kind of classifier needs to use specific

filters to extract from the data significant features to constitute the inputs of the network, which requires domain expertise and mathematical tools to pre-process the data. These neural networks are now referred as "shallow", in contrast of the "deep" networks, which comprises several hidden layers. Deep neural networks (DNN) present the very important advantage that these features can be directly extracted from raw data, and learned automatically from a general-purpose training method [LeCun et al., 2015]. Speech recognition was one among the initial successful applications presenting outstanding results of DNNs [Dahl et al., 2012].

Convolution Neural Networks (CNN) are a specific type of DNN that fits well to applications where data are in the form of multiple arrays, as 2D images or 1D signals. CNNs have been successfully used in several big data analysis, focusing in pattern recognition and computer vision, but also on signal interpretation like audio, voice and EEG processing. CNNs naturally explore, through the general adopted architecture of multiple layers divided in a convolution layer and a pooling layer, the compositional hierarchies present in signals, e.g. voice decomposition in phonemes, syllables, words and sentences. Image processing adopting FPGAs are also among the most significant results of CNNs [Farabet et al., 2010].

FPGA PERFORMANCE AND PARALLEL ARCHITECTURE

This decade has seen a new generation of System-on-a-Chip (SoC) FPGAs, where there is a regular microprocessor and a pure FPGA embedded in the same encapsulation, connected through a high velocity bus. Several development boards are now available that encompass memory chips and other convenient peripherals. Together with the recent release of OpenCL compilers, parallel implementation of algorithms will become a more common task. OpenCL is a framework that provides a common platform to develop parallel application for multiple CPUs, GPUs and FPGAs [Hill et al., 2015]. Specifically for FPGAs, it will make it easier to deal with several hardware intricacies that presently requires domain knowledge.

Figure 2 presents the proposed architecture that are capable to implement multicore varied configurations, which comes in handy to solve multiple algorithms partial reconfiguration, considering a common parameterized structure that conforms to the application. The host computer (HP) is the on-chip processor that commands all the FPGA processing, communicating with the parallel processor (PP) through the high velocity bus, transferring all data to be processed from the input RAM (iRAM) and receiving the processed data to store in the output RAM (oRAM). The HP also indicates the respective instructions that the PP shall apply to the data, which may include the partial reconfigurability or even complete regeneration. The PP includes an Arbitrator (PPA) which controls the internal data flow to allocate the subarrays to the local memories and also controls the instruction execution. These functions are executed through the Elementary Array Processor (EAP) which counts with a given parameterized number of elementary processors and respective local work memory, shown only in Figure 3.

Figure 3 presents some details of the PP modules, the interconnections between the modules, and the elementary processor (EP) and respective local memory (LRAM). The global RAM, which is composed of the iRAM and the oRAM, directly accesses the local EP memory (LRAM), according to signals generated as described in the above paragraph. PPA is the module that generates the control signals according to the specified distribution of parallel tasks between the several EPs. The proposed configuration is a typical Single Instruction Multiple Data (SIMD), where all EPs executes the same instruction over its particular piece of data copied into its LRAM, that is a subarray of the complete data array originally installed in the iRAM, and conveniently distributed by the PPA. Notice that,

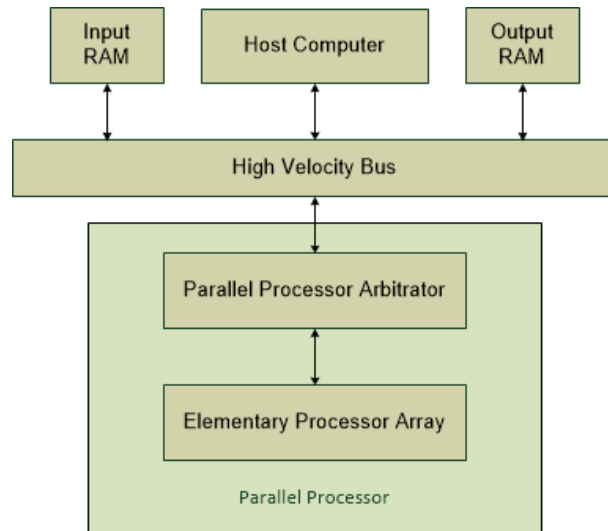


Figure 2: Proposed parallel architecture

for the sake of performance, and due to the importance of memory provoked latencies for parallel algorithms in general, the local RAM can also be separated in input and output RAM.

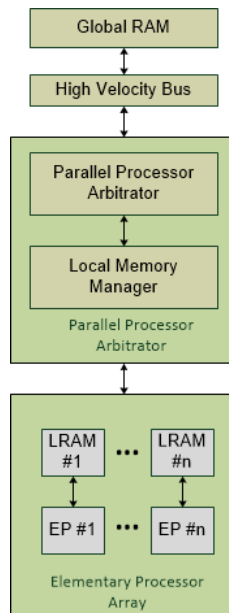


Figure 3: Architecture details

To conclude the architecture description, the EPs represent the core of the parallel formulation, being the hardwired version of the iteration of the software version of the algorithm that is being parallelized. For example, for an SVM classifier implementation, it may be the kernel formula implemented in a specific designed circuit, or, in a CNN, it may represent the convolution expression itself. Notice that for each application, each EP may be reconfigured according to the progress of the data processing, considering the most efficient circuit for each case. In the same way, the training phase may occur first to determine the respective parameters, and then in the classification phase the EPs

are reconfigured to implement the classification. This is a key advantage to consider, taking into account that incremental learning is indeed a common requirement for real applications, and an efficient training algorithm is also necessary. Finally, the configuration to be adopted must take into account the reconfiguration times, considering that hard real-time applications are the target here.

CONCLUSION

The FDD proposed architecture is under development to implement the diverse classification methods here presented. Some preliminary results show that, comparing the host implemented software version of the algorithm with the hardware version, acceleration of at least 10 times may be expected, and eventually much more with the progress of the work. The configuration was tested for a computer vision application in pedestrian detection on photos. An SoC board was used for this test with a dual-core A9-Cortex ARM processor and a Cyclone V Altera FPGA.

REFERENCES

- [A. Oniésko, 2001] A. Oniésko, P. Lucas, M. D. (2001). *Comparison of rule-based and Bayesian network approaches in medical diagnostic systems*. Artif Intell Med.
- [Agrawal and R., 1994] Agrawal, R. and R., S. (1994). *Fast algorithms for mining association rules*. VLDB.
- [B. Liu and Ma, 2000] B. Liu, Y. and Ma, C. W. (2000). Improving an association rule based classifier. In *4th European Conference on Principles of Data Mining and Knowledge Discovery*.
- [Britto Jr et al., 2014] Britto Jr, A., Sabourin, R., and Oliveira, L. (2014). Dynamic selection of classifiers-a comprehensive review. *Pattern Recognit.*
- [Buchanan and Shortli, 1984] Buchanan, B. and Shortli, E., editors (1984). *Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project*. Addison-Wesley, Reading, MA.
- [Cortes and Vapnik, 1995] Cortes, C. and Vapnik, V. (1995). *Support-vector networks, Machine Learning*.
- [Dahl et al., 2012] Dahl, G. E., Yu, D., Deng, L., and Acero, A. (2012). Context-dependent pre-trained deep neural networks for large vocabulary speech recognition. *IEEE Trans. Audio Speech Lang. Process.*
- [Farabet et al., 2010] Farabet, C., Martini, B., Akselrod, P., Talay, S., LeCun, Y., and Culurciello, E. (2010). Hardware accelerated convolutional neural networks for synthetic vision systems. In *IEEE Int. Symp. Circuits Syst.*
- [Farabet et al., 2009] Farabet, C., Poulet, C., Han, J. Y., and LeCun, Y. (2009). Cnp: An fpga-based processor for convolutional networks. In *FPL2009*.
- [Hill et al., 2015] Hill, K., Craciun, S., George, A., and Lam, H. (2015). Comparative analysis of opencl vs. hdl with image-processing kernels on stratix-v fpga. In *IEEE 26th Int. Conf. Appl. Syst. Archit. Process.*
- [JAKKULA, 2006] JAKKULA, V. (2006). Tutorial on support vector machine (svm). Technical report, School of EECS, Washington State University.
- [Jantsch, 2003] Jantsch, A. (2003). *Modeling Embedded Systems and SoCs: Concurrency and Time in Models of Computation*. Morgan Kaufmann, 1st edition.
- [Jantsch, 2005] Jantsch, A. (2005). *Embedded Systems Handbook*, chapter Models of Embedded Computation. CRC Press.
- [Jantsch and Sander, 2005] Jantsch, A. and Sander, I. (2005). Models of computation and languages for embedded system design. *Computers and Digital Techniques, IEE Proceedings -*, 152(2):114–129.
- [Kotsiantis et al., 2006] Kotsiantis, S., Zaharakis, I., and Pintelas, P. (2006). Machine learning: a review of classification and combining techniques. *Artificial Intelligence Review*.

- [LeCun et al., 2015] LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*.
- [Lee and Sangiovanni-Vincentelli, 1998] Lee, E. and Sangiovanni-Vincentelli, A. (1998). A framework for comparing models of computation. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 17(12):1217–1229.
- [Lotte et al., 2007] Lotte, F., Congedo, M., Lecuyer, A., Lamarche, F., and Arnaldi, B. (2007). A review of classification algorithms for eeg-based brain computer interfaces. *Journal of Neural Engineering*.
- [Loubach, 2016] Loubach, D. S. (2016). A runtime reconfiguration design targeting avionics systems. In *2016 IEEE/AIAA 35th Digital Avionics Systems Conference (DASC)*, Sacramento, USA.
- [Moguerza and Muñoz, 2005] Moguerza, J. and Muñoz, A. (2005). *Support vector machines with applications*. Statistical Science.
- [Mutter et al., 2004] Mutter, S., Hall, M., and Frank, E. (2004). *Using classification to evaluate the output of confidence-based association rule mining*. Springer.
- [Soe and Zaw, 2008] Soe, S. M. M. and Zaw, M. P. P. (2008). *Design and Implementation of Rule-based Expert System for Fault Management*, World Academy of Science, Engineering and Technology.
- [Sun and Huang, 2011] Sun, H.-C. and Huang, Y.-C. (2011). Support vector machine for vibration fault classification of steam turbine-generator sets. In *Proc. Eng.*
- [Tang, 2013] Tang, Y. (2013). Deep learning using linear support vector machines. In *International Conference on Machine Learning*.
- [Wen and Lu, 2007] Wen, Y.-M. and Lu, B.-L. (2007). Incremental learning of support vector machines by classifier combining. In *11th Pacific-Asia Conference on Knowledge Discovery and Data Mining*.
- [Zaiane et al., 2002] Zaiane, O. R., Antonie, M.-L., and Coman, A. (2002). Mammography classification by an association rule-based classifier. In *International Workshop on Multimedia Data Mining*.