FEATURE EXTRACTION AND IMAGE RETRIEVAL ON AN AUTOMATA STRUCTURE

Tiffany Ly, Rituparna Sarkar, Kevin Skadron, Fellow, IEEE and Scott T. Acton, Fellow, IEEE

C.L. Brown Department of Electrical & Computer Engneering, University of Virginia
Charlottesville, Virginia USA

ABSTRACT

An automata processor can execute pattern matching in parallel which brings potential to accelerate image recognition. In this paper, we present a novel process of implementing image retrieval using multinary representation for use on an automata framework. Images are encoded into discriminative and unique regular expression descriptors in such a way that can be used for classification purposes. The regular expression descriptors are streamed through sets of non-deterministic finite automata (NFA). Results show that image retrieval can be implemented on automata structures that can achieve dramatic improvement over general purpose processors or graphics processors in efficiency.

Index Terms**—** non-finite automata, image retrieval, pattern matching

I. INTRODUCTION

Image retrieval is still a challenging problem in the field of image processing. One of the major challenges of large scale image retrieval is that it requires searching through large databases where images specific to an object category may have significant content variations. To capture the intra-category variation while make the discrimination between inter-category more prominent, computing discriminative image feature descriptors is a crucial step. Additionally it requires a good classifier or similarity measure to classify images or compare pairs of images for retrieval. Both classifier design and similarity based image search for large scale retrieval are computationally expensive and consequently, online image retrieval applications require considerable parallelism. The Automata Processor (AP) is a new hardware accelerator that can perform highly complex pattern matching applications [1]. It executes parallel processing of thousands of non-deterministic *finite automata state machines* that represent different regular expression patterns. This is ideal for pattern matching applications with large datasets, including Brill tagging, bioinformatics, and machine learning [2]–[4]. For example, the DNA motif search problem [3] found speedups with AP structure when compared to conventional methods. This automata framework has the potential to provide acceleration on many image retrieval applications, particularly those with massive datasets that require parallelism. In this paper, we establish a method for image classification that would be able to take advantage of the AP's acceleration.

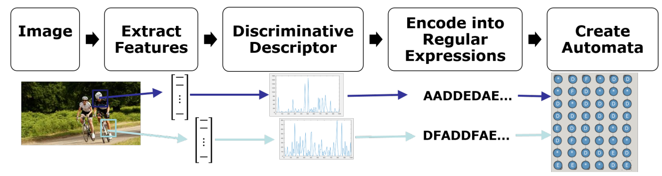

Fig. 1. Overview of implementing image retrieval on automata.

A number of existing methods in the literature develop discriminative image features to capture the intra as well as inter-class variations. These works exploit various image characteristics, such as color, texture, and object shape, to compute the global image descriptors. The image feature descriptors are integrated with a similarity or dissimilarity measure to perform image classification. Other works obtain features using local image information. Some of the more popular works using SIFT [5], analyze local regions based on object keypoints to obtain local image features, while histogram of oriented gradients [6] uses local image gradients to obtain the feature descriptors. Some methods aim to increase robustness by aggregating the local features into histograms [7]. Methods such as the spatial pyramid [8] achieve robustness by incorporating spatial correspondence of the local features to compute the final image feature descriptor.

Dimensionality reduction techniques aim to keep significant information such as applying principle component analysis [9], Fisher vectors, Gaussian mixture model [10] etc. However, the descriptors are still significantly large and need appropriate similarity measures to compare features of images or learn a classifier using class labels associated with the images.

This paper expands feature descriptor to multinary representation and present a matching method for classifying images on an automata framework. Our method allows for implementation of image retrieval on the new Automata Processor (AP). The AP significantly reduces the time it takes to search large image datasets for retrieval by taking advantage of the AP's massive parallelism. We propose a method in which the feature descriptors are quantized and encoded to regular expressions and finally incorporated as state machine in the AP. We compare the efficiency of this classification method with current state-of-the-art image retrieval methods on the CPU [11], and show how our multinary representation method could be exploited to acquire faster and better image retrieval accuracy.

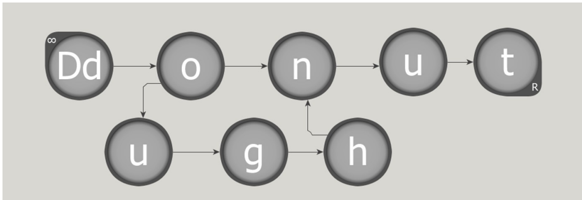

Fig. 2. Example on automaton: Finding the word [Dd](o|ough)nut from input text.

II. AUTOMATA PROCESSOR

The Automata Processor (AP) is a scalable hardware accelerator that can execute thousands of non-deterministic finite automata in parallel by finding regular expression patterns in an input data stream. Users can program automata structures and load them onto the hardware.

Automata on the processor are made up of state transition elements (STE) that are programmed to match on individual symbols or arbitrary character classes. In addition, the AP board also contains Boolean elements, such as AND, NAND, OR and NOR, and counter elements. The counters can activate an STE or report when they reach the threshold specified by the user. These elements are connected through a programmable routing network. As STEs match, they activate successor nodes. The final STE of an automaton is a reporting STE which reports when the state machine reaches an accept state. The first-generation AP is a PCI-Ecpress accelerator with an offload model mediated by a driver, but the board can also be programmed to raise interrupts when report bits are set. An chip holds 49K STEs, and the first-generation boards have 32 chips, so an AP board holds up to 1536K STE. It is important to note that the input stream is fanned out to all active APs across the board. In other words, all active STEs inspect a new input symbol every cycle. This means that thousands of state machines can be executed in parallel, where the input character gets sent through every clock cycle [1].

Figure 2 shows a simple example of an automaton on the AP. The first STE in the machine is matched when a "D" or "d" is put through the input data stream. Once an STE is matched it activates the next STE. This automaton reports if either spelling of doughnut is found.

III. IMAGE SIMILARITY USING MULTINARY REPRESENTATION

The AP has potential speed up on image retrieval applications with massive datasets by performing a massive number of image matching operations in parallel. To take advantage of this framework, images/image features need to be encoded into regular expressions or loaded onto the AP of its other programming modalities [12], [13]. These strings of characters which represent an image or a particular category of images are ultimately encoded as state machines on the automata. Test images are streamed as input and classified to an image category. Instead of using a similarity measure to classify feature descriptors, this image retrieval method may require exact matching of state machines. However, in Section IV, we propose a method to relax the need for exact matching, but in both scenarios the image descriptors must be highly distinctive. Figure 1 shows the overview of our method of implementing image retrieval on automata structures. The two crucial steps for image retrieval on automata are to extract discriminative features and mapping the features as regular expressions. Once different image categories have unique regular expression patterns, they can be represented as automata.

III-A. Feature Extraction

In the literature, both global and local image features have been employed for the task of image classification. Features extracted from local image regions are often aggregated in some manner to obtain a global feature [7], [8]. The features extracted form the images can be used for both supervised and unsupervised image classification. For unsupervised image classification, along with extracting discriminative feature descriptor, a robust similarity measure is also necessary. Based on the type of feature being used, or the type of feature encoding used, different similarity measures may be preferred. For example in [8], a histogram matching kernel was implemented on spatial pyramid features. Whereas in [14], sparse codes are compared using a compression-based similarity measure. For supervised classification, a robust classifier needs to be designed, which again demands modeling of proper classifier functions. Implementing the retrieval on AP can be an advantage in this context. Since the AP only allows for matching based on regular expressions, once the feature descriptors of the images are extracted, a robust mapping of the features to regular expressions is the only requisite, in contrast to designing case-specific similarity measures or classifiers.

III-B. Encoding Feature Descriptors to Regular Expressions

Once a discriminative feature descriptor is attained for every image, the descriptors are encoded into regular expressions before compiling them into state machines. Since the automata does not take in floating point numbers, the descriptor values are binned to 8-bit characters using this equation. The range of the bins are divided equally and correspond to the minimum and maximum feature descriptor ranges. The regular expression descriptor is derived by

$$R_i = char(y_i * (\beta/k)) \tag{1}$$

where i is the length of the descriptor, y is the image descriptor, k is the range of descriptor values for that method, and β is the number of characters used.

IV. IMAGE RETRIEVAL ON THE AP

Automata are created for each category within a dataset from the regular expression descriptor of the training images, and these are loaded onto the AP board. The regular expression descriptors of the test images are then computed as a preprocessing step, and then sent as an input data stream to be matched in parallel with each candidate descriptor on the AP board. The test image is classified to the category with the most automata matches. Since the AP

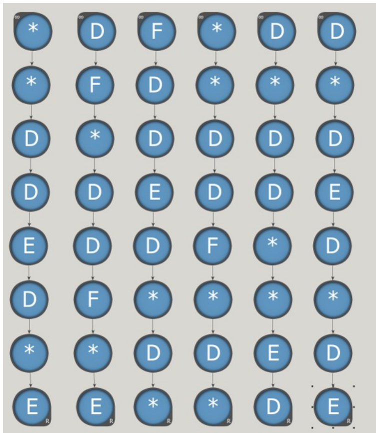

Fig. 3. An example of automata with regular expression patterns with threshold.

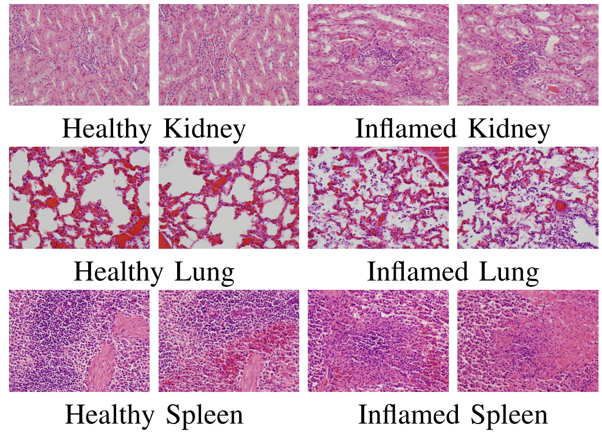

Fig. 4. Samples of three ADL organ tissue datasets: (a) kidney (b) lung and (c) spleen.

requires exact matching, a threshold can be applied on the regular expression patterns on the automata to allow for more lenient matching. Each STE can allow for up to 8-bit characters. In the AP, an STE with * symbol means that any character will match with the activated STE. Here, we set a threshold by comparing the characters at every index of the regular expression descriptor of the training dataset. If there is greater than a percentage of mismatches at that index then the symbol is hard-coded as * symbol, as shown in Figure 3. The threshold gives more flexibility in exact pattern matching without losing significant information. In the figure, each automata represents a pattern of an image in the training dataset that is to be matched by the input data stream.

V. EXPERIMENTS

We evaluate our results with two datasets: ADL dataset [15], and the Vehicle dataset using two different feature extraction and representation methods. We compare our results with SHIRC [15], and SLIDE [14], respectively.

V-A. Evaluation for ADL datasets

The ADL database contains kidney, lung and spleen tissue datasets with healthy and inflamed tissues. Each dataset contained about 330 images. Figure 4 shows the major challenge for each dataset is the slight dissimilarity between healthy and inflamed tissues. For each organ, 115 images per class are used for training and 40 images per class are used for testing.

For this dataset, we use a Gabor bag-of-words method to represent the feature descriptor. 32 Gabor features were extracted from 1910 generated superpixels of every tissue image. A codebook was trained via k-means clustering with k=500 centers to then attain a histogram descriptor for every image, size $\mathbb{R}^{1\times500}$. The regular expression descriptors of all the training images are concatenated. If there are X images in the training dataset, then there would be X automata of length 500. The descriptor values for the first dataset range from [0,1]. These descriptors have an unknown non-linear distribution where most of the values lie close to 0 and a few values very large arbitrary values. This is a challenge when matching with the automata because it requires exact pattern matching, rather than using a similarity measure between the descriptors. To aid this, a threshold is set for the allowable mismatches at each index of the descriptor using 250 randomly chosen training images. Figure 5 shows the effect of the threshold for allowable mismatches on classification accuracy with T=10%, 20%, and 30%. For every dataset, a threshold of T=10% reported the best accuracy.

Table I shows a confusion matrix using our method at a 10% threshold in comparison with the simultaneous sparsity model for histopathological image representation and classification (SHIRC), which is a sparsity model that learns a dictionary for RGB color channels [15]. There are three confusion matrices for each of the organ datasets. The class label on that row represents the class the test image actually belongs to, and the column is the class the method classified it to. The bold numbers are the true positives attained using our retrieval method. The method deployed on the automata framework does not do better in most cases. However, the lower performance was largely due to unclassified images since our method uses exact pattern matching.

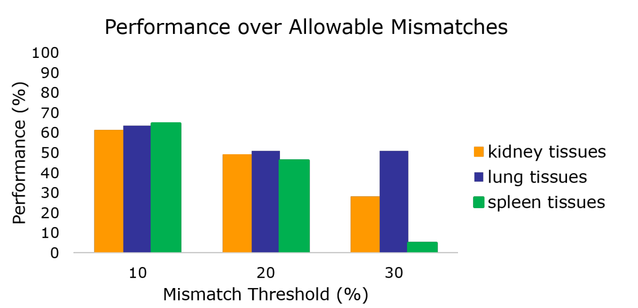

Fig. 5. Performance for the kidney, spleen, and lung datasets when the threshold for allowable mismatches is set to T= 10, 20 and 30 %.

Table I. Confusion Matrix for ADL Dataset

		Kidney		Lung		Spleen	
	Class	Healthy	Inflamed	Healthy	Inflamed	Healthy	Inflamed
Gabor w/ Superpixels	Healthy	**61.3**	38.7	**63.6**	36.4	**64.6**	35.4
	Inflamed	32.5	**67.5**	47.3	**52.6**	24.4	**75.6**
SHIRC [15]	Healthy	92.0	8.0	91.0	9.0	90.8	9.2
	Inflamed	16.3	83.7	28.6	71.4	30.6	69.4

V-B. Evaluation for Vehicle dataset

The vehicle dataset contain four categories-airplane, car, motorbike, and ships with 70, 50, 70, and 36 images, respectively. The images were obtained partly from Google, Caltech-101 dataset [16] and the Inria GRAZ02 dataset [17]. The sample images are shoen in Figure 6, There are a few challenges with this dataset. Within each category, the objects are variable in rotation, color, size and type. The car images have varying frequency of objects in a single image. Some images also have low contrast between foreground and background which makes it difficult to extract objects features. Superpixels were generated on every image to mitigate this challenge by grouping the image into segments based on spatial and color information. For each image, a dictionary was learned from the HOG descriptors using the following minimization.

$$X = \underset{D,x}{\arg\min} \|Y - DX\|_2^2 \ \text{ s.t.} \forall i,\ \lambda\|x\|_0 \leq e \tag{2}$$

The HOG descriptor represents the input signal, $Y \in \mathbb{R}^{60\times1200}$, which is represented as linear combination of dictionary, $D \in \mathbb{R}^{60\times500}$, while minimizing the reconstruction error. X are the sparse codes for the signal Y, and e is the sparsity constraint [18]. The values of the atoms range from [-1,1] and are encoded into regular expression patterns without a threshold. Each dictionary atom in the training dataset are represented as automata on the AP. The dictionary atoms of the test images are sent through the input data stream to be matched. Thus the learned dictionary atoms are used as image features for this dataste. A test image is classified to a category for which maximum number of atom matches to that category's automata.

The retrieval-accuracy results using an automata framework for both datasets are given in Table II and are compared with state-of-the-art methods implemented on similar datasets. Image retrieval on an automata framework performed significantly better when using dictionary atoms as descriptors than when using Gabor bag of words descriptors. Retrieval using regular expressions still did slightly worse than the SLIDE method for the vehicle dataset.

Table II. Overall retrieval performance

Method	Acc. (%)	Runtime (s)
*Gabor w/ Superpixels*on ADL dataset using AP	63.2	**2.967e-5**
SHIRC [15] on ADL dataset	80.5	.55
HOG w/ Superpixels on Vehicle dataset using AP	79.5	**.06**

V-C. Run-time Comparison

The computational cost of image classification on an automata framework was computed using a run-time estimation as explained by the authors in [19]. The AP processes a new, 8-bit input symbol every clock cycle and is stalled by

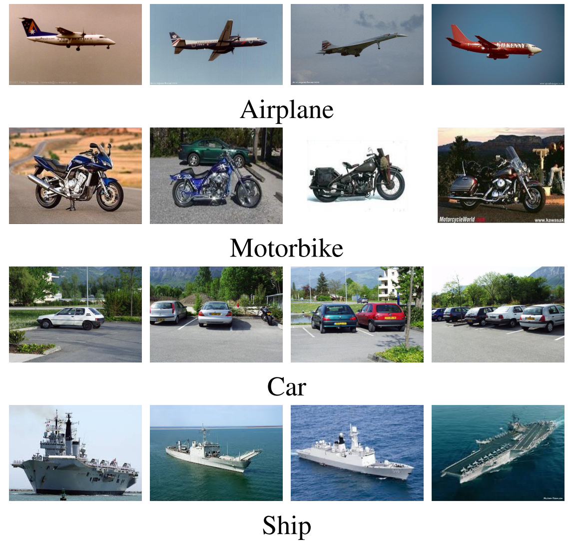

Fig. 6. Vehicle dataset with four categories: (a) airplane (b) motorbike (c) car and (d) ship.

40 nanoseconds in an output buffer every time it reports. The run-time estimation for the AP is calculated to take $[(16 + 40p + l)] * 7.5$ nanoseconds to run, where 16 is the inital setup latency, p is the number of output vectors per cycle, l is the number of STEs reported in that cycle, and 7.5 ns is the clock cycle. The computational cost for both datasets deployed on the AP are compared to the runtime of comparison methods, as shown in Table II. The comparison only reports the time it takes to classify the descriptors. The SHIRC method takes .55s to classify one image running in Matlab on a 64-bit Windows 7 system equipped with Intel Core i72600 3.4-GHz processor and 8 GB RAM [15]. We compute the classification run-time in each method for one image and see a significant speedup.

VI. DISCUSSION

There were several notable discoveries and challenges in our work that can be improved on in future work. Overall, our experiments show that multiple feature extraction methods can be encoded into regular expressions and used to implement image retrieval on an automata framework. However, because the AP methods do not yet achieve state-of-the-art accuracy, our results suggest a trade-off between speed and accuracy. The reduced accuracy may be acceptable in applications requiring significant speedup. However, this work is just the first step in exploring potential feature descriptors and mappings for automata processing. While the accuracy of image classification with regular expressions has not reached current state-of-the-art accuracy, there is clear room for future work to achieve improvement.

Feature Extraction. The second experiment shows that more discriminative descriptors are simpler to implement on exact matching automata and attain better retrieval accuracy. Future work includes combining feature extraction methods and reducing to a discriminative representation.

Mapping distributions. As in the first experiment, the feature descriptor may not have a known distribution which is a challenge when encoding the descriptor to regular expressions. The experiment demonstrated manipulation of automata STE symbols to aid this challenge. There are other potential methods to encode these descriptors such that information is not lost and thus improve performance. Nonlinear feature descriptors may be more efficiently encoded to regular expressions by using kernel mapping methods.

Image Matching on the AP. We showed a method of matching on an automata framework where images are classified based on how many automata they match. Further manipulation of AP elements and re-configuring automata structures that do not require exact descriptor matching can be applied. This could improve accuracy and reduce run-time of image retrieval applications [20].

VII. CONCLUSIONS AND FUTURE WORK

The Automata Processor allows potential accelerating pattern matching applications in image retrieval. In this paper, we proposed a process for encoding an image to regular expressions for implementation of image retrieval on an automata framework. The process requires acquiring discriminative feature descriptors and encoding them into regular expressions. While the accuracy for image classification deployed on the AP has not reached that of state-of-the-art methods, we find a large run-time speed-up. This motivates further work to identify feature descriptors and encoding methods that are more effective for automata implementations. This work could also be extended to multimodal classification using multiple feature extraction and mapping methods.

VIII. ACKNOWLEDGEMENTS

This work was supported in part by the Army Research Office (W911NF-15-1-0275), NSF grant EF-1124931, Micron Technology and by C-FAR, one of six centers of STARnet, a Semiconductor Research Corporation program sponsored by MARCO and DARPA.

IX. REFERENCES

[1] P. Dlugosch, D. Brown, P. Glendenning, M. Leventhal, and H. Noyes, "An efficient and scalable semiconductor architecture for parallel Automata Processing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 12, pp. 3088–3098, 2014.

[2] K. Zhou, J. J. Fox, K. Wang, D. E. Brown, and K. Skadron, "Brill tagging on the Micron Automata processor," in *Semantic Computing (ICSC), 2015 IEEE International Conference on*. IEEE, 2015.

[3] I. Roy and S. Aluru, "Finding motifs in biological sequences using the Micron Automata Processor," in *Parallel and Distributed Processing Symposium, 2014 IEEE 28th International*. IEEE, 2014.

[4] T. Tracy II, Y. Fu, I. Roy, E. Jonas, and P. Glendenning, "Towards machine learning on the automata processor," in *International Conference on High Performance Computing*. Springer, 2016, pp. 200–218.

[5] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[6] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1. IEEE, 2005.

[7] L. Fei-Fei and P. Perona, "A bayesian hierarchical model for learning natural scene categories," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 2. IEEE, 2005.

[8] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 2. IEEE, 2006.

[9] Y. Ke and R. Sukthankar, "PCA-SIFT: A more distinctive representation for local image descriptors," in *Computer Vision and Pattern Recognition, Proceedings of the 2004 IEEE Computer Society Conference on*, vol. 2. IEEE, 2004.

[10] F. Perronnin, Y. Liu, J. Sánchez, and H. Poirier, "Large-scale image retrieval with compressed fisher vectors," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2010.

[11] K. Wang, K. Angstadt, C. Bo, N. Brunelle, E. Sadredini, T. Tracy, II, J. Wadden, M. Stan, and K. Skadron, "An overview of Micron's Automata processor," in *Proceedings of the Eleventh IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis*, 2016.

[12] K. Angstadt, W. Weimer, and K. Skadron, "RAPID Programming of Pattern-Recognition Processors," in *Proceedings of the Twenty-First International Conference on Architectural Support for Programming Languages and Operating Systems*. ACM, 2016.

[13] J. Wadden, V. Dang, N. Brunelle, T. Tracy II, D. Guo, E. Sadredini, K. Wang, C. Bo, G. Robins, M. Stan *et al.*, "Anmlzoo: a benchmark suite for exploring bottlenecks in Automata Processing engines and architectures," in *Workload Characterization (IISWC), 2016 IEEE International Symposium on*. IEEE, 2016.

[14] R. Sarkar and S. T. Acton, "Slide: Saliency guided image dictionary and image similarity evaluation," in *2016 IEEE International Conference on Image Processing (ICIP)*, 2016.

[15] U. Srinivas, H. Mousavi, C. Jeon, V. Monga, A. Hattel, and B. Jayarao, "SHIRC: A simultaneous sparsity model for histopathological image representation and classification," in *2013 IEEE 10th International Symposium on Biomedical Imaging*. IEEE, 2013.

[16] L. Fei-Fei, R. Fergus, and P. Perona, "Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories," *Computer Vision and Image Understanding*, vol. 106, no. 1, pp. 59–70, 2007.

[17] M. Marszalek and C. Schmid, "Accurate object localization with shape masks," in *2007 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2007.

[18] M. Elad and M. Aharon, "Image denoising via sparse and redundant representations over learned dictionaries," *IEEE Transactions on Image Processing*, vol. 15, no. 12, pp. 3736–3745, 2006.

[19] I. Roy, A. Srivastava, M. Nourian, M. Becchi, and S. Aluru, "High Performance Pattern Matching using the Automata Processor," in *Parallel and Distributed Processing Symposium, 2016 IEEE International*. IEEE, 2016.

[20] I. Roy, A. Srivastava, and S. Aluru, "Programming techniques for the Automata Processor," in *Parallel Processing (ICPP), 2016 45th International Conference on*. IEEE, 2016.