

## RECONFIGURABLE ARCHITECTURES AND ALGORITHMS: A RESEARCH SURVEY

RAJEEV WANKAR

*Department of Computer and Information Sciences, University of Hyderabad,  
P.O. Central University,  
Hyderabad, AP 500046, India  
wankarcs@uohyd.ernet.in  
http://uohyd.ernet.in/~wankarcs*

RAJENDRA AKERKAR

*Technomathematics Research Foundation  
Kolhapur, India  
raakerkar@yahoo.com*

Ever since the introduction of the Dynamically Reconfigurable Buses, the architecture gained a lot of popularity amongst the researchers and scientists for its high performance computing with general purpose processor used. It is a powerful model of computation in which communication pattern between the processors could be changed during the execution. Following the years several new architectures and efficient algorithms for these were proposed, and their implementation using FPGA's have been shown. This paper presents a survey on the different architectures proposed, and few important algorithms presented for these specialized architectures over the period of last two decades.

*Keywords:* PARBS, R-MESH, RN, LARPBS, Polymorphic Torus Network, AROB.

### 1. Introduction

P-RAM model is a fundamental model of parallel computation which allows concurrent read but not the concurrent write operations [FW78]. Another model which removes this restriction on concurrent write is W-RAM model. Kucera [Kuc82] showed how efficient parallel algorithm can be devised using this model. In their paper it has been shown that with  $n^2$  processors we can determine the minimum of  $n$  numbers in constant time, using  $n^4$  processors we can determine the connected component of a graph with  $n$  vertices in  $O(\log n)$  time, with  $n^5$  processors minimum spanning tree of an edge weighted graph with  $n$  vertices can be obtained in  $O(\log n)$  time, with  $n^4$  processors all pairs shortest path problem can be solved in  $O(\log n)$  time and with  $n^4$  processors a topological sorting of  $n$  vertices of directed acyclic graph can be found in  $O(\log n)$  time, all on W-RAM model. Majority of the practical shared memory computers with coarse grain processing elements are based on these models.

Following years different other organizations of processors were also proposed, such as pyramid computer, mesh of tree and meshes with broadcast buses [Bok84]. These

organizations were static in nature and communication pattern between the processors could not be changed during the execution of the algorithm.

In March 1988, in the conference on advanced research in VLSI technology Miller, Kumar *et al.* [MKRS88] proposed a new powerful model of computation known as Meshes with reconfigurable buses. After the introduction, this architecture gained a lot of popularity for its high performance computing with general purpose processor used. They described their architecture as a VLSI array of processors, overlaid with a reconfigurable bus system. The model proposed by Miller, Kumar *et al.* offers dynamic reconfigurability during the execution and allows making different configurations to fulfill the different computational needs.

From the application developer perspective, for a given application, at a given time, the reconfigurable architecture is a spatial structure of the device that will be modified such as to use the best computing approach to speed up that application. If a new application has to be computed, the device structure will be modified again to match the new application. Contrary to the Von Neumann computers, which are programmed by a set of instructions to be executed sequentially, the structure of reconfigurable devices are changed by modifying all or part of the hardware at compile-time or at run-time, usually by downloading a so called bit stream into the device.

In this survey paper we highlight the work carried out by the scientist in last few years and present few important results on the algorithmic development on these reconfigurable architectures for solving some fundamental problems. The survey paper is organized in the following way: Section 2 introduces basic architectures of dynamically reconfigurable architecture; Section 3 introduces algorithms development on Reconfigurable MESH, Processor Arrays with Reconfigurable Bus Systems (PARBS) and Polymorphic torus network. In Section 4 we talk about the research carried out in Reconfigurable Networks and other networks. In Section 5 algorithms development on Linear Arrays with Reconfigurable Pipelined Bus System (LARPBS), Array with Reconfigurable Optical Buses (AROB) and CD-PARBS are discussed. In Section 6, we highlight some of the application areas where reconfigurable computing elements can be adopted.

## **2. Basic architectures of dynamically reconfigurable architecture**

The basic computational unit of reconfigurable mesh is the Processing Element (PE) which consists of switches, small storage and an ALU. A PE is capable of performing (1) Setting up a connection pattern, (2) Read from or write onto a sub bus or memory, (3) Performing logical and arithmetic operations and (4) Disconnecting itself from the bus in one unit time. Four parameters are used to characterize the Reconfigurable bus models (A) *Width*: It refers to the data width of the PE. There are two models which differ in the length of the operand of the PE, Bit model and Word model. (B) *Delay*: it is the time needed to propagate a signal on the buses. The two models of PEs are (1) Unit delay

model: no matter how far signal has travelled, (2) Logarithmic delay model:  $O(\log_2 N)$  time is needed. (C) *Bus Access*: each PE is connected to the bus through its port and will either read or write to it. There are two common models: (1) ER Model (Similar to CREW P-RAM), (2) CR Model (Similar to CRCW P-RAM). (D) *Connection Pattern*: Each PE can set the connection between its four ports based on local data or a global instruction. There are 15 different connection patterns possible. Models differ in the number of connection patterns (a subset of 15), which they allow [BK97].

Based on these classifications, various models of reconfigurable bus system appeared in the literature. Most of these models are synchronous in nature and permit unconditional global switch as well as the local switch settings. Unconditional switch setting is performed by broadcasting a global instruction from a central controller.

These models differ in the way they are allowed to make internal connections, a few to note are: PARBS (Processor Arrays with Reconfigurable Bus System), RMESH (Reconfigurable Mesh), RN (Reconfigurable Network) and Polymorphic Torus Network.

*PARBS*: It is the most general and most powerful model. In PARBS no restriction is placed on allowed connections that means at each node all 15 patterns of internal connections (notation  $\{xy\}$  to mean that port  $x$  and  $y$  are connected to each other) are possible, these are: (1) no connections -  $\{\}$ , (2) two-port connections -  $\{NS\}$ ,  $\{EW\}$ ,  $\{NW\}$ ,  $\{NE\}$ ,  $\{SW\}$ ,  $\{SE\}$ , (3) three-port connections -  $\{EWS\}$ ,  $\{EWN\}$ ,  $\{SNE\}$ ,  $\{SNW\}$ , (4) four-port connections -  $\{EWSN\}$ , (5) two-pair connections -  $\{EW, SN\}$ ,  $\{EN, WS\}$ ,  $\{ES, WN\}$ [LS95].

*Polymorphic Torus*: It is identical to the PARBS except that the rows and columns of the underlying mesh wrap around [LM89].

*RN*: The Reconfigurable Network is a general model in which PEs may not lie at the grid point and a bus segment may join an arbitrary pair of PEs [APRS91]. "In RN model, each I/O port of a PE is connected to at most one other port".

*RMESH*: It is a two dimensional mesh where the PEs are located on the intersection of the grid lines of reconfigurable bus [MKRS93].

### 3. On Reconfigurable MESH and PARBS

The early work on the array processors with static global bus for finding maximum of a set of values, stored one per processor on an array processors appeared in [Bok84]. Bokhari presented two phase algorithm that uses conventional links during the first phase and the global bus during the second phase. He used two types of interconnection patterns: the eight neighbors and the four neighbors, and shown that the time required for finding the maximum using the two phase algorithm is  $O(n^{2/3})$ , assuming the propagation speed of the global bus to be a constant independent of the size of the array. The case where propagation speed is logarithmic in the number of processors, the time to find maximum is  $O(n^2 \log n)^{2/3}$ , for both types of arrays.

The work of Bokhari was extended to obtain the optimal bounds for finding maximum on array processors with  $k$  global buses by Alok Aggarwal [Agg86]. He generalized this model and demonstrated that the bound  $T = \Omega\left(\left(n^d G n / k\right)^{\frac{1}{d}+1}\right)$ . He assumed that every processor is connected to all  $k$  buses and it receives all  $k$  broadcasts simultaneously at interval of  $Gn$  time step. In his work, he obtained the bounds for matrix multiplication and other functions too.

Miller, Kumar *et al.* [MKR88] presented solution of many important and basic problems. They proposed  $O(\log n)$  time algorithm on unit delay model and  $O(\log^2 n)$  time algorithm on logarithmic delay model for the parallel prefix computation. They presented a constant time algorithm for finding OR of  $n$  bit numbers on unit delay model and  $O(\log n)$  time algorithm on logarithmic delay model. With their proposed work, maximum of  $\sqrt{N}$  data items on mesh of size  $N$  was obtained in  $O(1)$  time. They also showed that any normalized algorithm, running  $T(N)$  time on a mesh of tree of base size  $N$  can be simulated on a reconfigurable mesh of size  $N$  to finish in  $O(T(N))$  time under the unit delay model, and in  $O(T(N) * \log n)$  time under the log time delay model. They presented several results for the graph related problems. In their paper they described algorithms with better parallel time complexities than the existing best ones. They showed how the reconfigurable meshes can be used to simulate certain fundamental techniques that have been developed for P-RAM and W-RAM model of computations.

After the introduction of the dynamically reconfigurable architecture many models appeared in the literature. Polymorphic torus network is an interconnection network for a massively parallel time-grained SIMD system. The design goal of this architecture was to provide a high communication bandwidth under packaging constraints. This goal was achieved by the polymorphic principal who injects switching capabilities to every node of a base network. The polymorphic approach gives flexibility in reconfiguring the switches, individually and dynamically, to match the algorithmic requirements. Li and Maresca [LM89] presented configurations for few basic arithmetic operations and have shown that the OR of  $N = n \times n$  Boolean numbers  $x_i, 0 \leq i \leq N - 1$ , distributed uniformly over the polymorphic torus network of size  $n \times n$  can be obtained in constant time. They also shown that the maximum/minimum of given  $N = n \times n$ ,  $k$ -bit numbers  $x_i, 0 \leq i \leq N - 1$ , distributed uniformly over the polymorphic torus network, the maximum operation can be performed in  $O(k)$  time. Same way they proposed a  $O(\log n)$  time algorithm for summing  $N$  numbers.

Mesh connected computers are widely used for their regular structure and simple interconnections which are suitable for VLSI implementation. Chen *et al.* [CCCS90], proposed an algorithm to perform semi-group computations in  $O(n^{1/8})$  time having  $(N^{5/8} \times N^{5/8})$  rectangular 2-mesh connected computers with multiple broadcasting.

Sorting is one of the most fundamental problems in computer science. Many researchers worked for finding constant time parallel solution of this problem. Wang, Chen *et al.* [WCL90] gave a constant time algorithm to sort  $N$  data items on a three diagonal array of  $O(N^3)$  processors. In their approach they used a four triangular processor arrays with a reconfigurable bus system whose bottom processors are connected to a square processors array, each processor is having 6 ports. They used the concept of binary summation by proving that the triangular processor array does compute the sum of the input binary sequence.

Many problems in science and engineering can be formulated in terms of directed and undirected graphs. Designing a parallel graph algorithm is both theoretically & practically important. Many researchers are extensively engaged in finding graph formulation and parallelization of important problems. Wang and Chen [WC90] proposed a  $O(1)$  time algorithm for computing transitive closure of an undirected graph. They designed two algorithms, one on a  $n \times n \times n$  PARBS, and other on a 2-D,  $n^2 \times n^2$ , PARBS. In their excellent work, they presented constant time parallel algorithms for many problems including recognizing bipartite graph, finding connected components, articulation point, bi-connected components bridges and minimum spanning tree in undirected graphs.

The algebraic path problem is a class of problem which includes transitive closure, all pairs shortest paths, minimum spanning tree etc. One of the methods for solving the problem of shortest path is using matrix multiplication. In the worse case this matrix multiplication takes  $O(\log n)$  time but in some cases  $O(1)$  time is possible. Chen, Wang *et al.* [CWL92] have shown that these problems can be solved in  $O(\log n)$  time with matrix multiplication using a  $n^2 \times n \times n$ , PARBS.

Addition and multiplication operations are very important functions in many computer applications. Being the fundamental operations, the speed of the addition/ multiplication is the most important factor in computations. Many parallel algorithms for addition and multiplication on PARBS appeared in the literature. Thangavel & Muthuswamy [TM93] proposed a  $O(1)$  time algorithm for computing the sum of two  $n$  bit binary numbers on linear PARBS and  $O(\log n)$  time for computing product of two  $n$  bit numbers on a  $n \times 2n$ , PARBS.

Chen, Wang *et al.* [CWL93] gave new approach, based on functional decomposition, for deriving algorithms on PARBS. They have shown that computing the logical XOR of  $n$ -bits can be done in  $O(1)$  time on a  $2n \times 3$ , PARBS, summation of  $n$  bits can be done in  $O(1)$  time on a  $n \times (n+1)$ , PARBS, summing  $n, m$  bit integers can be done in  $O(1)$  time on a  $2mn \times 2mn$ , PARBS and multiplication of two  $n$ -bit binary integers can be done in  $O(1)$  time on a  $4n^2 \times 4n^2$ , PARBS.

Matrix multiplication is an important basic operation in many computations. Park, Kim *et al.* [PKK93] proposed an architecture, based on the reconfigurable mesh, which can compute two  $N \times N$ , matrix multiplications in constant time. They assumed that each element  $N$  is represented using  $O(\log n)$  bits and requires  $N^2 \times N$ , reconfigurable mesh. Fragopoulou [Fra93] proposed an algorithm for summation of  $N$  numbers in the range  $[0, 2^b]$  in  $O(b + \log \log n)$  time on a  $\sqrt{N} \times \sqrt{N}$ , reconfigurable mesh and developed it as a straight forward application of the  $O(\log \log n)$  time binary digit summation.

An exhaustive work on parallel computation on reconfigurable meshes appeared in the paper of Miller, Kumar *et al.* [MKRS93]. They have presented efficient fundamental data movement operations. They have shown how to embed other parallel architectures into reconfigurable mesh and presented efficient solutions of graph and image problems that rely on the fundamental data movement operations.

Chen and Chen [CC94] presented a constant time sorting algorithm by adopting 3-D Reconfigurable Mesh with only  $O(n^{2/3})$  processors. They developed the algorithm on a  $n^{1/2} \times n^{1/2} \times n^{1/2}$ , 3D-RM. They extended the result to  $k$ -dimensional RM for  $k \geq 3$ , consequently a  $O(4^{k+1})$  time sorting algorithm is realized by adopting a  $n^{1/(k+1)} \times n^{1/(k+1)} \times n^{1/(k+1)}$ , RM.

Jang and Kumar [JK95], in their paper have shown non trivial ways to use the RM to solve several arithmetic problems in constant time. These solutions are obtained by novel ways to represent numbers and by exploiting the reconfigurability of the arithmetic. In particular a constant time algorithm to add  $n$ ,  $k$ -bit numbers using the  $n \times nk$  bit model of RM is shown. Using their technique, an optimal sorting algorithm on the RM has been derived. The algorithm sorts  $n$  numbers in constant time using  $n \times n$  processors. They claimed that the algorithm runs on all known variants of the RM model.

Koji Nakano [Nak95a] has given algorithms to optimally initializing the RM in which every processor executes the same program and learn its  $x$  coordinate and  $y$  coordinate, that is each PE( $i, j$ ) learns the value of  $i$  and  $j$ . It has been shown that initializing a RM of the bit transfer model can be performed in  $O(\log n)$  time and the same operation of executive word transfer model can be performed in  $O(\log \log n)$  time. In another correspondence [Nak95b], an efficient but not optimal algorithm for obtaining prefix sums of  $n$  binary values has been proposed. The algorithm takes

$O\left(\frac{\log n}{\sqrt{m \times \log m + \log \log n}}\right)$  time on  $n \times m$  reconfigurable mesh of the word model.

It has also been shown that the prefix-sums of  $n$  binary values can be computed in

$O\left(\frac{\log n}{\sqrt{m \times \log m + 1}}\right)$  time on  $n \times m$  reconfigurable mesh of the word model, if the

reconfigurable mesh has communication capability that allows simultaneous sending to the same bus.

Nakano and Wada [NW95] presented an algorithm for summing  $n$ ,  $d$ -bit integer on the bit and word model of reconfigurable bus architecture. They have shown that the sum can be obtain in constant time on a  $2dn \times 2n$  RM of the bit model, product of  $n$ -bit and  $n'$  bit integer can be computed in constant time on a  $2n(n+n'+1) \times 2n$  RM of the bit model.

Further the sum of  $n$  binary values can be computed in  $O\left(\frac{\log n}{\sqrt{m \times \log m}}\right)$  time on a  $n \times m$  RM of the bit model. They have presented many other results on bit and word models.

Later, in their other communication Nakano and Wada [NW98] have shown following algorithms to compute the sum of  $n$   $d$ -bit integers on reconfigurable parallel computation models: (1) a constant-time algorithm on a reconfigurable mesh of the bit model of size  $\sqrt{n} \log^{O(1)} n \times d\sqrt{n}$ , (2) an  $O(\log^* n)$ -time algorithm on a reconfigurable mesh of the bit model of size  $\sqrt{n/\log^* n} \times d\sqrt{n/\log^* n}$ , (3) an  $O(\log d + \log^* n)$ -time algorithm on a reconfigurable mesh of the word model of size  $\sqrt{n/(\log d + \log^* n)} \times \sqrt{n/(\log d + \log^* n)}$ , and (4) an  $O(\log^* n)$ -time algorithm on a VLSI reconfigurable circuit of area  $O(dn/\log^* n)$ , improving all previously claimed timings.

Bokka, Gurla *et al.* [BGOS95] proposed constant time solutions for testing an arbitrary polygon for convexity, solving the point location problem, solving the supporting lines problems, solving the stabbing problem, determining the minimum perimeter corner triangle for a convex polygon, determining the  $k$ -maximal vertices of a restricted class of polygons and other related problems in constant time, using the  $\sqrt{n} \times \sqrt{n}$  RM model of PARBS. They assumed that one or two  $n$  vertex polygons are pretiled one vector per processor onto a RM of size  $\sqrt{n} \times \sqrt{n}$ .

One of the fundamental issues which arise is, whether the reconfigurable model is a basis for the design of massively parallel computers? Or in other words, this self simulation problem ask, can the given algorithm, designed for a large RM, be executed on a smaller RM? In their work Asher, Gordon and Schuster [AGS95] have shown how this simulation can be carried out optimally. They have shown a technique to achieve asymptotically optimal self simulation on models which allows buses to switch column and row edges, provided that the bus is a linear path of connected edges.

Divide and conquer is one of the most favorite design paradigm for solving sequential or parallel algorithms. In the paper of Lai and Sheng [LS95], it has been shown that dividing a problem evenly is not necessarily a good approach. For some problems natural decomposition may be preferable, in which the problem is divide into subproblems along their natural boundaries, resulting in a irregular decomposition. Taking this approach, they obtained a new R-Mesh algorithm that triangulates a set of point in  $O(1)$  time.

Precisely, they have shown that triangulating a uni-monotone, simple polygon of  $n$  vertices can be computed in  $O(1)$  time on a  $n \times n$  Reconfigurable mesh.

Chen, Olariu *et al.* [COS<sup>+</sup>96] given a constant time tree algorithm on RM of size  $n \times n$ . They have presented some parallel algorithms for basic data movement. In their work they have shown that the sorting of  $n$  left and right parentheses as input, where the string is well formed, can be determined in  $O(1)$  time on a RM of size  $n \times n$ . In case the string is well formed, all matching pairs can be found in  $O(1)$  time on the same reconfigurable mesh.

List ranking problem is one of the fundamental design paradigms for parallel algorithms. In their work Hiyashi, Nakano *et al.* [HN096] suggested that an instance of size  $n$  of the list ranking problem can be solved in  $O(1)$  time on a  $n \times n$  RMESH, provided that a good ruling set is available. Later, Kim and Park [KP2K] shown that the same deterministic and randomized time complexities can be achieved using only  $O(n^{1+\epsilon})$  processors, where  $\epsilon$  is an arbitrary positive constant  $< 1$ . To reduce the number of processors, they adopted a reconfigurable mesh of high dimensions and develop a new technique called path embedding.

Jang, Nigam *et al.* [JNKS97] developed constant time algorithms for computational geometry problems. These problems include convex hull,  $k$ -dimensional maxima, two set dominance counting, smallest enclosing box, all pairs nearest neighbor and triangularization, all on the RM. They have shown that given  $n$  points, stored in a row of the reconfigurable mesh, all these problems can be solved in  $O(1)$  time on a  $N \times N$  RM.

Jang, Park and Kumar [JPK97] have proposed a  $O(1)$  time algorithm to multiply two  $N$  bit numbers using  $N \times N$  bit model of RM. The result is achieved by using a technique for data representation and data movement and using multidimensional Radar Transform (RT). They used the known concept of RT, which is a transform in a finite field (generally in a ring), has cyclic convolution property and is without round off errors. The algorithm has been extended to result in  $AT^2$ , optimally over  $1 \leq T \leq \sqrt{N}$ . in a variant of the bit model of VLSI. They used three different schemes to represent numbers.

Bondalapati and Kumar [BK98] developed a formal methodology for mapping loops onto reconfigurable architectures. They developed a parameterized abstract model of reconfigurable architecture which is general enough to capture a wide range of configurable systems. They have given a polynomial time algorithm to compute the optimal sequence of configuration for one important variant of the problem. They have proved that the optimal sequence of configuration of  $N$  iterations of a loop statement with  $p$  tasks, when each task can be executed in one of  $m$  possible configurations, can be computed in  $O(pm^3)$  time.

Yingyu et al. [YY+2K] proposed efficient algorithms for computing the minimum spanning tree of an  $n$ -vertex undirected graph. One runs on  $n \times n$  reconfigurable mesh with time complexity of  $O(\log^2 n)$ , other runs with time complexity of  $O(\log n)$  on an  $n^{1+\epsilon} \times n$  reconfigurable mesh, where  $0 < \epsilon < 1$  is a constant.

Multiple additions are the problem of adding  $N$   $b$ -bit integers. Prefix sums and multiple additions play fundamental roles in many algorithms, particularly on the reconfigurable mesh (R-Mesh). Scaling algorithms on the R-Mesh to run with the same or increased efficiency on fewer processors is a challenging and important proposition. Trahan and Vaidyanathan presented [TV02] new algorithms that scale with increasing efficiency for multiple addition, prefix sums, and matrix–vector multiplication. They have shown that

for  $T = O\left(\sqrt{\frac{b}{\log(b + \log N)}}\right)$  and  $0 \leq T \leq N$ , given  $N$   $b$ -bit integers, the prefix sums of these numbers can be computed on a word-model LR-Mesh of size  $O\left(\frac{\left(\frac{b}{T} + \log \frac{N}{T}\right)^2}{\log\left(\frac{b}{T} + \log \frac{N}{T}\right)}\right) \times O\left(\frac{N}{T} \cdot \left(\frac{b}{T} + \log \frac{N}{T}\right)\right)$  in  $O(T)$  time. Furthermore, if  $T = \Omega(\log(b + \log N))$ , then the prefix sums are in word format; otherwise, they are in BIN format.

#### 4. On Reconfigurable Network (RN) and other Networks

The work done by Asher, Pegeg *et al.* [APRS91] is a detailed description of the Reconfigurable Network (RN). They investigated the theoretical limits of the computation power of a constant degree reconfigurable network with a polynomial number of switches and focused the operations, requiring constant time. They have presented a precise definition of Reconfigurable Network, a model of PARBS, and presented some basic tools, exemplified via the design of efficient algorithms for the problems of addition and sorting. In particular they have constructed a constant time optimal area sorting network. They have shown the relationship between RN and branching programs. They also investigated the relationship between the RN and P-RAM models and shown that an EREW P-RAM may simulate any RN in logarithmic time. They have also shown, by introducing different kind of switches, that the power of RN can be enhanced so every problem in P-time can be answered in constant time by a RM.

The hyperbolic broadcast bus network consists of processors which are connected by global buses only. Tsai, Tsai *et al.* [THT<sup>+</sup>97] designed constant time basic operations for finding the maximum/minimum of  $N$  numbers, each of size  $O(\log n)$  and computing a matrix multiplication operation of two  $N \times N$  matrix. Based on these two operations algebraic path problem, connectivity problem and several problems are solved in  $O(\log n)$  time.

Biing-Feng Wang [Wang98] addressed the problem of simulating the CRCW PRAM on reconfigurable networks. He shown that if  $N$  and  $M$ , respectively, be the numbers of processors and memory cells contained in the CRCW PRAM then a two-dimensional  $N \times MN^{1/r}$  reconfigurable network can simulate any operation performed on the CRCW PRAM in  $O(1)$  time, where  $r \geq 2$  and is a constant. If  $N \leq M$ , then any operation performed on the CRCW PRAM can be simulated in  $O(1)$  time on a  $r$ -dimensional  $N^{1/r-1} \times N^{1/r-1} \times \dots \times N^{1/r-1} \times (M/N^{r-2r-1})$  reconfigurable network, where  $r \geq 2$  and is a constant.

### 5. On CD-PARBS and Optical bus based Networks

An Array with Reconfigurable Optical Buses (AROB) is essentially a  $m \times n$  RM in which the buses are implemented using optical technology. In their paper, Rajsekaran and Sahni [RS97] presented efficient algorithms for sorting, selection and packet routing on the AROB model. Their algorithm sorts  $n$  general keys in  $O(1)$  time on AROB of size  $n^\varepsilon \times n$ , for any constant  $\varepsilon > 0$ . They have also shown that selection from, out of  $n$  elements can be done in randomized  $O(1)$  time using  $n$  processors.

Another practical parallel computational model known as linear arrays with reconfigurable pipelined bus system (LARPBS), which uses fiber optics communication, has been proposed by Pan and Li [PL98]. They have shown several basic movement operations on this model, these include broadcast multicast compression, split, binary prefix sum, maximum finding etc. Using these operations several image processing algorithms are also presented.

Pan et al. [PHL98] also proposed a parallel quick-sort algorithm on the same model (LARPBS) and shown that for a set of  $N$  numbers, the quick-sort algorithm runs in  $O(\log_2 N)$  average time on a linear array with a reconfigurable pipelined bus system of size  $N$ . If the number of processors available is reduced to  $P$ , where  $P < N$ , the algorithm runs in  $O((N/P) \log_2 N)$  average time and is still scalable.

Parallel matrix multiplication problem is the heart of many scientific problems. Li, Pan and Zheng [LPZ98a, b] presented fast and cost efficient parallel algorithms for a number of important and fundamental matrix problems including computing the inverse of a matrix (or a lower/upper triangular matrix), the characteristic polynomial and the determinant of a matrix, the  $N^{\text{th}}$  power of a matrix, a LU and QR factorization of a matrix and solving linear system of equations, using reconfigurable pipelined bus systems. They have shown that the inverse of a lower triangular matrix can be obtained in  $O(\log N)^{1+\delta}$

time with  $O\left(N^3 / \left(\frac{8}{7}\right)^{(\log N-1)^\delta}\right)$ ,  $1 \leq \delta \leq 1$ , number of processors. The LU factorization of

a matrix can be performed in  $O(\log N)^{2+\delta}$  time with  $O\left(N^4 / \left(\frac{8}{7}\right)^{(\log N)^\delta}\right)$ ,  $1 \leq \delta \leq 1$ ,

processors. In another communication they have presented efficient parallel matrix multiplication algorithm for linear arrays with reconfigurable pipelined bus system. They developed five algorithms with varying degree of parallelism. In the literature, the best known matrix multiplication takes  $O(N^\beta)$  where  $\beta < 2.375$  [DW90], but the constant associated with these methods are so high that the parallelization is only of the theoretical interest. In their approach they described implementation of several primitive communication operations on LPARBS which are the building blocks of their algorithms. They have shown that matrix multiplication can be performed using one of their five algorithms presented on LARPBS model in  $O(\log n)$  time with  $O(N^{2.8074})$  processors, where each processor has  $O(\log n)$  memory.

Since PARBS cannot control the direction of the signal flow, for many directed graph problems it not possible to find correct connections on PARBS. Hence directed graph algorithms, including the redundant arc elimination problem cannot be solves always correctly. A new modified model called D-PARBS (directed PARBS) has been proposed to solve these directed graph problems by Kuo, Hsu *et al.* [KHF99]. They have shown constant time algorithms with  $O(N^3)$  processors to solve topological sort, transitive closure, cyclic graph checking and strongly connected problems on directed graphs.

Using the same model of computation an algorithm for the triangularization of a matrix whose graph is a directed acyclic graph, popularly known as dag, was proposed by Wankar *et al.* [WCF2K]. One of the algorithms for obtaining this special form has been given by Sargent and Westerberg. Their approach is practically good but sequential in nature and cannot be parallelized easily. In this work they presented a parallel algorithm which is based on the observation that, if we find the transitive closure matrix of a directed acyclic graph, count the number of entries in each row, sort them in the ascending order of their values and rank them accordingly, we get a lower triangular matrix. We show that all these operations can be done using 3-d CD-PARBS (Complete Directed PARBS) in constant time. The same approach can be used for the block cases, producing the same relabeling as produced by Tarjan's [Tar72] algorithm, in constant time.

Li, Yi Pan *et al.* [LPH2K] solved a number of important and interesting problems from graph theory on a linear array with a reconfigurable pipelined optical bus system. Their algorithms were based on fast matrix multiplication and extreme value finding algorithms. Some of the important results includes: The product of two  $N \times N$  Boolean matrices can be calculated in  $O(1)$  time by using  $O(N^3/\log n)$  processors. The tree problem for an undirected graph with  $N$  vertices can be solved in  $O(\log n)$  time by using  $N^3$  processors. The transitive closure of a directed graph with  $N$  vertices can be found in  $O(\log n)$  time by using  $O(N^3/\log n)$  processors. The strong components of a directed graph with  $N$  vertices can be found in  $O(\log n)$  time by using  $O(N^3/\log n)$  processors. The connected components of an undirected graph with  $N$  vertices can be found in  $O(\log n)$  time by using  $O(N^3/\log n)$  processors.

Eshaghian, Lili Hai [EH01] presented an electro-optical architecture called the Optical Reconfigurable Mesh (ORM). The ORM has two layers: the deflection layer and the processing layer. The processing layer is an  $N \times N$  reconfigurable mesh. They used additional deflection layer, situated directly above the processing layer, provides unit-time free space optical interconnections for the processors. They presented many algorithms using this new architecture and shown that given an  $N \times N$  image  $G$ , the convex hulls of all figures in  $G$  can be found in  $O(\log n)$  time using an  $N \times N$  ORM.

The Euclidean distance transform (EDT) is an operation to convert a binary image consisting of black and white pixels to a representation where each pixel has the Euclidean distance of the nearest black pixel. It has many applications in computer vision and image processing. Datta and Soundaralakshmi [DS04] presented two algorithms for computing the EDT on LARPBS. The first algorithm runs in  $O(\log \log n)$  time for a binary  $N \times N$  image on an LARPBS with  $N^{2+\epsilon}$  processors, for any fixed  $\epsilon$ ,  $0 < \epsilon < 1$ , and the second algorithm runs in  $O((N^2/P^2)\log \log p)$  time if the LARPBS has only  $p^{2+\epsilon}$  processors, for  $p < N$ .

Using linear array with a reconfigurable pipelined bus system Wang [Wang07] proposed to solve two and three dimensional all nearest neighbor (D\_ANN) problem, from image processing perspective. For two-dimensional (2D) binary image of size  $N \times N$ , he devise an 2D\_ANN using a LARPBS of size  $N^{2+\epsilon}$ , where  $0 < \epsilon < 1$  and for a three dimensional (3D) binary image of size  $N \times N \times N$ , he devised an algorithm for solving the 3D\_ANN problem using a LARPBS of size  $N^{3+\epsilon}$ , where  $0 < \epsilon < 1$ .

The reconfigurable mesh is a model for massively parallel computing for which many algorithms with very low complexity have been developed. These algorithms execute cycles of bus configuration, communication, and constant-time computation on all processing elements in a lock-step. In this paper, Giefers and Platzner [GP07] investigated the use of reconfigurable meshes as coprocessors to accelerate important algorithmic kernels. They discussed the development of a reconfigurable mesh on FPGA technology, including the host integration and the programming tool flow.

## 6. Applications

We present certain fields where the use of reconfiguration can be of great interest. Moreover since the field is still emerging, many new areas of application are likely to be developed in the future.

In order to secure market segments, manufacturers must release their products as quickly as possible. In many cases, a well working product can be released with less functionality. The manufacturer can then upgrade the product on the field to incorporate new functionalities. While this approach works with normal Von Neumann processors, it is not the case with application specific instruction set processors. A reconfigurable

device provides such capabilities of being upgraded in the field, by changing the configuration.

Contrary to software development, errors discovered after the production mean enormous loss because the produced pieces become either unusable or a great adaptation effort must be spent for the deployment. Rapid prototyping allows a device to be tested in real hardware before the final production. In this sense, errors can be corrected without affecting the pieces already produced. A reconfigurable device is useful here, because it can be used several times to implement different versions of the final product until an error-free state.

The design of ubiquitous computing system is cumbersome task that cannot be dealt with only at compile time. Because of uncertainty and unpredictability of such systems, it is impossible, at compile time, to address all scenarios that can happen at run-time, because of unpredictable changes in the environment. We need computing systems that are able to adapt their behavior and structure to change operating and environmental conditions, to time-varying optimizing objectives, and to physical constraints such as changing protocols and new standards. We call those computing systems Adaptive Computing System. Reconfiguration can provide a good fundament for the realization of adaptive systems, because it allows system to quickly react to changes by adopting the optimal behavior for a given run-time scenario.

## Conclusion

Several model of dynamically reconfigurable architecture have been proposed during last 20 years time. Several efficient/optimal parallel algorithms have been designed, but their implemented using FPGA's remains to be a difficult job. On the other side there are several other problems/applications as highlighted in section 6 for which the design of parallel algorithms using these architectures remains to be open research problems.

## References

- [Agg86] Alok Aggarwal, "Optimal Bounds for Finding Maximum on Array of Processors with k Global Buses", IEEE Transaction on Computers, Vol. c-35, No. 1, January 1986, pp. 62-64.
- [AGS95] Yosi Ben-Asher, Dan Gordon and Assaf Schuster, "Efficient Self-Simulation Algorithms for Reconfigurable Arrays", Journal of Parallel and Distributed Computing, 30 (1995), pp. 01-22.
- [APRS91] Y. Ben-Asher, D. Peleg, R. Ramaswami and A. Schuster, "The Power of Reconfiguration", Journal of Parallel and Distributed Computing, 13 (1991), pp.139-153.
- [BGOS95] V. Bokka, H. Gurla, S. Olaru and J.L. Schwing, "Constant Time Convexity Problems on Reconfigurable Meshes", Journal of Parallel and Distributed Computing, 27 (1995), pp.86-99.
- [BK97] K. Bondalapati and V.K. Prasanna Kumar, "Reconfigurable Meshes: Theory and Practice", Reconfigurable Architecture Workshop, International Parallel Processing Symposium, April 1997.
- [BK98] K. Bandalapati and V.K. Prasanna Kumar, "Mapping Loops onto Reconfigurable Architectures", International Workshop on Field Programmable logic, September 1998.
- [Bok84] S.H. Bokhari, "Finding Maximum on an Array Processor with a Global Bus", IEEE Transaction on Computers, Vol. c-33, No. 2, February 1984, pp. 133-139.

- [CC94] Yen-Cheng Chen and Wen-Tsuen Chen, "Constant Time Sorting on Reconfigurable Meshes", IEEE Transaction on Computers, Vol. 43, No. 6, June 1994, pp. 749-751.
- [CCCS90] Y. C. Chen, W. T. Chen, G. H. Chen, J. P. Sheu, Designing Efficient Parallel Algorithms on Mesh-Connected Computers with Multiple Broadcasting IEEE Transactions on Parallel and Distributed Systems Volume 1, Issue 2 (April 1990) Pages: 241 - 246
- [COS+96] G-H Chen, S. Olariu, J.L. Schwing, B-F. Wang and J. Zhang, "Constant Time Tree Algorithms on Reconfigurable Meshes on Size  $n \times n$ ", Journal of Parallel and Distributed Computing, 26 (1995), pp. 137- 150.
- [CWL92] G-H Chen, B-F Wang and C.J. Lu, "On the Parallel Computation of the Algebraic Path Problem", IEEE Transaction on Parallel and Distributed Systems, Vol. 3, No. 2, March 92, pp. 251-256.
- [CWL93] Gen-Huey Chen, Biing-Feng Wang and H. Li, "Deriving Algorithms on Reconfigurable Network Based on Functional Decomposition", Theoretical Computer Science 120 (1993), pp. 215-227.
- [DS04] Amitava Datta and Subbiah Soundaralakshmi, Fast and scalable algorithms for the Euclidean distance transform on a linear array with a reconfigurable pipelined bus system, Journal of Parallel Distributed Computing 64 (2004) 360–369.
- [DW90] D. Coppersmith and S. Winograd, "Matrix Multiplication via Arithmetic Progression", Journal of Symbolic Computation, Vol. 9, 1990, pp. 251-280.
- [EH01] Mary M. Eshaghian, Lili Hai, An Optically Interconnected Reconfigurable Mesh, Journal of Parallel and Distributed Computing 61, pp. 737-747 (2001).
- [Fra93] P. Fragopoulou, "On the Efficient Summation of N Numbers on an N Processor Reconfigurable Mesh", Parallel Processing Letters, Vol. 3 No.1 (1993), pp. 71-78.
- [FW78] Steven Fortune and James Wyllie, Parallelism in Random Access Machines, Proceedings of the tenth annual ACM symposium on Theory of computing San Diego, California, Pages: 114 – 118, 1978.
- [GP07] Heiner Giefers and Marco Platzner, A Many-Core Implementation Based on the Reconfigurable Mesh Model, FPL 2007, International Conference on, Field Programmable Logic and Applications, 27-29 Aug. 2007, Page(s): 41-46
- [HN096] Tatsuya Hayashi, Koji Nakano and Stephan Olariu, "Efficient List Ranking on the Reconfigurable Mesh with Applications", Hayashi Laboratory Technical Report TR96-03, may 1996.
- [JK95] Ju-Wook Jang and V.K. Prasanna Kumar, "An Optimal Sorting Algorithm on Reconfigurable Mesh", Journal of Parallel and Distributed Computing, 25 (1995), pp. 31-41.
- [JNKS97] Ju-Wook Jang, Madhusudan Nigam, V.K. Prasanna Kumar and S. Sahni, "Constant time Algorithm for Computational Geometry on the Reconfigurable Mesh", IEEE Transaction on Parallel and Distributed Systems, Vol. 8, No. 1, January 1997.
- [JPK97] Ju-Wook Jang, Heonchul Park, V.K. Prasanna Kumar, "An Optimal Multiplication Algorithm on Reconfigurable Mesh", IEEE Transactions on Parallel and Distributed Systems, Vol. 8, No. 5, May 1997.
- [KHF99] Chi-Jung KUO, Chiun-Chieh HSU and Wei-Chen FANG, "Parallel Directed Graph Algorithms on Directional Processor Arrays with Reconfigurable Bus Systems", New generation Computing, 17(1999), pp.175-200.
- [KP2K] Sung-Ryul Kim and Kunsoo Park, Efficient List Ranking Algorithms on Reconfigurable Mesh, in the 6th Annual International Conference, COCOON 2000 Sydney, Australia, July 26–28, 2000 Proceedings.
- [Kuc82] L. Kucera, "Parallel Computation and Conflict in Memory Access", Information Processing Letters, Vol. 14, number 2, April 1982, pp. 93-96
- [LM89] Hungwen Li and Massimo Maresca, "Polymorphic Torus Network", IEEE Transaction on Computers, Vol. 38, No. 9, September 1989, pp. 1345-1351.

- [LPH2K] Keqin Li, Yi Pan, Mounir Hamdi, Solving graph theory problems using reconfigurable pipelined optical buses, *Parallel Computing* 26 (2000) 723-735.
- [LPZ98a] K. Li, Yi. Pan and S. Q. Zheng, "Fast and Processor Efficient Parallel Matrix Manipulation Algorithm on a Linear Array with a Reconfigurable Pipelined Bus System", *IEEE Transactions on Parallel and Distributed Systems*, Vol. 9, No. 8, August 1998, pp. 507-520.
- [LPZ98b] K. Li, Y. Pan and S.Q. Zheng, "Fast and Efficient Parallel Matrix Manipulation on a Linear Array with a Reconfigurable Pipelined Bus System", *High Performance Computing Systems and Applications*, Kluwer Academic Press, 1998.
- [LS95] Ten-Hwang Lai, M.J. Sheng, "Triangulation on Reconfigurable Meshes: A Natural Decomposition Approach", *Journal of Parallel and Distributed Computing*, 30 (1995), pp. 38-51
- [MKRS88] R. Miller, V.K. Prasanna Kumar, Dionisios Reisis and Quentin F. Stout, "Meshes with Reconfigurable Buses", *Proc. 15th MIT Conference on Advance Research in VLSI*, March 1988, pp. 163-178.
- [MKRS93] R. Miller, V.K. Prasanna Kumar, Dionisios Reisis and Quentin F. Stout, "Parallel Computations on Reconfigurable Meshes", *IEEE Transactions on Computers*, Vol. 42, No. 6, June 1993.
- [Nak95a] Koji Nakano, "Optimal Initializing Algorithms for a Reconfigurable Mesh", *Journal of Parallel and Distributed Computing*, 24 (1995), pp. 218-223.
- [Nak95b] Koji Nakano, "Prefix sums algorithms on Reconfigurable Meshes", *Parallel Processing Letters*, Vol.5 No.1 (1995), pp. 23-35.
- [NW95] Koji Nakano and Koichi Wada, "Integer Summing Algorithms on Reconfigurable Meshes", *IEEE First International Conference on Algorithms and Architectures for Parallel Processing*, Brisbane, Australia, April 1995, pp. 19-21.
- [NW98] Koji Nakano, Koichi Wada, Integer summing algorithms on reconfigurable meshes, *Theoretical Computer Science* 197 (1998) pp. 57-77.
- [PKK93] Heonchul Park, H.J. Kim and V. Prasanna Kumar, "An O(1) time Optimal Algorithm for Multiplying Matrices on Reconfigurable Mesh", *Information Processing Letters*, 47(1993), pp. 109-113.
- [PL98] Yi Pan, Kequin Lin, "Linear Arrays with Reconfigurable Pipelined Bus System-Concept and Applications", *Journal of Information Sciences*, 106(1998), pp. 237-258.
- [PHL98] Yi Pan, Mounir Hamdi, Keqin Li, Efficient and scalable quicksort on a linear array with a reconfigurable pipelined bus system, *Future Generation Computer Systems* 13 (1997/98) pp. 501-513.
- [RS97] Sanguthevar Rajsekaran and Sartaj Sahni, "Sorting, Selection and Routing on the Arrays with Reconfigurable Optical Buses", *IEEE Transactions on Parallel and Distributed Systems*, Vol. 8, No. 11, Nov.1997.
- [Tar72] Robert Tarjan, "Depth-First Search and Linear Graph Algorithms", *SIAM Journal of Computing*, Vol. 1, No.2, 1972, pp. 146-160.
- [THT+97] Horng-Ren Tsai, Shi-Jinn Horng, Shun-Shan Tsai, Tzong-Wann Kao and Shung-Shing Lee, "Solving an Algebraic Path Problem and Some Related problems on a Hyper Bus Broadcast network", *IEEE Transactions on Parallel and Distributed Systems*, Vol. 12, No. 5, Dec. 1997.
- [TM93] P. Thangavel, V.P. Muthuswamy, "Parallel Algorithms for Addition and Multiplication on Processor Arrays with Reconfigurable Bus Systems", *Information Processing Letters*, 46(1993), pp. 89-94.
- [TV02] Jerry L. Trahan, Ramachandran Vaidyanathan, Scaling multiple addition and prefix sums on the reconfigurable mesh, *Information Processing Letters* Vol. 82, 6(2002), pp. 277-282.
- [Wang98] Biing-Feng Wang, Simulating the CRCW PRAM on reconfigurable networks, *Theoretical Computer Science* 205 (1998) pp. 231-242.
- [Wang 07] Yuh-Rau Wang, An efficient O(1) time 3D all nearest neighbor algorithm from image processing perspective, *J. Parallel Distributed Computing* Vol. 67, Issue 10 (2007), pp. 1082 – 1091.

- [WC90] Biing-Feng Wang and Gen-Huey Chen, "Constant Time Algorithms for the Transitive Closure and Some Related Problems on Processor Arrays with Reconfigurable Bus Systems", IEEE Transaction on Parallel and Distributed Systems, Vol. 1, No. 4, October 1990, pp. 500-507.
- [WCF2K] Rajeev Wankar, N.S.Chaudhari and Elfriede Fehr, "A Constant time parallel algorithm for the triangularization of a sparse matrix using CD-PARBS", Technical Report B-00-05, March 1, 2000, Freie Universität Berlin, Germany.
- [WCL90] Biing-Feng Wang, Gen-Huey Chen and Ferng-Ching Lin, "Constant Time Sorting on a Processor Array with Reconfigurable Bus System", Information Processing Letters, 34(1990), pp. 187-192.
- [YY+2K] AN Yingyu, XU Yinlong, GU Xiaodong and CHEN Guoliang, Efficient Minimum Spanning Tree Algorithms on the Reconfigurable Mesh, Vol.15 No.2 J. Comput. Sci. & Technol. Mar. 2000.