All Sciences Proceedings

5th International Conference on Applied Engineering and Natural Sciences

July 10-12, 2023 : Konya, Turkey

All Sciences Proceedings
http://as-proceeding.com/

https://www.icaens.com/

A Performance Constrained Metaheuristic Algorithm for 2D Mesh Based NoC

Aneeqa Khurshid*[1], Muhammad Iram Baig [2]

[1]Department of Electrical Engineering, University of Engineering and Technology Taxila, Pakistan
[2]Department of Electrical Engineering, University of Engineering and Technology Taxila, Pakistan

*(anikakhurshid86@gmail.com, iram.baig@uettaxila.edu.pk)

Abstract – Integrating large number of Intellectual Property (IP) cores onto a single chip has been made possible with the technology scaling and shrinking in the size of transistors. This has led designers to create a scalable and flexible on-chip network infrastructure – Network on Chip (NoC). Mapping application graphs on multiple cores of NoC architecture is one of the challenging research problems. The all in all performance of NoC network depends primarily on an efficient and cost-effective mapping technique which optimizes several performance metrics. These metrics primarily consists of communication cost, energy, throughput, latency, power dissipation and simulation time. A metaheuristic state-of-the-art nature inspired mapping approach for NoC's called Hunger Games Search Algorithm (HGSA) has been introduced in this research work. The devised algorithm minimizes NoC energy and power consumption based on six standard available embedded application benchmarks. Experimental results demonstrate that the presented technique outperforms as compared to other existing nature-inspired meta-heuristic application mapping approaches.

Keywords – Network on Chip, Hunger Games, Communication Cost, Metaheuristic, Application Mapping

I. INTRODUCTION

Rapid shrinkage in the transistors' size has made it possible to integrate large number of components on a single platform resulting in a multipurpose system-on-chip (SoC). A SoC consists of an entire system including processors, memory devices, I/O and A/D modules.

Technology scaling, on one end, provides a considerable amount of area to map more logic on a silicon chip, but on the other end come with the challenges at application layer. Traditional on-chip communication infrastructure becomes the bottleneck because a single bus cannot handle more than one communication at a time. Hence less priority tasks have to wait for a long time till more priority tasks have been completed, causing delays. Moreover, the bandwidth of the data is also restricted in a bus based system. So designers were forced to design a system to cater all these problems [1]. A promising solution to this issue is Network on Chip (NoC) – an architecture where cores are connected through routers [2]. NoC architecture is a highly scalable design with improved latency and bandwidth. On-chip networks usually connect multiple number of cores on a single chip [3]. Fig 1 shows a traditional bus based and mesh based NoC architecture. Communication between them is done through routers via some interface. Hence we can say that the basic building block of an on-chip network is router. Data communication between the cores/processing elements (PEs) is done through a message passing technique. The overall performance of the system depends on the type of structure of the interconnection between cores as due to a principle NoC takes around 40-42% of the total system power [4]. As a result, latency, power and overall performance of the system is highly dependent on the selection of topology of on-chip networks. Mesh topology is considered as the most extensively used topology in comparison to existing state of the art topologies in the NoC structure [5].

A mesh based NoC architecture ensures efficient parallel communication between cores because of same link sizes between cores and high bandwidth.

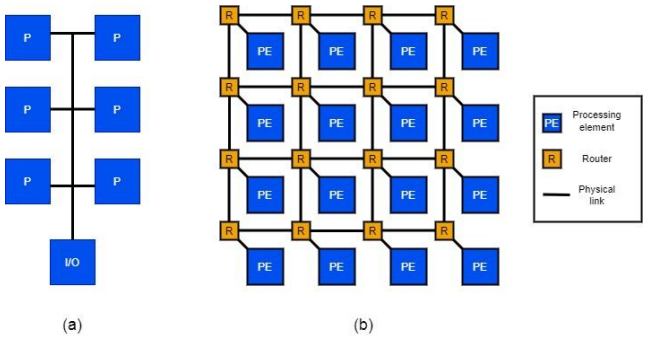

Fig. 1 – (a) bus based architecture (b) NoC architecture

Our research objective is mapping of applications, that how tasks get mapped on to the NoC tiles in an optimized way. Mapping various applications on cores is a (Non-Polynomial) N-P hard optimization issue which means that the range of search space rises factorially as the number of cores are escalated [6]. The relation for the region of search space is given by *c!* where *c* symbolizes the total number of cores. For example, for a 4 core network the size of search space is 4! = 24. On the basis of this context, numerous techniques for application mapping have been recommended that use iterative exact and search-based optimization methods. This has got a very significant role to play in the performance of the overall system as it directly influences the admissible delay of the router, the required link bandwidth and the total communication time. Trying to resolve complex problems by linear mathematical based solutions is not a useful approach. Hence scientists have developed different metaheuristic or heuristic methods as a solution to these complex optimization problems [7]. Basically a metaheuristic approach, after exploring the whole challenging search space with multiple local optima, provides a satisfactory solution depending upon the complexity of the problem. The optimum solution over several discussed NoC performance metrics is significantly dependent on the choice of the finest metaheuristic or heuristic technique.

After getting inspired from this, we proposed a novel meta-heuristic algorithm in this work, named Hunger Games Search Algorithm (HGSA), that takes inspiration from the supportive behavior of animal species where activity in searching for food is dependent on the level of an animals' hunger [8]. The proposed method is based on the behavior of animals when they get hungry and search for food for themselves.

The rest of this paper is organized as: Section 2 discusses some prior work in this research area. Section 3, 4 describes the proposed algorithm and the methodology used to gain the desired results. Section 5,6 covers the experimental setup and results of the research and last Section 7 gives concluding statements about the entire work with some future avenues.

II. Literature Review

Several algorithms were proposed to solve the application mapping problem in past years. Branch and Bound Algorithm was presented by Hu [9] where energy consumption was reduced by placement of IPs onto the NoC topologically. It was proved faster than SA algorithm.

Genetic Algorithm (GA) was also presented for efficient mapping that composed of two steps [10]. In this method the performance parameter that was considered was computation time only, other parameters like energy consumption were not bought into account. In 2016 P.K. Sahu [11] proposed DPSO Discrete Particle Swarm Optimization for 2D and 3D mesh which mimics the behaviors of birds flocking and fishes schooling, it has multiple solutions known as particles that cooperate with their neighbor particles and results in problem space. Qualities of particles are estimated through fitness value. Each solution is like a particle in the search space that has a fitness value. The bigger task is divided into small sizes which are assigned to PE according to their nature. Similar results were produced as compared with ILP mapping with less computation time.

IP mapping using NMAP algorithm was proposed by Murali [12]. In this method hop count was minimized in three steps: firstly, cores were initialized, secondly path with minimum hops was identified and in the last swapping between cores were done. Although by using this method better results were produced but it did not work well for lesser number of cores.

In 2016 Elnaz presented an algorithm named XY-ADB with the aim to improve latency [13]. In this paper an abstract graph was also extracted from core graph and mapping was done using this approach. First application is mapped in X direction and next in Y direction. Rasoul [14] proposed RASMAP for efficient mapping. In this method task with highest

priority is placed at the center of the mesh and then the rest of the tasks are placed based on the degree of proportion. Noxim simulator was used to calculate the communication cost. Network performance is improved by using a cluster based mapping approach [15]. In [16] and [17] some nature inspired algorithms were proposed like PSO and ACO that improved system performance by reducing latency and communication cost. A heuristic algorithm named MOCA was proposed for mesh architectures having low energy requirements [18]. In comparison with NMAP, MOCA has lesser complexity.

In Onyx (constructive heuristic algorithm) [19] lozenge shape approach is used by mapping first core in the center and then the remaining cores are placed at 1-2 hop distance. In CastNet [20] initial mapping is done using overall and average communication bandwidth and the rest of cores are mapped based on priority. Symmetric groups are created to select the first core for mapping the task. The first task is mapped after which the remaining tasks are mapped by calculating the communication time with the mapped task. And remaining cores are selected by calculating the communication cost. VOPD, MPEG-4 benchmarks are used.

In 2018 C.H Huang et al [21] proposed the Hierarchical and Dependency Aware (HAD) technique of mapping tasks. The bottom left core of NoC architecture is selected and given the name of Global Manager (GM) which executes HAD algorithm. In the first step the number of available cores for mapping is related with the task transition and in the next step cores placement is decided through the Mean occupied Position (MP). This algorithm reduces latency and energy consumption in comparison with FF and region-based mapping techniques. In [18], V. Tsoutsouras et al. presented distributed task mapping (DRTRM), based on controllers and local managers. It uses the cores known as manager, initial and controller. The initial core picks cores for mapping, resources are managed through the manager core, and the activity of the platform is controlled through the controller core. It results in reducing the communication overhead.

SBMAP algorithm was proposed by Sarzamin where energy consumption and system complexity were reduced by dividing the application into different segments. The results improved the energy consumption upto 53% [22]. Chicken swarm optimization (SCSO) used a clustering method for cognitive base and reduced the overall power consumption [23].

In [24], an improved ant colony mapping technique is proposed, at first tasks are divided based on their task size. The substructure of NoC maps the task when the size of task is lesser otherwise they are arranged in different clusters and then the tasks are divided according to their dependencies if they are using the same resource node, then algorithm is applied to the resource node for task mapping. It reduces the overall latency of the NoC network, along-with communication power.

In [25], a new strategy named Fibonacci tree Optimization (FTOS) is proposed to cater the problem of task scheduling in wireless sensor based network. It outperforms in term of energy consumption as compared to differential evolution (DE), PSO and Artificial Bee Colony (ABC). In [26], authors presented a reliability based technique. The proposed graph presented is further separated into its sub-graphs, to lessen the transmission flow. As a product, traffic within every graph increases while communication flow is reduced between the sub-graphs, this technique efficiently routes redundant communication packets through the non-uniform traffic distribution across network channels. Table 1 summarizes some of the related works.

III. HUNGER GAMES SEARCH ALGORITHM (HGSA)

This section discusses the main vision behind HGS algorithm's search logic. Afterwards the proposed algorithm and its mathematical model would be presented.

A. *Logic of Food Search*

Animals have a close interaction with their surroundings. Their sense organs help them to extract useful information from the environment and in making beneficial decisions. One of the stimulus which act as the reason behind the actions and behaviors of animal kingdom is Hunger. Scientists agree on this fact that hunger for food or drink make the living condition of animals more stable as it is the main driving force behind their daily activities involving searching for food etc.

Like humans, social life in animals also help them in finding sources of food and getting rid of predators. Their chances of staying alive is also

increased due to the collaboration between them that exists naturally. According to the natural law of evolution, animals that are in a good state of health can get food sources to a greater degree and also have longer life as compared to unhealthy ones. We can call this law as *hunger games*, in which any incorrect decision made by an individual can pave the way for the loss of one's life. Thus, at times when food source is bounded, a rational game is played among hungry animals in getting food and conquer the state of play. Hence the reason behind an individual's success depends on the mobility of species and the rational decisions that were made.

IV. Problem Formulation

This section contains key definitions regarding mapping applications on 2D mesh. In on-chip networks an interdependent network of tasks represents an application (Application Graph). These tasks are then scheduled on a network of available cores via scheduler (Topology Graph). Lastly an effective algorithm is utilized to map the topology graph on the NoC architecture.

A. Definition I: Application Graph (AG)

An application graph (AG) is a directed weighted graph

$$AG = G_1 (C, L)$$

where each vertex of the graph indicates cores or tasks of the given application i.e, $C = \{c_1, c_2, c_3, \ldots, c_n\}$. L indicates the links or edges representing the communication volume between cores such that $L = \{l_{i,j} \mid i, j = 1, 2, 3, \ldots, n\}$. Here n is the total number of cores in the topological network.

B. Definition II: Topology Graph (TG)

A network topology graph or TG is defined as a directed graph

$$TG = G_2 (N, E)$$

Where N belongs to n_i is the set of nodes/tiles that are attached to the PEs of the NoC topology i.e. $N = \{n_1, n_2, n_3, \ldots, n_n\}$. The shortest communication path between the adjacent tiles is symbolized by the set of directed edges $e_{i,j}$ belongs to E such that $\{i, j = 1, 2, 3, \ldots\}$. $V_{i,j}$ shows the communication volume or weight between the tiles (n_i, n_j).

C. Definition III: NoC Application Mapping

Mapping applications on on-chip networks is a 1-1 function F: $G_1 \rightarrow G_2$ such that every task is mapped efficiently to a node of topology graph if and only if tasks/cores of application graph are less than or equal to the nodes in TG

$$|C| \leq |N|$$

Table 1. Related Works

Mapping Algorithm	Benchmarks	Optimization goal	Result Comparison	Areas of improvement
[5]	MPEG-4, VOPD, MP3,H.263	Reduction in Power Consumption	Communication Cost	Increased Simulation time
[12]	MPEG4, DSD, VOPD, MWD, PIP	Reduction in Communication cost and bandwidth	Communication Cost	Not Optimized
[28]	PIP,MPEG-4, VOPD	Enhanced throughput, Communication cost, and latency	GA, NMAP	Convergence rate is slower
[29]	DVOPD, MWD, VOPD	Reduction in Power Consumption	PSO, GA,	Enhanced Comm. Cost, and slower rate of convergence
[22]	PIP, 80211arx MMS, VOPD	Reduction in Communication cost and consumption of power	Random NMAP, and BB	Not Optimized
[20]	MWD, VOPD, MPEG4, MP3enc/dec, H.263	Enhanced energy and Communication cost	GA, ILP, Random, SA	Not Optimized
[23]	MWD,VOPD, MPEG-4, 263eMP3d	Reduction in Communication cost	ACO, PSO, SA, GA,	Convergence rate is slow

V. PERFORMANCE EVALUATION MODELS

For the calculation of performance parameters following mathematical models are used in this work [27].

A. *Bit Energy Model*

Bit energy is calculated by:

$$En_{bit} = E_{sw} + E_{lk} \quad (1)$$

En_{bit} is the total energy of sending one bit from source to sink including energy of switch (E_{sw}) and energy of links (E_{lk}). Avg energy is given by:

$$En_{t(i.j)} = X_m \times E_{sw} + (X_m - 1) \times E_{lk} \quad (2)$$

Where the Manhattan distance is given by X_m from the source node (s_i, s_j) to sink (d_i, d_j) and its expression is:

$$X_m = |s_i - d_i| + |s_j - d_j| \quad (3)$$

Total consumption of energy in sending one-bit data is given by:

$$En_{tot} = \sum_{i.j}^{X_m} (En_{ti,tj} \times Bnd_{ti,tj}) \quad (4)$$

Here Bnd denote the bandwidth tile from t_i to t_j.

We can calculate the total Communication cost (C.Cost) using the following expression:

$$C.Cost = \sum_{i.j}^{X_m} (En_{ti,tj} \times X_m) \quad (5)$$

By means of this model, each mapping will have its own cost and energy solution, but our basic aim is to minimize the entire network's energy and cost. We have also used this model for our research solution.

B. *CMOS Cell Library Model*

By using CMOS Cell Library Model, avg latency and power overhead of NoC architecture is estimated by using data from standard cell libraries of CMOS. The expression of average latency is given by:

$$Lt_{avg} = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{N_i} \sum_{j=1}^{N} Lt_{i,j} \quad (6)$$

Where $Lt_{i,j}$ denote the *jth* packet latency, N_i denote the quantity of packets that are being received by processor, time for warm -up is represented by *i* and the total number of processors is specified by *N*.

The average throughput of NoC architecture is expressed as:

$$Th_{avg} = \frac{1}{(T_{sim} - T_{wrm})N} \sum_{i=1}^{N} N_i \quad (7)$$

Here T_{sim} and T_{wrm} symbolize the simulation time and time for warm-up in the NoC platform respectively. Average power consumed by NoC architecture is calculated using:

$$P_{avg} = \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{N} [(1 - \alpha_{i,j}) P_{inact,j} + \alpha_{i,j} P_{act,j}] \quad (8)$$

Where $P_{act,j}$ and $P_{inact,j}$ is the active and inactive post layout power of j component. The factor $\alpha_{i,j}$ represents the percentage of component *j* in router *i*. In the end the average consumed energy by packets in NoC is expressed as:

$$E_{pkt} = \frac{(T_{sim} - T_{wrm})}{N N_{pkt}} \sum_{i=1}^{N} \sum_{j=1}^{N} [(1 - \alpha_{i,j}) P_{inact,j} + \alpha_{i,j} P_{act,j}] \quad (9)$$

Where N_{pkt} denotes the total amount of packets that pass through the NoC architecture.

VI. PERFORMANCE EVALUATION MODELS

The suggested algorithm HGSA accepts Application graph, TG, and network graph, NG, that serve as an input and gives out an effective mapping of the desired tasks on the bi-dimensional NoC topological structure as an output.

A. *Parameter Settings*

First of all, the initial population of the hungry animals is randomly generated at time = 0 seconds using the clustered initial mapping technique and communication weight between the tiles of the task graph. Afterwards various parameters of HGSA are initialized in the beginning so that the proposed iterative method is controlled. These input parameters include N (total number of individuals), *Tmax* (number of maximum iterations), *Hunger(sum)* (sum of hungry feelings of each individual) and $\underset{R_C}{\rightarrow}$ (range controller) etc. Task

graphs of standard benchmarks along-with the communication bandwidth between the associated cores serves as the input whereas communication cost (calculated using eq. 5) denoted by "*C.Cost"* behaves as the fitness function under consideration. In each iteration, location of individuals (L_{best}) in the search space is updated as long as the solution with best fitness value is obtained. We get optimized mapping solution in the end for a certain application or task graph.

B. Mathematical Model of Algorithm

This sub-section presents, in detail, the mathematical model of the proposed HGSA algorithm. Here method of approaching food of hungry animals is described and how hunger plays an important role in searching for food.

a) Approach Food

At the time of hunting or searching for food, usually most of the animals collaborate with each-other, but there also are some animals that don't uphold.

An individual's behavior during foraging and its collegial communication is represented by the following *central equation*. It also serves as the basic game instruction for the proposed HGS algorithm.

$$\overrightarrow{X(t+1)} = \begin{cases} Game_1{:}\overrightarrow{Loc(t)}.\left(1+rndn(1)\right), & n_1 < l \\ Game_2{:}\overrightarrow{Wt_1}.\overrightarrow{Loc_b} + \overrightarrow{R_C}.\overrightarrow{Wt_2}\left|\overrightarrow{Loc_b} - \overrightarrow{Loc(t)}\right|, & n_1 > l, n_2 > V_c \\ Game_2{:}\overrightarrow{Wt_1}.\overrightarrow{Loc_b} - \overrightarrow{R_C}.\overrightarrow{Wt_2}\left|\overrightarrow{Loc_b} - \overrightarrow{Loc(t)}\right|, & n_1 > l, n_2 < V_c \end{cases} \quad (10)$$

Where

$\overrightarrow{R_C}$ is within the range of [−β, β] and defines the range of activity;

$n1$, $n2$ are random numbers, within the range of [0,1];

$rndn$ (1) is a random number which fulfills normal distribution;

t = number of current iteration;

$Wt1$, $Wt2$ = weights of respective hunger;

$Loc\mathit{b}$ = best individual's location of the current iteration;

$Loc(t)$ = location of each individual;

l= a parameter used to enhance the algorithm's performance (value to be discussed during setting of parameter)

$Loc(t) \cdot (1 + randn\ (1))$ shows the behavior of an individual during the random search of food at current location;

$|\mathrm{Loc}b - \mathrm{Loc}(t)|$ represents the activity range of an individual at the present location and time. If we multiply it by weight Wt2, it shows that activity range is affected by hunger.

As we know that when an individual has satiate its hunger, so it will not search for food anymore. In order to bound the activity range a range controller parameter $\overrightarrow{R_C}$ has been inserted whose value is progressively abridged to 0. Addition or deduction of the activity range that is centered on $W\mathrm{t}1 \cdot \mathrm{Loc}b$ signifies that the existing individual get information from its companions when it arrives at the location where food is situated, and then begin their food search again at the existing location after the foodstuff acquirement. If an error arises in getting to the correct position, then a parameter $W\mathrm{t}1$ is used to cater for it. The interpretation for the variation control parameter V_C is described as follows:

$$V_C = \mathrm{sech}(|Fit_{all}(i) - BFit|) \quad (11)$$

where $i \in 1,2,3 \ldots, n$,

sech= secant hyperbolic function defined as ($\mathrm{sech}(x) = 2/e^{x} + e^{-x}$).

$Fit_{all}(i)$ = fitness value of every single individual;

$BF\mathrm{it}$ = value of best fitness found so far during the existing iterative process;

The formula for ranging controller $\overrightarrow{R_C}$ is as follows:

$$\overrightarrow{R_C} = shrink\ (2 \times rand - 1) \quad (12)$$

$$shrink = 2 \times (1 - \frac{t}{T_{max}}) \quad (13)$$

where $rand$ = a random number that is within the series of (0,1);

$T\mathrm{max}$ = number of maximum iterations.

From the central equation we can deduce that, based on the grouping of food source points the directions of search can be categorized into two basis:

- **Search based on *Loc***: In the first game instruction, an individual want to search for food on

its own without the cooperation of its peers. It lacks the teamwork spirit and is not dependent on anyone.

➢ **Search based on *Locb***: In the second game instruction, an individual searches food based on the best location (*Locb*) and is dependent on parameters like $\underset{R_C}{\rightarrow}$,*Wt1* and *Wt2*. These parameters allow the calculation of an individual's position accurately and also depicts cooperation between different species during food search.

By following the rules of the central HGS equation, it is easier for a specie to find food locations far-off or close to the optimal solution. It gives assurance, to some extent, to search for all possible locations that are within the borderline of solution space. This idea is also applicable to search areas with higher dimensions.

b) Hunger Role

Here a mathematical model is proposed to model the starvation characteristics of hungry individuals. From the first equation, value of *Wt1* and *Wt2* can be expressed as follows:

$$\overrightarrow{Wt_1(i)} = \begin{cases} hunger(i).\frac{N}{hunger_{sum}} \times r_4, & r_3 < l \\ 1, & r_3 > l \end{cases} \quad (14)$$

$$\overrightarrow{Wt_2(i)} = (1 - e^{(-|hunger(i) - hunger_{sum}|)}) \times r_5 \times 2 \quad (15)$$

where

r3, r4 and r5 = random numbers that are within the range of [0,1];

hunger(i) = hunger of every individual;

hunger(sum) = sum of hungry feelings of every individual;

N = total number of individuals.

From the above equation, the value of an individual's hunger is given as:

$$hunger(i) = \begin{cases} 0, & Fit_{all}(i) == BFit \\ hunger(i) + H, & Fit_{all}(i) != BFit \end{cases} \quad (16)$$

where $Fit_{all}(i)$ and *BFit* is defined above. An additional parameter named *hunger sensation* (H) is introduced based on the value of original hunger. Initially the value of hunger of best individual was set as 0. Hence, we can say that every individual will have a different value of H defined as:

$$TH = \frac{Fit_{all}(i) - BFit}{WFit - BFit} \times 2 \times r_6 \times (UB - LB) \quad (17)$$

$$H = \begin{cases} LH \times (1 + r), & TH < LH \\ TH, & TH \geq LH \end{cases} \quad (18)$$

Where

R6 = random number that is within the range of [0,1];

WFit = value of worst fitness obtained so far during the iteration process;

LB, UB = lower limit/bound & upper limit/bound of the search space respectively; [0,1];

Their values can be controlled to make the algorithm perform in a better way. The flow diagram of the proposed algorithm HGSA is represented in Figure 2.

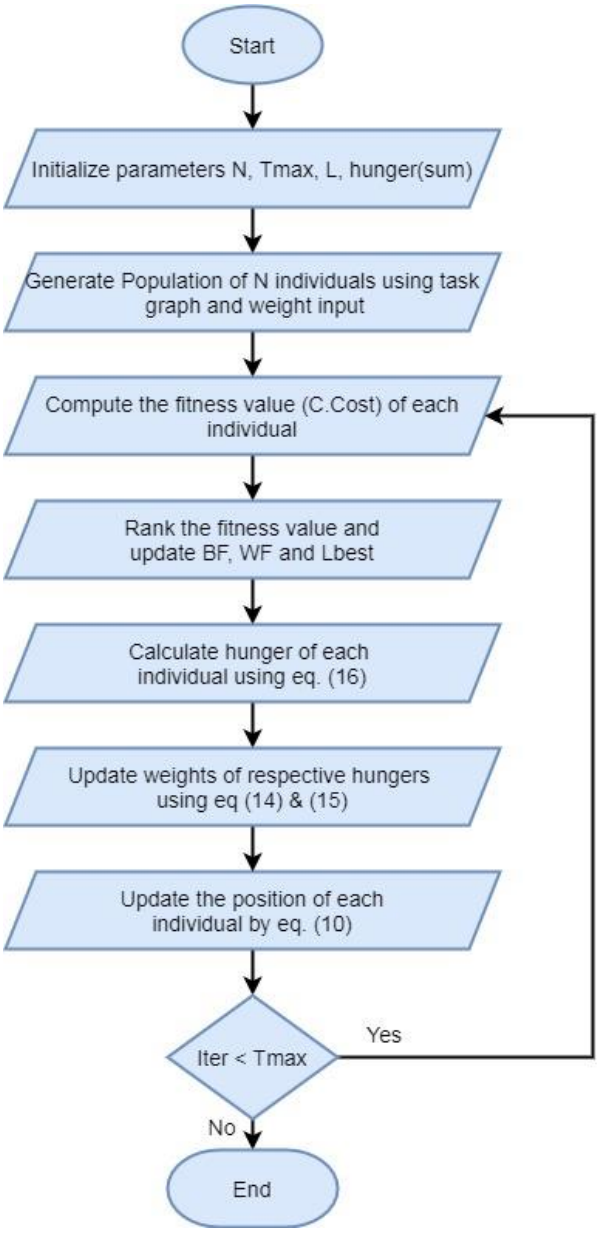

Fig. 2 – Flow Chart of Hunger Games Search Algorithm

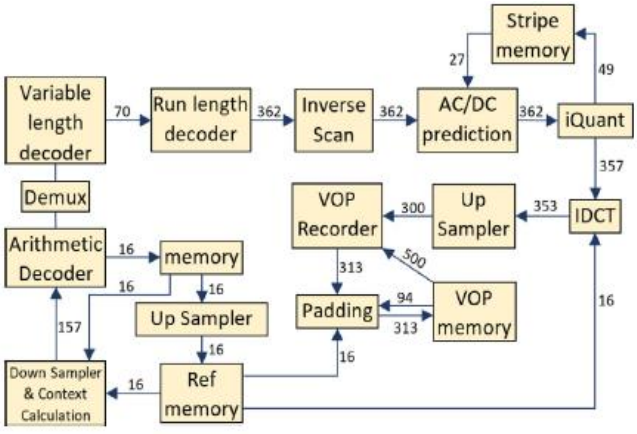

Fig. 3 – VOPD Architectural Representation

VII. RESULTS AND DISCUSSION

In this section we present the results of mapping using proposed HGSA for 2D NoC for standard six embedded application graphs, e.g. MWD, VOPD, MPEG-4, MP3enc MP3dec, and 263enc MP3dec etc. These standard graphs are depicted in the Table 2. The size of the considered network is similar i.e. 4*4 for all applications taken into consideration. In order to do a just evaluation with former contemporary architectures, the size of the network is identical. There are 16 sub-tasks in a VOPD embedded application. Memory, down sampler, quantize, etc are part of sub tasks. The rate of communication between these tasks is done in MB/s which is represented on the respective ends. Figure 3 represents the VOPD architecture diagram while the task graphs of MPEG-4 and VOPD is shown in figure 4. The sub-tasks of VOPD application can easily be mapped on a 4*4 mesh topological network.

Among the six considered performance matrices, the results for communication cost and simulation time are obtained through python whereas remaining parameters' results are obtained through NoCTweak simulator.

Table 2 - Details of Standard embedded application graphs with mesh sizes

Benchmark used	**Description**	**Number of nodes**	**Number of edges**	**Size of mesh**
VOPD	Video Object Plane Decoder	16	21	4X4
MPEG-4	MPEG Decoder	12	26	4X4
MWD	Multi Window Display	12	13	4X4
MP3enc MP3dec	MP3 Encoder	13	14	4X4
263enc MP3dec	263 Encoder	12	12	4X4
263enc MP3dec	263 Encoder	14	15	4X4
PIP	Picture in Picture	8	8	3X3

A. Experimental Setup

To estimate the performance of the presented mapping algorithm HGSA, six standard embedded applications are used and several experimentations

were performed. The presented algorithm was tested for 2D NoC with several state of the art algorithms such as ILP, PSO, ACO, GA, BA, SBMAP, CastNet and SCSO. Python was used for writing code for the proposed HGSA algorithm and then rest of the part was executed on NoCTweak Simulator. NoCTweak is an open source type hardware simulator that is built on SystemC and uses VHDL language. It's simulation environment is highly flexible as it allows settings of various network parameters. All trials were carried out on Intel Core i7 PC having 16GB RAM, 1TB ROM, 64-bit Operating system and 2.50 GHz processor. The Table 3 provides the default parameter settings that were used for a uniform simulation. This allows a just comparison with the already existing state-of-the-art work.

Table 3 - Setting of Simulation Environment

Type of Network	**Torus and 2-D Mesh**
Type of platform	Embedded / Synthetic traffic
Embedded application	MPEG, MWD, VOPD, MP3 enc, MP3 dec
Type of Packet delivery	Without ACK
Sending ACK policy	Optimally send ACK
Distribution of Packet	Exponential
Length of fixed packet	10 (flits)
Rate of Flit injection	0.1,0.2,0.3,….1.0 (flits/cycle/node)
Mapping algorithm	Random/ Proposed
Form of router	Wormhole pipeline
Algorithm used for routing	XY dimension
Selection of output channel	XY-Ordered
Size of buffer	8
Type of pipeline	8
Stages of Pipeline	4(RC, VA, SA, XB)
Input applied voltage	1(V)
Frequency of input clock	1000 MHZ
Frequency of operating clock	1000 MHZ
Time for warm up	20000 cycles

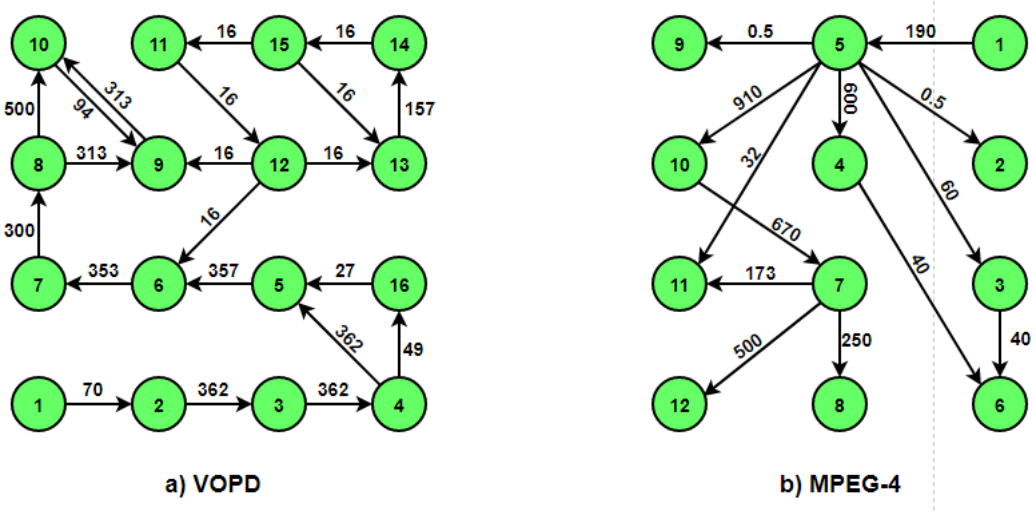

Fig. 4 Tasks graphs of (a) VOPD (b) MPEG-4

Table 4: Communication Cost Comparison

Communication Cost (BW * Xm)MB/s						
Mapping Algorithm	**VOPD**	**% Saving**	**MPEG-4**	**% Saving**	**MWD**	**% Saving**
ILP	4119	0.00%	3567	0.00%	1120	0.00%
ACO	-	-	3633	1.9%	-	-
PSO	4119	0.00%	3567	0.00%	1120	0.00%
GA	4218	2.4%	3772	5.74%	1321	17.94%
BA	4119	0.00%	3567	0.00%	1120	0.00%
CastNet	4135	0.39%	3852	7.99%	1280	14.285%
SBMAP	4125	0.15%	4474	25.427%	1280	14.285%
Proposed Algorithm	**4119**		**3567**		**1120**	
Mapping Algorithm	**MP3enc MP3dec**	**% Saving**	**263enc MP3dec**	**% Saving**	**263dec MP3dec**	**% Saving**
ILP	17.021	0.00%	230.407	0.00%	19.823	0.00%
PSO	17.021	0.00%	230.407	0.00%	19.823	0.00%
ACO	17.231	1.23%	-	-	-	-
BA	17.834	4.78%	231.450	0.45%	19.936	0.57%
GA	17.133	0.66%	230.698	0.13%	19.911	0.44%
CastNet	17.021	0.00%	230.407	0.00%	19.823	0.00%
SBMAP	-	-	230.432	0.011%	-	-
Proposed Algorithm	**17.021**		**230.407**		**1120**	

Table 5: Power Analysis

Power (mW)						
Mapping Algorithm	**VOPD**	**MPEG-4**	**MWD**	**MP3enc MP3dec**	**263enc MP3dec**	**263dec MP3dec**
ILP	**1528**	**1137**	**1012**	**1228**	**1286**	**1211**
ACO	**1920**	**1423**	**1218**	**1498**	**1599**	**1738**
PSO	**1841**	**1357**	**1112**	**1507**	**1445**	**1561**
GA	**1843**	**1356**	**1109**	**1507**	**1445**	**1561**
BA	**1634**	**1247**	**1110**	**1486**	**1323**	**1313**
CastNet	**1945**	**1590**				
SBMAP	**1650.9**	**1235**				
Proposed Algorithm	**1510.1**	**1209**	**1014**	**1228**	**1281**	**1146**

B. Performance Analysis

In this section of the paper the overall performance of the presented algorithm HGSA over six benchmarks (described in Table 2) is presented along-with their comparison and percentage deviation with already existing algorithms.

a) Communication Cost

This sub-section deals with the overall performance of the presented algorithm HGSA regarding communication cost calculation and the results obtained are compared against already existing mapping algorithms. Table 4 shows the valuation of communication cost for 2D NoC with benchmarks (VOPD, MWD, MPEG4, MP3encMP3dec, 263decMP3dec and 263encMP3dec). After analysing the results, it is noted that HGSA showed 1.5%, 4.5%, 1.9%, 7.5% and 9.8% improvement in Communication cost over ACO, GA, BA, CastNet and SBMAP algorithms respectively.

b) Average Power Analysis

Power overhead comparative analysis of proposed HGSA-based method with ILP, ACO, GA, SBMAP, CastNet, and PSO based mapping algorithms is presented in Table 5. Results specify that the proposed HGSA approach relatively consumes less power than conventional algorithms for various embedded standard applications. Figure 4 shows power savings over various algorithms. The results conclude that the proposed algorithm saves power on average about 2.27% over ILP, 27.2% over ACO, 19.23% over PSO, 19.1% over GA and 9.9% over BA.

c) Computation/Simulation Time

The analysis of the proposed mapping approach HGSA for computation time in comparison to other already existing state of the art mapping algorithms and results are presented in the Table 6. Table 7 gives percentage improvement over several algorithms. It can be seen that the proposed algorithm uses less simulation time in comparison to already existing algorithms such that it improved approx. 86% over PSO, 42.8% over BA, 85.3% over GA and 1.4% over SCSO.

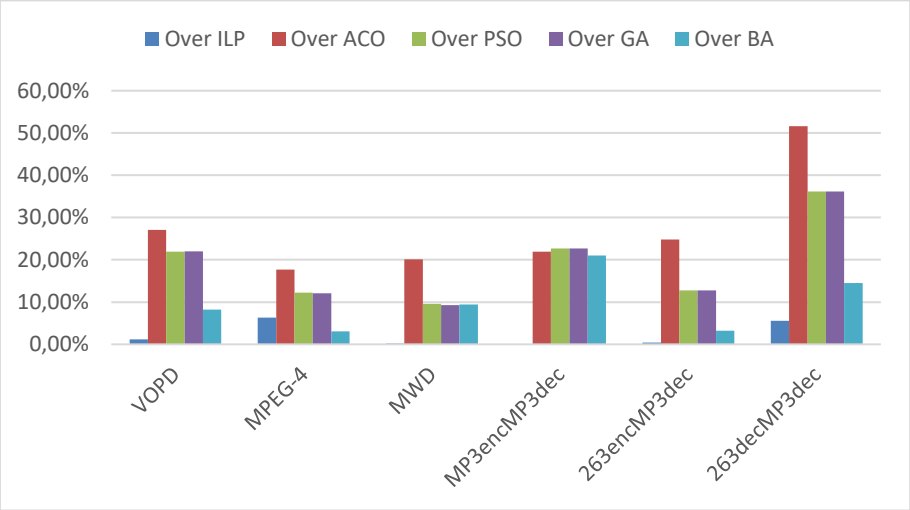

Fig. 5 Power Savings %

Table 6: Comparison of Computation Time

CPU Time (seconds)						
Mapping Algorithm	**MPEG-4**	**VOPD**	**MWD**	**MP3enc MP3dec**	**263enc MP3dec**	**263dec MP3dec**
ILP	22.34	4679.341	210.021	1435.012	193.035	4897.21
ACO	18.652			1196.856		
PSO	3.465	3.785	3.432	3.194	3.185	3.188
GA	3.234	3.925	3.42	3.194	3.185	3.174
BA	2.925	2.231	2.894	2.653	2.345	2.35
SCSO	2.010	2.231	1.996	1.785	1.527	1.511
Proposed Algorithm	**2.010**	**2.215**	**1.945**	**1.771**	**1.525**	**1.472**

Table 7: Percentage improvement in CPU Time

HGSA Savings % in CPU Time				
	Over PSO	**Over GA**	**Over BA**	**Over SCSO**
MPEG4	72.3%	60.8%	45.5%	0.00%
VOPD	70.8 %	77.2 %	0.72%	0.72%
MWD	76.4 %	75.8 %	48.7%	2.62%
263decMP3dec	109.6 %	111.4 %	59.3%	2.64%
263encMP3dec	106.7 %	106.7 %	53.1%	0.13%
MP3encMP3dec	80.3%	80.3%	49.7%	0.79%

d) Latency Analysis

Latency is defined as the amount of time elapsed between the launch of a packet from the source node to the time when it reaches the destination or sink node. To check the performance of the network the Analysis of latency on the network is also performed using HGSA. It was done through different traffic pattern on the mesh topology. Traffics are tornado and random. Uniform pattern distribute the traffic in a uniform manner which help to balance the load. The analysis of mesh based NoC network is via NoCTweak simulator using xy-routing. Figure 6 shows the performance of HGSA for network latency and shows that HGSA has performed better as compared to other mapping algorithms. HGSA outperformed GA by 16.6%, PSO by 12.1%, BA by 8.8% and SCSO by 4.32%.

e) Energy and Throughput Analysis

Energy and throughput comparison of discussed embedded application graphs is shown in Table 8. It can be determined from the obtained results that our proposed HGSA has performed relatively better in network energy in comparison to some existing heuristic algorithms. In case of energy HGSA improved ILP, GA, CastNet and SBMAP by 1.47%, 1.71%, 9.19% and 0.1% respectively. Analysing power overhead gives 6.2% less power consumption over ILP, GA and CastNet.

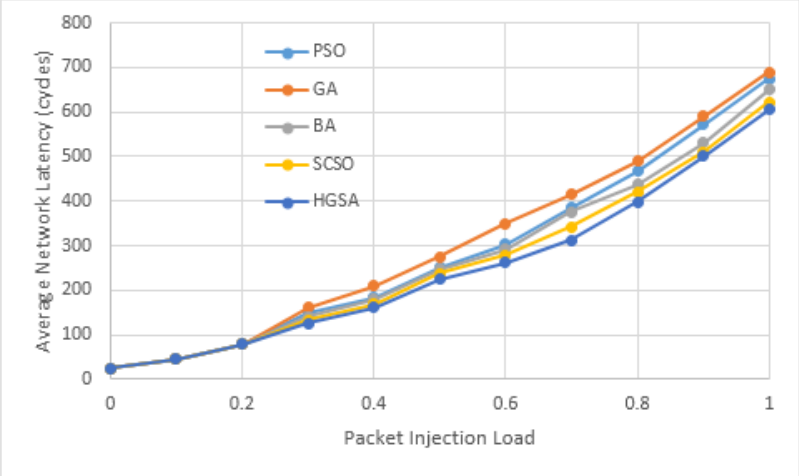

Fig. 6 Average Latency for Uniform Traffic

Table 8: Throughput and Energy Analysis

	Energy (pJ/packet)		**Throughput (packets/cycle)**	
Mapping Algorithm	**VOPD**	**MPEG\4**	**VOPD**	**MPEG4**
ILP	264.608	179.543	0.05	0.027
GA	268.324	177.127	0.05	0.027
CastNet	264.332	203.462	0.05	0.027
SBMAP	264.042	-	0.047	-
Proposed Algorithm	**263.781**	**174.86**	**0.047**	**0.027**

VIII. CONCLUSION

In this paper, a novel metaheuristic mapping algorithm was presented to optimally map several embedded applications on 2D mesh based on-chip networks. The proposed algorithm HGSA is based on behaviour of animals during food search when they get hungry. In this algorithm, our basic aim is to make improvement in communication cost which is the objective function that in turn improves the efficiency scales like power consumption, latency and throughput of the network etc. We compared the results of the proposed algorithm among embedded application graphs to already existing application mapping algorithms. Results of HGSA given in above section are compared with ILP, BA, PSO, ACO, SCSO, CastNet and SBMAP and our proposed algorithm outperformed in all parameters including power dissipation where our algorithm shows improvement of 2.27% over ILP, 19.1% over GA, 19.23% over PSO respectively and also proposed algorithm takes less computation time as compared to other algorithms. Two types of traffics tornado and random traffic pattern were used for latency and HGSA shows improvement as compared to earlier algorithms.

The proposed results can be utilized further for 3D topologies also. For further improvements some other algorithms can also be merged in a hybrid manner to enhance the efficiency of the network.

ACKNOWLEDGMENT

I would like to express my sincere gratitude and appreciation to University of Engineering and Technology Taxila, Pakistan. I would also like to thank Prof. Dr. Muhammad Iram Baig for his assistance in writing this article.

REFERENCES

[1] K. Tatas, K. Siozios, D. Soudris, and A. Jantsch, *Designing 2D and 3D Network-on-Chip Architectures*. New York, NY: Springer New York, 2014. doi: 10.1007/978-1-4614-4274-5.

[2] T. Bjerregaard and S. Mahadevan, “A survey of research and practices of Network-on-chip,” *ACM Comput. Surv.*, vol. 38, no. 1, pp. 1-es, Jun. 2006, doi: 10.1145/1132952.1132953.

[3] W.-C. Tsai, Y.-C. Lan, Y.-H. Hu, and S.-J. Chen, “Networks on Chips: Structure and Design Methodologies,” *Journal of Electrical and Computer Engineering*, vol. 2012, p. e509465, Nov. 2011, doi: 10.1155/2012/509465.

[4] L. Bononi and N. Concer, “Simulation and analysis of network on chip architectures: ring, spidergon and 2D mesh,” in *Proceedings of the Design Automation & Test in Europe Conference*, Mar. 2006, vol. 2, p. 6 pp.-. doi: 10.1109/DATE.2006.243841.

[5] S. Tosun, O. Ozturk, and M. Ozen, “An ILP formulation for application mapping onto Network-on-Chips,” in *2009 International Conference on Application of Information and Communication Technologies*, Oct. 2009, pp. 1–5. doi: 10.1109/ICAICT.2009.5372524.

[6] W. Amin *et al.*, “Performance Evaluation of Application Mapping Approaches for Network-on-Chip Designs,” *IEEE Access*, vol. 8, pp. 63607–63631, 2020, doi: 10.1109/ACCESS.2020.2982675.

[7] Y. Khamayseh, W. Mardini, M. Aldwairi, and H. Mouftah, “On the Optimality of Route Selection in Grid Wireless Sensor Networks: Theory and Applications,” *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications*, vol. 11, no. 2, pp. 87–105, Jun. 2020, doi: 10.22667/JOWUA.2020.06.30.087.

[8] Y. Yang, H. Chen, A. A. Heidari, and A. H. Gandomi, “Hunger games search: Visions, conception, implementation, deep analysis, perspectives, and towards performance shifts,” *Expert Systems with Applications*, vol. 177, p. 114864, Sep. 2021, doi: 10.1016/j.eswa.2021.114864.

[9] J. Hu and R. Marculescu, “Energy-aware mapping for tile-based NoC architectures under performance

constraints," in *Proceedings of the ASP-DAC Asia and South Pacific Design Automation Conference, 2003.*, Jan. 2003, pp. 233–239. doi: 10.1109/ASPDAC.2003.1195022.

[10] T. Lei and S. Kumar, "A two-step genetic algorithm for mapping task graphs to a network on chip architecture," in *Euromicro Symposium on Digital System Design, 2003. Proceedings.*, Sep. 2003, pp. 180–187. doi: 10.1109/DSD.2003.1231923.

[11] P. K. Sahu, T. Shah, K. Manna, and S. Chattopadhyay, "Application Mapping Onto Mesh-Based Network-on-Chip Using Discrete Particle Swarm Optimization," *IEEE Trans. VLSI Syst.*, vol. 22, no. 2, pp. 300–312, Feb. 2014, doi: 10.1109/TVLSI.2013.2240708.

[12] S. Murali and G. De Micheli, "Bandwidth-constrained mapping of cores onto NoC architectures," in *Automation and Test in Europe Conference and Exhibition Proceedings Design*, Feb. 2004, vol. 2, pp. 896-901 Vol.2. doi: 10.1109/DATE.2004.1269002.

[13] E. Alikhah-Asl and M. Reshadi, "XY-axis and distance based NoC mapping (XY-ADB)," in *2016 8th International Symposium on Telecommunications (IST)*, Sep. 2016, pp. 678–683. doi: 10.1109/IS TEL .2016.7881908.

[14] R. Seidipiri, A. Patooghy, S. Afsharpour, and M. Fazeli, "RASMAP: An efficient heuristic application mapping algorithm for network-on-chips," in *2016 Eighth International Conference on Information and Knowledge Technology (IKT)*, Sep. 2016, pp. 149–155. doi: 10.1109/IKT.2016.7777757.

[15] A. Aravindhan, S. Salini, and G. Lakshminarayanan, "Cluster Based Application Mapping Strategy for 2D NoC," *Procedia Technology*, vol. 25, pp. 505–512, Jan. 2016, doi: 10.1016/j.protcy.2016.08.138.

[16] B. Chopard and M. Tomassini, "Particle Swarm Optimization," in *An Introduction to Metaheuristics for Optimization*, Cham: Springer International Publishing, 2018, pp. 97–102. doi: 10.1007/978-3-319-93073-2_6.

[17] A. Colorni, M. Dorigo, and V. Maniezzo, "Distributed Optimization by Ant Colonies," presented at the Proceedings of the First European Conference on Artificial Life, Jan. 1991.

[18] K. Srinivasan and K. S. Chatha, "A technique for low energy mapping and routing in network-on-chip architectures," in *ISLPED '05. Proceedings of the 2005 International Symposium on Low Power Electronics and Design, 2005.*, Aug. 2005, pp. 387–392. doi: 10.1145/1077603.1077695.

[19] M. Janidarmian, A. Khademzadeh, and M. Tavanpour, "Onyx: A new heuristic bandwidth-constrained mapping of cores onto tile-based Network on Chip," *IEICE Electron. Express*, 2009, Accessed: Jan. 20, 2023. [Online]. Available: https://www.semanticscholar.org/paper/Onyx%3A-A-new-heuristic-bandwidth-constrained-mapping-Janidarmian-Khademzadeh/f9b4825e92559c1ca48ad3ccc1bcf077282dc319

[20] S. Tosun, "New heuristic algorithms for energy aware application mapping and routing on mesh-based NoCs," *Journal of Systems Architecture*, vol. 57, no. 1, pp. 69–78, Jan. 2011, doi: 10.1016/j.sysarc.2010.10.001.

[21] C.-H. Huang, C.-Y. Chen, and H.-Y. Huang, "Hierarchical and Dependency-Aware Task Mapping for NoC-based Systems," in *2018 11th International Workshop on Network on Chip Architectures (NoCArc)*, Oct. 2018, pp. 1–6. doi: 10.1109/NOCARC.2018.8541206.

[22] S. Khan, S. Anjum, U. A. Gulzari, T. Umer, and B.-S. Kim, "Bandwidth-Constrained Multi-Objective Segmented Brute-Force Algorithm for Efficient Mapping of Embedded Applications on NoC Architecture," *IEEE Access*, vol. 6, pp. 11242–11254, 2018, doi: 10.1109/ACCESS.2017.2778340.

[23] A. Alagarsamy, L. Gopalakrishnan, S. Mahilmaran, and S.-B. Ko, "A Self-Adaptive Mapping Approach for Network on Chip With Low Power Consumption," *IEEE Access*, vol. 7, pp. 84066–84081, 2019, doi: 10.1109/ACCESS.2019.2925381.

[24] J. Fang, T. Yu, and Z. Wei, "Improved Ant Colony Algorithm Based on Task Scale in Network on Chip (NoC) Mapping," *Electronics*, vol. 9, no. 1, Art. no. 1, Jan. 2020, doi: 10.3390/electronics9010006.

[25] L. Wu and H. Cai, "Energy-Efficient Adaptive Sensing Scheduling in Wireless Sensor Networks Using Fibonacci Tree Optimization Algorithm," *Sensors*, vol. 21, no. 15, Art. no. 15, Jan. 2021, doi: 10.3390/s21155002.

[26] A. Patooghy, H. Tabkhi, and S. G. Miremadi, "RMAP: A Reliability-Aware Application Mapping for Network-on-Chips," in *2010 Third International Conference on Dependability*, Jul. 2010, pp. 112–117. doi: 10.1109/DEPEND.2010.25.

[27] A. Tran and B. Baas, "NoCTweak: a Highly Parameterizable Simulator for Early Exploration of Performance and Energy of Networks On-Chip," 2012. Accessed: Jan. 20, 2023. [Online]. Available: https://www.semanticscholar.org/paper/NoCTweak%3A-a-Highly-Parameterizable-Simulator-for-of-Tran-Baas/af729f08ee299a3430306abf0f855f8ffb6033eb

[28] P. K. Sahu, P. Venkatesh, S. Gollapalli, and S. Chattopadhyay, "Application Mapping onto Mesh Structured Network-on-Chip Using Particle Swarm Optimization," in *2011 IEEE Computer Society Annual Symposium on VLSI*, Chennai, India, Jul. 2011, pp. 335–336. doi: 10.1109/ISVLSI.2011.21.

[29] J. Li *et al.*, "Bat Algorithm Based Low Power Mapping Methods for 3D Network-on-Chips," in *Theoretical Computer Science*, vol. 768, D. Du, L. Li, E. Zhu, and K. He, Eds. Singapore: Springer Singapore, 2017, pp. 277–295. doi: 10.1007/978-981-10-6893-5_21.