A Performance Constrained Metaheuristic Algorithm for 2D Mesh Based NoC

Aneeqa Khurshid\textsuperscript{1}, Muhammad Iram Baig \textsuperscript{2}

\textsuperscript{1}Department of Electrical Engineering, University of Engineering and Technology Taxila, Pakistan
\textsuperscript{2}Department of Electrical Engineering, University of Engineering and Technology Taxila, Pakistan

\textsuperscript{1}(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

1. 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](image)

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 over average and overall 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 \( L = \{ l_{i,j} | i, j = 1, 2, 3, \ldots, n \} \). 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 \( \{ i, j \in \{1, 2, 3, \ldots\} \} \). \( V_{i,j} \) 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

<table>
<thead>
<tr>
<th>Mapping Algorithm</th>
<th>Benchmarks</th>
<th>Optimization goal with</th>
<th>Result Comparison</th>
<th>Areas of improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td>[12]</td>
<td>MPEG4, DSD, VOPD, MWD, PIP</td>
<td>Reduction in Communication cost and bandwidth</td>
<td>Communication Cost</td>
<td>Not Optimized</td>
</tr>
<tr>
<td>[28]</td>
<td>PIP, MPEG-4, VOPD</td>
<td>Enhanced throughput, Communication cost, and latency</td>
<td>GA, NMAP</td>
<td>Convergence rate is slower</td>
</tr>
<tr>
<td>[29]</td>
<td>DVOPD, MWD, VOPD</td>
<td>Reduction in Power Consumption</td>
<td>PSO, GA,</td>
<td>Enhanced Comm. Cost. and slower rate of convergence</td>
</tr>
<tr>
<td>[22]</td>
<td>PIP, 80211arx MMS, VOPD</td>
<td>Reduction in Communication cost and consumption of power</td>
<td>Random NMAP, and BB</td>
<td>Not Optimized</td>
</tr>
<tr>
<td>[23]</td>
<td>MWD, VOPD, MPEG-4, 263eMP3d</td>
<td>Reduction in Communication cost</td>
<td>ACO, PSO, SA, GA,</td>
<td>Convergence rate is slow</td>
</tr>
</tbody>
</table>
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:

\[ E_{\text{bit}} = E_{\text{sw}} + E_{\text{lk}} \]  

(1)

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

\[ E_{\text{t}(i,j)} = X_m \times E_{\text{sw}} + (X_m - 1) \times E_{\text{lk}} \]  

(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:

\[ X_m = |s_i - d_i| + |s_j - d_j| \]  

(3)

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

\[ E_{\text{tot}} = \sum_{i,j} X_m E_{\text{t}(i,j)} \times \text{Bnd}_{t_i,t_j} \]  

(4)

Here \( \text{Bnd} \) denote the bandwidth tile from \( t_i \) to \( t_j \).

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

\[ \text{C.Cost} = \sum_{i,j} X_m E_{\text{t}(i,j)} \times X_m \]  

(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:

\[ L_{t_{\text{avg}}} = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{N_l} \sum_{j=1}^{N} L_{t_{i,j}} \]  

(6)

Where \( L_{t_{i,j}} \) denote the \( j \)th packet latency, \( N_l \) 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:

\[ T_{\text{avg}} = \frac{1}{(T_{\text{sim}}-T_{\text{wrm}})N} \sum_{i=1}^{N} N_l \]  

(7)

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

\[ P_{\text{avg}} = \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{N} \left( (1 - \alpha_{i,j})P_{\text{act},j} + \alpha_{i,j}P_{\text{act},j} \right) \]  

(8)

Where \( P_{\text{act},j} \) and \( P_{\text{inact},j} \) 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:

\[ E_{\text{pkt}} = \frac{\sum_{i=1}^{N} \sum_{j=1}^{N} \left( (1 - \alpha_{i,j})P_{\text{inact},j} + \alpha_{i,j}P_{\text{act},j} \right)}{N_{\text{pkt}}} \]  

(9)

Where \( N_{\text{pkt}} \) 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), \( T_{\text{max}} \) (number of maximum iterations), \( \text{Hunger(\text{sum})} \) (sum of hungry feelings of each individual) and \( \text{\( \rightarrow \)} \text{\( \text{R}_c \)} \) (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.

\[
\begin{align*}
\text{Game}_1: & \rightarrow \frac{L_{oc}(t)}{20}, (1 + \text{randn}(1)), n_1 < l, n_2 > l, n_3 \rightarrow \text{loc}_2(n_1, n_2, n_3) \\
\text{Game}_2: & \rightarrow \frac{L_{oc}(t)}{20} + \rightarrow \frac{L_{oc}(t)}{20} - \rightarrow \text{loc}_2(n_1, n_2, n_3), n_1 > l, n_2 > l, n_3 \rightarrow \text{loc}_2(n_1, n_2, n_3)
\end{align*}
\]

Where

\( \rightarrow \) is within the range of \([-\beta, \beta]\) and defines the range of activity;

\( n_1, n_2 \) are random numbers, within the range of \([0,1] \);

\( \text{randn} (1) \) is a random number which fulfills normal distribution;

\( t \) = number of current iteration;

\( Wt_1, Wt_2 \) = weights of respective hunger;

\( \text{Loc}_b \) = best individual’s location of the current iteration;

\( \text{Loc}(t) \) = location of each individual;

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

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

\[ |\text{Loc}_b - \text{Loc}(t)| \] represents the activity range of an individual at the present location and time. If we multiply it by weight \( Wt_2 \), 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 \( R_C \) has been inserted whose value is progressively abridged to 0. Addition or deduction of the activity range that is centered on \( Wt_1 \). \( \text{Loc}_b \) 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 acquisition. If an error arises in getting to the correct position, then a parameter \( Wt_1 \) is used to cater for it. The interpretation for the variation control parameter \( V_C \) is described as follows:

\[
V_C = \text{sech}(|\text{Fit}_{all}(i) - \text{BFit}|)
\] (11)

where \( i \in 1,2,3 \ldots, n \),

\( \text{sech} = \text{secant hyperbolic function defined as} \)

\( \text{sech}(x) = 2/e^x + e^{-x} \).

\( \text{Fit}_{all}(i) \) = fitness value of every single individual;

\( \text{BFit} \) = value of best fitness found so far during the existing iterative process;

The formula for ranging controller is as follows:

\[
\begin{align*}
\text{shrink}(2 \times \text{rand} - 1) & \\
\text{shrink} = 2 \times (1 - \frac{x}{T_{max}})
\end{align*}
\] (12)

(13)

where \( \text{rand} \) = a random number that is within the series of \((0,1)\);

\( T_{max} \) = 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:

\[ \text{Search based on Loc:} \] 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 \( \text{Locb} \) and is dependent on parameters like \( r_c \rightarrow W1 \) and \( W2 \). 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 \( W1 \) and \( W2 \) can be expressed as follows:

\[
Wt_1(i) = \begin{cases} 
\text{hunger}(i) \times \frac{N}{\text{hunger}_{\text{sum}}} \times r_4, & r_3 < l \\
1 & r_3 > l 
\end{cases}
\]

\[
Wt_2(i) = (1 - e^{-|\text{hunger}(i) - \text{hunger}_{\text{sum}}|}) \times r_5 \times 2
\]

where

- \( r_3, r_4 \) and \( r_5 \) = random numbers that are within the range of \([0,1]\);
- \( \text{hunger}(i) \) = hunger of every individual;
- \( \text{hunger}_{\text{sum}} \) = 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:

\[
\text{hunger}(i) = \begin{cases} 
0, & \text{Fit}_{\text{all}}(i) == \text{BFit} \\
\text{hunger}(i) + H, & \text{Fit}_{\text{all}}(i)! = \text{BFit}
\end{cases}
\]

where \( \text{Fit}_{\text{all}}(i) \) and \( \text{BFit} \) 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:

\[
TH = \frac{\text{Fit}_{\text{all}}(i) - \text{BFit}}{\text{WFit} - \text{BFit}} \times 2 \times r_6 \times (UB - LB) \quad (17)
\]

\[
H = \begin{cases} 
\text{LH} \times (1 + r), & TH < \text{LH} \\
TH, & TH \geq \text{LH}
\end{cases}
\]

Where

- \( R6 \) = random number that is within the range of \([0,1]\);
- \( \text{WFit} \) = 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.
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>
<thead>
<tr>
<th>Benchmark used</th>
<th>Description</th>
<th>Number of nodes</th>
<th>Number of edges</th>
<th>Size of mesh</th>
</tr>
</thead>
<tbody>
<tr>
<td>VOPD</td>
<td>Video Object Plane Decoder</td>
<td>16</td>
<td>21</td>
<td>4X4</td>
</tr>
<tr>
<td>MPEG-4</td>
<td>MPEG Decoder</td>
<td>12</td>
<td>26</td>
<td>4X4</td>
</tr>
<tr>
<td>MWD</td>
<td>Multi Window Display</td>
<td>12</td>
<td>13</td>
<td>4X4</td>
</tr>
<tr>
<td>MP3enc MP3dec</td>
<td>MP3 Encoder</td>
<td>13</td>
<td>14</td>
<td>4X4</td>
</tr>
<tr>
<td>263enc MP3dec</td>
<td>263 Encoder</td>
<td>12</td>
<td>12</td>
<td>4X4</td>
</tr>
<tr>
<td>263enc MP3dec</td>
<td>263 Encoder</td>
<td>14</td>
<td>15</td>
<td>4X4</td>
</tr>
<tr>
<td>PIP</td>
<td>Picture in Picture</td>
<td>8</td>
<td>8</td>
<td>3X3</td>
</tr>
</tbody>
</table>

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

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

Fig. 4 Tasks graphs of (a) VOPD (b) MPEG-4
### Table 4: Communication Cost Comparison

<table>
<thead>
<tr>
<th>Mapping Algorithm</th>
<th>VOPD</th>
<th>% Saving</th>
<th>MPEG-4</th>
<th>% Saving</th>
<th>MWD</th>
<th>% Saving</th>
</tr>
</thead>
<tbody>
<tr>
<td>ILP</td>
<td>4119</td>
<td>0.00%</td>
<td>3567</td>
<td>0.00%</td>
<td>1120</td>
<td>0.00%</td>
</tr>
<tr>
<td>ACO</td>
<td>-</td>
<td>-</td>
<td>3633</td>
<td>1.9%</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>PSO</td>
<td>4119</td>
<td>0.00%</td>
<td>3567</td>
<td>0.00%</td>
<td>1120</td>
<td>0.00%</td>
</tr>
<tr>
<td>GA</td>
<td>4218</td>
<td>2.4%</td>
<td>3772</td>
<td>5.74%</td>
<td>1321</td>
<td>17.94%</td>
</tr>
<tr>
<td>BA</td>
<td>4119</td>
<td>0.00%</td>
<td>3567</td>
<td>0.00%</td>
<td>1120</td>
<td>0.00%</td>
</tr>
<tr>
<td>CastNet</td>
<td>4135</td>
<td>0.39%</td>
<td>3852</td>
<td>7.99%</td>
<td>1280</td>
<td>14.285%</td>
</tr>
<tr>
<td>SBMAP</td>
<td>4125</td>
<td>0.15%</td>
<td>4474</td>
<td>25.427%</td>
<td>1280</td>
<td>14.285%</td>
</tr>
<tr>
<td><strong>Proposed</strong></td>
<td></td>
<td></td>
<td>4119</td>
<td></td>
<td>3567</td>
<td>1120</td>
</tr>
<tr>
<td><strong>Algorithm</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Mapping</strong></td>
<td><strong>Algorithm</strong></td>
<td><strong>MP3enc</strong></td>
<td><strong>MP3dec</strong></td>
<td><strong>% Saving</strong></td>
<td><strong>MP3enc</strong></td>
<td><strong>MP3dec</strong></td>
</tr>
<tr>
<td>ILP</td>
<td>17.021</td>
<td>0.00%</td>
<td>230.407</td>
<td>0.00%</td>
<td>19.823</td>
<td>0.00%</td>
</tr>
<tr>
<td>PSO</td>
<td>17.021</td>
<td>0.00%</td>
<td>230.407</td>
<td>0.00%</td>
<td>19.823</td>
<td>0.00%</td>
</tr>
<tr>
<td>ACO</td>
<td>17.231</td>
<td>1.23%</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>BA</td>
<td>17.834</td>
<td>4.78%</td>
<td>231.450</td>
<td>0.45%</td>
<td>19.936</td>
<td>0.57%</td>
</tr>
<tr>
<td>GA</td>
<td>17.133</td>
<td>0.66%</td>
<td>230.698</td>
<td>0.13%</td>
<td>19.911</td>
<td>0.44%</td>
</tr>
<tr>
<td>CastNet</td>
<td>17.021</td>
<td>0.00%</td>
<td>230.407</td>
<td>0.00%</td>
<td>19.823</td>
<td>0.00%</td>
</tr>
<tr>
<td>SBMAP</td>
<td>-</td>
<td>-</td>
<td>230.432</td>
<td>0.011%</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td><strong>Proposed</strong></td>
<td><strong>Algorithm</strong></td>
<td><strong>17.021</strong></td>
<td><strong>230.407</strong></td>
<td><strong>1120</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Table 5: Power Analysis

<table>
<thead>
<tr>
<th>Mapping Algorithm</th>
<th>VOPD</th>
<th>MPEG-4</th>
<th>MWD</th>
<th>MP3enc MP3dec</th>
<th>263enc MP3dec</th>
<th>263dec MP3dec</th>
</tr>
</thead>
<tbody>
<tr>
<td>ILP</td>
<td>1528</td>
<td>1137</td>
<td>1012</td>
<td>1228</td>
<td>1286</td>
<td>1211</td>
</tr>
<tr>
<td>ACO</td>
<td>1920</td>
<td>1423</td>
<td>1218</td>
<td>1498</td>
<td>1599</td>
<td>1738</td>
</tr>
<tr>
<td>PSO</td>
<td>1841</td>
<td>1357</td>
<td>1112</td>
<td>1507</td>
<td>1445</td>
<td>1561</td>
</tr>
<tr>
<td>GA</td>
<td>1843</td>
<td>1356</td>
<td>1109</td>
<td>1507</td>
<td>1445</td>
<td>1561</td>
</tr>
<tr>
<td>BA</td>
<td>1634</td>
<td>1247</td>
<td>1110</td>
<td>1486</td>
<td>1323</td>
<td>1313</td>
</tr>
<tr>
<td>CastNet</td>
<td>1945</td>
<td>1590</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SBMAP</td>
<td>1650.9</td>
<td>1235</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Proposed</strong></td>
<td><strong>Algorithm</strong></td>
<td><strong>1510.1</strong></td>
<td><strong>1209</strong></td>
<td><strong>1014</strong></td>
<td><strong>1228</strong></td>
<td><strong>1281</strong></td>
</tr>
</tbody>
</table>

### 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 %](image)

### Table 6: Comparison of Computation Time

<table>
<thead>
<tr>
<th>Mapping Algorithm</th>
<th>MPEG-4</th>
<th>VOPD</th>
<th>MWD</th>
<th>MP3enc</th>
<th>MP3dec</th>
<th>263enc</th>
<th>MP3dec</th>
<th>263dec</th>
<th>MP3dec</th>
</tr>
</thead>
<tbody>
<tr>
<td>ILP</td>
<td>22.34</td>
<td>4679.34</td>
<td>210.021</td>
<td>1435.012</td>
<td>193.035</td>
<td>4897.21</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ACO</td>
<td>18.652</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PSO</td>
<td>3.465</td>
<td>3.785</td>
<td>3.432</td>
<td>3.194</td>
<td>3.185</td>
<td>3.188</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GA</td>
<td>3.234</td>
<td>3.925</td>
<td>3.42</td>
<td>3.194</td>
<td>3.185</td>
<td>3.174</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BA</td>
<td>2.925</td>
<td>2.231</td>
<td>2.894</td>
<td>2.653</td>
<td>2.345</td>
<td>2.35</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SCSO</td>
<td>2.010</td>
<td>2.231</td>
<td>1.996</td>
<td>1.785</td>
<td>1.527</td>
<td>1.511</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Proposed Algorithm</td>
<td>2.010</td>
<td>2.215</td>
<td>1.945</td>
<td>1.771</td>
<td>1.525</td>
<td>1.472</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 7: Percentage improvement in CPU Time

<table>
<thead>
<tr>
<th></th>
<th>HGSA Savings % in CPU Time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Over PSO</td>
</tr>
<tr>
<td>MPEG4</td>
<td>72.3%</td>
</tr>
<tr>
<td>VOPD</td>
<td>70.8%</td>
</tr>
<tr>
<td>MWD</td>
<td>76.4%</td>
</tr>
<tr>
<td>263decMP3dec</td>
<td>109.6%</td>
</tr>
<tr>
<td>263encMP3dec</td>
<td>106.7%</td>
</tr>
<tr>
<td>MP3encMP3dec</td>
<td>80.3%</td>
</tr>
</tbody>
</table>

\( d \) 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 patterns on the mesh topology. Traffics are tornado and random. Uniform pattern distributes the traffic in a uniform manner which helps 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.

![](image.png)

Fig. 6 Average Latency for Uniform Traffic
Table 8: Throughput and Energy Analysis

<table>
<thead>
<tr>
<th>Mapping Algorithm</th>
<th>Energy (pJ/packet)</th>
<th>Throughput (packets/cycle)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>VOPD</td>
<td>MPEG4</td>
</tr>
<tr>
<td>ILP</td>
<td>264.608</td>
<td>179.543</td>
</tr>
<tr>
<td>GA</td>
<td>268.324</td>
<td>177.127</td>
</tr>
<tr>
<td>CastNet</td>
<td>264.332</td>
<td>203.462</td>
</tr>
<tr>
<td>SBMAP</td>
<td>264.042</td>
<td>-</td>
</tr>
<tr>
<td>Proposed Algorithm</td>
<td>263.781</td>
<td>174.860</td>
</tr>
</tbody>
</table>

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


