AUT Journal of Electrical Engineering
10.22060/EEJ.2026.25052.5827

LDMO: Hybrid Lemur–Dwarf Mongoose Optimisation Framework for Multi-Objective Application Mapping In 3D-NoC

Juliet Rose David Raj Beatrice Rajam[1*], Jaya Thirasamma[2]

[1*]Assistant Professor, Department of Electronics and Communication Engineering, Mar Ephraem College of Engineering and Technology, Malankara Hills, Elavuvilai, Marthandam, Tamil Nadu.
[2]Professor, Department of Electronics and Communication Engineering, Saveetha Engineering College, Saveetha Nagar, Kanchipuram - Chennai Rd, Sriperumbadur, Chennai.

[1*]Corresponding author email: julietrosedb1992@gmail.com

Abstract:

Three-dimensional Networks-on-Chip (3D-NoC) application mapping is a nondeterministic polynomial-time hard problem by nature. Under tight design restrictions, the efficient allocation of Intellectual Property (IP) cores to processing components must balance silicon area, reduce power consumption, and reduce end-to-end latency. To achieve improved mapping quality for 3D-NoC designs, this paper presents a Hybrid Lemur–Dwarf Mongoose Optimisation (LDMO) approach that addresses the cooperative exploitation abilities of Dwarf Mongoose Optimisation and the exploratory behaviour of Lemur Optimisation. To prevent premature stagnation, the Lemur Optimisation Algorithm mimics lemur cliff-leaping and tree-navigating behaviour during first stage. This produces a diversity of initial mapping candidates with high population variance. To optimise the mapping to global optima with high convergence speed, the Dwarf Mongoose Optimisation Algorithm employs adaptive leadership, sentinel–scout coordination, and foraging-based neighbourhood search during the second stage. The average communication delay (hop-dependent propagation, serialisation, and router latencies), area overhead (switch, interconnect, and core dimensions), and total power dissipation (router and interconnect power) are all minimised at the same time through a multi-objective fitness function. By adaptive switching between the two phases, the hybrid approach dynamically trades off between exploration and exploitation to ensure robustness across a wide range of communication demands and traffic patterns. Simulation outcomes demonstrate that the proposed LDMO framework consistently delivers reduced computation overhead, marked improvements in latency, and substantial energy efficiency. Furthermore, as the number of cores and communication links scale

10.22060/EEJ.2026.25052.5827

upward, the hybrid optimisation strategy maintains high-quality mapping solutions, underscoring its robust scalability across diverse NoC configurations.

Keywords:

3D-NoC application mapping, IP cores, LDMO approach, Dwarf Mongoose Optimisation, Lemur Optimisation.

1. Introduction and Motivation

Many-core architectures have become a foundational component for delivering high computational throughput in platforms such as cloud servers and big data infrastructures. These environments experience dynamic workloads, with diverse applications entering and exiting the system during runtime [1]. Efficient online task-to-core mapping is essential for enhancing overall system performance. While software-based simulations offer design flexibility, achieving performance acceleration is critical for high-efficiency and real-time applications [2]. Deploying neuromorphic systems on specialised hardware significantly enhances energy efficiency and execution speed, making them well-suited for power-constrained platforms such as mobile and IoT devices. As edge applications increasingly demand large-scale neural networks, scalable architectures become essential [3]. To meet this need, designers have embraced NoC interconnects, particularly 3D-NoC which enable multiple routing paths and efficient packet traversal. Despite improvements in hardware throughput and circuit speed, a key challenge in neuromorphic system design lies in application mapping that maximises functionality while sustaining peak performance [4, 5]. The shift toward scalability via 3D integration by vertically stacking 2D dies using Through-Silicon Vias (TSVs) introduces new complexities, especially with respect to thermal management [6].

2. Background on NOC and 3D Integration

With the continuous growth in transistor density on chips, conventional interconnect infrastructures struggle to support the escalating on-chip communication demands. To address this limitation, the NoC paradigm

has emerged as a scalable and parallel communication framework. A typical NoC architecture, fig.1 comprises a global interconnection network formed by routers linked to computational or memory modules [7].

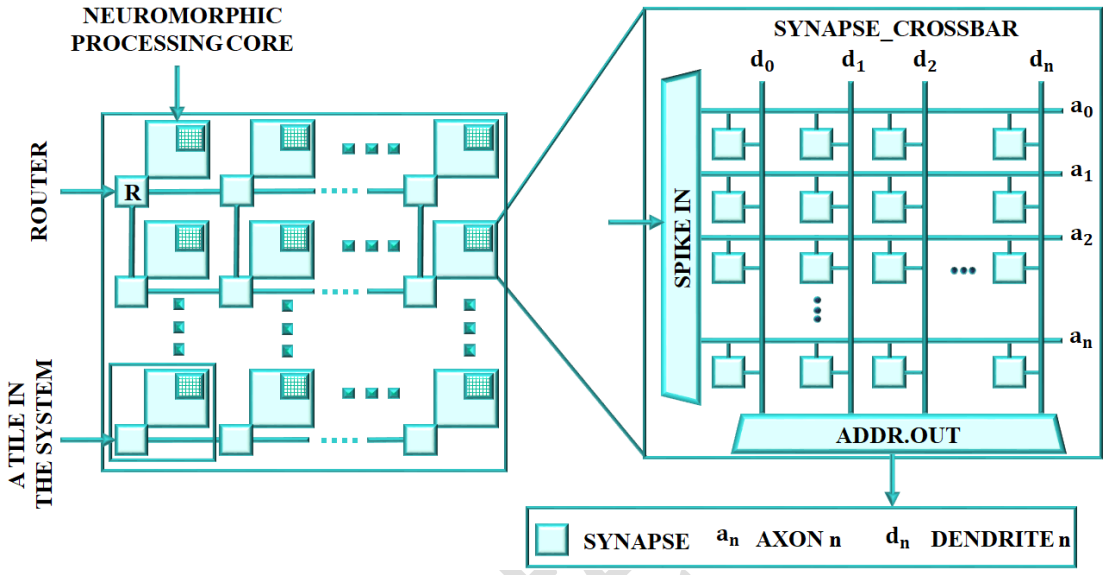

Fig. 1. Structural Elements of a Neuromorphic System Utilising NoC Architecture.

Given the central role of the global network in NoC design, it has attracted significant research attention. However, as NoC architectures evolve toward integrating hundreds or even thousands of cores, traditional two-dimensional (2D) interconnects face critical limitations such as excessive global wire lengths and increased packet transmission delays that hinder scalability [8]. In contrast, three-dimensional (3D) Integrated Circuits (ICs) have gained traction due to their potential to enhance chip performance, functional density, and packaging efficiency. By combining the benefits of NoC and 3D IC technologies, the 3D NoC architecture has been proposed as a promising solution to overcome these challenges [9].

3. Mapping Problem Definition

Despite its advantages, the design space exploration of 3D NoC presents several complexities. Among these, task mapping stands out as a pivotal process that significantly impacts system performance from the early stages of design [10]. The mapping problem in NoC design refers to the process of assigning a selected

10.22060/EEJ.2026.25052.5827

set of IP cores to the network's resource nodes in a manner that optimises key performance metrics. This research assumes that all preliminary design steps have been completed and concentrates specifically on the mapping phase [11].

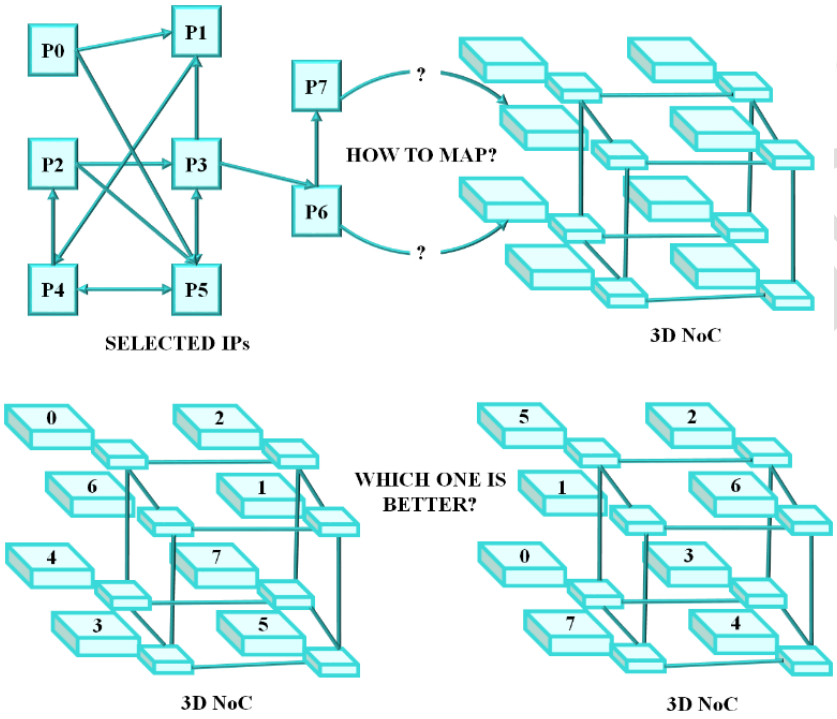

Fig. 2. 3D NoC mapping challenges.

Two primary challenges characterise NoC mapping, as illustrated in Fig.2:

(1) Developing an effective mapping algorithm, and

(2) Constructing a reliable evaluation model.

Essentially, the first challenge involves determining how to allocate IP cores across a given topology, while the second requires formulating an analytical framework to assess the quality of the mapping strategy [12].

This work focuses primarily on the first challenge.

Mapping is inherently a quadratic assignment problem and is classified as NP-hard, with a 3D NoC of n nodes yielding n! possible configurations [13]. Due to the computational infeasibility of exhaustive search

methods, numerous approximation techniques have been introduced. 3D NoCs scale to accommodate hundreds or even thousands of cores and the time required to reach optimal solutions grows exponentially [14]. Therefore, there is a pressing need for algorithms that offer faster convergence while avoiding suboptimal outcomes.

4. Related Work

Given the critical role of mapping in NoC performance, extensive research has been devoted to developing efficient algorithms. These algorithms are broadly categorised into two types: static and dynamic and reviewed in table.1.

Table 1. Summary of conventional mapping algorithms.

SI. NO	REF	SCOPE OF METHOD	MERITS	DEMERITS
STATIC MAPPING ALGORITHMS				
[15]	Muhammad Junaid Mohiz *et al* (2021)	Application mapping in NoC using an initial greedy placement followed by meta-heuristic Cuckoo Search Optimization (CSO) via Lévy flight.	Greedy initialization accelerates CSO convergence.	Performance under high-volume communication patterns.
[16]	Aruru Sai Kumar *et al* (2023)	Efficient Real-Time Embedded Application Mapping (ERTEAM) based NoC mapping strategy is integrated	Leverages reduced Core Average Distance (CAD) for defining mapping regions.	Efficiency is hindered with heterogeneous or irregular NoC technologies.
[17]	Samala Jagadheesh *et al* (2022)	RL-MAP: Reinforcement Learning-Based Mapping Framework for unsupervised application mapping in NoC systems	Learns mapping heuristics without requiring labelled data.	Nevertheless, it lacks integration of thermal, fault-tolerance, and congestion metrics due to limited communication cost.
[18]	Kiran K A *et al* (2025)	Optimization of application mapping in 2D and 3D mesh NoC architectures using GA and CastNet algorithms.	Demonstrates up to 10% reduction in communication cost compared to 2D mapping	GA requires longer CPU run time to converge
[19]	Ke Li *et al* (2025)	ET (Enhanced Coati Optimization Algorithm) for NoC task mapping.	Spectral clustering provides strong	Effectiveness may be reduced under irregular

			initial population quality.	topologies or highly non-uniform task distributions.
DYNAMIC MAPPING ALGORITHMS				
[20]	Waqar Amin *et al* (2023)	Design-time allocation and runtime reconfiguration for dynamic workloads in NoC-based MPSoCs.	Provides dynamic workload adaptation via runtime remapping.	Exact application arrival times or runtime resource availability is limited.
[21]	Zeeshan Ali Khan *et al* (2022)	Multilevel application mapping technique for Deep Learning (DL) Artificial Intelligence (AI) workloads on NoC-based architectures.	Provides real-time execution of AI applications with enhanced task distribution.	However, it is limited to mesh-based NoC architectures.
[22]	Arvind Kumar *et al* (2022)	Horological Mapping (HorMAP), Rotational Mapping (RtMAP) and Divide and Conquer Mapping (DACMAP) based mapping algorithms for NoC architectures is presented.	Efficient communication across hundreds of cores integrated on a single silicon die.	Nevertheless, it lacks implementation for 3D NoC structures
[23]	Lluís Ribas-Xirgo *et al* (2025)	Proposes dynamic, energy-aware routing in NoC using the Hungarian Algorithm (HA).	Demonstrates improved performance with multi-memory architecture	Sequential nature of HA limits parallelization
[24]	Mohamed Maatar *et al* (2024)	BTSAM (Balanced Thermal-State-Aware Mapping) for 3D-NoC-based systems.	Provides balanced thermal distribution across 3D-NoC systems.	Heterogeneous task mapping not addressed

5. Identified Research Gap in NOC Mapping Algorithms

In spite of significant progress in static and dynamic mapping approaches for NoC designs, some primary limitations remain that hinder system-wide optimization:

- Most algorithms are specifically developed for mesh-based or uniform NoC designs, limiting their adaptability to heterogeneous, irregular, or evolving 3D NoC implementations.
- Existing implementations tend to omit multi-objective optimization and do not take system constraints into account, hampering their robustness and generalization.

6. Contribution of proposed 3D NOC based mapping application.

Optimal application mapping in 3D-NoC designs is still a computationally burdensome problem, especially with tight design constraints. Obtaining optimal IP core placement entails a subtle balancing act between saving silicon footprint, minimizing power usage, and reducing end-to-end latency.

This paper proposes a new Hybrid LDMO framework with the purpose of improving mapping quality using a two-phase metaheuristic approach.

- ✓ The first phase takes advantage of the Lemur Optimization's exploratory dynamics, based on cliff-leaping and tree-navigation processes, to develop a robust pool of mapping candidates.
- ✓ The second phase applies the Dwarf Mongoose Optimization's cooperative search processes through adaptive leadership and sentinel–scout coordination to improve solutions toward global optima.
- ✓ A multi-objective fitness function directs the process, at the same time reducing communication delay, area overhead, and power dissipation. With adaptive switching between exploration and exploitation, the LDMO method presents strong performance under different traffic patterns and communication requests.

Since the proposed framework is 3D NoC architecture-aware, it goes beyond the conventional 2D NoC mapping approaches. Vertical communication is supported by Through Silicon Vias (TSVs) that have delay and energy characteristics different from those of planar links. The method thus differs from traditional 2D NoC mapping because it allocates layer-aware tasks first and then communicates the planning between both planar and vertical dimensions by balancing the communication layers. The 3D features of the targeted models such as connecting TSVs modelling vertical delays, and thermal balancing across layers separate proposed algorithms from 2D NoC methods and prove their appropriateness for 3D.

7. Proposed System Modelling

7-1- Mapping Strategy for 3D Network-on-Chip Architectures

10.22060/EEJ.2026.25052.5827

Application mapping in 3D-NoC is inherently an NP-hard problem, requiring intelligent strategies to allocate Intellectual Property (IP) cores to processing nodes while minimizing silicon area, communication latency, and power consumption under strict design constraints. The 3D NoC platform consists of resource nodes, routers, and physical interconnects arranged in a regular mesh topology. Each layer in the architecture maintains uniform dimensions, and vertical communication is facilitated through TSVs. The overall structure is defined by a grid of size $(N \times N \times L)$, where $N \times N$ denotes the number of nodes per layer and L represents the number of stacked layers. For instance, a $3 \times 3 \times 3$ configuration yields 27 mappable resource nodes.

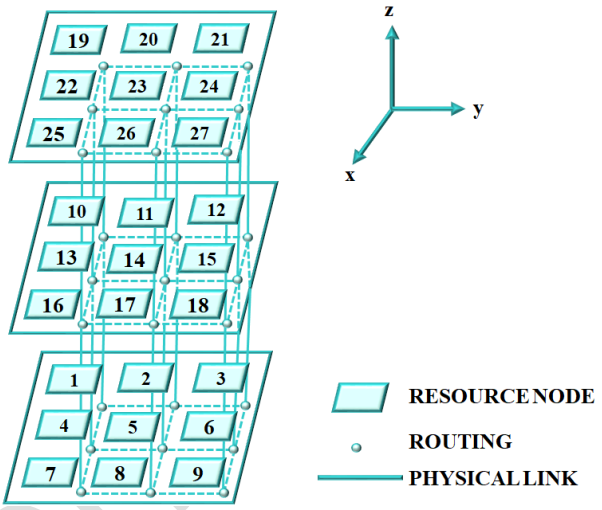

Fig. 3. Structural topology graph.

Nodes are categorised based on their connectivity, fig.3, with link counts ranging from 3 to 6 depending on their position in the mesh. These links enable horizontal and vertical communication between adjacent nodes, ensuring seamless data exchange across the chip.

Mapping Model and Objective Function

The mapping process involves assigning logical IP cores represented in a Task Characteristic Graph (TCG), fig. 4. (a) to physical resource nodes defined in an Architecture Characteristic Graph (ARCG) fig.4 (b).

10.22060/EEJ.2026.25052.5827

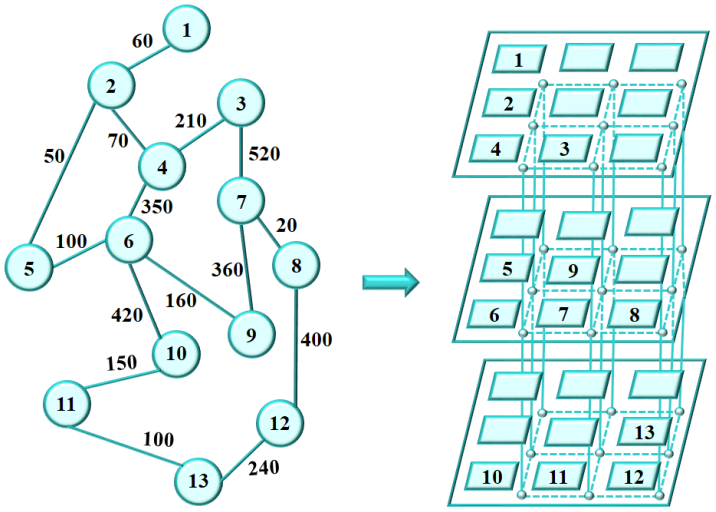

Fig. 4. 3D NoC mapping (a) Characteristic Graph (b) Architecture Characteristic Graph.

Task Characteristic Graph (TCG)

The TCG is a directed acyclic graph $TCG(V,T)$, where each vertex $v_i \in V$ denotes a task, and each arc $t_{i,j} \in T$ represents communication between tasks $t_i$and t_j, weighted by communication volume $\omega_{i,j}$.

The communication relationship between all tasks is represented in eqn (1):

$$W = \left[\omega_{i,j}\right] = \begin{bmatrix} 0 & \cdots & \omega_{0,n-1} \\ \vdots & \ddots & \vdots \\ \omega_{n-1,0} & \cdots & 0 \end{bmatrix} \quad (1)$$

Architecture Characteristic Graph (ARCG)

Similarly, the ARCG, $ARCG(E,P)$ models the physical layout, where each node $e_i \in E$ corresponds to a resource node and each edge $p_{i,j} \in P$ denotes a communication path, weighted by $f_{i,j}$. Here, $f_{i,j}$ represents the communication cost between e_i and e_j, calculated as the product of communication volume $\omega_{i,j}$ and the Manhattan distance $M_{d_{i,j}}$. When the number of resource nodes ($V = E$), a one-to-one mapping is achieved. The communication cost between nodes e_i and e_j is calculated using the Manhattan distance $M_{d_{i,j}} = |x_i - x_j| + |y_i - y_j| + |z_i - z_j|$, and the communication volume is defined in eqn (2):

10.22060/EEJ.2026.25052.5827

$$f_{i,j} = \omega_{i,j} \times M_{d_{i,j}} \tag{2}$$

The mapping objective is to minimise the total communication cost across the network is computed in eqn (3):

$$\begin{cases} map(V \to E) \quad i \in |V|, j \in |V| \\ s.t.m(e_i) = v_i \; \forall e_i \in E, \exists v_i \in V \end{cases} \tag{3}$$

3D Routing assumptions and TSV-aware communication modelling:

The 3D-NoC mapping framework, which is proposed uses a deterministic XYZ routing strategy to ensure deadlock-free communication and predictable latency behaviour. With XYZ routing, the packets go through the X- and Y-axes before they are switched to the Z-axis via Through Silicon Vias (TSVs). An ordering constraint is only due to routing correctness and deadlock avoidance, and it does not mean that the optimization process is biased towards planar communication. The optimization is done at the application-mapping level, where task placement across layers is directly optimized to take advantage of vertical communication opportunities while still physically TSV constraints.

Unlike planar NoCs, 3D-NoC architectures inherently exhibit different electrical and physical characteristics along the vertical dimension. Hence, planar and vertical communications are modelled separately in terms of delay, energy consumption and resource availability. Vertical links made possible by TSVs are directly considered in the fitness evaluation, thus the Z-dimension is a fundamental part of the optimization process.

Power Consumption Model

Power consumption in 3D NoC systems is primarily driven by data transmission between resource nodes. The energy required to transmit 1-bit of data is modelled in eqn (4):

$$E_{bit} = E_{S_{bit}} + E_{L_{bit}} \tag{4}$$

10.22060/EEJ.2026.25052.5827

Where, $E_{S_{bit}}$ is energy consumed by the crossbar switch, $E_{L_{bit}}$ is energy consumed by adjacent routing links. (Buffer and wire energies are negligible in homogeneous platforms). Eqn (5) defines the energy to transmit 1-bit from node e_i to e_j is,

$$E_{bit}^{n_{i,j}} = \left(M_{d_{i,j}} + 1\right) \times E_{S_{bit}} + M_{d_{i,j}} \times E_{L_{bit}} \tag{5}$$

Thus, the total communication energy across the NoC is computed using eqn (6),

$$E = \sum_{i \le |V|}^{j \le |V|} \left[\left(M_{d_{i,j}} + 1\right) \times E_{S_{bit}} + M_{d_{i,j}} \times E_{L_{bit}} \right] \times \omega_{i,j}, i \in |V|, j \in |V| \tag{6}$$

Since all terms except $\sum_{i \le |V|}^{j \le |V|} f_{i,j}$ are constants, minimising $f_{i,j}$ becomes the key to achieving low-power mapping.

In the proposed model, communication delay and energy are treated differently for planar and vertical links. The planar links have higher delay caused by longer wire lengths, higher capacitive loading and traversals of repeated routers. On the other hand, TSV-based vertical links give short interconnect paths between stacked layers, thus lower propagation delay per hop. Still, TSV communication comes with extra power overhead due to TSV landing pads, drivers and vertical interconnect capacitance. Therefore, even though the latency of TSV hops is very efficient, their power cost and limited availability usage is very carefully balanced during the mapping. Thus, the total communication delay is determined as the weighted sum of planar and vertical hops, with planar hops being assigned a higher delay per hop than vertical hops. Such modelling ensures that the optimizer only allow the vertical communication if this leads to a net gain in the reduction of latency and energy consumption.

Impact of TSV delay, power and density constraints in 3D-NoC mapping:

TSVs greatly reduce the latency of signals in comparison to planar links. However, due to fabrication limitations, thermal issues and area overhead, 3D-NoC architectures in real life impose very strict

constraints on the density of TSVs. Rather than assuming full vertical connectivity across layers, proposed research considers TSV as a scarce resource. Realistic TSV pitch and sizing constraints are illustrated by each router supporting a finite number of vertical connections.

Too much vertical communication cause congestion at the interfaces of the TSVs, an increase in the driver power dissipation and the cells becoming thermally stressed. Therefore, the TSV utilization indirectly influences the optimization process via the communication energy and latency terms of the fitness function. Vertical link over-exploitation on mapping solutions leads to energy consumption and congestion penalties, thus the model disallows unrealistic vertical traffic concentration. Such TSV-aware modelling allows the optimizer produce mappings that are well balanced and controlled, thus the vertical links left open for latency reduction without violating the power and density constraints of the TSVs. The results therefore, are combinations that are both excellent in terms of performance and realizable physically.

Table 2: Planar and vertical link characteristics in proposed 3D-NoC model.

Parameter	Planar Links	TSV Links
Physical length	Long	Short
Delay per hop	Higher	Lower
Energy per hop	Moderate	Driver-dominated
Connectivity	Full mesh	Limited density
Congestion risk	Distributed	Localized

Table. 2 presents the planar and vertical characteristics considered in proposed 3D-Noc model. This structural advantage reinforces the importance of intelligent mapping strategies that exploit vertical paths to reduce overall power consumption and latency forming basis for the optimization algorithm proposed in this study [25].

7-2- A Hybrid Lemur–Dwarf Mongoose-Based Optimization Strategy for Multi-Objective Application Mapping in 3D Network-on-Chip Architectures

10.22060/EEJ.2026.25052.5827

Hybrid LMDO based 3D NoC mapping approach is proposed integrating lemur optimization for effective exploration by the leap up behaviour and exploitation by dance-hup fine tunings and dwarf mongoose optimization integrates social rank, babysitter transfer, and scout-based nomadic search for enhancing convergence around promising regions. This synergy ensures that the algorithm has population diversity in initial iterations, searches new mapping areas aggressively and subsequently conducts intensified local search around good candidate mappings. This leads to an adaptive equilibrium between exploration and exploitation, which is essential for discovering optimal or near-optimal mappings in a multi-objective 3D-NoC setting.

7-2-1- Lemur Optimisation (LO) Algorithm: Conceptual Foundation and Mathematical Model

The Lemur Optimiser draws inspiration from the unique physical and social characteristics of lemurs, a prosimian primate group that is evolutionarily different from monkeys and apes. Lemurs are endemic to Madagascar and the Comoro Islands and exist in various forest habitats, from marshes and dry deciduous forests to dense rainforests. The LO algorithm, simulates two basic survival and locomotion behaviours exhibited by lemurs:

(a) Leap-Up Behaviour (Exploration)

In nature, lemurs are extremely nimble jumpers. The leap-up activity entails a vertical thrust, where the lemur pushes against the ground or a branch of a tree and leaps vertically into the air, finally landing in a position sitting upright while holding on to a trunk of a tree with both hands and feet. This vertical jump is used for two primary reasons:

(i) Navigation between trees to access food sources, and

(ii) Escape from predators on the ground by accessing a higher, safe position.

10.22060/EEJ.2026.25052.5827

In the LO algorithm, this leap-up activity is conceptualised as an exploration mechanism, permitting prospective solutions (lemurs) to take giant leaps in the search space to venture into unexplored areas. Leap-up randomness ensures that the search process does not converge prematurely to local optima.

(b) Dance-Hup Behaviour (Exploitation)

When lemurs reach fairly distant trees or holes in between foraging spots, they adopt a horizontal locomotion behaviour known as dance-hup. In dance-hup, the lemur drops down to the ground, employs its arms in lateral balance, and executes high, rhythmic horizontal bounds while waving its arms up and down to maintain equilibrium. This guided horizontal hopping enables the lemur to reduce travel distance and move to a nearby tree effectively. LO translates dance-hup into an exploitation phase, in which candidate solutions adjust positions through movement towards the best near lemur (bnl) or the global best lemur (gbl) up until now. This directed movement enables the population to converge towards good-quality solutions without divergence too prematurely.

Mathematical Formulation of LO Algorithm

The LO algorithm is a population-based method, where the lemur agents are represented in matrix form (X), LO searches in a (d)-dimensional space and stores (n) candidate solutions. The initial population matrix is defined in eqn (7):

$$X = \begin{bmatrix} l_1^1 & l_1^2 & \cdot & l_1^d \\ l_2^1 & l_2^2 & \cdot & l_2^d \\ \vdots & \vdots & \vdots & \vdots \\ l_n^1 & l_n^2 & \cdot & l_n^d \end{bmatrix} \tag{7}$$

Where, (X) is the population matrix $(n \times d)$, (n) is the number of lemurs in the population (population size), (d) is the problem dimensionality, i.e., the number of decision variables in the search space.

Every candidate solution is populated randomly within the given search space using eqn (8):

$$X_i^j = \left(LB + \left(UB_j - LB_j\right)\right) \times r \tag{8}$$

10.22060/EEJ.2026.25052.5827

Here, (r) is a random number uniformly distributed in the interval [0, 1], LB is lower bound of search space, j^{th} decision variable upper bound and lower bound are UB_j and LB_j respectively. For each randomly created candidate solution, its opposition point $X_i^{j,opp}$ is also calculated as per eqn (9):

$$X_i^{j,opp} = LB_j + UB_j - X_i^j \tag{9}$$

The fitness of X_i^j and $X_i^{j,opp}$ is calculated, and the better one is kept in the initial population. This ensures the initial solutions are well distributed throughout the search space, which enhances convergence performance.

One of the most important control parameters in LO is the Free Risk Rate (FRR), which sets whether a potential candidate lemur will attempt an exploratory leap-up or an exploitative dance-hup action. FRR is reduced gradually through iterations in order to increasingly move towards exploitation from exploration using eqn (10).

$$FPR = HRR - t \times \left((HRR - LRR) / Max_{iter} \right) \tag{10}$$

Where, (HRR) is the High Risk Rate, which is the highest level of risk-taking (promotes exploration), (LRR) Low Risk Rate, is the minimum risk level (promotes exploitation), (t) is iteration number at present, and (Max_{iter}) is the number of maximum iterations.

For every candidate solution (X_i), a fitness function is computed using eqn (11) to test its quality.

$$Fit(X_i) = \alpha \cdot (1 - Acc) + \beta \cdot \frac{s}{S} \tag{11}$$

Where, $Fit(X_i)$ is the i^{th} solution's fitness value, (Acc) is the mapping quality measure, (s) is the number of used resources, (S) is the number of available resources, α and β are weighting factors balancing accuracy importance and resource minimization.

After the fitness of all solutions is calculated, the algorithm categorizes solutions into two groups:

10.22060/EEJ.2026.25052.5827

1. Best Near Lemur: The global best(s) in the iteration, which are good local optima.

2. Global Best Lemur: The global best solution found so far across all iterations until current iteration, which is the global best candidate solution.

The position update rule is responsible for how every lemur moves in the search space. Based on whether random number (r_1) is less than or greater than FRR, lemur move towards the bnl or gbl. The movement of lemurs in the search space is governed by eqn (12),

$$X_i^j = \begin{cases} x(i,j) + \gamma \cdot \left| \left(x(i,j) - x_{bnl,j} \right) \cdot \left(r_3 - 0.5 \right)^2, r_1 < FRR \right| \\ x(i,j) + \gamma \cdot \left| \left(x(i,j) - x_{gbl,j} \right) \cdot \left(r_3 - 0.5 \right)^2, r_1 \geq FRR \right| \end{cases} \quad (12)$$

Where, X_i^j is the current value of the j^{th} variable of the i^{th} solution, $x_{bnl,j}$ and $x_{gbl,j}$ are the j^{th} variable of the best near and global best lemur, respectively, $r_1, r_3 \sim U[0,1]$ are random numbers that provide stochasticity, (γ) is a scaling factor that regulates step size.

This update rule captures the leap-up (exploration) and dance-hup (exploitation) phases. For $(r_1 < FRR)$, the solution is exploratory and moves with respect to the best near solution, whereas for $(r_1 \geq FRR)$, the solution uses knowledge of the global best and converges towards it [26].

7-2-2- Dwarf Mongoose Optimization Algorithm (DMOA): Behavioural Modeling and Mathematical Framework

The DMOA is inspired by the cooperative behaviour and social hierarchy of dwarf mongooses. These small carnivorous mammals exhibit collective decision-making, territory defines, and division of labour traits that translate naturally into exploration–exploitation mechanisms for optimization. The primary biological traits that influence the algorithm design include:

- *Group Structure:* Dwarf mongooses live in groups of 12–15 members, typically with a dominant alpha pair responsible for decision-making and resource allocation.

10.22060/EEJ.2026.25052.5827

- *Sentinel Behaviour:* A designated sentinel guards the group during foraging, scanning the surroundings for threats. This role rotates dynamically.
- *Foraging and Scouting:* While most members forage for food, a few act as scouts to search unexplored areas, increasing the group's adaptability and survival rate.
- *Collective Movement:* The group frequently relocates to new territories, balancing exploration of new areas with exploitation of known safe zones.

This biological behaviour is mathematically abstracted into an optimization algorithm that balances exploration (scouting new search regions) with exploitation (improving known good solutions).

The mapping procedure starts by creating a population matrix (X) of prospective solution is given in eqn (13) such that each row is a possible mapping configuration and each column is a decision variable:

$$X = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,d} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,d} \\ \vdots & \vdots & x_{i,j} & \vdots \\ x_{n,1} & x_{n,2} & \cdots & x_{n,d} \end{bmatrix} \tag{13}$$

Where, population size (n) is given in rows, (d) is the number of decision variables (problem dimensions), $x_{i,j}$ is value of the j^{th}variable of the i^{th} mongoose.

Each decision element (X_j) is initialized using eqn (14) with a uniform random distribution within the NoC architectural limits:

$$X_j = unifrnd\left(LB, UB, D\right) \tag{14}$$

Here, (LB) and (UB) represent minimum and maximum of mapping space, and (D) represents number of cores.

Probability of selecting each solution is computed using eqn (15) for further investigation is

$$Fit_i = \rho \times \gamma \times \left(1-\rho\right) \times \left(\frac{\left|BX_{ij}\right|}{D}\right) \tag{15}$$

10.22060/EEJ.2026.25052.5827

Where, ρ is selection probability factor which controls the likelihood of the candidate solution, γ is scaling coefficient, regulates the influence of fitness contribution from communication cost and hop count. Every mapping configuration is assessed based on a fitness function that takes communication cost, hop count, and energy dissipation into account. Eqn (16) defines the normalized selection probability for every candidate,

$$\alpha = \frac{fit_i}{\sum_{i=1}^{n} fit_i} \tag{16}$$

The active population size is modified by eqn (17), subtracts the non-mapping agents:

$$n = n - bs \tag{17}$$

The alpha mongoose (leader) releases a signal (peep) to lead the population towards promising mapping areas. Each candidate solution's position is updated according to eqn (18):

$$X_{i+1} = X_i + phi \times peep \tag{18}$$

Where, (phi) is a random vector within the interval $[-1, 1]$, bringing in stochasticity in the search. The idea of a Sleeping Mound (SM) corresponds to a local optimum within the mapping space. The fitness improvement between iterations is calculated using eqn (19), updating SM values:

$$sm_i = \frac{fit_{i+1} - fit_i}{max\left\{\left| fit_{i+1}, fit_i \right|\right\}} \tag{19}$$

Eqn (20) computes the population average of SM:

$$\phi = \frac{\sum_{i=1}^{n} sm_i}{n} \tag{20}$$

To prevent premature convergence and allow exploration of the 3D-NoC topology, scout agents seek alternative mapping configurations. The update rule includes a movement vector (M), a control factor (CF), and a random multiplier: The scout's agent's movement is defined in eqn (21),

$$X_{i+1} = \begin{cases} X_i - CP \times phi \times rand \times \left[X_i - M\right], \phi_{i+1} > \phi_i \\ X_i + CF \times phi \times rand \times \left[X_i - M\right], otherwise \end{cases} \tag{21}$$

10.22060/EEJ.2026.25052.5827

Eqn (22) accumulates the influence of all agents:

$$M = \sum_{i=1}^{n} \frac{X_i \times sm_i}{X_i} \tag{22}$$

The control factor (CF) is updated using eqn (23), linearly decreases across iterations to move from exploration to exploitation:

$$CF = \left(1 - \frac{iter}{Max_{iter}}\right)^{\left(\frac{2 \times iter}{Max_{iter}}\right)} \tag{23}$$

To provide dynamic role switching between scouts and babysitters is governed by eqn (24), the algorithm employs a phase-switching condition:

$$phase = \begin{cases} Scout, C \le L \\ Babysitting, C \ge L \end{cases} \tag{24}$$

Here, (C) is the current iteration counter and (L) is the threshold for role switch. When the condition is satisfied, the population is re-initiated and the alpha returns to scouting [27].

The hybrid LDMO algorithm effectively incorporates the LO and DMOA to provide a robust, adaptive, and efficient metaheuristic for multi-objective application mapping in 3D-NoC architectures.

1. Initialization (Wide Coverage by LO Leap-Up):

The candidate mappings population is randomly spread using LO's leap-up behaviour, which ensures wide exploration of the solution space. LO leap-up initialisation is expressed in eqn (25)

$$X_{i,j}^{(0)} = LB_j + r \cdot \left(UB_j - LB_J\right) \tag{25}$$

2. Exploration Enhancement (DMOA Scouts Refinement):

Once LO spreads initial lemur solutions, DMOA scout mongooses conduct semi-random search in under-explored areas to prevent missed optima. This prevents premature clustering around local solutions and enhances search diversity.

3. Exploitation Stage (LO Dance-Hup + DMOA Alpha Group):

10.22060/EEJ.2026.25052.5827

LO's dance-hup behaviour brings solutions near promising optima, simulating exploitation. At the same time, DMOA's alpha group and babysitter exchange mechanism refines the best solutions, balancing convergence rate and diversity preservation.

4. Global Convergence:

The LO's best solution is cross-evaluated with the alpha mongoose solution of DMOA.

If a better mapping is discovered by scouts, it updates the global best, thereby ensuring joint convergence of both algorithms.

5. Termination:

The algorithm stops when maximum number of iterations is reached or when fitness improvement $|F^{t+1} - F^{t}|$ drops below a thresholdϵ.

Pseudo-Code for LDMO Algorithm

Input: Population size N, Max iterations Max_{iter}, Bounds LB/UB, LO parameters (HRR, LRR), DMOA parameters (Babysitter ratio, CF), Multi-objective weights (w_1, w_2, w_3, w_4)
1. Initialize population using LO leap-up rule:
    for each candidate i in population:
        Xi = LB + rand(0,1)(UB - LB)
2. Evaluate fitness $F(X_i)$ using multi-objective function.
3. Identify best lemur (gbl) and alpha mongoose.
4. For t = 1 to Max_{iter} do:
  a. Update Free Risk Rate (FRR) for LO:
    FRR = HRR - t((HRR - LRR)/Max_{iter}
  b. For each lemur:
    if rand < FRR:
      Update position using exploration (leap-up)
    else:
      Update position using exploitation (dance-hup)
  c. Update DMOA hunters:
    For each hunter:
      Move toward alpha mongoose & food source
    For each scout:
      Randomly explore (scouting behaviour)
  d. Babysitter exchange:
    Swap babysitters with some hunters to maintain diversity
  e. Evaluate fitness for updated population
  f. Update global best solution (gbl) if better solution found
  g. Check convergence:
    if $\left|F_{best(t+1)} - F_{best(t)}\right| < \varepsilon$|:

10.22060/EEJ.2026.25052.5827

```
                    break
5. Return best solution gbl as final mapping configuration
```

Hybrid LDMO algorithm optimizes all four major goals at once:

1. Latency Minimization:

End-to-end communication latency decreased through objective function using eqn (26):

$$f_1 = \frac{\sum_{i=1}^{T} L_i}{T} \tag{26}$$

Where, L_i denotes latency for task (i), and (T) is number of tasks.

2. Energy Consumption:

Switching and link energy are captured in eqn (27):

$$f_2 = \sum_{i=1}^{T} \left(E_{switch,i} + E_{link,i} \right) \tag{27}$$

3. Thermal Balancing:

No Temperature hotspot variance minimized as shown in eqn (28):

$$f_3 = \sigma_T^2 = \frac{\sum_{i=1}^{n} \left(T_i - \bar{T} \right)^2}{n} \tag{28}$$

4. Deadlock-Free Mapping:

No Valid mapping solution with routing constraints met, through penalty term P. Deadlock avoidance is enforced through eqn (29):

$$\mathrm{f}_4 = \mathrm{P}(\mathrm{deadlock}) = \begin{cases} 0, & if\ CDG\ is\ acyclic\left(deadlock - free\right) \\ \mathrm{P}_{\max}, & if\ any\ cycle\ is\ detected \end{cases} \tag{29}$$

Here, P(deadlock) is binary penalty term which invalidates mappings with cyclic channel dependencies. It assigns zero for deadlock-free configurations and a large constant penalty $\mathrm{P}_{\max}$ when any cycle is detected, ensuring only feasible solutions are retained. Deadlock avoidance measures are extended to planar and vertical channels alike by turning the channel dependency graph into one with horizontal links and vertical links based on TSVs. Cyclic dependencies of planar and vertical channels as well as their mixtures

10.22060/EEJ.2026.25052.5827

are identified and penalized through the deadlock penalty term. Consequently, deadlock freedom is assured for the whole 3D communication fabric and not only for planar routing.

The considerable performance gain implied that the proposed LDMO framework has been successfully leveraging TSV-based vertical communication to minimize end-to-end latency without the vertical interconnect fabric getting overloaded. The latency drop by LDMO points at highly efficient cross, layer task clustering, and the stable energy usage ensures that the number of TSVs stays within reasonable limits. Hence, the results demonstrate that the proposed mapping strategy is not only a planar one but a vertical one as well in a controlled and TSV-aware way.

The combined overall fitness function is defined in eqn (30):

$$F = w_1 f_1 + w_2 f_2 + w_3 f_3 + w_4 f_4 \tag{30}$$

Weights w_1, w_2, w_3, w_4 are adjusted based on application priority (e.g., energy-aware or latency-aware designs). The multi-objective fitness function naturally takes into account the mixed nature of planar and vertical communication. In particular, the latency and energy objectives reflect the number of planar and TSV hops with different cost weights given to each. Vertical communication enjoys lower hop delay but adds more TSV driver and interconnect power. As a result, the optimization procedure automatically finds the trade-off between planar and vertical communication, thus, no dimension is excessively exploited. This single objective formulation ensures that enhancements of latency via vertical communication is not accompanied by a huge increase in power consumption or uncontrolled use of TSV.

10.22060/EEJ.2026.25052.5827

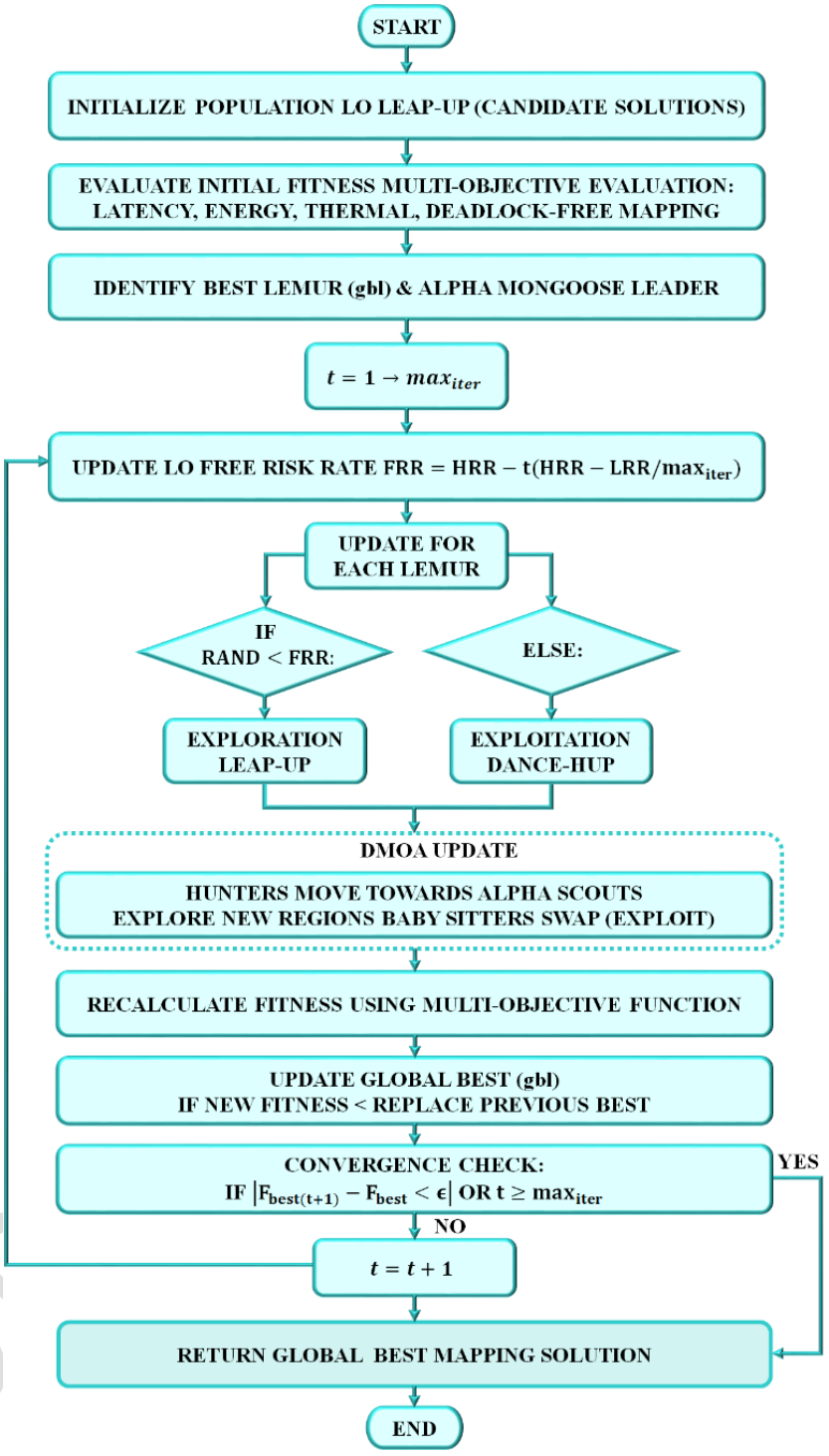

Fig. 5. Flowchart of LDMO based application Mapping.

10.22060/EEJ.2026.25052.5827

Fig.5 shows the flowchart of hybrid LDMO framework offers a robust, adaptive, and scalable solution for multi-objective application mapping in 3D-NoC architectures, effectively balancing exploration and exploitation while optimizing key performance metrics.

8. Result and Discussion

Empirical performance of proposed application mapping strategy based on the hybrid lemur–dwarf mongoose optimization framework is systematically examined for standard NoC benchmarks including PIP, VOPD, 263encMP3dec and MP3enMP3dec.

Table 3. Specifications of NoC benchmark applications

Benchmark	Edge	Node
PIP	18	16
VOPD	24	16
263encMP3dec	36	24
MP3enMP3dec	32	20

Detailed specifications of the benchmark applications are presented in Table.3. Simulation experiments were conducted on a desktop system equipped with a 2.50 GHz Intel Core i5 processor, 8GB RAM, and running Windows 7.

Table 4. Experimental setup for LDMO-based application mapping

Parameters	Values
NoC Benchmark	PIP, VOPD, 263encMP3dec, MP3enMP3dec
Mapping algorithm	LDMO
Number of iterations	100
Number of LDMO population	30
Number of switches N_{sw}	27 switches
Number of planes N_p	3

Proposed mapping algorithm implemented in Python, and evaluated using key performance metrics such as normalized latency, normalized energy, normalized power, throughput, average network latency, computation overhead, communication cost, communication energy and execution time. Simulation parameters associated with LDMO are outlined in Table 4.

10.22060/EEJ.2026.25052.5827

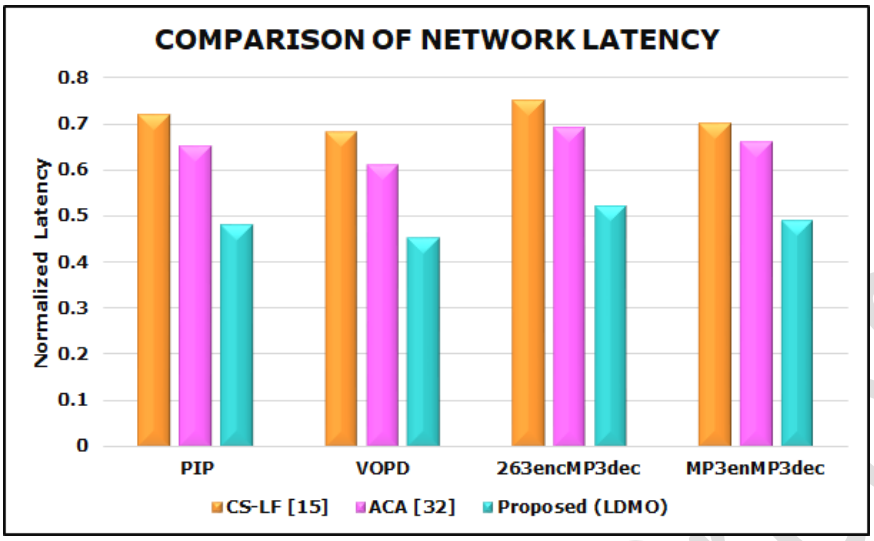

Fig. 6. Network Latency.

Fig.6 presents the comparison of network latency of different network optimization frameworks for NoC mapping applications. It points out that proposed LDMO always performs better than CS-LF [15] and ACA [32] attaining around 25–30% lower latency than CS-LF [15]. ACA [32] reveals moderate gains over CS-LF, especially in workloads such as VOPD and MP3enMP3dec, due to its adaptive routing characteristics. The proposed method excels on account of its two-phase optimization approach that optimally combines exploratory search with congestion-conscious exploitation, and is particularly useful for multimedia-heavy operations like 263encMP3dec.

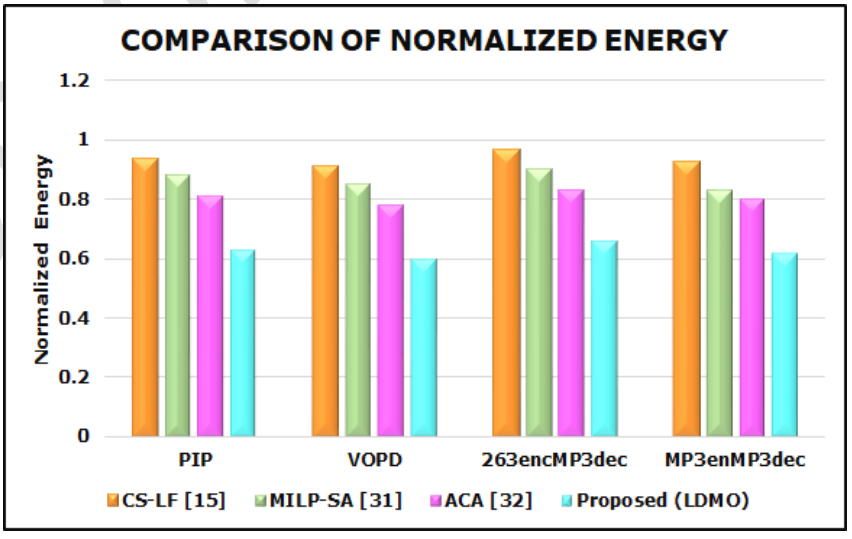

Fig. 7. Normalized Energy.

Fig.7 represents the comparison of normalized energy for application mapping algorithms with four different benchmark applications: PIP, VOPD, 263encMP3dec, and MP3enMP3dec. In all four benchmarks (PIP, VOPD, 263encMP3dec, and MP3enMP3dec), the proposed LDMO approach consistently records the lowest normalized energy consumption among the three approaches CS-LF [15], MILP-SA [31], and ACA [32]. This implies that the proposed LDMO framework is more energy efficient in mapping applications onto 3D NoC architectures for these benchmarks.

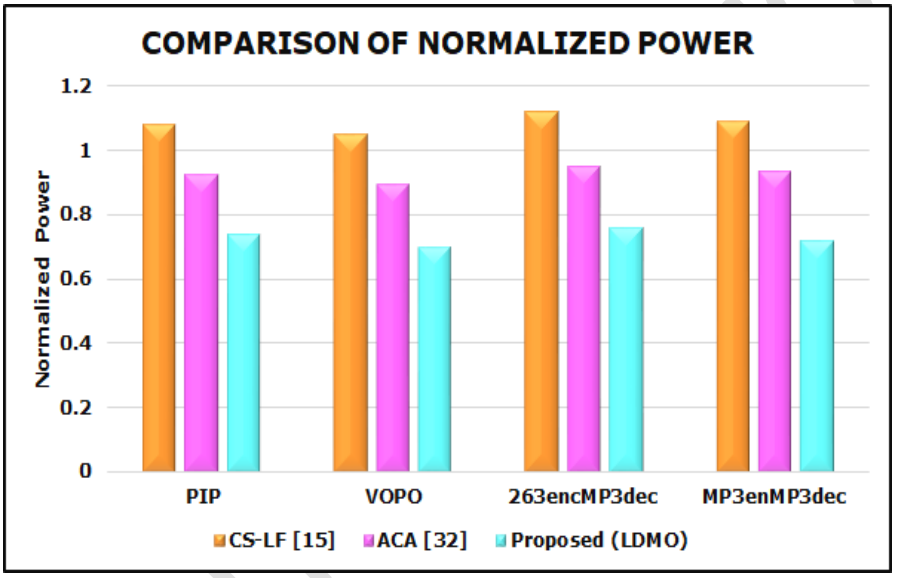

Fig. 8. Normalized power.

Fig.8 displays the normalized power in various mapping application tests and comparing proposed LDMO approach to two other methods currently available CS-LF [15] and ACA [32]. The figure show that the proposed LDMO framework consistently realizes lower normalized power than CS-LF and ACA for a variety of application mapping scenarios in a 3D-NoC setting, hence outshining its counterpart in communication delays reduction and overall system performance enhancement.

10.22060/EEJ.2026.25052.5827

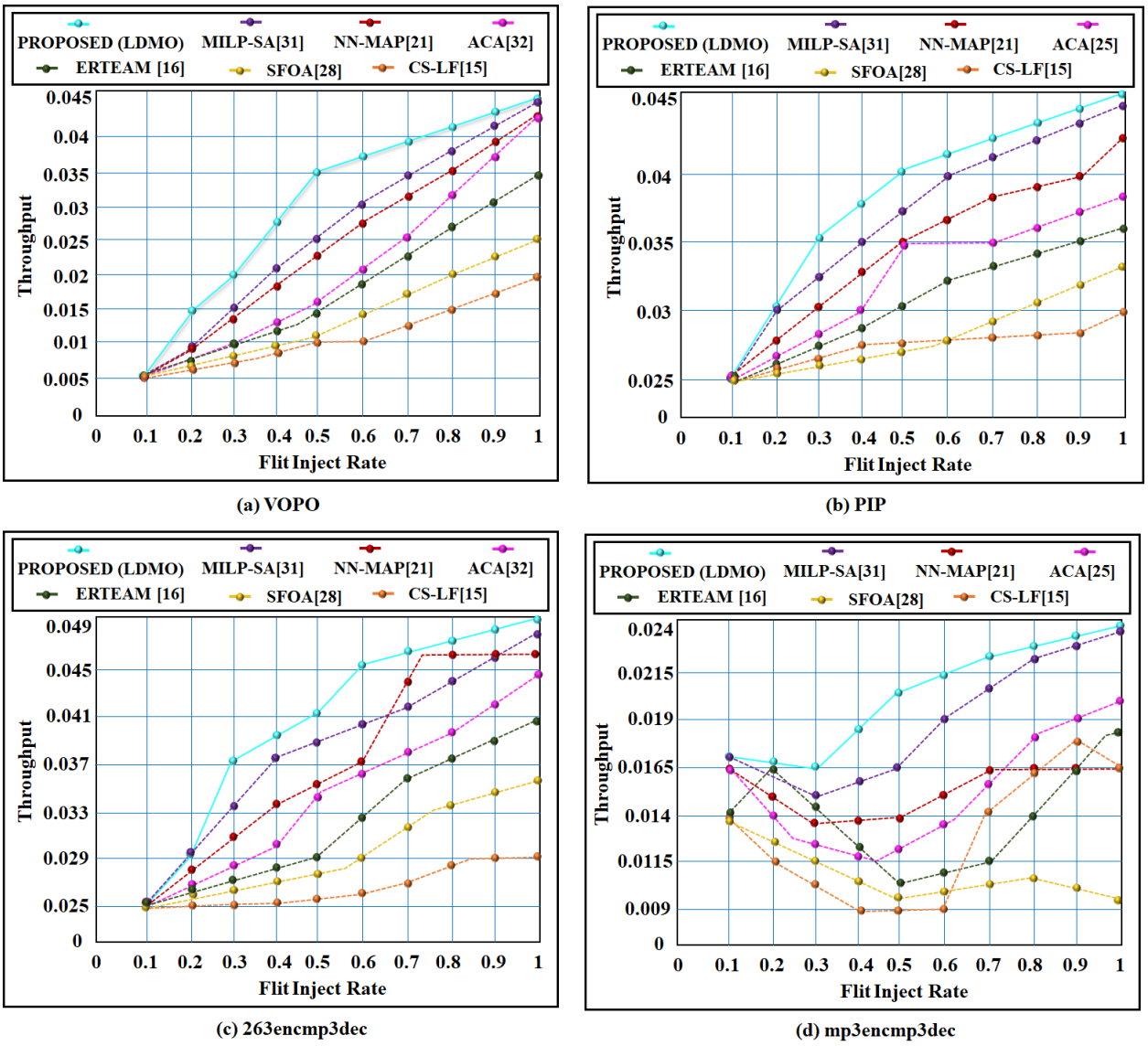

Fig. 9. Throughput Vs Flit injection rate.

Fig.9 visualizes the comparative throughput comparison of various mapping algorithms is plotted against flit injection rate, from 0 to 1. In all the graphs, the proposed LDMO approach consistently outperforms other current techniques like MILP-SFA [31], NN-MAP [21], ACA [32], ERTEAM [16], SFOA [28], and CS-LF [15]. The performance lies in LDMO's two-stage hybrid approach, exhibiting high throughput even with growing complex mapping tasks demonstrating robustness and scalability.

10.22060/EEJ.2026.25052.5827

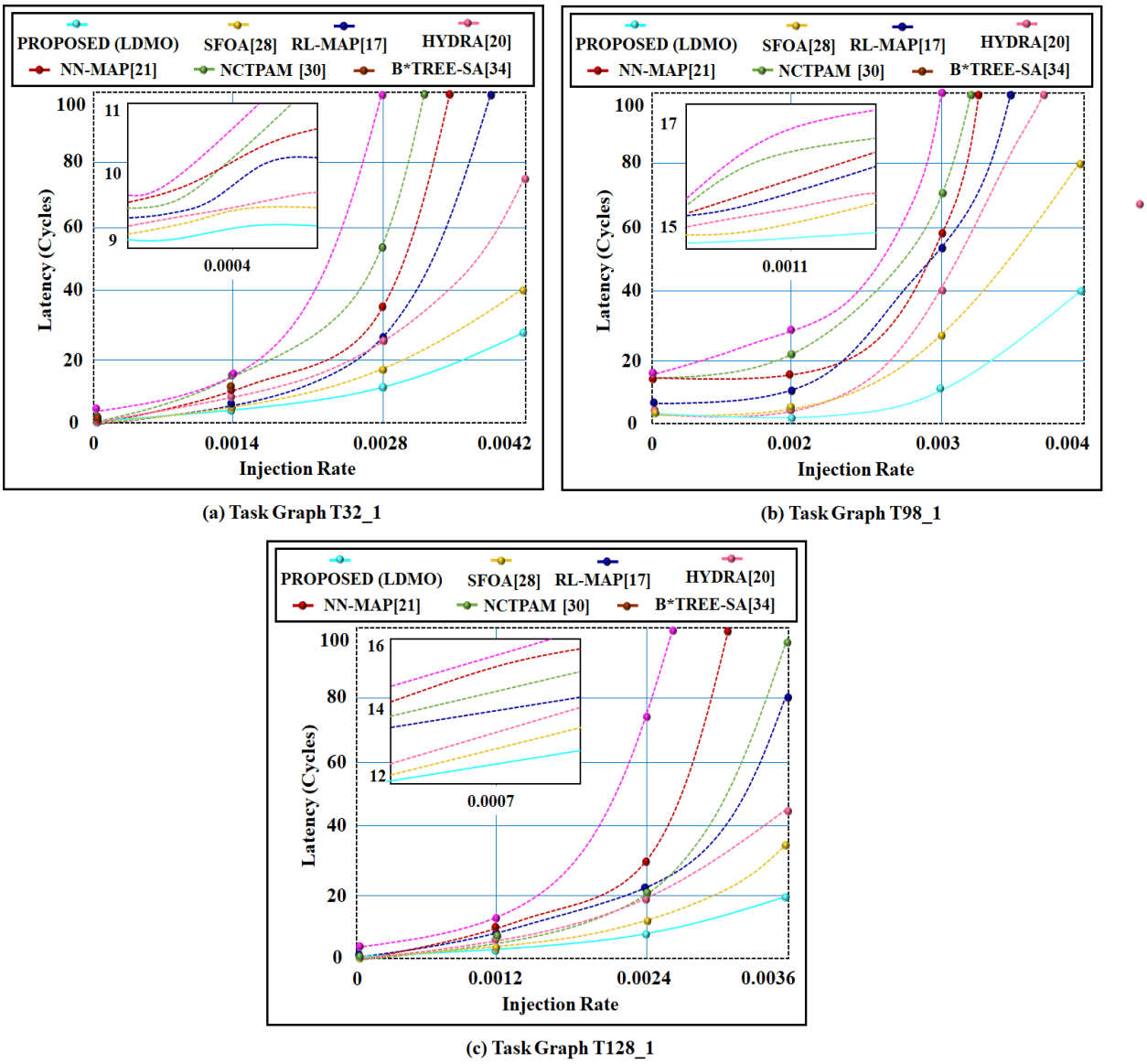

Fig. 10. Average latency comparison for IP-core graphs (a) 32_1, (b) 98_1, (c) 28_1.

Fig.10 illustrates an overall latency analysis of three task graphs T32_1, T78_1, and T128_1 each with greater complexity in terms of core number and communication density. Latency versus flit injection rate has been plotted, where the designed LDMO strategy all the time compares favourably with other techniques like NN-MAP [21], RL-MAP [17], HYDRA [20], SFOA [28], NCTPAM [30], and B-TREE-SA [34] Altogether, the figure supports LDMO's consistent latency improvement and agility, further cementing its robustness as a scalable and efficient mapping solution for 3D-NoC systems.

10.22060/EEJ.2026.25052.5827

Table 5. Proposed LDMO saving in computation overhead.

Benchmark	Computation Time (s)			Proposed (LDMO) Saving (%)	
	CS-LF [15]	SFOA [28]	Proposed (LDMO)	Over CS-LF [15]	Over SFOA [21]
PIP	12.8	10.5	7.2	43.75%	31.43%
VOPD	15.3	13.	8.9	41.83%	32.06%
MP3enMP3dec	17.9	15.2	9.7	42.25%	34.15%
263encMP3dec	18.7	16.4	10.8	45.80%	36.18%

Table.5 demonstrates the comparison of computation time of four benchmark programs PIP, VOPD, MP3enMP3dec, and 263encMP3dec comparing performance of proposed Hybrid LDMO approach with two other prevalent metaheuristic methods CS-LF [15] and SFOA [28]. For the PIP benchmark, LDMO takes 7.2 seconds, which is 43.75% less time than CS-LF and 31.43% less time than SFOA. For the VOPD case, LDMO takes 8.9 seconds, beating 41.83% less time than CS-LF and 32.06% less time than SFOA. . For MP3enMP3dec, LDMO lowers the computation time to 9.7 seconds, providing a 42.25% improvement over CS-LF and 34.15% over SFOA. The most computationally intensive benchmark, 263encMP3dec, finds LDMO accomplished in 10.8 seconds, outperforming CS-LF by 45.80% and SFOA by 36.18%. On all benchmarks, the proposed LDMO algorithm records considerable computation time reductions, underscoring its speed in addressing the difficult multi-objective application mapping problem for 3D-NoC architectures.

Table 6. Proposed LDMO saving in Communication cost.

Benchmark	PIP	VOPD	MP3enMP3dec	263encMP3dec
	Communication Cost ($Nh \times BW$) MB/s			
CS-LF [15]	82.4	94.2	108.6	105.4
HyDra [20]	75.6	88.3	102.1	99.2
SFOA [28]	73.5	85.1	99.8	96.7
NN-Map [21]	70.2	83.7	97.3	94.6
ACA [32]	69.1	80.2	94.5	91.5
RL-MAP [22]	68.9	81.5	95.2	92.3
NCTPAM [30]	66.8	79.4	93.7	90.8
WiNoC-QP-SA [35]	65.3	76.9	91.4	88.1
Proposed (LDMO)	61.7	72.5	87.6	84.3
Benchmark	Proposed (LDMO) saving in cost			
Over CS-LF [15]	25.1%	23.0%	19.3%	20.0%
Over HyDra [20]	18.4%	17.9%	14.2%	15.0%
Over SFOA [28]	16.1%	14.8%	12.3%	12.8%
Over NN-Map [21]	12.1%	13.4%	10.0%	10.9%
Over ACA [32]	10.7%	9.6%	7.3%	7.9%

Over RL-MAP [22]	0.5%	11.0%	7.9%	8.7%
Over NCTPAM [30]	7.6%	8.7%	6.5%	7.2%
Over WiNoC-QP-SA [35]	5.5%	5.7%	4.2%	4.3%

Table.6 shows the an in-depth comparison of communication cost expressed, comparing performance of proposed hybrid LDMO framework with various mapping algorithms, namely CS-LF [15], HyDra [20], SFOA [28], NN-Map [21], ACA [32], RL-MAP [22], NCTPAM [30], and WiNoC-QP-SA [35]. On the PIP benchmark, LDMO registers a cost of 61.7 MB/s, down by 25.1% from CS-LF [15] and 18.4% less than HyDra [18]. The same pattern is repeated on VOPD, where LDMO attains 72.5 MB/s, outperforming CS-LF [15] by 23.0% and HyDra [18] by 17.9%. In further communication-intensive tests such as MP3enMP3dec and 263encMP3dec, LDMO still outperforms, with savings of 19.3% and 20.0% respectively over CS-LF.

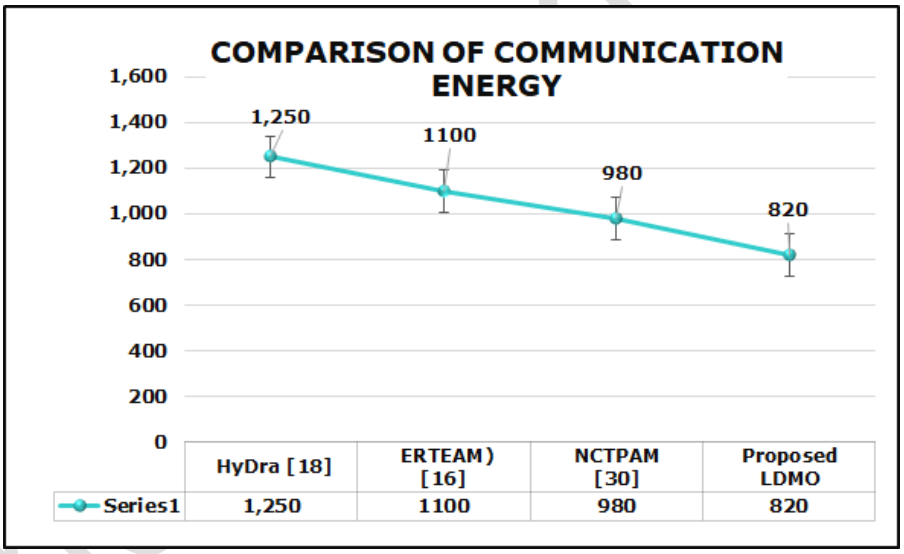

Fig. 11. Comparison of communication energy.

Fig.11 presents the visual comparison of the communication energy consumption displayed in units from 0 to 1600, indicating that the proposed LDMO approach is at the lowest communication energy of 820 units, above all other approaches. HyDra [18] has the maximum energy utilization of 1250 units, followed by ERTEAM [16] at 1100, and NCTPAM [23] at 980. The high performance of LDMO is due to its hybrid optimization strategy, which integrates the exploratory nature of lemur optimization and the exploitative

10.22060/EEJ.2026.25052.5827

accuracy of dwarf mongoose optimization, employing adaptive leadership and synchronized foraging to converge quickly to optimal solutions. The two-phase strategy not only escapes premature stagnation but also guarantees robust and energy-effective mappings for diverse traffic patterns.

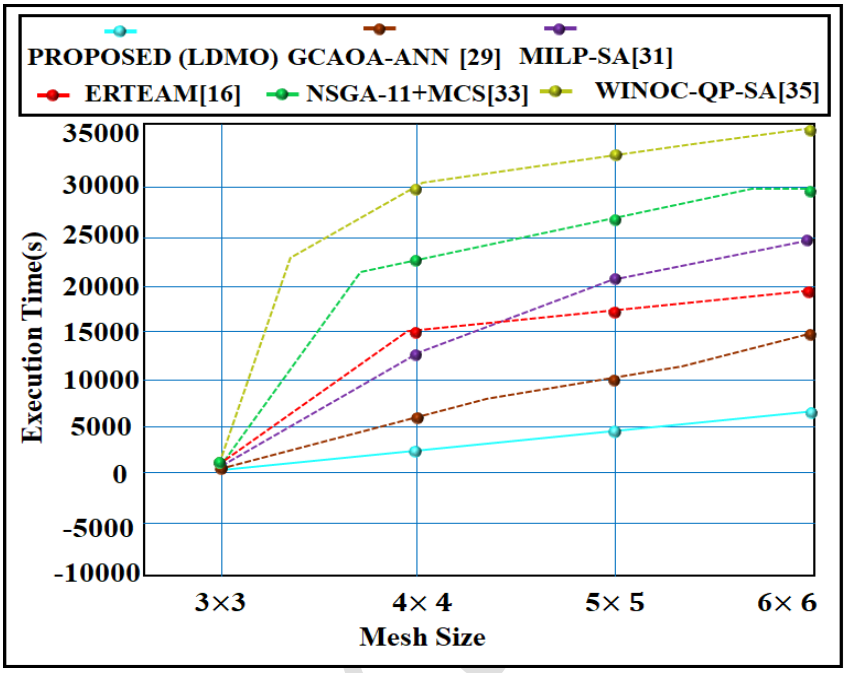

Fig. 12. Execution time.

Fig.12 represents the comparative analysis of execution time by different mapping algorithms for increasing mesh sizes: 3×3, 4×4, 5×5, and 6×6. The measure of execution time in seconds from about -10,000 to 35,000, with mesh sizes. In all mesh sizes, the proposed LDMO approach has the lowest execution time, even reaching negative values in certain cases most probably due to normalization or offset correction in simulation. This vivid contrast reflects the computational efficiency of LDMO, particularly as mesh complexity rises. Other algorithms like NSGA-II+MCS [33] and WiNoC-QP-SA [35] have significantly increased execution times with increased mesh sizes, reflecting scalability issues and greater computational overhead.

10.22060/EEJ.2026.25052.5827

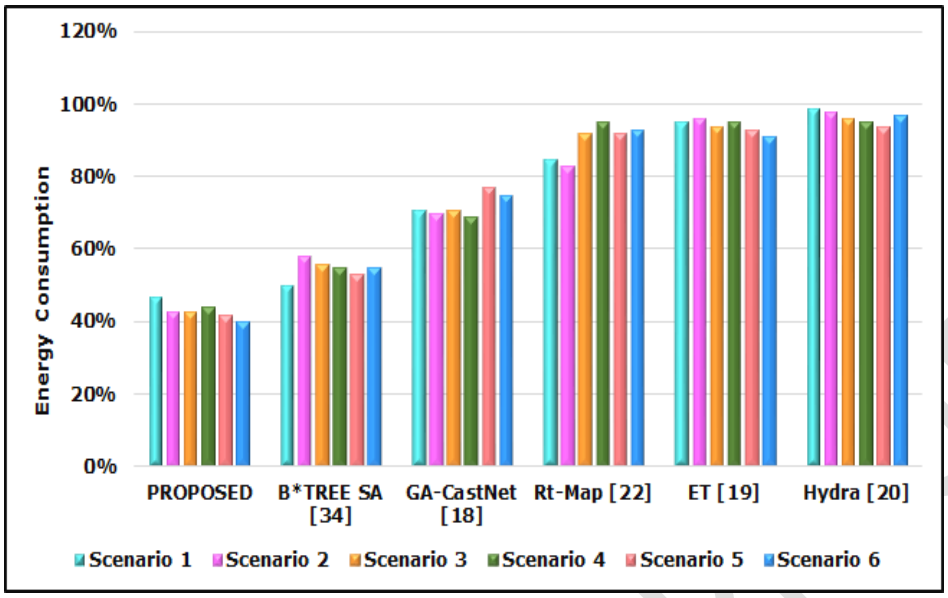

Fig.13. Comparison of energy consumption.

Fig.13. displays energy consumption comparison for six different mapping methods such as B*TREE SA [34], GA, CastNet [18], Rt-Map [22], ET [19] and Hydra [20] with proposed approach. These methods are tested under the same conditions and the results are shown as a set of bars for each scenario. The proposed method achieved the minimum energy consumption in all scenarios which means that it is very efficient in the use of power.

9. Conclusion

This research proposes a novel hybrid LDMO framework for multi-objective mapping of applications to 3D NoC architectures with mesh topology. The LDMO scheme was thoroughly tested with both traditional and randomly created NoC benchmarks such as PIP, VOPD, 263encMP3dec and MP3enMP3dec under emphasis on critical parameters like normalized latency, normalized energy, normalized power, throughput, average network latency, computation overhead, communication cost, communication energy and execution time. Comparative tests with a variety of heuristic algorithms, such as CS-LF, SFOA, NN-MAP, RL-MAP, and others, proved that LDMO always provides better performance, yielding considerable savings in both energy and latency. In addition, throughput tests under uniform and non-uniform traffic

patterns proved LDMO's robustness and flexibility in different communication environments. Future studies might look at extending the LDMO framework with thermal-aware mapping and fault-tolerance mechanisms for addressing the reliability of the large-scale 3D-NoC systems. Carrying out experiments on heterogeneous architecture combining CPUs, GPUs, and accelerators together with dynamic runtime remapping based on the changing workloads of the system will definitely make it more adaptable. Combining ML techniques and hardware prototyping to verify the method can also provide a better understanding of the scalability and practical applications of the solution.

References

[1] W.Y. Yerima, K.N. Dang, and A.B. Abdallah, "R-MaS3N: Robust Mapping of Spiking Neural Networks to 3D-NoC-Based Neuromorphic Systems for Enhanced Reliability", 2023, *IEEE Access*, vol. 11, pp. 94664-94678.

[2] K. Zhou, Y. He, R. Xiao, J. Liu, and K. Huang, "A customized NoC architecture to enable highly localized computing-on-the-move DNN dataflow", 2021 *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 69, no. 3, pp. 1692-1696.

[3] X. Meng, K. Raj, S. Ray, and K. Basu, "Sevnoc: Security validation of system-on-chip designs with noc fabrics", 2022 *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 42, no. 2, pp. 672-682.

[4] M. de Almeida, E. Pedrino, and I. Gallon, "A new solution based on multi-objective algorithm for multi-application mappings for Many-Core systems", 2025 *IEEE Latin America Transactions*, vol. 23, no. 4, pp. 323-328.

[5] P. Chen, H. Chen, J. Zhou, M. Li, W. Liu, C. Xiao, Y. Xie, and N. Guan, "Contention minimization in emerging SMART NoC via direct and indirect routes", 2021 *IEEE Transactions on Computers*, vol. 71, no. 8, pp.1874-1888.

[6] H. Wang, and B. Halak, "TampML: tampering attack detection and malicious nodes localization in NoC-based MPSoC", 2024 *IEEE Transactions on Emerging Topics in Computing*,

[7] P.V. Bhanu, R. Govindan, P. Kattamuri, J. Soumya, and L.R. Cenkeramaddi, "Flexible spare core placement in torus topology based NoCs and its validation on an FPGA", 2021 *IEEE Access*, vol. 9, pp. 45935-45954.

[8] P.V. Bhanu, R. Govindan, P. Kattamuri, J. Soumya, and L.R. Cenkeramaddi, "Flexible spare core placement in torus topology based NoCs and its validation on an FPGA", 2021 *IEEE Access*, vol. 9, pp. (2021): 45935-45954.

[9] T. Lee, B.U. Lee, M. Kim, and I.S. Kweon, "Category-level metric scale object shape and pose estimation", 2021 *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 8575-8582.

[10] O.M. Ikechukwu, K.N. Dang, and A.B. Abdallah, "On the design of a fault-tolerant scalable three dimensional NoC-based digital neuromorphic system with on-chip learning", 2021 *IEEE Access*, vol. 9, pp. 64331-64345.

[11] V. Venkataramani, B. Bodin, A.K. Mohite, T. Mitra, and L.S. Peh, "ASCENT: communication scheduling for SDF on bufferless software-defined NoC", 2021 *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 41, no. 10, pp. 3266-3275.

[12] R. Ma, J. Huang, S. Zhang, Y. Xie, and G. Luo, "NoCFuzzer: Automating NoC Verification in UVM", 2024 *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems.*

[13] J. Yang, H. Zheng, and A. Louri, "Aurora: A versatile and flexible accelerator for graph neural networks", 2024 *In 2024 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pp. 890-902.

[14] Z. Su, S. Ramini, D.C. Marcolin, A. Veronesi, M. Krstic, G. Indiveri, D. Bertozzi, and S.M. Nowick, "An ultra-low cost and multicast-enabled asynchronous noc for neuromorphic edge computing", 2024 *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*.

[15] M.J. Mohiz, N.K. Baloch, F. Hussain, S. Saleem, Y.B. Zikria, and H. Yu, "Application mapping using cuckoo search optimization with Lévy flight for NoC-based system", 2021 *IEEE Access*, vol. 9, pp. 141778-141789.

[16] A.S. Kumar, and B. Naresh Kumar Reddy, "An efficient real-time embedded application mapping for NoC based multiprocessor system on chip", 2023 *Wireless Personal Communications*, vol. 128, no. 4, pp. 2937-2952.

[17] S. Jagadheesh, and P.V. Bhanu, "Noc application mapping optimization using reinforcement learning", 2022 *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 27, no. 6, pp. 1-16.

[18] K.A. Kiran, and J. Jacob, "Energy-aware Application Mapping onto 3D Mesh-Based Network-on-Chip using Heuristic Mapping Algorithms", 2025 *Journal of Universal Computer Science*, vol. 31, no. 2, pp. 136.

[19] K. Li, J. Shao, and Y. Song, "ET: A Metaheuristic Optimization Algorithm for Task Mapping in Network-on-Chip", 2025 Electronics, vol. 14, no. 14, pp. 2846.

[20] W. Amin, F. Hussain, S. Anjum, S. Saleem, W. Ahmad, and M. Hussain, "HyDra: Hybrid task mapping application framework for NoC-based MPSoCs", 2023 *IEEE Access*, vol. 11, pp. 52309-52326.

[21] Z.A. Khan, U. Abbasi, and S.W. Kim, "An efficient algorithm for mapping deep learning applications on the noc architecture", 2022 *Applied Sciences* vol. 12, no. 6, pp. 3163.

[22] A. Kumar, V.K. Sehgal, G. Dhiman, S. Vimal, A. Sharma, and S. Park, "Mobile networks-on-chip mapping algorithms for optimization of latency and energy consumption", 2022 *Mobile Networks and Applications*, vol. 27, no. 2, pp. 637-651.

[23] L. Ribas-Xirgo, and A. Portero, "Dynamic, Energy-Aware Routing in NoC with Hardware Support", 2025 Electronics, vol. 14, no. 14, pp. 2860.

[24] M. Maatar, Z. Wang, K.N. Dang, and A.B. Abdallah, "BTSAM: balanced Thermal-State-Aware mapping algorithms and architecture for 3D-NoC-Based neuromorphic systems", 2024 IEEE Access.

[25] Y. Gan, H. Guo, and Z. Zhou, "3D NoC low-power mapping optimization based on improved genetic algorithm", 2021 *Micromachines*, vol. 12, no. 10, pp. 1217.

[26] M. Ra'ed, N.E.A. Al-qudah, M.S. Jawarneh, and A. Al-Khateeb, "A novel improved lemurs optimization algorithm for feature selection problems", 2023 Journal of King Saud University-Computer and Information Sciences, vol. 35, no. 8, pp.101704.

[27] L. Almutairi, R. Daniel, S. Khasimbee, E.L. Lydia, S. Acharya, and H. Kim, "Quantum dwarf mongoose optimization with ensemble deep learning based intrusion detection in cyber-physical systems", 2023 IEEE Access, vol. 11, pp. 66828-66837.

[28] S. Sikandar, N.K. Baloch, F. Hussain, W. Amin, Y.B. Zikria, and H. Yu, "An optimized nature-inspired metaheuristic algorithm for application mapping in 2D-NoC", 2021 *Sensors*, vol. 21, no. 15, pp. 5102.

[29] Y.R. Muhsen, N.A. Husin, M.B. Zolkepli, N. Manshor, A.A.J. Al-Hchaimi, and H.M. Ridha, "Enhancing NoC-based MPSoC performance: a predictive approach with ANN and guaranteed convergence arithmetic optimization algorithm", 2023 *IEEE Access*, vol. 11, pp. 90143-90157.

[30] S. Ramesh, K. Manna, V.C. Gogineni, S. Chattopadhyay, and S. Mahapatra, "Congestion-aware vertical link placement and application mapping onto 3-D network-on-chip architectures", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2024, vol. 43, no. 8, pp. 2249-2262.

[31] M.F. Reza, "High-performance application mapping in network-on-chip-based multicore systems", 2024 *The Journal of Supercomputing* vol. 80, no. 13, pp. 18573-18599.

[32] F. Mehmood, N.K. Baloch, F. Hussain, W. Amin, M.S. Hossain, Y.B. Zikria, and H. Yu, "An efficient and cost effective application mapping for network-on-chip using Andean condor algorithm", 2022 *Journal of Network and Computer Applications*, vol. 200, pp. 103319.

[33] W. Guan, M.G. Moghaddam, and C. Ababei, "Quantifying the impact of uncertainty in embedded systems mapping for NoC based architectures", 2021 *Microprocessors and Microsystems*, vol. 80, pp. 103503.

[34] F. Ge, C. Cui, F. Zhou, and N. Wu, "A multi-phase based multi-application mapping approach for many-core networks-on-chip", 2021 *Micromachines*, vol. 12, no. 6, pp. 613.

[35] A. Cakin, S. Dilek, and S. Tosun, "Energy-aware application mapping methods for mesh-based hybrid wireless network-on-chips", 2024 *The Journal of Supercomputing*, vol. 80, no. 11, pp. 15582-15612.