

RESEARCH REVIEW REPORT

MTDA: Efficient and Fair DPU Offloading Method for Multiple Tenants

Rohit Bankar - rbankar@purdue.edu

I. PAPER EVALUATION

A. Research Objectives & Problem Statement

- Multi-tenant clouds share a single DPU, so tenants compete for cores/accelerators → throughput drops, latency spikes, and per-tenant performance becomes erratic.
- Goal: provide **stable, fair, and efficient** DPU offloading across diverse apps/traffic by isolating tenants and allocating DPU internals proportionally to demand.

B. Advancing State-of-the-Art

- Prior systems:
 - **DPDK/iPipe**: better raw offload but no fairness; large fluctuations under contention.
 - **FairNIC**: strict static partitioning → isolation but poor adaptability to unbalanced load.
 - **LogNIC**: model-driven allocation needs offline characterization; struggles with dynamic demand.
- **MTDA** contributes: (i) per-tenant **virtual channels**, (ii) a **credit-based allocation** that reflects actual needs (requests, core usage, accelerator needs), and (iii) a **traffic-aware scheduler** that reallocates credits on the fly.

C. Proposed Solution and Key Insights

- **Independent virtual channels** (by TenantID) prevent cross-tenant queue head-of-line blocking before requests hit HW.
- **Credit model**: abstracts DPU cores/accelerators as **credits**; tenant i's credits grow with its request rate and required HW.
- **Traffic-aware scheduling**: monitors channels; on **EXHAUSTED** or **EXPIRED** events, runs credit reallocation.

D. Experimental Setup and Results

- Evaluated on a BlueField-2 DPU testbed with multiple tenants (analytics, routing, firewall) under balanced and skewed traffic, comparing against DPDK/iPipe/FairNIC/LogNIC.
- MTDA consistently raised throughput and cut median/p99 latency across tenants, with notably better stability under load imbalance. Overhead versus hand-tuned allocation was small (single-digit percent).

E. Open Problems / Limitations

- Joint optimization of DPU compute with memory/NIC bandwidth is missing, and portability beyond the tested hardware remains to be shown.
- Credit formulas and thresholds may need workload-specific tuning, and integration with cluster-level fairness and network scheduling is still open.

II. PAPER DISCUSSION

A. Future Research Directions

- **Multi-resource fairness:** extend credits to **DRAM BW** and **NIC BW**; integrate with network-fairness (e.g., SoftBW/Falloc-style policies).
- **RL/online learning:** learn credit-reallocation and time-slice policies per-tenant to optimize Fair1Fair_1Fair1/Fair2Fair_2Fair2, p99, and throughput jointly.
- **Heterogeneous DPUs / multi-cluster:** topology-aware crediting across accelerators, NUMA, and cross-DPU placements.

B. Abstractions OR Niche

- **Abstraction:** “credit-as-a-resource” API: (requests, core_occupancy, accel_vector) → credits → timeslice with fairness targets; export Fair1Fair_1Fair1/Fair2Fair_2Fair2/p99 as control signals for higher-level schedulers.
- **Niche:** noisy multi-tenant edges and micro-DC pods where strict isolation (FairNIC) wastes headroom and reactive schedulers (iPipe) oscillate.

C. Research Connections

- Complements **FairNIC** (isolation) by adding **adaptive fairness**; improves on **LogNIC** by eliminating the offline characterization requirement.
- Can feed **DeepScaling/CORE-ML** style controllers with fair credits per tenant to coordinate vertical/horizontal scaling upstream.