# A Hybrid GA-Game-Theoretic Approach for Joint UPF Placement and Traffic Routing in Next Generation Networks

Mohammed Dhiya Eddine Gouaouri*, Miloud Bagaa*, Messaoud Ahmed Ouameur*, Bertrand Hugo†, Daniel Massicotte*, Adlen Ksentini‡
*Dept. of Electrical and Computer Engineering, Université du Québec à Trois-Rivières, Canada
{mohammed.dhyia.eddine.gouaouri, miloud.bagaa, messaoud.ahmed.ouameur, daniel.massicotte}@uqtr.ca
†Centre de recherche de recherche d'Hydro-Québec, Montréal, QC
bertrand.hugo@hydroquebec.com
‡EURECOM, Campus SophiaTech, France
adlen.ksentini@eurecom.fr

***Abstract*—The 5G core network relies on flexible deployment of User Plane Functions (UPFs) across geo-distributed edge servers to meet latency and efficiency demands. We study the joint optimization of UPF placement and traffic routing, aiming to minimize deployment cost, energy consumption, and user-plane latency. The problem is modeled as a bilevel optimiztion program, where the upper level determines UPF placement, while the lower level assigns gNBs to UPFs and decides traffic load balancing scheme between UPFs. Due to the problem's NP-hardness, we design two hybrid approaches: (i) GA-MILP, which couples a genetic algorithm with exact MILP-based routing, and (ii) GA–game-theoretic, which replaces the MILP with matching and auction models for scalability. Simulations show that GA-MILP achieves the most cost- and energy-efficient deployments, while the game-theoretic variants provide faster convergence with competitive latency performance.**

***Index Terms*—5G, Edge Computing, User Plane Function, Optimization, Genetic Algorithms, Game Theory**

## I. Introduction

The 5G mobile network marks a transformative shift in wireless communication, offering unprecedented capabilities in terms of data rates, latency, and device connectivity. With enhanced mobile broadband, ultra-reliable low-latency communication, and massive machine-type communications, 5G enables emerging applications such as autonomous vehicles, industrial automation, smart cities, and immersive media. To meet these diverse requirements, 5G adopts a flexible architecture centered around Network Function Virtualization (NFV) and edge computing, which decouples network functions from dedicated hardware and implements them as VNFs on commodity servers [1]. This enables dynamic instantiation, scaling, and migration of core functions, such as the User Plane Function (UPF), closer to the edge where latency-sensitive services demand low round-trip times. The service-based architecture of the 5G core further allows network slices to be deployed and optimized independently, making NFV a foundational technology for agile and responsive next-generation networks.

In this work, we address the joint optimization of UPF deployment and traffic routing in geo-distributed 5G edge networks. We focus on two UPF types: Intermediate-UPFs (I-UPFs) near gNBs for initial traffic processing, and Anchor-UPFs (A-UPFs) forwarding traffic to the Data Network (DN). The objective is to minimize deployment cost, energy consumption, and latency by selecting energy-efficient edge servers, associating gNBs with optimal I-UPFs, and balancing traffic across A-UPFs. We formulate the problem as a bilevel optimization model: the upper-level is an integer linear program (ILP) for UPF placement, while the lower-level is a mixed-integer linear program (MILP) for gNB assignment and traffic routing. To overcome computational intractability, we propose two hybrid approaches: (i) GA-MILP, which combines a genetic algorithm (GA) with exact optimization, and (ii) GA–game-theoretic, where gNB assignment is modeled as a matching game and traffic routing as an auction game. The main contributions are:

- A bilevel optimization model for joint UPF placement and traffic routing under various traffic demands.
- Two hybrid algorithms: GA-MILP and GA–game-theoretic, balancing optimality and scalability.
- Extensive simulations evaluating convergence, efficiency, and scalability against state-of-the-art baselines.

## II. Related Works

The deployment of 5G core user plane functions (UPFs) has received growing attention. Recent works apply deep reinforcement learning (DRL) for scaling UPF instances [2], or formulate multi-objective ILPs for joint UPF placement and chaining [3], [4]. These approaches improve cost and response time but generally assume static traffic and overlook energy efficiency. Our problem is also related to the broader literature on VNF placement and routing (VNF P&R). Several studies formulate ILP-based models to minimize energy [5], or combine approximation and matching theory to incorporate traffic-aware costs [6]. Others extend the model with delay or power constraints [7], or adopt decomposition methods such as Benders [8]. Learning-based approaches have also been explored, including DRL and multi-agent methods [9], [10], but these often face scalability and convergence issues in large-scale networks. Dynamic variants have been studied

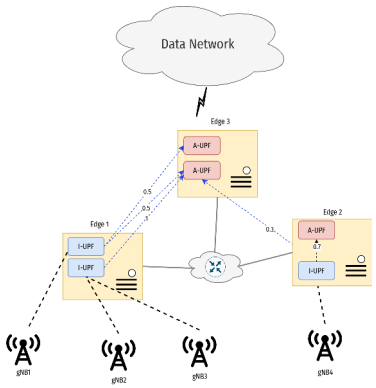


Figure 1: Architectural diagram of the 5G core network user plane deployment in the edge network.

using online scaling [11] or finite-horizon placement with heuristics [12], yet most lack performance guarantees. In contrast, we focus on the dynamic deployment of the 5G user plane in edge environments under time-varying demands. We jointly optimize I-UPF and A-UPF placement with traffic routing, formulating the problem as a bilevel model. Unlike prior DRL or heuristic methods, our algorithm couples upper-level placement with lower-level routing dynamics, achieving scalable and latency-aware optimization for edge-enabled 5G networks. In addition, our proposed method aims to provide guaranteed constraint satisfaction plus the avoidance of extensive trial-and-error training required by DRL methods.

## III. Problem formulation

In this section, we present the system model and formulate the UPF placement and traffic routing problem for 5G core network user plane.

### A. System Model

Figure 1 illustrates the 5G core user plane deployed over an edge network composed of a set of base stations (gNBs) $\mathcal{G}$ and edge servers $\mathcal{N}$. Each edge server acts as a potential hosting node for UPFs, either Intermediate-UPFs (I-UPFs) or Anchor-UPFs (A-UPFs). gNBs connect to edge servers via high-speed backhaul links (fiber or wireless). At each time slot $t$, gNB $g \in \mathcal{G}$ handles an aggregated traffic load $T_g$. Each edge server $n \in \mathcal{N}$ has limited compute and memory capacities denoted by $C_n^{\text{cpu}}$ and $C_n^{\text{mem}}$. I-UPFs serve as access-level interconnection points, while A-UPFs act as PDU session anchors providing connectivity to the data network (DN). Each I-UPF can connect to multiple A-UPFs for traffic routing and load balancing. All UPFs are assumed homogeneous: their resource demands are $c_I^{\text{cpu}}, c_I^{\text{mem}}$ for I-UPFs and $c_A^{\text{cpu}}, c_A^{\text{mem}}$ for A-UPFs, respectively, where $c_I^{\text{cpu}}$ represents the number of packets processed per time unit.

### B. Problem formulation

We aim to jointly optimize the cost of UPF deployment and traffic routing. First, a network operation decides the number and location of all types of UPF instances to minimize deployment cost and energy consumption of edge servers. Based on that decision, a second optimization step aims to decide how to assign gNBs to the I-UPFs and the load balancing scheme between the I-UPFs and the selected A-UPFs.

The decision variables for UPF placement and traffic routing include:

The placement decision variable of I-UPFs is defined as

$$x_{i,n} = \begin{cases} 1, & \text{if I-UPF } i \in \mathcal{I} \text{ is deployed on edge server } n \in \mathcal{N} \\ 0, & \text{otherwise} \end{cases}$$

The placement decision variable of A-UPFs is defined as:

$$y_{j,n} = \begin{cases} 1, & \text{if A-UPF } j \in \mathcal{A} \text{ is deployed on edge server } n \in \mathcal{N} \\ 0, & \text{otherwise} \end{cases}$$

We also define the gNB-to-I-UPF assignment variable as:

$$z_{g,i} \begin{cases} 1, & \text{if gNB } g \in \mathcal{G} \text{ is assigned to I-UPF } i \in \mathcal{I} \\ 0, & \text{otherwise} \end{cases}$$

For load balancing traffic from I-UPFs to A-UPFs, we define a variable for traffic splitting weight denoted by $r_{i,j} \in [0, 1]$

*1) Deployment cost:* We assume that all I-UPF and A-UPF instances are associated with their deployment cost which can be quantified by the startup time of the instance. We assume that the deployment cost is independent of the location of UPF deployment, hence we use the total number of deployed UPF instances denoted by $\mathcal{J}_{\text{cost}}$, an equivalent deployment cost proxy model. This cost is defined as:

$$\mathcal{J}_{\text{cost}} = \sum_{i \in I} \sum_{n \in N} x_{i,n} + \sum_{j \in A} \sum_{n \in N} y_{j,n} \tag{1}$$

*2) Energy consumption:* As stated in some earlier works, the CPU is the main power consumption component on a server [13]. Following prior work [14], [15], we adopt a linear CPU utilization-based power model. Thus, the power consumption of server $i \in \mathcal{I}$ is given by:

$$P(u_n^{CPU}) = P_n^{\text{idle}} + (P_n^{\text{peak}} - P_n^{\text{idle}}) \cdot u_n^{\text{cpu}} \tag{2}$$

Where $P_n^{\text{idle}}$ and $P_n^{\text{peak}}$ denote the idle power and the peak power of edge server $n \in \mathcal{N}$, $u_n^{\text{cpu}}$ denotes the CPU utilization of edge server $n$. Hence the total energy consumption of edge servers is defined as follows:

$$\mathcal{J}_{\text{energy}} = \sum_{n \in \mathcal{N}} \left( \sum_{i \in \mathcal{I}} x_{i,n} \cdot P(c_I^{\text{cpu}}) + \sum_{j \in \mathcal{A}} y_{j,n} \cdot P(c_A^{\text{cpu}}) \right) \tag{3}$$

*3) User plane latency:* The gNBs are connected to the edge servers via optical fiber or millimeter wave backhaul links. The backhaul delays between the gNBs and the edge servers are proportional to the geographical distances. We denote by $d_{g,n}$ the backhaul delay between gNB $g \in \mathcal{G}$ and edge server $n \in \mathcal{N}$. Then the user plane latency can be expressed as:

$$\mathcal{J}_{\text{latency}} = \frac{1}{|\mathcal{G}|} \sum_{\substack{g \in \mathcal{G} \\ i \in \mathcal{I} \\ j \in \mathcal{A} \\ n, n' \in N}} z_{g,i} \cdot x_{i,n} \cdot y_{j,n'} \cdot T_g \cdot (d_{g,n} + r_{i,j} \cdot d_{n,n'} + \frac{1}{c_I^{\text{cpu}}} + \frac{r_{i,k}}{c_A^{\text{cpu}}}) \tag{4}$$

$$\min_{x,y} \quad \alpha \mathcal{J}_{\text{cost}} + (1-\alpha)\mathcal{J}_{\text{energy}} \tag{5a}$$

$$\text{s.t.} \quad \sum_{n\in\mathcal{N}} x_{i,n} \le 1 \forall i \in \mathcal{I}, \tag{5b}$$

$$\sum_{n\in\mathcal{N}} y_{j,n} \le 1 \forall j \in \mathcal{A}, \tag{5c}$$

$$\sum_{i\in\mathcal{I}} x_{i,n} c_I^{\text{cpu}} + \sum_{j\in\mathcal{A}} y_{j,n} c_A^{\text{cpu}} \le C_n^{\text{cpu}} \forall n \in \mathcal{N}, \tag{5d}$$

$$\sum_{i\in\mathcal{I}} x_{i,n} c_I^{\text{mem}} + \sum_{j\in\mathcal{A}} y_{j,n} c_A^{\text{mem}} \le C_n^{\text{mem}} \forall n \in \mathcal{N}, \tag{5e}$$

$$\mathcal{J}^*_{\text{latency}} \le |\mathcal{G}| \cdot L_{\max}, \tag{5f}$$

$$\mathcal{J}^*_{\text{latency}} = \min_{z,r} \quad \mathcal{J}_{\text{latency}}, \tag{5h}$$

$$\sum_{i\in\mathcal{I}} z_{g,i} = 1 \forall g \in \mathcal{G},$$

$$z_{g,i} \le \sum_{n\in\mathcal{N}} x_{i,n} \forall g \in \mathcal{G}, i \in \mathcal{I},$$

$$r_{i,j} \le \sum_{n\in\mathcal{N}} (x_{i,n} + y_{j,n}) \forall i \in \mathcal{I}, j \in \mathcal{A},$$

$$\sum_{j\in\mathcal{A}} r_{i,j} = 1 \forall i \in \mathcal{I}$$

Figure 2: Bi-level optimization formulation of UPF placement and traffic routing.

The expression of $\mathcal{J}_{\text{latency}}$ is the average end-to-end delay experienced by traffic flows from the gNBs through the deployed UPFs to the DN:

- The quantity $\frac{1}{c_I^{\text{cpu}}}$ denotes the processing delay at the I-UPF, modeled as the inverse of the available CPU capacity $c_I^{\text{cpu}}$ allocated for processing packets.
- The quantity $\frac{r_{i,j}}{c_A^{\text{cpu}}}$ denotes the processing delay at the A-UPF $j$, weighted by the fraction $r_{i,j}$ of the traffic sent to it, and inversely proportional to its CPU capacity $c_A^{\text{cpu}}$.

*4) Optimization objective:* We formulate the UPF placement and traffic routing problem for 5GC network user plane as a bilevel optimization problem as shown in Figure 2.

This formulation captures the hierarchical nature of decisions involved in UPF deployment and traffic routing in 5G networks. The upper-level problem determines the number and placement of I-UPF and A-UPF instances on candidate nodes in the infrastructure. The objective is to minimize a weighted combination of the deployment cost $\mathcal{J}_{\text{cost}}$ and energy consumption $\mathcal{J}_{\text{energy}}$, governed by a trade-off parameter $\alpha$.

The upper-level constraints are the following:

- Each I-UPF and A-UPF instance can be placed on at most one candidate node.
- The cumulative compute resources consumed by the deployed UPFs cannot exceed the capacity of any node.
- The end-to-end average latency achieved through the lower-level problem must satisfy the maximum latency constraint $L_{\max}$ per gNB.

After deciding the number and location of UPFs, the **lower-level problem** aims to assign gNBs to I-UPFs and splitting traffic from I-UPFs to A-UPF. Specifically:

- Each gNB must be assigned to exactly one deployed I-UPF.
- A gNB can only be assigned to an I-UPF that has been placed by the upper level.
- Similarly, traffic splitting from I-UPFs to A-UPFs is only allowed towards deployed A-UPFs.
- All the traffic of an I-UPF must be fully split among the deployed A-UPFs.

The lower-level objective $\mathcal{J}_{\text{latency}}$ minimizes the total end-to-end latency, accounting for transmission delays between nodes and processing delays at the UPFs.

This bilevel formulation reflects the separation between strategic placement decisions at the upper level and operational routing decisions at the lower level, as observed in real-world 5G architectures. The upper-level decisions impact the feasibility and performance of the lower-level routing problem, while the lower-level solution provides feedback on the quality of placement through its impact on latency.

Since the problem is NP-hard and intractable, searching for an optimal solution within a bounded time is not practical, especially for large scale problem. Hence a faster approach needs to be proposed.

## IV. A Hybrid Optimization Approach with Genetic Algorithm and MILP

Genetic Algorithms (GAs) are metaheuristic search methods inspired by natural selection [16]–[18]. We use GA for the upper-level UPF placement, while the lower-level gNB assignment and traffic routing is solved using MILP solvers such as Gurobi. This coupling balances global exploration with precise evaluation. Each solution is encoded as two binary matrices: $x \in \{0,1\}^{|\mathcal{I}|\times|\mathcal{N}|}$ for I-UPFs and $y \in \{0,1\}^{|\mathcal{A}|\times|\mathcal{N}|}$ for A-UPFs, with each row indicating the placement of one UPF. The initial population of size $B$ is generated randomly, and a repair operator enforces feasibility by ensuring exactly one placement per UPF and respecting node capacity constraints. For each $(x, y)$, the lower-level MILP is solved to obtain routing variables $(z, r)$ and latency. Constraint violations (multiple placements, capacity overloads, infeasible routing) are penalized. The fitness value is given by:

$$\text{fitness} = \alpha_1 \mathcal{J}_{\text{cost}} + \alpha_2 \mathcal{J}_{\text{energy}} + \alpha_3 \max\{0, \mathcal{J}_{\text{latency}} - L_{\max}\} + \text{penalty}$$

Lower fitness values correspond to better solutions. Elitism preserves the best individual each generation, while tournament selection (top 10%) balances convergence speed with diversity. Crossover is performed row-wise: each UPF inherits placement bits partly from one parent and partly from the other, followed by a repair step to restore feasibility. Mutation reassigns a UPF to a different valid node with probability $\mu$, introducing diversity and avoiding premature convergence. The GA runs for $M$ generations or stops earlier if a fully feasible solution is found.

## V. A Hybrid Optimization Approach with Genetic Algorithms and Game-Theoretic Models

While the GA–MILP framework provides high-quality solutions, its reliance on solving a MILP for each candidate limits scalability as the network size grows. To address this, we propose a second hybrid approach that retains GA for UPF placement but replaces the MILP with game-theoretic approximations. The lower level is decomposed into two subproblems: (i) gNB-to-I-UPF association via a matching game, and (ii)

I-UPF to A-UPF routing via an auction game. Both admit efficient distributed solutions, significantly reducing runtime while preserving the structure of the original bilevel model.

### A. gNB-to-I-UPF Assignment

The association problem resembles bin-packing: each gNB must connect to exactly one I-UPF, while I-UPFs may serve multiple gNBs subject to capacity. We formulate this as a many-to-one matching game, where gNBs act as *proposers* and I-UPFs as *acceptors*. Preferences are latency-driven: gNBs prefer the nearest I-UPFs, while I-UPFs prioritize gNBs with low expected load. The goal is to reach a *stable matching*, i.e., no gNB–I-UPF pair would both prefer to be matched to each other over their current assignments.

We adapt the deferred acceptance algorithm: in each round, unmatched gNBs propose to their most preferred feasible I-UPF. I-UPFs tentatively accept proposals up to their load capacity, ranking gNBs by increasing latency, and reject the excess. This process continues until no unmatched gNB can improve its assignment. The resulting assignment is stable and respects capacity constraints.

**Algorithm 1** gNB to I-UPF Matching with Capacity Constraints

**Input:** gNBs $\mathcal{G}$, I-UPFs $\mathcal{I}$, demands $T_g$, capacities $c_i$, latencies $d_{gi}$
**Output:** Assignment $z_{gi}$
Initialize unmatched set $\mathcal{S} = \mathcal{G}$
**while** $\mathcal{S} \neq \emptyset$ **do**
  Each $g \in \mathcal{S}$ proposes to next preferred I-UPF
  Each $i$ tentatively accepts lowest-latency gNBs within $c_i$, rejecting excess
  Update $\mathcal{S}$ with rejected gNBs
**end**
**return** $z$

This mechanism is computationally efficient (polynomial complexity in the number of gNBs and I-UPFs) and produces a distributed solution that can be executed locally if preference lists are known.

### B. I-UPF to A-UPF Routing

Once gNBs are associated with I-UPFs, the second subproblem is to route I-UPF traffic to A-UPFs. Each I-UPF must distribute its load across available A-UPFs, respecting capacity while minimizing delay. We model this as an *auction game*: I-UPFs act as *buyers* submitting bids proportional to delay and price, while A-UPFs act as *sellers* adjusting their unit price $p_j$ based on congestion.

At each iteration, every I-UPF $i$ solves a local linear program to determine its optimal split $r_{ij}$:

$$\min_{r_i} \quad f(r_i) = \sum_{j \in \mathcal{A}} r_{ij} \cdot l_i \cdot (d_{ij} + p_j) \tag{8}$$

$$\text{s.t.} \quad \sum_{j \in \mathcal{A}} r_{ij} = 1, \quad r_{ij} \geq 0 \quad \forall j \in \mathcal{A}. \tag{9}$$

Each A-UPF $j$ then updates its price according to its current load $L_j$:

$$p_j \leftarrow p_j \cdot \frac{L_j}{c_A^{\text{cpu}}}. \tag{10}$$

Over successive rounds, prices increase for overloaded A-UPFs and decrease for underutilized ones, balancing the traffic distribution. This iterative bidding process converges to an equilibrium routing that respects capacity and minimizes latency.

**Algorithm 2** Bidding Game for I-UPF to A-UPF Routing

**Input:** Assignments $z$, delays $d_{ij}$, A-UPF capacity $c_A$, demands $T_g$, max rounds, tolerance $\epsilon$
**Output:** Routing decision $r_i$
Initialize prices $p_j \leftarrow 0$
**for** *each round* **do**
  Compute I-UPF loads from $z, T_g$
  Each $i$ solves LP (8) given prices $p$
  Compute A-UPF loads $L_j$
  Update prices $p_j \leftarrow p_j \cdot L_j / c_A$
  **if** *prices change* $\leq \epsilon$ **then**
    break
  **end**
**end**
**return** $r$

*a) Conditional gradient descent-based traffic routing LP approximation:* In this paragraph, we describe a method to accelerate the search for a solution for the traffic split LP. The method is a gradient descent-based approach, but working on the constrained problem.

Although the per-I-UPF traffic split problem is a relatively small-scale LP, solving it exactly at each bidding iteration via a generic solver (such as simplex or interior-point) can become computationally expensive when the number of I-UPFs and A-UPFs grows, especially within the iterative price adjustment process. To further accelerate convergence, we employ the Frank–Wolfe (FW for short) conditional gradient method [19], [20] as an approximate solver for the traffic split LP. The FW method is particularly well-suited for our setting since the feasible set of $r_i$ is a simplex, defined by the constraints $\sum_{j \in \mathcal{A}} r_{ij} = 1$ and $r_{ij} \geq 0$.

At each FW iteration $k$, given the current traffic split vector $r_i^{(k)}$, we compute the gradient of the objective function with respect to $r_i$, namely

$$\nabla f(r_i^{(k)}) = l_i \cdot (d_{ij} + p_j)_{j \in \mathcal{A}},$$

which corresponds to the marginal cost of assigning additional traffic to each A-UPF. We then solve a linear minimization oracle over the simplex, which simply selects the A-UPF $j^\star$ with the minimum marginal cost and assigns all traffic to it, yielding the search direction $s_i^{(k)}$.

$$s_i^{(k)} \in \text{argmin}_{s_i \in \{r_i \,|\, \sum_{j \in \mathcal{A}} r_{ij} = 1 \text{ and } r_i \geq 0\}} \nabla f(r_i^{(k)}) \cdot s_i$$

The solution is updated via

$$r_i^{(k+1)} = r_i^{(k)} + \gamma_k \left( s_i^{(k)} - r_i^{(k)} \right)$$

Where $\gamma_k \in (0, 1]$ is a step size, chosen so that it is reduced at each iteration as $\gamma_k = \frac{2}{k+2}$.

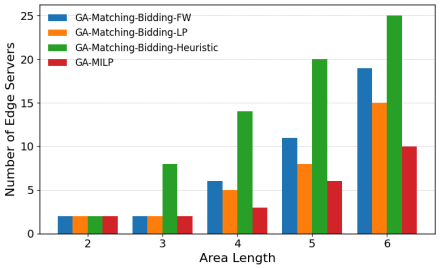


Figure 3: Deployment cost

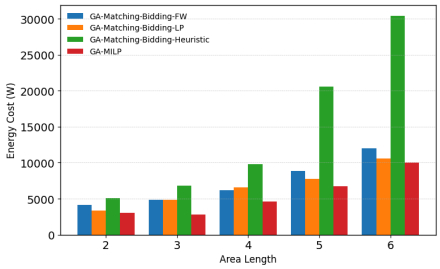


Figure 4: Energy cost

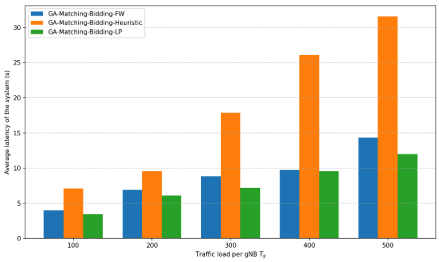


Figure 5: Average latency

## VI. Experimentation

We perform simulations to evaluate the proposed algorithms. gNBs and edge servers are distributed in a squared area ($x \times x$ km$^2$, where $x$ is the area length) area according to independent homogeneous Poisson point processes with densities of $1$ gNB/km$^2$ and $1$ server/km$^2$. Edge servers are homogeneous with identical resource demands, and their idle and peak power are uniformly sampled. Traffic load and propagation delays are generated randomly within given ranges. The main simulation parameters are summarized in Table I. The parameters of the GA are selected to balance solution quality and computational efficiency, particularly in light of the high computational complexity of the MILP formulation and the cost of fitness function evaluation.

Table I: Simulation Parameters

| Parameter | Value |
|---|---|
| gNB density in simulation area | 1 gNB/km$^2$ |
| Edge server density in simulation area | 1 server/km$^2$ |
| CPU capacity per server ($C_n^{\text{cpu}}$) | 32 cores |
| Memory capacity per server ($C_n^{\text{mem}}$) | 128 GB |
| I-UPF CPU demand ($c_I^{\text{cpu}}$) | 4 cores |
| A-UPF CPU demand ($c_A^{\text{cpu}}$) | 6 cores |
| Server idle power ($P_n^{\text{idle}}$) | $\mathcal{U}(200, 400)$ W |
| Server peak power ($P_n^{\text{peak}}$) | $\mathcal{U}(800, 1000)$ W |
| Traffic load per gNB ($T_g$) | $\mathcal{U}(200, 800)$ PDUs |
| Delay: gNB–server ($d_{g,n}$) | $\mathcal{U}(0.5, 3)$ ms |
| Delay: server–server ($d_{n,n'}$) | $\mathcal{U}(0.2, 1)$ ms |
| Max latency per gNB ($L_{\max}$) | 2000 ms |
| Trade-off weight ($\alpha$) | 0.5 |
| GA population size ($B$) | 20 |
| GA number of generations ($M$) | 10 |
| GA mutation probability ($\mu$) | 0.05 |
| GA tournament selection ratio | 10% |
| Max bidding game rounds | 20 |
| Tolerance parameter ($\epsilon$) | 0.001 |

The algorithms for evaluation and comparison are summarized as follows:

- **GA-MILP**: This algorithm is discussed in section IV. It optimizes the upper-level problem with genetic algorithm, and the lower-level MILP is solved optimally using Gurobi solver.
- **GA-Matching-Bidding-LP**: As discussed in section V. It optimizes the upper-level problem with GA, and decomposes the lower-level problem into two subproblems. The gNB assignment is solved using a matching game and the traffic routing is solved using a bidding game with exactly solving the LP in 8.
- **GA-Matching-Bidding-Heuristic**: Similar to the previous algorithm, but the LP is solved approximately using a heuristic. The heuristic greedily selects the set of A-UPFs to route traffic into and weigh them by the delay cost.
- **GA-Matching-Bidding-FW**: As discussed in V-B0a, it optimizes the per-I-UPF routing using the Frank-Wolfe conditional gradient descent method.

### A. Convergence Time Analysis

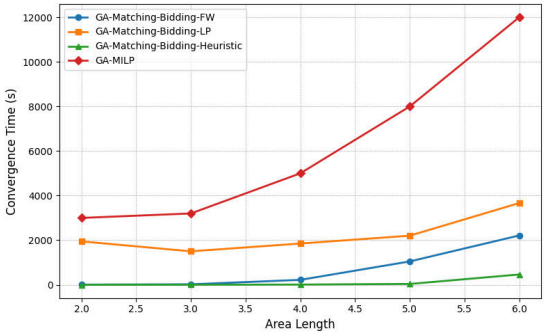


Figure 6: Convergence time analysis of the proposed algorithms

Figure 6 shows convergence time as the area size grows. GA-MILP exhibits the steepest growth (3,000–12,000 s), reflecting the cost of solving the MILP exactly. In contrast, GA-Matching-Bidding variants scale much better: the Heuristic is fastest, with sub-second runtimes for small to medium areas, while FW and LP provide intermediate performance. LP slightly outperforms FW on small cases but grows more quickly for larger scales. These results confirm that replacing the MILP with matching-based heuristics yields substantial scalability gains.

### B. Deployment Cost Analysis

Figure 3 shows deployment cost, measured as the number of active edge servers. GA-MILP achieves the most cost-efficient deployments by placing UPFs on the minimum number of servers while meeting constraints. In contrast, GA-Matching-Bidding-Heuristic incurs the highest cost, as its latency-focused strategy activates more servers. The GA-Matching-Bidding-FW and GA-Matching-Bidding-LP approaches lie in between, with FW slightly less efficient at larger scales.

### C. Energy cost analysis

Figure 4 depicts energy cost of the different algorithms. The proposed methods exhibit clear stratification. GA-MILP

achieves the lowest energy consumption across all area sizes, while GA-Matching-Bidding-LP remains within a narrow margin (5–20%) and displays the flattest growth with scale, indicating better asymptotic behavior under increasing spatial demand. FW consistently trails MILP and LP (20–30% higher at larger area lengths) reflecting residual sub-optimality in its approximate routing that leaves excess capacity active. In contrast, the heuristic degrades sharply, exceeding GA-MILP by more than a factor of two. Hence, GA-Matching-Bidding-LP offers a near-optimal, computationally attractive alternative with superior scaling, GA-Matching-Bidding-FW is comparable to GA-Matching-Bidding-LP but less efficient and the GA-Matching-Bidding-Heuristic is unsuitable for energy-constrained deployments.

### D. Average Latency Analysis

Figure 5 illustrates the evolution of average latency with increasing traffic load per gNB. The system latency is computed by aggregating traffic latency at each gNB and averaging over all gNBs. In this experiment, the area length is set to $x = 4$ km, and only GA–game-theoretic approaches are considered due to the limited scalability of GA-MILP. The results show that GA-Matching-Bidding-LP consistently achieves the lowest latency, highlighting the benefit of exact LP-based routing. GA-Matching-Bidding-FW provides comparable performance, outperforming the heuristic approach. In contrast, the heuristic exhibits the highest latency, with a sharp increase beyond a traffic load of 300, indicating suboptimal routing decisions. These findings emphasize the trade-off between computational efficiency and latency performance, and the critical role of routing optimization in 5G edge networks.

## VII. Conclusion

This paper investigated the joint optimization of UPF placement and traffic routing in 5G edge networks, formulated as a bilevel problem with ILP-based placement and MILP-based routing. To address its NP-hardness, we proposed two hybrid approaches: GA-MILP and GA–game-theoretic methods. Simulation results show that GA-MILP achieves the best cost and energy efficiency, while GA–game-theoretic approaches significantly reduce runtime with limited performance degradation. This highlights a clear trade-off between optimality and scalability, especially at larger network scales. Future work will focus on online adaptive optimization and dynamic traffic handling, including the integration of reinforcement learning techniques with rigorous complexity and convergence analysis.

## References


[1] J. L. Vieira, E. L. Macedo, A. L. Battisti, J. Noce, P. F. Pires, D. C. Muchaluat-Saade, A. C. Oliveira, and F. C. Delicato, "Mobility-aware sfc migration in dynamic 5g-edge networks," *Computer Networks*, vol. 250, p. 110571, 2024.

[2] H. T. Nguyen, T. Van Do, and C. Rotter, "Scaling upf instances in 5g/6g core with deep reinforcement learning," *IEEE Access*, vol. 9, pp. 165 892–165 906, 2021.

[3] I. Leyva-Pupo and C. Cervelló-Pastor, "Efficient solutions to the placement and chaining problem of user plane functions in 5g networks," *Journal of Network and Computer Applications*, vol. 197, p. 103269, 2022.

[4] Y. Li, X. Ma, M. Xu, A. Zhou, Q. Sun, N. Zhang, and S. Wang, "Joint placement of upf and edge server for 6g network," *IEEE Internet of Things Journal*, vol. 8, no. 22, pp. 16 370–16 378, 2021.

[5] X. Zhang, Z. Xu, L. Fan, S. Yu, and Y. Qu, "Near-optimal energy-efficient algorithm for virtual network function placement," *IEEE Transactions on Cloud Computing*, vol. 10, no. 1, pp. 553–567, 2019.

[6] C. Pham, N. H. Tran, S. Ren, W. Saad, and C. S. Hong, "Traffic-aware and energy-efficient vnf placement for service chaining: Joint sampling and matching approach," *IEEE Transactions on Services Computing*, vol. 13, no. 1, pp. 172–185, 2017.

[7] A. Varasteh, B. Madiwalar, A. Van Bemten, W. Kellerer, and C. Mas-Machuca, "Holu: Power-aware and delay-constrained vnf placement and chaining," *IEEE Transactions on Network and Service Management*, vol. 18, no. 2, pp. 1524–1539, 2021.

[8] I. Ljubić, A. Mouaci, N. Perrot, and É. Gourdin, "Benders decomposition for a node-capacitated virtual network function placement and routing problem," *Computers & Operations Research*, vol. 130, p. 105227, 2021.

[9] S. Wang, C. Yuen, W. Ni, Y. L. Guan, and T. Lv, "Multiagent deep reinforcement learning for cost-and delay-sensitive virtual network function placement and routing," *IEEE Transactions on Communications*, vol. 70, no. 8, pp. 5208–5224, 2022.

[10] L. Gu, D. Zeng, W. Li, S. Guo, A. Y. Zomaya, and H. Jin, "Intelligent vnf orchestration and flow scheduling via model-assisted deep reinforcement learning," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 2, pp. 279–291, 2019.

[11] Z. Luo and C. Wu, "An online algorithm for vnf service chain scaling in datacenters," *IEEE/ACM Transactions on Networking*, vol. 28, no. 3, pp. 1061–1073, 2020.

[12] C. Pham, D. T. Nguyen, N. H. Tran, K. K. Nguyen, and M. Cheriet, "Optimized iot service chain implementation in edge cloud platform: a deep learning framework," *IEEE Transactions on Network and Service Management*, vol. 18, no. 1, pp. 538–551, 2021.

[13] C. Möbius, W. Dargie, and A. Schill, "Power consumption estimation models for processors, virtual machines, and servers," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 6, pp. 1600–1614, 2013.

[14] S. Chen, J. Chen, and H. Li, "Joint optimization of upf placement and traffic routing for 5g core network user plane," *Computer Communications*, vol. 216, pp. 86–94, 2024.

[15] M. Dayarathna, Y. Wen, and R. Fan, "Data center energy consumption modeling: A survey," *IEEE Communications surveys & tutorials*, vol. 18, no. 1, pp. 732–794, 2015.

[16] S. W. Kareem, K. W. H. Ali, S. Askar, F. S. Xoshaba, and R. Hawezi, "Metaheuristic algorithms in optimization and its application: A review," *JAREE (Journal on Advanced Research in Electrical Engineering)*, vol. 6, no. 1, 2022.

[17] R. Ramalingam, J. Shobana, K. Arthi, G. Elangovan, S. Radha, and N. Priyanka, "An extensive investigation of meta-heuristics algorithms for optimization problems," in *Metaheuristics Algorithm and Optimization of Engineering and Complex Systems*. IGI Global, 2024, pp. 223–241.

[18] S. M. Almufti, A. A. Shaban, Z. A. Ali, R. I. Ali, and J. D. Fuente, "Overview of metaheuristic algorithms," *Polaris Global Journal of Scholarly Research and Trends*, vol. 2, no. 2, pp. 10–32, 2023.

[19] M. Frank, P. Wolfe *et al.*, "An algorithm for quadratic programming," *Naval research logistics quarterly*, vol. 3, no. 1-2, pp. 95–110, 1956.

[20] M. Jaggi, "Revisiting frank-wolfe: Projection-free sparse convex optimization," in *International conference on machine learning*. PMLR, 2013, pp. 427–435.