Dynamic Flow Rule Placement for Real-Time Energy Optimization in SDN

Sibananda Behera , Namita Panda , Sudhansu Shekhar Patra *

Posted Date: 9 April 2026

doi: 10.20944/preprints202604.0666.v1

Keywords: SDN; dynamic flow-rule placement; energy-aware routing; TCAM; batch scheduling receding horizon optimisation (RHO); convex relaxation algorithm

 doi:10.20944/preprints202604.0666.v1

Article

Dynamic Flow Rule Placement for Real-Time Energy Optimization in SDN

Sibananda Behera, Namita Panda and Sudhansu Shekhar Patra *

School of Computer Engineering, KIIT Deemed to be University, Bhubaneswar, India
* Correspondence: sudhanshupatra@gmail.com

Abstract

Software-Defined Network (SDN) renders flexible traffic engineering, but consumes a lot of energy. There is an overhead on the control-plane because of flow-rule updates are done always and the energy consumption by the forwarding hardware. Current energy-aware SDN methods mostly focus on static or greedy optimisations. This can cause too many Ternary Content-Addressable Memory (TCAM) updates and unstable rule churn when traffic changes over time. This article introduces a Dynamic Flow Rule Placement (DFRP) framework for real-time energy optimisation in SDN. It reduces network energy usage, TCAM update costs, and rule churn all at the same time. The suggested framework uses convex relaxation method to take decisions on binary switches, links, and rule placement. It also uses a minimum-edit round scheme that only allows small rule changes between time slots. To further reduce instability in the control plane, batch scheduling and receding horizon optimisation (RHO) techniques are combinedly used. The system uses predicted traffic for future time slots to make decisions, but only the actions for the current time slot are executed.. The experiments are carried out on two real-world dynamic SNDlib topologies such as Germany50 and Nobel-Germany, using 288 five-minute traffic matrices over a one-day period. Comparative results against static and greedy baselines show that DFRP saves approx 30% energy while cutting down on TCAM update overhead and rule churn by approx 20%, consistently across both networks. Hence DFRP can be used on a dynamic traffic large scale networks for stable and energy-efficient SDN operations.

Keywords: SDN; dynamic flow-rule placement; energy-aware routing; TCAM; batch scheduling receding horizon optimisation (RHO); convex relaxation algorithm

---

1. Introduction

As cloud services, mobile apps, and data-heavy workloads are growing quickly, communication networks are using more and more energy all the time. Modern backbone and metro networks are often set up to handle peak traffic, but they actually work under very different loads, which wastes a lot of energy when traffic is low. SDN in Figure 1 has become a promising way to enable fine-grained traffic engineering and energy-aware network operation [1,2]. This is because it has logically centralised control and global network visibility.

One of the most important things that SDN makes possible is the ability to dynamically change the active network topology by turning off switches, links, and line cards that aren't being used as much and sending traffic over a smaller number of resources. Prior research has demonstrated that traffic consolidation and topology adaptation can result in substantial energy savings, but putting these ideas into action in real-world SDN deployments is still hard because control-plane operations are very expensive, especially when it comes to installing and deleting flow rules in TCAM. TCAM

 doi:10.20944/preprints202604.0666.v1

consumes a lot of energy and can also cause latency, signalling overhead, and temporary inconsistencies in the data plane [3,4].

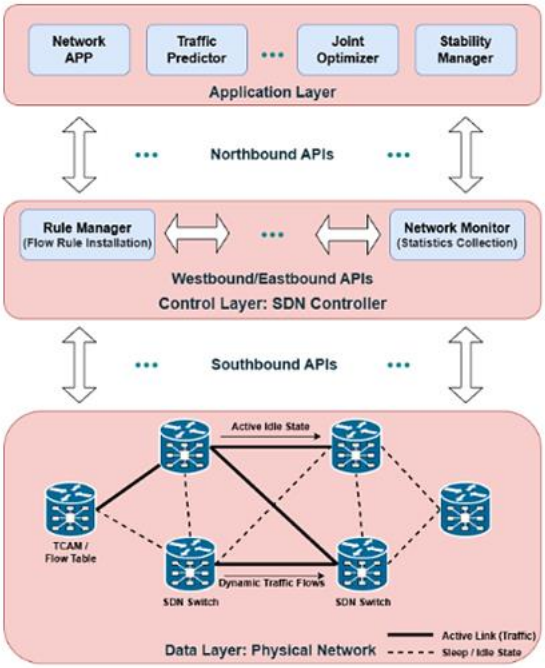


Figure 1. SDN Architecture.

Existing energy-aware SDN solutions largely rely on static routing configurations or greedy heuristics that react to instantaneous traffic conditions. While such approaches can reduce power consumption, they often incur excessive rule churn and control-plane instability when traffic fluctuates over time. Frequent reconfiguration of flow tables may negate energy savings by increasing TCAM update cost and can adversely affect network reliability. Moreover, many prior works overlook the temporal coupling between consecutive optimization decisions, treating each time slot independently and thereby amplifying oscillatory behavior.

To overcome these constraints, this paper examines the issue of Dynamic Flow Rule Placement for Real-Time Energy Optimisation in SDN, aiming to concurrently optimise network energy consumption, TCAM update costs, and rule churn, while ensuring stable network performance amidst fluctuating traffic demands. We contend that efficient energy optimisation in SDN must consider both the current network state and the temporal cost of transitioning between configuration. Inevitably suggested a Dynamic Flow Rule Placement (DFRP) framework that brings together three important parts. First, a convex relaxation-based optimisation model [5,6] is used to quickly get close to the inherently binary choices that come with activating a switch, using a link, and placing a rule. Second, a minimum edit round scheme is used to cut down on control-plane overhead by only making small changes to rules between time slots [30]. Third, batch scheduling and receding horizon optimisation (RHO) are used to make the most of short-term traffic forecasts. At each optimisation step, only the first decision is enforced, which stops oscillations and keeps the time stable [7,8].

The primary objectives of this work are as follows:

1. Create a dynamic, time-coupled problem for putting flow rules in SDN that finds the best balance between energy use, TCAM update cost, and rule churn.
2. Invent a DFRP framework that makes use of receding horizon optimisation, minimum-edit rule updates, and convex relaxation confirm that the network operates smoothly and efficiently.
3. Conduct a comprehensive evaluation on two real SNDlib backbone networks with realistic dynamic traffic, confirming that our method performs better than static and greedy baselines every time.

 doi:10.20944/preprints202604.0666.v1

4. This experiment evaluation systematically reveals the trade-off between energy efficiency and control plane stability in real time SDN optimisation.

The subsequent sections are set up like this. In Section 2, we go over other work that has been done on energy-aware SDN and dynamic rule management. Section 3 shows the model for the set up and how to work out the problem. Section 4, shows the proposed DFRP framework. Section 5 depicts about the experimental arrangement and the data sets. Section 6 looks at the results analysis, and finally Section 7 looks at the conclusion and future work.

2. Related Work

Modern Software-Defined Networks (SDN) must handle highly dynamic traffic while maintaining efficient and stable operation. Existing approaches reduce energy consumption by deactivating unused network elements, but often cause frequent rule updates and instability. Additionally, limited TCAM capacity restricts scalable flow rule placement. Therefore, there is a need for a framework that jointly optimizes energy efficiency, TCAM usage, and rule stability under real-time traffic conditions.

Authors in [9] investigate the performance of routing algorithms in SDN by examining both static and dynamic link cost techniques. This shows that dynamic link cost–based routing better responds to traffic variations than static schemes. Use of these dynamic metrics causes load balancing and less congestion and decreases end-to-end delay. The authors emphasize the value of accurate network status at the moment when SDN routing decisions occur. Authors in [10] proposed an energy-aware routing algorithm for SDN that dynamically selects paths based on the power states of switches and links. Their method reduces network energy consumption by consolidating traffic onto fewer active elements. However, the approach does not consider TCAM usage or rule-churn stability, which limits its suitability for highly dynamic networks. In [11], authors provide a holistic survey of GREEN SDN,covering energy saving in the paradigm of SDN. The paper gives a comprehensive classification of green approaches, such as traffic-aware routing, switch sleep modes and energy-aware resource management. They explain how efficient power optimization without degradation of network performance is possible, with the centralized control SDN. And open challenges and future directions for sustainable/eco-friendly SDN architectures are also presented by the authors. An iterative dynamic optimization model for software-defined networks that also solves load balancing and energy efficiency using the meta-heuristic Krill Herd algorithm is presented in [12]. This strategy is based on the global network perspective of SDN to achieve efficient traffic split together with reduced power levels. By dynamically loading flows, the approach alleviates link congestion as well as minimizes over-activation of network devices. Simulation results validate the enhancement of throughput, decrease in delay and substantial energy saving compared to traditional SDN optimization methods.

Article [13] introduced a dynamic routing optimization method for SDN that adjusts routing decisions in response to network changes. The purpose of the method is to employ the centralized SDN controller to improve load balancing and reduce congestion. Experimental results demonstrate superior network performance relative to static routing algorithms, especially in dynamic traffic conditions. A meta-heuristic driven dynamic routing optimisation model is presented by authors in [14] for SDN. The SDN architecture enables the global network view and centralized control that SDN has to make routing decisions on the fly. Through the process of dynamic path selection based on traffic conditions, network congestion can be reduced and it is possible to optimise traffic distribution. The proposed algorithm was compared with a conventional SDN routing process. It was found through evaluation results that the proposed scheme had less end-to-end delay, higher throughput and better resource utilisation than the typical SDN process. The article in [15] introduced FlowStat, an adaptive flow-rule placement algorithm. It was created to obtain flow statistics in software networks that have been defined by software. The system dynamically selects the position of a flow-control rule within the router in order to reduce processing load of the router and to alter how router

 doi:10.20944/preprints202604.0666.v1

resources are used. FlowStat optimizes the balance between monitoring's scalability and precision ensuring that network monitoring is accurate despite minimal network control-plane communication. Results obtained in software defined networks show that the use of this new architecture has a more efficient flow rule installation and maintenance in comparison to traditional architectures. Researchers [16] proposed a modified heuristic technique for the reduction of power consumption in Software Defined Networking. This algorithm minimizes the power consumption of the network by switching non-essential elements into a low power mode, thus preventing any reduction in network functionality. MHAES responds in real time to changes in network traffic through the use of a central SDN controller. Simulation outcomes indicate considerable energy saving while keeping delay and throughput at a satisfactory level. Article [17] introduced OFFICER, an effective framework for maximizing OpenFlow rules and enforcing endpoint policies in software-defined networks. This is set up as a linearly constrained convex optimization problem with the goal of maximizing the number of rules that can be loaded into the switch memory that is available, while still meeting policy constraints. OFFICER's distribution of rules across the network makes it much less likely that T CAMs will be used, which makes the network easier to scale. Tests have demonstrated that this method requires less overhead for enforcing policies than traditional methods.

Researchers in [18] developed an energy aware routing strategy for software defined networks based upon traffic compression methods. The approach reduces the number of active links by aggregating and compressing the data from multiple network flows, maintaining the routing constraints. Using a global view, which is available in the SDN controller, network paths are dynamically adjusted to cut back power usage. The results show considerable reductions in power consumption with network performance and quality of service affected only to a limited degree. The researchers recently proposed a SDN framework Energy and Blocking Aware Routing and Device Assignment for developing software defined networks using a mixed integer linear programming model. The scalability is addressed through the use of a genetic algorithm. The goal is jointly to minimize blocking calls and reduce network power usage. The results show that the system operates more efficiently and there is less likelihood of blocking with this system compared to other systems. The authors presented EDITORS [20], an energy-efficient dynamic task offloading system for SDN-enabled edge networks via deep reinforcement transfer learning. The model learns the best offloading strategies by taking into account how much energy is used, how long it takes, and how the workload changes. The method speeds up convergence and makes it easier to adjust by moving learned rules between edge nodes. Compared to traditional offloading methods, experimental results demonstrate big drops in both energy use and response time [18].

Table 1. Comparative Analysis of Existing Research Approaches.

Reference	Feature/Method	Energy Saving	TCAM Aware	Rule-Churn Control	RHO	Baseline Method
[9]	Static vs. dynamic link-cost routing	✗	✗	✗	✓	Static link-cost routing
[10]	Energy-aware routing using link/node power models	✓	✗	✗	✓	Shortest-path / static routing

 doi:10.20944/preprints202604.0666.v1

[11]	GREEN SDN survey & taxonomy	✓	✗	✗	✗	Traditional SDN routing & management
[12]	Dynamic routing optimization algorithm using SDN controller with adaptive path selection	✗	✗	✗	✗	Static shortest- path routing
[13]	Krill Herd metaheuristic optimization	✓	✗	✗	✓	Conventional SDN load balancing schemes
[14]	Metaheuristic-based dynamic routing	✗	✗	✗	✓	Shortest-path / traditional SDN routing
[15]	FlowStat adaptive flow rule placement	✗	✓	✓	✓	Static flow rule placement
[16]	MHAES energy-saving heuristic	✓	✗	✗	✓	Always on SDN network operation

3. Problem Formulations and System Model

Modern SDN must handle highly dynamic traffic while maintaining efficient and stable operation. Existing approaches reduce energy consumption by deactivating unused network elements, but often cause frequent rule updates and instability. Additionally, limited TCAM capacity restricts scalable flow rule placement. Therefore, there is a need for a framework that jointly optimizes energy efficiency, TCAM usage, and rule stability under real-time traffic conditions.

3.1. Baseline Routing Models

To ensure fair and reproducible evaluation, two reference routing strategies are implemented: a static routing baseline [9] and a greedy dynamic routing baseline [10]. Both models use the same traffic demands and network constraints as the proposed DFRP.

 doi:10.20944/preprints202604.0666.v1

3.1.1. Static Baseline Routing

Objective Function: $\min \sum_{(u,v)\in E} C_{uv}\ f_{uv}(t)$ (1)

Subject to Constraints

Flow Conservation:

$$\sum_{v:(u,v)\in E} f_{uv}^{sd}(t) - \sum_{v:(v,u)\in E} f_{vu}^{sd}(t) = \begin{cases} D_{sd}(t), & u = s \\ -D_{sd}(t), & u = d \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

Link Capacity Constraint:

$$\sum_{(s,d)} f_{uv}^{sd}(t) \le U_{uv}, \forall (u,v) \in E \quad (3)$$

Static Configuration Constraint:

$$\boldsymbol{x_{uv}(t) = x_{uv}(0), \forall t} \quad (4)$$

where

$\mathbf{G(V,E)}$: Network graph with nodes $\mathbf{V}$and links $\mathbf{E}$

$\mathbf{D_{sd}(t)}$: Traffic demand from source $\mathbf{s}$to destination $\mathbf{d}$at time $\mathbf{t}$

$\mathbf{f_{uv}^{sd}(t)}$: Flow on link $(\mathbf{u,v})$for demand $(\mathbf{s,d})$at time $\mathbf{t}$

$\mathbf{x_{uv}(t)} \in \{\mathbf{0,1}\}$: Link activation variable (1 if used, 0 otherwise)

$\mathbf{C_{uv}}$: Fixed link cost

$\mathbf{c_{uv}(t)}$: Time-varying link cost

$\mathbf{U_{uv}}$: Capacity of link $(\mathbf{u,v})$

Algorithm 1: Static Routing Baseline

Input: G(V, E), fixed link costs C_{uv}, traffic D(0)

1. Compute shortest paths using C_{uv}
2. Install flow rules at $t = 0$
3. For each time slot t do
4. Route traffic using shortest paths derived from C_{uv}.
5. Use the same link cost values C_{uv} for all t.
6. Keep all flow rules unchanged
7. End for
8. End Algo

3.1.2. Greedy Dynamic Routing Baseline

Objective Function: $\min \sum_{(u,v)\in E} c_{uv}(t)\ f_{uv}^{sd}(t)$ (5)

subject to the same constraints as the static model. Dynamic Update (Rule Change) Constraint:

$$\mathbf{x_{uv}(t) \neq x_{uv}(t-1)} \quad (6)$$

Algorithm 2: Greedy Dynamic Routing Baseline

Input: G(V, E), traffic D(t)

1. For each time slot t do
2. Measure link state and traffic
3. Compute link cost $C_{uv}(t)$
4. Find shortest paths using $C_{uv}(t)$
5. Replace all flow rules
6. End for
7. End Algo

 doi:10.20944/preprints202604.0666.v1

3.1.3. Proposed Dynamic Flow Rule Placement Model (DFRP)

The Dynamic Flow Rule Placement (DFRP) framework is designed Figure 2 to jointly optimize network energy consumption, TCAM usage, and rule churn under time-varying traffic demands, while explicitly enforcing temporal stability. Unlike the static and greedy baselines, DFRP introduces time coupling, minimum-edit reconfiguration, and receding horizon control into the optimization process [21,22]. This figure illustrates the end-to-end architecture of the proposed DFRP framework. It is shown how traffic inputs and network state are processed by the control plane to generate energy-aware, stability-optimized link and flow rule decisions. Network topology, traffic matrices, and benchmark dataset inputs are used by the system to track and forecast the present and future status of the network. In order to calculate the best routing and link activation choices under dynamic traffic circumstances, a convex optimisation engine with receding horizon control processes these inputs. A rule placer then uses batch scheduling and minimum-edit to enhance the calculated solution, lowering rule churn and increasing stability. The network's control activities, such as link ON/OFF states and flow rule changes in SDN switches, which establish traffic routing while guaranteeing energy efficiency and effective TCAM utilisation, make up the final output. Furthermore, the system generates and feeds back performance measures including energy consumption, latency, and rule churn, allowing for ongoing monitoring and adaptive optimisation over time [23–25].

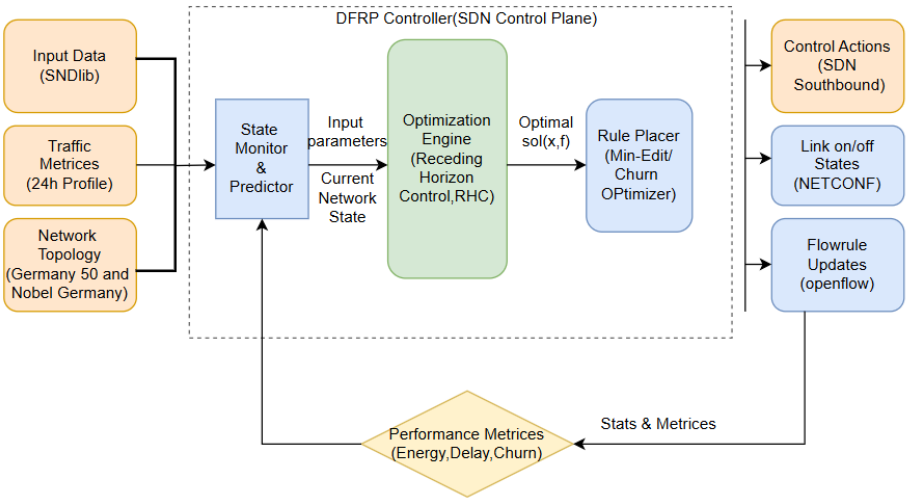


Figure 2. System Architecture of proposed DFRP Model.

3.1.3.1. Dynamic Flow Rule Placement (DFRP): Mathematical Model

i) Network and Time Model

We model the SDN network as a directed graph.

$$G = (V, E),$$

where V is the set of OpenFlow switches and E is the set of physical links. Time is discretized into slots

$$t \in \{1,2, \dots, T\},$$

corresponding to 5-minute intervals. Traffic demand at each slot is given by a time-varying demand matrix

$$D(t) = \{ D_{sd}(t) \mid s, d \in V \}. (7)$$

ii) Decision Variables

At each time slot t, DFRP determines:

- **Switch activation**

$$z_v(t) \in [0,1], \forall v \in V \ (8)$$

 doi:10.20944/preprints202604.0666.v1

- **Link activation**

$$x_{uv}(t) \in [0,1], \forall (u,v) \in E \ (9)$$

- **Flow allocation**

$$f_{uv}^{sd}(t) \geq 0, \forall (u,v) \in E, \forall (s,d) \ (10)$$

Binary activation variables are relaxed to continuous values in $[0,1]$ to enable convex optimization. Binary decisions are recovered via thresholding after optimization.

iii) Objectve Functions

Objective 1: Network Energy Minimization

The total energy consumption at time t is defined as:

$$J_{\text{energy}}(t) = \sum_{v \in V} P_{\text{switch}} \cdot z_v(t) + \sum_{(u,v) \in E} P_{\text{link}} \cdot x_{uv}(t) \ (11)$$

This objective encourages traffic consolidation and allows unused switches and links to enter deep sleep mode.

Objective 2: TCAM Usage Minimization

Let $R_v(t)$ denote the number of active flow rules installed at switch v at time t. TCAM usage is normalized by hardware capacity $TC_{\max}$:

$$J_{\text{tcam}}(t) = \sum_{v \in V} \frac{R_v(t)}{TC_{\max}} \ (12)$$

By penalizing TCAM occupancy, DFRP promotes path aggregation, reducing the number of distinct forwarding rules required in the data plane.

Objective 3: Rule Churn / Update Cost Minimization

To ensure stability, DFRP penalizes changes in network configuration between consecutive time slots:

$$J_{\text{churn}}(t) = \sum_{(u,v) \in E} | x_{uv}(t) - x_{uv}(t-1) | \ (13)$$

This L1-norm penalty acts as a damping term, discouraging frequent ON/OFF transitions and preventing route flapping.

Global Objective Function

The three objectives are combined into a weighted convex cost function:

$$\text{Min} \ \ J_{\text{global}}(t) = \alpha J_{\text{energy}}(t) + \beta J_{\text{tcam}}(t) + \gamma J_{\text{churn}}(t) \ (14)$$

where:

α controls energy savings,

β regulates TCAM pressure,

γ enforces temporal stability.

iv) Network Constraints

1) Flow Conservation

$$\sum_{u:(u,v) \in E} f_{uv}^{sd}(t) - \sum_{u:(v,u) \in E} f_{vu}^{sd}(t) = \begin{cases} D_{sd}(t), & v = s \\ -D_{sd}(t), & v = d \\ 0, & \text{otherwise} \end{cases} \ (15)$$

2) Capacity Constraints

$$\sum_{s,d} f_{uv}^{sd}(t) \leq C_{uv} \cdot x_{uv}(t), \forall (u,v) \ (16)$$

3) Activation Coupling

$$x_{uv}(t) \leq z_u(t), x_{uv}(t) \leq z_v(t) \ (17)$$

v) Receding Horizon Control (RHC)

At each time slot t, DFRP optimizes over a prediction horizon

$$[t, t+H],$$

Solving a multi-slot convex program:

 doi:10.20944/preprints202604.0666.v1

$$\underset{\{x(k),z(k),f(k)\}_{k=t}^{t+H}}{\text{Min}} \sum_{k=t}^{t+H} \left(\alpha J_{\text{energy}}(k) + \beta J_{\text{tcam}}(k) + \gamma J_{\text{churn}}(k)\right) \quad (18)$$

Only the first-slot decision $(x(t), z(t), f(t))$ is applied, and the optimization is repeated at $t+1$.

vi) Minimum-Edit Rule Placement

After solving the relaxed problem, DFRP applies only delta rule changes:

$$\Delta R(t) = R(t) \setminus R(t-1) \quad (19)$$

This ensures:

- Minimal TCAM updates
- Reduced control-plane signalling
- Stable data-plane behaviour

The proposed framework works as a time-driven closed-loop control system as shown in Figure 3. First, the network topology G (V, E) and link capacities are set up. The current traffic demand matrix D(t) is read and sent to a receding horizon optimization module at each discrete time slot. This module predicts changes in traffic that will happen soon and finds the best configuration using either a greedy energy-minimization strategy or the proposed DFRP strategy, which looks at both energy efficiency and stability at the same time. The controller figures out the set of active links and switches and the flow rules that need to be followed based on the chosen strategy. Then, only the changes to the state that are needed are made to the network to keep the control-plane overhead as low as possible. This process is repeated for each time slot until the end of the simulation horizon. Finally, the performance metrics like energy use, TCAM update cost and rule churn are collected and analysed.

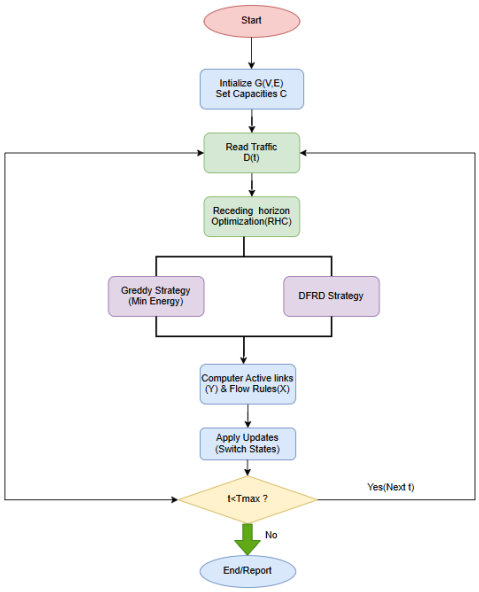


Figure 3. Flowchart of DFRP Model.

Algorithm 3: Dynamic Flow Rule Placement (DFRP) Using Receding Horizon Optimization

Input: Network topology $G(V, E)$, Link capacities C_{uv} for all $(u, v) \in E$,

Time varying traffic demands $D(t), t = 1, \ldots, T$

Prediction horizon H, Weight parameters α, β, γ

Output: Active switch states $z(t)$, Active link states $x(t)$, Flow routing decisions $f(t)$

Initialization

1. Set initial network state:

 doi:10.20944/preprints202604.0666.v1

2. $z(0) \leftarrow 1$ for all $v \in V$ # All switches active
3. $x(0) \leftarrow 1$ for all $(u, v) \in E$ # All links active
4. Initialize rule set $R(0)$
5. for each time slot $t = 1$ to T do

▷ **Step 1**: **Traffic Prediction**

6. Obtain traffic demand forecasts:
7. $D(k) = D(t + k), \quad \text{for } k = 0, \ldots, H$

▷ **Step 2**: **Receding Horizon Optimization**

8. Formulate a convex optimization problem over horizon $[t, t + H]$:
9. Decision variables:
10. $z(k), x(k), f(k) \quad \text{for } k = t, \ldots, t + H$

Objective:

11. $$\min_{\{x(k),z(k),f(k)\}_{k=t}^{t+H}} \sum_{k=t}^{t+H} \left(\alpha J_{energy}(k) + \beta J_{tcam}(k) + \gamma J_{churn}(k)\right)$$
12. Subject to (for all $k = t, \ldots, t + H$):
13. Flow conservation constraints
14. Link capacity constraints
15. Activation coupling constraints
16. $0 \leq z(k), x(k) \leq 1$

▷ **Step 3**: **Solve Optimization**

17. Solve the convex program using a suitable solver (e. g. , OSQP)

▷ **Step 4**: **Post Processing**

18. Extract first slot solution:
19. $z(t), x(t), f(t)$
20. Apply thresholding to obtain binary states:
21. $\bar{z}(t) = \text{round}(z(t))$
22. $\bar{x}(t) = \text{round}(x(t))$

▷ **Step 5**: **Minimum Edit Rule Placement**

23. Compute rule updates:
24. $\Delta R(t) = R(t) \setminus R(t - 1)$
25. Install/remove only $\Delta R(t)$ rules in TCAM

▷ **Step 6**: **Network Execution**

26. Apply $\bar{z}(t)$ and $\bar{x}(t)$ to the data plane
27. Install routing rules corresponding to $f(t)$
28. Update system state:
29. $R(t) \leftarrow R(t - 1) \cup \Delta R(t)$
30. end for

End Algo

The Algorithm-3 shows how to use an online, receding-horizon optimization [29] framework for DFRP model in SDN. The controller makes a small prediction of traffic demands for each time slot and then produces a convex optimization problem that minimizes network energy, TCAM use, and rule churn simultaneously. To improve the efficiency of convex optimization, binary switch and link

 doi:10.20944/preprints202604.0666.v1

activation decisions are switched to continuous variables. For stability reasons, we therefore add an L1-norm penalty which is put on changes to the configuration. Although the problem is solved over multiple slots, only the first-slot decision is taken, consistent with the receding horizon control paradigm. The control plane's overhead is further reduced through the use of a minimum edit strategy. It signifies that only little modifications to the rules are made in the TCAM. This design allows DFRP to achieve substantial energy saving while maintaining stable network operation, striking a balance between static configurations and aggressive greedy heuristics [26–28].

4. Result Analysis

This section outlines the experimental environment, datasets, baseline methods, evaluation metrics and methodology for validating the proposed Dynamic Flow Rule Placement (DFRP) framework.

4.1. Experimental Environment Settings

The proposed DFRP model was validated through a two-stage experimental implementation, consisting of an analytical simulation layer and a real-time SDN emulation layer. All experiments were organised on a computer with an Intel Core i7-10750H processor that runs at 2.60 GHz and 16 GB of RAM. The convex optimization problems in the receding-horizon control loop are solved using the OSQP solver via the CVXPY modeling framework [32,33].

To illustrate the practicality the DFRP model in a real SDN environment, the same DFRP logic is deployed in the Mininet network emulator controlled by the Ryu SDN controller. Communication between the controller and switches is performed using the OpenFlow 1.3 protocol. Two real-world ISP-scale backbone topologies, namely Germany50 and Nobel-Germany, are emulated. Network traffic is generated using the proposed diurnal stochastic Traffic Generation Model, which holds both long-term day–night run and short-term oscillations observed in operational carrier networks. The model simulates time-variant traffic over a continuous 24-hour cycle, which is further divided into 288 time slots of 5 minutes each. The analytically calculated demands are then translated into host-to-host traffic in Mininet. The Iperf2 tool is used to inject traffic at rates specified by the mathematical model. This comprehensive method ensures that the simulated network is exposed to the same traffic as the analytical model, thus enabling a realistic comparison between the Static, Greedy, and DFRP algorithms.

4.2. Dataset Description

Two realistic backbone network topologies, Germany50 and Nobel-Germany, are used from the Survivable Network Design Library (SNDlib) [29]. These ISP-scale datasets are widely used as benchmarks for research on telecommunication networks. They include networks of different sizes, link densities, and structural features, which makes them useful for realistic and varied evaluation scenarios.

4.2.1. Germany50 (Main Benchmark) and Nobel-Germany Topologies

The usual benchmark for evaluating performance is the Germany50 topology. It is made up of 50 nodes and 88 bi-directional physical connections that connect to 176 directed edges. A countrywide German research backbone network with several backup and alternate routes is shown in this design. It is ideal for assessing routing stability, rule churn, and energy-aware optimisation under complicated traffic situations because of its extensive connection and many alternative pathways. All connections are set up with an identical 40 Gbps capacity for simplicity and equity, representing a consistent optical core network.

Figure 4 depicts the physical configuration of the Germany50 and Nobel Germany topologies included into the simulations.

 doi:10.20944/preprints202604.0666.v1

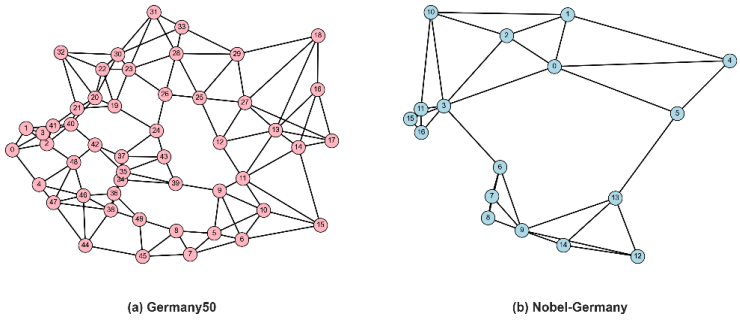


Figure 4. Germany50 and Nobel-Germany Topology (Testing for Scalability).

Another topology the Nobel-Germany is used to see the efficiency and performance of our proposed model on a smaller network and its ability to handle different traffic loads. This topology has 17 nodes and 26 bi-directional links, 52 directed edges, and represents a regional core network with lower redundancy than Germany50. Because of the limited alternative paths, Nobel-Germany is essential for analysing algorithm behaviour under traffic congestion. In Table 2 the detailed dataset specifications and sample entries of Germany50 topology are given, expressing the network structure, link characteristics, and dynamic traffic demands working in the experiments.

Table 2. Dataset Specifications and Sample Entries (Germany50).

a) Node Dataset of Germany50 Topology Showing Geographic Coordinates (Longitude and Latitude) of Network Switches

Nodes Dataset Germany50

id	x (longitude)	y (latitude)
Aachen	6.04	50.76
Augsburg	10.9	48.33
Bayreuth	11.59	49.93
Berlin	13.39	52.52

b) Link Dataset of Germany50 Topology Representing Directed Connections with Capacity and Power Attributes

Links Dataset

id	source	target	capacity	power active	power idle
Aachen_Wesel	Aachen	Wesel	1000.0	10.0	0.0
Wesel Aachen	Wesel	Aachen	1000.0	10.0	0.0
Aachen_Koeln	Aachen	Koeln	1000.0	10.0	0.0
Koeln	Koeln	Aachen	1000.0	10.0	0.0

 doi:10.20944/preprints202604.0666.v1

Aachen					
Aachen Duesseldorf	Aachen	Duesseldorf	1000.0	10.0	0.0

c) Dynamic Traffic Demand Dataset of Germany50 Topology with Time-Stamped Source–Destination Bandwidth Matrices (5-Minute Granularity)

Traffic Matrix Sample

Timestamp	Source	Destination	BW (mbps)
00:00	Aachen	Berlin	45.2
00:00	Augsburg	Bremen	12.1
00:00	Berlin	Munich	150.5
00:05	Aachen	Berlin	42.8
00:05	Berlin	Munich	148.2

The organized datasets are obtained from the SNDlib Germany50 topology [29]. The exact geographical location in terms of latitude and longitude is provided for each switch, which helps in precise topological representation. The link data set represents the directed links and their capacities and power status, which help in determining the usage of energy. The dynamic traffic demands are represented through time-stamped source-destination bandwidth matrices with a granularity of 5 minutes. This reflects the dynamic nature of network load.

4.3. Taffic Generaion Model

Due to the unavailability of real-world, fine-grained backbone traffic matrices, which are typically proprietary, we employ a synthetic yet realistic traffic generation model that has been widely adopted in SDN and traffic engineering research. This model is capable of simulating predictable traffic patterns during the day and random variations in traffic over shorter time periods.

Mathematical Traffic Model

Traffic demand at time slot t is defined as:

$$D(t) = D_{\text{base}} + A \cdot \sin\left(\frac{2\pi(t - t_{\text{shift}})}{T_{\text{period}}}\right) + \mathcal{N}(0, \sigma) \qquad (20)$$

Where:

D_{base} represents the baseline traffic load, ensuring that the network is never completely idle.

A denotes the amplitude of traffic variation, controlling the intensity of peak-hour demand.

The sinusoidal term models the diurnal day–night traffic cycle, with:

$T_{period} = 24$hours, corresponding to a full daily cycle,

t_{shift} aligning the peak traffic period with evening hours (approximately 20:00), and the minimum load with early morning hours (approximately 04:00).

$\mathcal{N}(0, \sigma)$ is a zero-mean Gaussian noise component with standard deviation σ, introduced to emulate realistic short-term traffic fluctuations and measurement uncertainty.

The traffic demand model reflects the most important time-dependent features of backbone networks, which have a baseline load, a daily cycle, and random changes [30,31]. The constant and sinusoidal terms represent the background traffic and the typical day-night cycle, respectively. A 24-

 doi:10.20944/preprints202604.0666.v1

hour cycle and phase shift are used to align the peak and off-peak demands with reality. To model the variability of demand and uncertainty over time, a zero-mean Gaussian noise process is added. This is a simple, realistic, and tractable traffic process that can be employed to evaluate SDN control algorithms that consider energy consumption and stability.

The simulation is a continuous 24-hour process, broken down into 288-time intervals, each lasting 5 minutes. This granularity of data set enables the simulation of topology changes, routing, and control plane dynamics in great detail. Traffic demands are allocated among source-destination pairs according to a gravity-based traffic model based on node centrality values. Nodes of greater topological centrality originate and receive more traffic, ensuring realistic aggregation patterns at core and aggregation routers.

Figure 5 shows the red dashed line as the predictable sinusoidal pattern of traffic during the day and night, with the most traffic at 20:00 and the least traffic at 08:00. The blue curve adds Gaussian noise to model short-term fluctuations, creating a realistic, highly dynamic ISP traffic pattern for evaluating DFRP under sudden load changes.

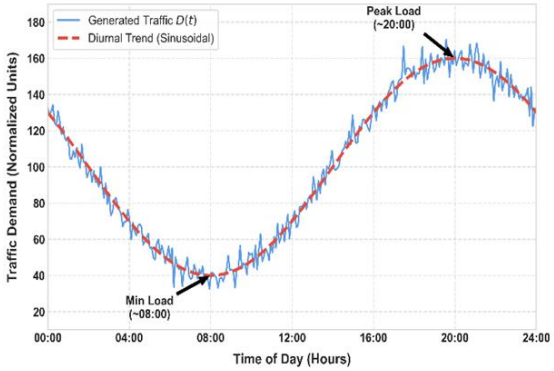


Figure 5. Diurnal Traffic Generation Model.

The Table 3 shows the key network hotspots indicating that Core and Aggregation switches (like S8, S23, and S29) are under the highest load, with peak loads of 160 Gbps. This uneven distribution reflects a hierarchical topology, where a small group of central nodes serving as huge transit hubs for the entire network. As a result, these specific switches are the main points of congestion that the DFRP algorithm has to actively manage to keep traffic flowing smoothly.

Table 3. Top 10 Switches by Traffic Demand (Peak Hour).

Switch ID	Role	Ingress Traffic (Gbps)	Egress Traffic (Gbps)	Total Load (Gbps)
S8	Core/Gateway	80.0	80.0	160.0
S23	Core	80.0	80.0	160.0
S44	Core	80.0	80.0	160.0
S29	Aggr	80.0	80.0	160.0
S30	Aggr	80.0	80.0	160.0
S19	Aggr	70.0	80.0	150.0
S35	Edge	80.0	60.0	140.0

 doi:10.20944/preprints202604.0666.v1

S33	Edge	40.0	80.0	120.0
S46	Edge	40.0	80.0	120.0
S9	Edge	20.0	80.0	100.0

4.4. Simulation Parameters

We chose the main configuration settings for all of the trials to be quite similar to those used in previous SDN energy-efficiency research. This made it easier to compare and reproduce the results as mentioned in Table 4. These parameters tell the RHC framework what it thinks about switch and link power use, TCAM capacity limits, objective weighting coefficients, and the prediction horizon. It talks about the assumptions about how much power switches and links use, the constraints on TCAM capacity, the objective weighting coefficients, and the RHC prediction horizon.

- P_{switch} and P_{link}define the static power draw of active hardware components.
- α, β, and γ control the relative importance of energy minimization, TCAM usage, and stability, respectively.
- The lookahead window $W = 3$ slots correspond to a 15-minute prediction horizon, balancing optimization accuracy and computational overhead.

Table 4. Simulation Parameters.

Parameter	**Value**
Network Topology	Germany50 (SNDlib)
Number of Nodes / Links	50 / 88
Link Capacity	1 Gbps – 10 Gbps
Switch Power (Active/Sleep)	300W / 20W
Link Power (Active/Sleep)	50W / 1W
Traffic Granularity	5 Minutes
Optimization Horizon	3 Slots (15 min)
Weights (Alpha/Beta)	1.0 / 0.1

Table 4 shows the simulation environment, which models the Germany50 topology with different link capacities (1–10 Gbps) and high-resolution traffic updates every five minutes. It sets up the physical reasons for optimization. A Receding Horizon of 15 minutes (3 slots) be applied on the control logic to reduce instability. Here the weight of α=1.0 and β=0.1 clarify that reducing energy consumption is more important than changing rules, hence the algorithm unifies so aggressively. A Receding Horizon of 15 minutes (3 slots) be applied on the control logic to reduce instability. Here the weight of alpha=1.0 and beta=0.1 clarify that reducing energy consumption is more important than changing rules, that is why the algorithm unifies so aggressively.

 doi:10.20944/preprints202604.0666.v1

4.5. Energy Consumption Analysis

The DFRP framework changes in real time to changes in traffic by selectively turning off links that aren't being used enough, especially at night when traffic is low. The framework turns on the network resources that are needed to keep performance up as traffic demand rises. Figure 6 shows the energy use profile over a 24-hour period. It clearly conveys that DFRP closely tracking the traffic curve while using less power than the Static routing method. This three-panel time-series plot illustrates the original findings of our discrete-time simulations which shows how much energy the network consumes at any given time frame of 24-hour cycle using the traffic model from Eq. (20). It shows the preliminary experimental data in a way that it makes easy to determine how our suggested DFRP algorithm saves energy over time in comparison with Static and Greedy baselines. This figure shows the actual performance of our new framework and was made straight from our experimental logs. It is one of the main contributions of this study.

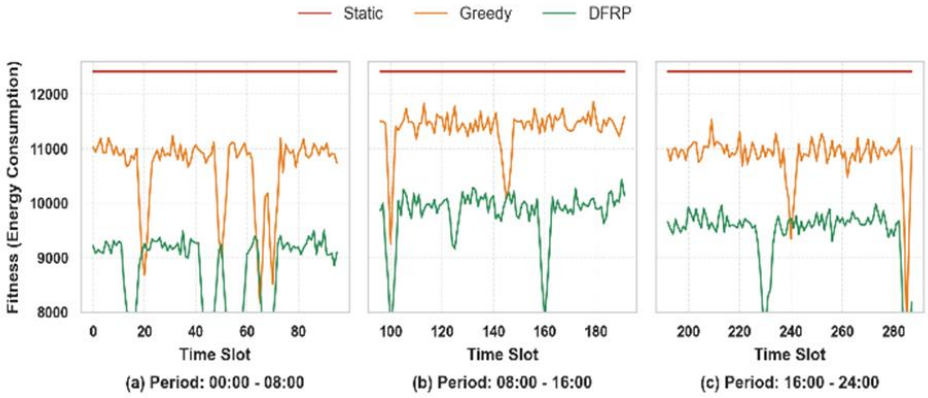
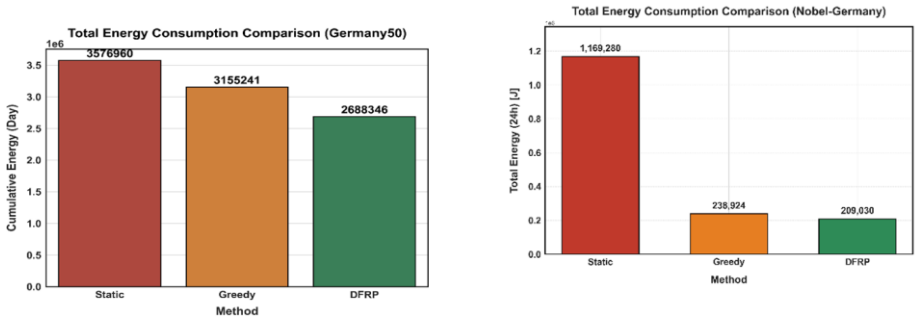


Figure 6. Energy Consumption over 24 Hours (Germany50).

The Figure 7 (a) and (b) illustrates how much energy the Germany50 and nobel-germany topologies uses in 24 hours respectively. In germay50, the Static scheme uses the most energy overall (3.57×10^6 J), whereas the Greedy heuristic cuts that down to 3.15×10^6 J. The proposed DFRP, on the other hand, uses the least amount of energy, 2.68×10^6 J, which is a 24.8% and 14.9% decrease in energy use in comparison to Static along with Greedy, respectively. Also there is a similar beneit in nobel-germany topology. This shows that the proposed method is energy efficient. The numerical comparison of Germany50 topology for the total energy consumption and corresponding energy savings achieved by each routing strategy is as follows.



a) Germany50 b) Nobel-Germany

Figure 7. Total Energy Consumption Comparison.

 doi:10.20944/preprints202604.0666.v1

Table 5. Energy Consumption Comparison (Germany-50).

Method	Total Energy	Energy Saving (%)
DFRP	2,688,346	24.84% vs Greedy
Greedy	3,155,241	11.78% vs Static
Static	3,576,960	0 %

Energy Saving Formula used

$$\text{Energy Saving (\%)} = \frac{\text{Static Energy} - \text{Method Energy}}{\text{Static Energy}} \times \mathbf{100} \quad (21)$$

The experimental results for the Germany 50 topology prove that the proposed DFRP method maintains the best energy saving of all the methods that were tested. The Static method utilizes about 28.5% more energy than DFRP, which implies that DFRP is better at reduce energy use overall. In between two baseline method, the Greedy method is better than the Static method, but it is less effective than DFRP by 18.4%. These results show undoubtedly that the DFRP algorithm can be operated on broad networks and is strong and energy-efficient, especially in Germany 50 topology which is medium in sized.

4.6. Active Infrastructure Analysis

Figure 8 shows the total amount of active switches fluctuates over a 24-hour period in Germany50 topology. The static technique keeps every single switch on all over the day, with an average of 50.00. The Greedy approach, on the other hand, decreases the typical number of switches to 45.63 with few adjustments. The proposed DFRP, on the other hand, lowers the average number of active switches to 39.05. This means that it can quickly rearrange switches based on traffic and uses less energy whenever traffic is changing.

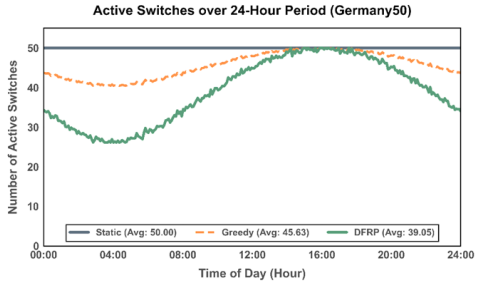
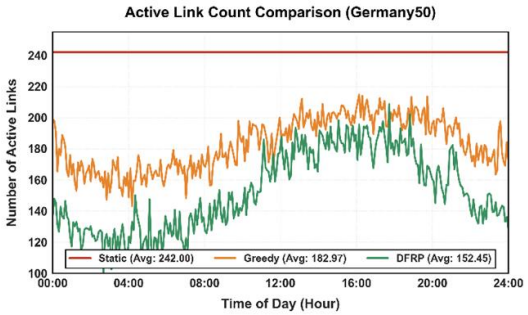


Figure 8. Active Switches over 24-Hour Period.

Figure 9 compares the number of active links over a 24-hour period for the Germany50 topology. The Static scheme maintains the entire links active all day and time (an average of 242 links), whereas the Greedy scheme minimizes the average to 182.97 links but shows noticeable changes over time. In contrast, DFRP achieves the lowest average of 152.45 active links, demonstrating effective traffic consolidation and superior energy-aware link management under dynamic load conditions.

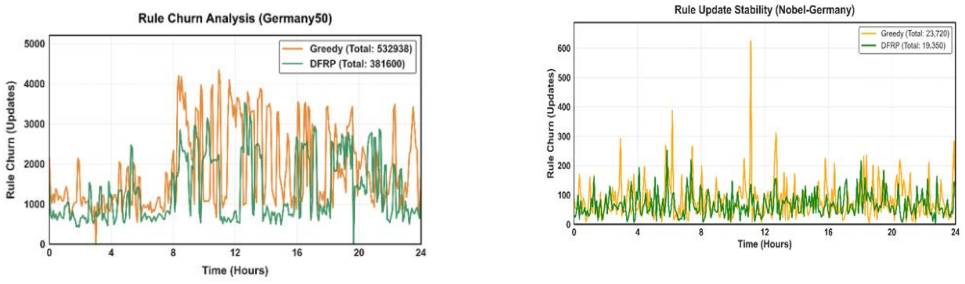


 doi:10.20944/preprints202604.0666.v1

Figure 9. Active Link Count Comparison (Germany50).

4.7. Control-Plane Overhead Analysis

Figure 10 (a) and (b) shows how the Greedy and proposed DFRP solutions vary rules over the course of a 24-hour period according to the Germany50 and Nobel-Germany topologies. For Germany50 topology, the Greedy solution requires a large number of rule updates (532,938 in total) that happen very frequently and with large variations, and this clearly shows that the control plane is not stable. On the other hand, DFRP reduces the total number of rule churns to 381,600, which is a 28.4% reduction, and this clearly shows that it is more stable and enables more accurate flow rule reconfigurations during traffic changes. The For Nobel- Germany topology, Greedy technique yields 23,720 rule modifications, indicating recurrent spikes that denote control-plane instability. The proposed DFRP reduces overall rule churn to 19,350 updates, yielding an estimated 18.4% drop and demonstrating improved stability in dynamic traffic conditions.



(a) Germany50 (b) Nobel-Germany

Figure 10. Rule Churn Analysis.

Figure 11 illustrates the control-plane overhead in terms of rule churn on the Germany50 topology in the form of a box plot. Greedy has a large variance and a larger spread of outliers, where rule updates sometimes exceed the 4,500-6,800-update range, signifying the instability of the control plane, while for Static, the setup-churn is zero. Compared with Greedy, DFRP reduces the median churn value (~1,100 updates) and variance significantly, which confirms the improvement of the control-plane stability of SDN in a dynamic traffic environment.

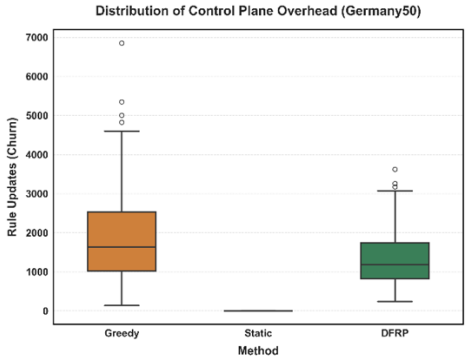


Figure 11. Distribution of Control Plane Overhead.

Figure 12 shows the periodic behaviour of the TCAM update costs based on the Germany50 topology. The Static approach does not have any TCAM updates throughout the day, while the Greedy method has frequent and high update costs, reaching as high as 50×10^2 updates per slot, signifying high rule churn. However, the DFRP approach significantly reduces the TCAM update

 doi:10.20944/preprints202604.0666.v1

costs while offering more economical and trustworthy update costs, thereby ensuring better control plane stability and the feasibility of the proposed method.

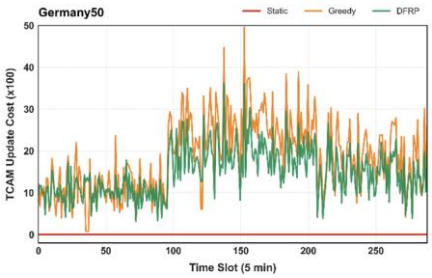


Figure 12. TCAM Update Cost.

4.8. Load and Reliability Trade-offs

Figure 13 illustrates the highest link consumption in a 24-hour period using the Germany50 topology. The Static approach retains the usage level at 50%, whereas the Greedy scheme frequently increases links to full capacity (peak usage of 100%), which demonstrates there's a lot of traffic. The suggested DFRP, on the other hand, limits peak utilization to about 58.1%, which allows the network functioning comfortably and securely even while traffic is changing.

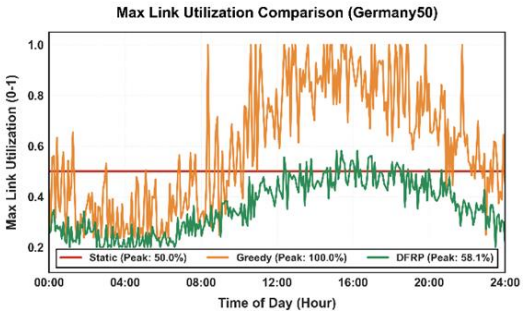


Figure 13. Max Link Utilization Comparison (Germany50).

Figure 14 indicates that shows the CDF of link utilization for the Germany50 topology comparing the level of traffic load aggregation. Static leaves a large part of the links having low load and promotes mild congestion, whereas Greedy concentrates heavily on high utilizations (hence is riskier in terms of congestion). On the contrary, the proposed DFRP realizes load balance consolidation by reducing traffic on a number of links without pushing traffic to the highly congested area and facilitate energy saving conditionally safe operation.

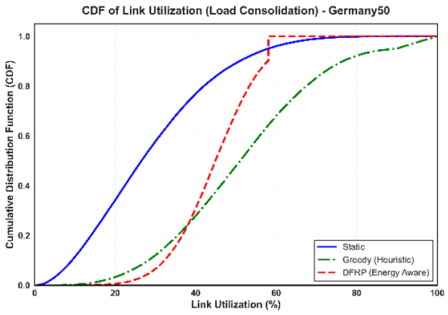
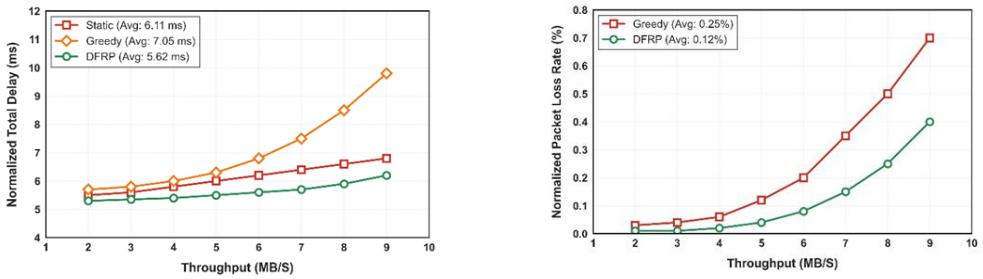


Figure 14. CDF of Link Utilization (Load Consolidation Effect).

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 April 2026 doi:10.20944/preprints202604.0666.v1



a) Throughput Vs Delay b) Throughput Vs Packet Loss

Figure 15. Throughput Vs Total Delay and Packet Loss Rate.

Figure 16 shows the energy–stability trade-off for the Germany50 topology by plotting average energy consumption against the average rule churn. The highest energy cost (≈12,500 units) is that of the Static strategy (zero churn), while the Greedy approach lowers it to ≈11,000 units at worst stability extremes (≈1,850 rule updates). The DFRP proposed in contrast, achieves a balanced operating point, reducing energy consumption to ≈9, 400 while maintaining high update rate of ≈1,320 hence showing better energy-efficient but stable SDN.

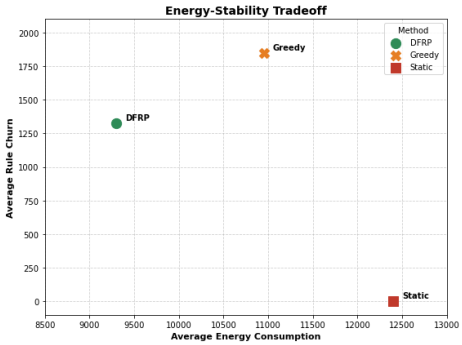


Figure 16. Energy-Stability Trade off.

Figure 17 (radar chart) compares Static, Greedy, and the proposed DFRP algorithm on five performance metrics for the Germany50 graph. DFRP always has the best-rounded performance, outperforming both Static and Greedy as to energy efficiency, latency, scalability, and load balancing. Although Static is more stable, DFRP has a better energy-stability trade-off with much improved overall multi-objective performance.

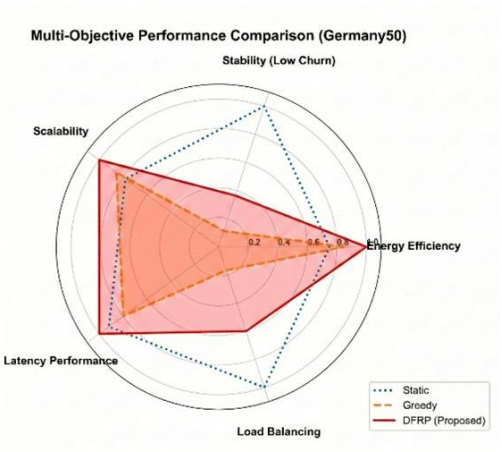


Figure 17. Multi-Objective Performance Comparison (Germany50).

 doi:10.20944/preprints202604.0666.v1

4.9. Time-Complexity Analysis

The time complexity of the proposed DFRP framework is dominated by the solution of the convex optimization problem executed at each time slot. With $|V|$ switches, $|E|$ Links, $|F|$ traffic flows and a receding horizon of length H, the number of variables grows as $O(H(|V|+|E|+|F|))$. Using a first-order solver such as OSQP, the per slot complexity scales polynomially and remains practical for small horizons ($H \leq 6$). Since only the first control action is applied, the total runtime increases linearly with the number of time slot. The optimization problem contains linear constraints (flow conservation, capacity and activation coupling) and a convex objective with L1 -norm terms. In the worst case, the per slot complexity is

$$O(\kappa \cdot H^3(|V|+|E|+|F|)^3),$$

Where kappa (κ)is the number of times a solver has to run prior to it gets a solution.

4.10. Performance Comparision

Tables 6 and 7 present a performance comparisons of the static, greedy, and proposed DFRP routing strategies across the Germany50 and Nobel-Germany topologies respectively that were tested for the complete 24-hour simulation period, summarizing their performance in terms of energy consumption, control-plane overhead, infrastructure utilization, and packet loss.

Table 6. Performance Comparison Germany50.

Metric	Static (Baseline)	Greedy Heuristic	DFRP (Proposed)	Remark
Total Energy (24h) [Joules]	3.57 × 10^6	3.15 × 106	2.68 × 106	DFRP is better 24.8% vs Static and 14.9% vs Greedy
Total Rule Churn (Updates)	0	5.32 × 10^5	4.25 × 10^5	DFRP is better 20.1% vs Greedy
Avg. Active Switches	50.00	45.63	39.05	DFRP is better 21.9% vs Static and 14.4% vs Greedy
Avg. Active Links	242.00	182.97	152.45	DFRP is better 37.0% vs Static and 16.6% vs Greedy
Max Link Utilization [%]	50.0%	100.0%	~58.1%	DFRP is better 42% Lower vs Greedy (Safer and Avoids congestion)
Avg. Packet Loss [%]	N/A	~0.25%	~0.12%	DFRP is better 52% Lower vs Greedy

Table 7. Performance Comparison Nobel-Germany.

Metric	Static (Baseline)	Greedy Heuristic	DFRP (Proposed)	Remark
Total Energy (24h) [Joules]	1.16 × 10^6	2.38 × 10^5	2.09× 10^5	DFRP is better 82.0% vs Static and 12.1% vs Greedy
Total Rule Churn (Updates)	0	2.37 × 10^4	1.93 × 10^4	DFRP is better 18.4% vs Greedy
Avg. Active Switches	17.00	3.73	3.25	DFRP is better 80.8% vs Static and 12.8% vs Greedy

 doi:10.20944/preprints202604.0666.v1

Metric	Static (Baseline)	Greedy Heuristic	DFRP (Proposed)	Remark
Avg. Active Links	66.00	8.36	7.58	DFRP is better 9.3% vs Greedy
Max Link Utilization [%]	50.0%	219% (Congested)	89%	DFRP is better 59% Lower vs Greedy, Avoids severe congestion
Avg. Packet Loss [%]	N/A	~0.35%	~0.32%	DFRP is better 9% Lower vs Greedy

5. Conclusions

The study presented a Dynamic Flow Rule Placement (DFRP) framework for real-time energy efficiency in SDN that operates under dynamic traffic conditions. The suggested method looks at more than just short-term energy savings. It looks at network energy use, TCAM use, and how stable control-plane rule changes are all at the same time. DFRP keeps network reconfiguration stable and efficient over time by using convex relaxation, minimum-edit rule placement, batch scheduling, and receding horizon control. The framework was tested on real-world SNDlib backbone topologies and showed that it used less energy and had less overhead in the control plane than static and greedy techniques. Overall, DFRP is a useful and balanced technique to combine conservative static routing with aggressive energy-driven heuristics. This makes SDN control more robust and energy-efficient. There are multiple methods that improve the suggested DFRP architecture. The initial future research is going to investigate through adaptive and resilient traffic predictor models to enhanced the quality of decision-making when demand trends change a lot. Second, introducing integer-aware or multiple-integer convex solvers that could assist to fill in the gaps that the relaxation generates while still making the issue easy to solve by splitting it into smaller portions or employing hierarchical control. Third, the energy model can be enlarged stronger by including different kinds of switch topologies, power states at the port level, and the energy the controller uses to analyse information. This makes it possible to optimize more accurately. Also, placing constraints on control plane latency and signalling would help us have a better idea of how much deployment overhead there is. Last but not least, testing DFRP on emulation platforms or programmable data planes and adding support for multi-controller or multi-domain SDN scenarios are both viable approaches to make it more relevant in the real world.

Author Contributions: "Conceptualization, S.B., and S.S.P..; methodology, N.P., and S.S.P.; software, S.S.P, and N.P; validation, S.B., and N.P.; formal analysis, S.B. and N.P.; investigation, S.S.P.; resources, S.B, and S.S.P.; data curation, S.B.; writing—original draft preparation, S.B.; writing—review and editing, N.P., and S.S.P.; visualization, N.P.; supervision, S.S.P.; project administration, N.P. All authors have read and agreed to the published version of the manuscript.".

Funding: "This research received no external funding".

Data Availability Statement: No datasets were generated or analyzed during the current study.

Conflicts of Interest: "The authors declare no conflicts of interest.".

References

1. Farhady, H., Lee, H., & Nakao, A. (2015). Software-defined networking: A survey. Computer Networks, 81, 79-95.
2. Rout, S., Sahoo, B., Patra, S. S., & Sahoo, S. S. (2018). Energy efficient routing in software defined networking by efficient traffic rule monitoring techniques. Journal of Advanced Research in Dynamical and Control Systems (JARDCS), 10(14), 650-662.

 doi:10.20944/preprints202604.0666.v1

3. Ben Said, Rachid & Tam, Sakirin & Tanriover, Omer. (2022). Rule Placement-Based Energy-Aware Routing in SDN: Review. 10.1007/978-981-19-2130-8_91
4. Qi, P., Pan, C., Xu, X., Wang, J., Liang, J., & Zhou, W. (2025). A Review of Dynamic Traffic Flow Prediction Methods for Global Energy-Efficient Route Planning. Sensors, 25(17), 5560. https://doi.org/10.3390/s25175560
5. S. Behera, N. Panda, S. S. Patra, T. Khan and L. Barik, "Flow-Rule Placement in Software-Defined Networks for Energy-Aware Routing," 2025 International Conference on Sustainable Communication Networks and Application (ICSCN), Theni, India, 2025, pp. 367-373, doi: 10.1109/ICSCN67106.2025.11308572.
6. Akin, E., & Korkmaz, T. (2019). Comparison of routing algorithms with static and dynamic link cost in software defined networking (SDN). IEEE Access, 7, 148629-148644.
7. Wang, R., Jiang, Z., Gao, S., Yang, W., Xia, Y., & Zhu, M. (2014, June). Energy-aware routing algorithms in software-defined networks. In Proceeding of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks 2014 (pp. 1-6). IEEE.
8. Moin, S., Karim, A., Safdar, K., Iqbal, I., Safdar, Z., Vijayakumar, V., … & Abid, S. A. (2018). GREEN SDN — An enhanced paradigm of SDN: Review, taxonomy, and future directions. Concurrency and Computation: Practice and Experience, 32(21). https://doi.org/10.1002/cpe.5086
9. Forghani M, Soltanaghaei M, Boroujeni FZ (2024) Dynamic optimization scheme for load balancing and energy efficiency in software-defined networks utilizing the krill herd meta-heuristic algorithm. Comput Electr Eng 114:109057
10. El-Hefnawy, N. A., Raouf, O. A., and Askr, H. (2022). Dynamic routing optimization algorithm for software defined networking. Computers, Materials and Continua, Vol. 70, No. 1, pp. 69–89.
11. Chen, J., Xiao, W., Zhang, H., Zuo, J., & Li, X. (2024). Dynamic routing optimization in software-defined networking based on a metaheuristic algorithm. Journal of Cloud Computing, 13(1), 41.
12. S. Bera, S. Misra and A. Jamalipour, "FlowStat: Adaptive Flow-Rule Placement for Per-Flow Statistics in SDN," in IEEE Journal on Selected Areas in communications, vol. 37, no. 3, pp. 530-539, March 2019, doi: 10.1109/JSAC.2019.2894239.
13. Agg, P. A., & Johanyák, Z. C. (2023). Saving Energy Using the Modified Heuristic Algorithm for Energy Saving (MHAES) in Software-Defined Networks. Sensors, 23(23), 9581. https://doi.org/10.3390/s23239581
14. Nguyen, X. N., Saucez, D., Barakat, C., & Turletti, T. (2015, April). OFFICER: A general optimization framework for OpenFlow rule allocation and endpoint policy enforcement. In 2015 IEEE Conference on Computer Communications (INFOCOM) (pp. 478-486). IEEE.
15. Giroire F, Huin N, Moulierac J, Phan TK (2018) Energy-aware routing in software-defined network using compression. Comput J 61(10):1537–1556. https://doi.org/10.1093/comjnl/bxy029. ⟨hal-01920970⟩
16. Riveros-Rojas, G. J., Cespedes-Sanchez, P. P., Pinto-Roa, D. P., & Legal-Ayala, H. (2024). Energy-and-Blocking-Aware Routing and Device Assignment in Software-Defined Networking—A MILP and Genetic Algorithm Approach. Mathematical and Computational Applications, 29(2), 18. https://doi.org/10.3390/mca29020018
17. Baker, T., Al Aghbari, Z., Khedr, A. M., Ahmed, N., & Girija, S. (2024). EDITORS: Energy-aware DynamIc Task Offloading using Deep Reinforcement transfer learning in SDN-enabled edge nodes. Internet of Things, 25, 101118.
18. Said, R.B., Tam, S., Tanriover, O.O. (2022). Rule Placement-Based Energy-Aware Routing in SDN: Review. In: Sharma, H., Shrivastava, V., Kumari Bharti, K., Wang, L. (eds) Communication and Intelligent Systems. Lecture Notes in Networks and Systems, vol 461. Springer, Singapore. https://doi.org/10.1007/978-981-19-2130-8_91
19. Li W, Qin Z, Li K, Yin H, Ou L (2019) A novel approach to rule placement in software-defined networks based on OPTree. IEEE Access 7:8689–8700. https://doi.org/10.1109/ACCESS.2018.2889194
20. P. Kamboj and S. Pal, "Energy-Aware Routing in SDN Enabled Data Center Network," 2022 IEEE 21st International Symposium on Network Computing and Applications (NCA), Boston, MA, USA, 2022, pp. 123-130, doi: 10.1109/NCA57778.2022.10013588.
21. Jalili, A. (2025). Enhancing quality of service in SDNs through Pareto-optimized controller placement using NS-MF algorithm. International Journal of Nonlinear Analysis and Applications, 16(9), 157-167.

 doi:10.20944/preprints202604.0666.v1

22. Chen, T., Ling, Q., and Giannakis, G. B. (2017). An online convex optimization approach to proactive network resource allocation. IEEE Transactions on Signal Processing, Vol. 65, No. 24, pp. 6350–6364.
23. Mattingley, J., Wang, Y., and Boyd, S. (2011). Receding horizon control. IEEE Control Systems Magazine, Vol. 31, No. 3, pp. 52–65.
24. Wang, T., Liu, F., and Xu, H. (2017). An efficient online algorithm for dynamic SDN controller assignment in data center networks. IEEE/ACM Transactions on Networking, Vol. 25, No. 5, pp. 2788–2801
25. Assefa, B. G., and Özkasap, Ö. (2019). A survey of energy efficiency in SDN: Software-based methods and optimization models. Journal of Network and Computer Applications, Vol. 137, pp. 127–143.
26. Orlowski, S., Wessäly, R., Pióro, M., and Tomaszewski, A. (2010). SNDlib 1.0—Survivable network design library. Networks, Vol. 55, No. 3, pp. 276–286.
27. Wang, S., Zhang, X., Zhang, J., Feng, J., Wang, W., & Xin, K. (2015, September). An approach for spatial-temporal traffic modeling in mobile cellular networks. In 2015 27th International Teletraffic Congress (pp. 203-209). IEEE.
28. Naseri, A., Ahmadi, M., & PourKarimi, L. (2023). Placement of SDN controllers based on network setup cost and latency of control packets. Computer Communications, 208, 15-28.
29. Zhang, S., Ivancic, F., Lumezanu, C., Yuan, Y., Gupta, A., & Malik, S. (2014, June). An adaptable rule placement for software-defined networks. In 2014 44th annual IEEE/IFIP international conference on dependable systems and networks (pp. 88-99). IEEE.
30. Dolati, M., Khonsari, A., & Ghaderi, M. (2018, November). Consistent SDN rule update with reduced number of scheduling rounds. In 2018 14th International Conference on Network and Service Management (CNSM) (pp. 254-260). IEEE.
31. Forough, A. B., & Roshandel, R. (2017). Multi objective receding horizon optimization for optimal scheduling of hybrid renewable energy system. Energy and Buildings, 150, 583-597.
32. de Prada, C., Mazaeda, R., & Cristea, S. P. (2019). Receding horizon scheduling of processes with shared resources. Computers & Chemical Engineering, 125, 1-12.
33. Stellato, B., Banjac, G., Goulart, P., Bemporad, A., & Boyd, S. (2020). OSQP: An operator splitting solver for quadratic programs. Mathematical Programming Computation, 12(4), 637-672.
34. Diamond, S., & Boyd, S. (2016). CVXPY: A Python-embedded modeling language for convex optimization. Journal of Machine Learning Research, 17(83), 1-5.