# Paper 4: Distributed Service Function Chaining (DSFC) – In-Depth Explanation

## 1. Introduction

### 📌 What is this paper about?

Traditional centralized VNF placement leads to resource inefficiency and high costs. This paper proposes DSFC, a scalable, decentralized approach to optimally distribute VNFs across multiple machines.

### 📌 Why is this important?

• Centralized VNF placement assumes all VNFs are identical, leading to inefficiencies.  
• Bottlenecks occur when all VNFs are placed on a single machine.  
• A balance is needed between cost, latency, and resource efficiency.

### 📌 Key Objectives of DSFC:

✅ Decouples throughput from physical machine limits.  
✅ Improves scalability by distributing VNFs across multiple nodes.  
✅ Reduces congestion by optimizing network-wide utilization.  
✅ Uses heuristic (Kariz) to solve large-scale placement efficiently.

## 2. Understanding Service Function Chaining (SFC) & VNFs

### 📌 What is an SFC?

A Service Function Chain (SFC) is an ordered sequence of VNFs that packets must pass through before reaching their destination.

### 📌 Example of an SFC:

1️⃣ Firewall (FW): Protects against malicious traffic.  
2️⃣ Intrusion Detection System (IDS): Monitors and detects security threats.  
3️⃣ Network Address Translation (NAT): Modifies packet IP addresses.  
4️⃣ Traffic Optimizer (TO): Prioritizes traffic based on QoS.

### 📌 How Are SFCs Placed in a Network?

✅ Minimize latency & cost.  
✅ Ensure efficient resource utilization.  
✅ Distribute load to avoid congestion.

## 3. Problem with Centralized SFC Placement

### 📌 Limitations of Centralized Approaches:

❌ VNFs are placed on limited physical machines, creating bottlenecks.  
❌ Static placement causes congestion & resource underutilization.  
❌ High operational costs due to inefficient resource allocation.  
❌ No global coordination → suboptimal network performance.

### 📌 How DSFC Fixes This?

✅ Distributes VNFs across multiple physical machines.  
✅ Optimizes network resource usage while keeping costs low.  
✅ Uses a heuristic (Kariz) for fast, near-optimal placement.

## 4. Mathematical Model: Mixed Integer Programming (MIP) for DSFC

### 📌 Why MIP?

MIP (Mixed Integer Programming) finds optimal VNF placement while ensuring efficient resource allocation and minimizing cost and congestion.

### 📌 Decision Variables in MIP:

1️⃣ x\_{s,n} = 1 → If VNF s is placed at node n, otherwise 0.  
2️⃣ y\_{s,t} = 1 → If VNF s is migrated from node t, otherwise 0.  
3️⃣ U\_l → Bandwidth cost function.  
4️⃣ C\_n → Host resource allocation cost.

### 📌 Objective Function (Minimizing Cost & Migrations):

1️⃣ VNF Placement Cost:  
min Σ C\_n \* x\_{s,n} + Σ B\_l \* U\_l + Σ M\_t \* y\_{s,t}

### 📌 Constraints:

1️⃣ Each VNF must be placed exactly once: Σ x\_{s,n} = 1, ∀ s  
2️⃣ Node Resource Capacity Constraints: Σ x\_{s,n} \* R\_s ≤ C\_n, ∀ n  
3️⃣ Bandwidth Constraints: Σ U\_{s,l} ≤ B\_l, ∀ l

## 5. Heuristic Approach for Faster Placement – 'Kariz'

### 📌 Why Use a Heuristic?

✅ MIP is too slow for large-scale networks.  
✅ Kariz heuristic finds a near-optimal solution in seconds.

### 📌 How Kariz Works?

1️⃣ Sort SFC requests by priority (e.g., latency-sensitive chains first).  
2️⃣ Assign VNFs greedily to the lowest-latency node available.  
3️⃣ Ensure network capacity is not exceeded.

## 6. Results & Performance Analysis

### 📌 Experimental Setup:

• Fat-tree topology with 99 nodes and 162 links.  
• VNFs: Firewall, IDS, IPSec, WAN optimizer.  
• Comparison: Kariz heuristic vs. MIP model.

### 📌 Key Findings:

✅ Kariz achieves 76%-100% acceptance ratio (close to MIP).  
✅ Kariz incurs 24% lower operational cost compared to MIP.  
✅ Kariz is significantly faster, making it practical for real-world networks.

## 7. Key Takeaways from Paper 4

✅ DSFC distributes VNFs across multiple machines, improving throughput and efficiency.  
✅ MIP provides optimal VNF placement but is computationally expensive.  
✅ Kariz heuristic offers a fast, near-optimal alternative for large-scale deployments.  
✅ DSFC enhances network resource utilization while reducing costs and latency.