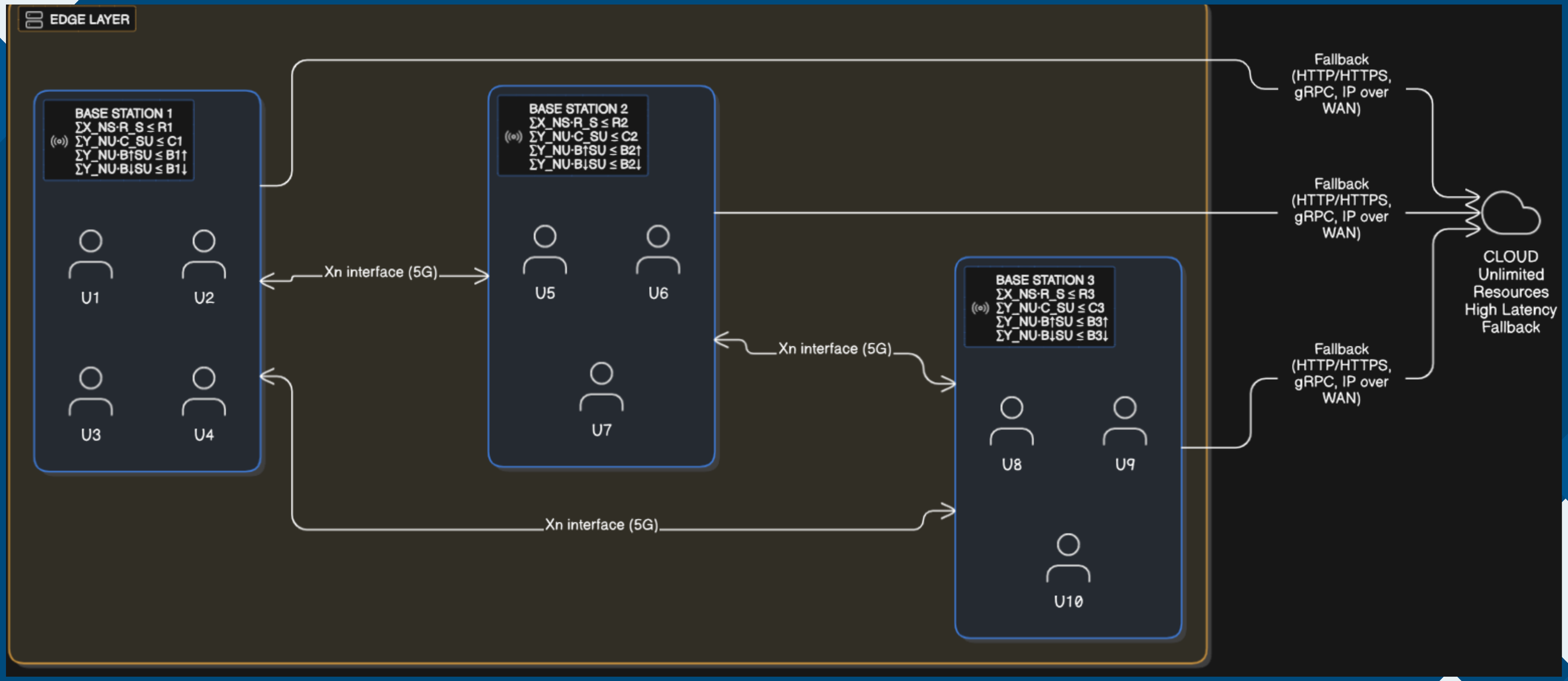
JOINT SERVICE PLACEMENT & REQUEST ROUTING IN MEC NETWORK

Presented by: Group 9

Problem Statement

JSPRR(Joint Service Placement and Request Routing) is a problem to place services and users in base stations such that the load to the cloud is minimised

Architecture



Architecture

- Communication between user and base station makes use of 5G which reliable, have low latency and has mobility support.
- Communication between base stations makes use of Xn interface, sometimes SDN/OpenFlow for custom which supports Load balancing, Handover, Service migration.
- Communication between base station and cloud makes use of HTTP/HTTPS, gRPC, QUIC, TCP/IP which supports web based services, low latency and reliable delivery over long-haul networks.

Algorithms Used

- Service Placement - Linear Programming relaxation means taking an integer programming problem and relaxing it by allowing those variables to take fractional values instead.
- Request Routing - Randomized Rounding Algorithm is a technique in approximation algorithms where fractional solutions are rounded to integers at random, with probabilities proportional to their fractional values, to route the service randomly to the bs.
- Dijkstra’s Algorithm - To find the nearest bs for the service.
- GA + AOC (other way of implementation)

LP Relaxation

LP relaxation is a standard technique in optimization

- Relax the integrality constraints on placement variables:

$$x_{i,v} \in [0,1] \quad \text{instead of } \{0,1\}$$

- Flow variables $f_{k,a}$ remain continuous ≥ 0.

Interpretation:

- $x_{i,v} = 0.7$ means "service i is fractionally placed 70% on node v" — a **probabilistic or fractional guidance**, not literal deployment.
- The LP now becomes a **linear program (LP)**, solvable in **polynomial time**.

Request Routing

Request routing is the process of directing user service requests to the most appropriate network resource (base station, edge server, or cloud) based on capacity, latency, and cost.

- In your JSPRR code, the ILP/LP solver assigns services to base stations/cloud.
- Currently, it’s an LP relaxation, meaning variables xsbx_{sb}xsb can be fractional (e.g., 0.3, 0.7).
- Fractional assignment is not realistic in practice:
- A service either runs on a base station (1) or it doesn’t (0).

Request Routing

- Solve LP Relaxation
- Use SimplexSolver to get fractional assignments xsb∈[0,1]x_{sb} \in [0,1]xsb∈[0,1].
- Randomized Rounding
- For each service sss:
  - Pick a base station bbb randomly, weighted by xsbx_{sb}xsb.
  - Assign the service to that base station (set xsb=1x_{sb}=1xsb=1, others 0).
- Check Capacity Constraints
- After rounding, some base stations may slightly exceed capacities.
- Apply simple adjustments:
  - Reassign overloaded services to less-loaded bases.
  - Or allow small violations if acceptable.
- Request Routing Execution
- The Routing Manager sends the user request to the chosen base station or cloud.
- Service executes there, responses follow the routed path.

Sample Results

```
LP fractional solution:
A: [0.0, 1.0, 0.0, 0.0, 0.0, 0.0]
B: [0.0, 1.0, 0.0, 0.0, 0.0, 0.0]
C: [0.0, 0.0, 0.3999999999999998, 0.0, 0.6000000000000002, 0.0]
D: [0.0, 0.0, 0.0, 0.0, 0.9999999999999997, 0.0]
E: [0.0, 0.0, 1.0, 0.0, 1.1102230246251568E-16, 0.0]
F: [0.0, 0.0, 1.0, 0.0, 0.0, 0.0]
G: [0.0, 0.0, 1.0, 0.0, 4.5102810375396676E-17, 0.0]
H: [0.0, 0.0, 0.9999999999999999, 0.0, 0.0, 0.0]
```

```
Link usages:
L1 (BS1-BS2): used 30.00 / cap 120.00
L2 (BS2-BS3): used 30.00 / cap 100.00
L3 (BS3-BS4): used 25.00 / cap 90.00
L4 (BS4-BS5): used 25.00 / cap 80.00
L5 (BS1-BS3): used 0.00 / cap 110.00
L6 (BS2-BS4): used 0.00 / cap 100.00
L7 (BS1-BS5): used 30.00 / cap 70.00
Lcloud1 (BS1-Cloud): used 0.00 / cap 500.00
Lcloud2 (BS2-Cloud): used 15.00 / cap 500.00
Lcloud3 (BS3-Cloud): used 42.00 / cap 500.00
Lcloud4 (BS4-Cloud): used 0.00 / cap 500.00
Lcloud5 (BS5-Cloud): used 0.00 / cap 500.00
```

```
Final placement:
A -> BS2
B -> BS2
C -> BS3
D -> BS5
E -> BS3
F -> BS3
G -> BS3
H -> Cloud
```

CI APPROACH IN JSPRR

Goal:

Reduce cloud load by serving more requests at BSs , Improve resource utilization (storage, CPU, uplink, downlink).Ensure users get low-latency service.

Inputs

User service requests, Available BS resources (storage, CPU, bandwidth) ,Service popularity distribution (demand pattern).

Algorithms Used:

A mixture of GA and ACO algorithms used for both service and route optimization.

HYBRID TECHNIQUE(GA+ACO)

Initialize population with random + heuristic placements.
GA Phase: Evolve service placements.
Evaluate fitness with routing results.
ACO Phase: For each placement, ants build routing solutions. Update pheromones based on cloud load + constraints.
Repair Step: If BS overload occurs → reroute users / adjust placements.
Iterate until convergence.

GENETIC ALGORITHM FOR SERVICE PLACEMENT

Chromosome Encoding:

Each chromosome = mapping of services to BSs.

Fitness Function:

Minimize cloud-served requests.

Balance storage + computation + bandwidth.

Operators:

Selection → choose best placements.

Crossover → combine two solutions.

Mutation → explore new placements.

Outcome: Efficient placement of services across BSs.

ANT COLONY OPTIMIZATION FOR ROUTING

Problem: Each user request must be routed to one BS or cloud.

ACO Process:

Ants = simulated agents exploring routing paths.

Pheromone trails = reinforce good routes (low congestion, feasible).

Heuristic info = service availability, BS resource load.

Result:

Users routed to least congested BS that has the service.

Cloud used only as last resort.

Results & Conclusion

The hybrid GA+ACO approach improved the hit ratio by serving more users at the edge compared to random placement. Resource utilization across base stations was balanced, ensuring no single node was overloaded. Overall, the method achieved higher accuracy, reduced cloud dependency, and better scalability in edge service delivery.

```
Saving datasets.zip to datasets (1).zip
Extracted files: ['users.csv', 'services.csv', 'bs.csv']
[Gen 0] best_fit=25 mean_fit=16.08
[Gen 3] best_fit=26 mean_fit=24.50
[Gen 6] best_fit=29 mean_fit=26.58
[Gen 9] best_fit=30 mean_fit=28.50
[Gen 11] best_fit=30 mean_fit=29.25

===== Final Detailed Results =====
Total Users: 30
Edge-served Users: 30
Cloud-served Users: 0
Accuracy: 100.00%
Hit Ratio: 100.00%
Miss Ratio: 0.00%

Per-BS resource utilizations:
BS0: services=[0, 1, 2, 3, 5, 6, 7, 8, 9]
     storage_used=190.0/300.0GB (63.3%)
     cpu_used≈2.77 GHz
     uplink_used≈23.50 Mbps
     downlink_used≈61.00 Mbps

BS1: services=[1, 2, 4, 6, 7, 8, 9]
     storage_used=167.0/250.0GB (66.8%)
     cpu_used≈1.40 GHz
     uplink_used≈11.50 Mbps
     downlink_used≈29.00 Mbps

BS2: services=[0, 1, 3, 4, 5, 6, 7, 8, 9]
     storage_used=180.0/320.0GB (56.2%)
     cpu_used≈2.79 GHz
     uplink_used≈28.00 Mbps
     downlink_used≈66.00 Mbps
```

INDIVIDUAL CONTRIBUTION

EDGE TEAM:

REQUEST ROUTING – [ESWARA RAJ M-CSE23022]

SERVICE PLACEMENT-[AKSHAY R-CSE23007]

CLOSEST BASE STATIONS-[AMARTHYA SUJAI-CSE23003]

Sim & EDGE NODE SETUP-[CHARANJITH-CSE23713]

CI TEAM:

GA PHASE-[ANKITH-CSE23133]

ACO PHASE-[CHAKRAVARTHY-CSE23753]

THANK YOU