2026/8/3

Memory-Enabled Quantum Gene Regulatory Networks for Adaptive Network Slice Embedding

Kazuki Sekizawa, Masaaki Yamauchi, Daichi Kominami, Hideyuki Shimonishi,
Tatsuya Otoshi, and Masayuki Murata
The University of Osaka, Japan
ICCCN 2026 / AIECN

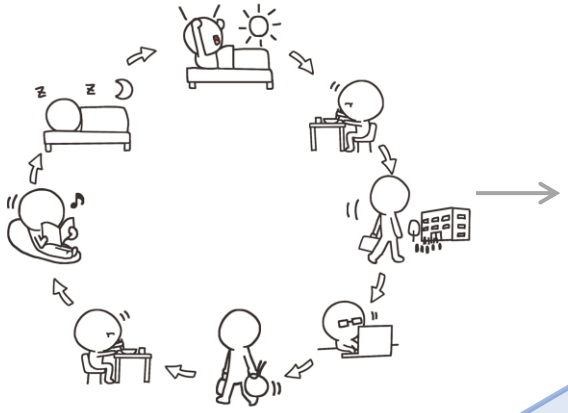
The University of Osaka

Background

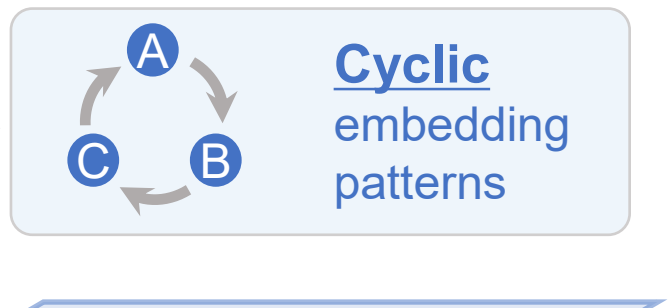
Existing approaches for network slice embedding problem do not sufficiently account for **multi-timescale dynamic conditions**.

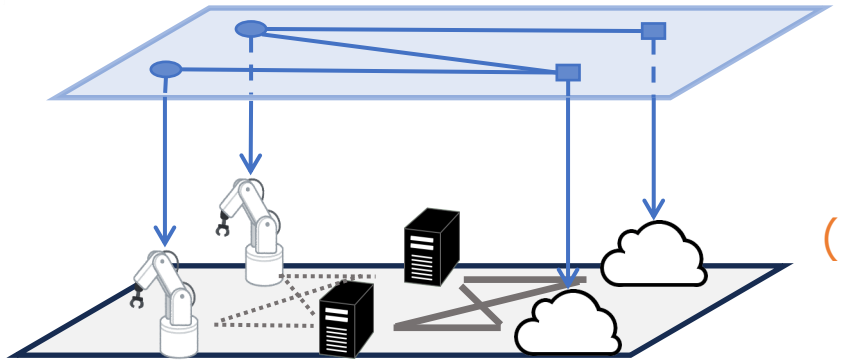
- **Cyclic fluctuations**: Recurrent changes by daily user activity patterns
- **Structural shifts**: Abrupt changes by infrastructure upgrades or link failures.

  → ML and RL approaches need external change-detection modules.



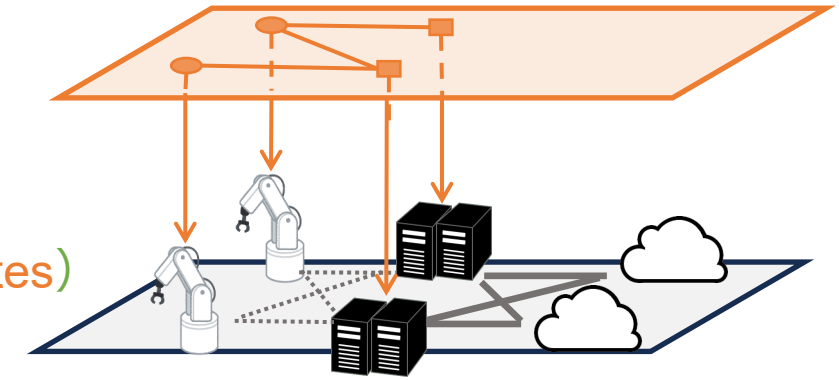
Daily activities





Structural shifts

(infrastructure updates)

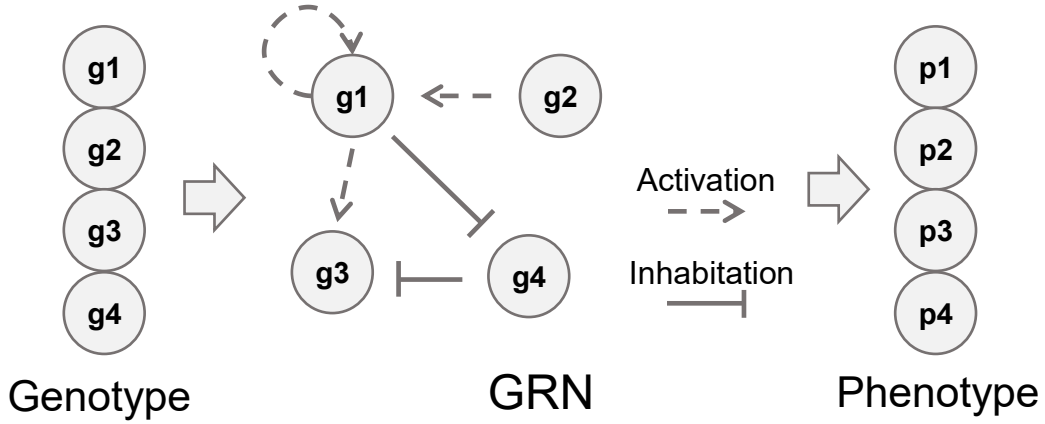


New embedding patterns are needed

Biologically Inspired Approach

Gene Regulatory Networks (GRNs)

- A network model representing gene-gene interactions



Key genetic biological mechanisms

Epigenetics

- The ability of a fixed genetic structure to express multiple states.

Cyclic fluctuations

- Rapidly recalls effective control strategies.

Enhanced by quantum GRN

Evolution

- Gradual modifications of the interaction structure.

Structural shifts

- Explores new control strategies for unseen environments.

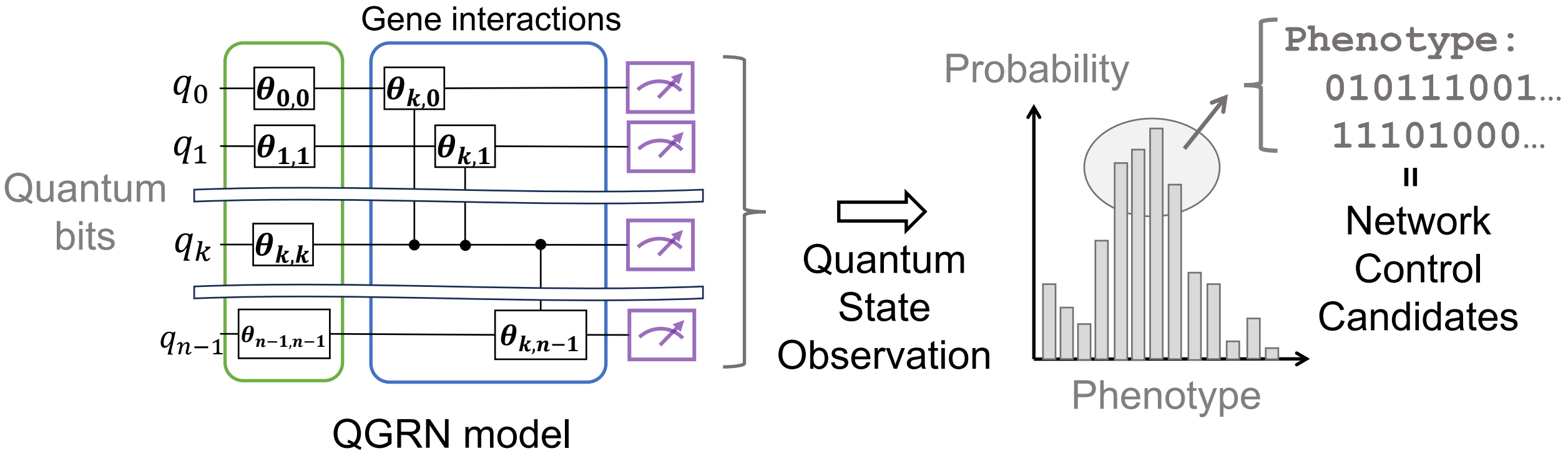
Enabling adaptation to multi-timescale dynamics w/o external change detection.

Quantum Gene Regulatory Networks (QGRNs)

QGRNs improve the capacity to **memorize and recall candidates.**

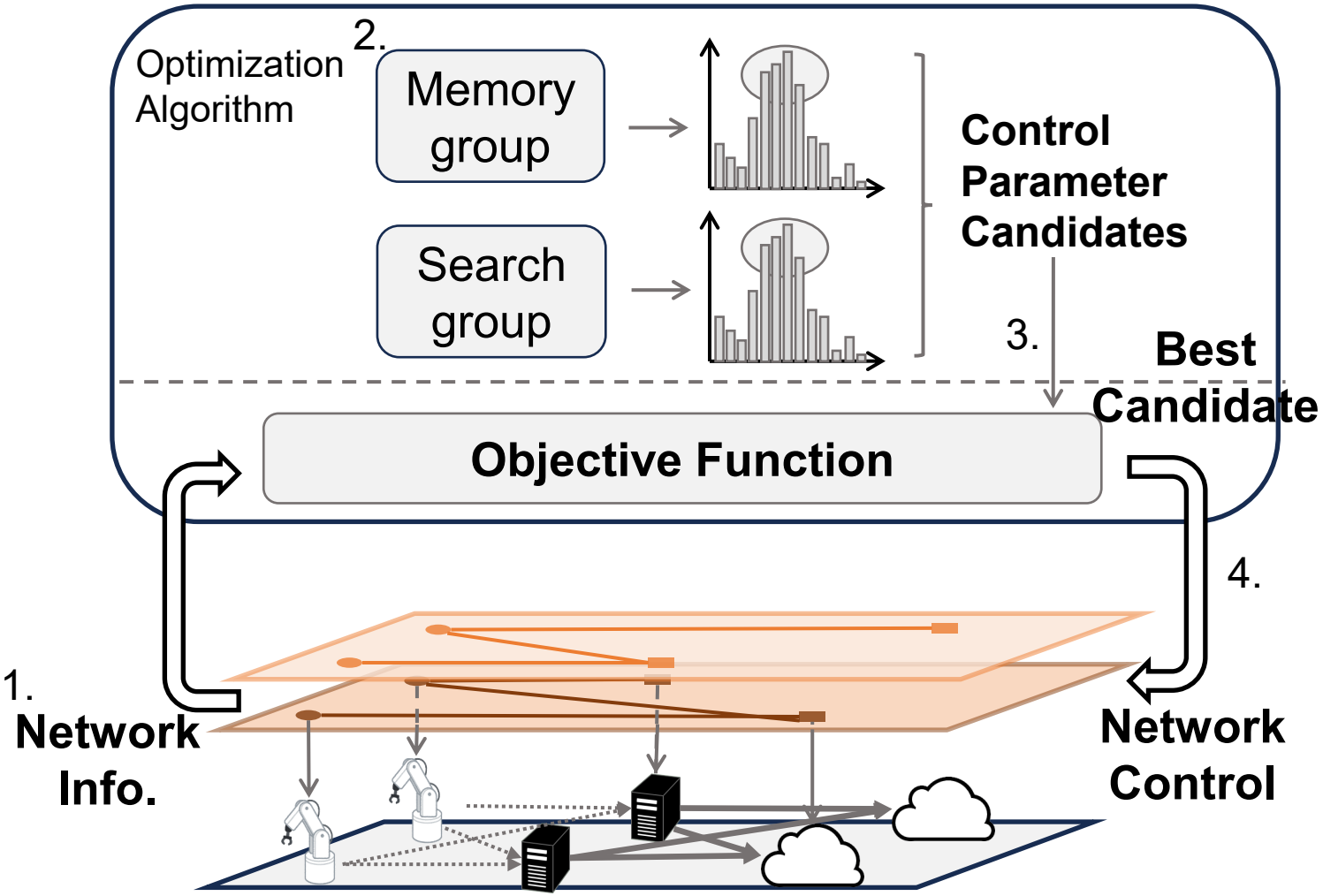
- Genes and their interactions are represented as quantum states and operations.
- A single QGRN yield a probabilistic distribution over phenotypes

→ A single QGRN structure can represent multiple network control strategies.



Framework Overview

The framework adapts to multi-timescale dynamics using 2 groups.

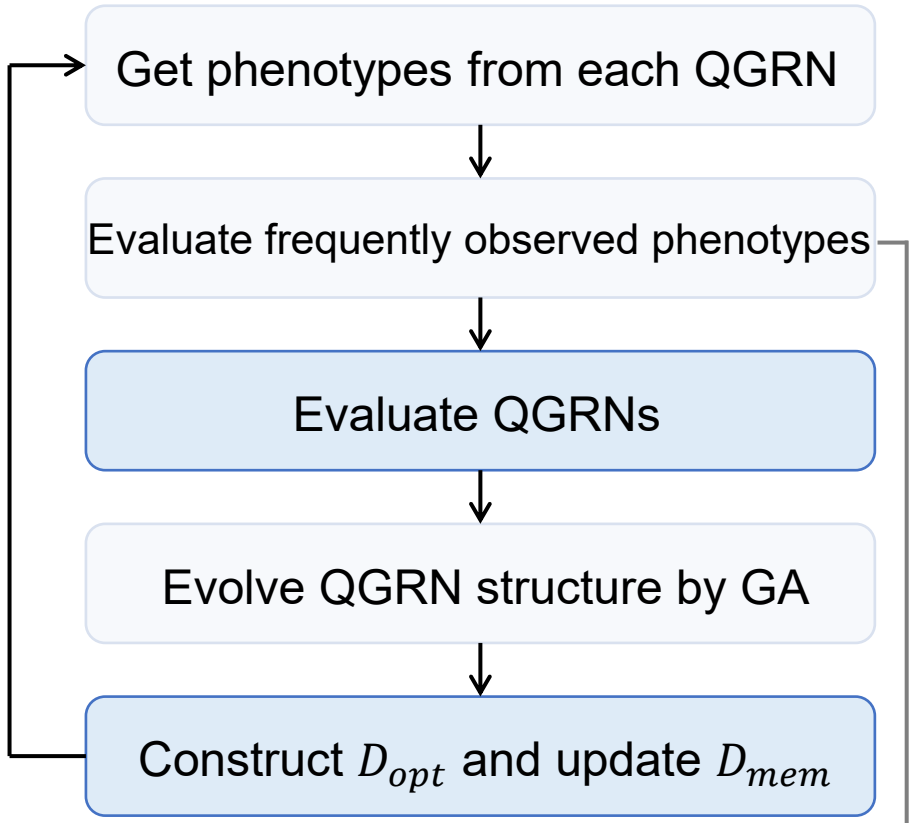


1. Collect network information.
2. Generate network-control strategies from QGRNs.
   - **Memory group** (for cyclic fluctuations):
     retains strategies from past environments
   - **Search group** (for structural shifts):
     explores strategies for the current environment
3. Evaluate the strategies.
4. Apply the best candidate to network control.

Algorithm overview

Evolving two groups and updating memory with high-performing phenotypes.

Algorithm flow



D_{opt} : Temporary distribution
D_{mem}: Memory target distribution

Evaluate QGRNs

- Memory group: Kullback–Leibler divergence

$$Fitness = -\sum D_{mem} \log(D_{mem}/D_i)\,.$$

- Search group: Mean objective value of the candidates

$$Fitness = -\sum_{i=1}^{n_{exp}} J\,(p_i)/n_{exp}.$$

Construct distributions for memory

$$D_{mem}(\tau+1) = \alpha D_{opt}(\tau) + (1-\alpha) D_{mem}(\tau)$$

Elites

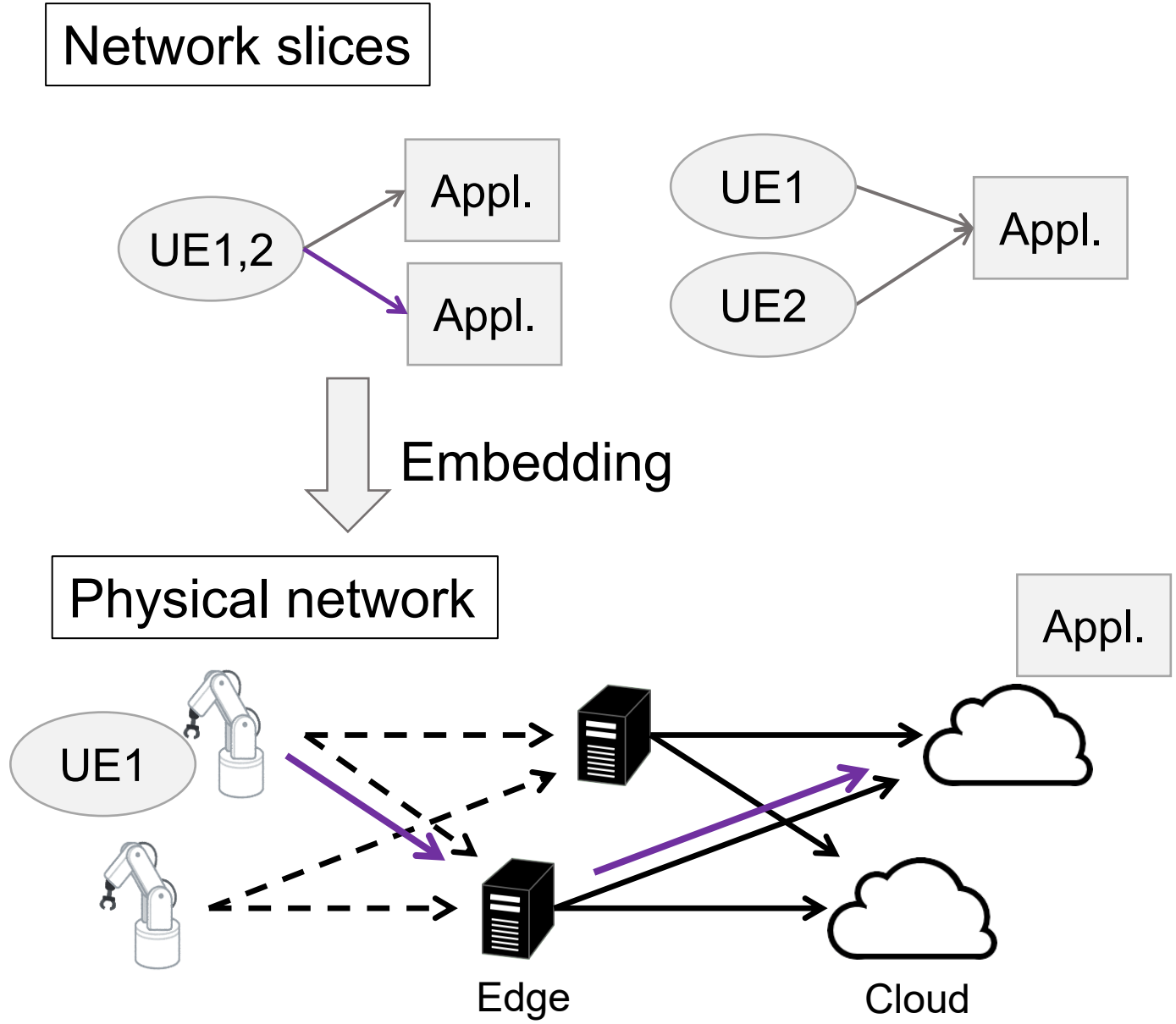
Evaluation Settings: Network topology

Virtual network slices

- 3 type slices
  - Virtual links and application nodes
  - Communication and computing requirements

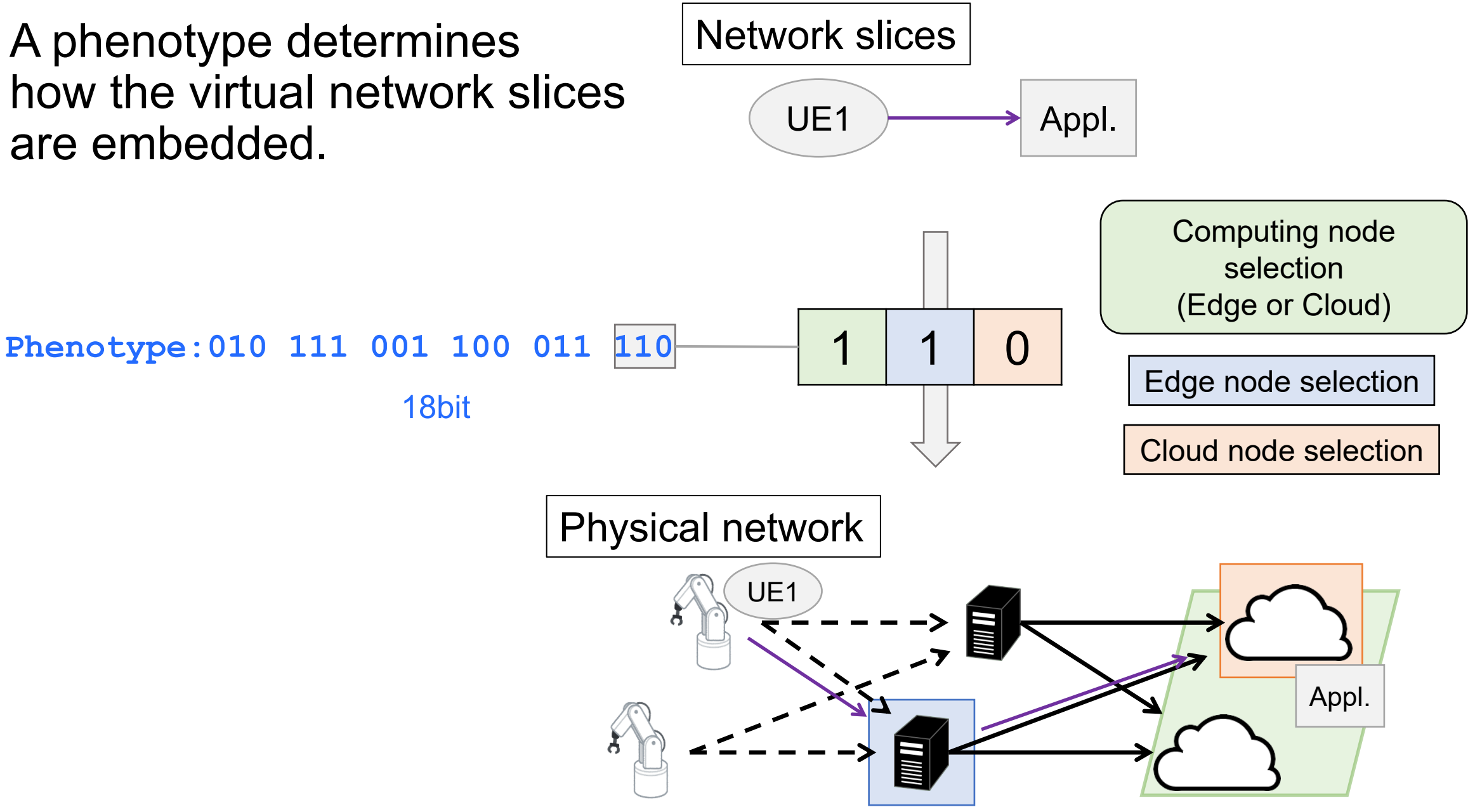
Physical network

- 2 UE, 2 edge servers, 2 cloud servers
  - Communication and computing capacities
- Wireless user-to-edge links
- Wired edge-to-cloud links



Evaluation Settings: **Phenotype-to-Decision Variable**

A phenotype determines how the virtual network slices are embedded.



Evaluation Settings: Problem Formulation

Objective is to **minimize total power consumption** while **satisfying all slice requirements**.

Objective function

$$J = \delta\left(E^{thr} + E^{com}\right) + (1 - \delta)\lambda$$

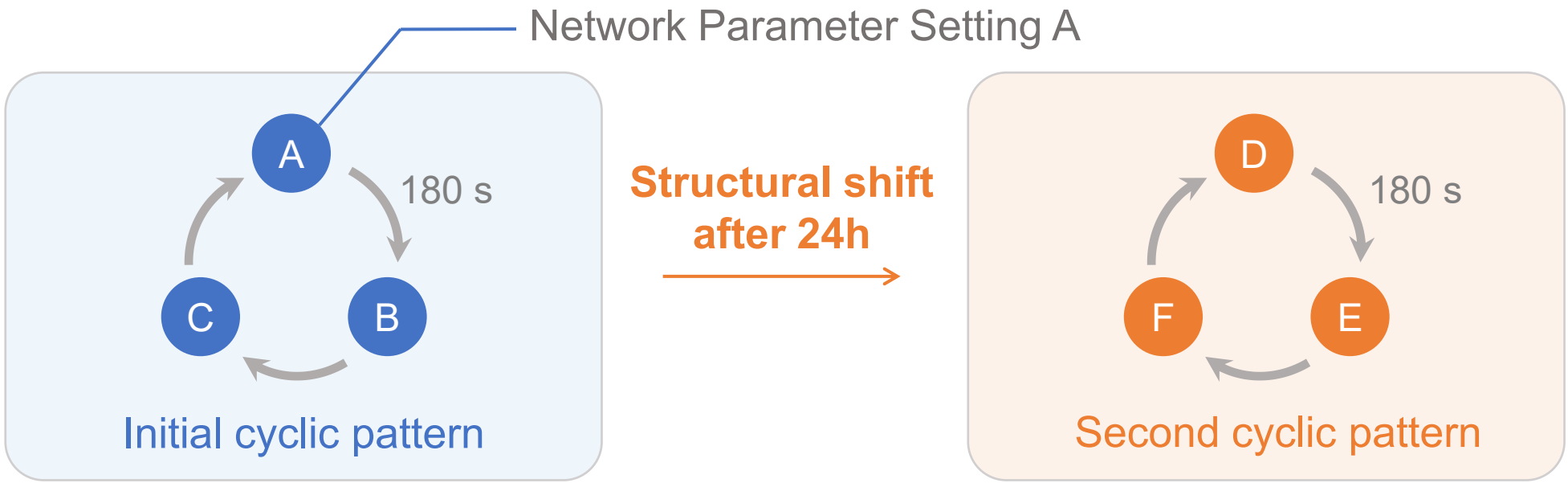
Variable	Description
E^{thr}	Transmission power
E^{com}	Computational power
λ	Penalty constant
δ	δ = 1 if all slices are embedded w/o violating any constraints, and 0 otherwise

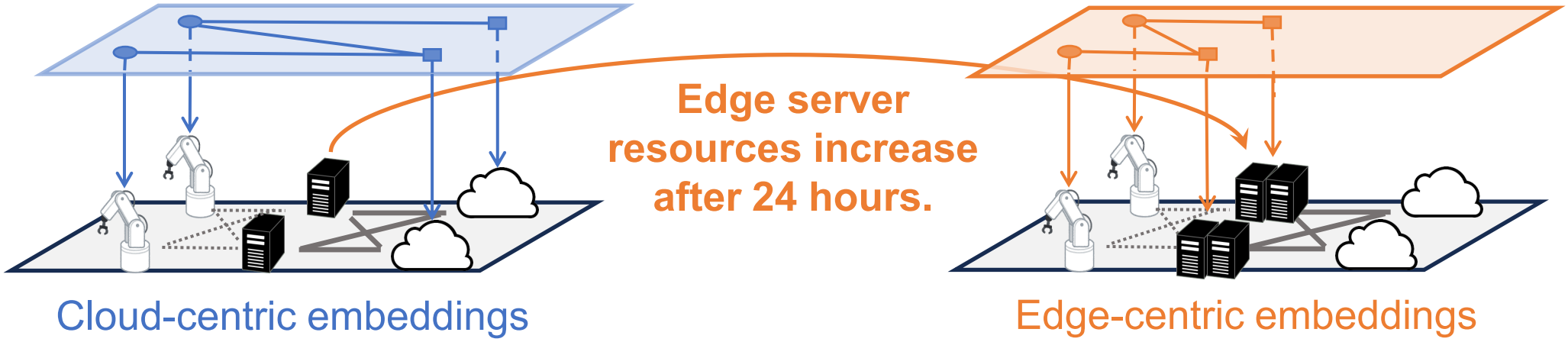
Constraints

- **Throughput**
  The total throughput of the virtual links mapped to each physical link must not exceed its capacity.
- **Latency**
  The total latency of the physical links used for each virtual link must not exceed its latency requirement.
- **Server resource**
  The total demand of the applications mapped to each node must not exceed its capacity.

Evaluation Settings: Dynamic Evaluation Scenario

We create a **multi-timescale scenario** by varying network parameters.



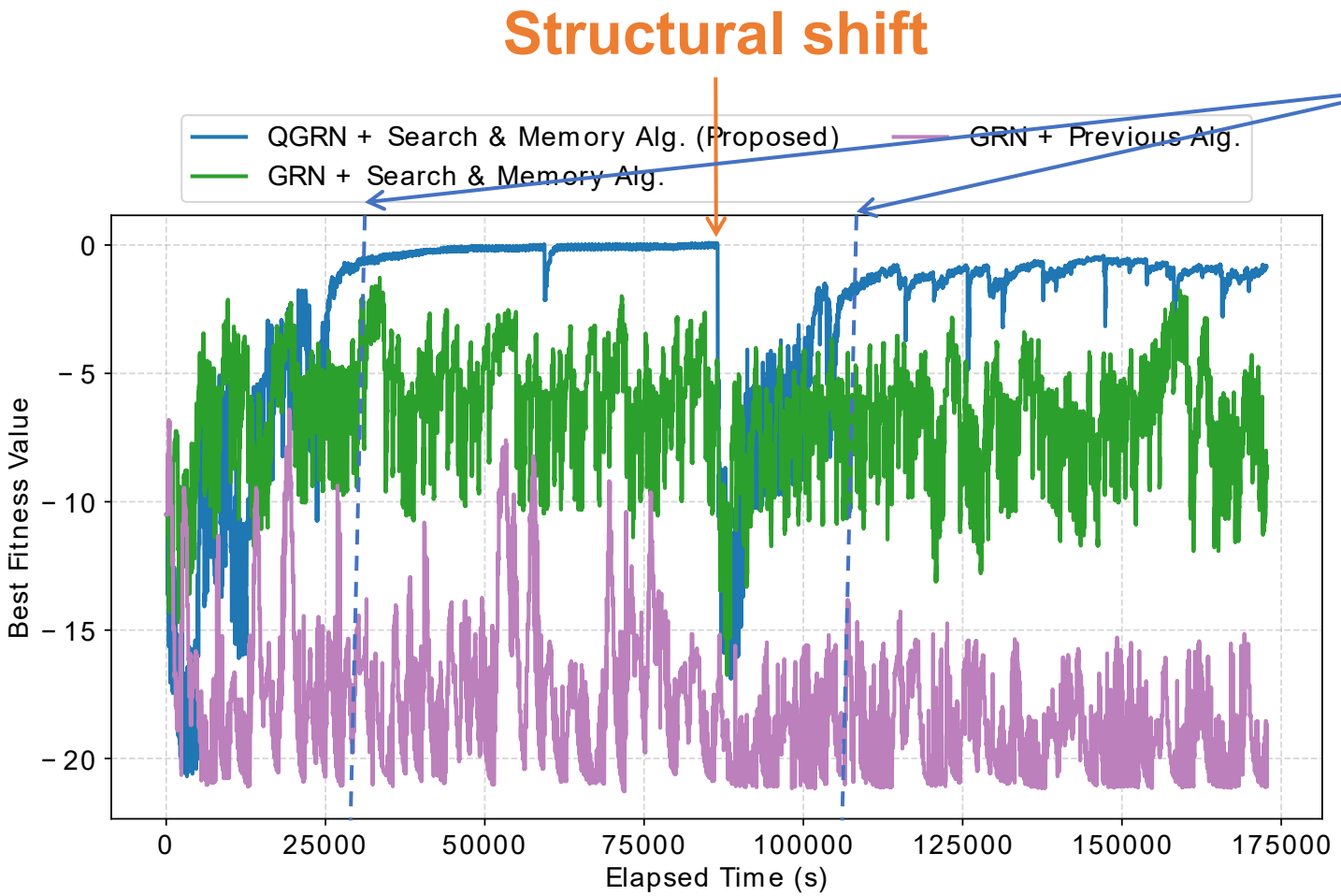


Evaluation Settings: Comparison Methods

Method	Purpose	Memory
QGRN + Search & Memory	Proposed method	✓
QGRN + Search Only	To isolate the effect of the memory group	-
GRN + Previous Algorithm	Conventional GRN baseline	✓
GRN + Search & Memory	To isolate the effect of the QGRN representation	✓
Simple GA	Conventional stochastic optimization baseline	-
Bandit (ϵ-greedy)	Reinforcement learning baseline	-

Result 1: Memory Stabilization

The proposed method **adapts to both cyclic changes and structural shifts.**



Proposed Algorithm (Blue)

- The fitness stabilized both cyclic changes and structural shift.

→ Adapting to multi-timescale conditions

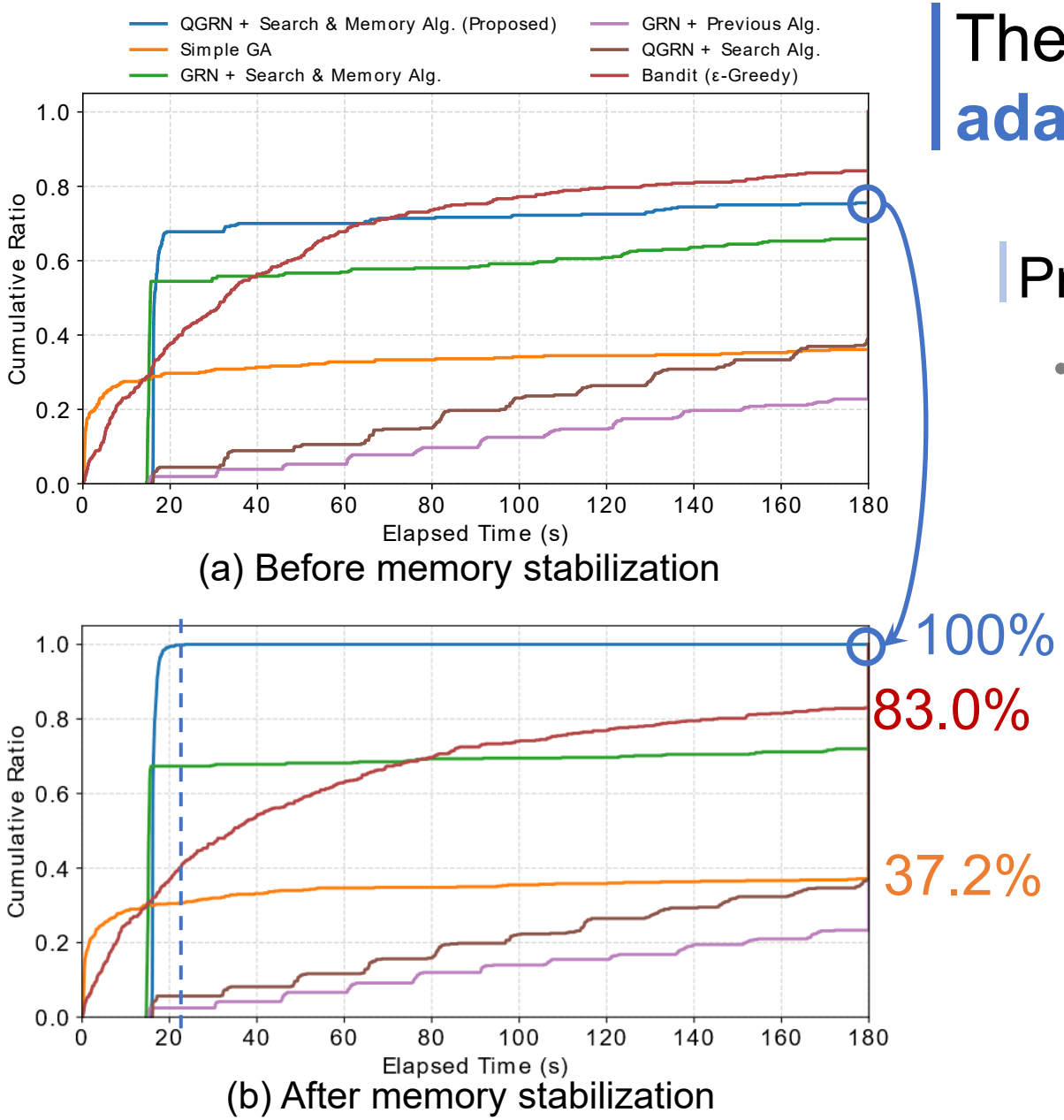
Previous Algorithms

- Persistently low fitness

→ quantum states enhance the memory capability

Transition of fitness values in the memory group: $Fitness = -\sum D_{mem} \log(D_{mem}/D_i)$.

Result 2: Faster Optimal Solution Discovery



The proposed method achieved **adapts multi-timescale conditions.**

Proposed Algorithm (Blue)

- Found the optimal solution within 22 s in all environments after memory stabilization.

QGRN: 16 s per generation

↕ Future quantum hardware will drastically reduce this gap

Simple GA: 12,000 generations within 16 s

Conclusion

Contributions

- We propose a QGRN-based optimization framework for adaptive network control.
- Demonstrating autonomous adaptability to both cyclic environmental fluctuations and structural changes w/o external change detection.

Future Work

- Validating the scalability of the proposed framework in larger environments
- Designing a truly self-evolving architecture inspired by biological systems.
- Evaluating quantum noise effects on real quantum devices.