General Topological Agent: A Unified Paradigm for Cross-Domain Structural Intelligence

Lanhaijian Hongqiao University Tech Network
hongqiao.tech
Shanghai, China

Abstract

A large number of complex decision-making problems in the real world are essentially structural intelligence problems, whose core lies in the analysis, decomposition, planning and optimization of high-dimensional coupled topological structures. Although scenarios such as industrial manufacturing drawing parsing, airline integrated scheduling, integrated driving-parking autonomous driving, chip design and molecule generation are quite different, they share the same structural dilemmas: state space explosion, dense constraints, global-local conflicts, sparse rewards and difficult computational convergence. This paper proposes General Topological Agent (GTA), a unified paradigm that constructs structured state space via topological decomposition, decouples macro planning and micro execution via Hierarchical Reinforcement Learning (HRL), and implements group-level low-computation policy optimization via Topology-GRPO. While maintaining high unity of core algorithmic ideas, this paradigm allows differentiated implementation paths in different scenarios according to domain constraints, truly realizing "same algorithm origin, heterogeneous scenarios, all roads lead to Rome". This paper gives complete algorithm frameworks, formal mathematical definitions and executable pseudo-codes for five typical scenarios: industrial manufacturing, integrated driving-parking autonomous driving, airline integrated scheduling, chip design and molecule generation, demonstrating the theoretical value and engineering potential of topological agent as the next-generation general structural intelligence.

I. UNIFIED MATHEMATICAL FOUNDATION OF GENERAL TOPOLOGICAL AGENT

A. A. Unified Representation of Topological Graph

Any complex system can be represented as a weighted undirected topological graph:

$$A^g(s,a) = Q^g(s,a) - V^g(s)$$

- $V = \{v_1, \ldots, v_n\}$: Nodes, representing basic entities in the domain. - $E = \{e_{ij}\}$: Edges, representing dependencies, constraints, connections or relations. - $W \in \mathbb{R}^{n \times n}$: Weight matrix, representing relation strength, cost, risk or constraint strength.

B. B. Topological Decomposition

Decompose the global graph into a set of non-overlapping subgraphs.

C. C. Hierarchical Reinforcement Learning (HRL)

- High-level policy: $\pi_h(a_h|G)$, global planning, partitioning, task scheduling. - Low-level policy: $\pi_l(a_l|G_s)$, local atomic operations, control, allocation. - Reward discount factor: $\gamma \in (0,1)$.

D. D. Topology-GRPO Optimization Objective

Calculate the advantage function grouped by topological subgraphs:

$$\max_{\pi_h, \pi_l} \mathbb{E}_{\pi_h, \pi_l} \left[\sum_{g=1}^{k} \sum_{t=1}^{T} \gamma^t R^g(s_t, a_t) \right]$$

Maximize the grouped cumulative reward.

II. ALGORITHM FRAMEWORKS AND MATHEMATICAL PSEUDO-CODES FOR FIVE SCENARIOS

A. A. Industrial Manufacturing: 2D/3D Engineering Drawing Parsing and Topology Optimization

Algorithm Framework 1. Construct geometric topological graph $G_M = (V_M, E_M, W_M)$. 2. Decompose into feature subgraphs: contours, holes, slots, faces. 3. High-level agent plans decomposition order. 4. Low-level agent executes edge merging, ring closure, PMI binding. 5. Topology-GRPO optimizes grouped by feature subgraphs.

Mathematical Form and Pseudo-Code

Algorithm 1 2D/3D Engineering Drawing Parsing and Topology Optimization

Require: CAD drawing I
Ensure: Topology optimization result O
$G_M \leftarrow \text{ConstructTopologicalGraph}(I)$
$\mathcal{G}_M \leftarrow \text{TopologicalDecomposition}(G_M)$
$\pi_h \leftarrow \text{HighLevelPolicy}(\mathcal{G}_M)$
for $G_{M_i} \in \mathcal{G}_M$ **do**
  $\pi_l^{(i)} \leftarrow \text{LowLevelPolicy}(G_{M_i})$
  $R_i \leftarrow \text{TopologyGRPO}(\pi_h, \pi_l^{(i)}, G_{M_i})$
end for
$O \leftarrow \text{AggregateResults}(R_1, \ldots, R_k)$

B. B. Integrated Driving-Parking Autonomous Driving: Perception-Decision-Control Topological Intelligence

Algorithm Framework 1. Construct dynamic environment topology $G_D = (V_D, E_D, W_D)$. 2. Decompose into driving, parking and obstacle subgraphs. 3. High-level decision: driving/parking/switch mode. 4. Low-level outputs steering, speed, parking trajectory. 5. Topology-GRPO optimizes grouped by local road topology.

Mathematical Form and Pseudo-Code

Algorithm 2 Integrated Driving-Parking Autonomous Driving

Require: Sensor data S
Ensure: Control command C
$G_D \leftarrow \text{ConstructDynamicGraph}(S)$
$\mathcal{G}_D \leftarrow \{G_{\text{drive}}, G_{\text{park}}, G_{\text{obs}}\}$
$m \leftarrow \pi_h(G_D)$ {driving/parking/switch} m drive
$C \leftarrow \pi_l^{\text{drive}}(G_{\text{drive}})$ park
$C \leftarrow \pi_l^{\text{park}}(G_{\text{park}})$ switch
$C \leftarrow \text{TransitionControl}(G_D)$
$R \leftarrow \text{TopologyGRPO}(\pi_h, \pi_l, G_D)$

C. C. Airline Integrated Scheduling: Joint Optimization of Fleet-Crew-Flight-Maintenance

Algorithm Framework 1. Construct global resource topology $G_A = (V_A, E_A, W_A)$. 2. Decompose into flight, fleet, crew, maintenance subgraphs. 3. High-level agent performs global resource scheduling and timing planning. 4. Low-level executes pairing, assignment, maintenance embedding. 5. Topology-GRPO optimizes cost, punctuality, robustness grouped by subgraphs.

Mathematical Form and Pseudo-Code

Algorithm 3 Airline Integrated Scheduling

Require: Flight plan F, resource pool R
Ensure: Scheduling scheme Ω
$G_A \leftarrow \text{ConstructResourceGraph}(F, R)$
$\mathcal{G}_A \leftarrow \{G_{\text{flight}}, G_{\text{fleet}}, G_{\text{crew}}, G_{\text{maint}}\}$
$\pi_h \leftarrow \text{GlobalScheduler}(G_A)$
for $G_{A_i} \in \mathcal{G}_A$ **do**
  $\pi_l^{(i)} \leftarrow \text{LocalOptimizer}(G_{A_i})$
  $\Omega_i \leftarrow \text{ApplyPolicy}(\pi_l^{(i)}, G_{A_i})$
end for
$\Omega \leftarrow \text{IntegrateSolutions}(\Omega_1, \ldots, \Omega_4)$
$R \leftarrow \text{TopologyGRPO}(\pi_h, \{\pi_l^{(i)}\}, G_A)$

D. D. Chip Design: Floorplan, Placement and Routing Topology Optimization

Algorithm Framework 1. Construct chip topological graph $G_C = (V_C, E_C, W_C)$. 2. Decompose into functional modules, clock tree, routing channels, I/O cells. 3. High-level agent plans global floorplan, module partitioning, timing constraints. 4. Low-level agent executes placement, routing, timing optimization, conflict resolution. 5. Topology-GRPO optimizes timing, power, area grouped by functional subgraphs.

Mathematical Form and Pseudo-Code

Algorithm 4 Chip Design: Floorplan, Placement and Routing

Require: Netlist N, constraints C
Ensure: Physical layout P

$G_C \leftarrow \text{ConstructChipGraph}(N, C)$
$\mathcal{G}_C \leftarrow \text{FunctionalDecomposition}(G_C)$
$\pi_h \leftarrow \text{FloorplanningPolicy}(\mathcal{G}_C)$
for $G_{C_i} \in \mathcal{G}_C$ **do**
  $\pi_l^{(i)} \leftarrow \text{PlacementRoutingPolicy}(G_{C_i})$
  $P_i \leftarrow \text{ExecutePlacementRouting}(\pi_l^{(i)}, G_{C_i})$
end for
$P \leftarrow \text{MergeLayouts}(P_1, \ldots, P_k)$
$R \leftarrow \text{TopologyGRPO}(\pi_h, \{\pi_l^{(i)}\}, G_C)$

E. E. Molecule Generation: Drug / Functional Molecule Topology Design

Algorithm Framework 1. Construct molecular topological graph $G_{Mo} = (V_{Mo}, E_{Mo}, W_{Mo})$. 2. Decompose into skeleton, functional groups, pharmacophores, chemical bonds. 3. High-level agent designs molecular skeleton and pharmacophore topology. 4. Low-level agent executes atom connection, bond adjustment, fragment assembly, conformation optimization. 5. Topology-GRPO optimizes activity, stability, synthesizability grouped by substructures.

Mathematical Form and Pseudo-Code

Algorithm 5 Molecule Generation: Drug / Functional Molecule Topology Design

Require: Design target T
Ensure: Molecular structure M

$G_{Mo} \leftarrow \text{ConstructMolecularGraph}(T)$
$\mathcal{G}_{Mo} \leftarrow \text{ChemicalDecomposition}(G_{Mo})$
$\pi_h \leftarrow \text{ScaffoldDesignPolicy}(\mathcal{G}_{Mo})$
for $G_{Mo_i} \in \mathcal{G}_{Mo}$ **do**
  $\pi_l^{(i)} \leftarrow \text{FragmentAssemblyPolicy}(G_{Mo_i})$
  $M_i \leftarrow \text{AssembleFragments}(\pi_l^{(i)}, G_{Mo_i})$
end for
$M \leftarrow \text{CombineFragments}(M_1, \ldots, M_k)$
$R \leftarrow \text{TopologyGRPO}(\pi_h, \{\pi_l^{(i)}\}, G_{Mo})$

III. UNIFIED PARADIGM AND CROSS-DOMAIN SUMMARY

The General Topological agent maintains a fully consistent algorithmic core across five scenarios:

1) **Topological modeling**: Unify domain problems as graph structures.
2) **Topological decomposition**: Split complex systems into decoupled substructures.
3) **Hierarchical decision-making**: High-level for global strategy, low-level for fine-grained operations.
4) **Topology-GRPO**: Grouped optimization by subgraphs, greatly reducing computation.

Scenario constraints determine implementation differences:

- **Industrial manufacturing**: Geometric validity, PMI semantic binding.
- **Autonomous driving**: Real-time performance, safety constraints, dynamic topology.
- **Airline scheduling**: Timing constraints, multi-resource coupling, robustness.
- **Chip design**: Timing closure, routing conflict-free, PPA optimization.
- **Molecule generation**: Chemical validity, bioactivity, synthesizability.

This fully embodies: algorithms share the same origin and structure, while domain paths differ; all roads lead to Rome, but each one walks on its own track.

Topological Agent is not only an algorithm, but also a unified worldview for cross-domain structural intelligence. It can be seamlessly extended to more complex structural decision scenarios such as intelligent manufacturing, smart cities, network architecture and system design, providing a new paradigm for the landing of general artificial intelligence.