GiPH: Generalizable Placement Learning for Adaptive Heterogeneous Computing

Yi Hu, Chaoran Zhang, Edward Andert, Harshul Singh, Aviral Shrivastava, James Laudon, Yanqi Zhou, Bob Iannucci, Carlee Joe-Wong
Contact: yihu@andrew.cmu.edu

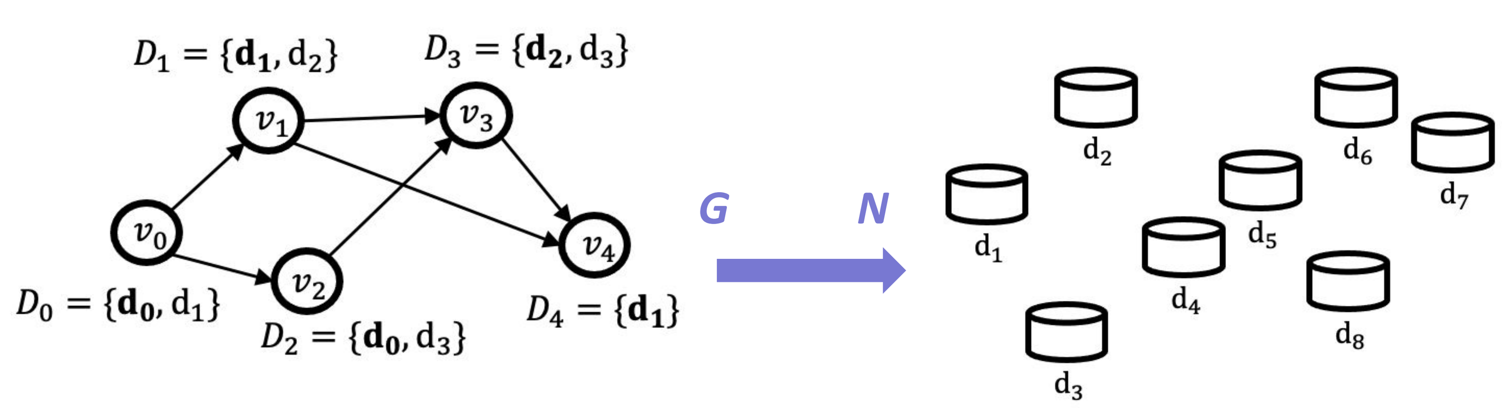
Introduction

In heterogeneous computing systems, careful choice of which parts of the application to run on which device can significantly affect ***latency***, e.g., compute-intensive tasks should be run on devices with more computation resources. The key placement challenges are:

- ***Diverse*** compute/communication capabilities
- ***Volatile:*** devices can become unavailable/new devices enter the system
- ***Heterogeneous*** functions or types (CPUs/GPUs/sensors)

We propose GiPH, a reinforcement learning-based approach to learning placement policies that can adapt to dynamic device networks.

Placement in Heterogeneous Computing



Task graph G *(directed acyclic graph)*

- Defines a distributed application
- Nodes V: computation/sensing tasks of different workloads/requirements
- Edges E: communication and inter-task dependency

Target computing network N

- Defines a cluster of interconnected devices
- Devices D: the set of devices with different compute/communication capabilities
- Each task v_i can only be mapped to a subset of devices $D_i \subseteq D$

Placement problem

- A mapping from the set of tasks to the set of devices
  $$\mathcal{M}^{G\to N}: V \to D$$
- Objective $\min \rho(\mathcal{M}|G,N) \quad s.t. \mathcal{M}(v_i) \in D_i$

Makespan minimization *(critical for time-sensitive applications)*

- Makespan: the time duration from the start of the first task's execution to the end of the last task's execution (i.e., completion time)
- Equal to the total communication and computation cost along the ***critical path***

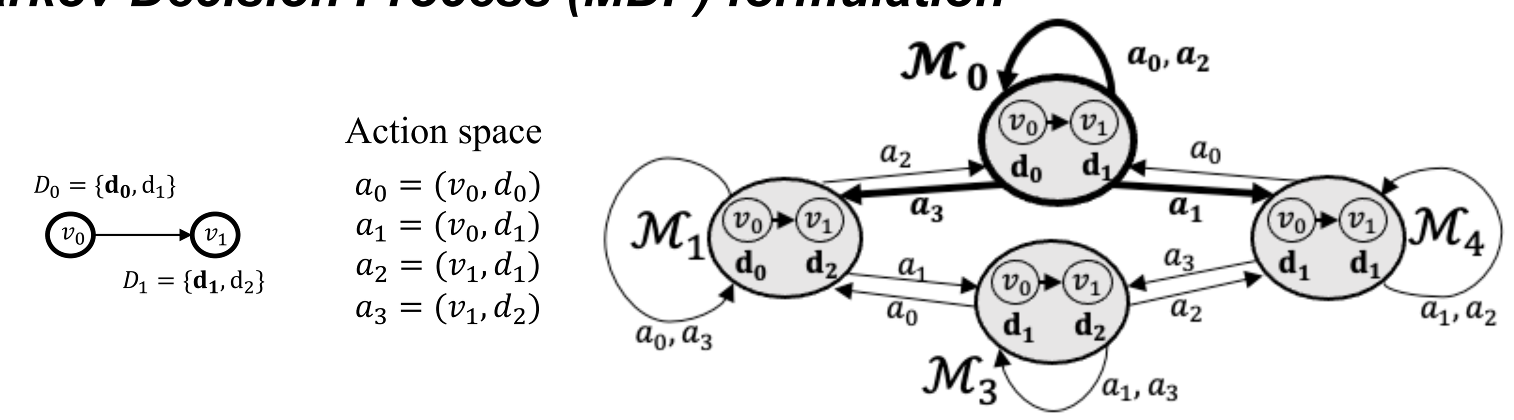
$$\min_{\mathcal{M}} \rho(\mathcal{M}|G,N) = \min_{\mathcal{M}} \max_{p\in P(G)} \left(\sum_{i\in p} c_i^{comp}(\mathcal{M}) + \sum_{(i,j)\in p} c_{ij}^{comm}(\mathcal{M}) \right)$$

GiPH

We formulate the placement problem as a *search problem*, where *incremental changes* are made to the current placement

Current placement (current state) → relocate one of the tasks (action) → transition to the updated placement (next state)

Markov Decision Process (MDP) formulation

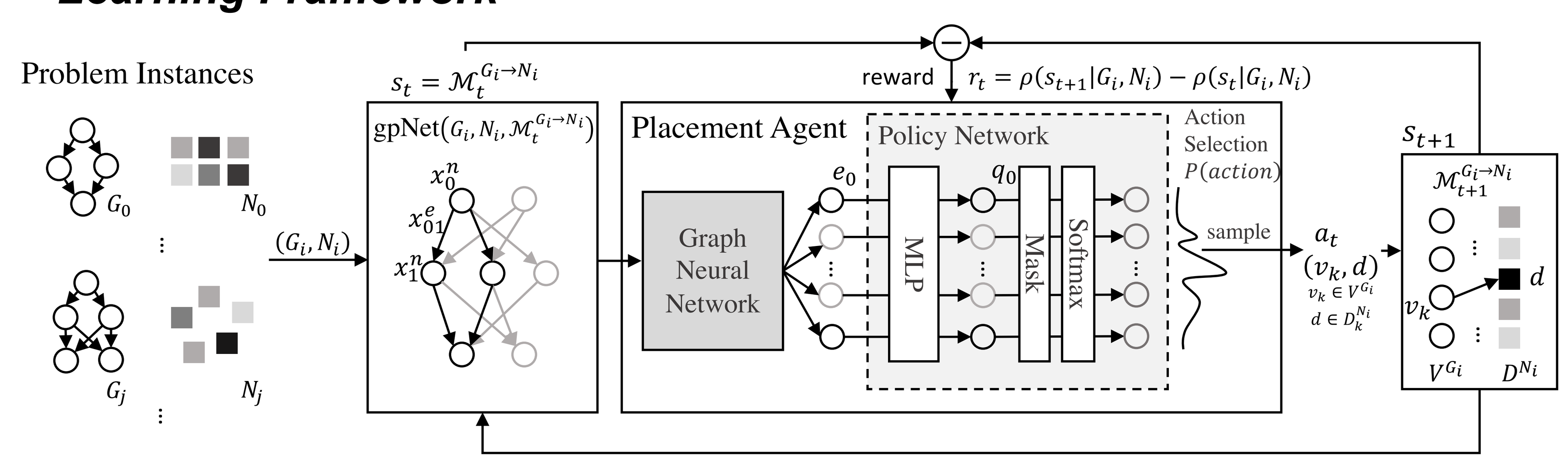


State Space: the set of all feasible placement
Action Space: the set of all task and device pair that satisfies constraints

- $a_t = (v_i, d_j)$: place v_i on d_j

Reward: the performance improvement $r_t = \rho(s_{t+1}|G,N) - \rho(s_t|G,N)$

Learning Framework



gpNet: a graph representation to efficiently encode information

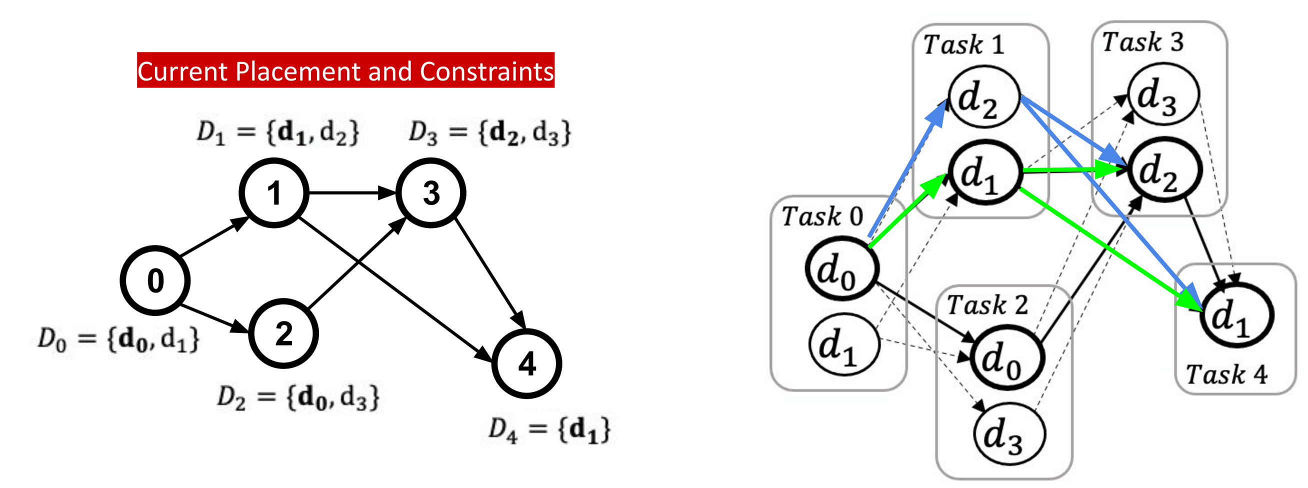
- Each node corresponds to one action
- Local graph structure of $a_t = (v_i, d_j)$ corresponds to v_i being re-placed to d_j

Graph Neural Network: calculates embedding for each action

- Embed the placement information as a set of vectors
- Message passing: $e_u = h_2(\sum_{v\in\xi(u)} h_1([e_v || x_{uv}^e])) + x_u^n$

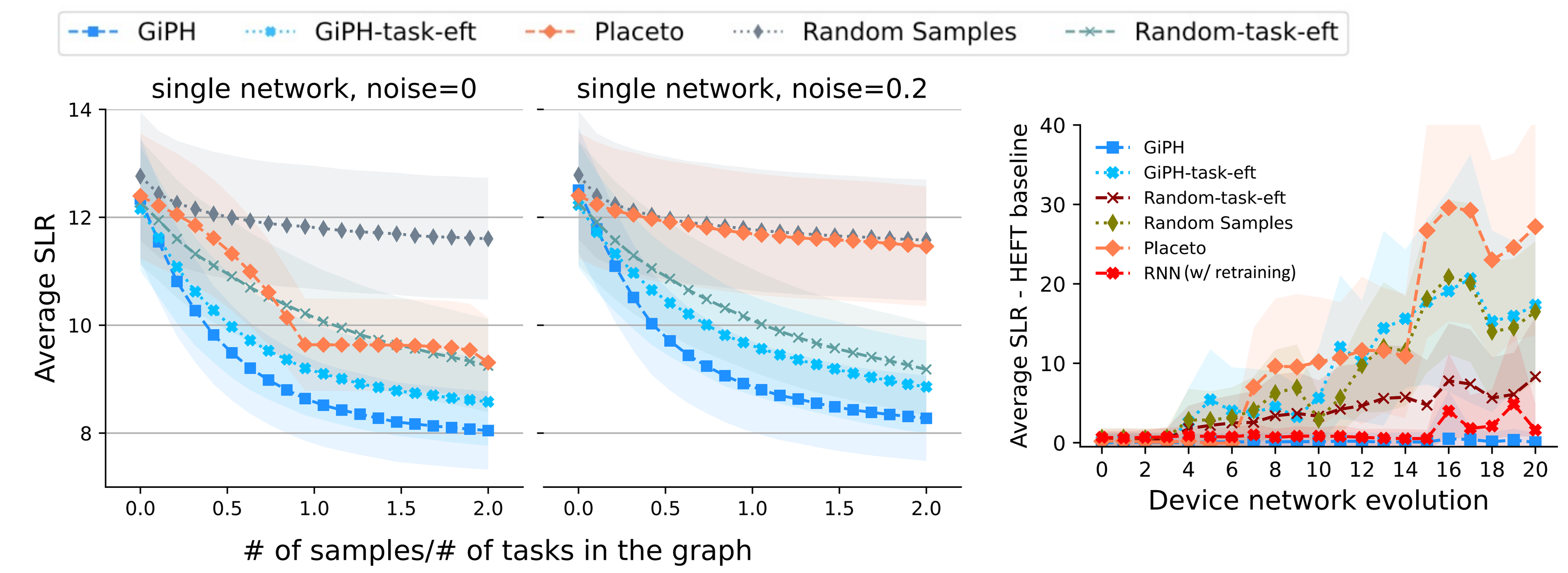
Policy Network: decides an action (i.e., relocating a task) to take

- Learnable score function: $q_a = g(e_a)$
- Softmax action selection: $\pi(a|s) = q_a / \sum_{b\in A_t} q_b$



gpNet example

Evaluation

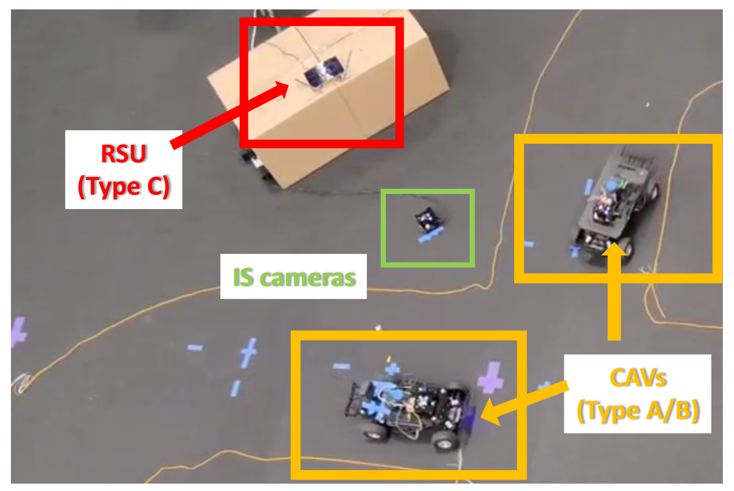


Efficiency: GiPH finds better placement within fewer steps
Adaptivity: As the device network changes, GiPH maintains stable performance

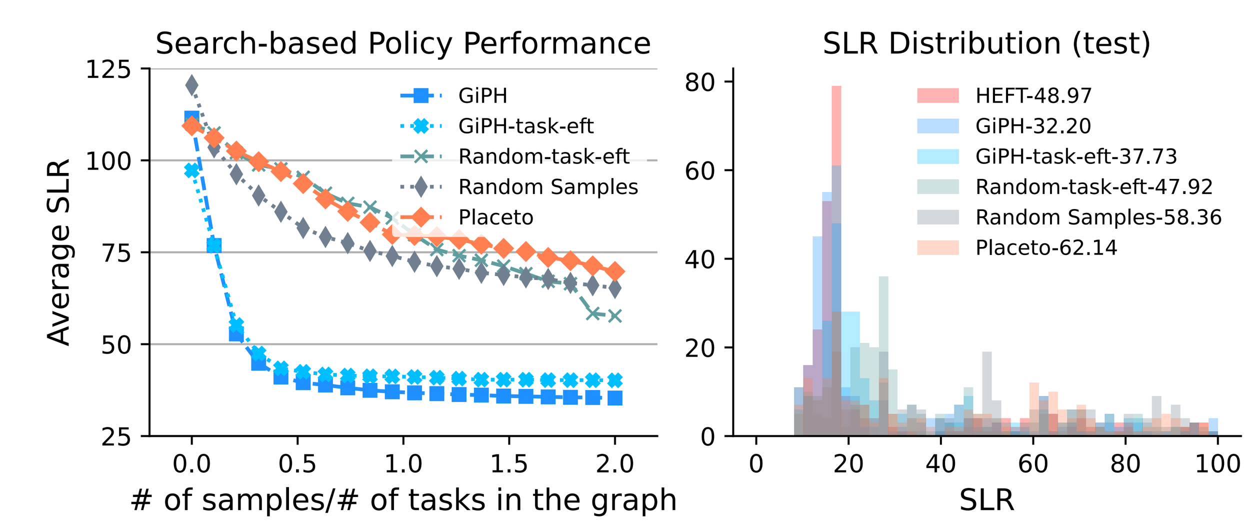
- *GiPH vs. RNN-Placer from HDP [1]*: GiPH adapts to new device clusters, while the RNN-placer needs to be retrained
- *GiPH vs. Placeto [2]*: GiPH identifies critical tasks and adjust their placements more frequently during the search, while Placeto updates each task placement equally for exactly once

Case Study: Cooperative Sensor fusion

- Autonomous driving with roadside units (RSUs), infrastructure (IS) cameras, and CAVs
- Relocation overhead (data migration/initialization) measured in real-world deployment
- Realistic application trace simulated using Simulation of Urban MObility (SUMO)



Real-world deployment. Type A: Jetson Nano. Type B: Jetson TX2. Type C: Core i7 7700K with GTX 1080.



References

[1] A. Mirhoseini, A. Goldie et al (2018). "A Hierarchical Model for Device Placement." International Conference on Learning Representations.

[2] R. Addanki, S. B. Venkatakrishnan et al. (2019). "Placeto: learning generalizable device placement algorithms for distributed machine learning." Proceedings of the 33rd International Conference on Neural Information Processing Systems.

This work was supported by funding from NSF grants CPS-1645578, CPS-1646235, CNS-2106891, and USDOT UTC grant 69A3551747111.