MLSys 2023

Carnegie Mellon University

GiPH: Generalizable Placement Learning for Adaptive Heterogeneous Computing

Yi Hu[1], Chaoran Zhang[1], Edward Andert[2], Hashul Singh[1], Aviral Shrivastava[2], James Laudon[3], Yanqi Zhou[3], Bob Iannucci[3], Carlee Joe-Wong[1]

[1]Department of Electrical and Computer Engineering, CMU
[2]School of Computing and Augmented Intelligence, ASU
[3]Google

Overview

- **Placement in Heterogeneous Computing**
  - Motivation
  - Problem Formulation
- **Related Work**
- **GiPH**
- **Evaluation**
- **Conclusion**

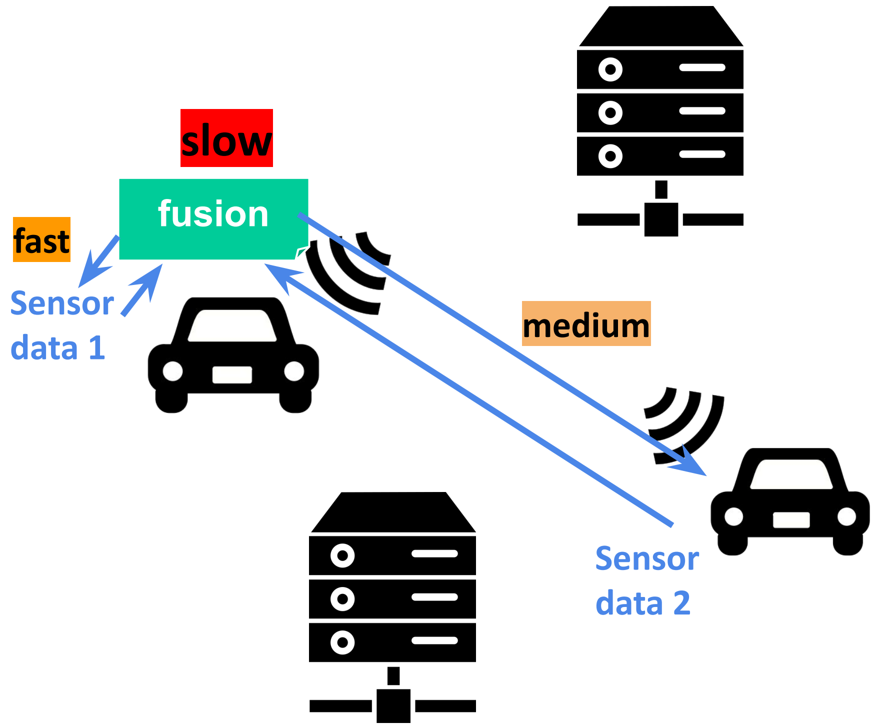
Motivation

- **Highly distributed, hundreds of nodes: *LATENCY***
  - Time-sensitive data processing
  - Precise timing requirements
  - Eg., light-free traffic control

Source: 'Rush Hour' by Black Sheep Films

Placement is the KEY

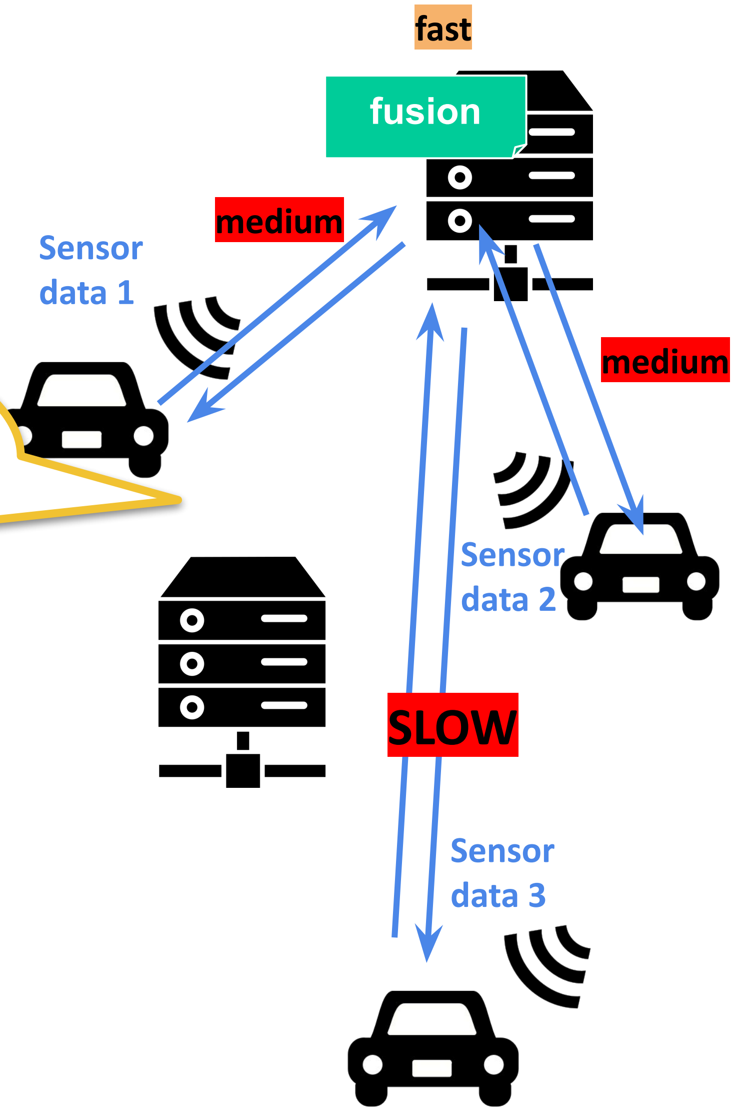
Example: sensor fusion



Placement is the KEY

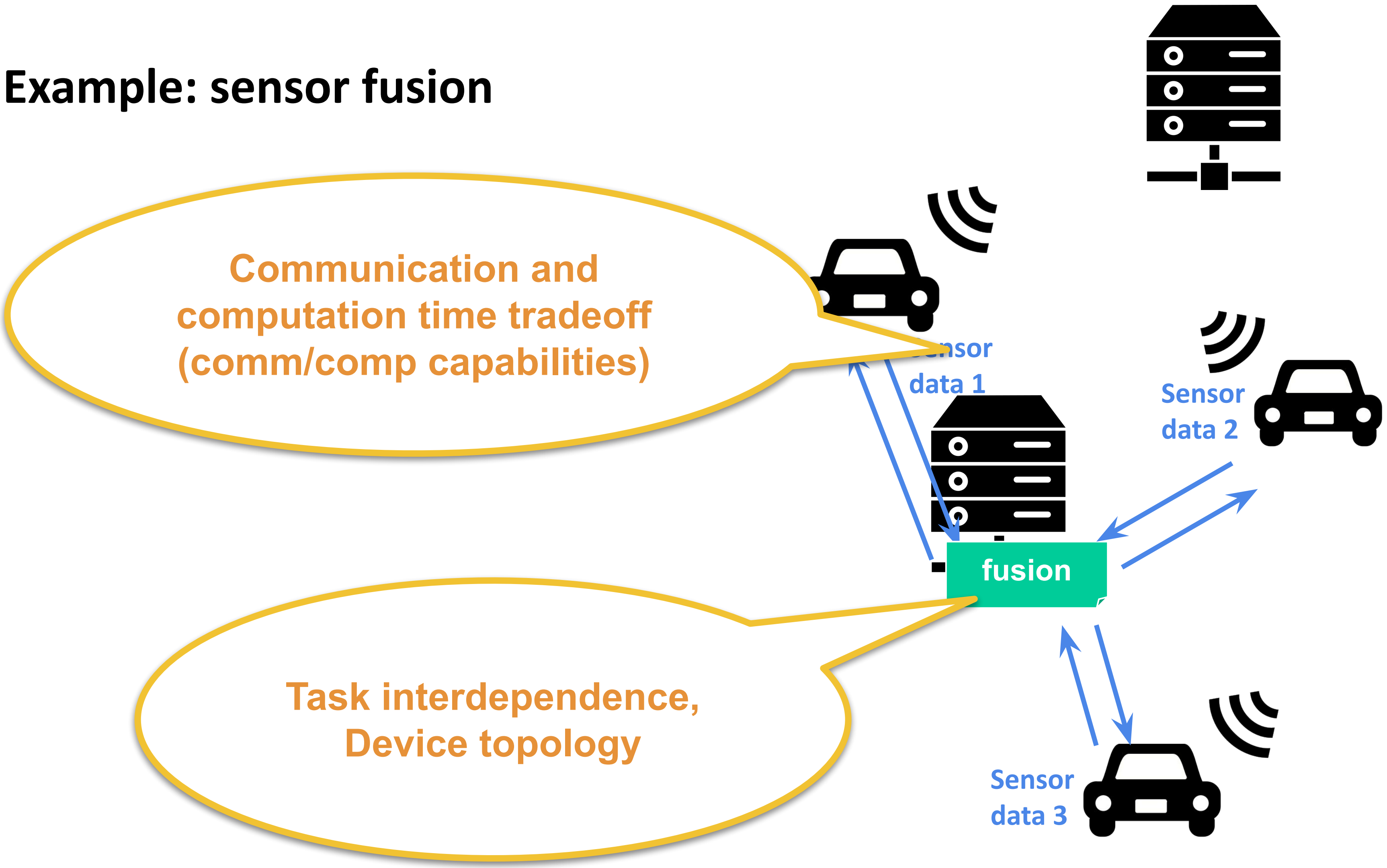
Example: sensor fusion

Communication and computation time tradeoff (comm/comp capabilities)



Placement is the KEY

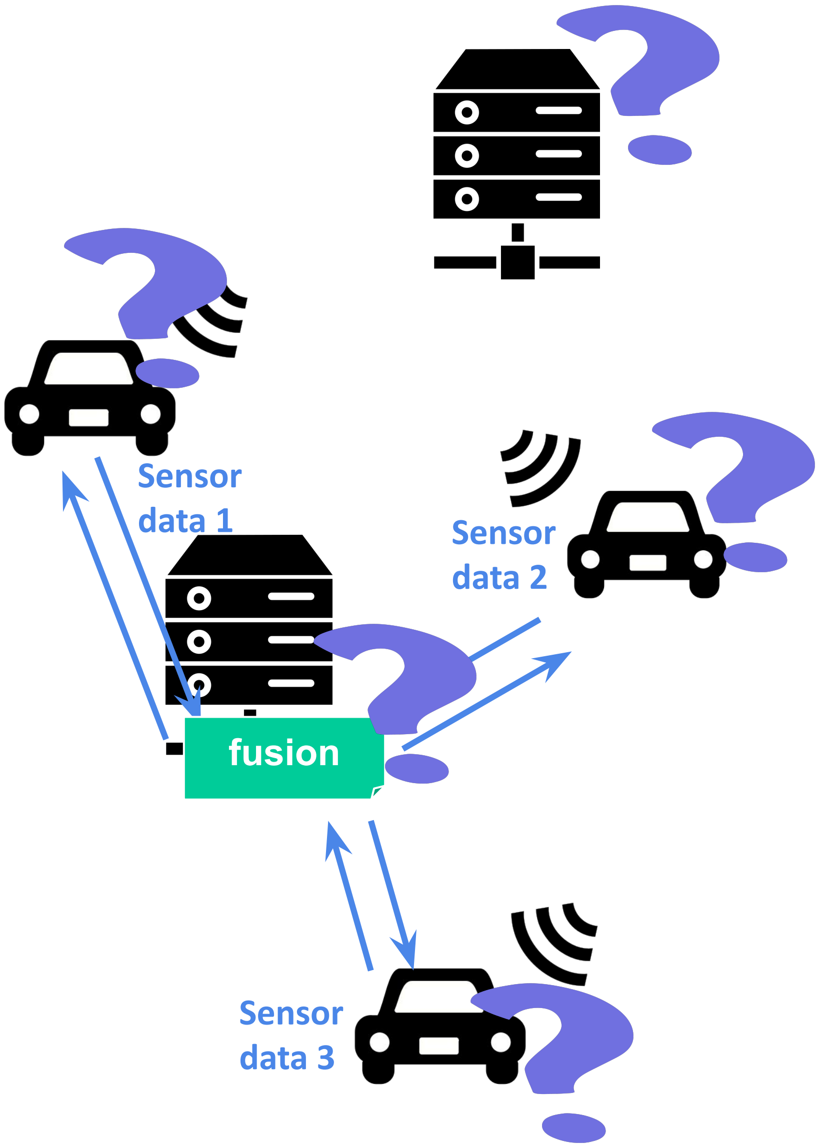
Example: sensor fusion



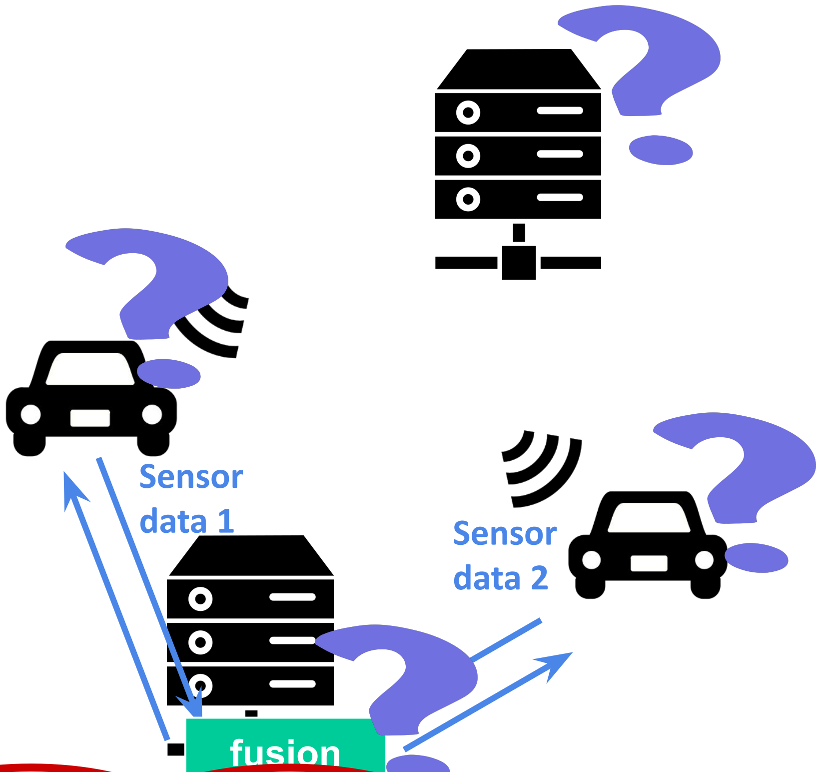
Placement is the KEY

Challenges:

- Devices are *heterogeneous*
  - Different types:
    - CPUs/GPUs
    - PCs/Servers/UEs
  - Various compute/communication capabilities - *tradeoff*
  - Functionalities
- Devices can be *volatile*
  - Some device becomes unavailable
  - New device enters the system



Placement is the KEY

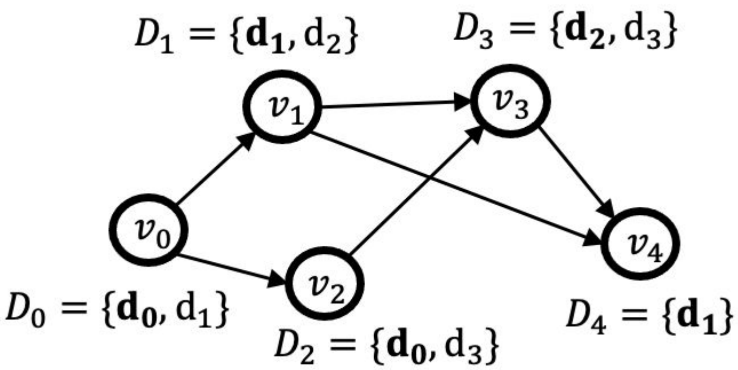


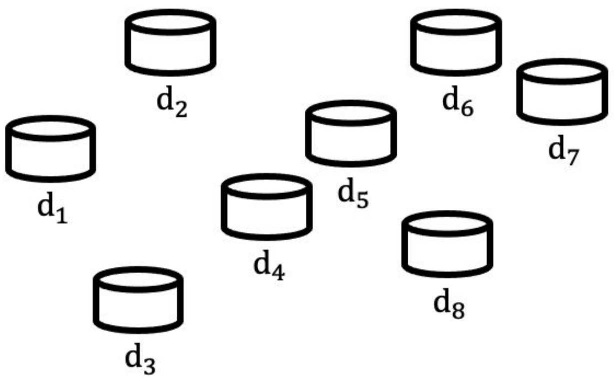
Challenges:

- Devices are <u>*heterogeneous*</u>
  - Different types:
    - CPUs/GPUs
    - PCs/Servers/UEs
  - Various compute/communication capabilities - *tradeoff*
  - Functionalities
- Devices can be <u>*volatile*</u>
  - Some device becomes unavailable
  - Ne

Require a solution that can **scale to different number of devices** and can **efficiently encode information** as the device set changes.

Placement Problem





A compute application *G* (DAG)

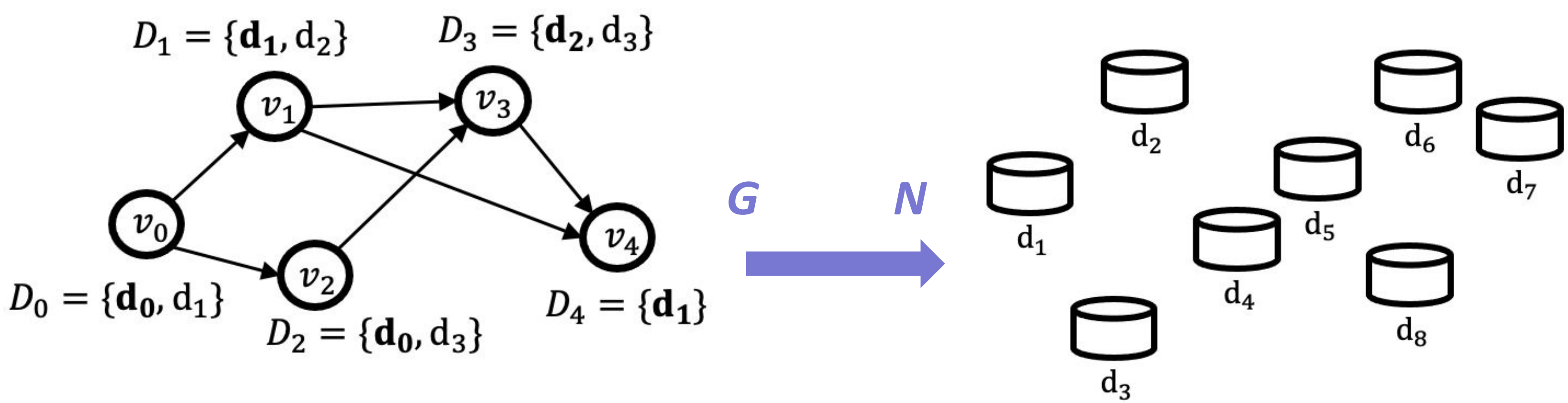
The set of tasks *V*

with placement constraints $D_i \subseteq D$

A target computing network *N*

The set of devices *D*

Placement Problem



A compute application *G* (DAG)

The set of tasks *V*

with placement constraints $D_i \subseteq D$

A target computing network *N*

The set of devices *D*

Placement $\mathcal{M}^{G \to N} : V \to D$

Objective $\min \rho(\mathcal{M} | G, N) \quad \text{s.t } \mathcal{M}(v_i) \in D_i$

Placement: Makespan Minimization

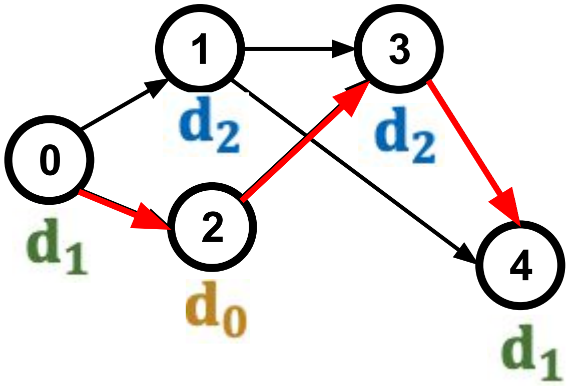
For time-sensitive applications, it is important to minimize the ***completion time***, i.e., makespan

- The time duration from the start of the first task's execution to the end of the last task's execution

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

- The total cost along the critical path
- Depends on the placement of all tasks
- NP-hard

Hard to place the whole graph all at once!



Related Work

- Scheduling Heuristics in Heterogeneous Computing
- RL-based Device Placement for Neural Network Training

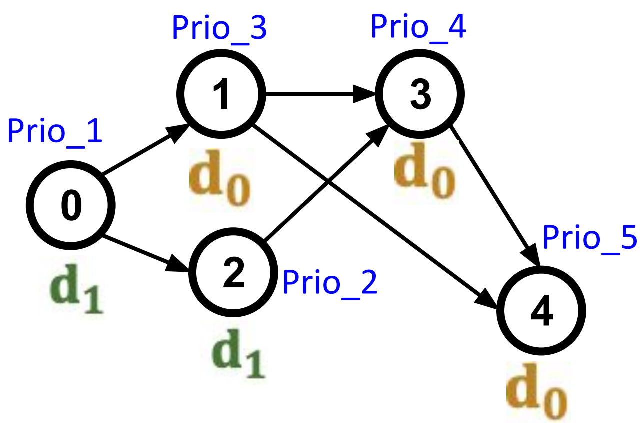
Related Work: Scheduling Heuristics

- **Rely on simple strategies and hand-crafted features**
- **E.g., Heterogeneous Earliest Finish Time (*HEFT*)[1]**
  - Give each task a priority that maintains the topological ordering of the tasks
  - Starting with the highest priority, place each task to a device that will result in the *earliest finish time* (EFT) of that task

[1] H. Topcuoglu, S. Hariri and Min-You Wu, "Performance-effective and low-complexity task scheduling for heterogeneous computing," in *IEEE Transactions on Parallel and Distributed Systems*, vol. 13, no. 3, pp. 260-274, March 2002, doi: 10.1109/71.993206.

Related Work: Scheduling Heuristics

- **Rely on simple strategies and hand-crafted features**
- **E.g., Heterogeneous Earliest Finish Time (*HEFT*)[1]**
  - Give each task a priority that maintains the topological ordering of the tasks
  - Starting with the highest priority, place each task to a device that will result in the *earliest finish time* (EFT) of that task



[1] H. Topcuoglu, S. Hariri and Min-You Wu, "Performance-effective and low-complexity task scheduling for heterogeneous computing," in *IEEE Transactions on Parallel and Distributed Systems*, vol. 13, no. 3, pp. 260-274, March 2002, doi: 10.1109/71.993206.

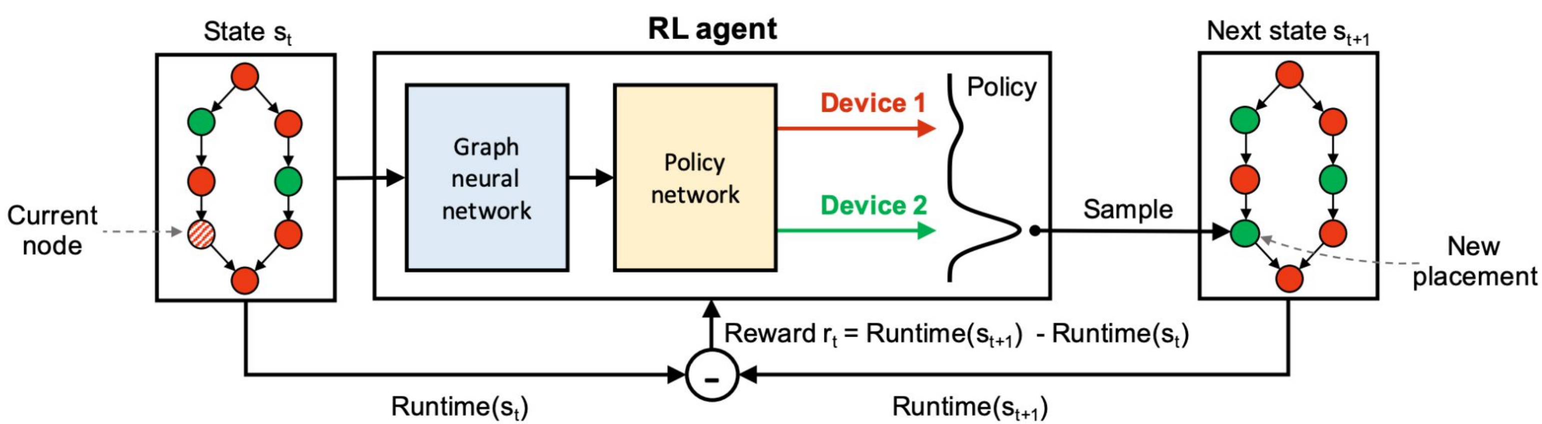
Related Work: RL-based Device Placement

- **Predict a placement for each task**
- **Hierarchical model for device placement (*HDP*)[2]**
  - An RL policy is trained for each graph
  - An RNN-based placer: encoder/decoder pair to predict one device for each node in the order of the inputs
  - <u>*Does not generalize to new neural networks/device clusters*</u>

[2] Mirhoseini, Azalia et al. "A Hierarchical Model for Device Placement." *International Conference on Learning Representations* (2018).
[3] Ravichandra Addanki, Shaileshh Bojja Venkatakrishnan, Shreyan Gupta, Hongzi Mao, and Mohammad Alizadeh. 2019. Placeto: learning generalizable device placement algorithms for distributed machine learning. Proceedings of the 33rd International Conference on Neural Information Processing Systems. Curran Associates Inc., Red Hook, NY, USA, Article 358, 3981–3991.

Related Work: RL-based Device Placement

- **Predict a placement for each task**
- **Hierarchical model for device placement (*HDP*)[2]**
  - An RL policy is trained for each graph
  - An RNN-based placer: encoder/decoder pair to predict one device for each node in the order of the inputs
  - *Does not generalize to new neural networks/device clusters*
- ***Placeto*[3]**
  - A GNN is used to embed graph-level features
  - *Does not generalize to new device clusters*



[2] Mirhoseini, Azalia et al. "A Hierarchical Model for Device Placement." *International Conference on Learning Representations* (2018).
[3] Ravichandra Addanki, Shaileshh Bojja Venkatakrishnan, Shreyan Gupta, Hongzi Mao, and Mohammad Alizadeh. 2019. Placeto: learning generalizable device placement algorithms for distributed machine learning. Proceedings of the 33rd International Conference on Neural Information Processing Systems. Curran Associates Inc., Red Hook, NY, USA, Article 358, 3981–3991.

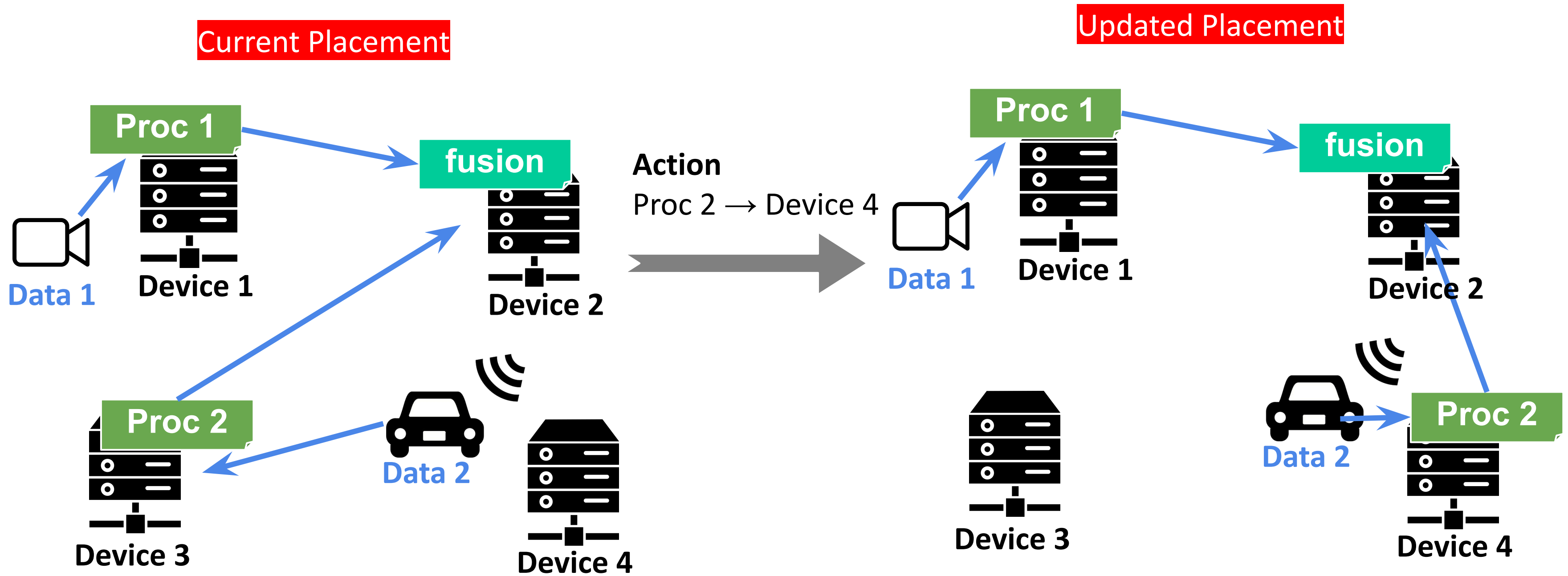
GiPH

- Fully generalizable placement learning
- Adaptive to network changes

MDP Formalism

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

- Current placement→ **take an action (update the current placement) → transition to a new state → reward (improvement)**

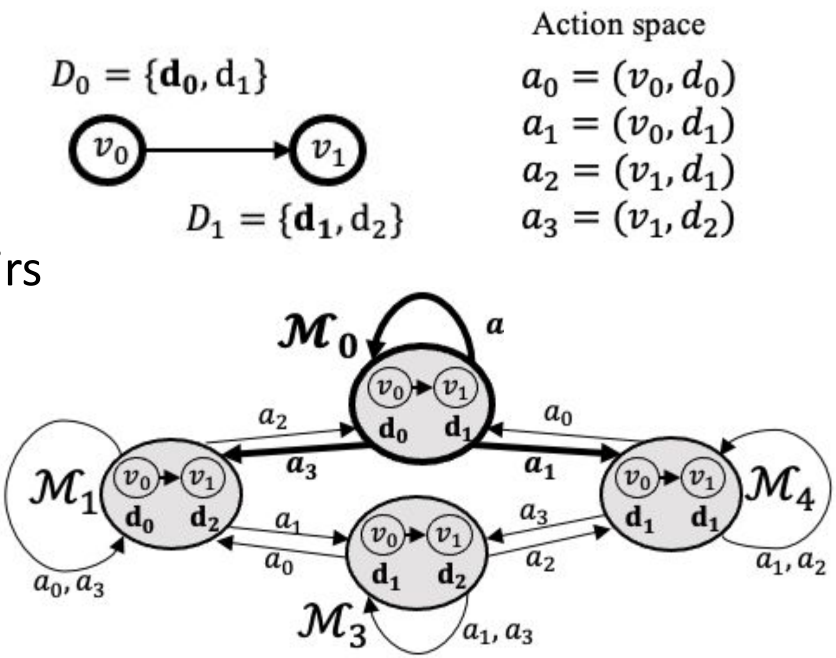


MDP Formalism

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

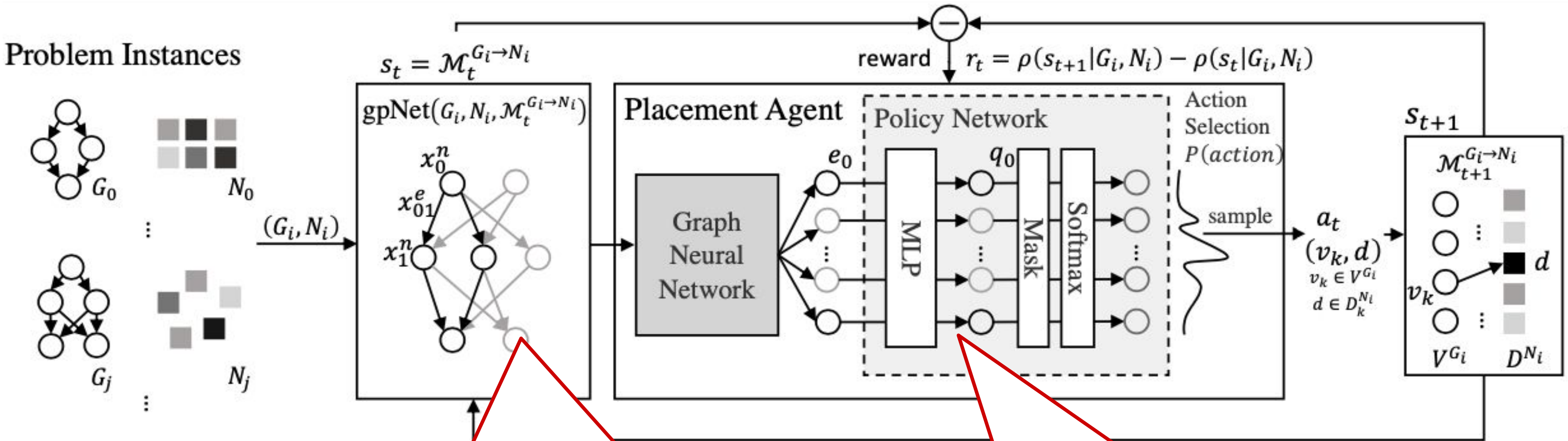
- **State space**
  - set of all feasible placements
- **Action space**
  - set of feasible task and device pairs
  - $a_t = (v_i, d_j)$ place v_i on d_j
- **Reward**
  - The performance improvement

$$r_t = \rho(s_{t+1}|G, N) - \rho(s_t|G, N)$$



GiPH: Framework

GiPH: Generalizable Placement with the ability to adapt to dynamic Heterogeneous networks



gpNet: a graph representation

- Encode information of an arbitrary task graph and network pair
- Capture all task- and device-related features
- Has a *local graph structure* corresponding to each possible task relocation

Placement Agent: GNN + policy network

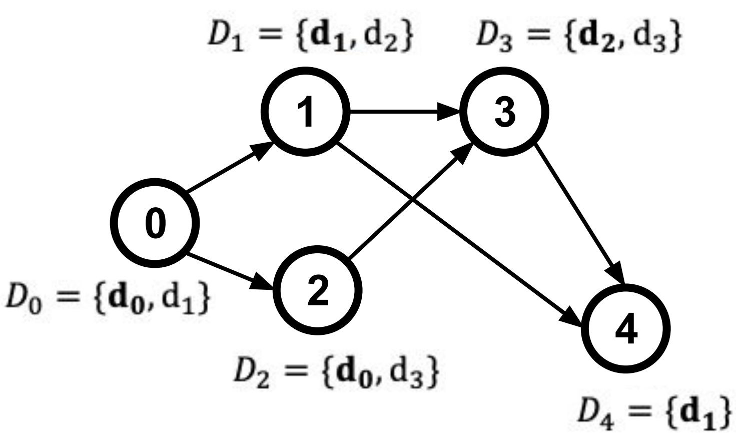
- Take a gpNet as input
- *GNN*: calculate an embedding for each action
- *Policy network*: decides an action (i.e., relocating a task) to take

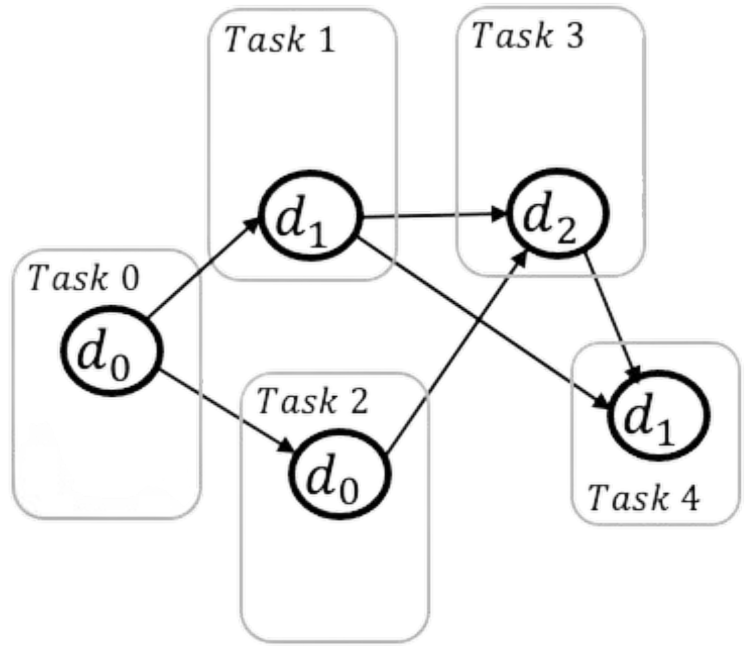
gpNet Representation

An efficient graph representation to encode information

- Each node corresponds to one action
- Local graph structure corresponds to an alternative task placement

Current Placement and Constraints



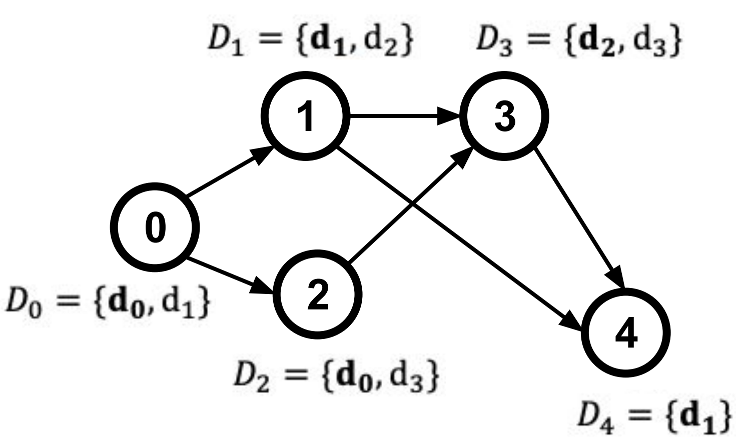


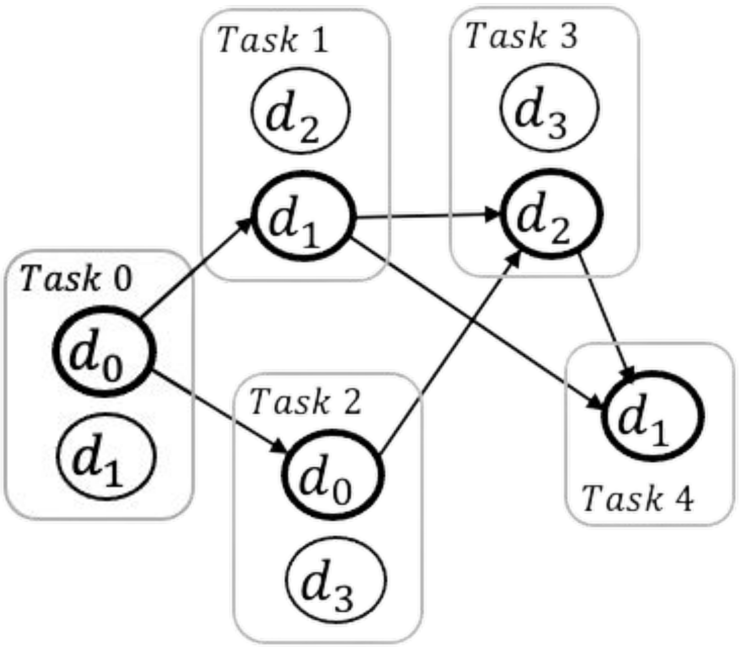
gpNet Representation

An efficient graph representation to encode information

- Each node corresponds to one action
- Local graph structure corresponds to an alternative task placement

Current Placement and Constraints



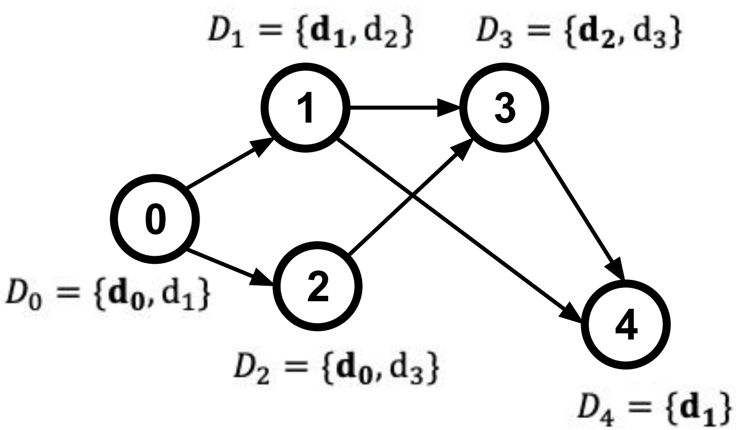


gpNet Representation

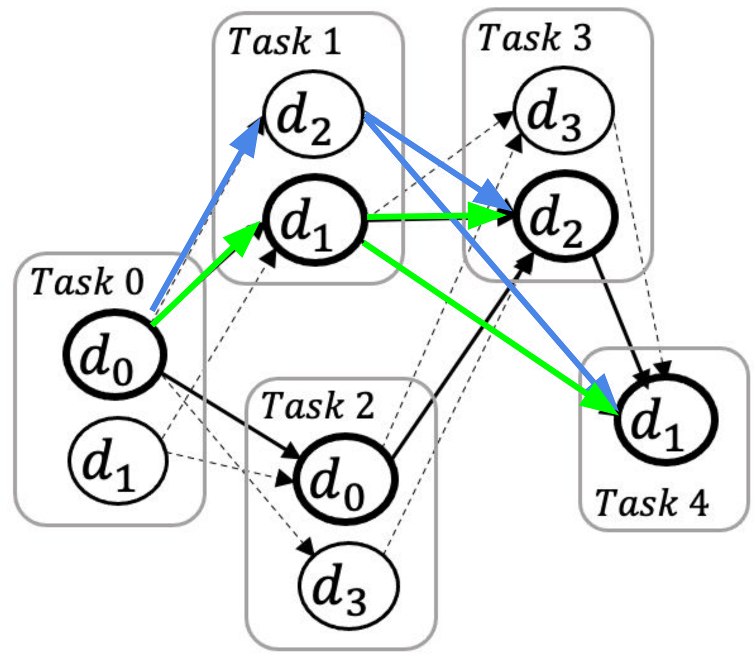
An efficient graph representation to encode information

- Each node corresponds to one action
- Local graph structure corresponds to an alternative task placement

Current Placement and Constraints

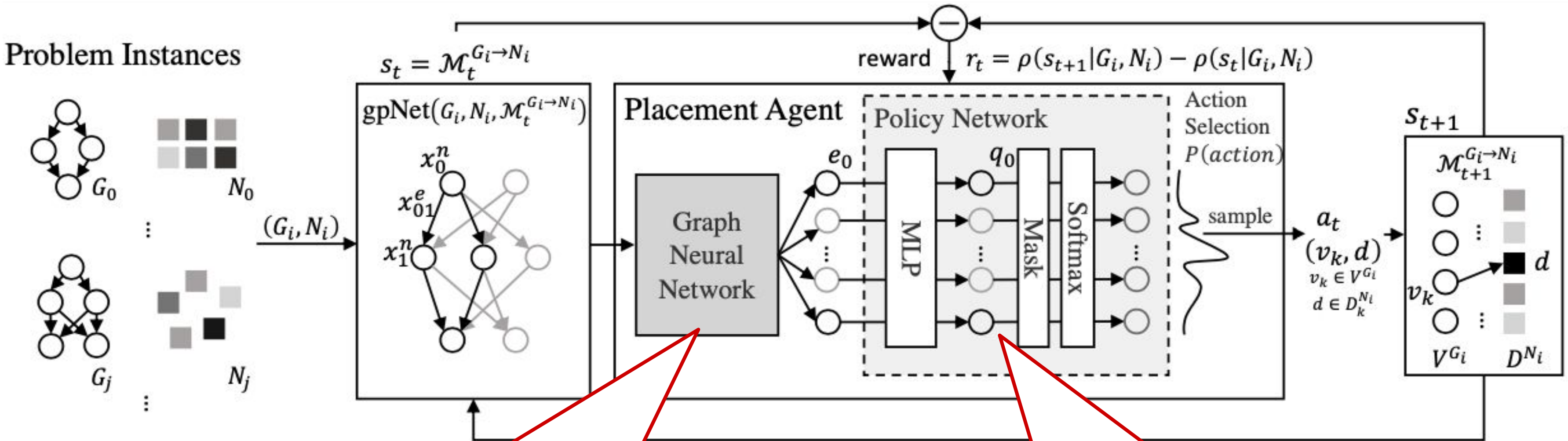


gpNet



GiPH: Neural Network Design

Scalable placement policy: GNN + RL policy network



GNN: takes a gpNet as input and embeds the placement information as a set of vectors

$$e_u = h_2\left(\sum_{v \in \xi(u)} h_1\left([e_v \parallel x_{vu}^e]\right)\right) + x_u^n$$

RL policy network: decides the action of re-placing one of the task (placement update step)

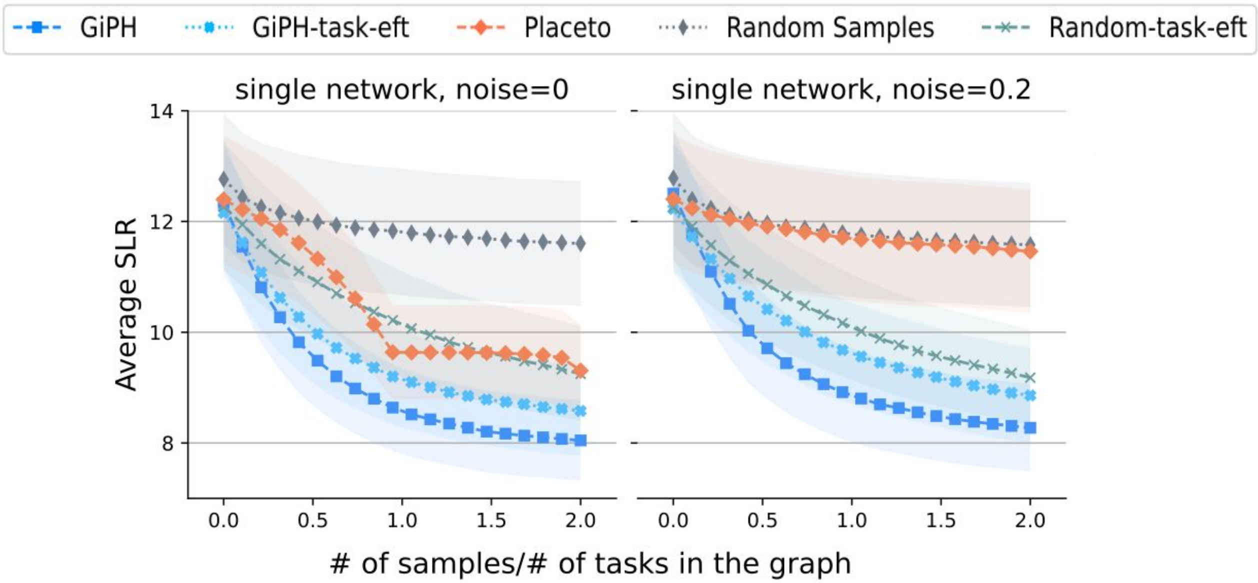
- Score function $q_a = g(e_a)$
- Softmax action selection

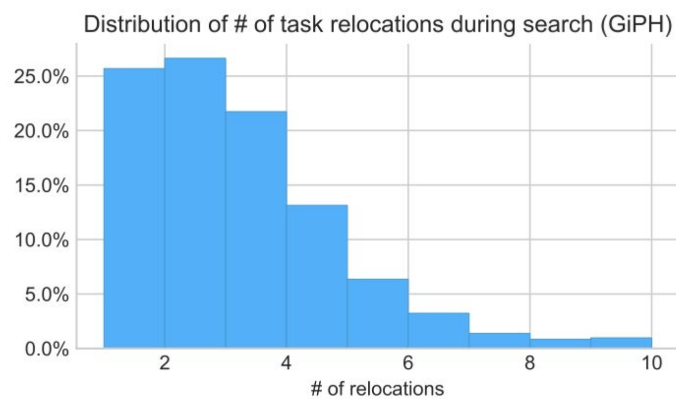
$$\pi(a|s) = \frac{q_a}{\sum_{b \in A} q_b}$$

Evaluation

- Performance: Schedule Length Ratio (normalized makespan) minimization
- Evaluation is done on graphs not in the training dataset
- Case Study: Cooperative Sensor Fusion

Search Efficiency: GiPH vs. Placeto



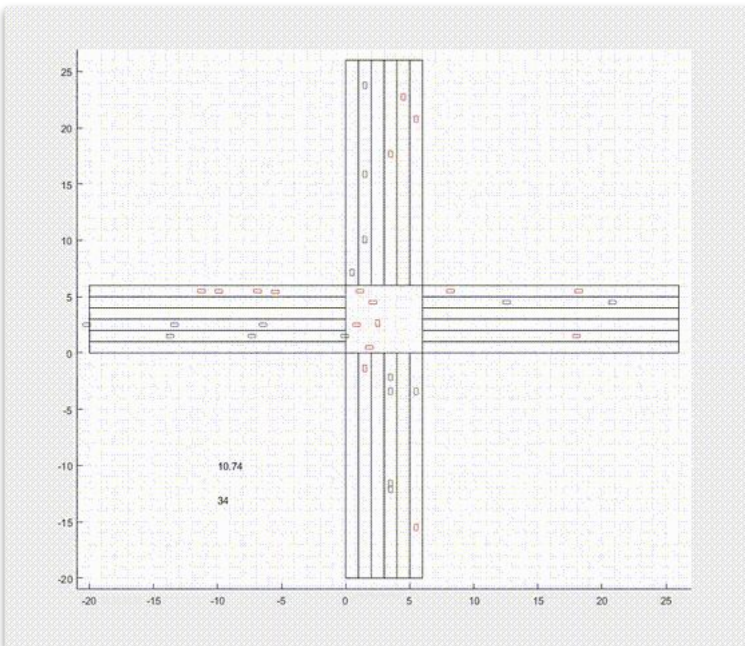


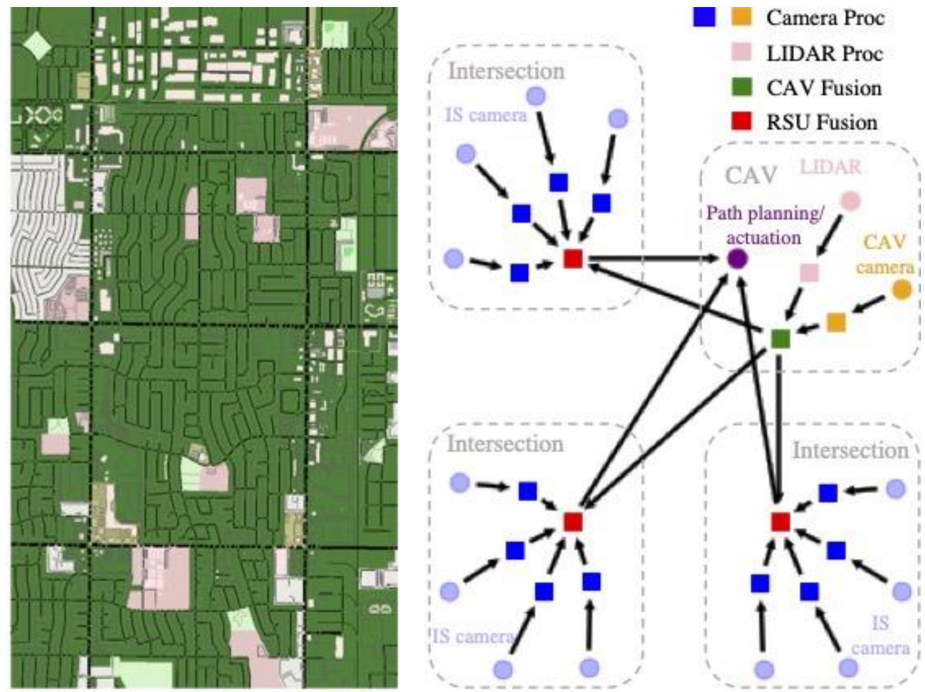
Placeto: visit task equally

GiPH: adjust the placement of "critical" tasks more frequently within the same number of search steps

Cooperative Sensor Fusion

- Autonomous driving with roadside units (RSUs), infrastructure camera sensors, and CAVs
- *Simulation of Urban MObility (SUMO)*

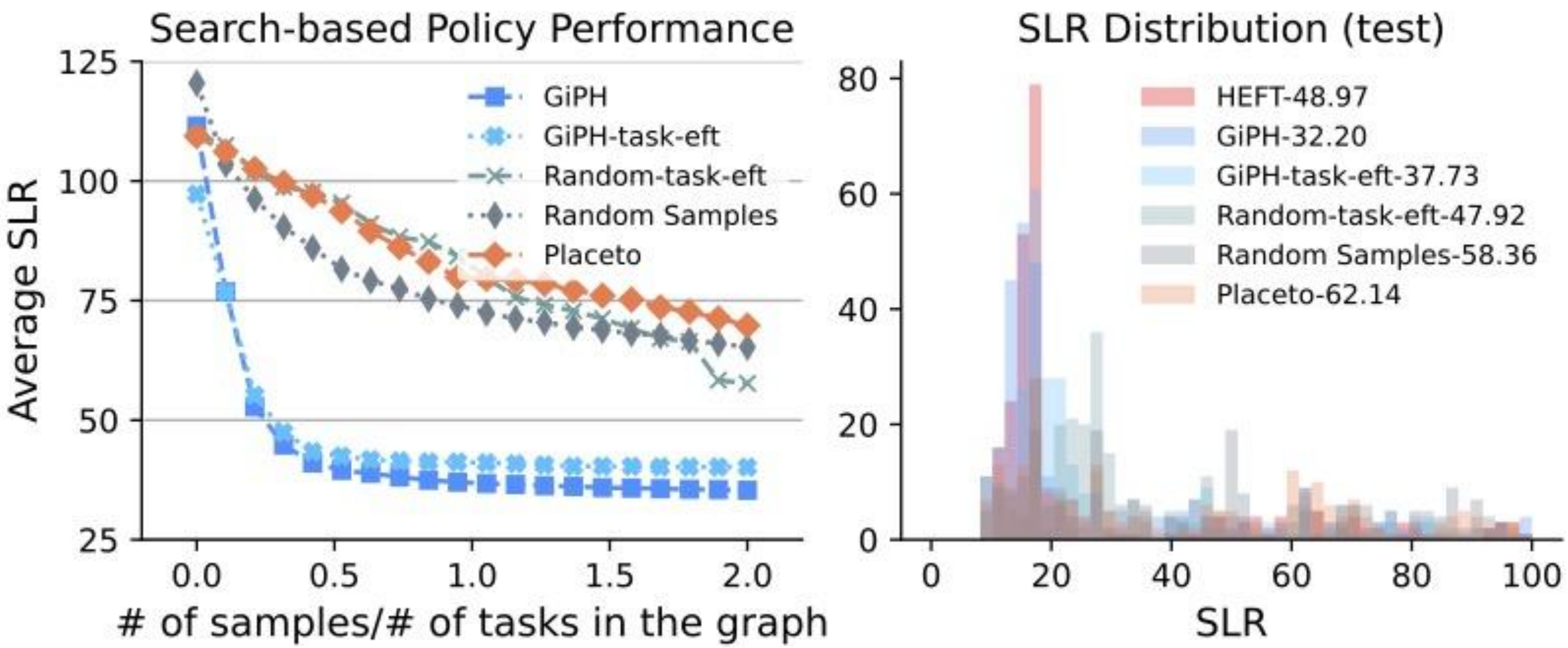




(a) (b)

a 6-block area in the center of Tempe AZ

Cooperative Sensor Fusion



- Find ***better*** placement (up to 30.5% lower SLR) with ***higher*** search efficiency than baselines

Conclusion

- **Formulate the learning problem as a search problem**
  - the policy outputs incremental placement improvement steps
- **Propose GiPH for adaptive placement learning**
  - an RL-based framework for learning ***generalizable*** placement policies for selecting a sequence of placement update steps that ***scale*** to problems of arbitrary size
- **Evaluate on synthetic data and present a case study**
  - GiPH finds placements with up to 30.5% lower SLR, searching up to 3X faster than other search-based placement policies.

➢ **Next step: real-world deployment**
  - Realistic dynamics that accounts for potential relocation overhead and dynamic application arrivals

Thanks!

- Code: https://github.com/uidmice/placement-rl
- Contact: yihu@andrew.cmu.edu

Placement: Makespan Minimization

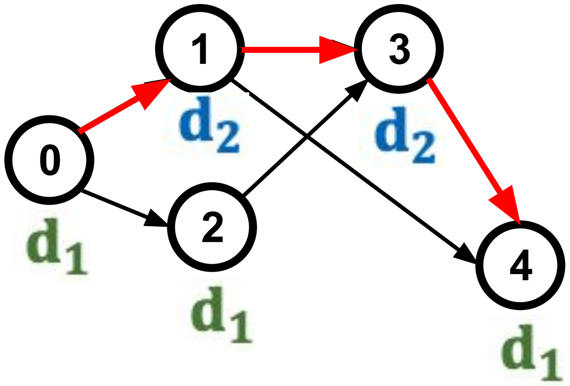
For time-sensitive applications, it is important to minimize the ***completion time***, i.e., makespan

- The time duration from the start of the first task's execution to the end of the last task's execution

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

- The total cost along the critical path
- Depends on the placement of all tasks
- NP-hard

Hard to place the whole graph all at once!

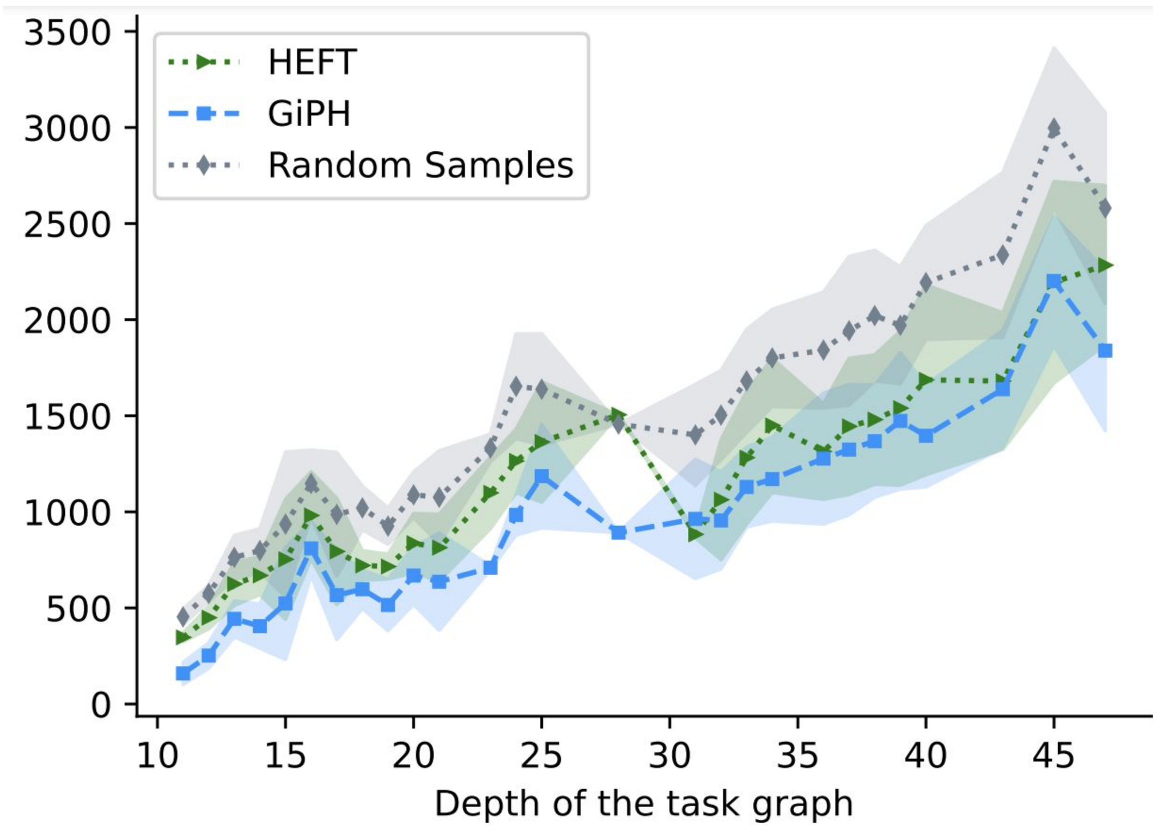


Future Work

- **Consider realistic dynamics**
  - Potential relocation overhead
  - Dynamic application arrival
- **Deploy GiPH on real-world device clusters**
  - Real-world system with centralized administration
- **Extend the work for dynamic task relocation**
  - Time-sensitive task relocation with overhead

Contact: yihu@andrew.cmu.edu

Total Energy Minimization



Challenges

- Combinatorial in nature (NP-hard)
- Devices are heterogeneous
  - Different types
  - Various compute/communication capabilities
- Devices can enter and exit the system
  - The set of devices changes
  - May also change the application graph (data sources)

Require a solution that can **scale to different number of devices** and can **efficiently encode information** as the device set changes.

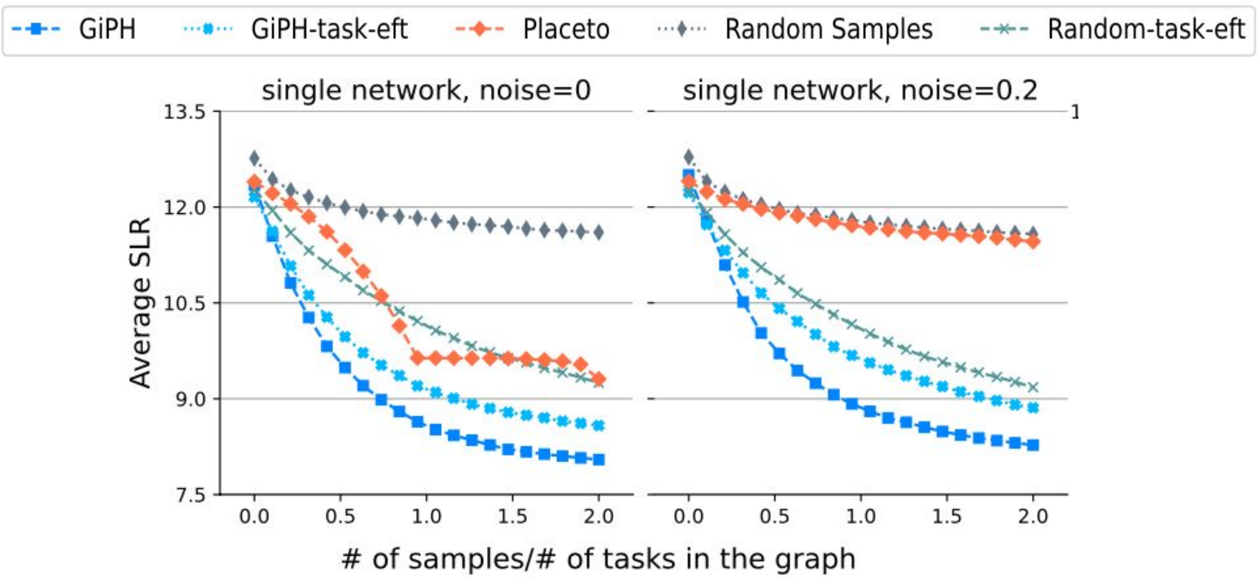
Experiments

- **RL Training: REINFORCE**

$$\theta \leftarrow \theta + \alpha \sum_{t=0}^{T} \gamma^t \nabla_\theta \log \pi_\theta(a_t|s_t) \left(\sum_{t'=t}^{T} \gamma^{t'-t} r_{t'} - b_t \right)$$

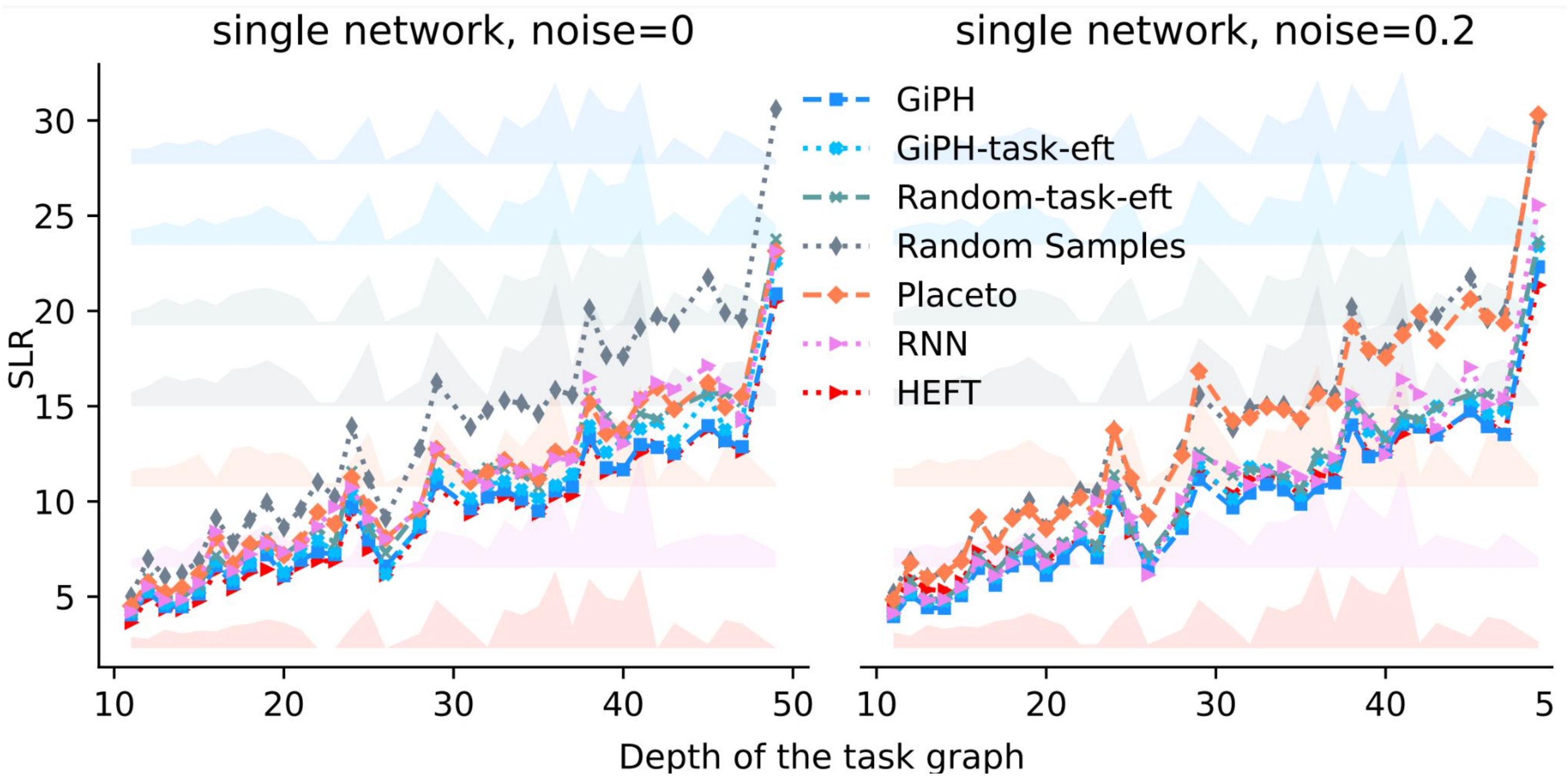
- **Baselines:**
  - Random, HEFT, Placeto, RNN-based Placer
  - Random task selection + EFT device selection
  - GiPH task selection + EFT device selection
- **Datasets:**
  - Synthetic data
  - Realistic application traces (case study)
- **Metrics:**
  - Placement quality (Schedule Length Ratio)
  - Placement adaptivity

Evaluation: Search Efficiency



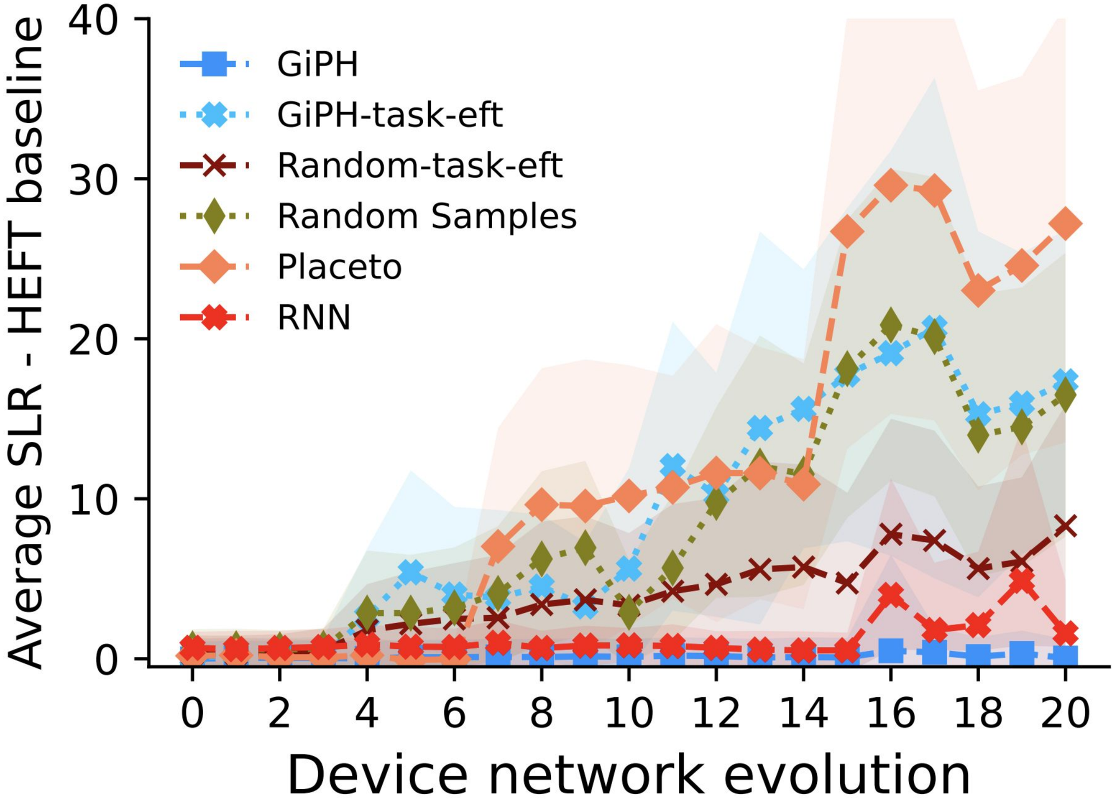
- Find ***better*** placement (up to 30.5% lower SLR) with ***higher*** search efficiency than baselines

Evaluation: Placement Quality



- GiPH outperforms HEFT on 59% of test cases, and ties on 5.2%.
- RNN-placer trained on individual test cases

Evaluation: Adaptivity



- Test on a ***changing*** device network
- As the device network changes, GiPH maintains ***<u>stable performance</u>***