RESEARCH STATEMENT

Junyoung Park
Department of Industrial and Syetems Engineering
KAIST

1 Personal Belief

In recent years, machine learning (ML) has taken fundamental roles in technological innovations, automating simple tasks and improving the quality of life. This, however, has resulted in disruptions in the workforce and provokes a new type of Luddism. Hence, I believe that these powerful tools should be used to solve problems beyond human capabilities, such as those from the computational sciences (e.g., engineering, operation research, and numerical analysis) that require super-human capabilities to find patterns in a wide variety of data.

2 Research Scope & Goals

Graph, a mathematical structure equipped with a set of objects (nodes) and their pairwise relationships (edges), is a natural way to represent the network systems, which range from microscopic (e.g., molecules) to macroscopic (e.g., semiconductor fab) systems. Graph representation learning is a field of ML, where the models learn to process graph-represented data and perform the downstream tasks. Among the graph-centric ML models, graph neural network (GNN) is the most popular model class. When GNNs are applied to solve computational science problems, GNN can enjoy a flexible yet powerful inductive bias encoded in the graph structure (e.g., the connected nodes in the graph are related) that can be leveraged to improve prediction and generalization performances. Those "relational" inductive bias (i.e., graph structure) of GNN brings a series of impactful scientific/engineering discoveries, including enabling the data-driven identification of Lagrangian, predicting the 3D structure of proteins, and solving long-standing scheduling problems.

Even though the graph representation provides a certain level of inductive bias to the model, GNN still exhibits limitations in sample complexities (i.e., need of a large amount of training data) and generalizations (i.e., counter-intuitive predictions for out of training samples). Unlike vision and acoustic systems, when modeling scientific/engineering systems, we expect to know certain aspects of the systems in a crude and/or partial manner. It becomes natural to bring the prior knowledge into the learning procedure of GNN so that the resulting GNN overcomes the mentioned limitations. In that regard, My research interest is the development of methods that seamlessly integrate the prior knowledge of systems into GNN so that it improves the prediction/generalization properties of GNN in solving scientific computations.

3 Previous & On-going Researches

Among the various types of scientific computing problems, I majorly have focused on solving two canonical problem classes of followings:

- **Model identification problems**: the graph ML learns to predict the future states or properties of the system as a fast-and-reliable simulation method.
- **Sequential decision making (or control) problems**: the graph ML model learns to control the state of the system.

3.1 GNN approaches to solve model identification problems

Solving the modeling problems, which is done to predict the future states of a given system, is essential in computational sciences. The prediction results are crucial when analyzing the target system, and can be combined with a model-based decision-making process to control the target system. I have investigated the potentials of prior knowledge-augmented GNN when solving modeling problems. The followings are the list of published or ongoing researches pursuing this research direction:

- **Physics-induced graph neural network: An application to wind-farm power estimation (Energy 2019) [1]**: This study proposes a physics-inspired GNN (PGNN) that estimates the power outputs of all wind turbines in any layout and wind conditions. PGNN employs the engineering wake model as the attention model to compute the intensity of interactions among the wind turbines and learns the parameters of the engineering model during training. Compared to pure data-driven GNN models, PGNN shows significant supremacy when predicting the powers from the wind turbines and has better scalability than the larger-sized wind farms. Furthermore, based on the fact that PGNN is a differentiable surrogate model, *gradient-based* wind farm layout optimization has been performed and has achieved accurate/reliable optimization results.
- **A Molecular Hyper-message Passing Network with Functional Group Information (Preprint)** [2]: This study proposes a hypergraph neural network model (MolHMPN) that predicts the properties of the molecules from their (hyper)graph representations, where the hypergraphs are constructed using functional group information from the chemistry domain. Here, the use of functional group information increases the molecule property classification/prediction performances and shows sample and parameter efficiency compared to the plain GNN models. Also, the efficacy of the functional groups is investigated by comparing them with other purely data-driven hypergraph construction methods. Furthermore, MolHMPN is designed so that the functional group information can be revised during training. The revision allows additional increment of the prediction performances of MolHMPN.
- **Convergent Graph Solvers (ICLR 2022) [3]**: This study proposes the convergent graph solvers (CGS), a deep learning method that learns iterative mappings to predict the properties of a graph system at its stationary state (fixed point) with guaranteed convergence. CGS systematically computes the fixed points of a target graph system and decodes them to estimate the system's stationary properties without the prior knowledge of existing solvers or intermediate solutions. CGS shows that the computation of network-constructed fixed points is a key ingredient to predict the network properties realized at stationary states of the system. CGS is applied to solve value iteration and elliptic PDE, which involve iterative computations. CGS shows better solution predictability and generalization from the applications than the baseline fixed depth GNN models.
- **Graph Neural Ordinary Differential Equations (AAAI DLGMA20, spotlight) [4]**: This study proposes the framework of continuous-depth graph neural networks (GNNs), blending discrete topological structures and differential equations. The proposed framework is compatible with various static GNN models and is extended to autoregressive and stochastic settings through hybrid dynamical system theory.
- **Input Convex Graph Neural Networks (in-progress)**: This study proposes a family of GNNs that maps inputs and outputs via a convex function (ICGNN), and such input convexity of GNN is attained by constraining a subset of parameters to be positive and a careful selection of activation functions. It shows that major GNN architectures can be reformulated to be input convex without significant modifications. The input-convexity of ICGNN improves the long-term prediction performance of the diffusion process as it guides GNNs to learn convex functions and improves the control performance as the control problem becomes convex. Furthermore, in a bi-level optimization setting where the inner problem is for control and upper problem is for design optimization, the convexity of ICGNN enables the unbiased estimation of inner-level problems' gradient and produces a better design.
- **Meta-SysId: A Meta-Learning Approach for Simultaneous Identification and Prediction (Neurips 2022, under review)**: This study propose a meta-learning method to model sets of systems that have behavior governed by common but unknown laws and that differentiate themselves by their context. Inspired by classical modeling-and-identification approaches, Meta-SysId learns to represent the common law through shared parameters and relies on online optimization to compute system-specific context. Compared to optimization-based meta-learning methods, the separation between class parameters and context variables reduces the computational burden while allowing batch computations and a simple training scheme. Meta-SysId is tesed on the polynomial regression, time-series prediction, model-based control, and real-world traffic prediction domains, empirically finding it outperforms or is competitive with meta-learning baselines.

3.2 GNN approaches to solve sequential decision-making (control) problems

To solve the sequential decision-making problems, the decision-making agent (e.g., the controller in optimal control) must acknowledge the effect of the current actions on the future states. The graph representation and additional inductive

biases can help the agent infer the current state from the graph observations and predict the future states/outcomes. I have investigated such approaches in combinatorial optimization (CO) and partial differential equation (PDE)-governed systems. The followings are the list of published or ongoing researches pursuing this research direction:

- **Learning to schedule job-shop problems: representation and policy learning using graph neural network and reinforcement learning (IJPR 2021) [5]**: This study proposes a framework to learn to schedule a job-shop problem (JSP) using GNN and reinforcement learning (RL) for differently-sized job-shop scheduling problems in a sequential decision-making framework. JSP is a CO problem with complex constraints, and each constraint describes the relationships among the problem entities. In this study, each type of relationship is modeled as the different edge types of graphs and processed by separate NNs. Those explicit distinctions of the type information lead to better representability of the graph entities and result in better scheduling performances. Furthermore, the proposed method shows remarkable generalization capabilities compared to baseline RL methods.
- **ScheduleNet: Learn to solve multi-agent scheduling problems with reinforcement learning (Preprint) [6]**: This study proposes ScheduleNet, an RL-based real-time scheduler that can solve various types of multi-agent scheduling problems. It formulates the multi-agent scheduling problem as an MDP with episodic reward (make-span) and derives a decentralized decision-making policy, effectively inducing coordination among multiple agents to complete the given tasks. ScheduleNet generalizes the approaches that have been investigated in the JSP paper so that it can solve scheduling problems where excessively many relation types exist by using a single NN but using a type-aware message aggregation scheme. ScheduleNet shows that such type-aware aggregation helps to attain significant improvements in scheduling performance and better scalability (generalization) in problem sizes.
- **Input Convex Graph Neural Networks (in-progress)**: This study proposes a family of GNNs that maps inputs and outputs via a convex function (ICGNN), and such input convexity of GNN is attained by constraining a subset of parameters to be positive and a careful selection of activation functions. It shows that major GNN architectures can be reformulated to be input convex without significant modifications. The input-convexity of ICGNN improves the long-term prediction performance of the diffusion process as it guides GNNs to learn convex functions and improves the control performance as the control problem becomes convex. Furthermore, in a bi-level optimization setting where the inner problem is for control and upper problem is for design optimization, the convexity of ICGNN enables the unbiased estimation of inner-level problems' gradient and produces a better design.
- **Sym-NCO: Leveraging Symmetricity for Neural Combinatorial Optimization (Neurips 2022, under review)** This paper presents an effective training scheme for deep reinforcement learning-based combinatorial optimization (CO) methods (DRL-NCO), Sym-NCO, that achieves significant performance increments to existing DRL-NCO methods. Sym-NCO is a regularizer-based trainin scheme that leverages universal symmetricities in various and solutions. Imposing symmetricities such as rotational and reflectional invariance can greatly improve generalization capability of DRL-NCO as symmetricities are invariant features shared by certain CO tasks. Our experimental results verify that our Sym-NCO greatly improves the performance of DRL-NCO methods in four CO tasks, including traveling salesman problem (TSP), capacitated vehicle routing problem (CVRP), prize collecting TSP (PCTSP), and orienteering problem (OP), without employing problem-specific techniques. Remarkably, Sym-NCO outperformed not only the existing DRL-NCO methods but also a competitive conventional solver, the iterative local search (ILS), in PCTSP at 240$\times$ faster speed.
- **Neuro CROSS exchange: Learning to CROSS exchange to solve realistic vehicle routing problems (Neurips 2022, under reivew)**: This paper proposes, Neuro CROSS exchange (CE), a learned improvement heuristics to solve various and practical vehicle routing problems (VRPs). CE is a meta-heuristic that solves various VRP, which improves the solutions of VRPs by swapping the sub-tours of the vehicles. Inspired by CE, we propose Neuro CE (NCE), a fundamental operator of *learned* meta-heuristic, to solve various VRPs while overcoming the limitations of CE (i.e., the expensive $\mathcal{O}(n^4)$ search cost). NCE employs graph neural network to predict the cost-decrements (i.e., results of CE searches) and utilizes the predicted cost-decrements as guidance for search to decrease the search cost to $\mathcal{O}(n^2)$. As the learning objective of NCE is to predict the cost-decrement, the training can be simply done in a supervised fashion, whose training samples can be prepared effortlessly. Despite the simplicity of NCE, numerical results show that the NCE trained with flexible multi-depot VRP (FMDVRP) outperforms the meta-heuristic baselines. More importantly, it significantly outperforms the neural baselines when solving distinctive special cases of FMDVRP (e.g., MDVRP, mTSP, CVRP) without additional training.

4 Future Research Directions

4.1 Neural accelerated DE solvers

In scientific computing, a similar class of problems with different parameter settings is proposed and solved routinely to analyze the target systems. This scenario is perfect for employing ML-accelerated solvers tailored for such problem

sets. The traditional solvers are devised (or tuned) to solve the entire classes of the problem, thus they can be slower to solve a specific set of problems. However, the learned solvers may offer a path to the faster solution search by enjoying the structural similarity among the problem samples and their corresponding solutions.

The majority of the learned differential equation solvers employ GNN and train the model to predict the discretized PDE solutions, and they show faster inference than widely used conventional methods with an allowable error level. However, these learned solvers still are at the risk of model introduced biases and limited generalization capabilities. The followings are the directions of research that I wish to explore to mitigate the limitations:

- **Interplay with classical solvers**: I believe the model-introduced bias can be alleviated by combining the ML model and existing numerical computation methods. A similar idea has been employed to accelerate the fixed point iterations, quadratic programming, linear-system solving, and numerical PDE solving and result in faster convergences while preserving convergence properties.
- **Respect the known rules**: I believe the generalization capability of learned solvers can be improved by incorporating the known rules of the modeling targets. For instance, time-translation invariance (e.g., energy conservation) or roto-translational eqiu/in-variance of the domain to solutions of PDE is one of the fundamental properties of physical simulations that should be considered in ML.

4.2 Practical neural CO solvers

Combined with recent ML developments, the learned CO solvers outperform the human-design heuristics/exact methods to solve benchmark CO problems (e.g., TSP, CVRP, JSP) in performance and computational speed. However, those learned CO solvers still are not suitable for practical scheduling problems as they are shown to be effective to small-sized problems or relatively simple constraints compared to real-world applications. The followings are the directions of research that I wish to explore to mitigate the limitations:

- **Problem size-scalablity**: Almost every learned CO solvers assume to encode/decode the entire C0 problem graph to solve it. However, this specifically prevents the learned solvers from being used in large-sized problems as this requires at least a linearly increasing memory bandwidth for the execution and training of the solver. It can be resolved by adopting localized/stochastic graph embedding techniques. Disentangling the memory usage and the size of the graph provides an opportunity for the learned solver to be used in larger problems.
- **Applicability for complex constraints/scenarios**: Due to the difficulty of constraint enforcement during training/inference of learned solvers, they tend to be used only in simpler or benchmark-like scheduling problems. The limitation can be fixed by integrating the learned solver as a component of the entire problem procedure. For instance, the learned solver is used to solve sub-problems of metaheuristics or pricing problems in Branch-price-cut algorithms.

References

[1] Junyoung Park and Jinkyoo Park. Physics-induced graph neural network: An application to wind-farm power estimation. *Energy*, 187:115883, 2019.

[2] Fangying Chen, Junyoung Park, and Jinkyoo Park. A hypergraph convolutional neural network for molecular properties prediction using functional group. *arXiv preprint arXiv:2106.01028*, 2021.

[3] Junyoung Park, Jinhyun Choo, and Jinkyoo Park. Convergent graph solvers. In *International Conference on Learning Representations*, 2022.

[4] Michael Poli, Stefano Massaroli, Junyoung Park, Atsushi Yamashita, Hajime Asama, and Jinkyoo Park. Graph neural ordinary differential equations. *arXiv preprint arXiv:1911.07532*, 2019.

[5] Junyoung Park, Jaehyeong Chun, Sang Hun Kim, Youngkook Kim, and Jinkyoo Park. Learning to schedule job-shop problems: representation and policy learning using graph neural network and reinforcement learning. *International Journal of Production Research*, 59(11):3360–3377, 2021.

[6] Junyoung Park, Sanjar Bakhtiyar, and Jinkyoo Park. Schedulenet: Learn to solve multi-agent scheduling problems with reinforcement learning. *arXiv preprint arXiv:2106.03051*, 2021.