Modular Construction Planning using Graph Neural Network Heuristic Search

Philip Hawkins[1], Frederic Maire[1], Simon Denman[1] and Mahsa Baktashmotlagh[2]
School of Electrical Engineering and Robotics, Faculty of Engineering, Queensland University of Technology[1],
School of Information Technology and Electrical Engineering, University of Queensland[2]

ABSTRACT

We present a framework for learning to design and assemble modular structures using a search guided by a *Graph Neural Network* heuristic, trained through *Imitation Learning*. We demonstrate the effectiveness of this framework by planning the construction of modular spanning truss structures of a range of shapes and sizes given simple specifications.

INTRODUCTION

Graphs provide a flexible and generic method of representing real-world binary relationships and structures. Generating novel structures that achieve an objective while adhering to constraints is an important problem in modelling physical, chemical and social systems. However, applying Deep Learning to graph generation presents several challenges. Among these are the discrete nature of graph structures, the fact that the same graph has many possible representations depending on the ordering of its nodes and edges, and the large solution space formed by the number of possible structures.

Generating graphs by sequential construction has been effective in domains with a continuous objective metric. However, an additional challenge arises when the objective is a binary pass-or-fail test. In this case, a process must look ahead to compare alternative sequences of actions to avoid progressing to a dead-end.

We address this challenge by training an agent to design and plan the assembly of simple modular truss structures, as illustrated in Figure 1. The objective is for the agent to find a sequence of assembly steps that complete a truss structure that spans from a starting platform to an assigned target support point while meeting structural constraints at each step of construction.

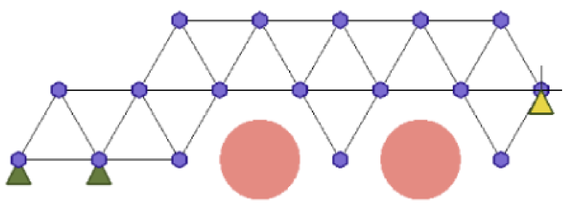

Figure 1. The truss structure, generated by neural heuristic search, is composed of *hexagonal hubs* and *connecting spokes*. These are represented as nodes and edges in the state graph. The static starting points are shown as green triangles on the right, with the target point shown as the yellow triangle to the left. The two red patches indicate obstacles that the truss must avoid. Not all generated trusses are optimal. In this example the two hubs extending down around the obstacles are not necessary for the stability of the structure.

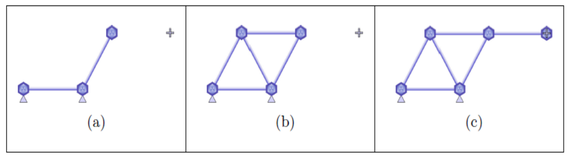

Figure 2. A **starting platform** is shown in this sequence as a pair of pinned hexagonal hubs connected by a spoke. The **+** to the right indicates the target point. To satisfy the stability constraint, each spoke must connect to at least one stable hub. A hub is considered stable if it is part of the starting platform, or if it is directly connected to two other stable hubs. The structure in (a) has a hub that is one spoke length from the target point. Placing a spoke connecting this hub to the target would violate the stability constraint. In (b), three spokes have been added to brace the structure, making all the hubs stable. A spoke may now be placed as shown in (c).

PROBLEM FORMULATION

In our problem, we require a single agent to plan, with no prior knowledge, the construction in the vertical plane of a two-dimensional truss structure composed of uniform hubs and connecting spokes. The completed structure will span the distance between a static starting platform and a randomly placed target point, as shown in Figure 1.

A plan is composed of a sequence of spoke placement instructions. At each step, the structure must be adequately supported by bracing struts. This is tested against a stability constraint requiring each spoke to connect to at least one stable hub. A hub is considered stable if it is part of the starting platform, or if it is directly connected to two other stable hubs. This is illustrated in Figure 2. The target criteria is that a hub at the target point is stable.

METHODOLOGY

Applying a search framework, we consider each stage of assembly as a state in the search space. The search objective is a structure that meets the target criteria, and the construction plan is provided by the path of actions required to reach it. An optimal solution for this could theoretically be provided by an A* graph search with a consistent heuristic if the search tree is not too large. However, in this problem, the branching factor is proportional to the perimeter of the structure, and in practice A* search time increases more than exponentially with respect to the length of the solution.

We address this by producing a search heuristic using a graph neural network. We train the graph network to estimate the cost-to-go value of a state by estimating the number of steps an optimal A* search would require to complete the structure. This is achieved through Imitation Learning of a structural stable distance heuristic from exemplar A* searches.

Optimal searches are computationally expensive and provide a limited set of training data. We therefore augment this by adding synthetic data derived from states that are adjacent to the optimal paths. We demonstrate that this approach allows our building agent to extrapolate to build structures much larger than those seen in training.

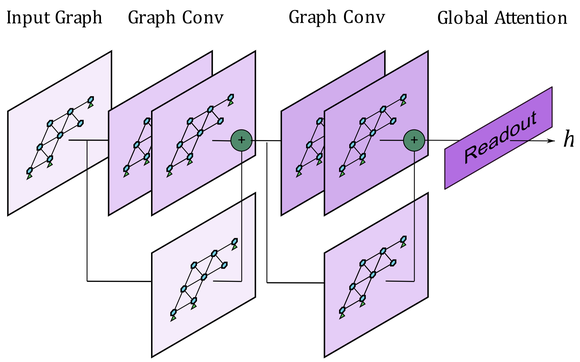

Figure 3. Stacked residual blocks of graph convolutions extract latent vector representations of the local graph region for each node. The cost-to-go heuristic readout is computed from these node vectors using a global soft attention layer.

It is more difficult for A* to navigate around unstable truss states than greedy search because the cost-so-far value, penalises leaves that are lower in the search tree. As illustrated in Figure 2, it is sometimes necessary to add spokes that do not reduce the heuristic distance to avoid instability. This weights the search toward states in the frontier that have not yet encountered instability. These low-cost states must be exhausted before A* considers the more costly states that must be traversed to avoid the instability. We therefore use a greedy search for inference and an A* search for exemplar data generation.

To ensure that data generation completes in reasonable time, we restrict the span distance of the exemplar searches. We run exemplar searches to determine optimal build sequences for a truss spanning to each target lattice vertex in a training range. This range was set at spans of up to three spokes from the starting platform. The A* search requires approximately 15,000 state expansions to complete the three-spoke example spanning targets.

RESULTS

We evaluated the efficiency of neural heuristics against the two simple distance heuristic baselines. Neural heuristics based on the network illustrated in Figure 3 were evaluated, comparing two graph convolutional operators: the graph attention operator (GAT) and the graph isomorphism operator (GIN).

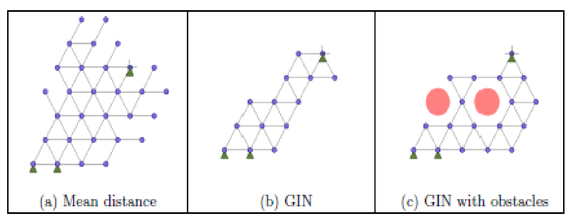

Figure 4. Examples of trusses generated for a five spoke span.

Table 1. Each agent was given an identical set of 25 target points for each spanning distance and tasked to plan an assembly sequence for each target. The mean path length and number of search tree nodes that were expanded are shown for each set. Search was halted after 1000 expansions. The ¨Completed'' column shows the rate of completion for each set due to this limit.

	span=5			*span=10*		
Search Heuristic	Path Length	Expanded	Completed	Path Length	Expanded	Completed
Min.	57.0±6.0	793.5±93.2	56.0%			0.0%
Mean	44.7±3.1	85.7±9.2	100.0%	152.0±13.8	392.1±50.3	100.0%
GAT	26.9±1.3	26.9±1.3	100.0%	68.4±2.4	70.2±3.1	100.0%
GIN	**22.5±1.2**	**22.5±1.2**	100.0%	**48.1±1.5**	**48.1±1.5**	100.0%

a) Without obstruction between the starting platform and the target point

	span=5			*span=10*		
Search Heuristic	Path Length	Expanded	Completed	Path Length	Expanded	Completed
Min.	52.2±5.2	984.6±29.6	4.0%			0.0%
Mean	43.8±2.2	91.1±10.6	100.0%	162.7±16.5	415.8±56.4	100.0%
GAT	37.2±2.2	84.5±73.6	96.0%	64.9±3.4	308.5±155.1	76.0%
GIN	**36.6±3.5**	**58.8±27.1**	100.0%	**57.4±2.3**	**60.1±3.5**	100.0%

b) With two obstructions as shown in Figure 1

CONCLUSION

We demonstrated that our approach simultaneously improved both search efficiency and material efficiency in comparison to agents using non-neural heuristics. A key factor in this improvement was the ability of the graph neural network to learn patterns from small structures at training time, and during inference scale this well beyond the spatial extents of the training examples.

An avenue for future research is to remove the hard stability constraint calculation and investigate learning the constraining dynamics directly from a physical simulation. This would follow a similar trajectory to the development from AlphaZero to MuZero in learning board game dynamics from experience. However, the truss assembly problem has a sparse reward. We therefore believe that imitation learning as described here, or another form of reward shaping will continue to be a necessary stage in modular design problems that follow a combinatorial search strategy.

ACKNOWLEDGEMENTS

We acknowledge continued support from the Queensland University of Technology (QUT) through the Centre for Robotics. This research is supported by an Australian Federal Government Research Training Program (RTP) scholarship. Computational resources and services used in this work were provided by the HPC and Research Support Group, Queensland University of Technology, Brisbane, Australia.