Graph Networks for Graph Theory

Sara Veneziale

Imperial College London

Plan

- AI for Pure Mathematics
- Reinforcement Learning
- Counterexamples in graph theory
- Graph Neural Networks
- GNNs for graph constructions

This is work in progress joint with Fanglan Feng

AI for Mathematics

Different flavours

- For **conjecture generation**: in knot theory and representation theory (Davies et al 2021), in number theory (He et al 2024), algebraic geometry (Coates et al 2023).

AI for Mathematics

Different flavours

- For **conjecture generation**: in knot theory and representation theory (Davies et al 2021), in number theory (He et al 2024), algebraic geometry (Coates et al 2023).
- To **accelerate computational routines**: in computer algebra (Peifer et al 2020), mathematical physics (Jejjala et al 2022).

AI for Mathematics

Different flavours

- For **conjecture generation**: in knot theory and representation theory (Davies et al 2021), in number theory (He et al 2024), algebraic geometry (Coates et al 2023).
- To **accelerate computational routines**: in computer algebra (Peifer et al 2020), mathematical physics (Jejjala et al 2022).
- To **automate proofs**: AlphaProof and AlphaGeometry.

AI for Mathematics

Different flavours

- For **conjecture generation**: in knot theory and representation theory (Davies et al 2021), in number theory (He et al 2024), algebraic geometry (Coates et al 2023).
- To **accelerate computational routines**: in computer algebra (Peifer et al 2020), mathematical physics (Jejjala et al 2022).
- To **automate proofs**: AlphaProof and AlphaGeometry.
- For **conjecture disproval**: in graph theory (Wagner 2021), knot theory (Freedman et al 2010), polytope theory (Swirszcz et al 2025).

AI for Mathematics

Different flavours

- For **conjecture generation**: in knot theory and representation theory (Davies et al 2021), in number theory (He et al 2024), algebraic geometry (Coates et al 2023).
- To **accelerate computational routines**: in computer algebra (Peifer et al 2020), mathematical physics (Jejjala et al 2022).
- To **automate proofs**: AlphaProof and AlphaGeometry.
- For **conjecture disproval**: in graph theory (Wagner 2021), knot theory (Freedman et al 2010), polytope theory (Swirszcz et al 2025).

In this talk we focus on using reinforcement learning to find counterexamples to conjectures.

Some questions

Key Problems

Sampling bias
What is a random variety or a random knot?

Some questions

Key Problems

Sampling bias
What is a random variety or a random knot?

Data representation
Can machine learning models respect symmetry?

Some questions

Key Problems

Sampling bias	Data representation	Explainability
What is a random variety or a random knot?	*Can machine learning models respect symmetry?*	*Can we get an insight into the proof?*

Some questions

Key Problems

Sampling bias	Data representation	Explainability
What is a random variety or a random knot?	*Can machine learning models respect symmetry?*	*Can we get an insight into the proof?*

Let's think about model choice and data representation today

RL for Counterexamples

AI to disprove conjectures

Wagner in 2021 used cross-entropy and a deep neural network to approximate the policy to find **counterexamples to conjectures** in graph theory.

RL for Counterexamples

AI to disprove conjectures

Wagner in 2021 used cross-entropy and a deep neural network to approximate the policy to find **counterexamples to conjectures** in graph theory.

Which conjectures?

“over all combinatorial structures X
an associated numerical quantity Z is bounded by B.”

Example conjecture

Conjecture from AutoGraphiX

Given any connected graph G on $n \geq 3$ vertices, with largest eigenvalue λ and matching number μ we have

$$\lambda + \mu - \sqrt{n-1} - 1 \geq 0$$

[The conjecture had already been disproved by Stevanović, who provided an example with 600 vertices.]

What is reinforcement learning?

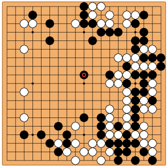
We will reframe the problem of building a graph as a game.But first, what is reiforcement learning?

What is reinforcement learning?

We will reframe the problem of building a graph as a game.But first, what is reiforcement learning?

Agent

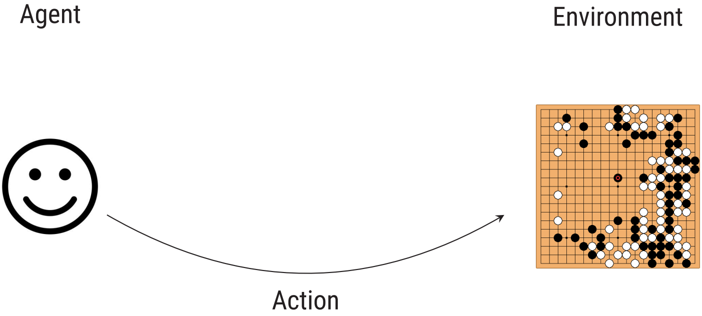
Environment



Wagner uses a simple reinforcement learning algorithm called **(deep) cross entropy method**.

What is reinforcement learning?

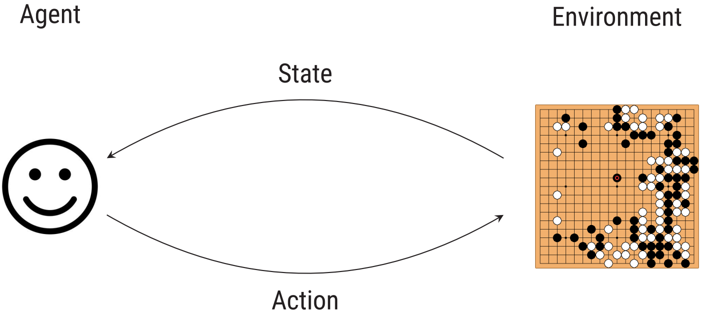
We will reframe the problem of building a graph as a game.But first, what is reiforcement learning?



Wagner uses a simple reinforcement learning algorithm called **(deep) cross entropy method**.

What is reinforcement learning?

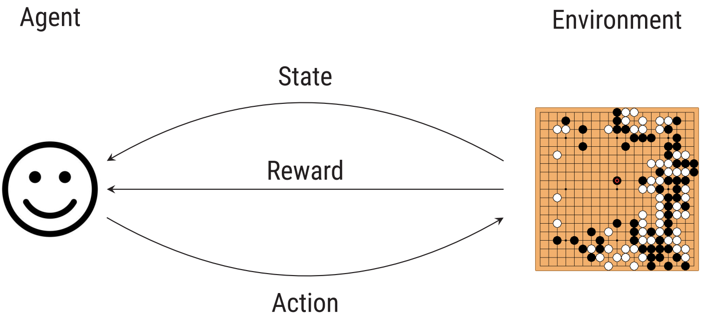
We will reframe the problem of building a graph as a game.But first, what is reiforcement learning?



Wagner uses a simple reinforcement learning algorithm called **(deep) cross entropy method**.

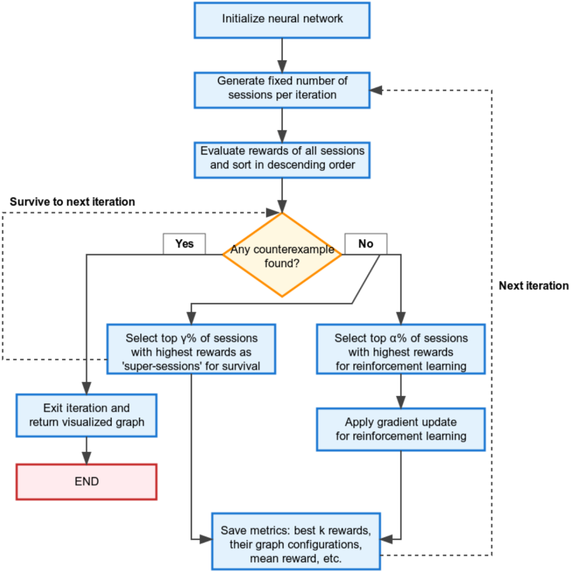
What is reinforcement learning?

We will reframe the problem of building a graph as a game.But first, what is reiforcement learning?



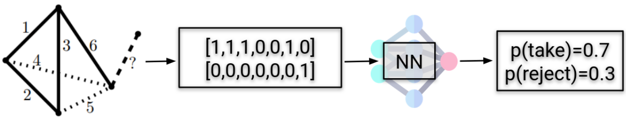
Wagner uses a simple reinforcement learning algorithm called **(deep) cross entropy method**.

Cross entropy



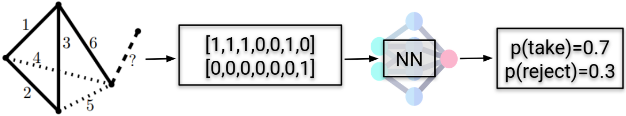
Method

For us, the game is *Do I keep this edge?*



Method

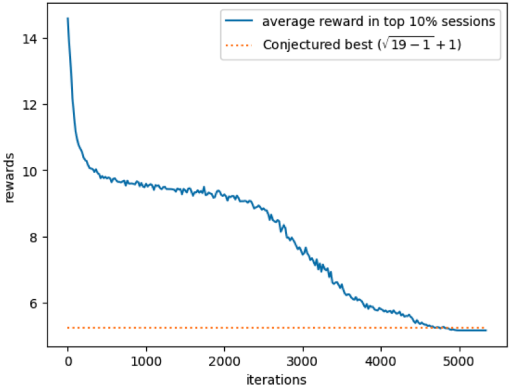
For us, the game is *Do I keep this edge?*



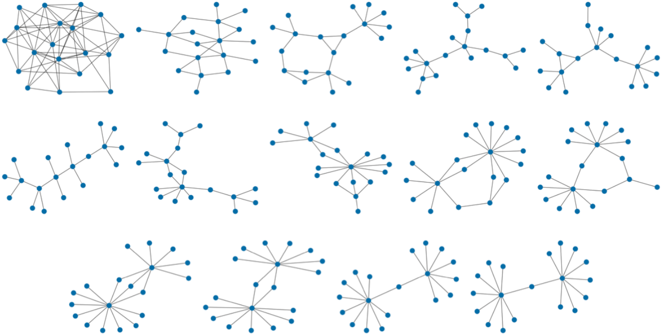
- Reward function: $\lambda + \mu$, the aim is to minimise it [Of course the reward depends on the conjecture you are disproving]
- Input: two vectors, one keeping track of the edges that have been taken so far, and one indicating the edge that is under consideration.
- The action are sampled according to an MLP.

Results

Wagner finds a counterexample to the above conjecture with $n = 19$.



Results



Further work

Limitations

- It is quite slow (it needs 5 000 iterations to find a counterexamples).
- It is not very robust (converges roughly 30% of the time).

Further work

Limitations

- It is quite slow (it needs 5 000 iterations to find a counterexamples).
- It is not very robust (converges roughly 30% of the time).

- Roucairol et al. 2022
- Angileri et al. 2024
- Ghebleh et al. 2024

But what about the graphs?

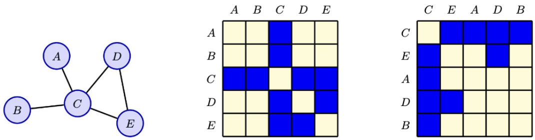
He uses a standard MLP on a representation that loses all graph theoretic information (plus we have to fix the number of nodes and a node ordering).

But what about the graphs?

He uses a standard MLP on a representation that loses all graph theoretic information (plus we have to fix the number of nodes and a node ordering).

How can we learn on graphs?

Consider a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. A convenient way to represent it is using an *adjacency matrix*, and use it as the input to a sequential or convolutional network. For this we have to choose a node ordering.

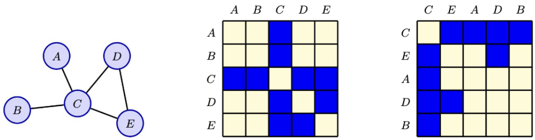


But what about the graphs?

He uses a standard MLP on a representation that loses all graph theoretic information (plus we have to fix the number of nodes and a node ordering).

How can we learn on graphs?

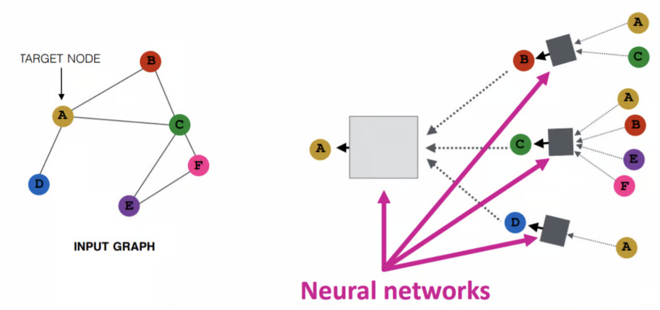
Consider a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. A convenient way to represent it is using an *adjacency matrix*, and use it as the input to a sequential or convolutional network. For this we have to choose a node ordering.



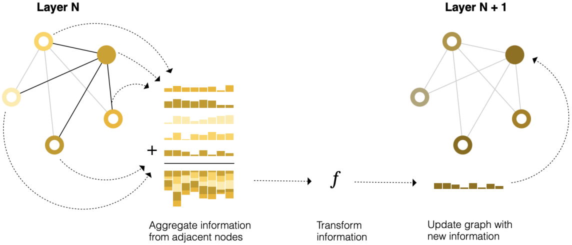
Problem: Most questions about graphs do not depend on the choice of node ordering.

Graph Neural Networks

Answer: Graph Neural Networks. They learn a meaningful embedding of the nodes by a series of layers that perform an operation called *message passing*.



Graph Neural Networks



Let h_i^k be the embedding of the ith node at layer k,

$$h_i^{k+1} = f\left(\sum_{j \in \mathcal{N}(i) \cup \{i\}} \frac{1}{\sqrt{\deg i}\sqrt{\deg j}} (\mathbf{W}_k^T \cdot h_j^k) + \mathbf{b_k}\right)$$

the weight matrix $\mathbf{W}_{k-1}$ (and the bias vector $\mathbf{b}_{k-1}$) is shared by all nodes and depends on the layer. Here f could be something like ReLU.

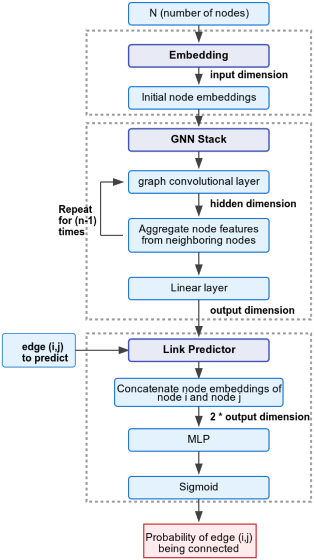
Graph Neural Networks

- **Node task**: e.g., use the final node embeddings for classification/regression.
- **Edge task**: e.g., get the probability of an edge between nodes *i* and *j* by

$$p(i,j) = \sigma((h_i^N)^T \cdot h_j^N)$$

  where σ is just the sigmoid function.
- **Graph task**: e.g., aggregate node embeddings to a unique graph embedding and use that for classification/regression.

Our model



N (number of nodes)
Embedding
input dimension
Initial node embeddings
GNN Stack
graph convolutional layer
hidden dimension
Aggregate node features from neighboring nodes
Repeat for (n-1) times
Linear layer
output dimension
edge (i,j) to predict
Link Predictor
Concatenate node embeddings of node i and node j
2 * output dimension
MLP
Sigmoid
Probability of edge (i,j) being connected

Comparison

We replace the MLP by our GNN model, to approximate the policy function, leaving the cross entropy method unchanged.

Preliminary results

Our method

- exhibits a more stable convergence (previous method converged only 30% of the times – whilst we converge roughly 90% of the times)

Comparison

We replace the MLP by our GNN model, to approximate the policy function, leaving the cross entropy method unchanged.

Preliminary results

Our method

- exhibits a more stable convergence (previous method converged only 30% of the times – whilst we converge roughly 90% of the times)
- converges much more quickly (20–30 iterations vs 5 000–10 000 iterations) and with fewer sessions per iteration (250–300 vs 1 000)

Comparison

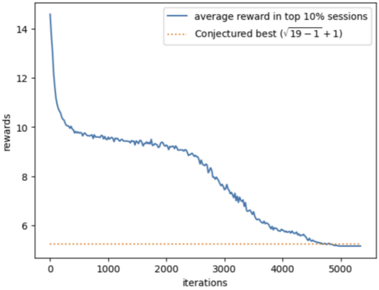
We replace the MLP by our GNN model, to approximate the policy function, leaving the cross entropy method unchanged.

Preliminary results

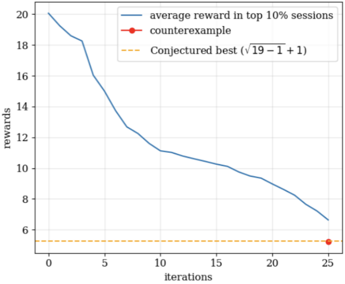
Our method

- exhibits a more stable convergence (previous method converged only 30% of the times – whilst we converge roughly 90% of the times)
- converges much more quickly (20–30 iterations vs 5 000–10 000 iterations) and with fewer sessions per iteration (250–300 vs 1 000)
- it will hopefully be much more flexible (e.g. node colouring problems, working with graphs with different number of nodes etc).

Comparison

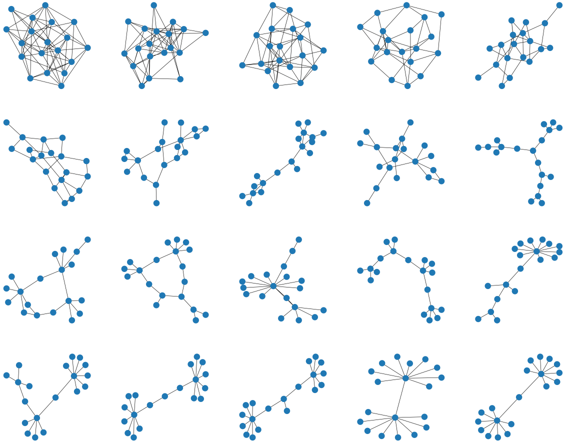


average reward in top 10% sessions
Conjectured best ($\sqrt{19-1}+1$)
rewards
iterations



average reward in top 10% sessions
counterexample
Conjectured best ($\sqrt{19-1}+1$)
rewards
iterations

Comparison



The End

Conclusions

- Reinforcement learning has proven itself to be a good tool to finding examples in spectral graph theory.

The End

Conclusions

- Reinforcement learning has proven itself to be a good tool to finding examples in spectral graph theory.
- GNNs open up new avenues.

The End

Conclusions

- Reinforcement learning has proven itself to be a good tool to finding examples in spectral graph theory.
- GNNs open up new avenues.
- We need to exploit the mathematical structure in the data representation.

The End

Conclusions

- Reinforcement learning has proven itself to be a good tool to finding examples in spectral graph theory.
- GNNs open up new avenues.
- We need to exploit the mathematical structure in the data representation.

Advertisement: We are organising an AI for Maths conference this summer at Imperial. Details: