Symmetry and Structure in Single and Multi-Agent RL and AI4Science

MILA Geometric Deep Learning Reading Group
02 Feb 2023

Elise van der Pol

Universiteit van Amsterdam
UvA - Bosch DELTA LAB
Microsoft

About me

Senior Researcher @ Microsoft Research AI4Science Amsterdam:

Reinforcement Learning & Deep Learning for Molecular Simulation

Before: PhD in Machine Learning @ AMLab, University of Amsterdam

Symmetry and Structure in Reinforcement Learning

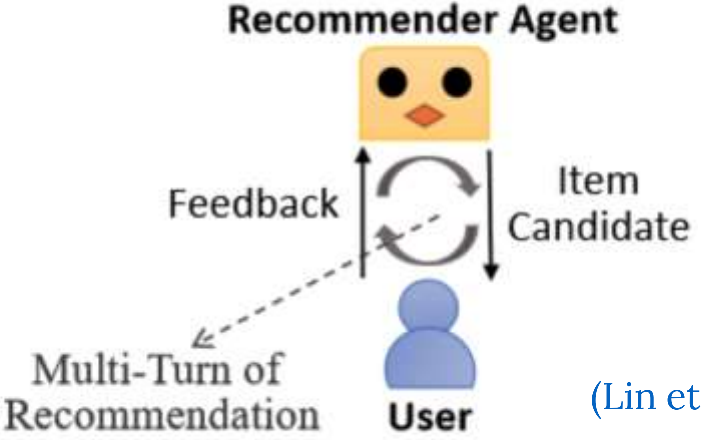
Find me online:

evanderpol@microsoft.com

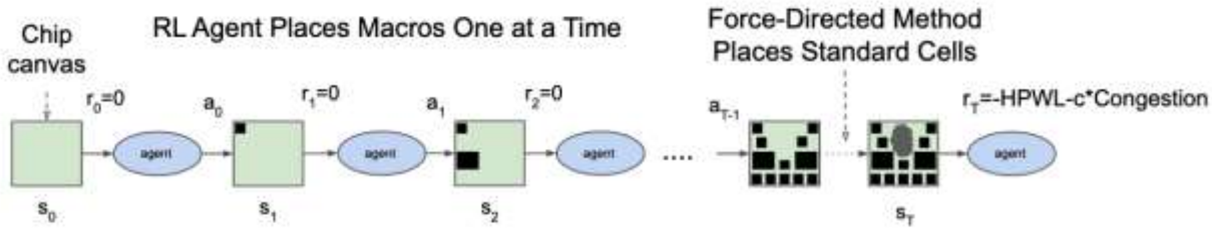
@ElisevanderPol

elisevanderpol.nl

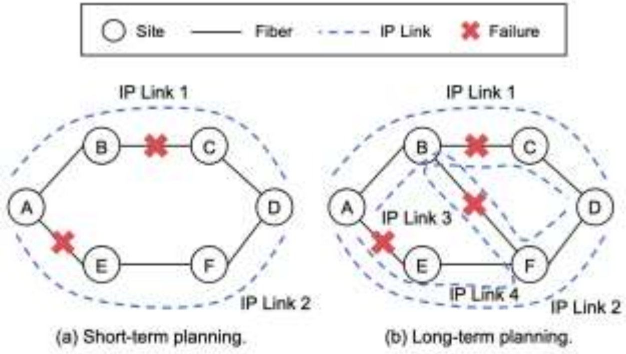
@elisevanderpol@sigmoid.social



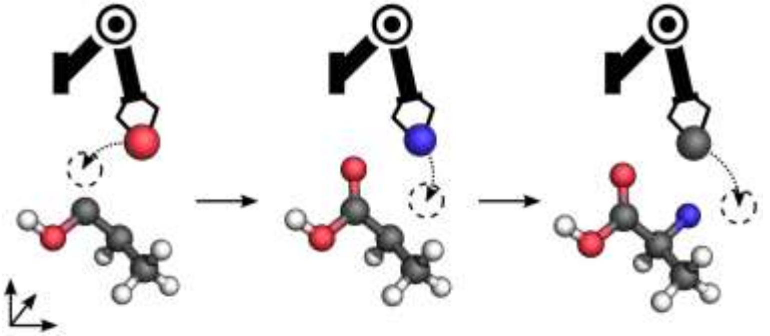
(Lin et al 2020)



Chip Placement with Deep Reinforcement Learning
(Mirhoseini et al., 2020)

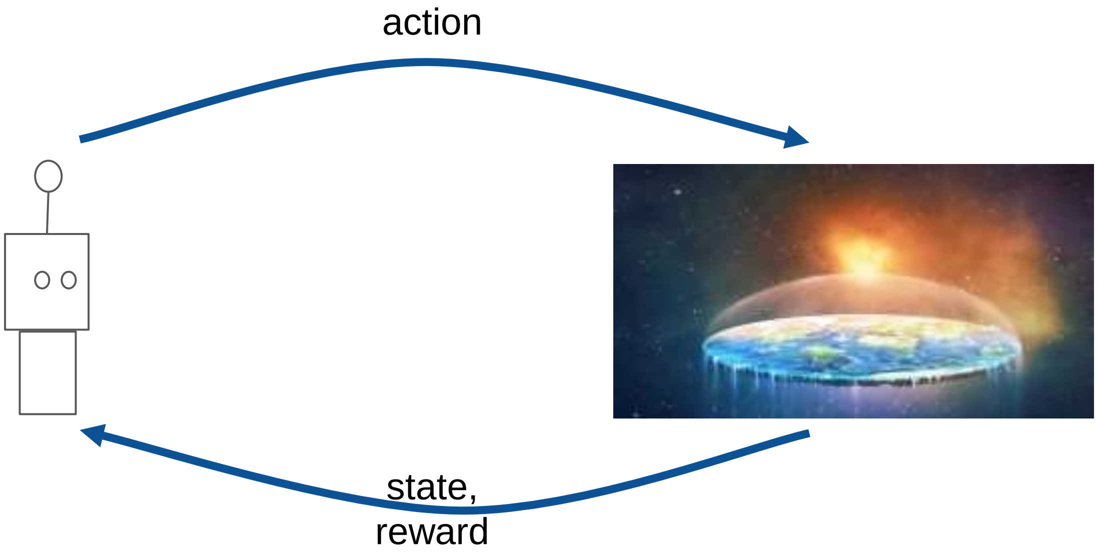


Network planning with deep reinforcement learning
(Zhu et al., 2020)



Reinforcement learning for molecular design guided by quantum mechanics (Simm et al., 2020)

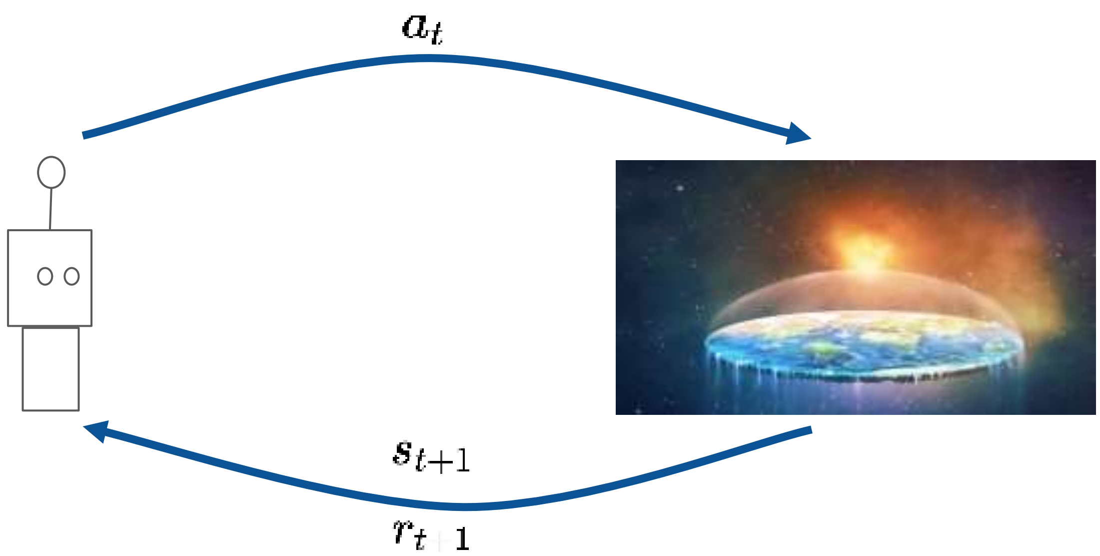
Reinforcement Learning



Learning from trial and error

Reinforcement Learning

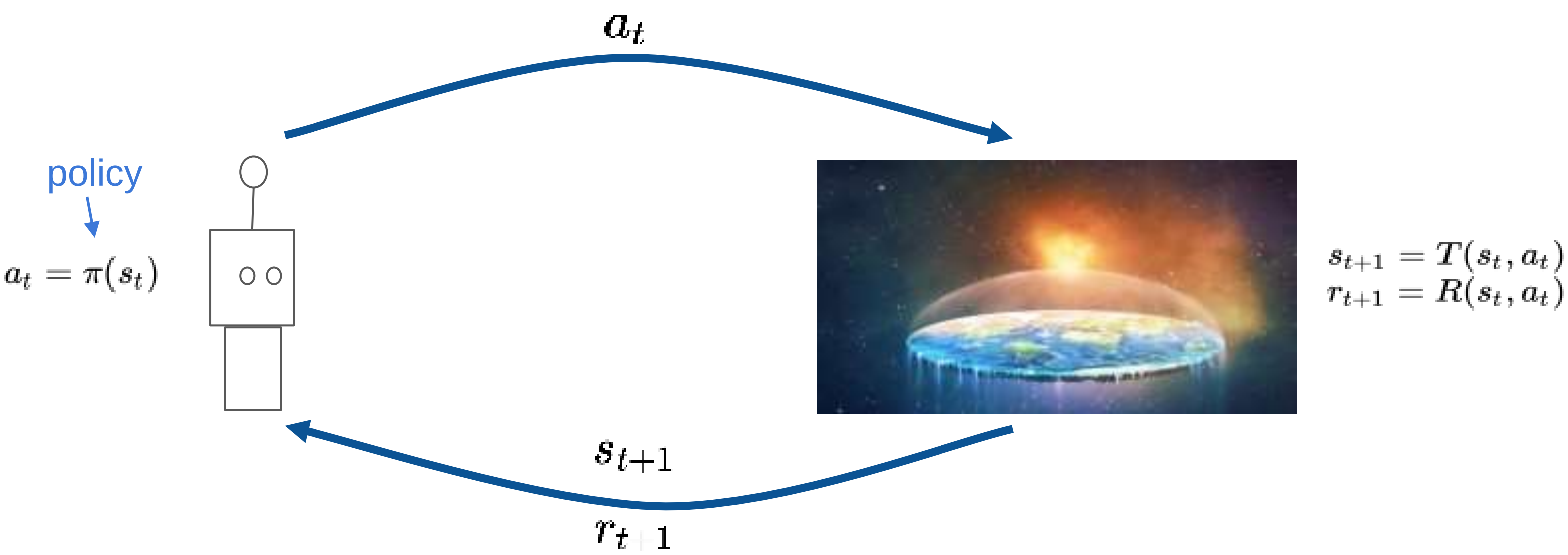
$a_t = \pi(s_t)$



$s_{t+1} = T(s_t, a_t)$
$r_{t+1} = R(s_t, a_t)$

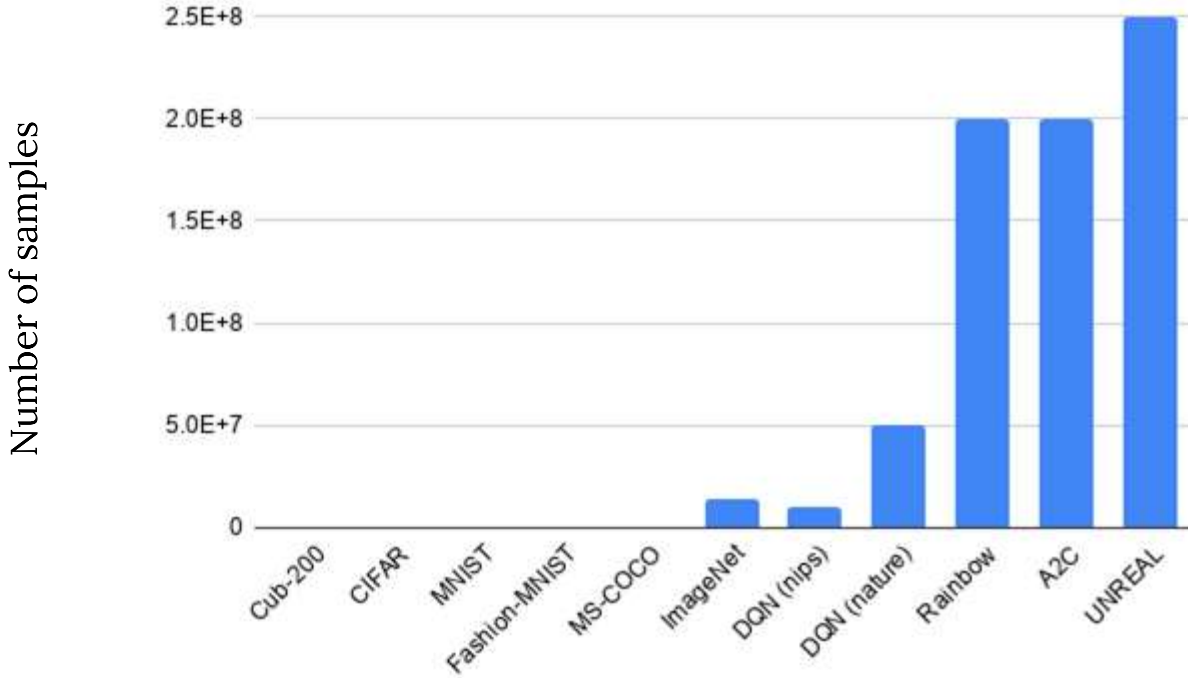
Markov Decision Process (MDP): $(\mathcal{S}, \mathcal{A}, T, R)$

Reinforcement Learning



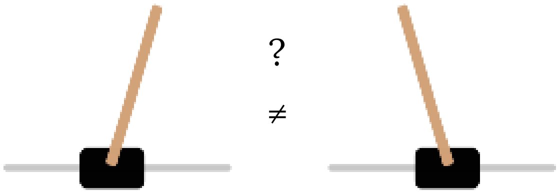
Goal: Policy that maximizes cumulative reward

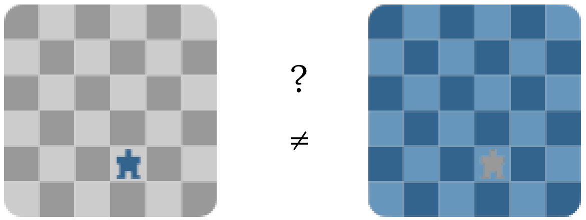
Reinforcement learning is very data hungry

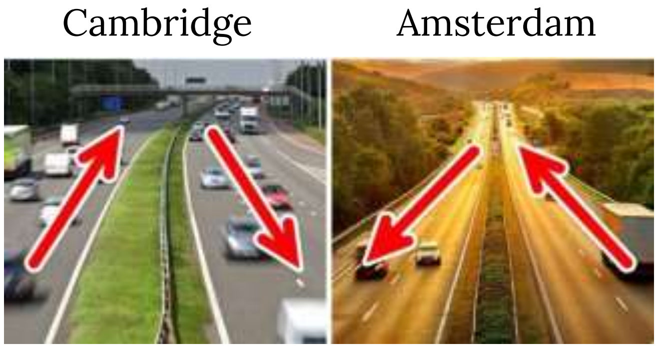


Sampling is slow and there is a real world cost to low reward states

Not all data is unique!



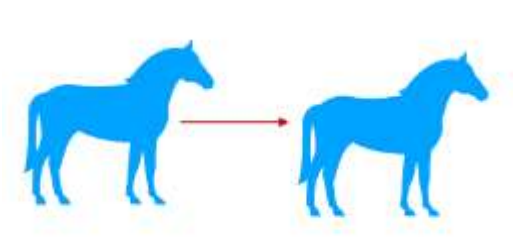




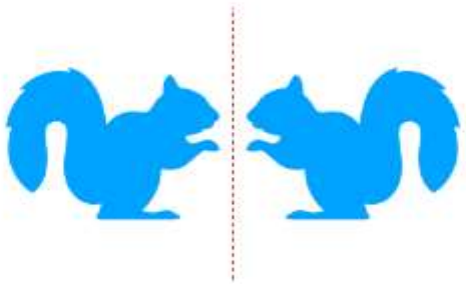
Some Background

What is a group?

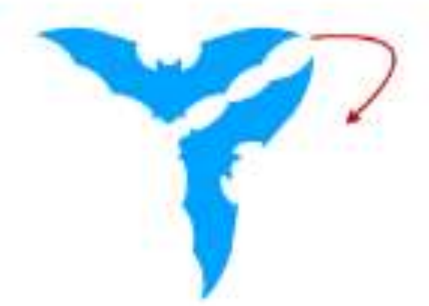
Examples:



Translations



Reflections



Rotations

A set with a binary operation obeying the group axioms

(identity, invertibility, closure, associativity)

Figures from Daniel Worrall's MLSS slides

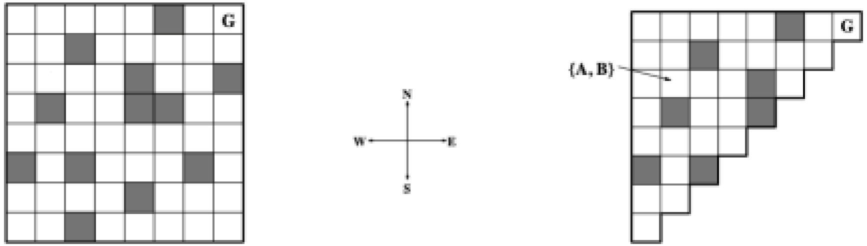
Symmetries in Reinforcement Learning

For all states and actions, and all group elements:

Dynamics are invariant under group transformations

$$R(s,a) = R(gs, ga)$$

$$T(s'|s,a) = T(gs'|gs,ga)$$



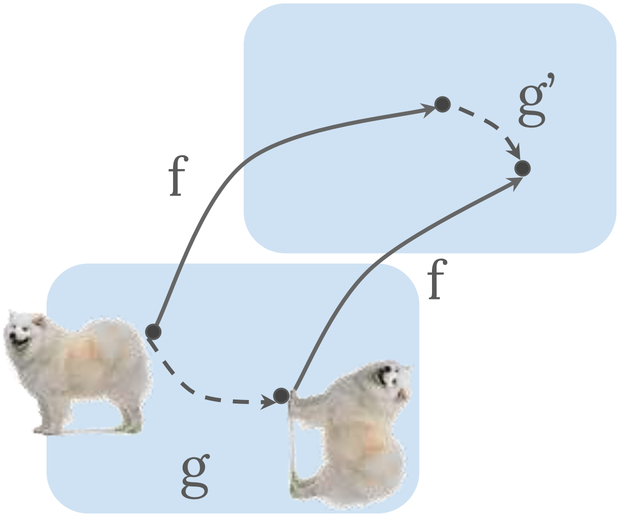
(a) (b)

(Ravindran & Barto 2004)

(s,a) and (gs,ga) are symmetric state-action pairs and have the same π^*

Equivariance

$$f(gx) = g' f(x)$$



Figures adapted from Daniel E. Worrall

MDP Homomorphic Networks: Group Symmetries in Reinforcement Learning

Elise van der Pol · Daniel E. Worrall · Herke van Hoof · Frans A. Oliehoek · Max Welling

Universiteit van Amsterdam

UvA - BOSCH DELTA LAB

TU Delft
Delft University of Technology

CIFAR

Neural Information Processing Systems

Homomorphism

Structure-preserving map between similar algebraic structures such that

$$f(x \cdot y) = f(x) \cdot f(y)$$

Examples:

Linear map between vector spaces

$$T(v + w) = T(v) + T(w)$$

Exponential function between the reals and the positive reals

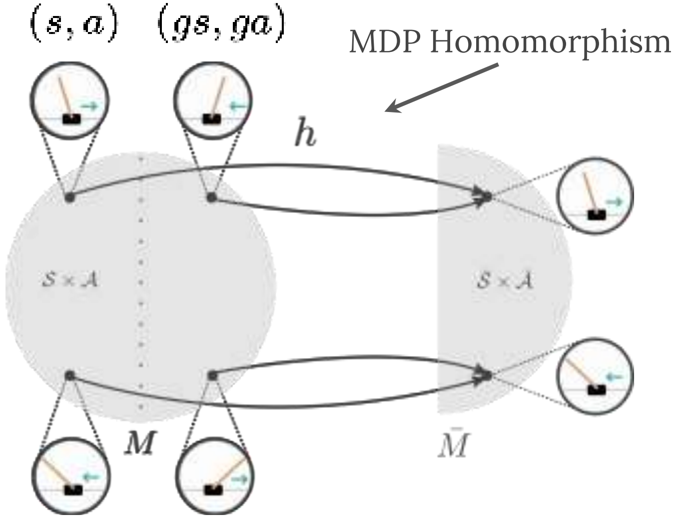
$$e^{x+y} = e^x e^y$$

Group representation between a group and the general linear group

$$\rho(g_1 g_2) = \rho(g_1)\rho(g_2)$$

MDP Homomorphisms

Map ground MDP → abstract MDP, preserve dynamics (Ravindran & Barto 2001)

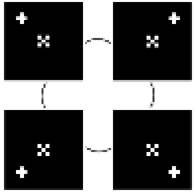


Ground/Original MDP Abstract/Reduced MDP

(s, a) and (gs, ga) are symmetric state-action pairs and have the same π^*

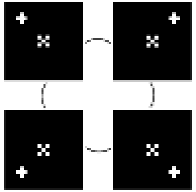
MDP Homomorphic Networks:
Group Symmetries in Reinforcement Learning

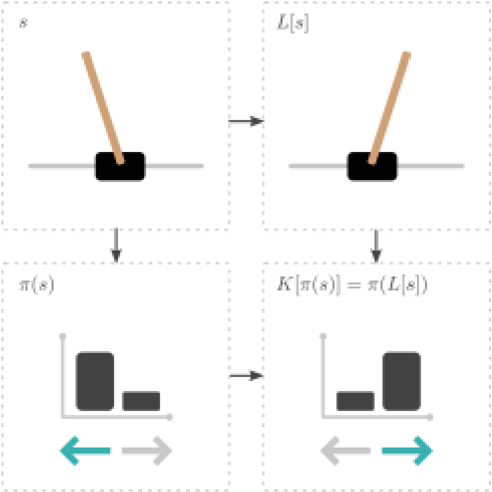
(van der Pol, Worrall, van Hoof, Oliehoek & Welling, NeurIPS 2020)



MDP Homomorphic Networks:
Group Symmetries in Reinforcement Learning

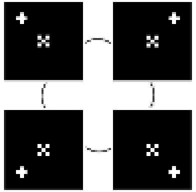
(van der Pol, Worrall, van Hoof, Oliehoek & Welling, NeurIPS 2020)

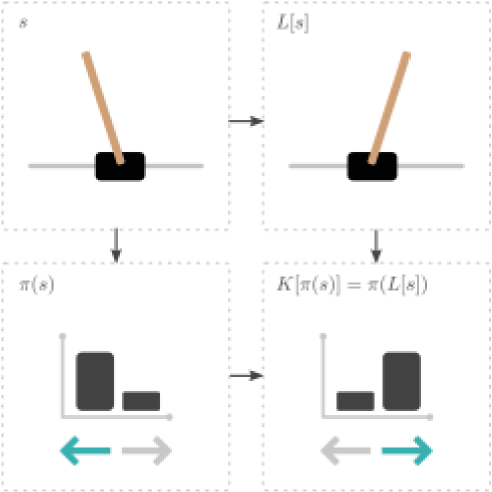




MDP Homomorphic Networks:
Group Symmetries in Reinforcement Learning

(van der Pol, Worrall, van Hoof, Oliehoek & Welling, NeurIPS 2020)





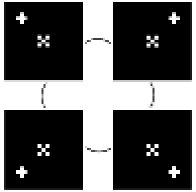
Symmetric (s, a) pairs have the same policy π :

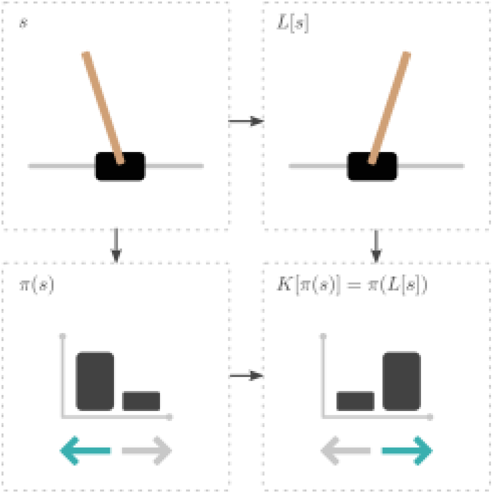
$$K[\pi(s)] = \pi(L[s])$$

L is a transformation on states, K a transformation on policies

MDP Homomorphic Networks: Group Symmetries in Reinforcement Learning

(van der Pol, Worrall, van Hoof, Oliehoek & Welling, NeurIPS 2020)





Symmetric (s, a) pairs have the same policy π :

$$K[\pi(s)] = \pi(L[s])$$

L is a transformation on states, K a transformation on policies

MDP homomorphic networks exploit symmetries in reinforcement learning

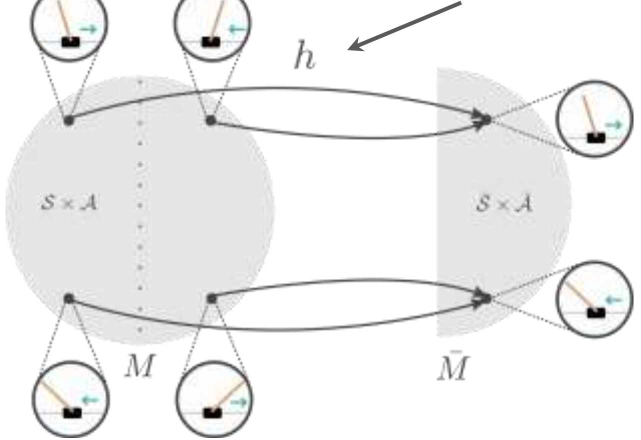
MDP Homomorphic Networks

We bridge MDP homomorphisms and equivariant networks

MDP Homomorphic Networks

We bridge MDP homomorphisms and equivariant networks

MDP Homomorphism

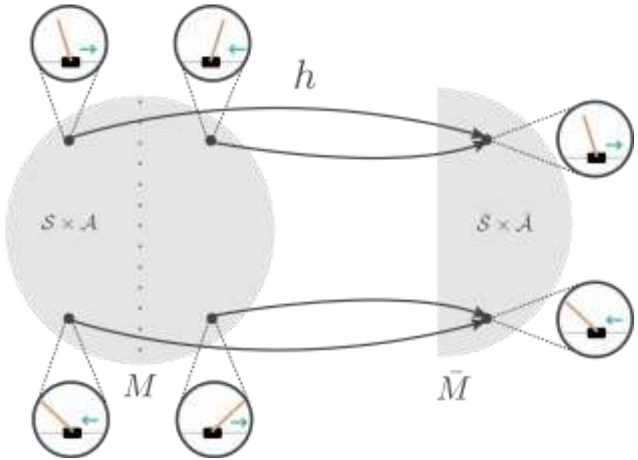


Problem with symmetries

Reduced problem

MDP Homomorphic Networks

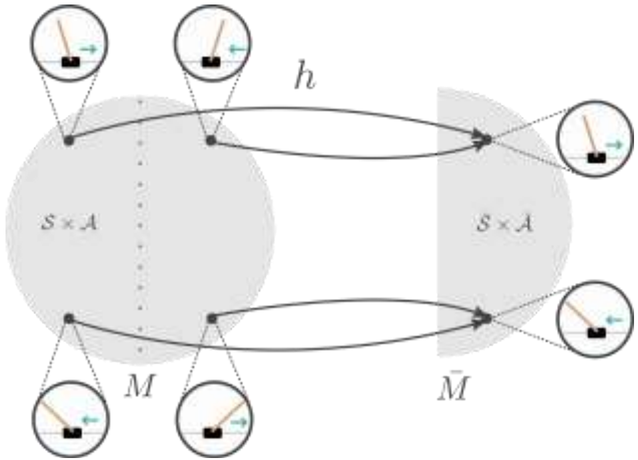
We bridge MDP homomorphisms and equivariant networks



We create deep networks constrained by MDP homomorphisms that enforce equivariance

MDP Homomorphic Networks

We bridge MDP homomorphisms and equivariant networks

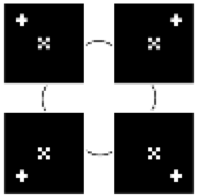


We create deep networks constrained by MDP homomorphisms that enforce equivariance

We introduce a new method, the Symmetrizer, to construct equivariant weights

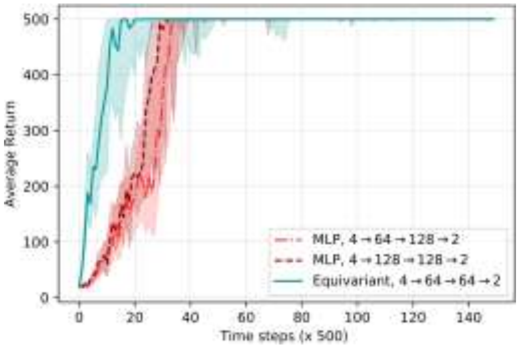
MDP Homomorphic Networks

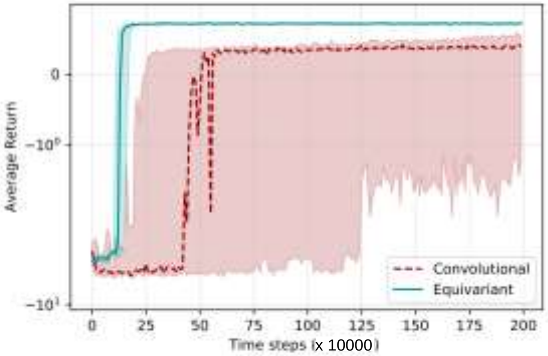
Cartpole
2 element symmetry group

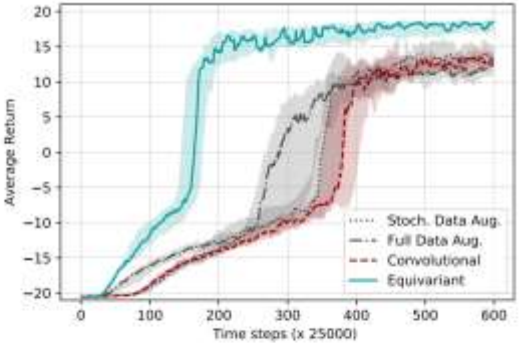


Grid World
4 element symmetry group

Pong
2 element symmetry group







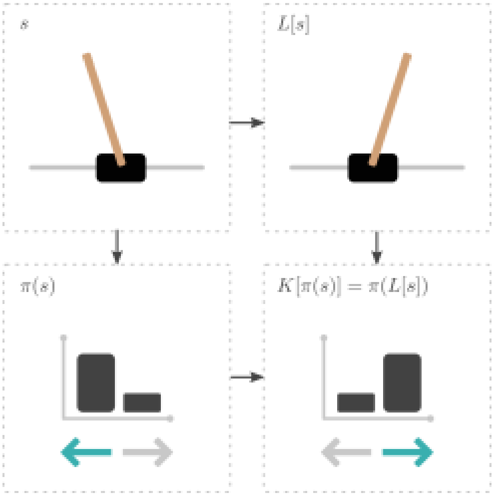
Fewer interactions with the world needed

Conclusion

Fewer interactions needed to obtain good policies with MDP Homomorphic networks

Useful in reinforcement learning problems that exhibit group symmetry

Symmetrizer: automatically constructs equivariant layers



Multi-Agent MDP Homomorphic Networks

Elise van der Pol · Herke van Hoof · Frans A. Oliehoek · Max Welling

Universiteit van Amsterdam

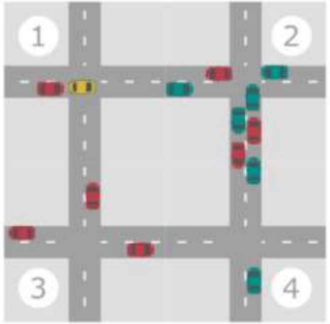
UvA - Bosch Delta Lab

TU Delft
Delft University of Technology

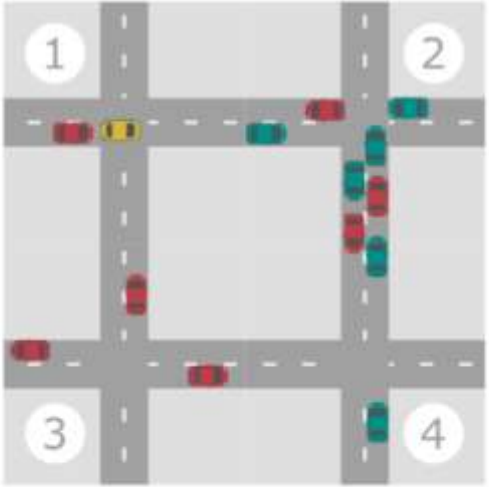
CIFAR

ICLR

Cooperative Multi-Agent Systems

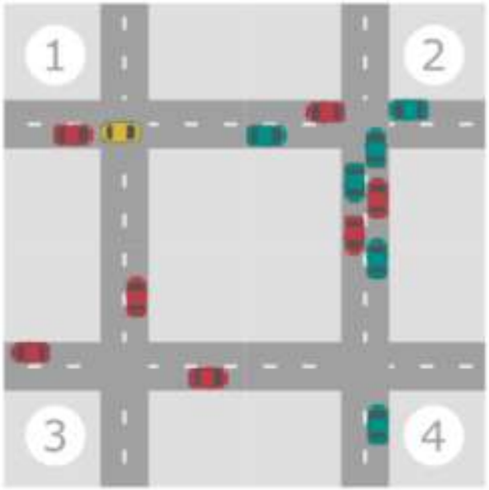


Setting: Centralized Training, Distributed execution

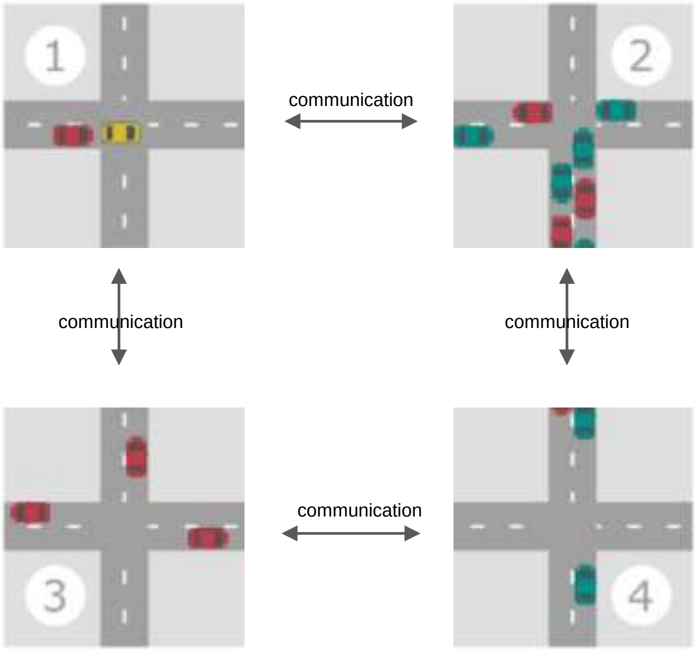


Centralized controller: puppeteer agent

Setting: Centralized Training, Distributed execution

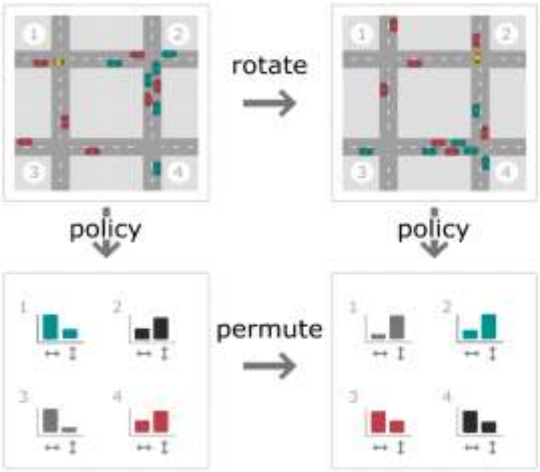


Centralized controller: puppeteer agent



Distributed controllers: local

Global symmetries in multi-agent decision problems

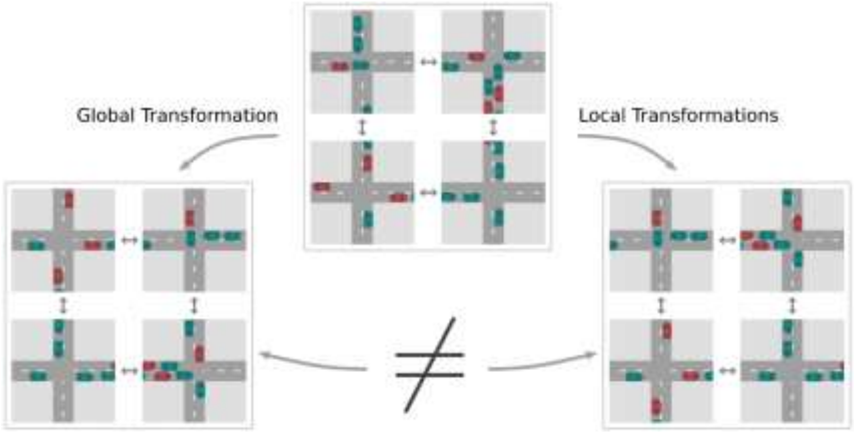


$$\vec{\pi}_\theta(L_g[\mathbf{s}]) = H_g[\vec{\pi}_\theta(\mathbf{s})]$$

Equivariance constraint on global states and joint policies.

Global symmetries in multi-agent systems

Equivariance to group symmetries: successful in single agent RL



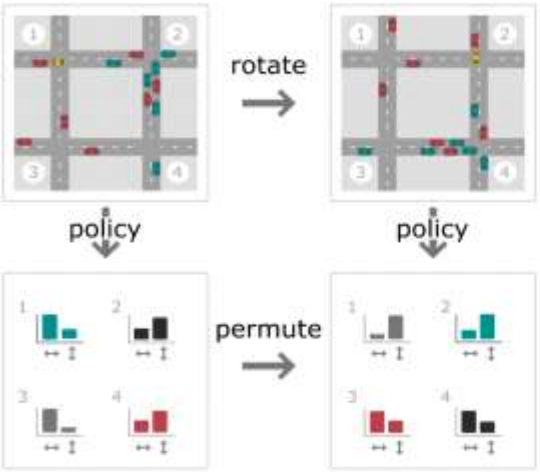
Local equivariance ≠ global equivariance

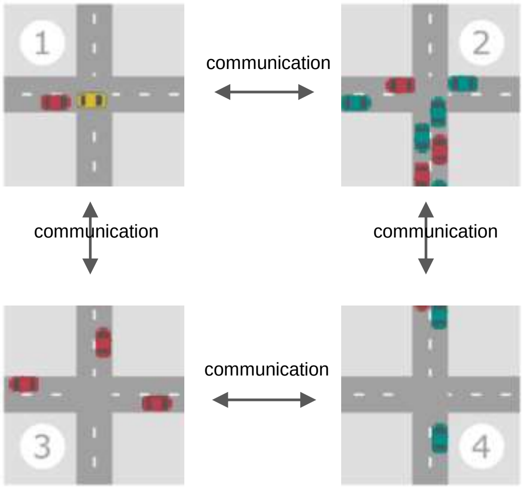
Single agent approaches: only applicable with centralized controllers

Multi-Agent MDP Homomorphic Networks

(van der Pol, van Hoof, Oliehoek & Welling, ICLR 2022)

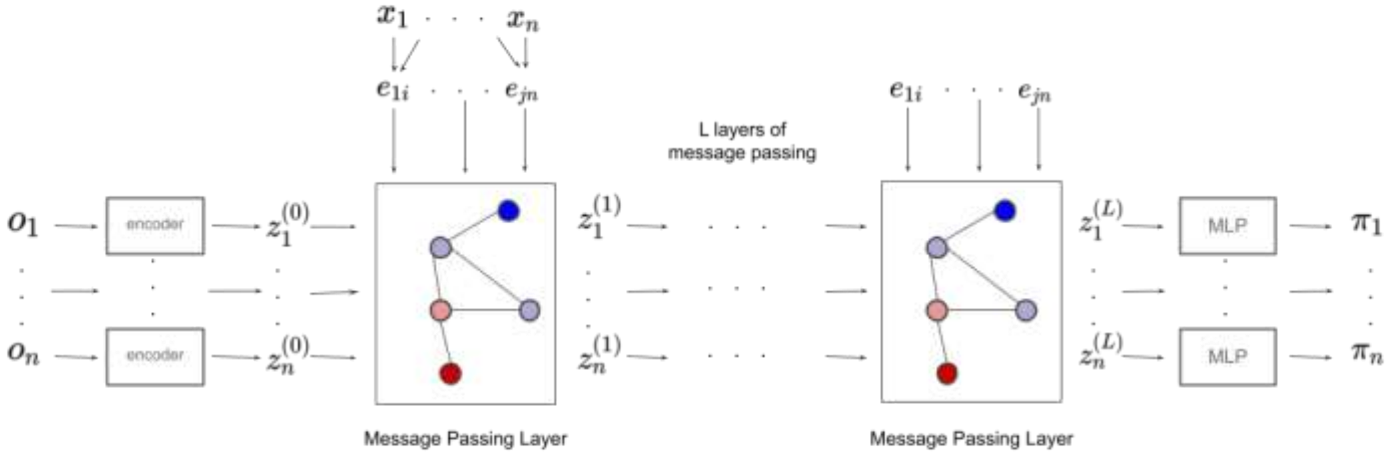
Use global symmetries with only local communication & local computation





Local equivariance constraints allow for distributed global symmetry.

Distributed Execution in Message Passing Networks



Distributed Execution in Message Passing Networks

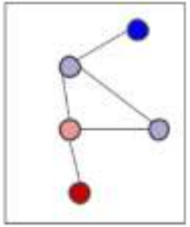
$o_1 \longrightarrow \boxed{\text{encoder}} \longrightarrow z_1^{(0)}$

Encode $z_i^{(0)} = \phi_{enc}(o_i)$

Distributed Execution in Message Passing Networks

Encode $$z_i^{(0)} = \phi_{enc}(o_i)$$

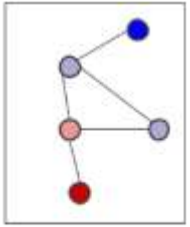
Messages $$m_{i \to j} = \phi_m(z_j^{(l)}, e_{ij})$$



Message Passing Layer

Distributed Execution in Message Passing Networks

Encode $$z_i^{(0)} = \phi_{enc}(o_i)$$



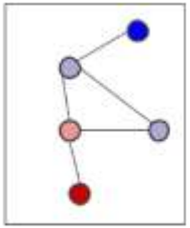
Message Passing Layer

Messages $$m_{i\to j} = \phi_m(z_j^{(l)}, e_{ij})$$

Aggregation $$m_i = \sum_{j\in \text{neighbors}} m_{j\to i}$$

Distributed Execution in Message Passing Networks

Encode $$z_i^{(0)} = \phi_{enc}(o_i)$$



Message Passing Layer

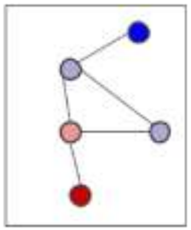
Messages $$m_{i \to j} = \phi_m(z_j^{(l)}, e_{ij})$$

Aggregation $$m_i = \sum_{j \in \text{neighbors}} m_{j \to i}$$

Node Update $$z_i^{(l+1)} = \phi_u(z_i^{(l)}, m_i)$$

Distributed Execution in Message Passing Networks

Encode $$z_i^{(0)} = \phi_{enc}(o_i)$$



Message Passing Layer

Messages $$m_{i \to j} = \phi_m(z_j^{(l)}, e_{ij})$$

Aggregation $$m_i = \sum_{j \in \text{neighbors}} m_{j \to i}$$

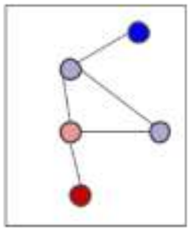
Node Update $$z_i^{(l+1)} = \phi_u(z_i^{(l)}, m_i)$$

Policy $$\pi_i = \text{MLP}_\theta(z_i^{(L)})$$

Distributed Execution in Message Passing Networks

$o_1 \longrightarrow \boxed{\text{encoder}} \longrightarrow z_1^{(0)}$

Encode $\quad z_i^{(0)} = \phi_{enc}(o_i)$



Message Passing Layer

Messages $\quad m_{i \to j} = \phi_m(z_j^{(l)}, e_{ij})$

Aggregation $\quad m_i = \sum_{j \in \text{neighbors}} m_{j \to i}$

Node Update $\quad z_i^{(l+1)} = \phi_u(z_i^{(l)}, m_i)$

$z_1^{(L)} \longrightarrow \boxed{\text{MLP}} \longrightarrow \pi_1$

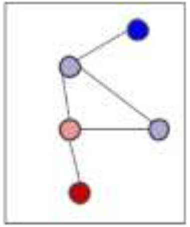
Policy $\quad \pi_i = \text{MLP}_\theta(z_i^{(L)})$

Require only local information + local communication at execution time.

Globally Equivariant Distributed Execution

$o_1 \longrightarrow \boxed{\text{encoder}} \longrightarrow z_1^{(0)}$

Encode $z_i^{(0)} = \phi_{enc}(o_i)$



Message Passing Layer

Messages $m_{i \to j} = \phi_m(z_j^{(l)}, e_{ij})$

Aggregation $m_i = \sum_{j \in \text{neighbors}} m_{j \to i}$

Node Update $z_i^{(l+1)} = \phi_u(z_i^{(l)}, m_i)$

$z_1^{(L)} \longrightarrow \boxed{\text{MLP}} \longrightarrow \pi_1$

Policy $\pi_i = \text{MLP}_\theta(z_i^{(L)})$

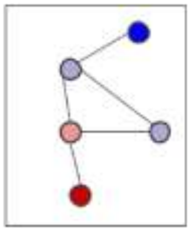
Globally Equivariant Distributed Execution

$o_1 \longrightarrow \boxed{\text{encoder}} \longrightarrow z_1^{(0)}$

Encode $\quad z_i^{(0)} = \phi_{enc}(o_i)$

$\boxed{L_g[z_i^{(0)}] = \phi_{enc}(R_g[o_i])}$

Equivariance constraint on encoder



Message Passing Layer

Messages $\quad m_{i\to j} = \phi_m(z_j^{(l)}, e_{ij})$

Aggregation $\quad m_i = \sum_{j\in \text{neighbors}} m_{j\to i}$

Node Update $\quad z_i^{(l+1)} = \phi_u(z_i^{(l)}, m_i)$

$z_1^{(L)} \longrightarrow \boxed{\text{MLP}} \longrightarrow \pi_1$

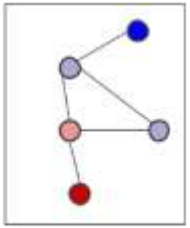
Policy $\quad \pi_i = \text{MLP}_\theta(z_i^{(L)})$

Globally Equivariant Distributed Execution

Encode $\quad z_i^{(0)} = \phi_{enc}(o_i)$

$$\boxed{L_g[z_i^{(0)}] = \phi_{enc}(R_g[o_i])}$$

Equivariance constraint on encoder



Message Passing Layer

Messages $\quad m_{i \to j} = \phi_m(z_j^{(l)}, e_{ij})$

Equivariance constraint on messages

$$\boxed{L_g[m_i^{(l)}] = \sum_{j \in \text{neighbors}} \phi_m(L_g[z_j^{(l)}], U_g[e_{ij}])}$$

Aggregation $\quad m_i = \sum_{j \in \text{neighbors}} m_{j \to i}$

Node Update $\quad z_i^{(l+1)} = \phi_u(z_i^{(l)}, m_i)$

Policy $\quad \pi_i = \text{MLP}_\theta(z_i^{(L)})$

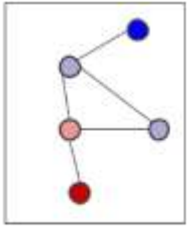
Globally Equivariant Distributed Execution

$o_1 \longrightarrow \boxed{\text{encoder}} \longrightarrow z_1^{(0)}$

Encode $\quad z_i^{(0)} = \phi_{enc}(o_i)$

$$\boxed{L_g[z_i^{(0)}] = \phi_{enc}(R_g[o_i])}$$

Equivariance constraint on encoder



Message Passing Layer

Messages $\quad m_{i\to j} = \phi_m(z_j^{(l)}, e_{ij})$

Equivariance constraint on messages

$$\boxed{L_g[m_i^{(l)}] = \sum_{j\in \text{neighbors}} \phi_m(L_g[z_j^{(l)}], U_g[e_{ij}])}$$

Aggregation $\quad m_i = \sum_{j\in \text{neighbors}} m_{j\to i}$

$$\boxed{P_g[\phi_u(z_i^{(l)}, m_i^{(l)}) = \phi_u(L_g[z_i^{(l)}], L_g[m_i^{(l)}])}$$

Node Update $\quad z_i^{(l+1)} = \phi_u(z_i^{(l)}, m_i)$

Equivariance constraint on node updates

$z_1^{(L)} \longrightarrow \boxed{\text{MLP}} \longrightarrow \pi_1$

Policy $\quad \pi_i = \text{MLP}_\theta(z_i^{(L)})$

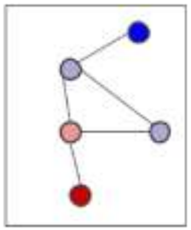
Globally Equivariant Distributed Execution

Encode

$$z_i^{(0)} = \phi_{enc}(o_i)$$

$$\boxed{L_g[z_i^{(0)}] = \phi_{enc}(R_g[o_i])}$$

Equivariance constraint on encoder



Message Passing Layer

Messages

$$m_{i \to j} = \phi_m(z_j^{(l)}, e_{ij})$$

Equivariance constraint on messages

$$\boxed{L_g[m_i^{(l)}] = \sum_{j \in \text{neighbors}} \phi_m(L_g[z_j^{(l)}], U_g[e_{ij}])}$$

Aggregation

$$m_i = \sum_{j \in \text{neighbors}} m_{j \to i}$$

$$\boxed{P_g[\phi_u(z_i^{(l)}, m_i^{(l)}) = \phi_u(L_g[z_i^{(l)}], L_g[m_i^{(l)}])}$$

Node Update

$$z_i^{(l+1)} = \phi_u(z_i^{(l)}, m_i)$$

Equivariance constraint on node updates

Policy

$$\pi_i = \text{MLP}_\theta(z_i^{(L)})$$

$$\boxed{\pi_i(L_g[z_i^{(L)}]) = P_g[\pi_i(z_i^{(L)})]}$$

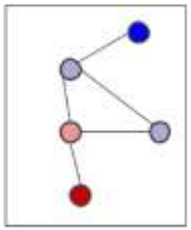
Equivariance constraint on local policies

Globally Equivariant Distributed Execution

Encode $z_i^{(0)} = \phi_{enc}(o_i)$

$$\boxed{L_g[z_i^{(0)}] = \phi_{enc}(R_g[o_i])}$$

Equivariance constraint on encoder



Message Passing Layer

Messages $m_{i\to j} = \phi_m(z_j^{(l)}, e_{ij})$

Equivariance constraint on messages

$$\boxed{L_g[m_i^{(l)}] = \sum_{j\in\text{neighbors}} \phi_m(L_g[z_j^{(l)}], U_g[e_{ij}])}$$

Aggregation $m_i = \sum_{j\in\text{neighbors}} m_{j\to i}$

$$\boxed{P_g[\phi_u(z_i^{(l)}, m_i^{(l)}) = \phi_u(L_g[z_i^{(l)}], L_g[m_i^{(l)}])}$$

Node Update $z_i^{(l+1)} = \phi_u(z_i^{(l)}, m_i)$

Equivariance constraint on node updates

Policy $\pi_i = \text{MLP}_\theta(z_i^{(L)})$

$$\boxed{\pi_i(L_g[z_i^{(L)}]) = P_g[\pi_i(z_i^{(L)})]}$$

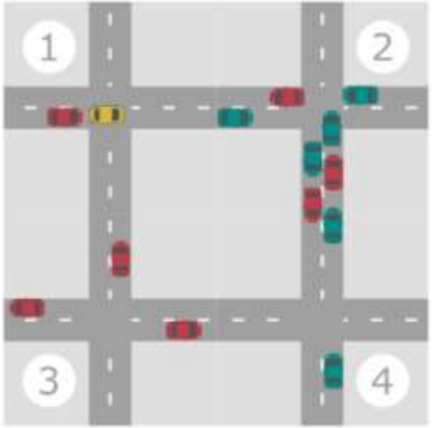
Equivariance constraint on local policies

Local equivariance constraints allow for distributed global symmetry.

Experiments

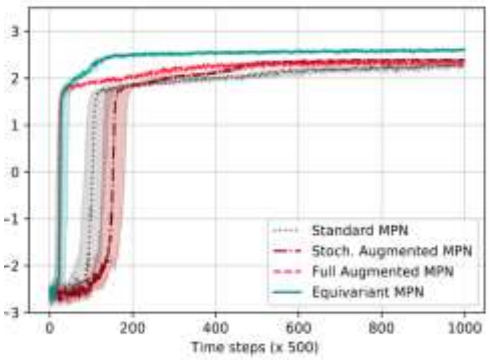
Wildlife monitoring

Traffic Light Control

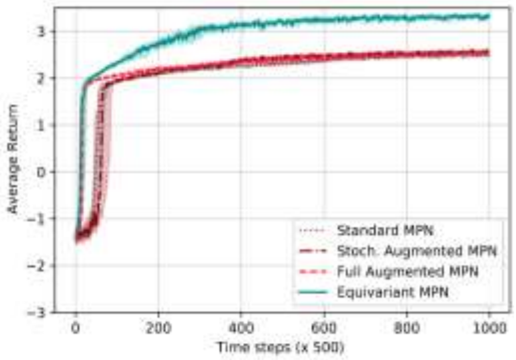


Evaluation

Wildlife monitoring (↑)
(grid world coordination)

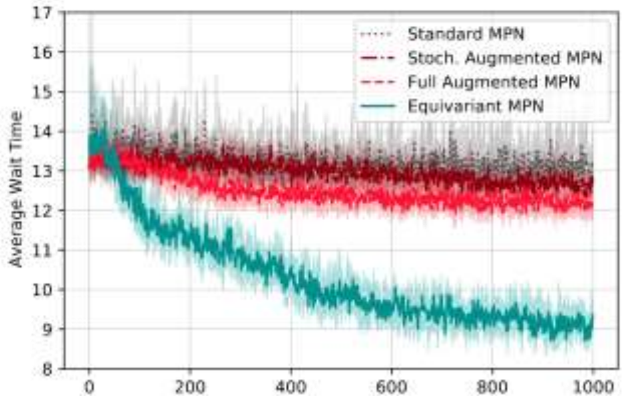


(a) 3 agents.



(b) 4 agents.

Traffic Light Control (↓)
(coordination, congestion)



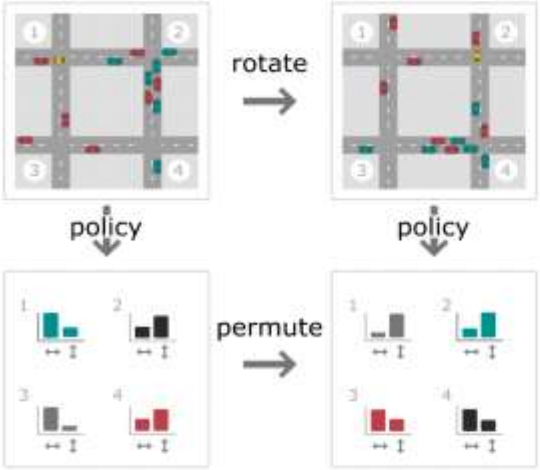
Multi-agent MDP Homomorphic Networks: improved data efficiency

Symmetry by equivariance improves over symmetry by augmentation

Conclusion

Global symmetry equivariance with only local communication & local computation

Including symmetry information helps with data efficiency, especially equivariance



Equivariant Networks for Zero-Shot Coordination





Darius Muglich · Christian Schroeder de Witt · Elise van der Pol · Shimon Whiteson · Jakob Foerster

UNIVERSITY OF OXFORD

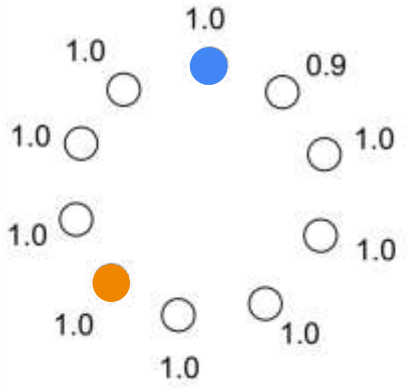
FLAIR

Microsoft

NEURAL INFORMATION PROCESSING SYSTEMS

Lever Game

Payoff if you and random, unseen partner choose same lever
Which lever should you choose?



Problem of mutually incompatible symmetry-breaking

Which lever to choose if you don’t know your partner’s decision making strategy?
(+ you cannot discuss: zero-shot coordination)

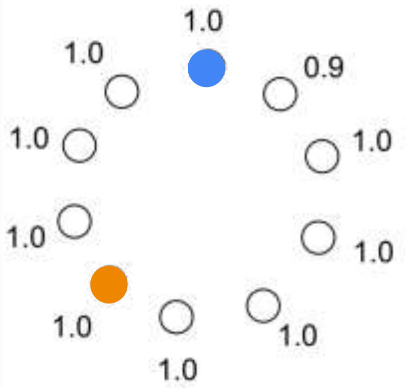
Image: Hu et al., 2020

Quick overview: Equivariant Networks for Zero-Shot Coordination

Lever game: mutually incompatible symmetry breaking in zero-shot coordination

Equivariant networks: exactly solve the symmetry breaking problem

Symmetrizing agents: empirical improvement on Hanabi challenge



Details, results, and proofs in the paper:

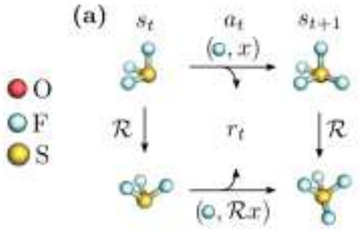
Equivariant Networks for Zero-Shot Coordination (Muglich et al., 2020)

Geometry: everywhere in decision making

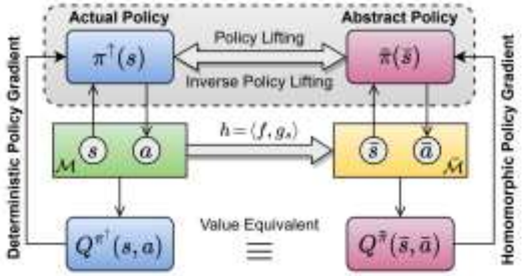
MDP homomorphic networks: **fewer interactions** needed in single and multi agent settings
→ Improve zero-shot coordination out of the box

Recent followup work & applications:

Sample Efficient Grasp Learning Using Equivariant Models
(Zhu et al., 2022)



Symmetry-Aware Actor-Critic for 3D Molecular Design
(Simm et al., 2020)



Continuous MDP Homomorphisms and Homomorphic Policy Gradient
(Rezaei-Shoshtari et al., 2022)

EqR: Equivariant Representations for Data-Efficient Reinforcement Learning
(Mondal et al., 2022)

AI4Science

Catalysis

Catalyst: substance that increases the rate of a chemical reaction without being consumed.

Catalysis: underpins 30% of the gross domestic product of the European economy [1].

Clean water

Clean air

Clean energy

Sustainable food

[1] “*Catalysis making the world a better place”,* Catlow et al, 2016, https://doi.org/10.1098/rsta.2015.0089

Computational Chemistry

Branch of theoretical chemistry: Computer simulations to solve chemistry problems

- Design new catalysts, materials, drugs, etc.
- Predict properties/outcomes of chemical reactions

Most accurate methods are only feasible for very small systems.

→ Trade-off between computational cost and accuracy

AI4Science: Machine learning approaches

Machine Learning for Chemistry

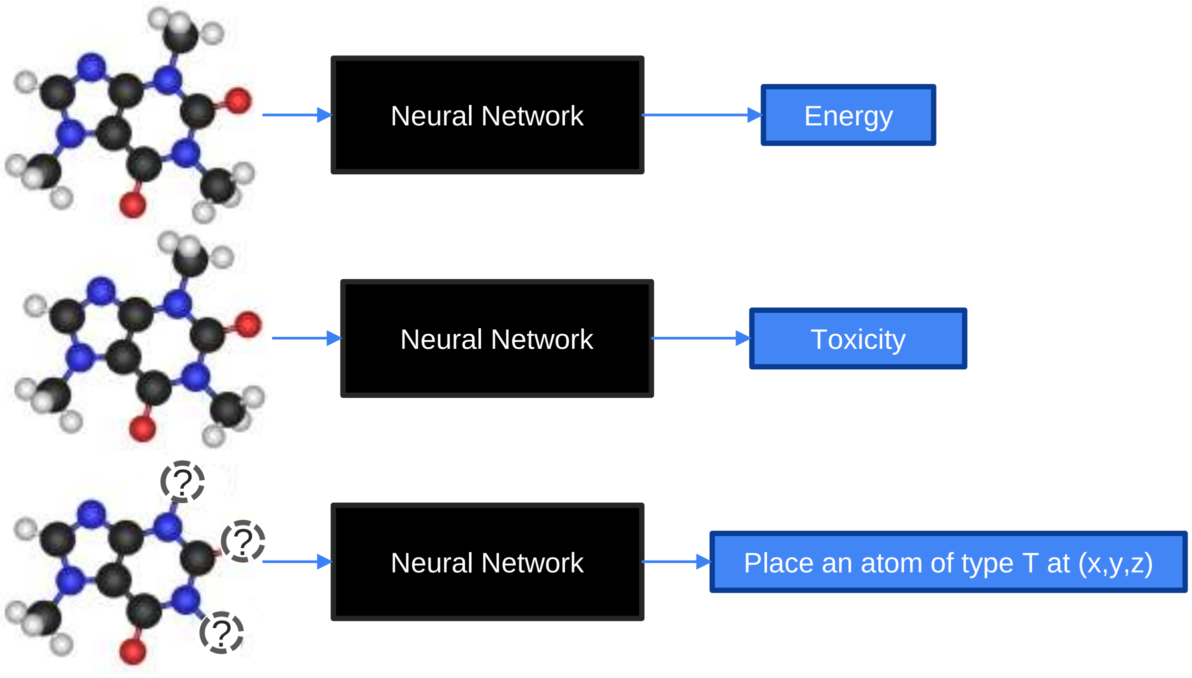
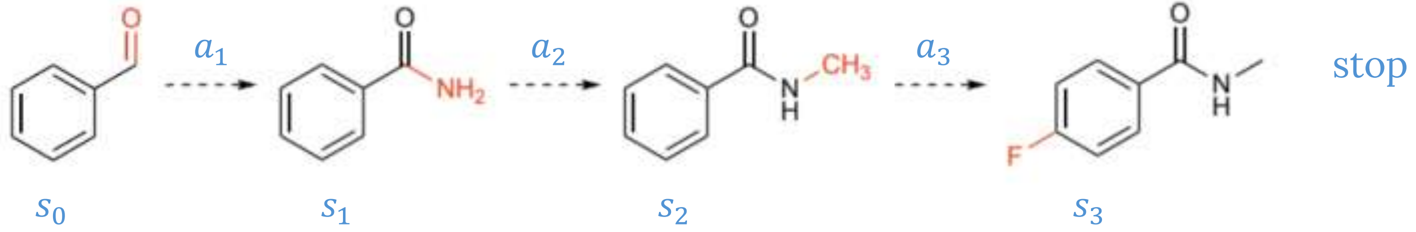


Image Marwin Segler

De Novo Design via RL/Planning



Goal: Construct Molecules with desirable properties via scoring function.

Reward: domain specific score for molecule given at stop action, else 0.
3D design: highly rotation symmetric.
Applications: drug design, batteries, catalysts.

Some work in this space:
- De Novo Drug Design Using Reinforcement Learning with Graph-Based Deep Generative Model (Atance et al., 2022)
- Molecular de-novo design through deep reinforcement learning (Olivecrona et al., 2017)
- Symmetry-aware actor-critic for 3D Molecular design (Simm et al., 2020)
- Generating Focussed Molecule Libraries for Drug Discovery with Recurrent Neural Networks (Segler et al., 2017)

Retrosynthesis Planning

Nice to be able to design molecules in silico!

But – no use if we cannot synthesize them!

Goal:
Finding synthesis routes for target molecules

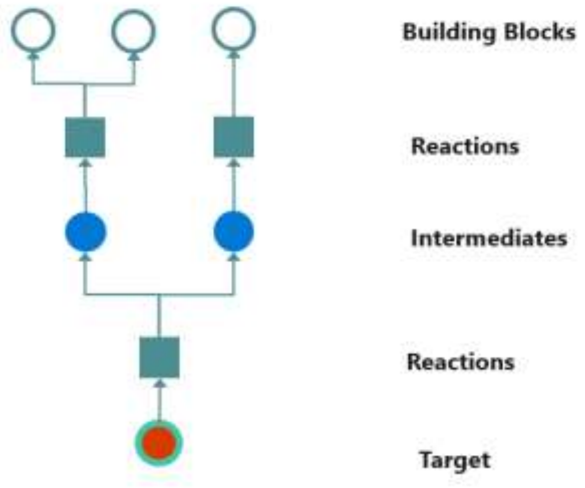
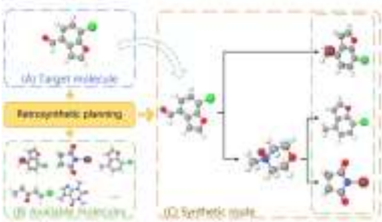


Image Austin Tripp

Retrosynthesis planning



RetroGraph: Retrosynthetic Planning with Graph Search (Xie et al., 2022)

Goal: Iteratively find buyable building blocks that react to desired product

Reward: successful route, minimizing cost, avoiding uncertain reactions, balanced route (branching vs linear), certain kinds of chemistry

Transitions: assumed known, but are they?

Some work in this space:
- Planning chemical syntheses with deep neural networks and symbolic AI (Segler et al., 2018)
- GRASP: Navigating Retrosynthetic Planning with Goal-driven Policy (Yu et al., 2022)
- RetroGraph: Retrosynthetic Planning with Graph Search (Xie et al., 2022)

Potential Energy Surface

- Surface described by a function $f: \mathbb{R}^{3N} \rightarrow \mathbb{R}$
- Input: 3D coordinates of N atoms
- Output: an energy $f(x_1, x_2, \cdots, x_N) = E$

Local minima: stable configurations

Symmetric:
rotating the molecule → same energy

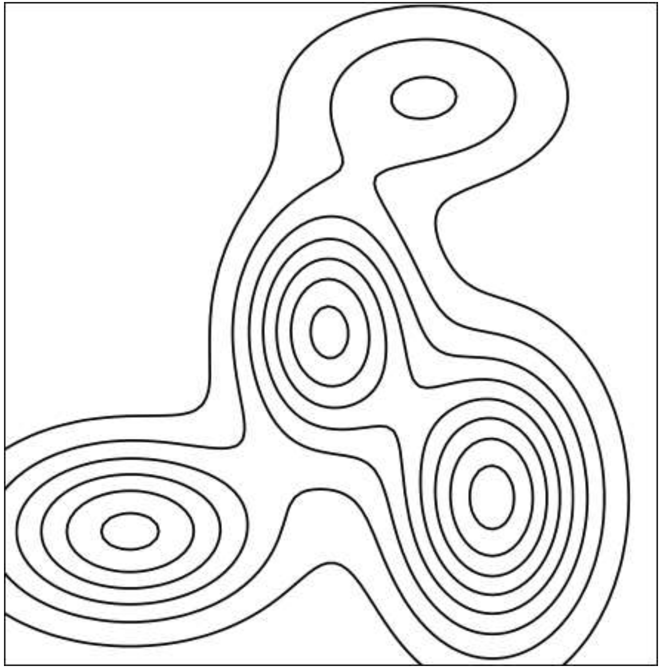


Image: Gregor Simm

Reaction modelling & prediction

1. Modelling reaction mechanisms:
    - Given a system of N atoms (point cloud of atoms in 3D)
    - Find all the **bond-changing** stable configurations

2. Predicting reaction performance:
    - Given stable start and stable end configuration, predict reaction rate

Exploring Reaction Networks

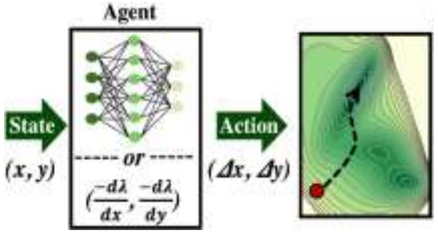


Figure: Mills et al 2022

Goal: Find all relevant minima of a potential energy surface

Reward: positive for interesting new minimum, 0 otherwise

Applications: reaction modelling e.g. in catalysis
3D point cloud: highly symmetric

Some work in this space:

- Exploring Potential Energy Surfaces Using Reinforcement Machine Learning (Mills et al., 2022)
- Discovering Catalytic Reaction Networks Using Deep Reinforcement Learning from First-Principles (Lan & An, 2021)
- Deep reinforcement learning for predicting kinetic pathways to surface reconstruction in a ternary alloy (Yoon et al., 2021)

Challenges in “Decision Making 4Science”

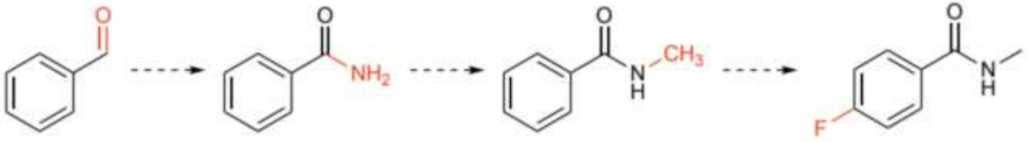
- Continuous, high-dimensional state-action spaces
- Usually, we want generalization between systems
- It’s often difficult to specify the MDP
- We don’t usually have access to the real world
- Simulators are often inaccurate or expensive (sometimes both)
- Data efficiency very important

Many interesting fundamental challenges to be solved!

Potential for real, meaningful impact

Takeaways

- Many sequential decision making problems in science
- Symmetry in decision making: improves data efficiency
- Symmetry and structure are everywhere in scientific problems
- Fundamental research with potential for meaningful impact
- Very exciting field to be working in!



https://physics4ml.github.io/

Find me online:

evanderpol@microsoft.com

@ElisevanderPol

elisevanderpol.nl

@elisevanderpol@sigmoid.social