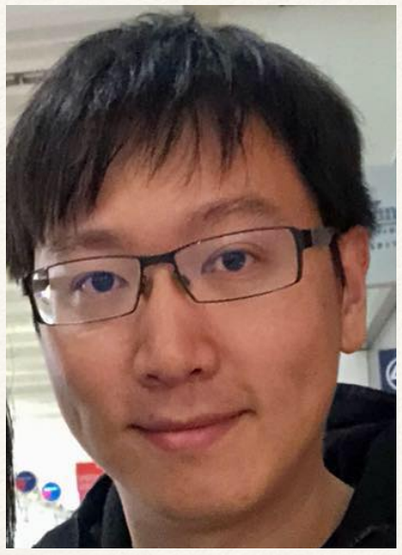
Kai-Wen Zhao



Wen-Han Kao

Kai-Hsin Wu

Generation of Ice States through RL

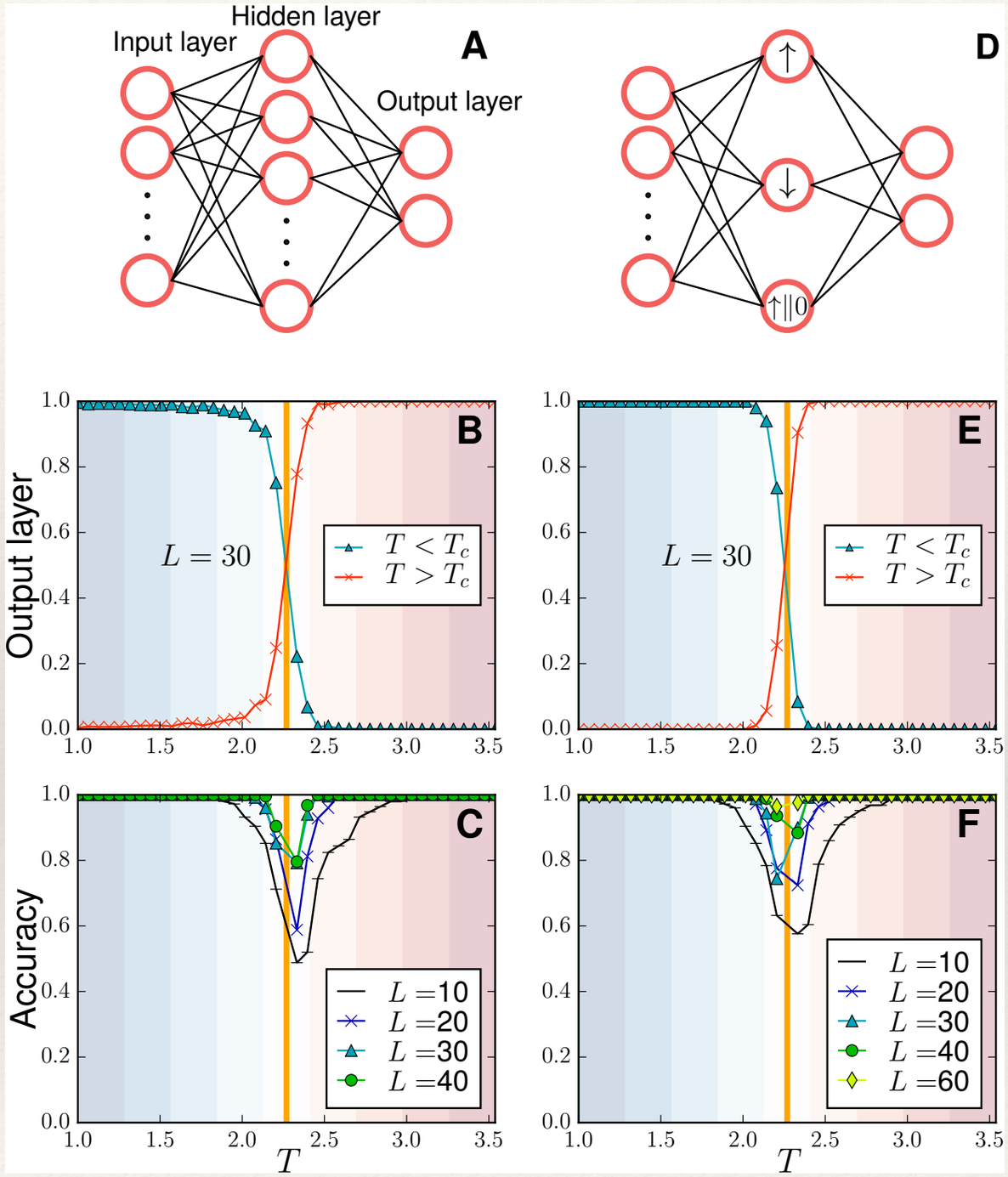
Ying-Jer Kao

Department of Physics
National Taiwan University
National Center for Theoretical Sciences
Boston University

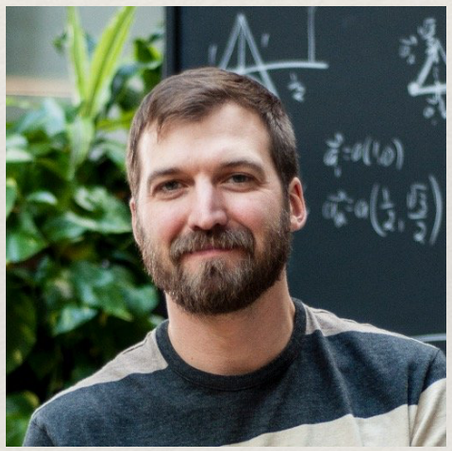
arXiv:1902.XXXX

How smart is the machine?

Can the machine figure out Physics?



Juan Carrasquilla



Roger Melko

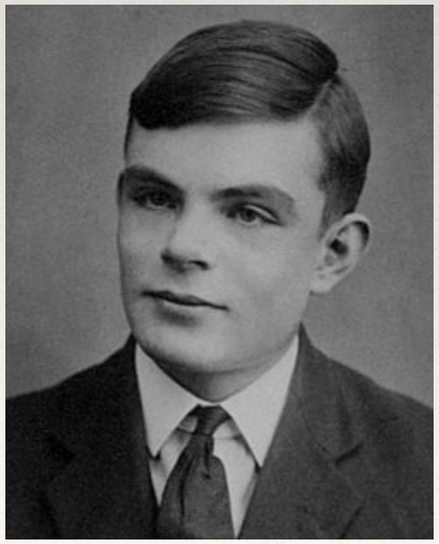
Nature Physics 13, 431–434 (2017)

https://physicsml.github.io/pages/papers.html

Machine Learning

- **Supervised Learning**
  - Classification
  - Dataset+label
- **Unsupervised Learning**
  - Generative models (RBM, VAE, GAN)
  - Dataset
- **Reinforcement Learning**
  - Model-less, reward driven
  - On-line learning
  - How human learns

“Instead of trying to produce a program to simulate the adult mind, why not rather try to produce one which simulates the child's? If this were then subjected to an appropriate course of education one would obtain the adult brain. ”



–Alan Turing

Reinforcement Learning

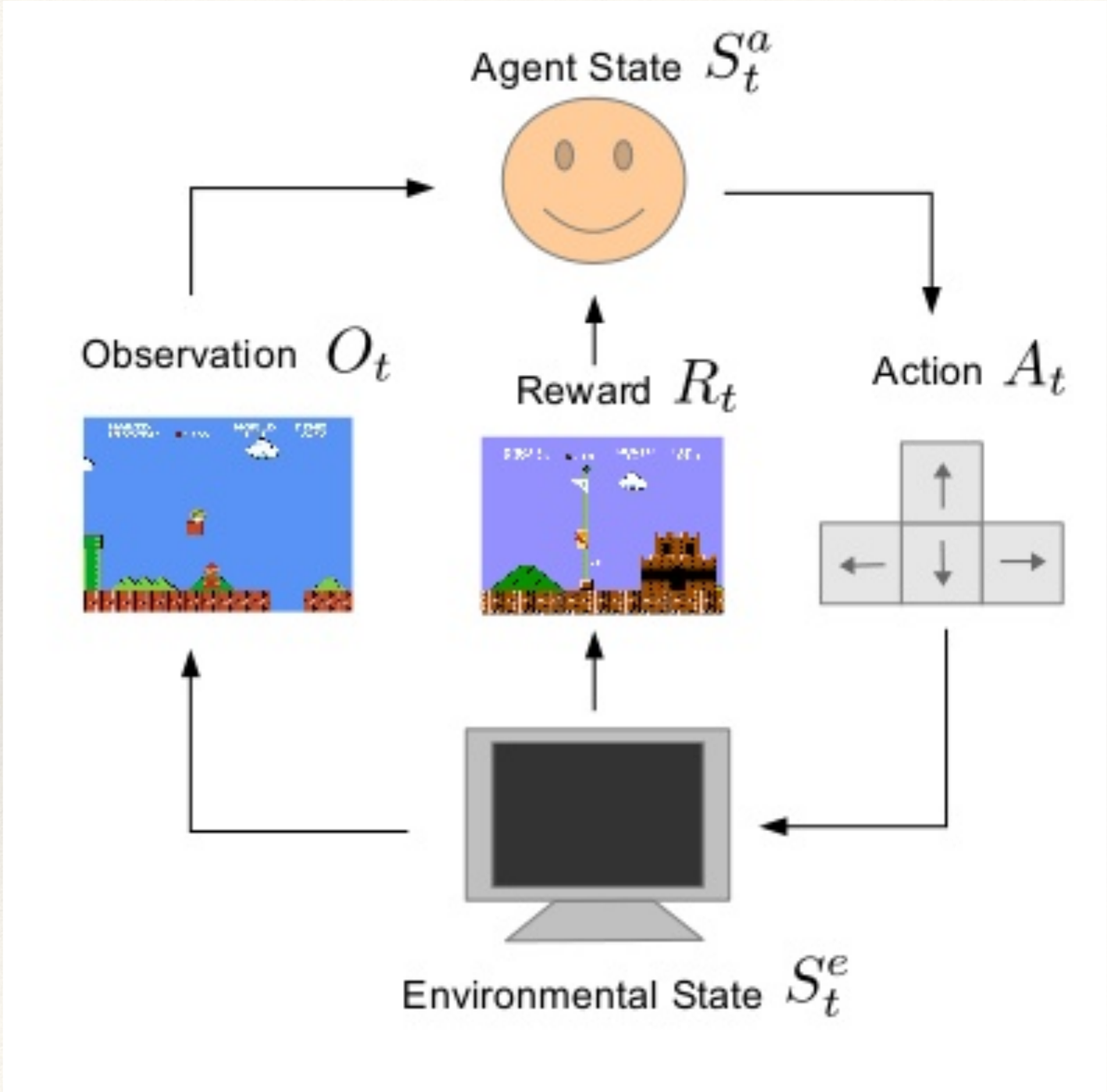
Atari games

Goal: Maximize Scores

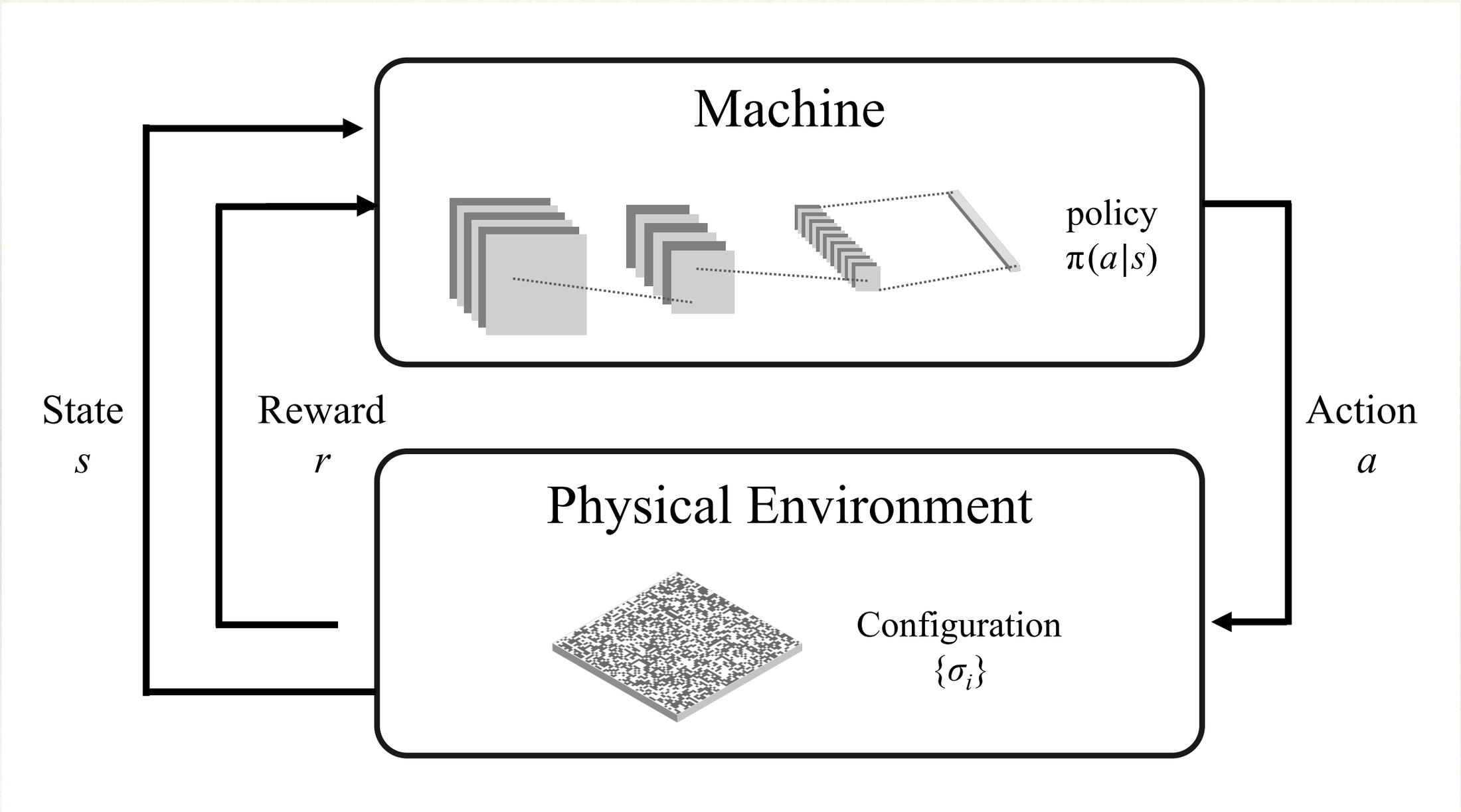
Locomotion Behavior

Goal: Maximize Moving Distance

Learning by Playing

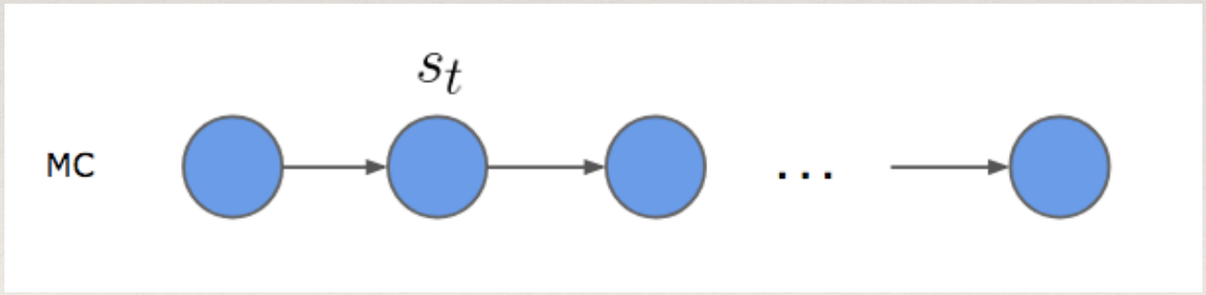


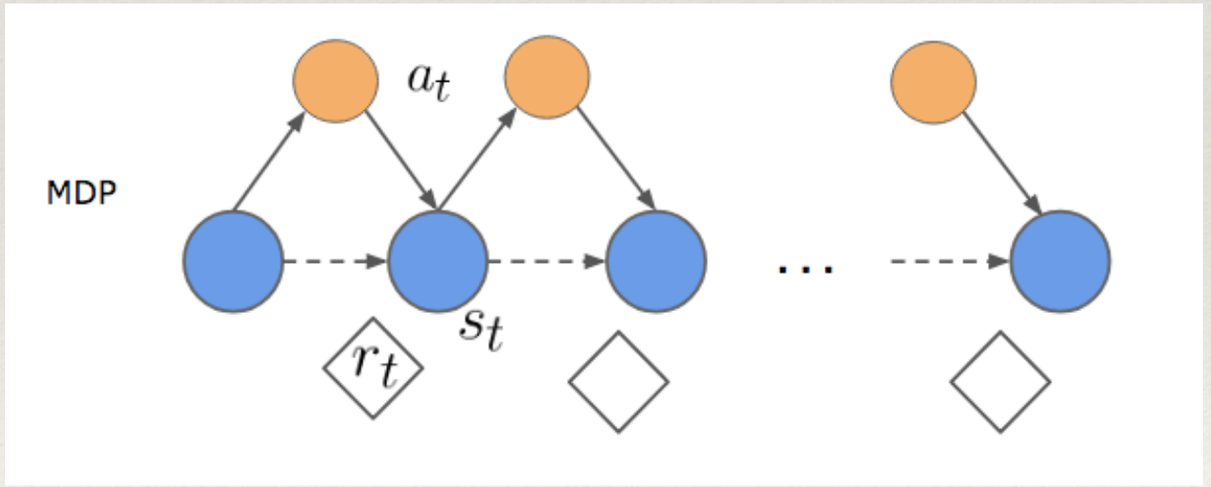
Agent as a Physicist



Markov Decision Process

- Markov Decision Process $M = \{S, T, A, r\}$.
- State Space, **S**: a set of states of environment
- Action Space, A: a set of actions Agent selects from at each time-step
- Transition operator, T: $p(s_{t+1} \mid s_t, a_t)$
- Reward function, r: $S \times A \rightarrow R$: a scalar value to characterize states





Transition Probability

$p(s_{t+1} \mid s_t)$

$\pi(a_t \mid s_t)$

$p(s_{t+1} \mid s_t, a_t)$

More Candies Please

- Goal of the reinforcement learning is to find a policy $\pi(a|s)$ that maximizes the sum of future rewards

$$J(\pi) = \mathbb{E}_{\tau\sim\pi}\left[\sum_t r_t\right] = \int d\tau \pi(\tau) R(\tau) \qquad \tau = \{(s_1, a_1), (s_2, a_2), \ldots\}$$

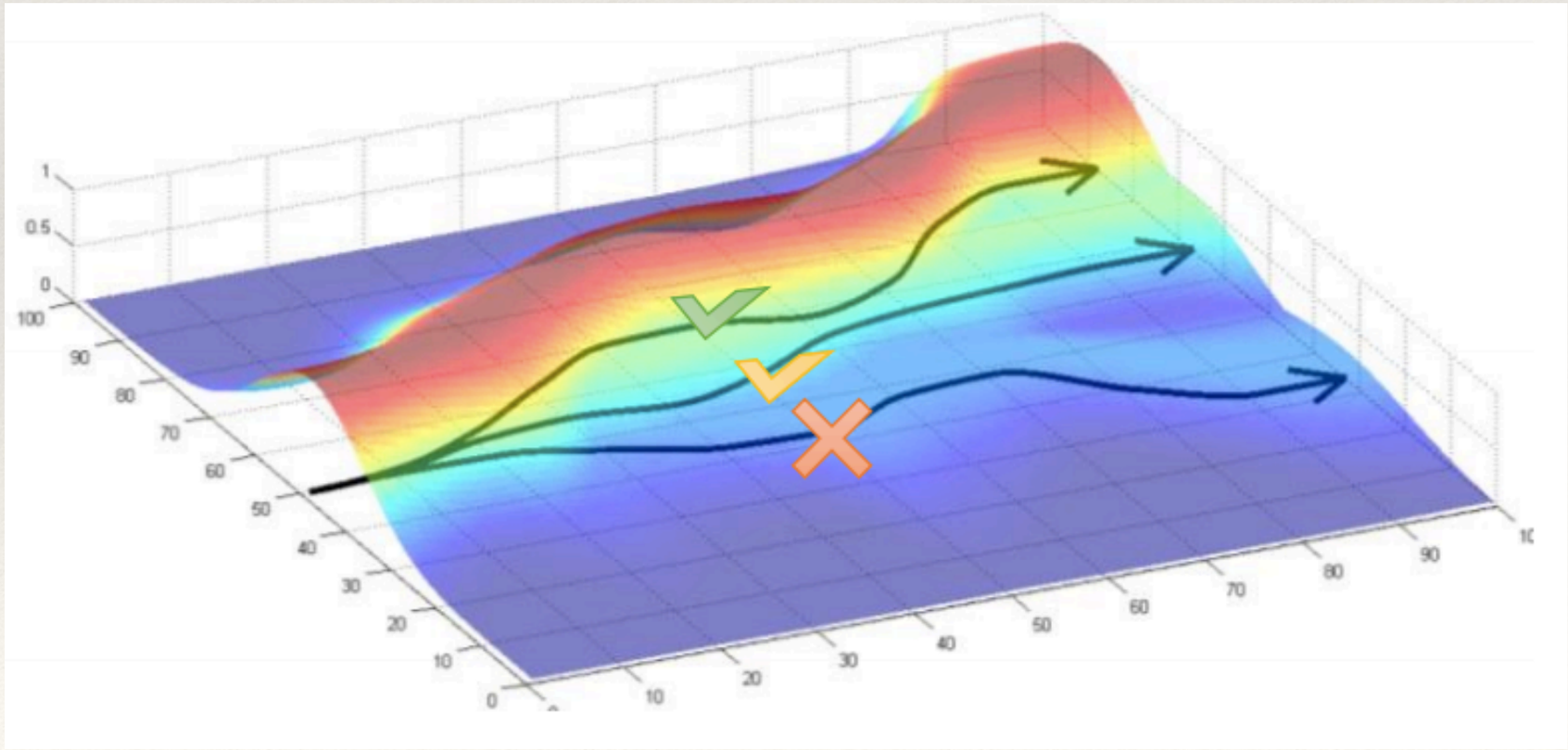
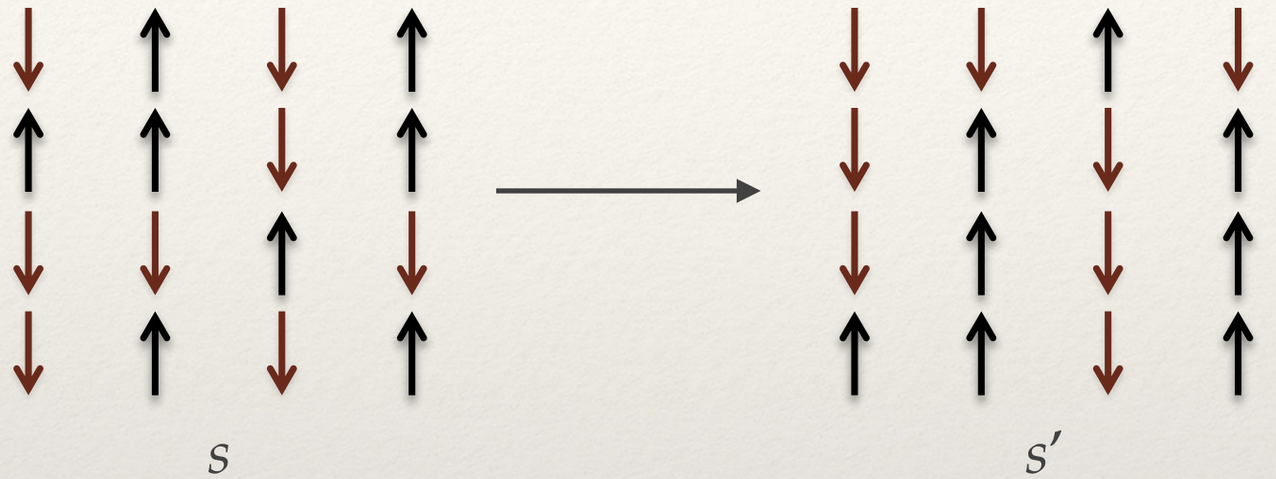


Image courtesy: Berkeley cs294-112

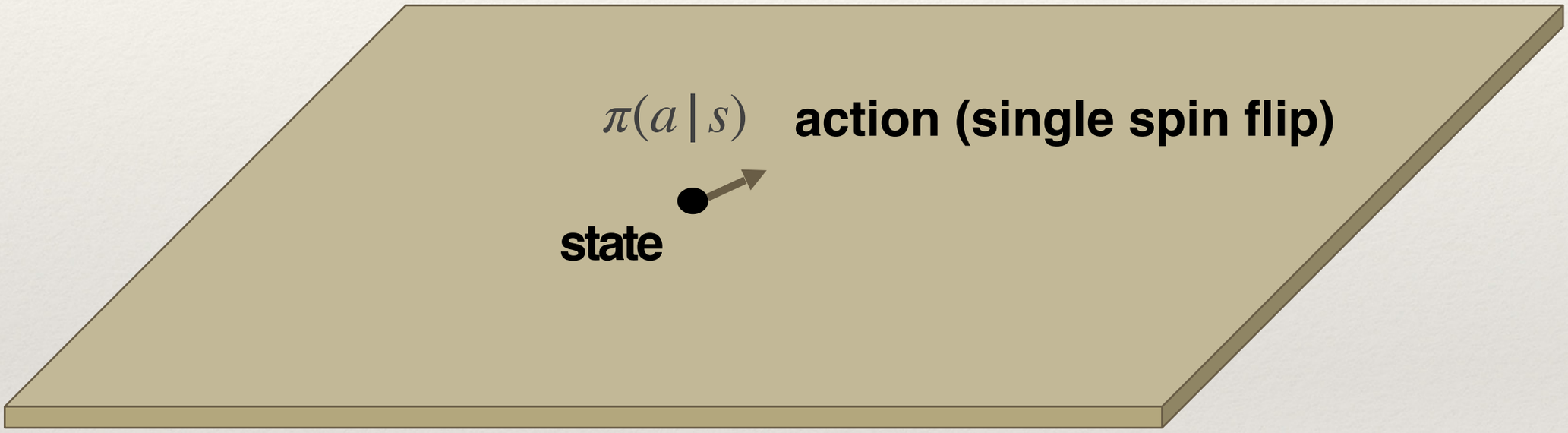
Markov Chain Monte Carlo



- Metropolis-Hastings algorithm

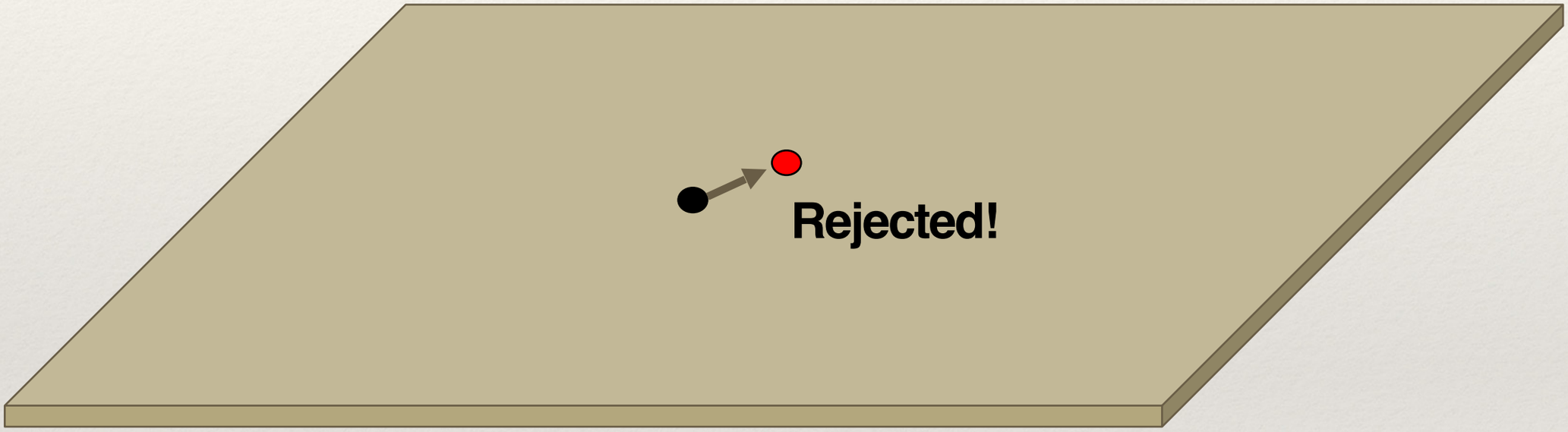
$$A(s \to s') = \min\left[1, \frac{W(s)}{W(s')} \cdot \frac{\pi(s \to s')}{\pi(s \to s')}\right]$$

MCMC



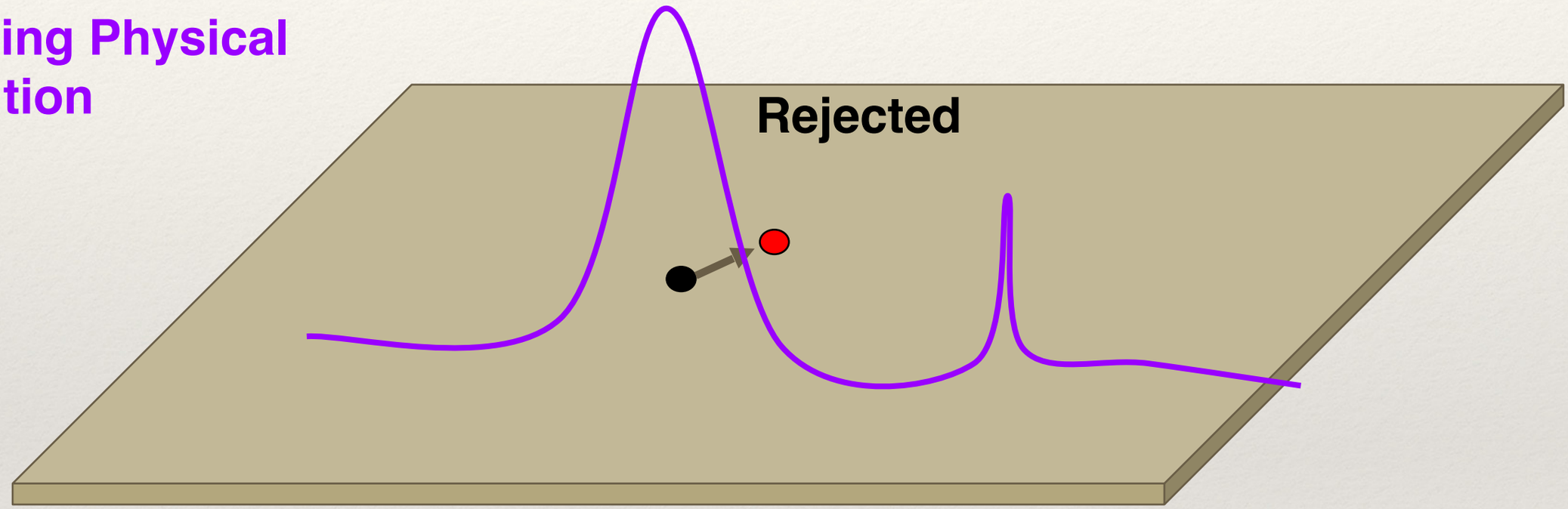
Configuration Space / Action x State Space

MCMC



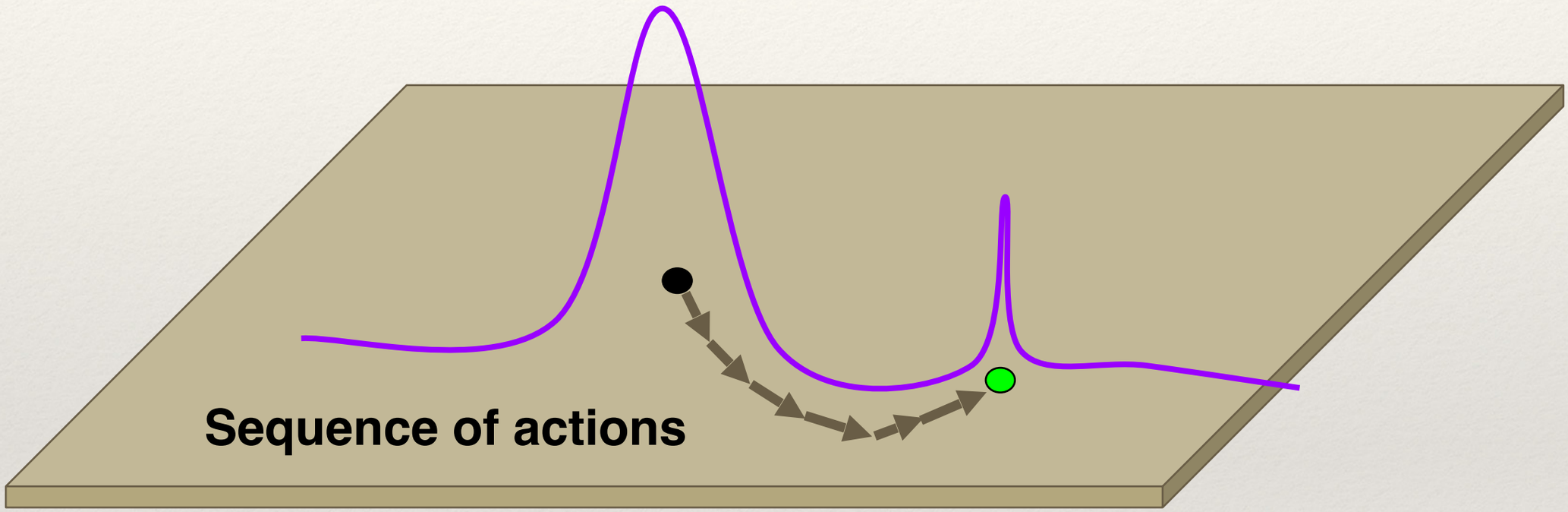
Configuration Space / Action x State Space

MCMC



Configuration Space / Action x State Space

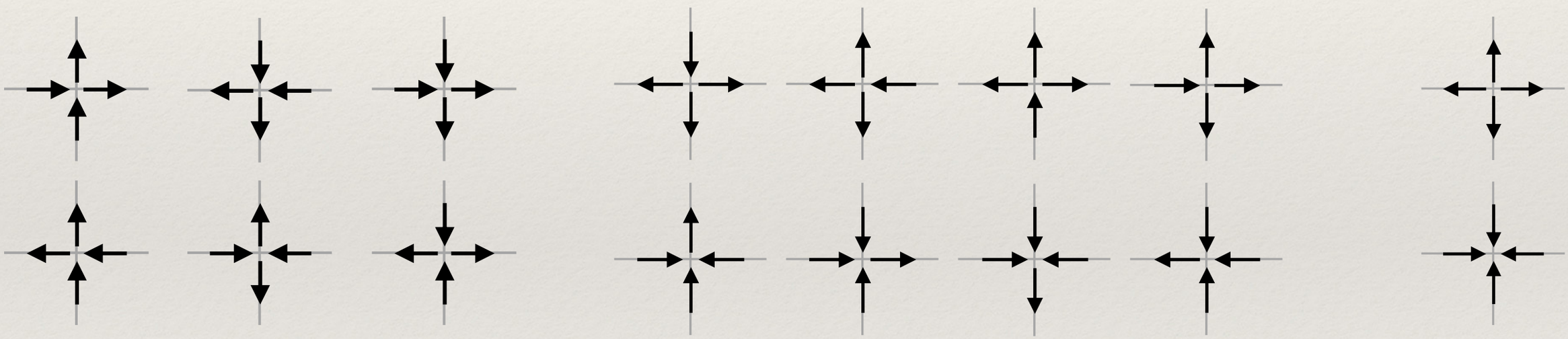
MCMC



Configuration Space / Action x State Space

Square Ice Model

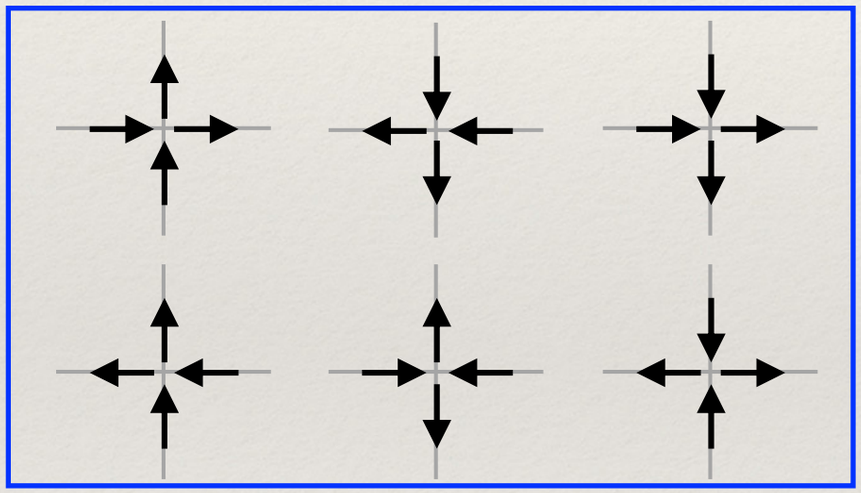
$E = 0$ $E = \epsilon$ $E = 2\epsilon$



Ice rule Defects

Square Ice Model

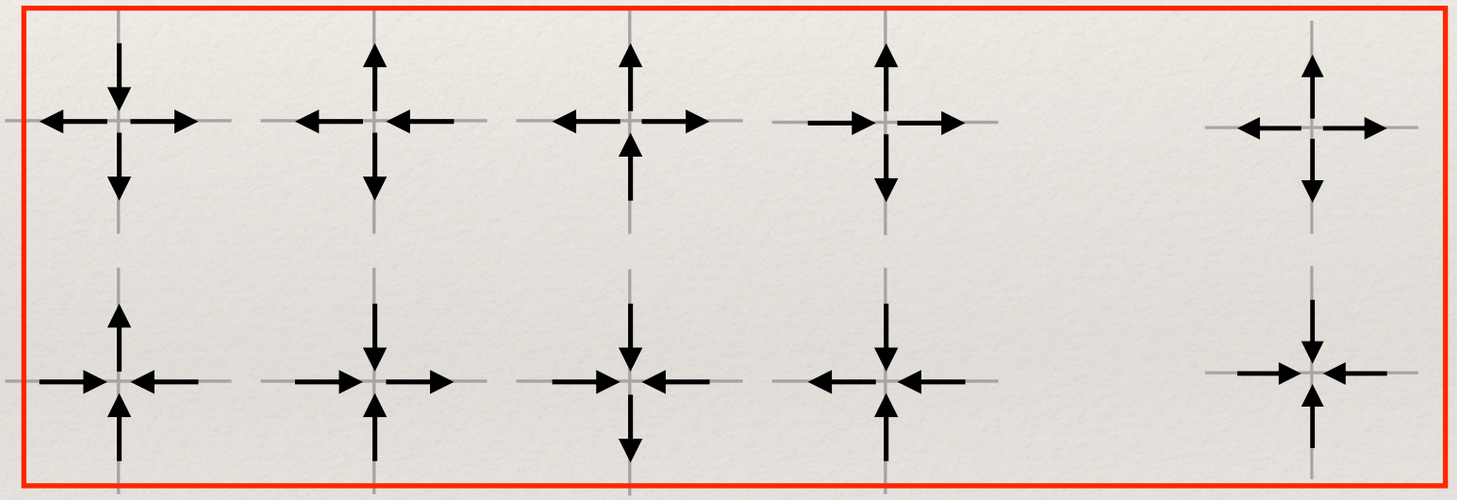
$E = 0$



Ice rule

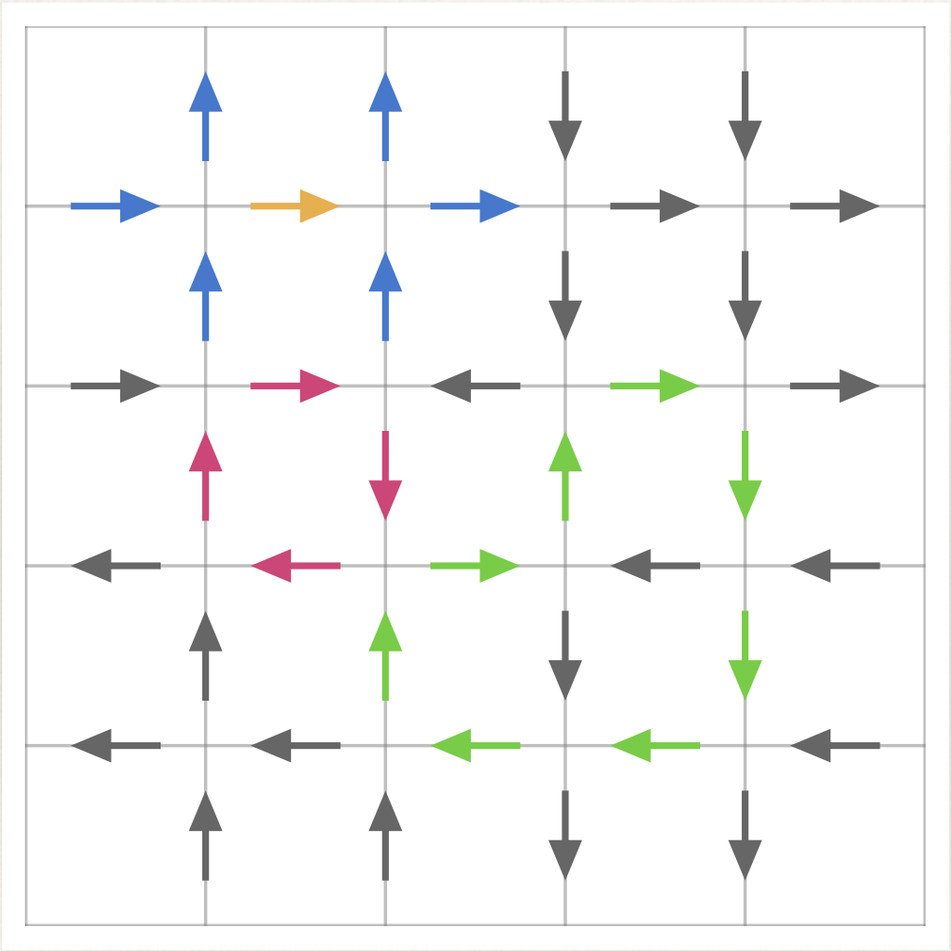
$E = \epsilon$

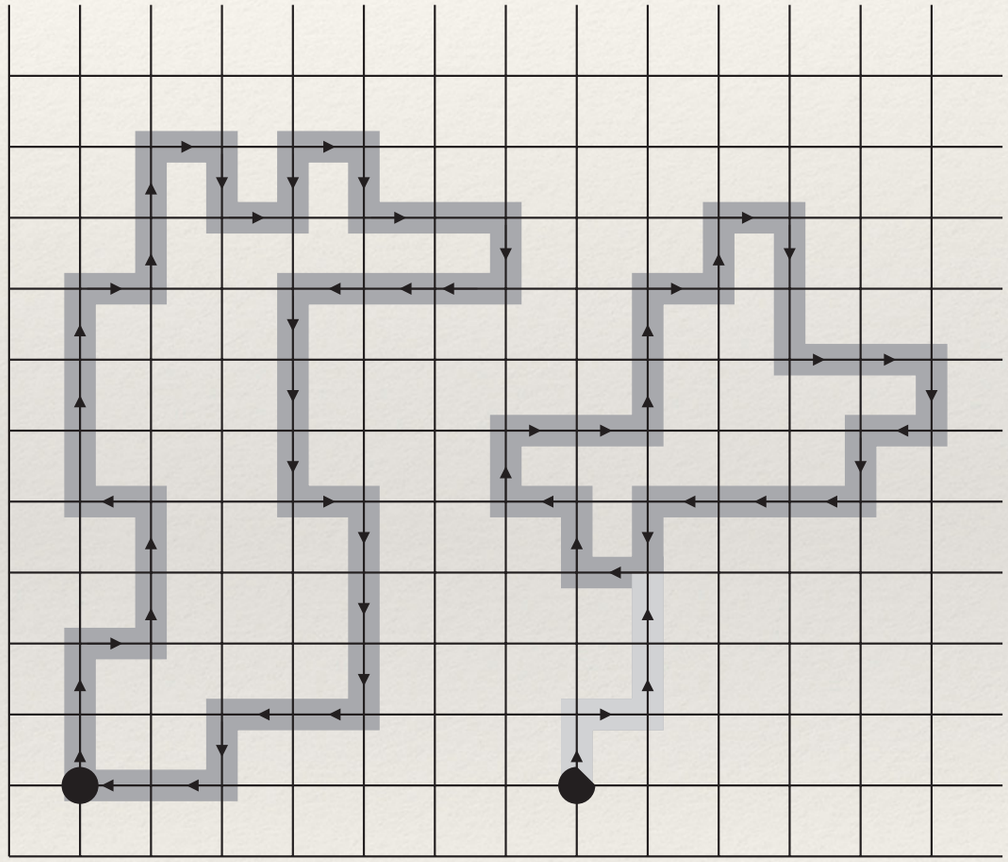
$E = 2\epsilon$



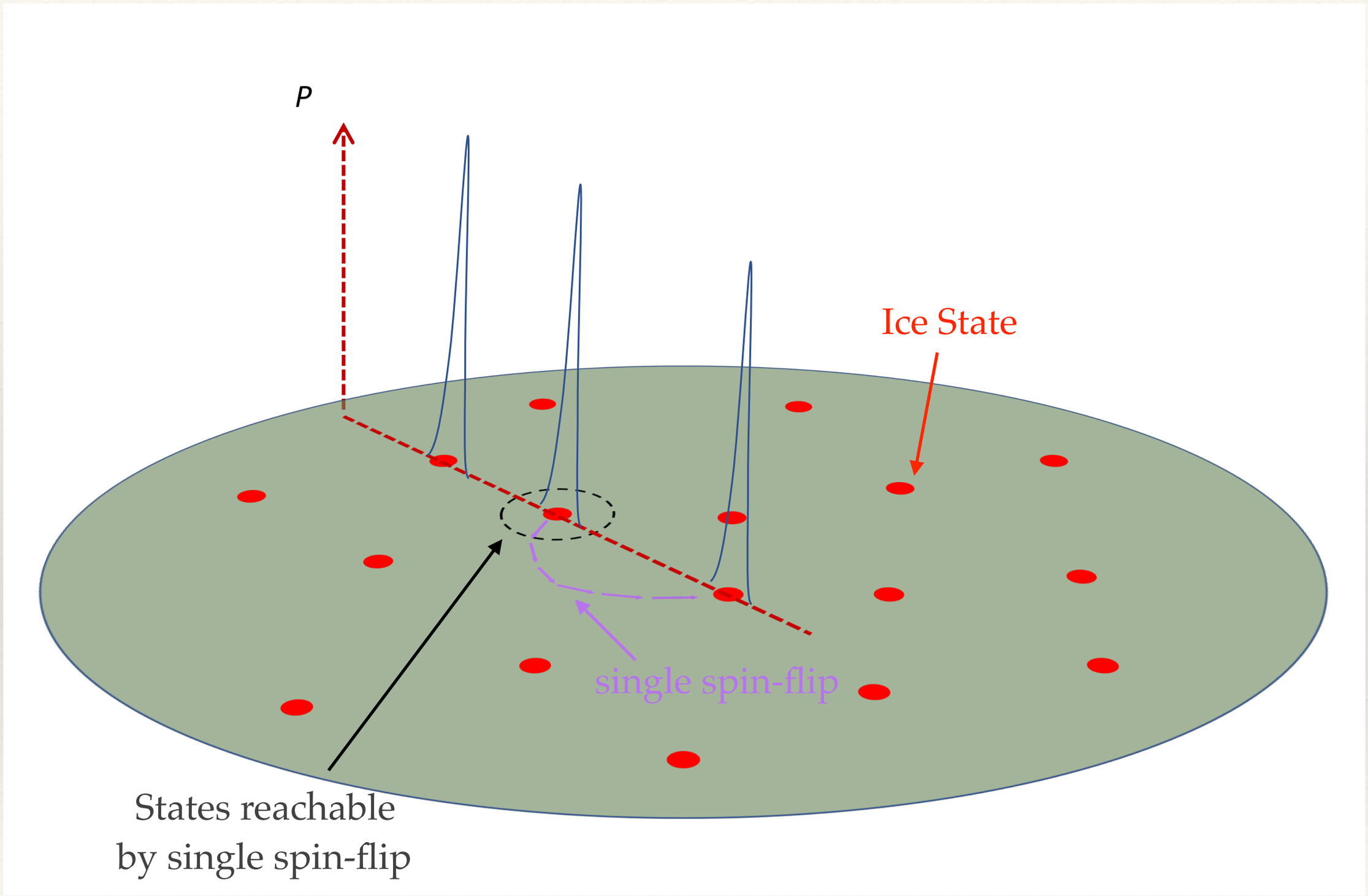
Defects

Ice State

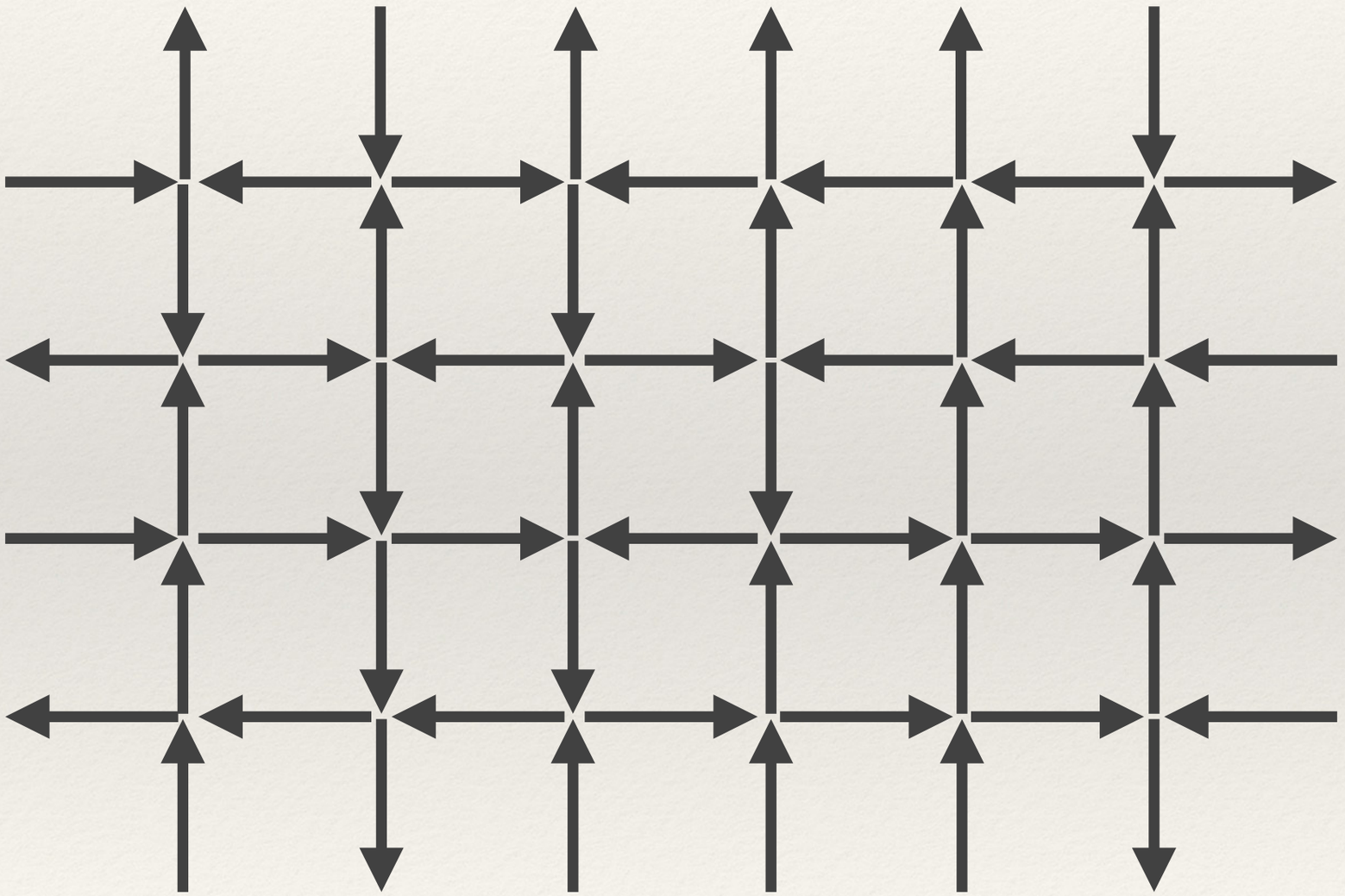




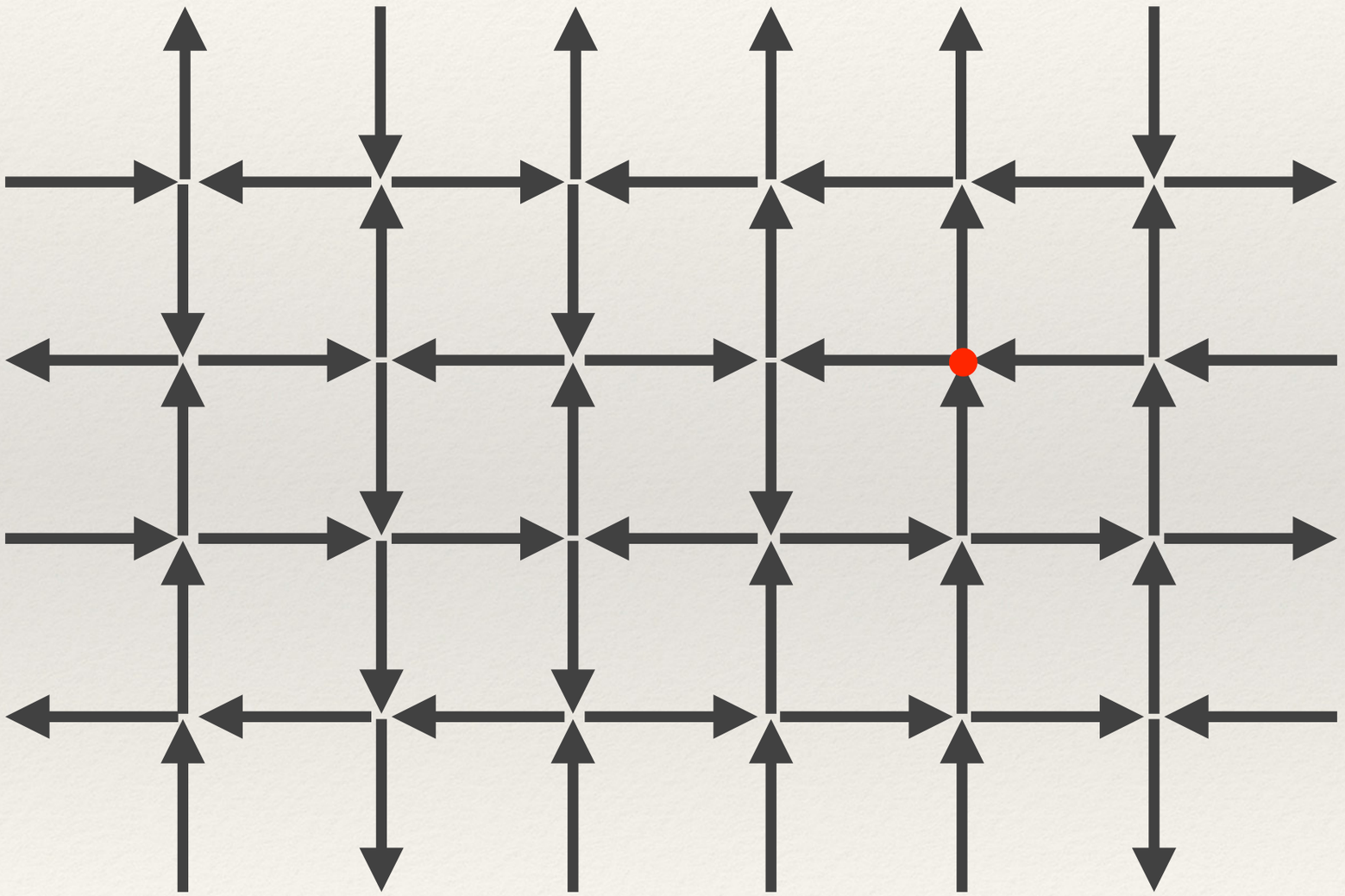
R. G. Melko, Master Thesis


P
Ice State
single spin-flip
States reachable
by single spin-flip

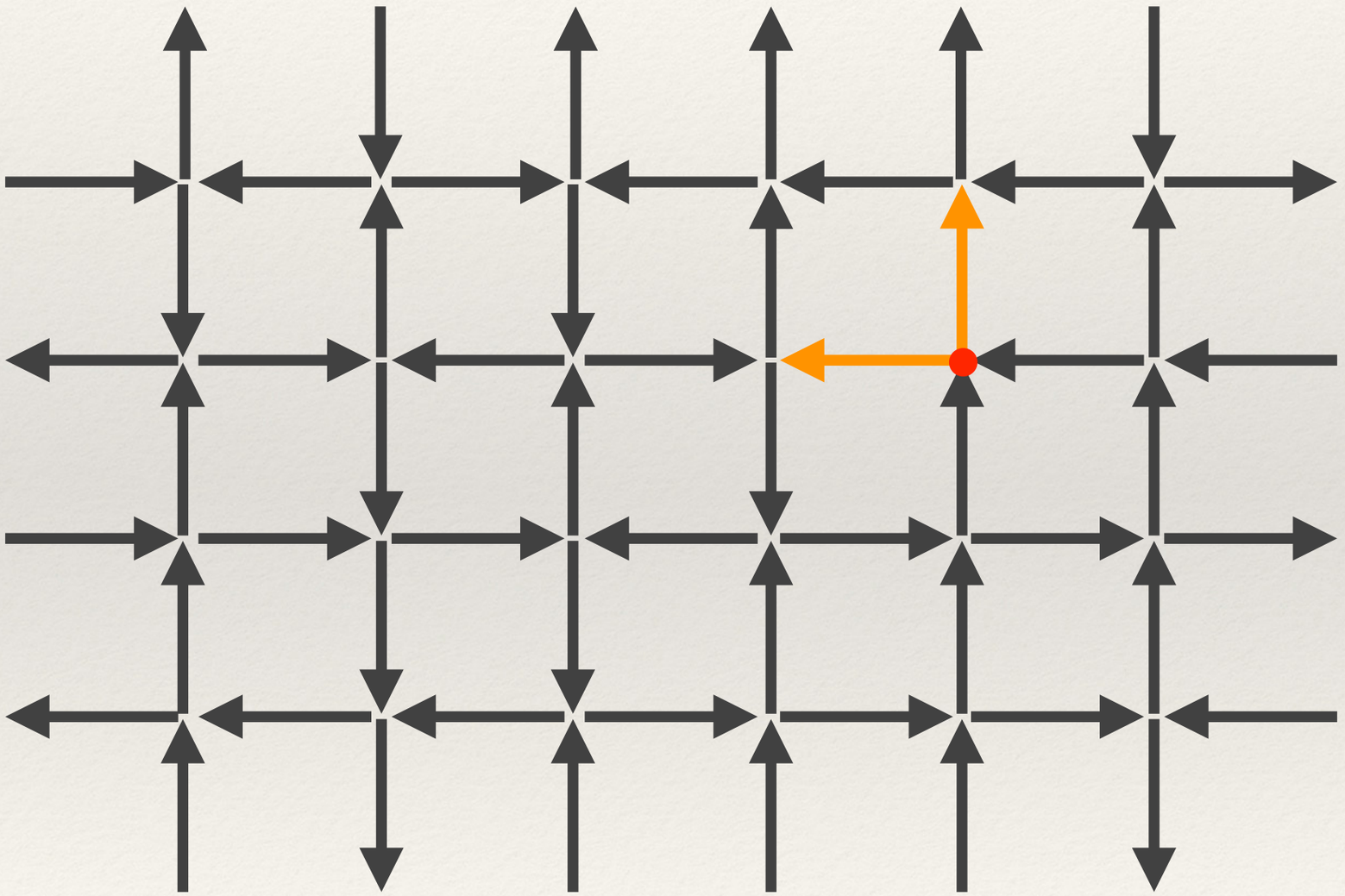
Loop Algorithm



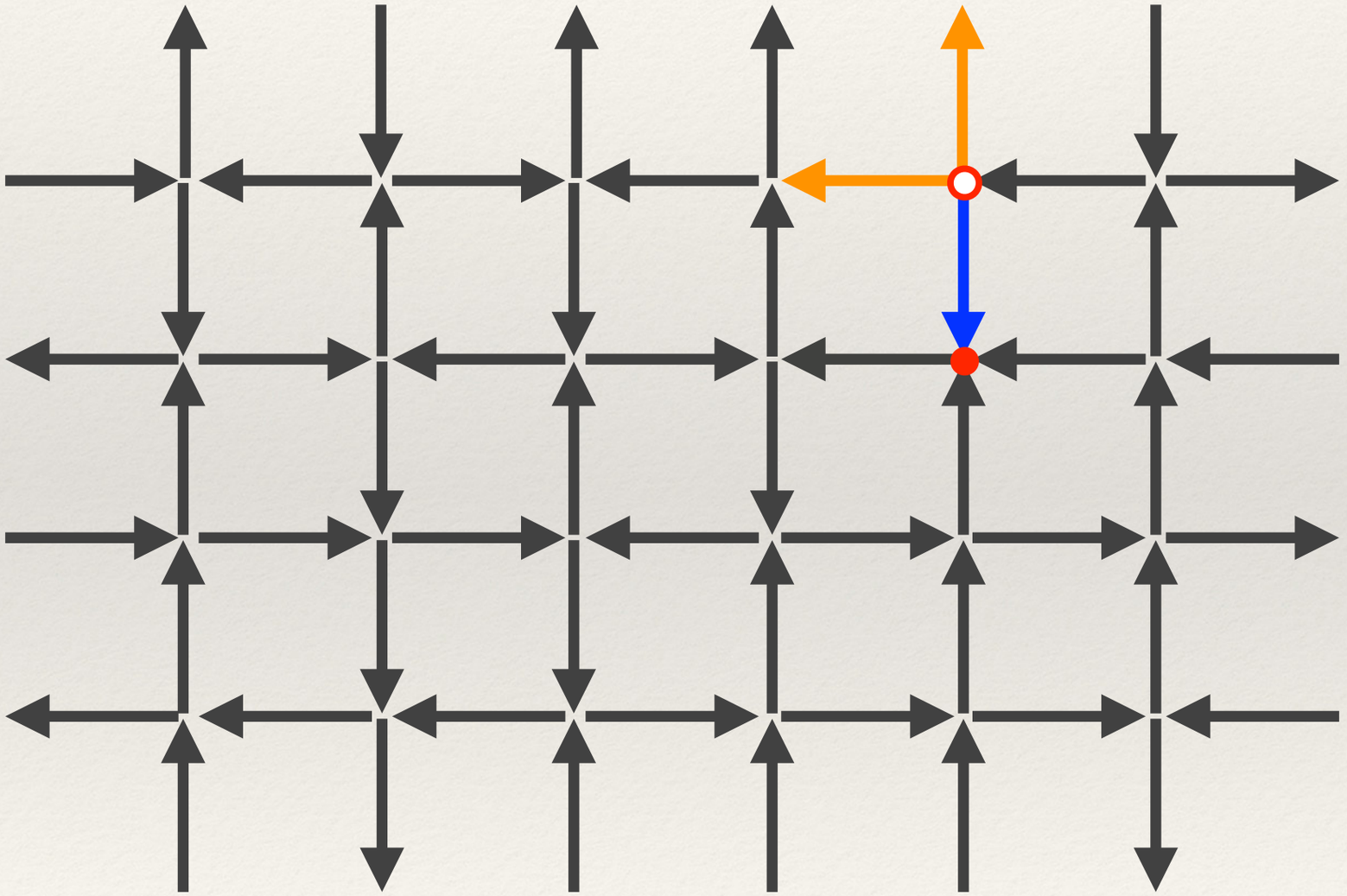
Loop Algorithm



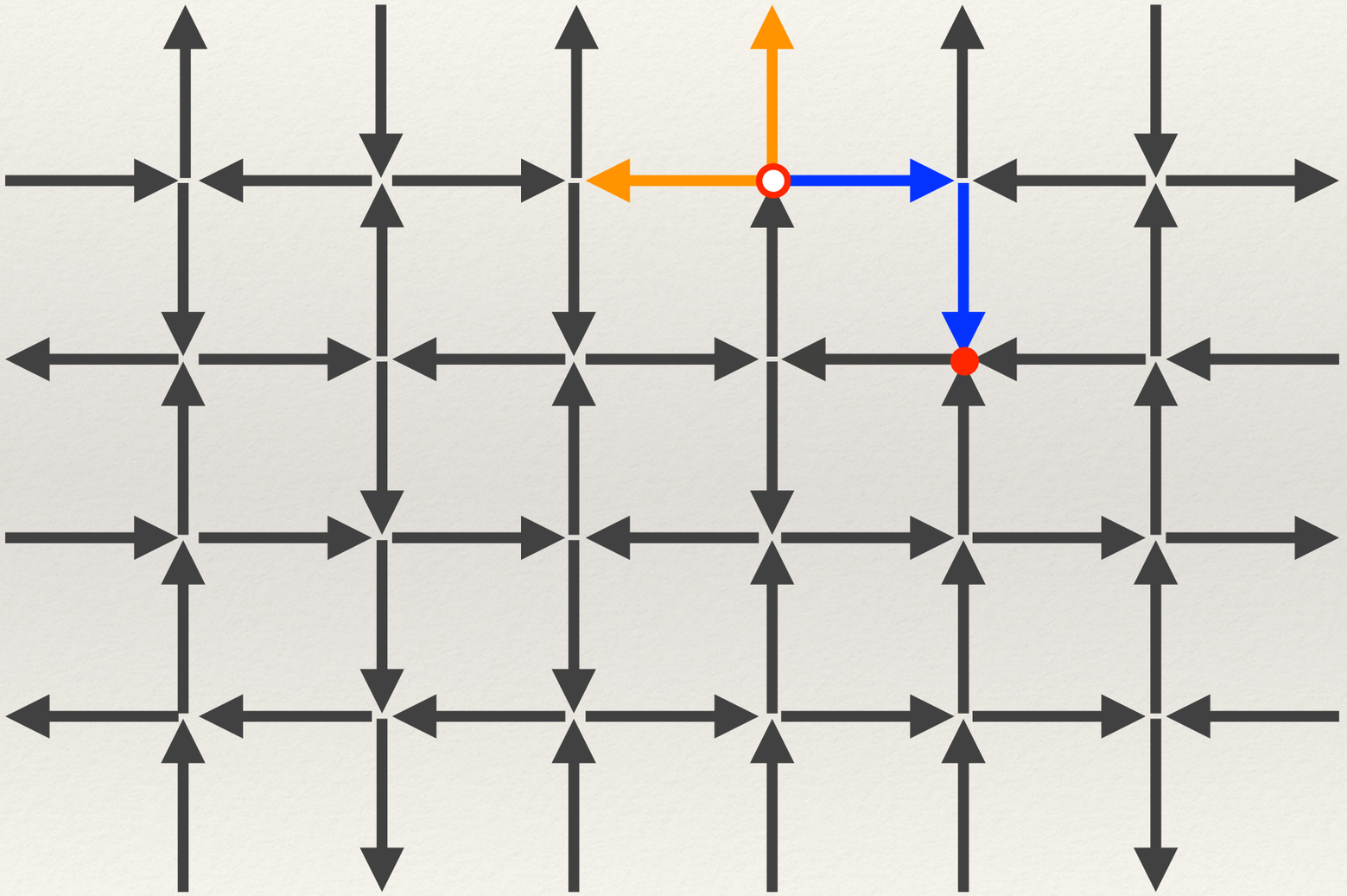
Loop Algorithm



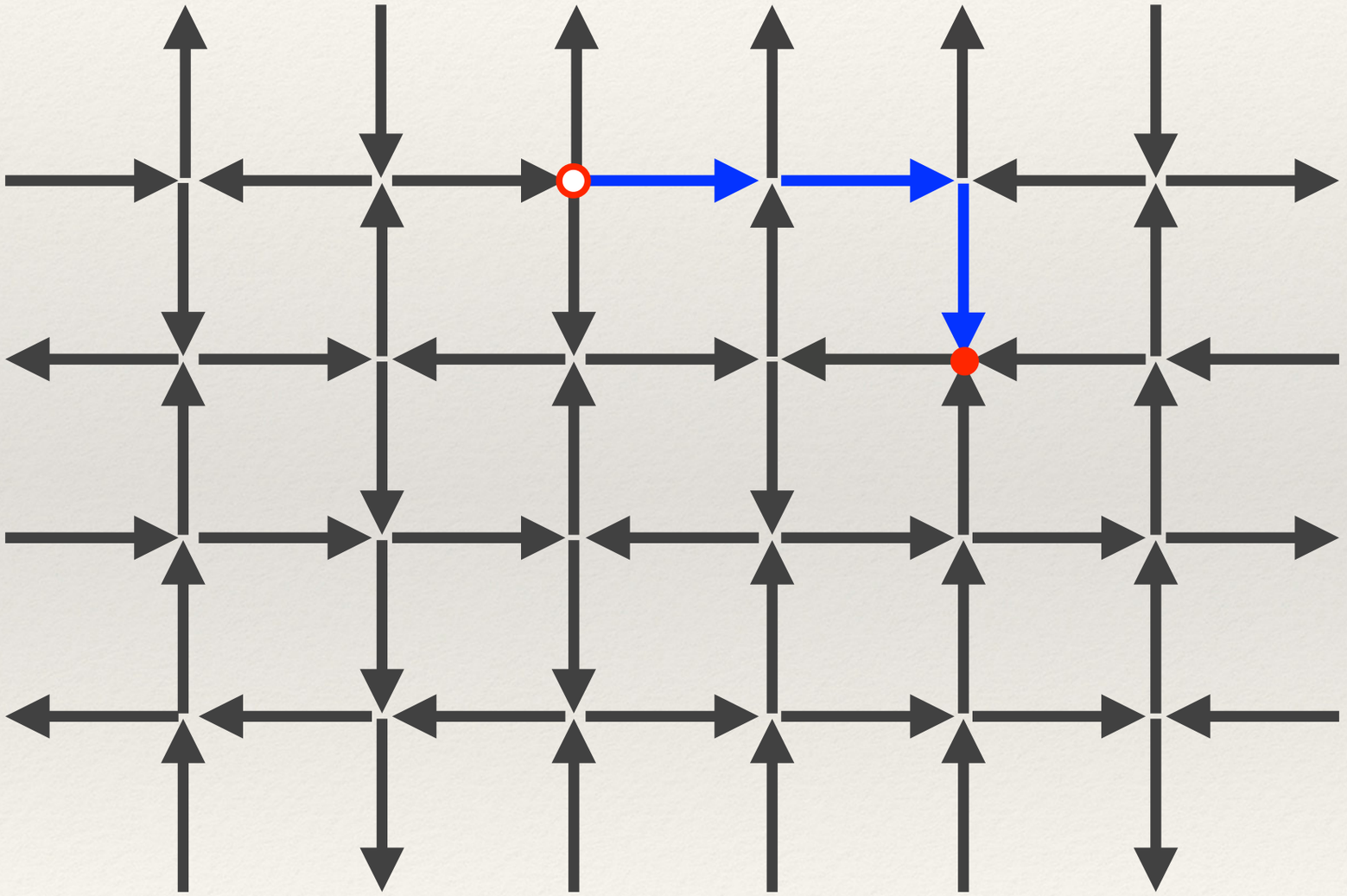
Loop Algorithm



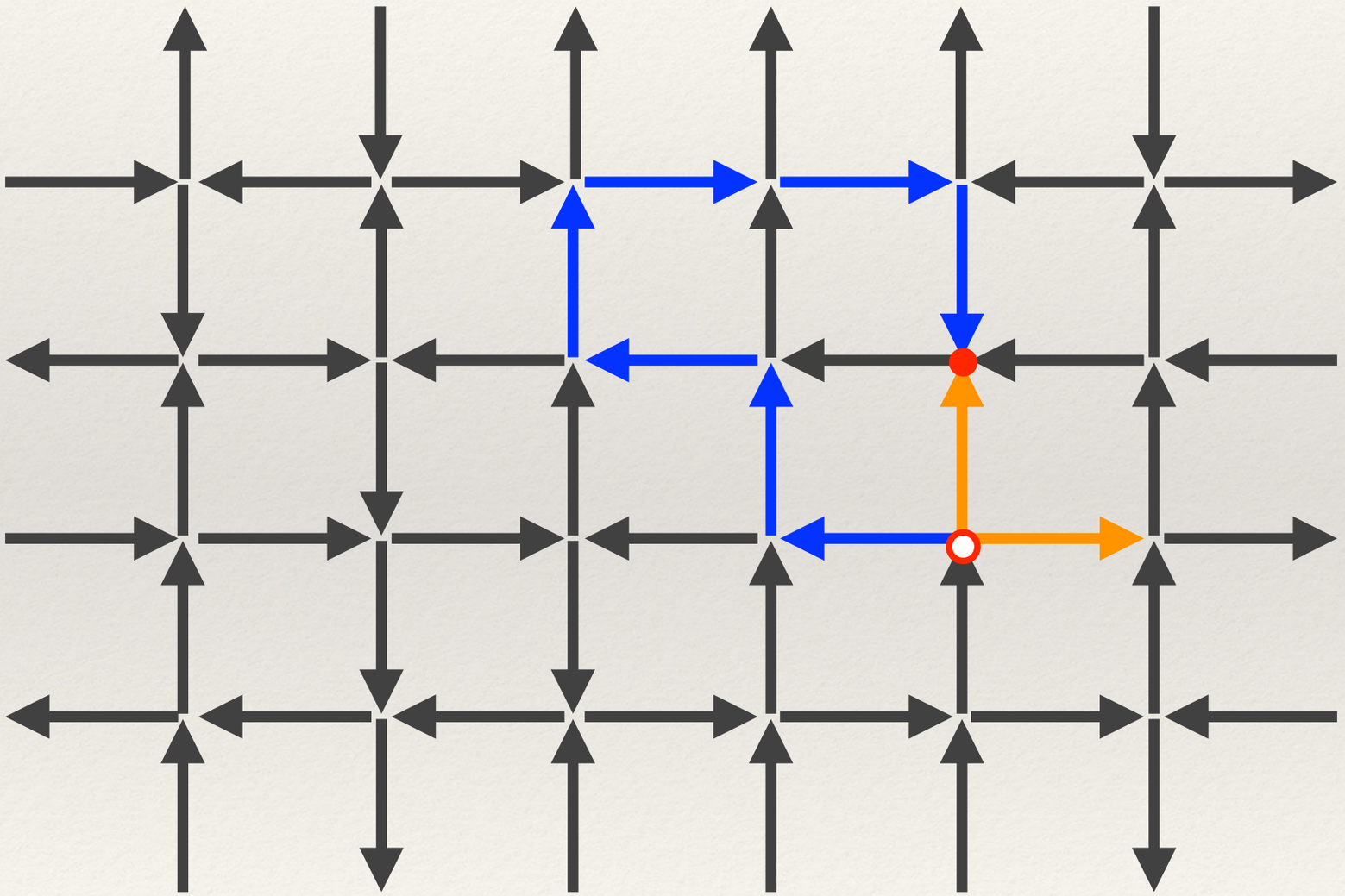
Loop Algorithm



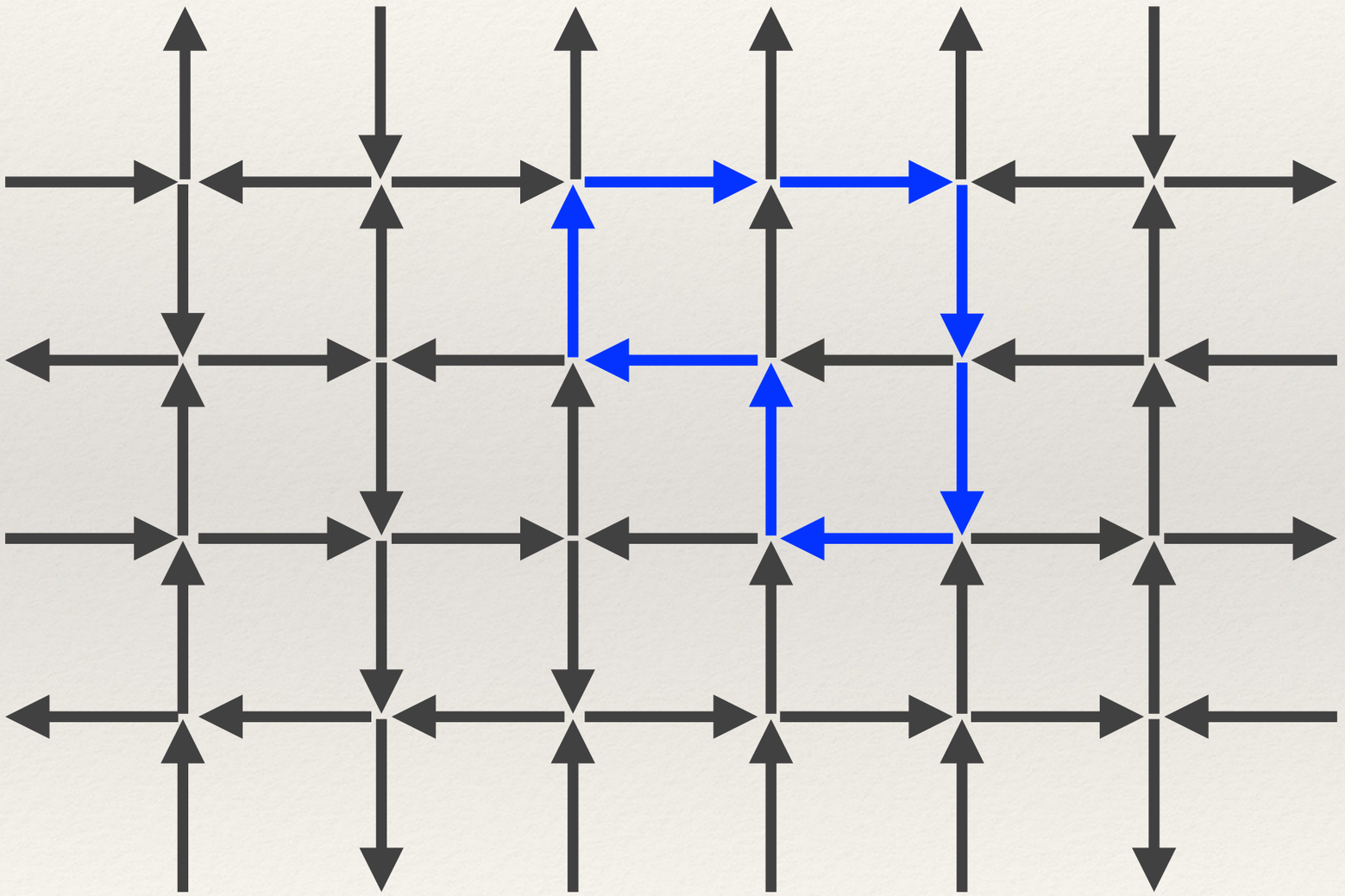
Loop Algorithm



Loop Algorithm



Loop Algorithm



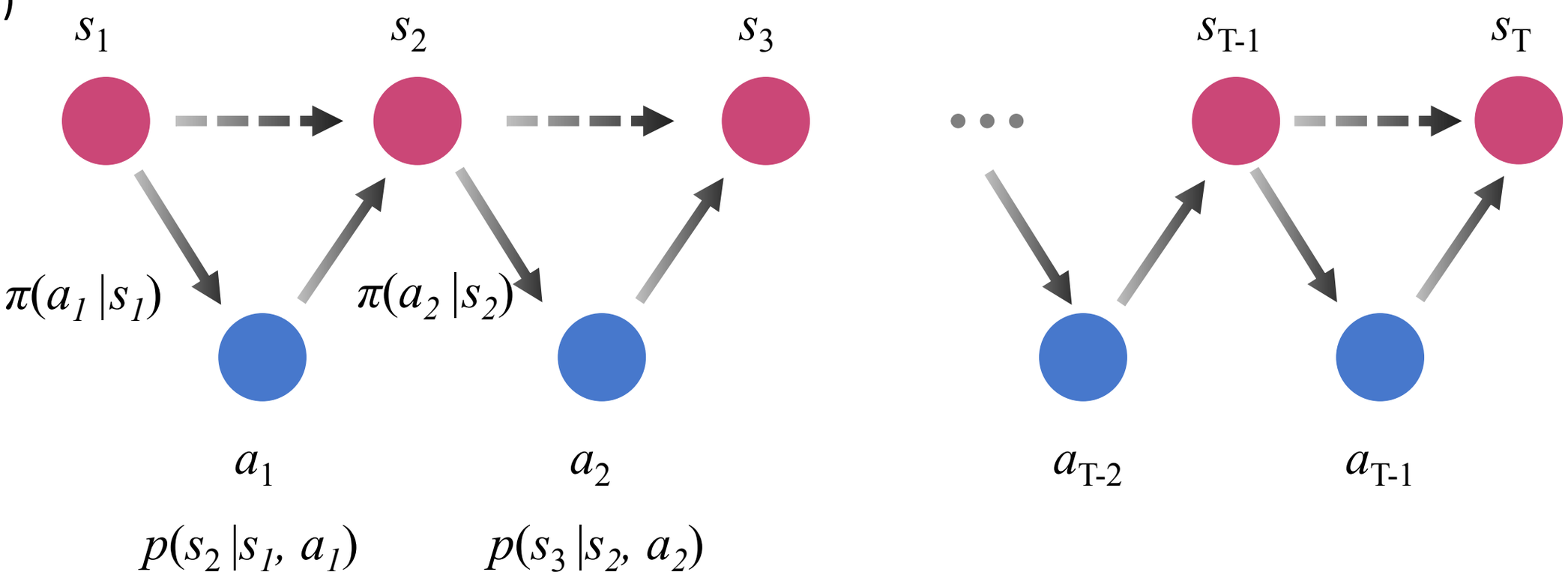
(a)

Markov Chain

$p(s_1 \rightarrow s_T)$

Markov Decision Process

(b)



MDP+Policy=MC

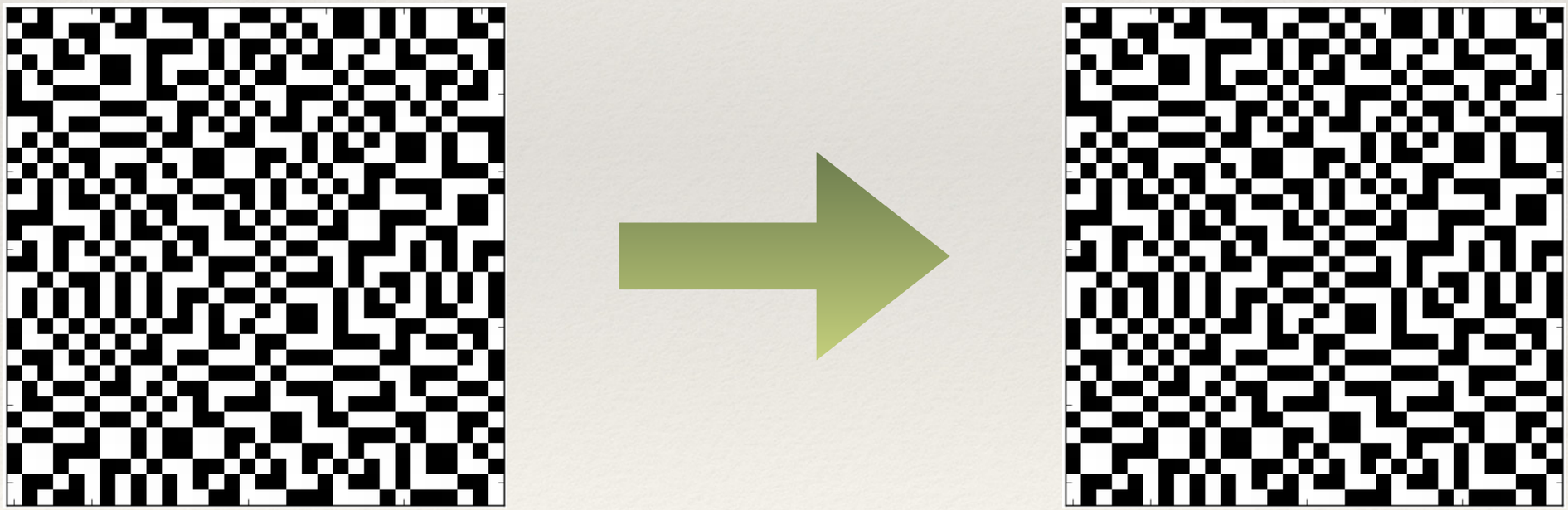
Exploration vs Exploitation

- **Exploitation**: Make the best decision given current information
- **Exploration**: Gather more information

- The best long-term strategy may involve short-term sacrifices
- Gather enough information to make the best overall decisions
- How to assign rewards (candies) to achieve a balance between the two?

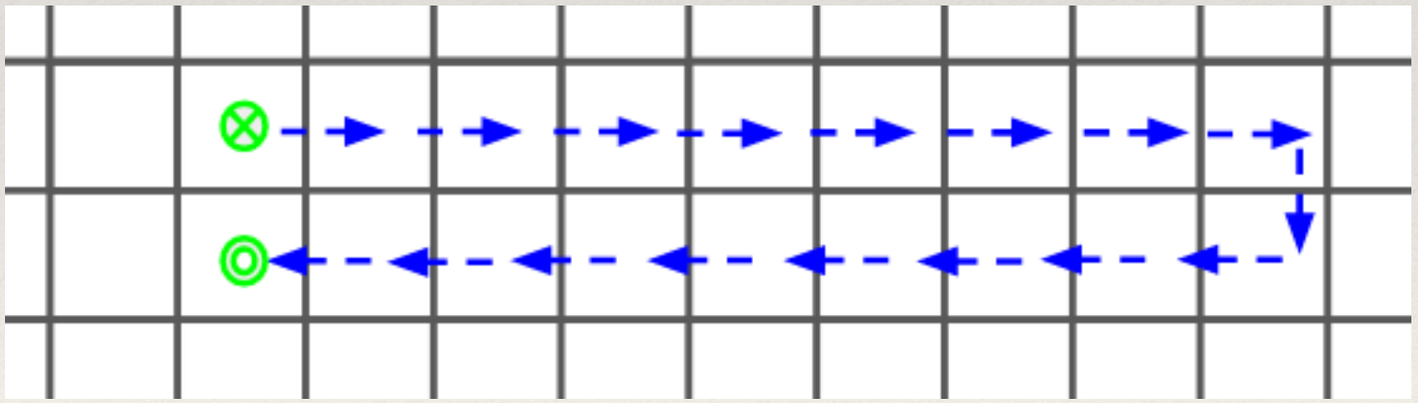
Assigning Rewards

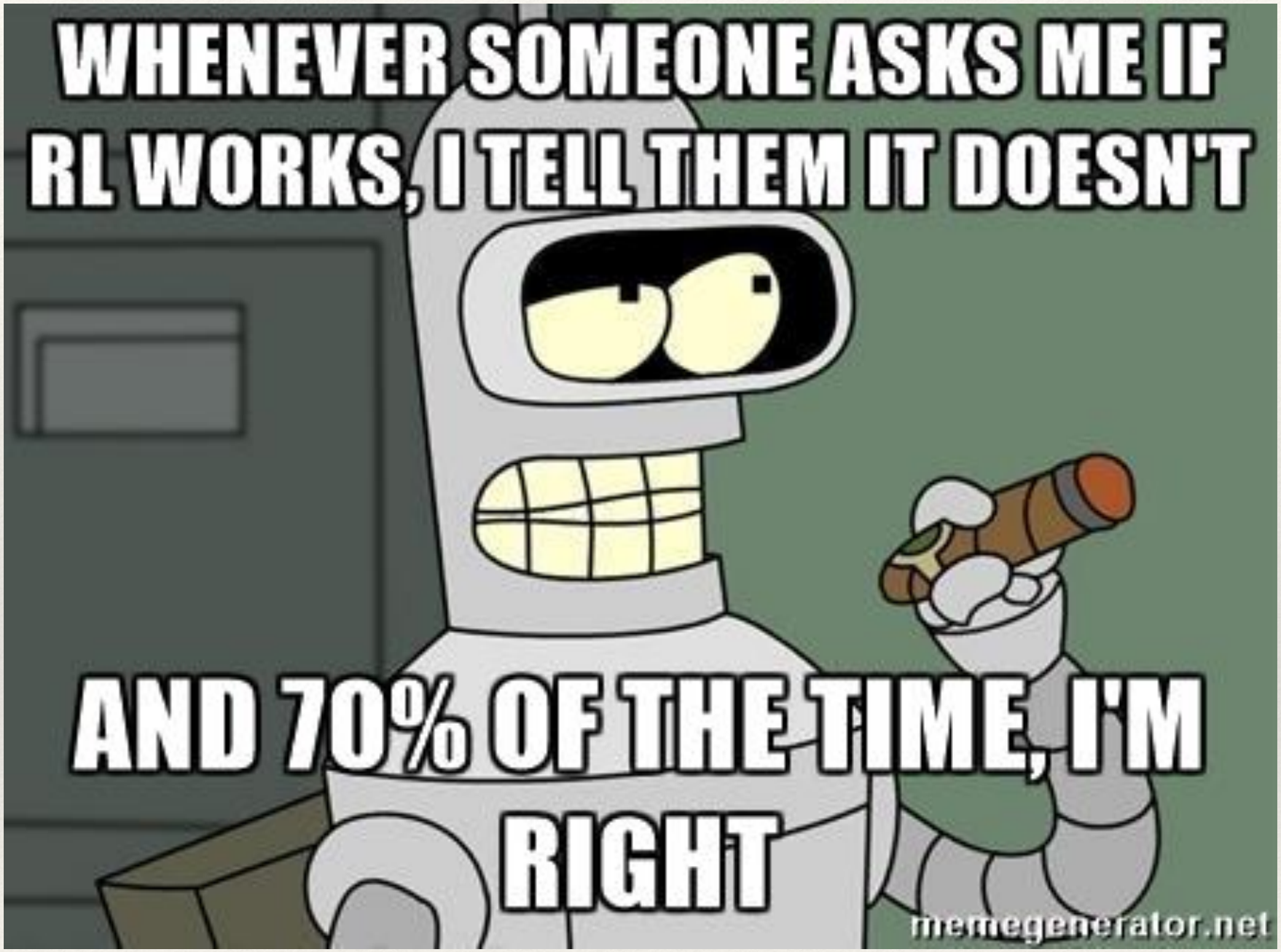
- Starting from an ice configuration, if you can generate an ice configuration, I will give you a candy.



Assigning Rewards

- Ok, I also want you to flip as many spins as possible.
- Reward: r = configuration change density when accepted, otherwise $r = 0$





https://www.alexirpan.com/2018/02/14/rl-hard.html

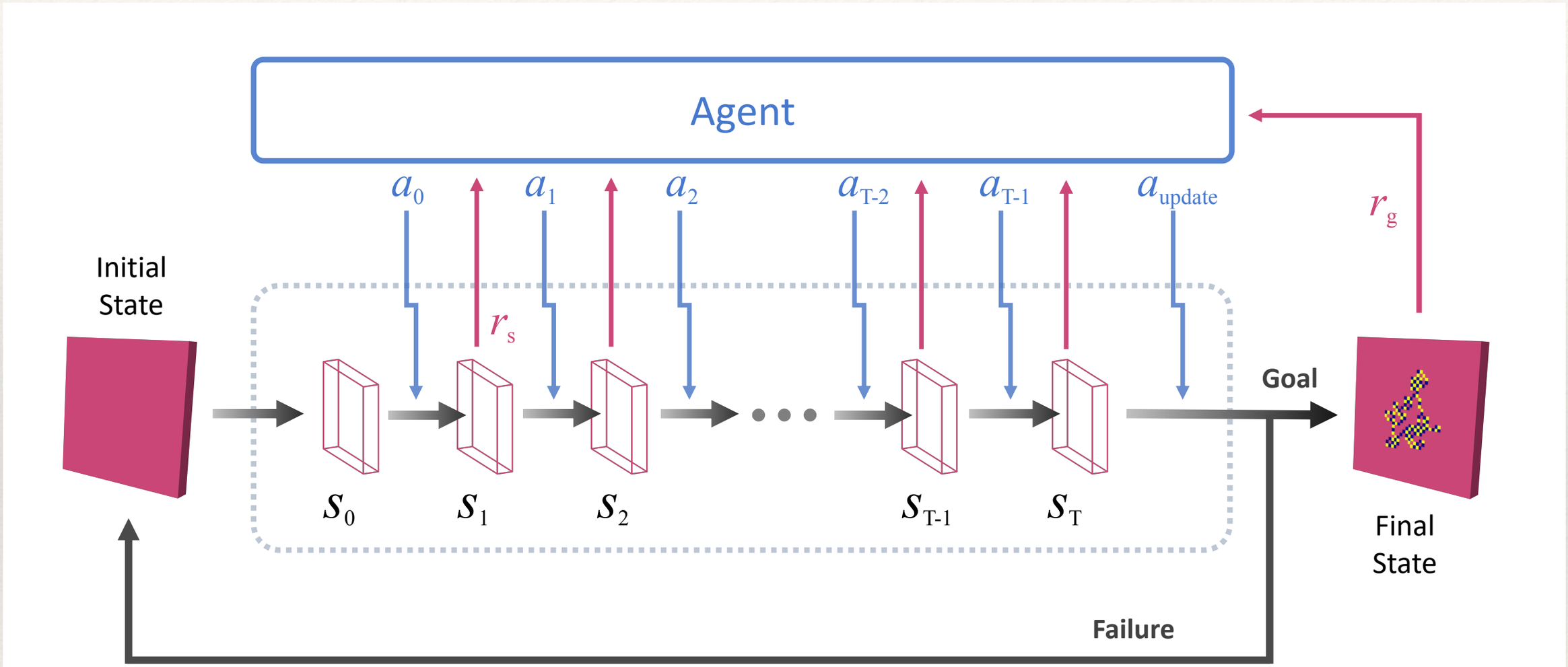
Other Failed Attempts

- Teach machine to generate ice configurations (GAN): Failed, discrete underlying distribution
- Teach machine what is a loop and mimic the loop generation (transfer learning): Failed, not a local object
- Leave breadcrumbs along the way…

Lessens learned

- Easy for human/program to check for ice rule violation.
- Let the agent generate some update pattern and decide when to apply the update to the spin configurations.
- Check if the new configuration satisfies the ice rule.
- Give rewards accordingly.

Ice Game: Interaction



$$r_T(s,a) = \begin{cases} r_g = 1, \text{ if episode ends up with ice states} \\ r_f = 0, \text{ if episode ends up with defects} \end{cases}$$

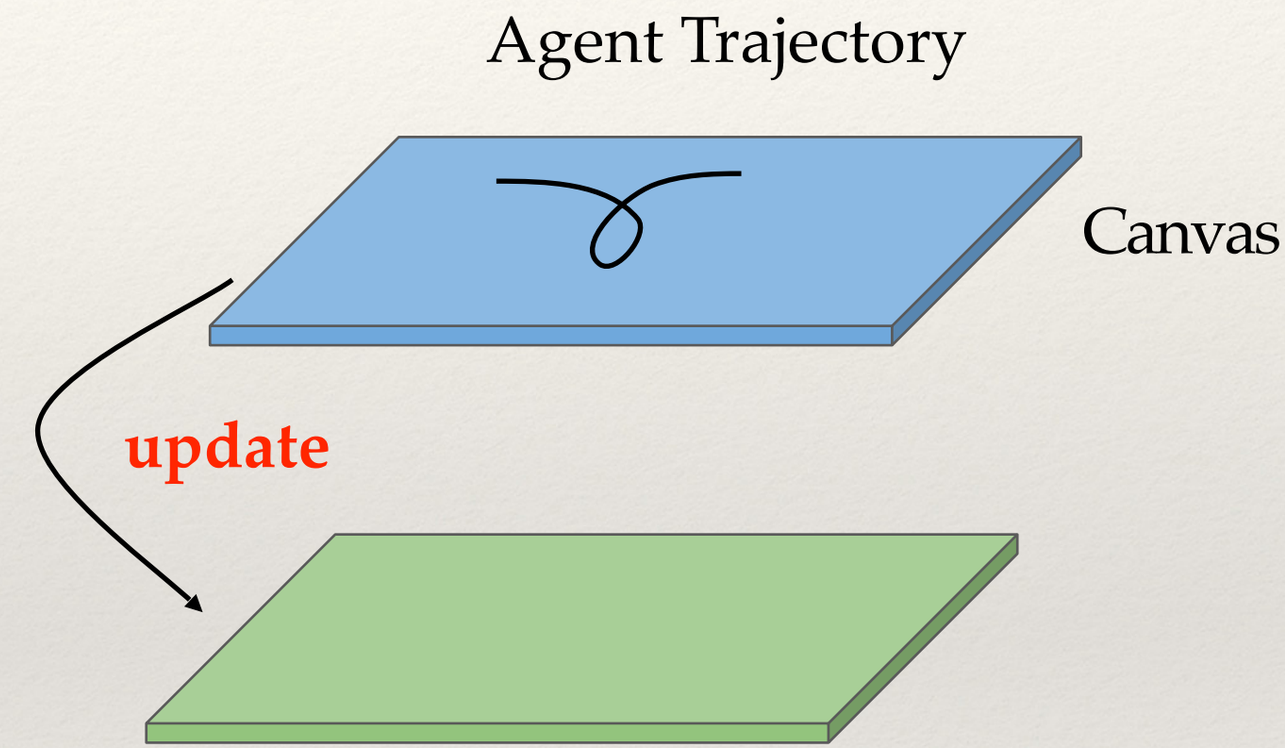
stepwise reward: $\frac{r_s}{r_g} \sim O(N^{-1})$

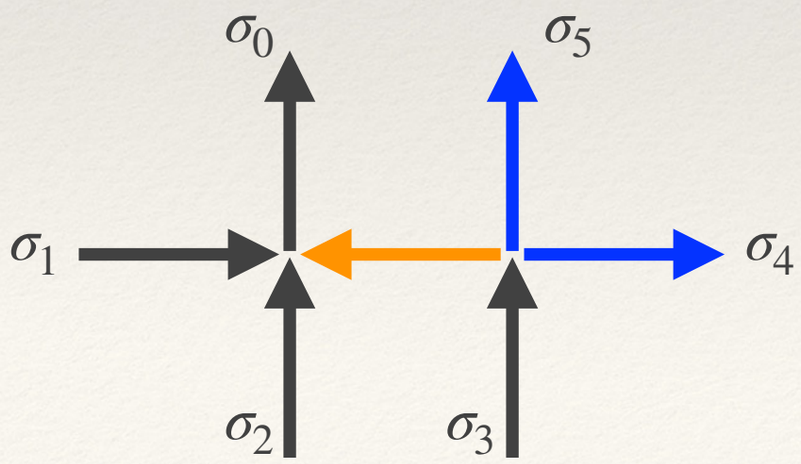
Ice Game: Action

Actions

Walking on Canvas

1	flip σ_0
2	flip σ_1
3	flip σ_2
4	flip σ_3
5	flip σ_4
6	flip σ_5
7	**update**





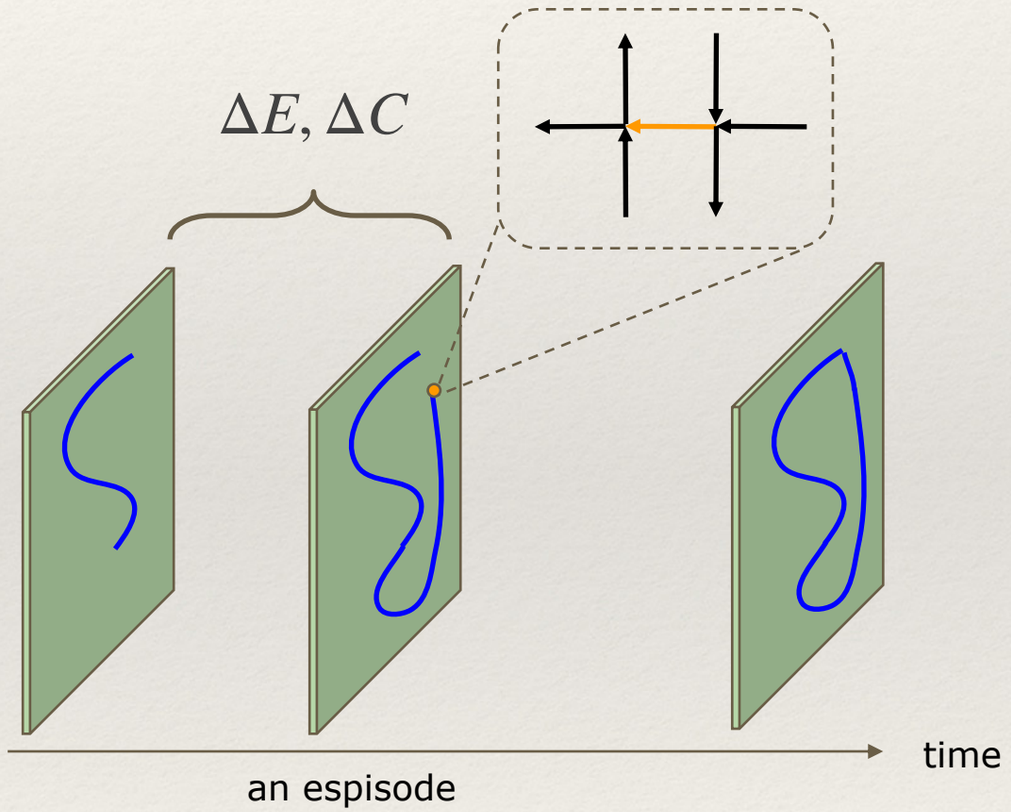
$$\pi_{\text{algo}} = \left[0,0,0,0,\frac{1}{2},\frac{1}{2},0\right]$$

Ice Game: Observations

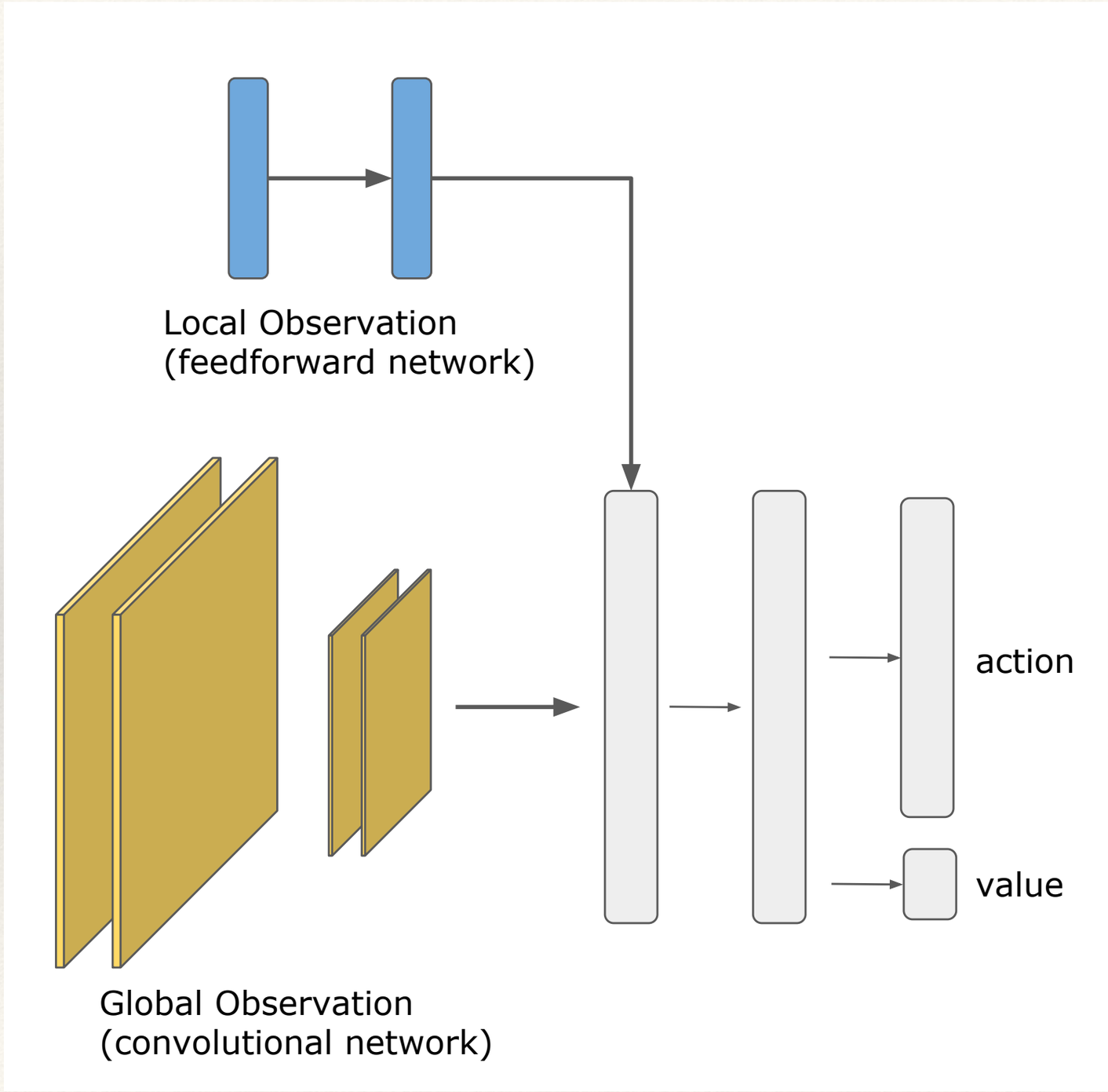
- **Local observations**
  $O_l = \{\sigma_i, \Delta E, \Delta C\}$,
  σ_i : neighboring spins
  ΔE: energy change
  ΔC: configuration change

- **Global observations**
  $O_g = S_t - S_0 = C_t$
  C_t : trajectory

- State s=$[O_l, O_g]$



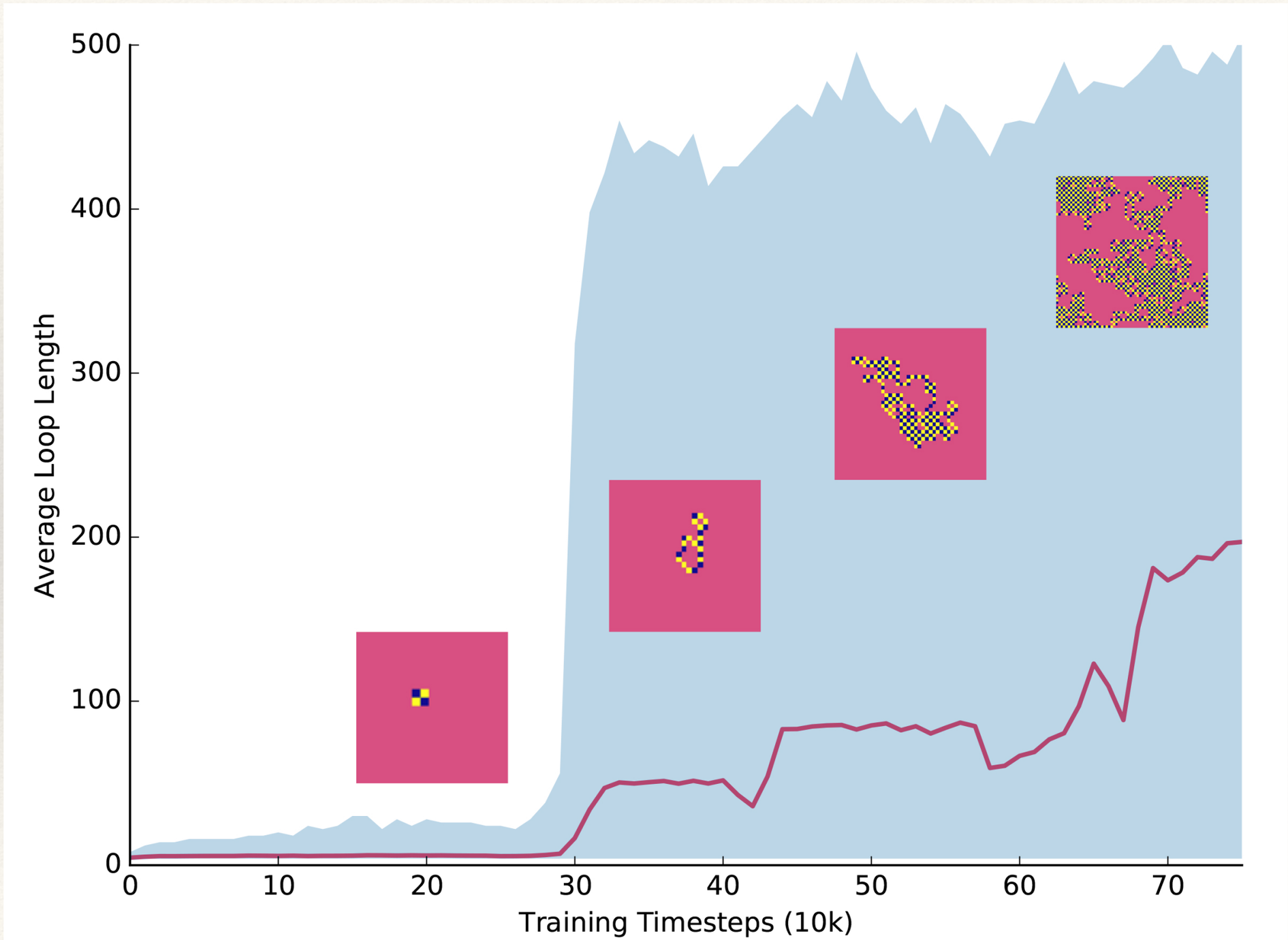
Ice Game: Architecture

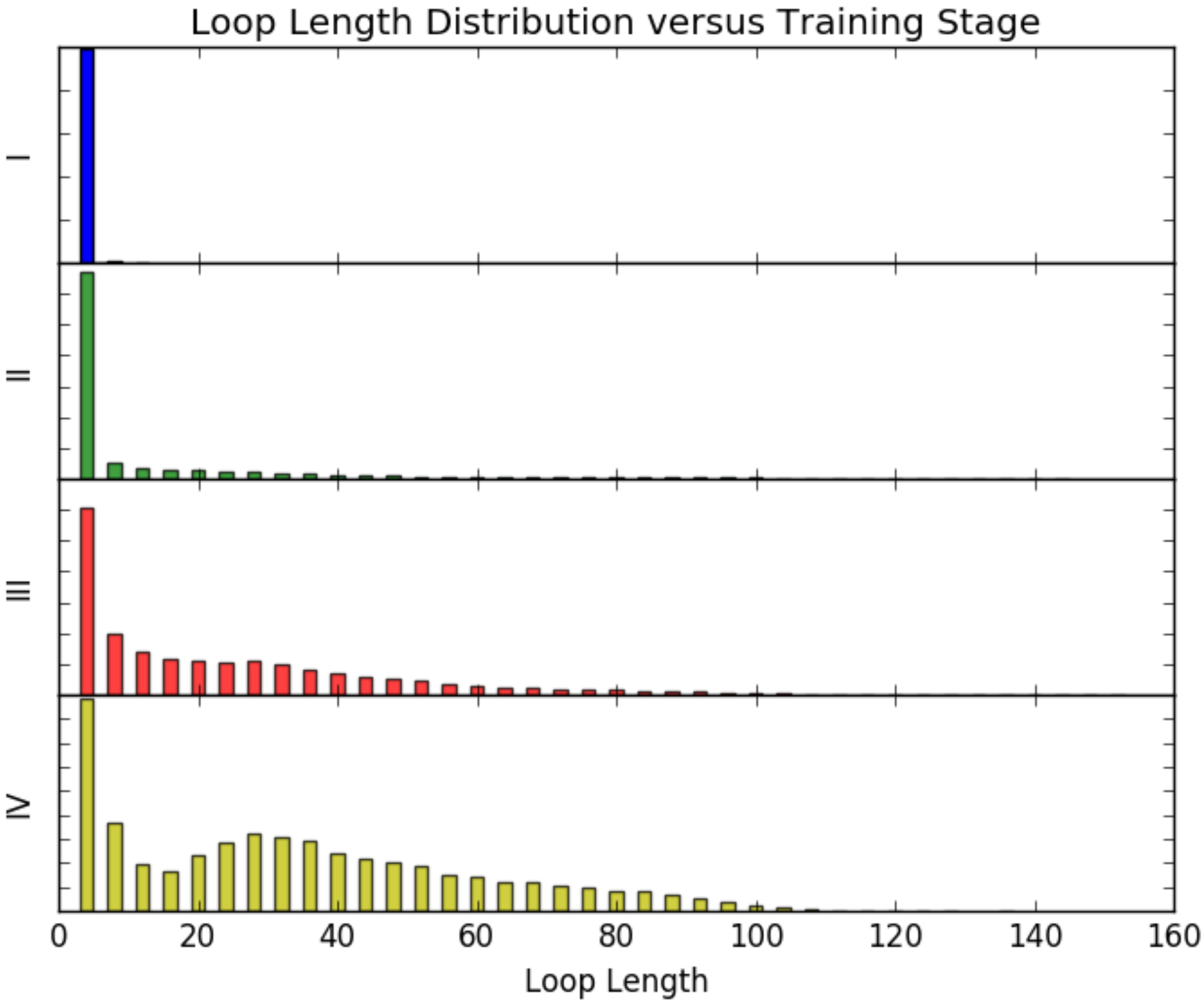


Learning Mechanism

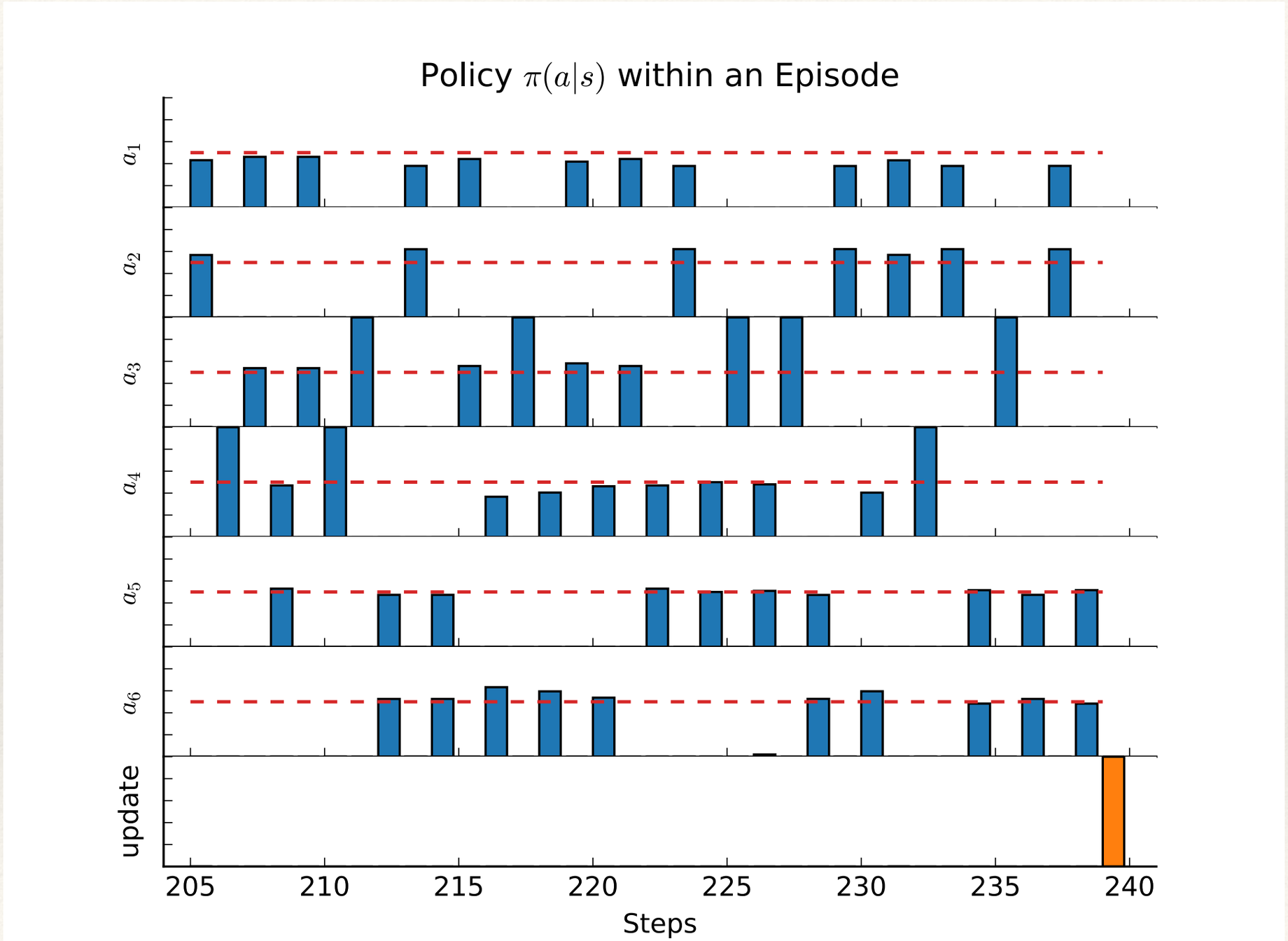
- Deep Q-neural network (DQN)
  - Atari, Alpha-Go, Fault-tolerant quantum computation
- Policy gradient (A3C)
  - Locomotion, Starcraft II
  - Directly optimize $\pi_\theta(a|s)$
  - Off-policy training

Training Process

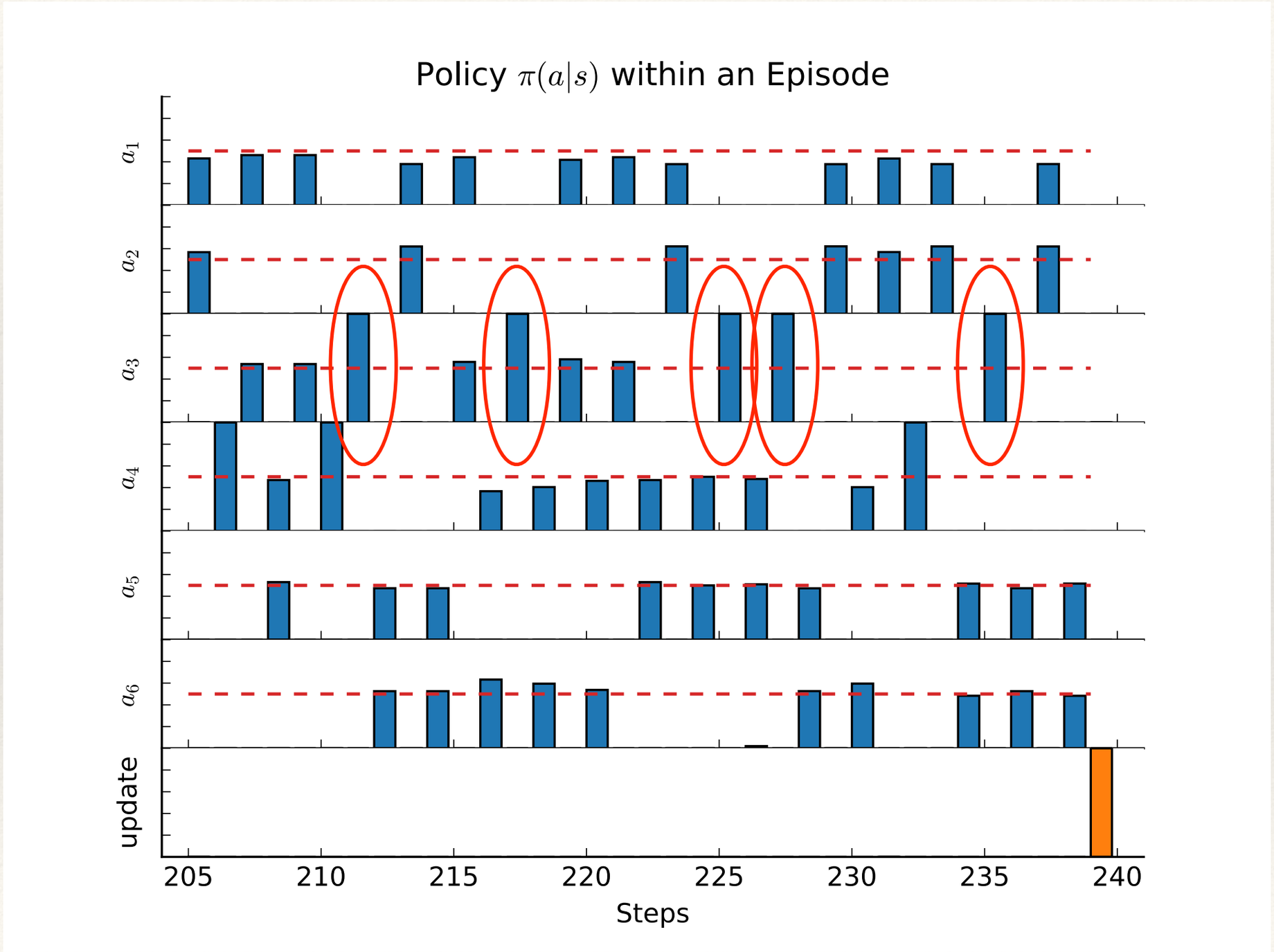



Loop Length Distribution versus Training Stage
I
II
III
IV
0
20
40
60
80
100
120
140
160
Loop Length

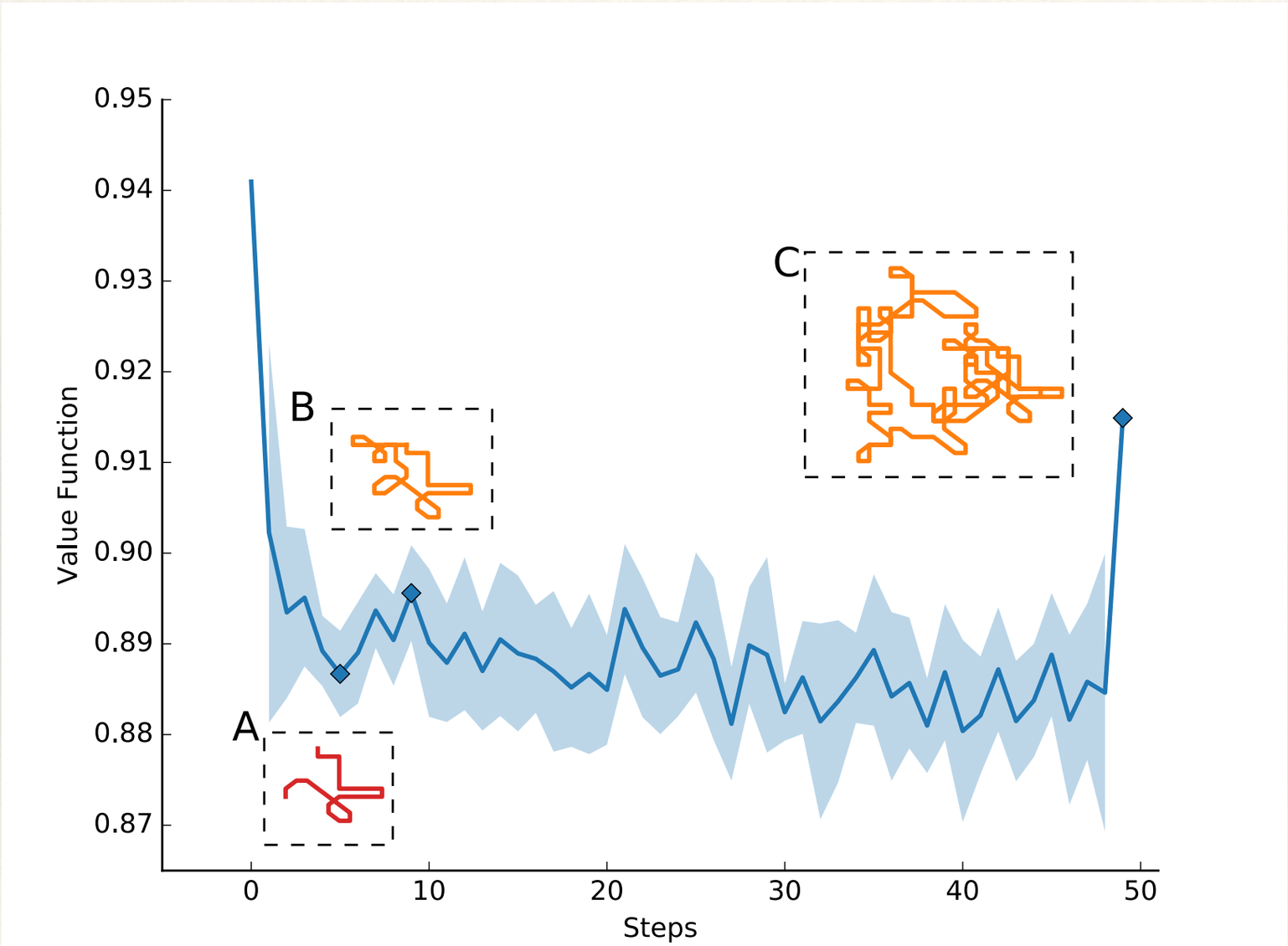
Learned Policy



Learned Policy



State-Value Function



- (A) **Local minimum** when endpoints are separated apart.
- (B) **Local maximum** when two ends move close to each other.
- (C) **Closed loop emergences.** The value function shows clear jump and large reward is expected.

Value: Future reward that an agent would receive by taking an action in a particular state

Physics Learned

- To propose a move, the machine has to
  - learn the ice-rule constraint
  - build a cluster without creating additional defects
  - learn to distinguish between open and closed loops

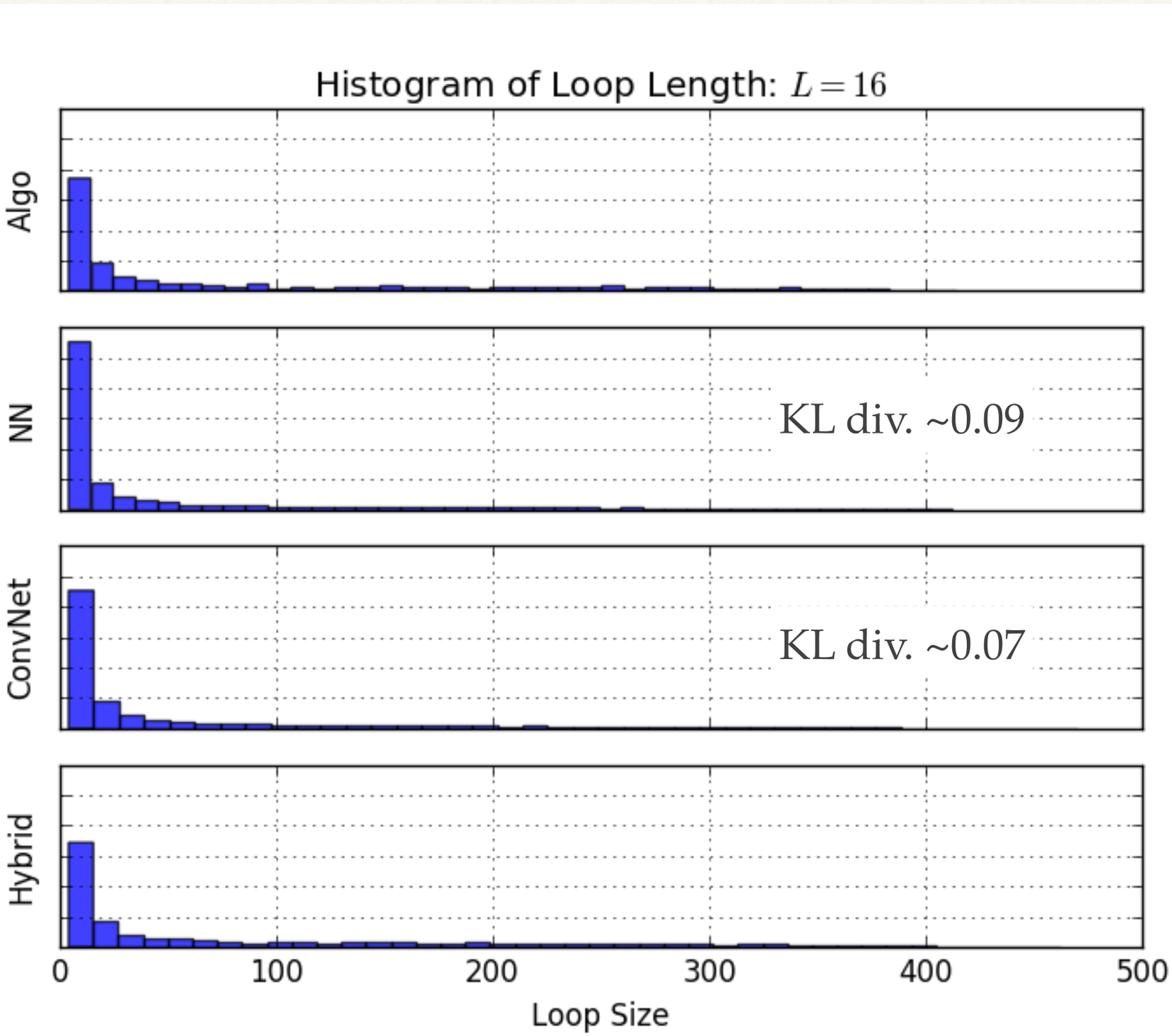
Inference

L=16

Network	Acceptance	Efficiency	Loop Size
Algorithm	96.2%	33 $\sim$ 96 %	4 $\sim$ 480 (96)
NN Policy	94.1%	24 $\sim$ 81 %	4 $\sim$ 436 (51)
CNN Policy	96.2%	31 $\sim$ 90 %	4 $\sim$ 518 (239)

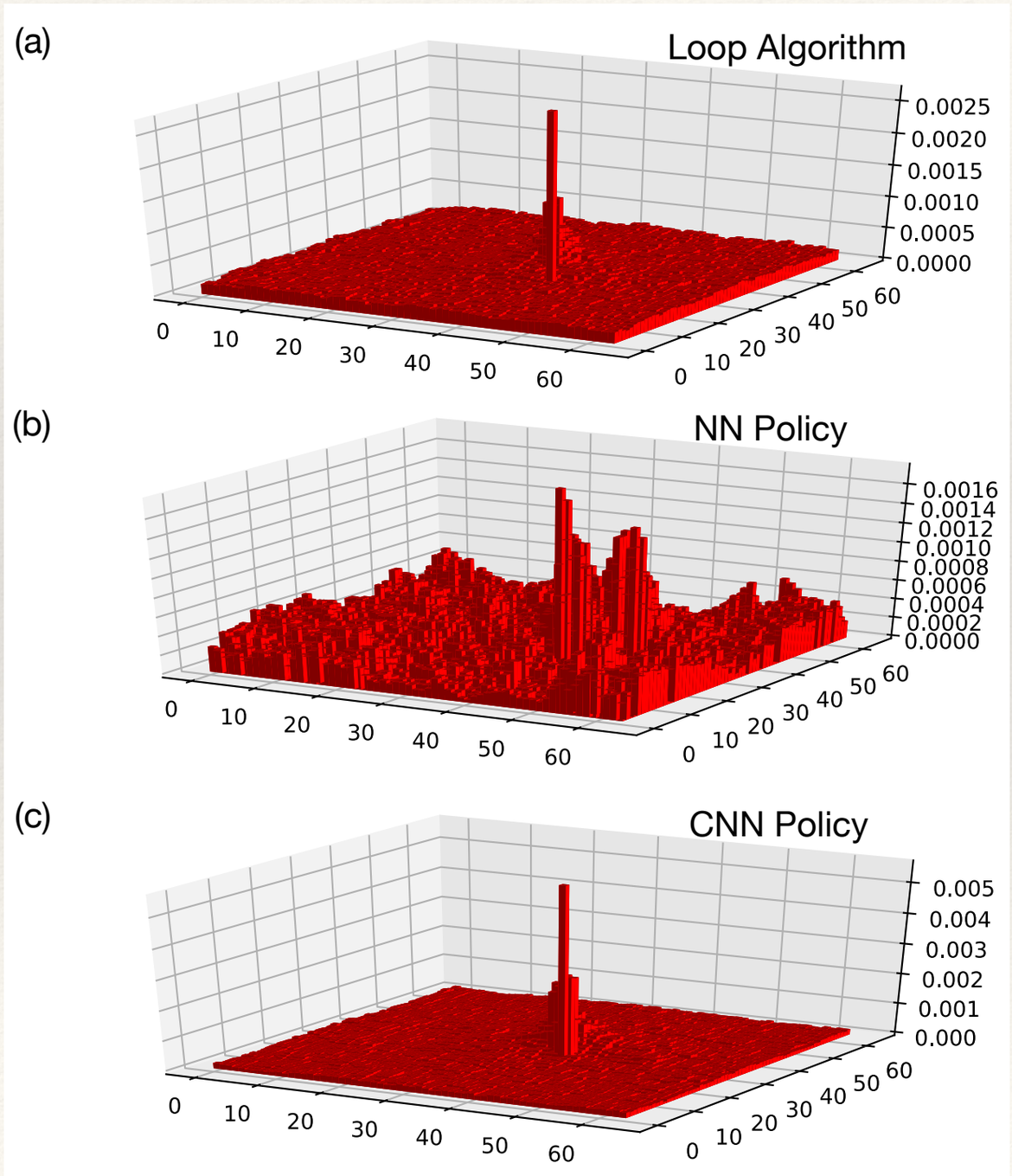
- Use the trained policy to generate configurations in MCMC
  - NN Policy: local observations only
  - CNN Policy: local+global observations
  - Efficiency= updated/visited

Loop distribution

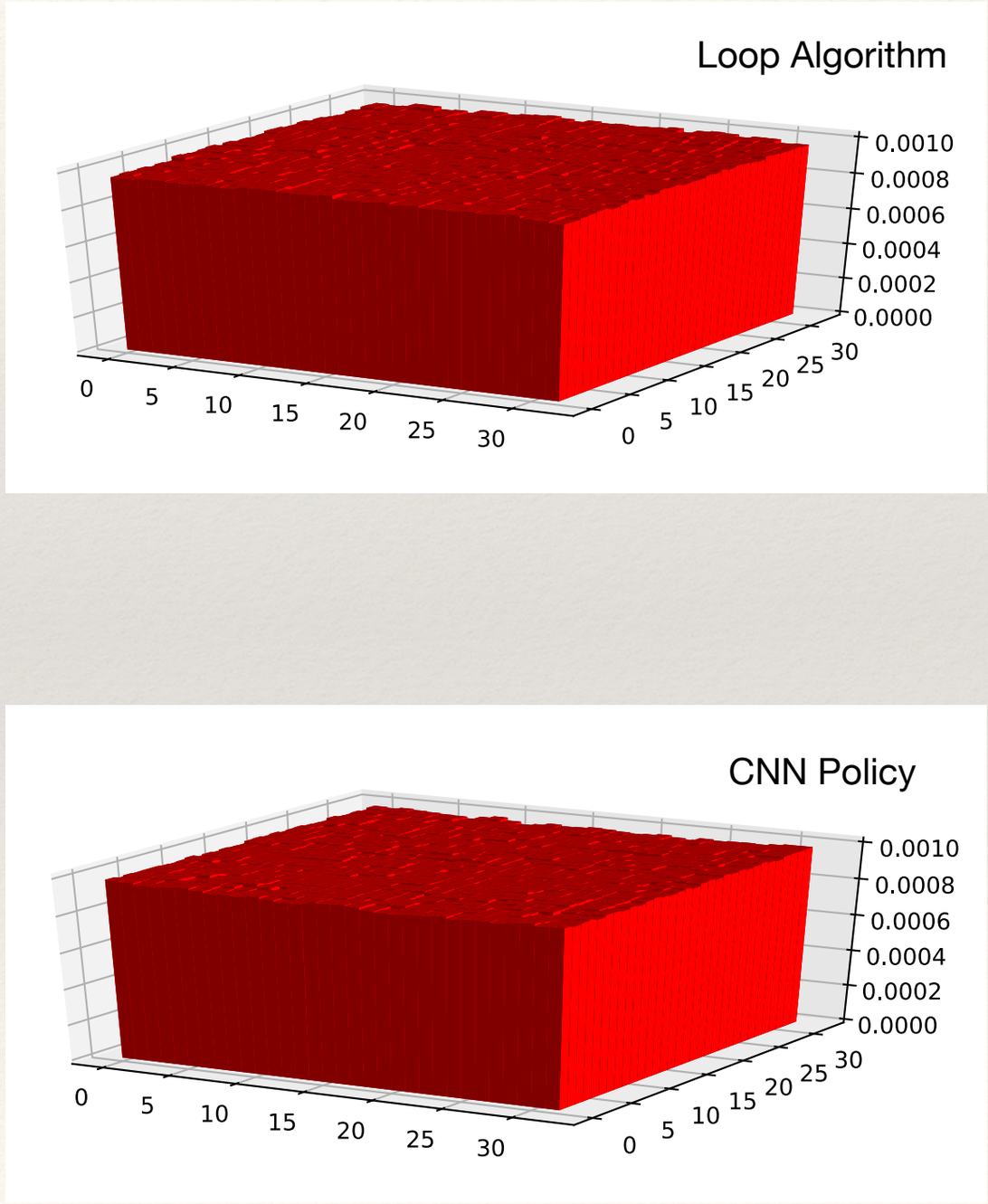


Markov Property

Same initial state and site

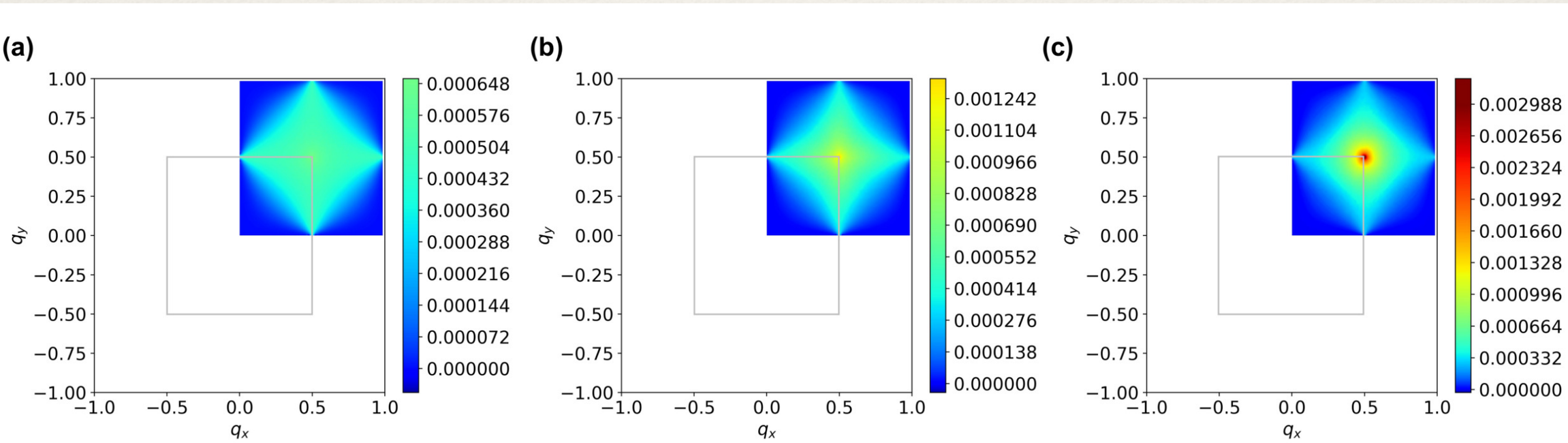


Random start



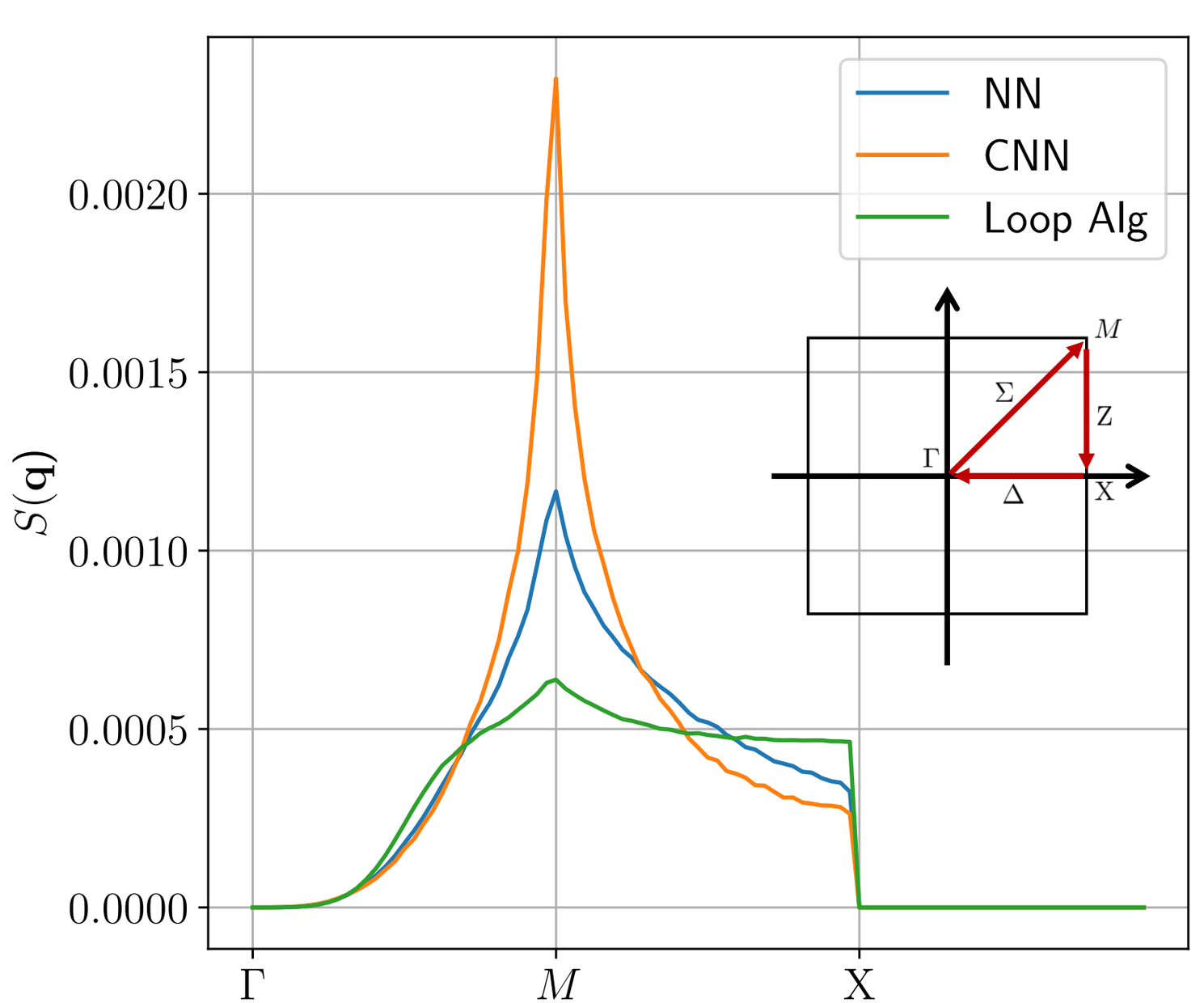
Structure Factors

Loop Algo. NN Policy CNN Policy



$$S(\mathbf{q}) = \left\langle \sigma_{\mathbf{q}} \sigma_{-\mathbf{q}} \right\rangle, \quad \sigma(\mathbf{q}) = \sum_r s_{\mathbf{r}} e^{-i\mathbf{q}\cdot\mathbf{r}},$$

Structure Factors

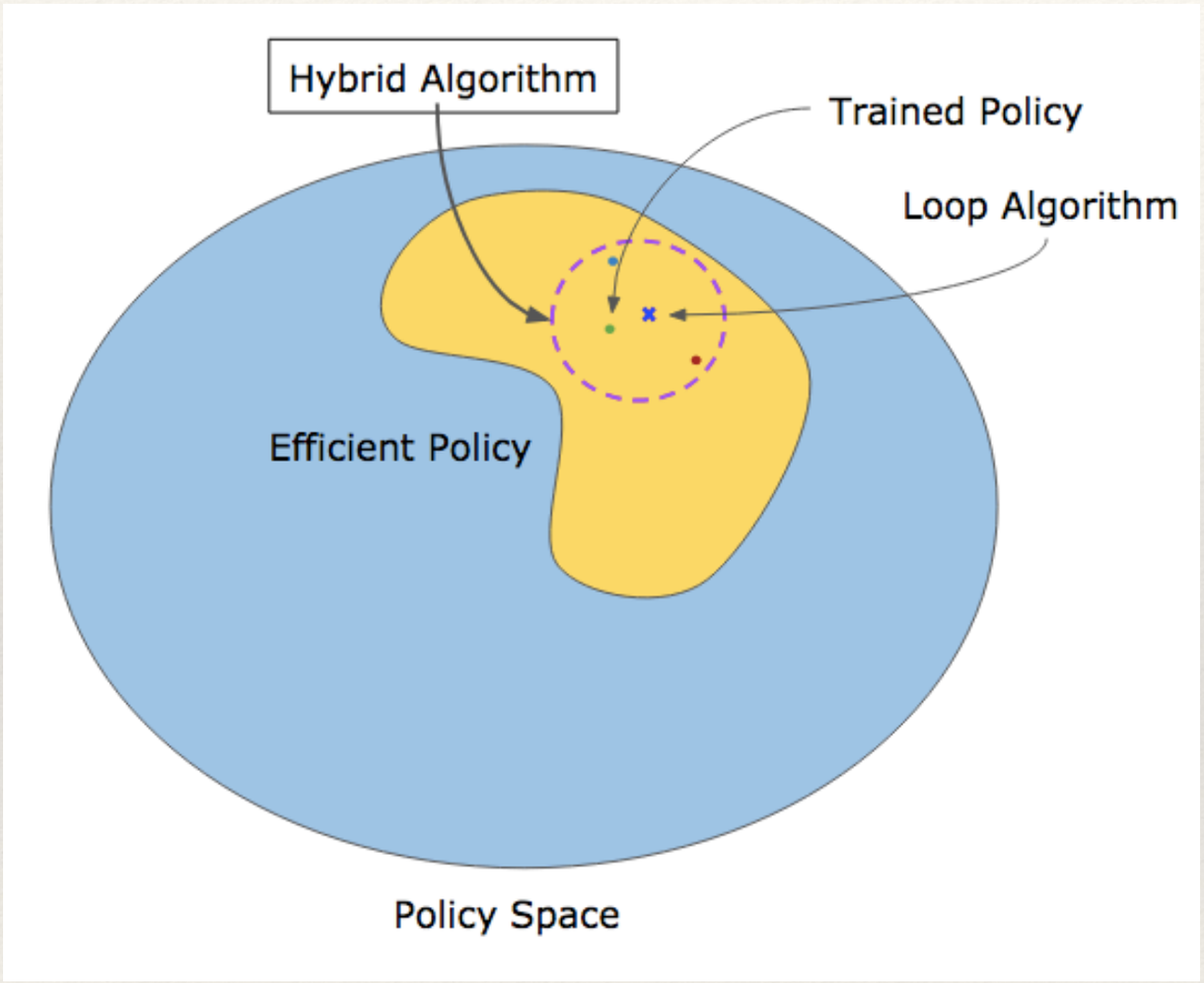


- NN and CNN policies generate stronger correlations at M point
- Sampling is biased

Caveats

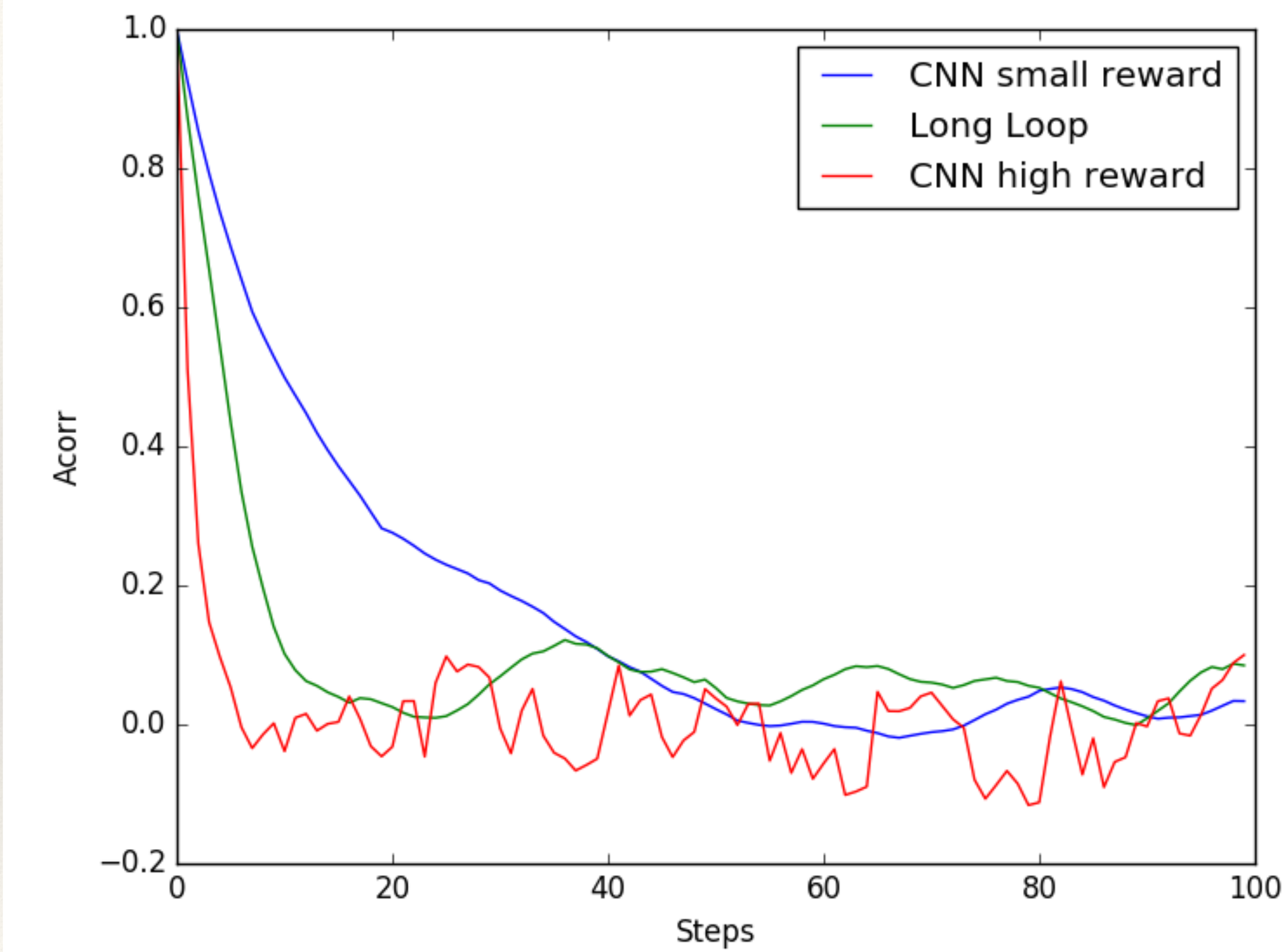
- Can not prove/show detailed balance.
- Unable to obtain the reverse transition probability.
- The algorithm is ergodic in space, not in phase space.

Hybrid Algorithm



- Mix-in with other update schemes
  - Loop algorithm
  - Generative models
  - Policy-Guided Monte Carlo (on-policy)
- Parametrize the policy with an invertible neural network

G. Torlai and R. G. Melko, Phys. Rev. B 94, 165134 (2016)
L. Huang and L. Wang, Phys. Rev. B 95, 035105 (2017)
L. Wang, Phys. Rev. E 96, 051301 (2017)
T. Bojesen, Phys. Rev. E 98, 063303 (2018)


CNN small reward
Long Loop
CNN high reward
Acorr
Steps
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
0
20
40
60
80
100

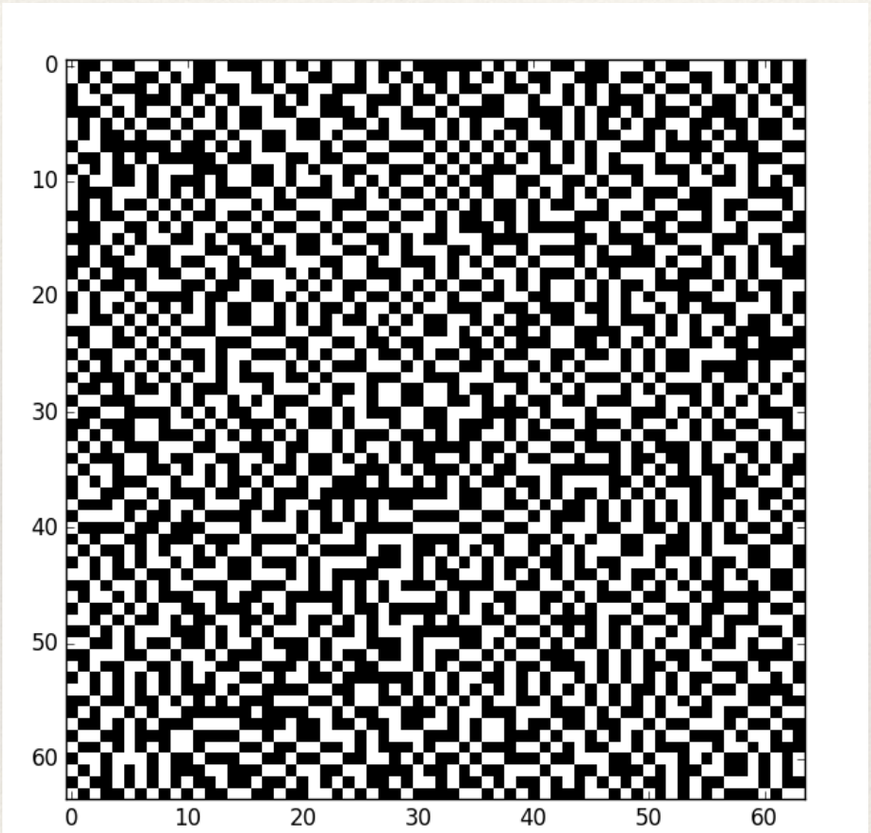
Summary

- Build an interface between machine and physical model based on Markov decision process.
- Loop-like update pattern emerges due to the ice rule.
- Both the ice rule (local) and closed-loop condition (global) are learned.
- From policy distribution, we can observe how the agent makes decisions.
- Not yet a legitimate MC update. Mixed-in with other algorithm necessary.

We just invented an ice machine…

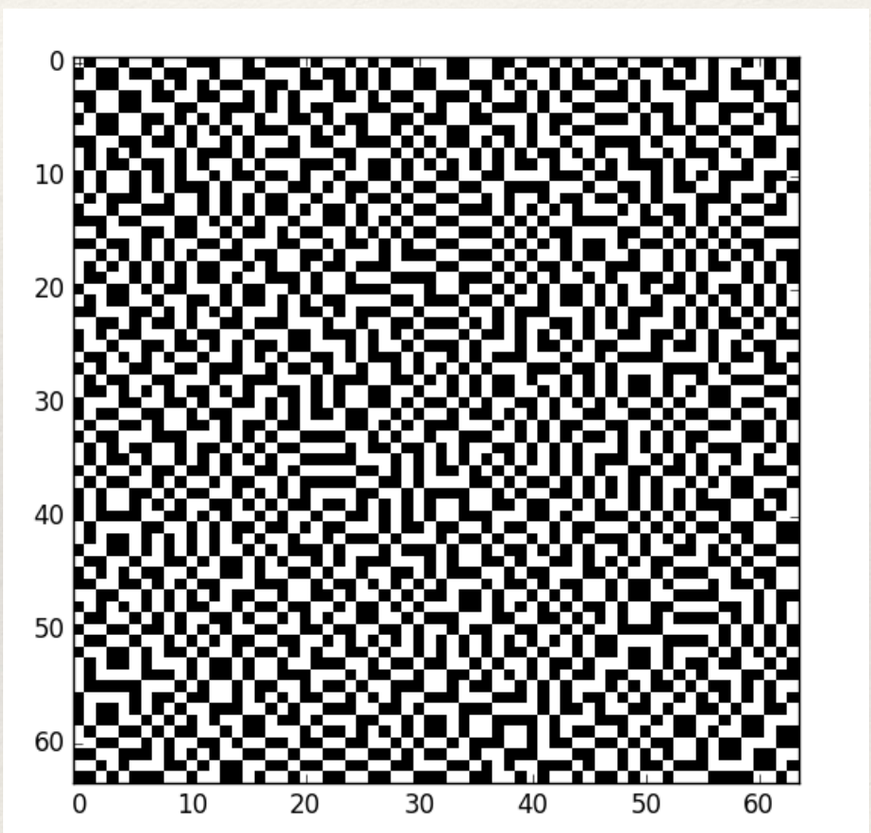
WGAN

WGAN



Defect Density = 0.05 (~200 sites)

Ice State



Defect Density = 0