Think, Act, and Learn with An Episodic Memory Graph

Ge Yang, Amy Zhang[1], Ari Morcos[1], Joelle Pineau[1], Pieter Abbeel[2], Roberto Calandra[1]

1. Facebook AI Research 2. UC Berkeley EECS

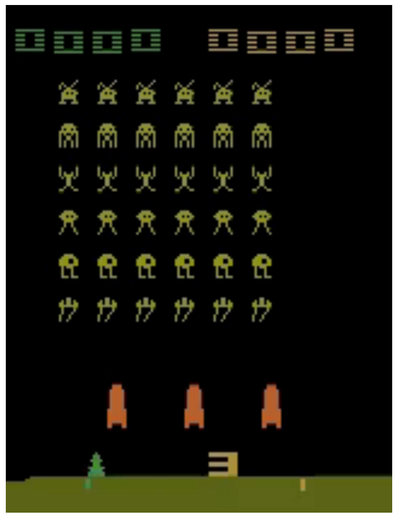
BERKELEY ARTIFICIAL INTELLIGENCE RESEARCH

A hallmark of intelligence is

1. achieve goals in a variety of circumstances
2. quickly adapt to changing situations

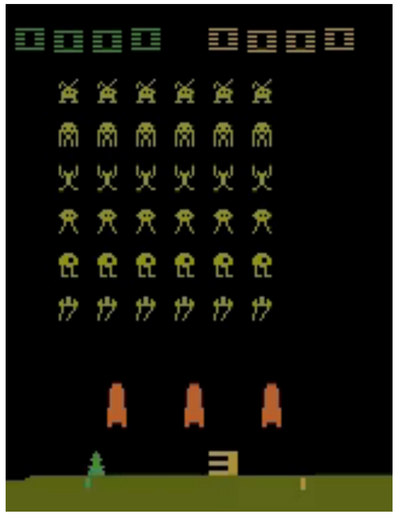
[1] Racanière et al, 2017

DQN



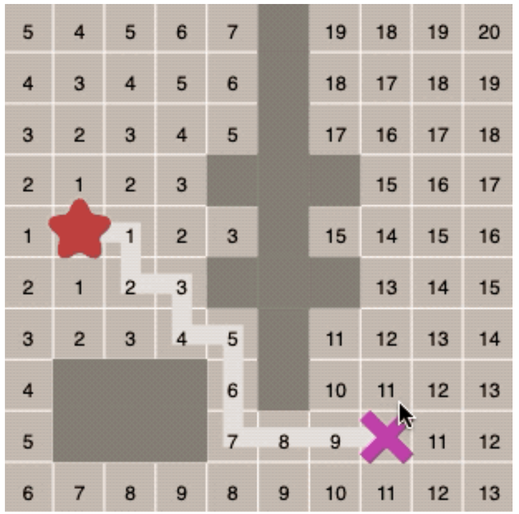
+ reactive, super-human (avg)
~ 1 week per task
- slow to learn/adapt

DQN



+ reactive policy
~ 1 week per task
- slow to learn/adapt

Graph Search



+ dynamic goals
+ adapt quickly
- discrete states
- perfect model

Our Question: Can we learn

An Agent

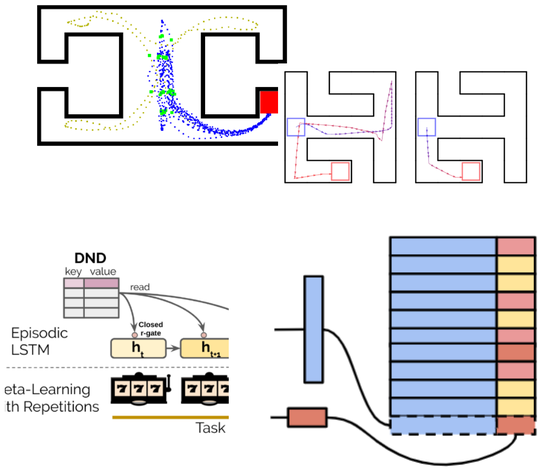
1. generalizes over a dynamic set of goals
2. adapt quickly in a changing environment?

Related Works

+ fast adaptation
- still expensive to train
- has trouble remembering

Meta-RL

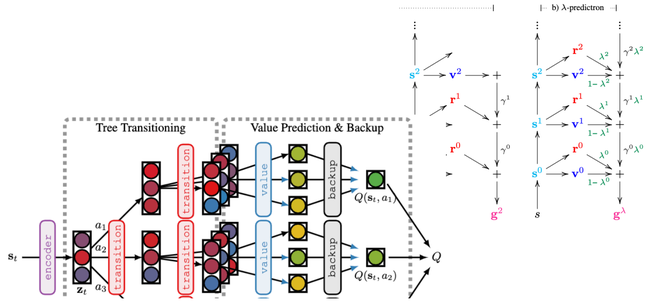
- RL^2 / L2RL
- NEC



+ long-term value
+ sample efficient
- single task
- not real dynamics — no adaptation

Value Prediction Network

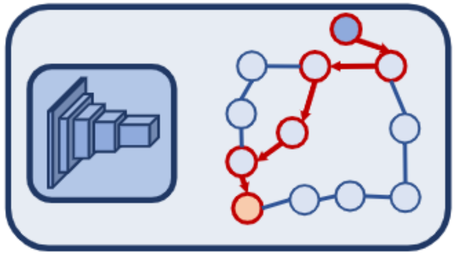
- Predictron/VPN/I2A
- TreeQN/MuZero



+ fast adaptation
- does not learn values
- weak policy
- not end-to-end

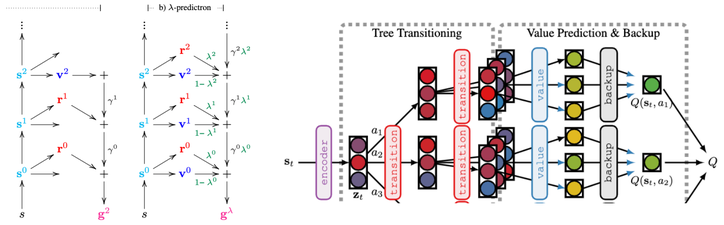
Topological Maps

- SPTM
- SoRB
- Plan2Vec

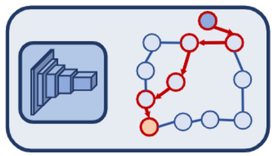


Related Works

Value-prediction Networks



Topological Maps



UVPN

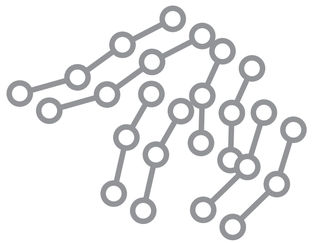
Integrate planning and value-learning

+ Value-learning important for generalization
+ Graph enables structural learning in complex domains
+ episodic memory enable fast adaptation

Background: Semi-Parametric Topological Memory

Learning Phase

Past Experience



Adaptation Phase

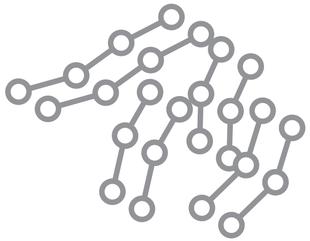
[2] Savinov et al, 2018

Background: Semi-Parametric Topological Memory

Learning Phase

Adaptation Phase

Past Experience



Local Metric

$d(o, o')$

Inverse Model

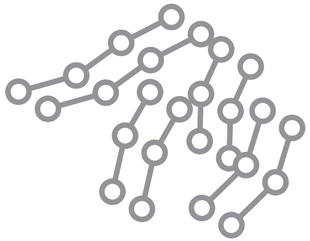
$l(a|o, o')$

[2] Savinov et al, 2018

Background: Semi-Parametric Topological Memory

Learning Phase

Past Experience



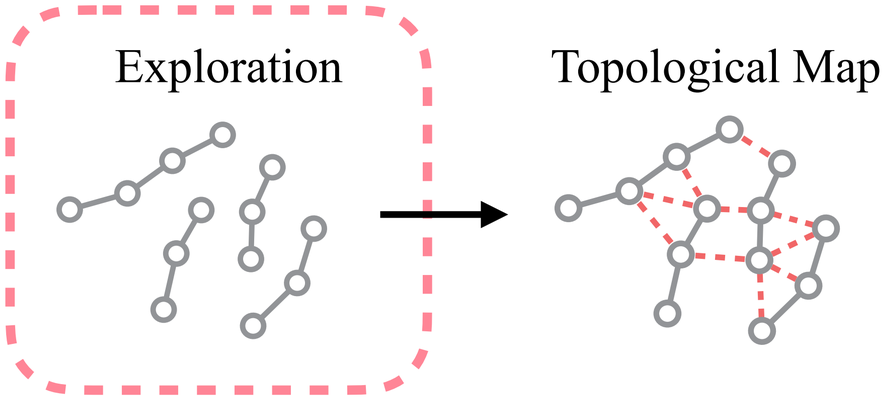
Local Metric

$$d(o, o')$$

Inverse Model

$$l(a|o, o')$$

Adaptation Phase

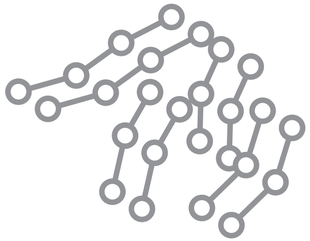


[2] Savinov et al, 2018

Background: Semi-Parametric Topological Memory

Learning Phase

Past Experience



Local Metric

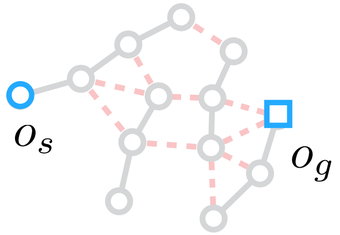
$$d(o, o')$$

Inverse Model

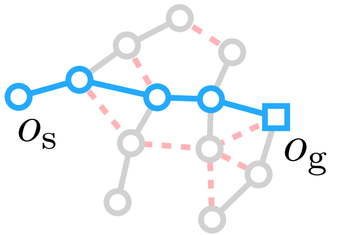
$$l(a|o, o')$$

Motion Planning

Localization



Graph Search

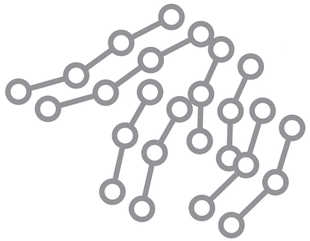


[2] Savinov et al, 2018

Background: Semi-Parametric Topological Memory

Learning Phase

Past Experience



Local Metric

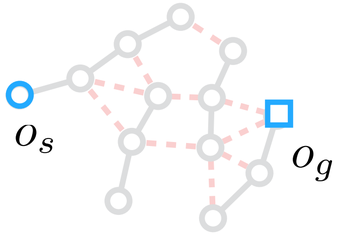
$$d(o, o')$$

Inverse Model

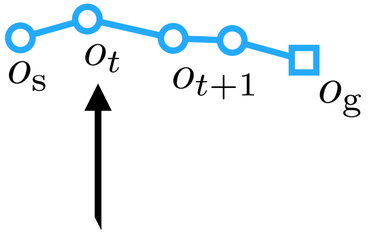
$$l(a|o, o')$$

Motion Planning

Localization



Graph Search



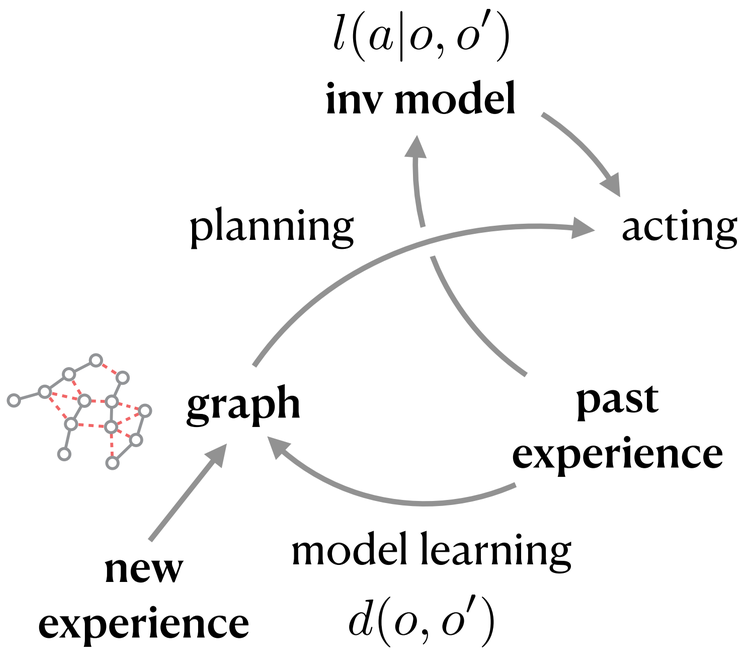
Act

$$a \sim l(a|o, o_t)$$

Environment

[2] Savinov et al, 2018

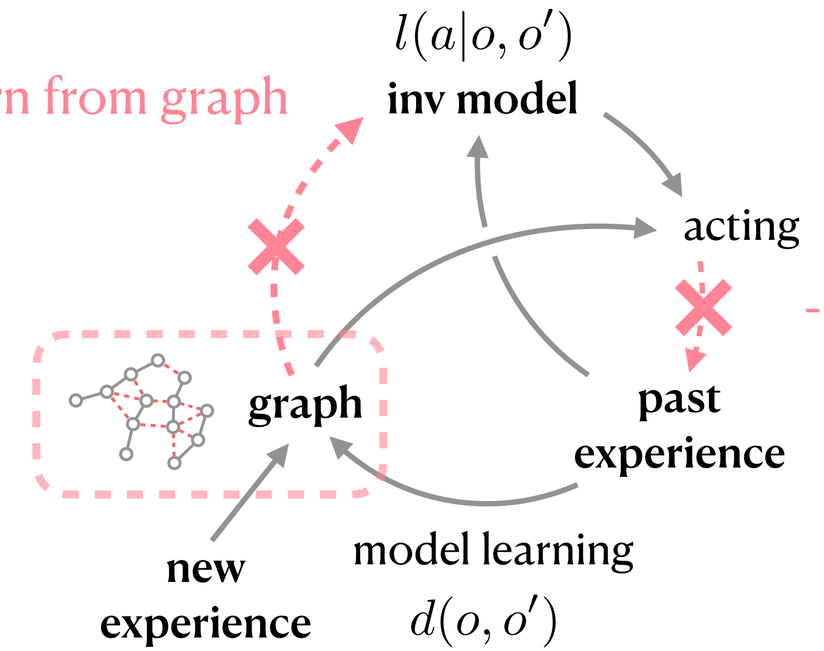
Summary


$l(a|o, o')$
inv model
planning
acting
graph
past
experience
new
experience
model learning
$d(o, o')$

What is Missing?

The Policy
- does not learn from graph

The Graph
- is fixed
- b/c size grows by $|V|^2$

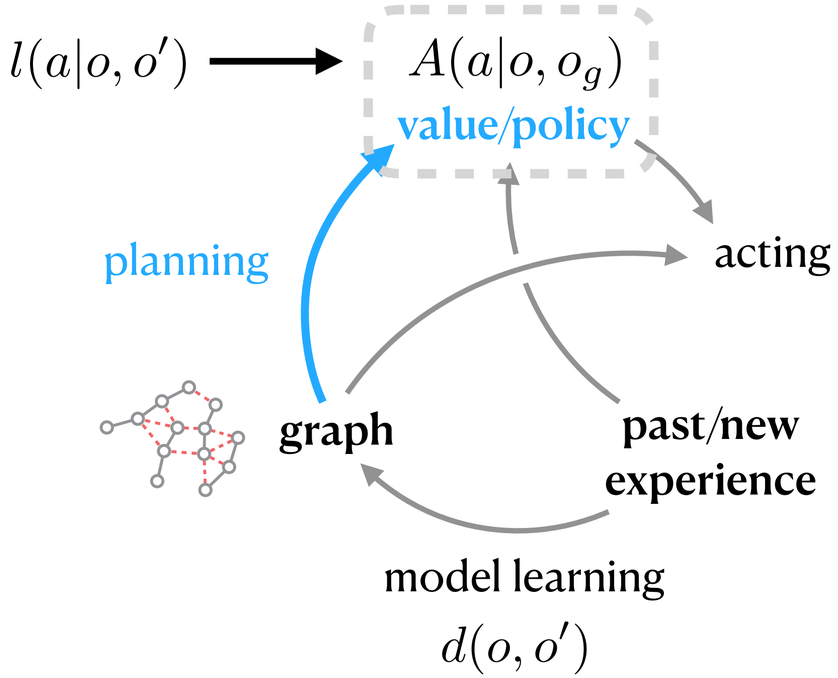


How can we close the loop?

- not a loop

[2] Savinov et al, 2018

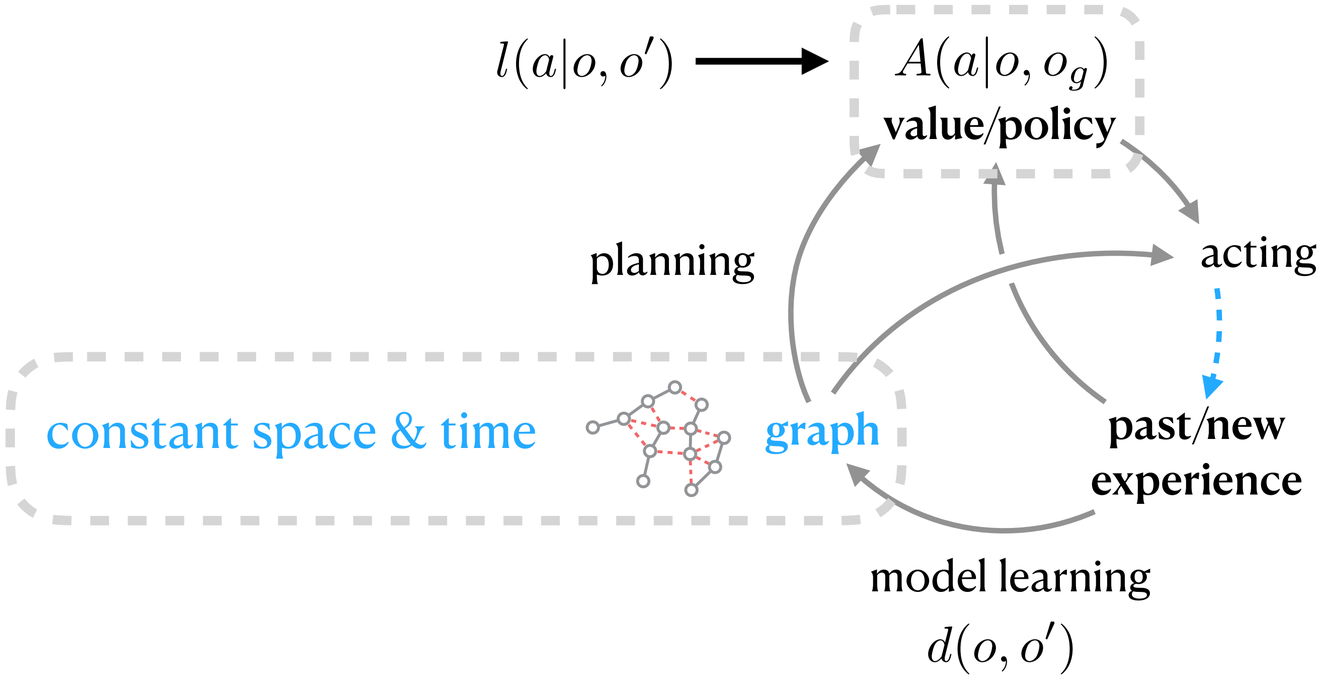
Our Approach



How can we close the loop?

1. long-range values from the graph
   - via Dyna-style unroll
2. continuously improve the graph

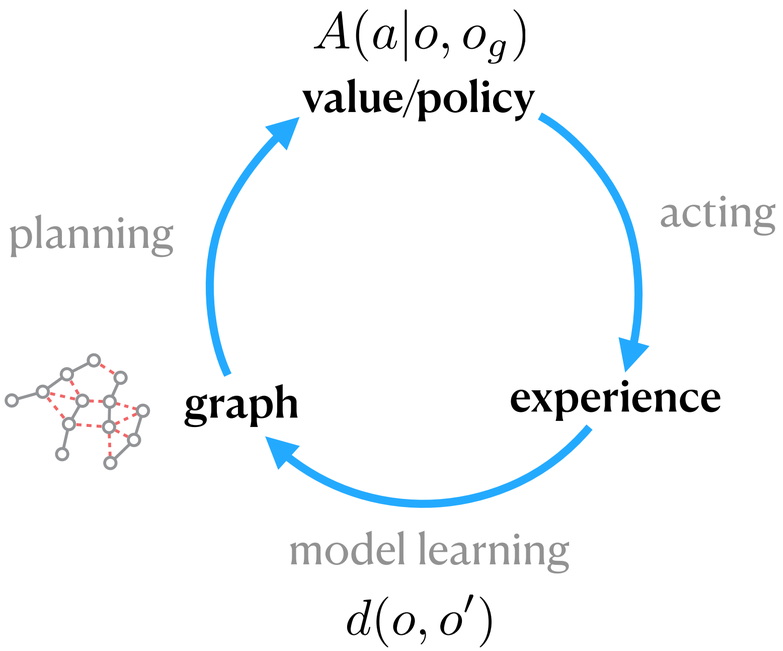
Our Approach



How can we close the loop?

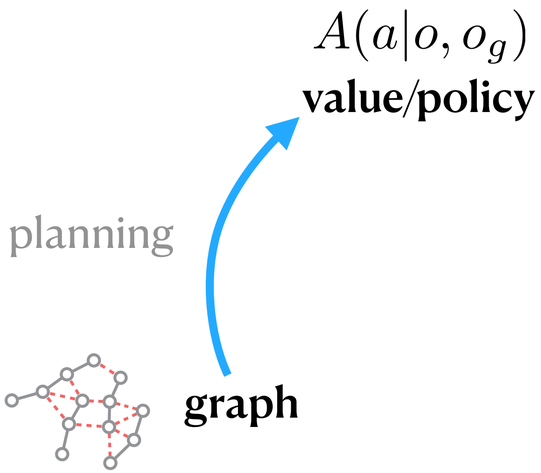
1. long-range values from the graph
   - via Dyna-style unroll
2. continuously improve the graph

Outline



1. Learning Value Estimates (UVPN)
2. A Dynamic Graphical Memory
3. Causal Structure Learning
4. Goal-Relabeled Expert Distillation (GRED)

1. Learning Value Estimates (UVPN)



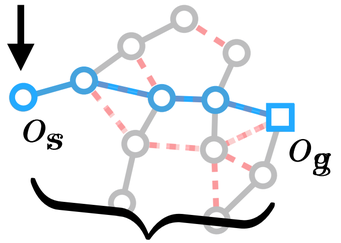
How can we learn a value estimate directly from the graph?

1. Learn $V(o, o_g)$
2. Learn $A(a \mid o, o_g)$

[2] Yang et al, 2020

How do we **1. Learn V(o, o_g)** ?

2. Learn A(a | o, o_g)



1. sample $\langle o, o_g \rangle$
2. plan with Dijkstra's
3. Learn V(o, o_g) by regressing towards $V(o, o_g) = -\sum_i d(o_i, o_{i+1})$

Now, does this work?

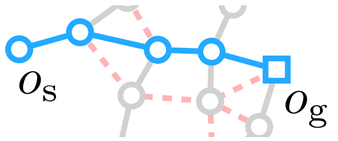
[2] Yang et al, 2020

How do we **1. Learn V(o, o_g)** ?

2. Learn A(a | o, o_g)

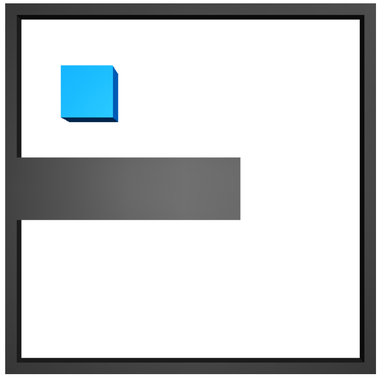
1. sample $\langle o, o_g \rangle$

2. plan with Dijkstra's

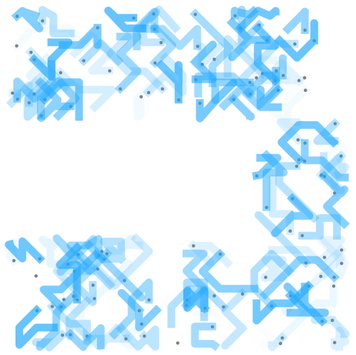


3. Learn V(o, o_g) by regressing towards $V(o, o_g) = -\sum_i d(o_i, o_{i+1})$

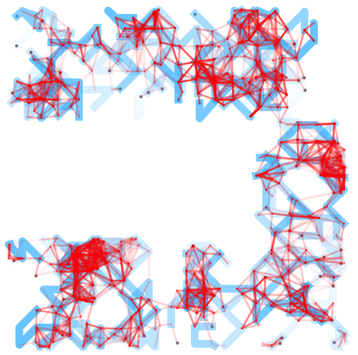
Now, does this work?



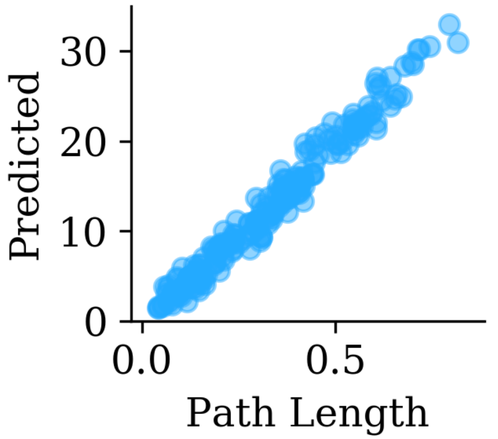
Data



Graph



Value Prediction

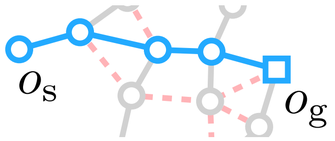


How do we **1. Learn V(o, o_g)** ?

2. Learn A(a | o, o_g)

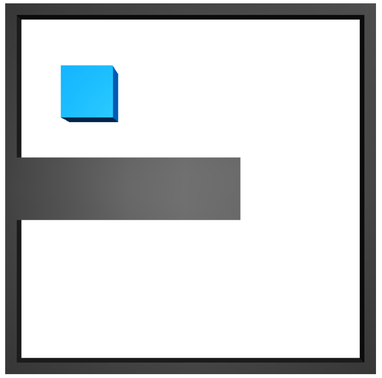
1. sample $\langle o, o_g \rangle$

2. plan with Dijkstra's



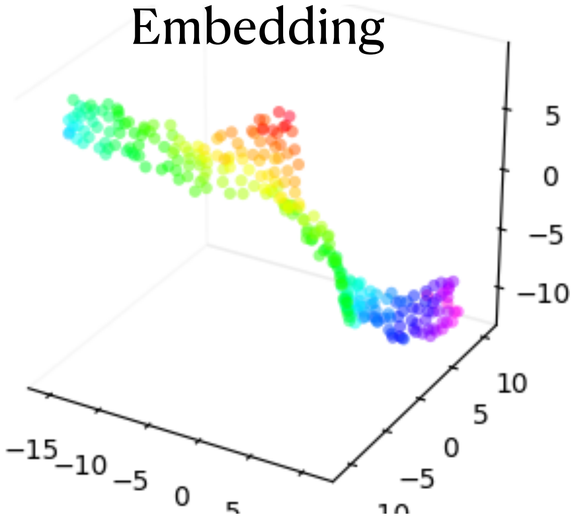
3. Learn V(o, o_g) by regressing towards $V(o, o_g) = -\sum_i d(o_i, o_{i+1})$

Now, does this work?

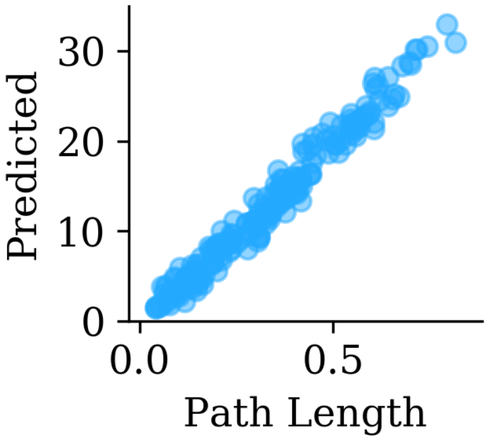


Grid

Embedding



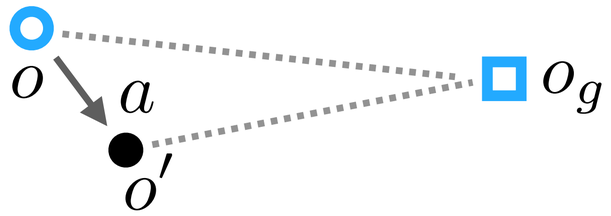
Value Prediction



1. Learn V(o, o_g)

How do we **2. Learn A(a | o, o_g)** ?

1. sample $\langle o_t, a_t, o_{t+1} \rangle$
2. sample random goal o_g (relabel goal)
3. Learn A(a | o, o_g) by regressing towards

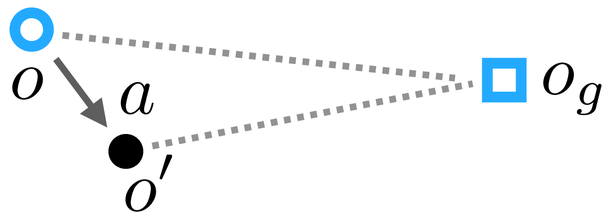


$$A(a|o, o_g) = V(o, o_g) - \left[V(o, o') + V(o', o_g)\right]$$

1. Learn V(o, o_g)

How do we **2. Learn A(a | o, o_g)** ?

1. sample $\langle o_t, a_t, o_{t+1} \rangle$
2. sample random goal o_g (relabel goal)
3. Learn A(a | o, o_g) by regressing towards

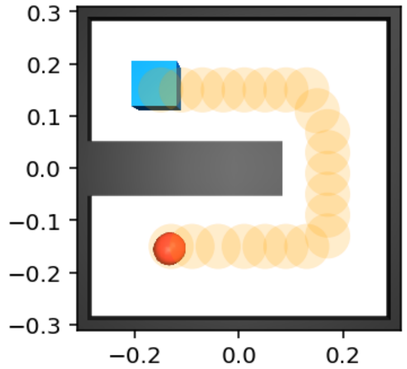


$$A(a|o, o_g) = V(o, o_g) - \left[V(o, o') + V(o', o_g)\right]$$

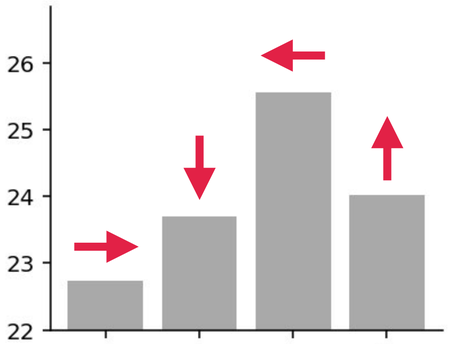
But, this won't work

agent view goal image

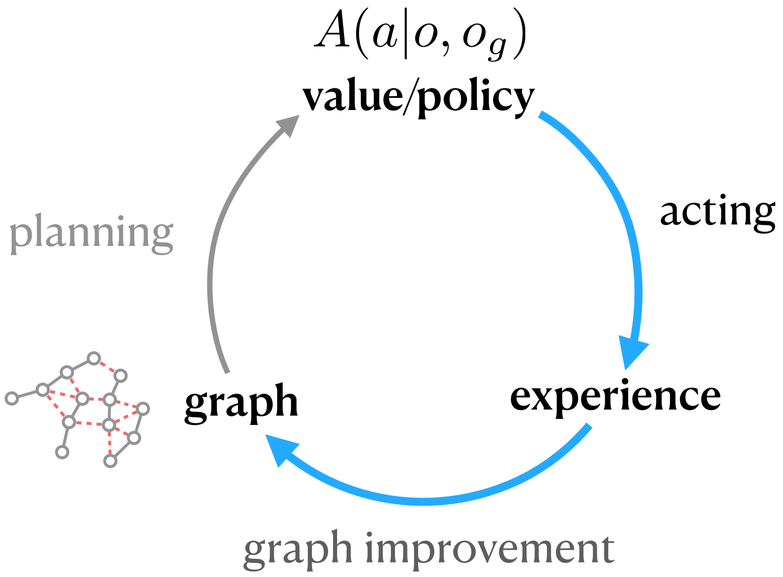
Configuration



Distance / Action

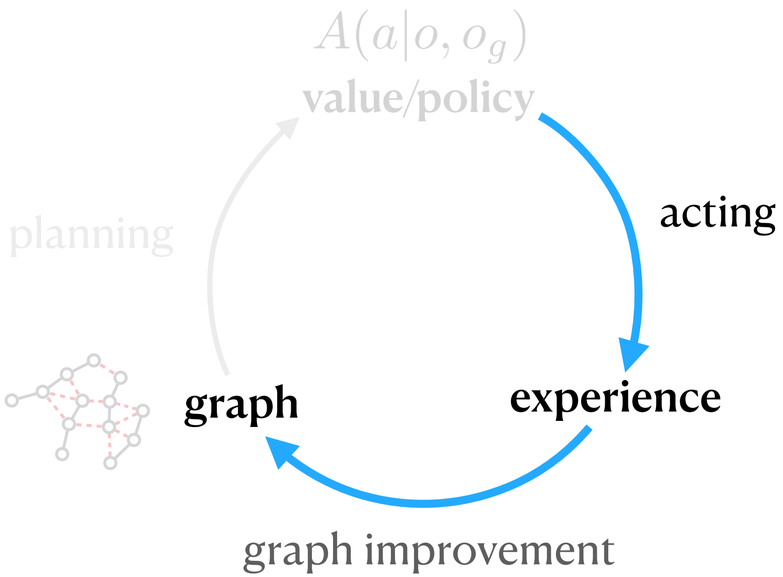


Improving the Graph with New Experience



agent view goal image

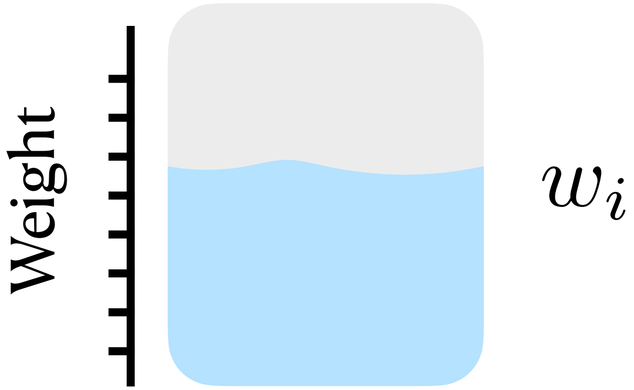
Improving the Graph with New Experience



1. Locally Weighted Cache Replacement
2. Causal Structure Learning on A Graph

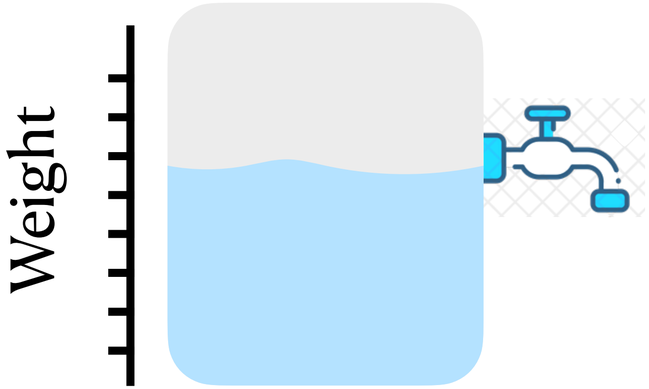
Density Weighted Cache

1. Assign a weight to each vertex.
2. On insertion, replace old node with the highest weight.



Density Weighted Cache

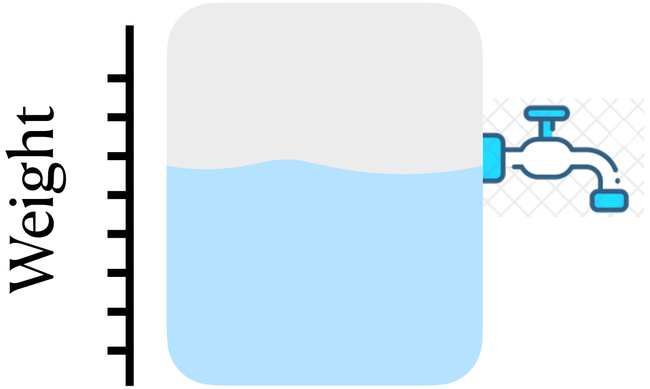
1. Assign a weight to each vertex.
2. On insertion, replace old node with the highest weight.
3. Or periodically remove nodes by a threshold



$$w_i = \sum_j \frac{1}{e_{ij}^p}$$

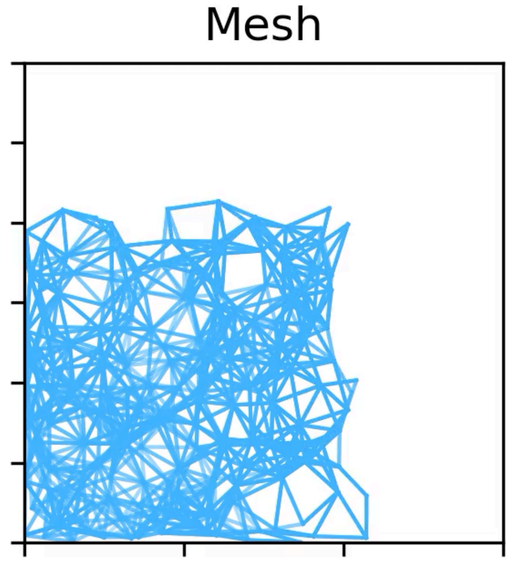
Density Weighted Cache

1. Assign a weight to each vertex.
2. On insertion, replace old node with the highest weight.
3. Or periodically remove nodes by a threshold



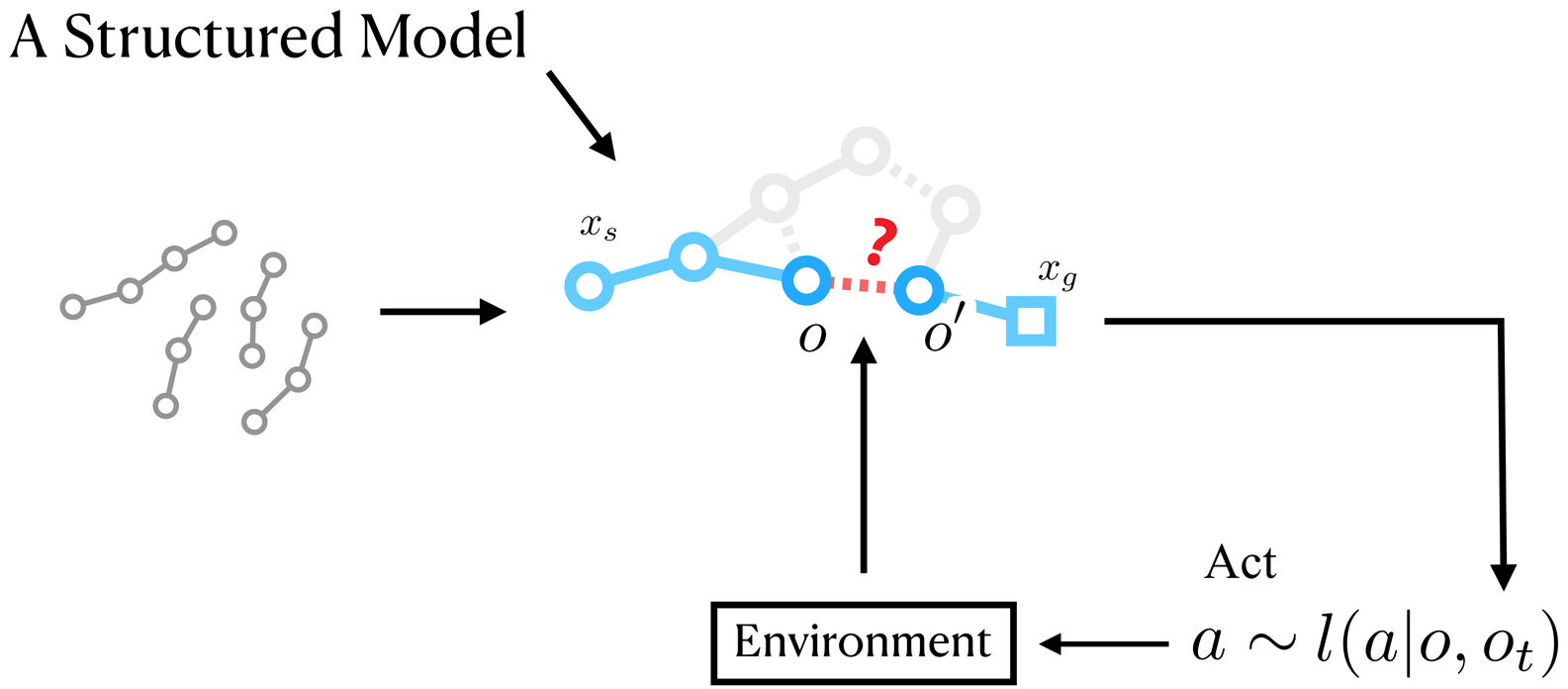
$$w_i = \sum_j \frac{1}{e_{ij}^p}$$

1 k insertion / sec
for |V| = 10 k



2. Causal Structure Learning

Improving the Graph

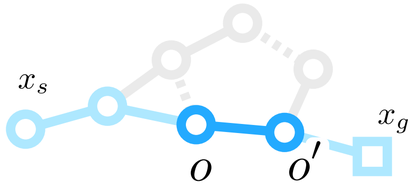


2. Causal Structure Learning

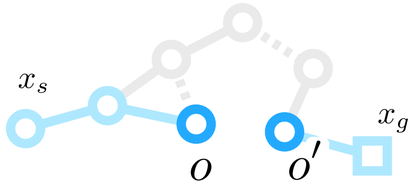
Improving the Graph

Hypothesis | Act | Outcome | Update

graph 1



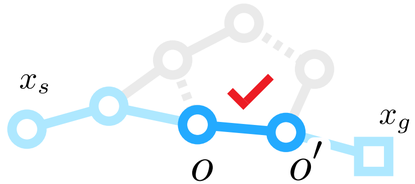
graph 2

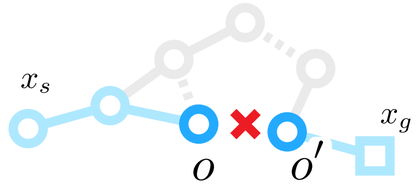


$$a \sim \pi(o, o')$$

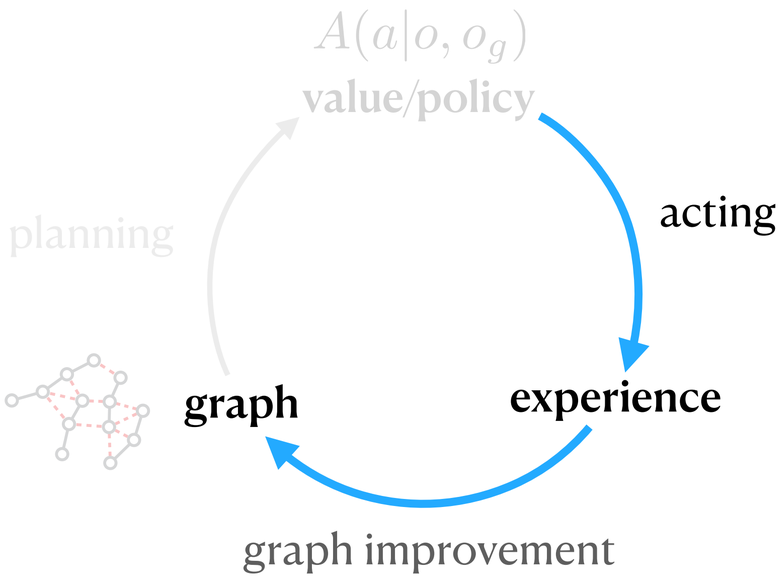
$\delta(o_{t+1}, o') < d_o$

otherwise





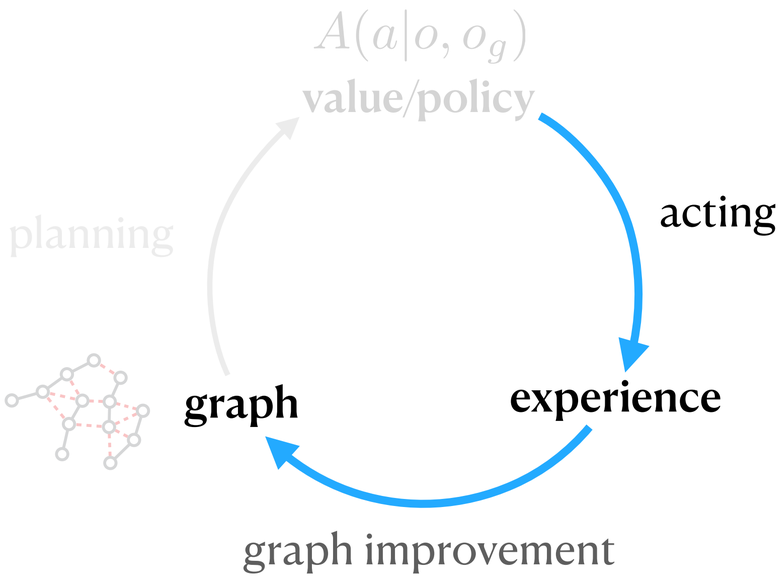
Improving the Graph with New Experience



1. Locally Weighted Cache Replacement
2. Causal Structure Learning on A Graph

How well does this work?

Improving the Graph with New Experience



Fixed Graph

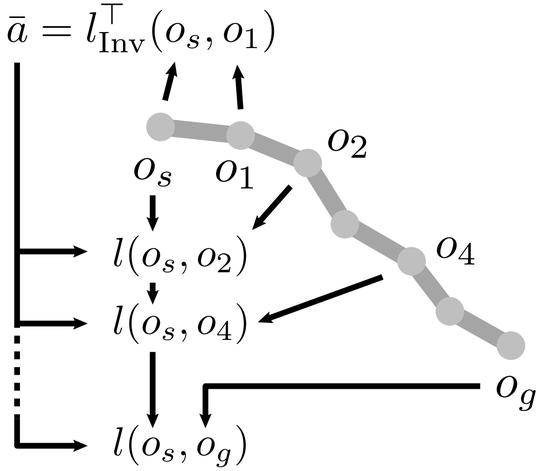
+ Dynamic Graph

+ Causal Structure Learning

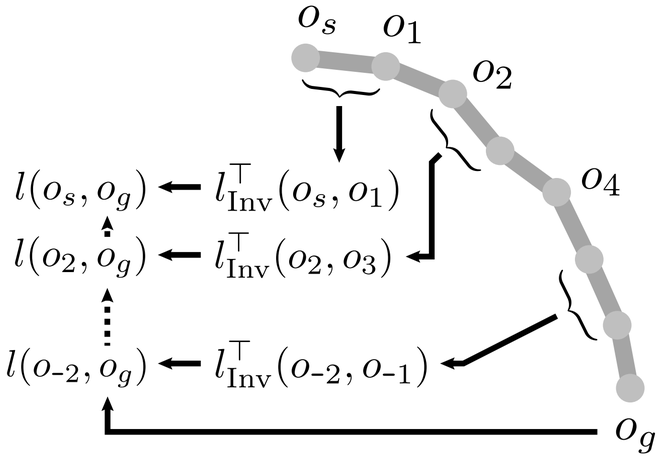
agent view goal

Goal Relabeled Expert Distillation

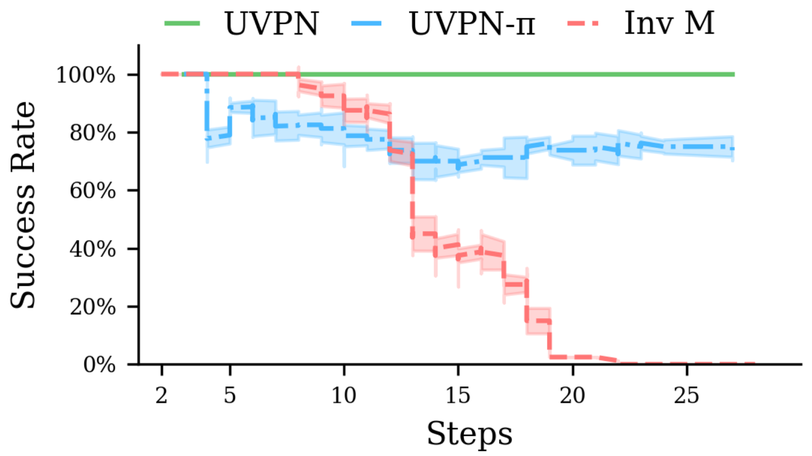
Forward Relabel



Backward Relabel



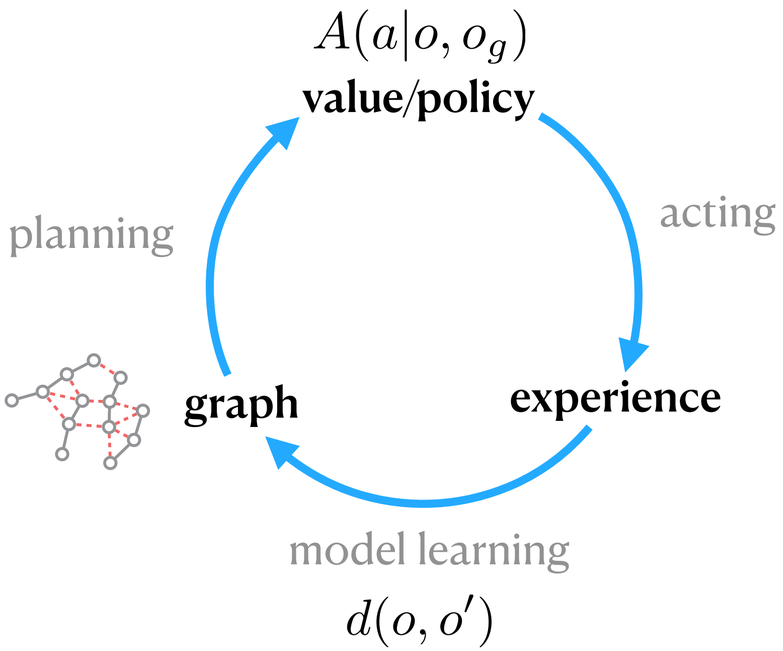
backward relabel performs better



GRED learns by distilling plans made by the search expert on the graph. The actions and the goals are both relabeled.

reactive policy gets 80% for all ranges

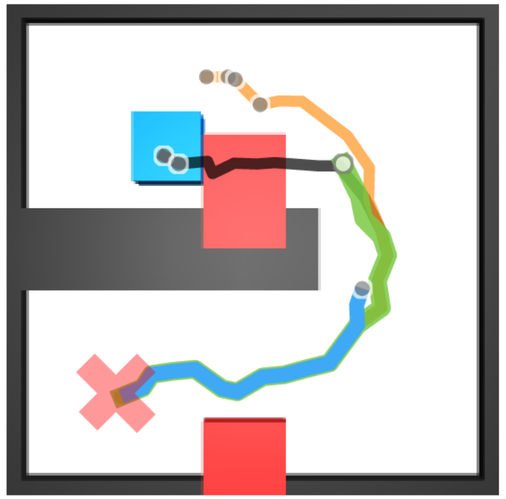
Outline



1. Locally Weighted Cache Replacement
2. Causal Structure Learning
3. Can this agent adapt to changing environments?
4. How about contact-rich environments?

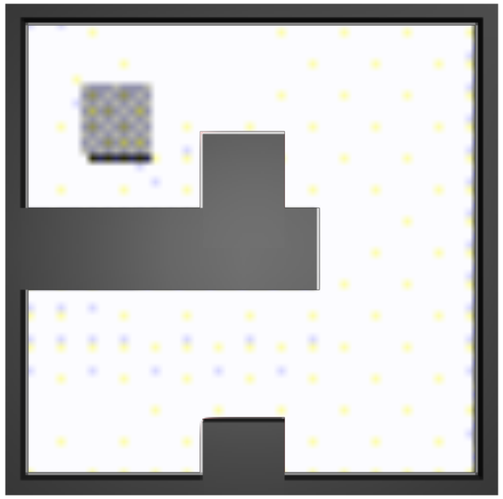
Adapting to A Changing Environment

Insert Walls



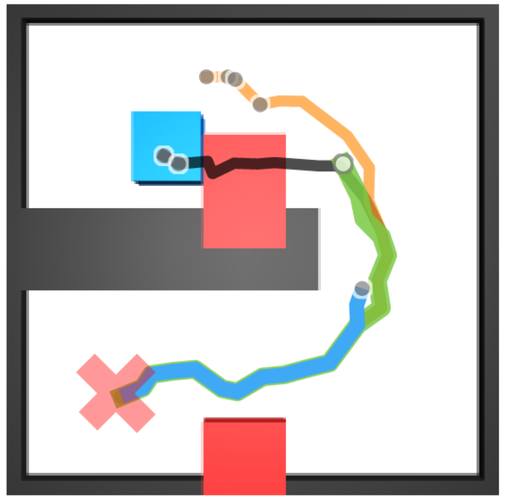
Model Update

Insert Walls



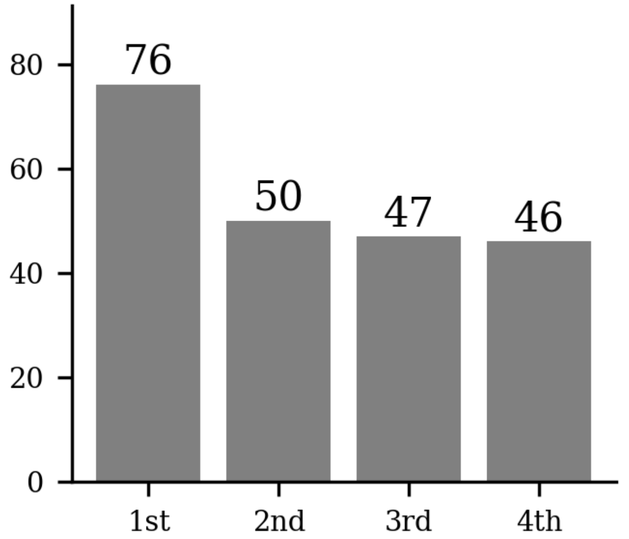
Adapting to A Changing Environment

Insert Walls



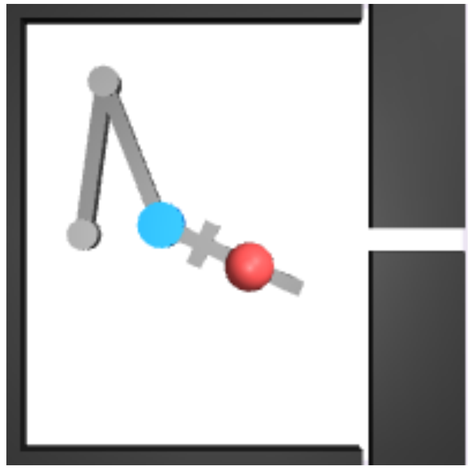
Model Update

Episode Duration

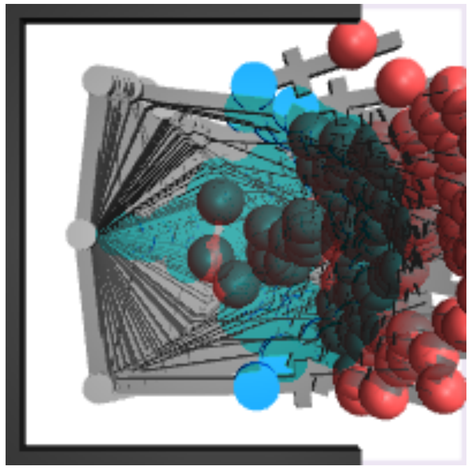


Peg-Insertion

Contact Rich Environment



View



Experience

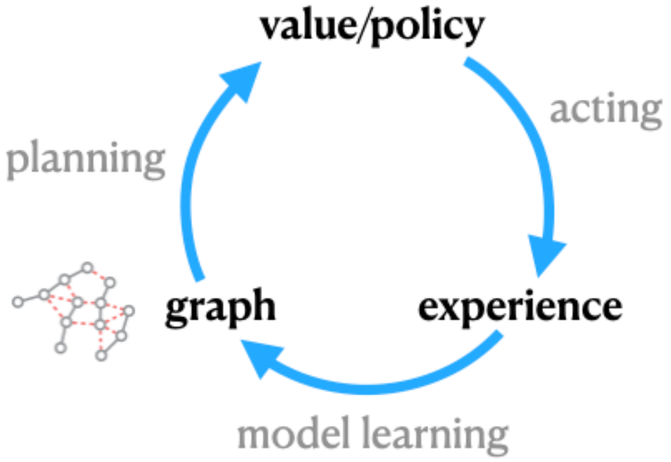
Agent View

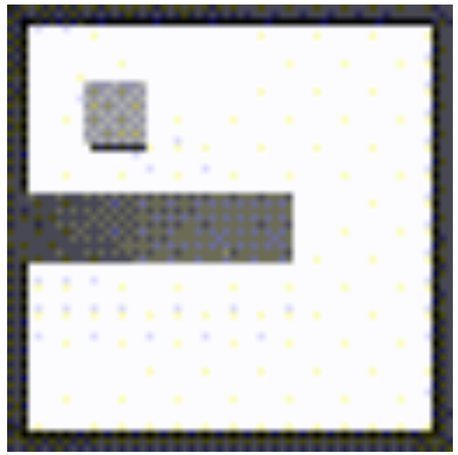
Key Takeaways

- Model Improvement is key for learning and robust adaptation
- Model-based exploration sends the agent to weak parts of the model
- It is possible to learn reactive policy from a model
- Then falling back to high-level planning when domain changes

Conclusion

We can learn flexible and robust agents
by combining model-based RL with episodic memory





Collaborators

Ge Yang, Amy Zhang, Ari Morcos, Joelle Pineau, Pieter Abbeel, Roberto Calandra





We can learn flexible and robust agents
by combining model-based RL with episodic memory