Reinforcement Learning with Limited Reinforcement

Using Bayes Risk for Active Learning in POMDPs

Finale Doshi (Cambridge, MIT)
Joelle Pineau (McGill)
Nick Roy (MIT)

Motivation

We desire an agent that

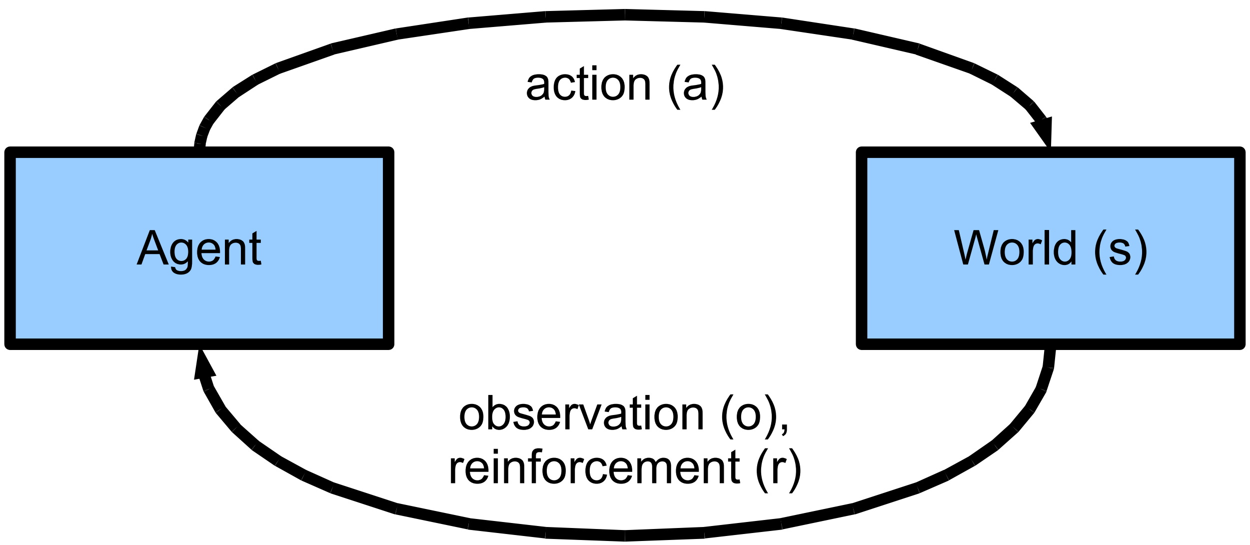
- **adapts** to new environments,
- behaves **robustly** in novel situations,
- and is **natural** to train.

How do we do this?

- risk-averse action selection
- particle filtering to track our world knowledge.
- asking for help!

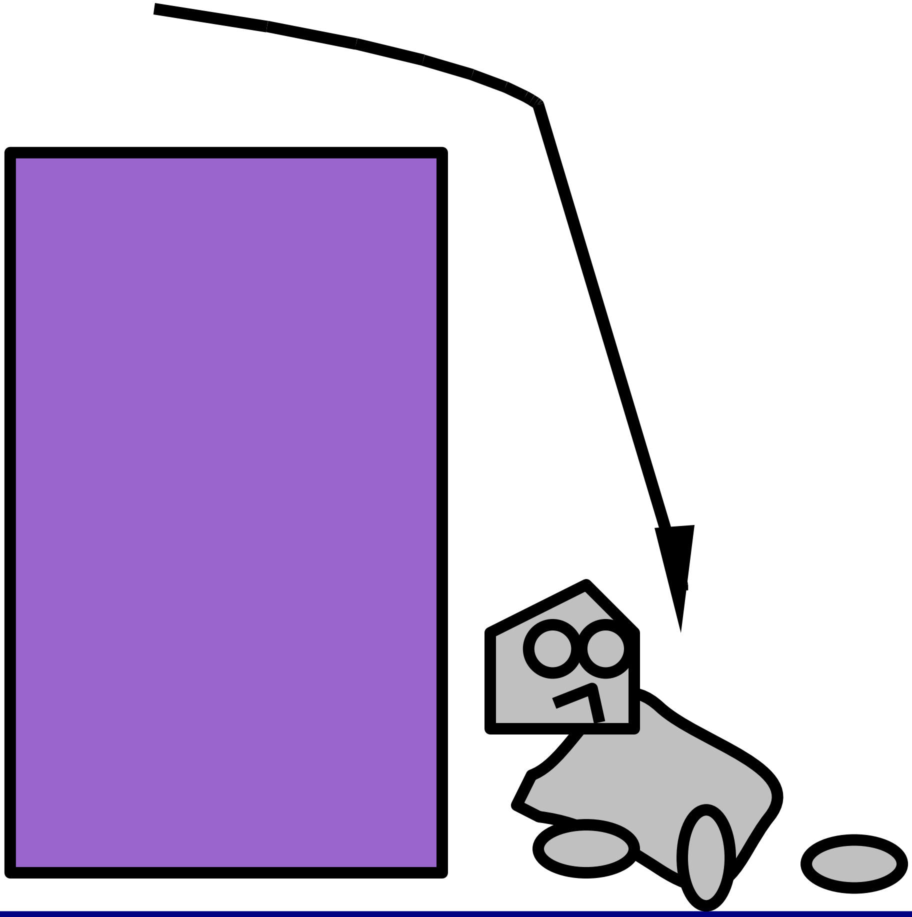
Reinforcement Learning Paradigm

- Agent performs actions, receives observations and rewards.
- Assume a Markov world; world dynamics consists of reward R(s,a), transition T(s'|s,a), and observation O(o|s,a) models.
- Agent must trade between learning about the world (exploration) and getting rewards (exploitation).



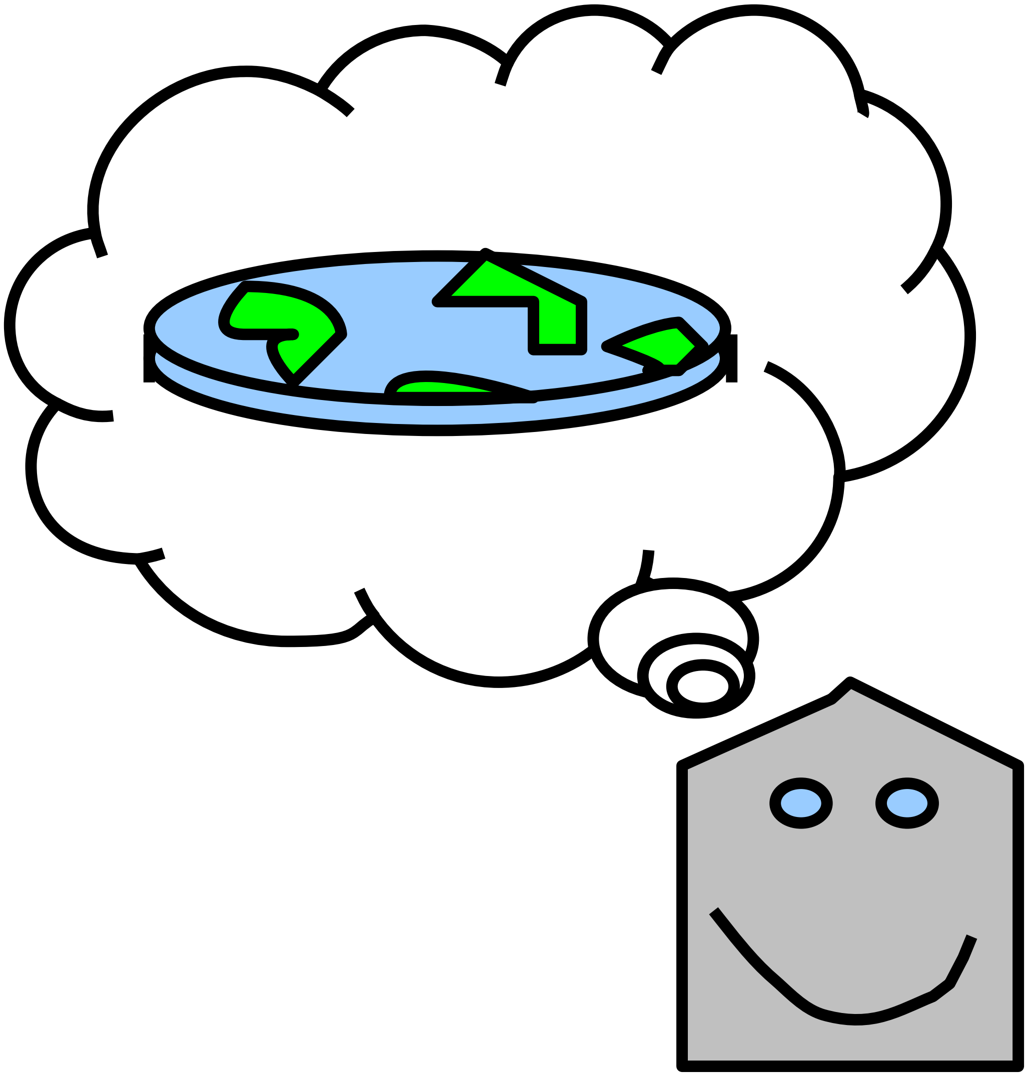
Common issues with RL

- Must make mistakes to learn.



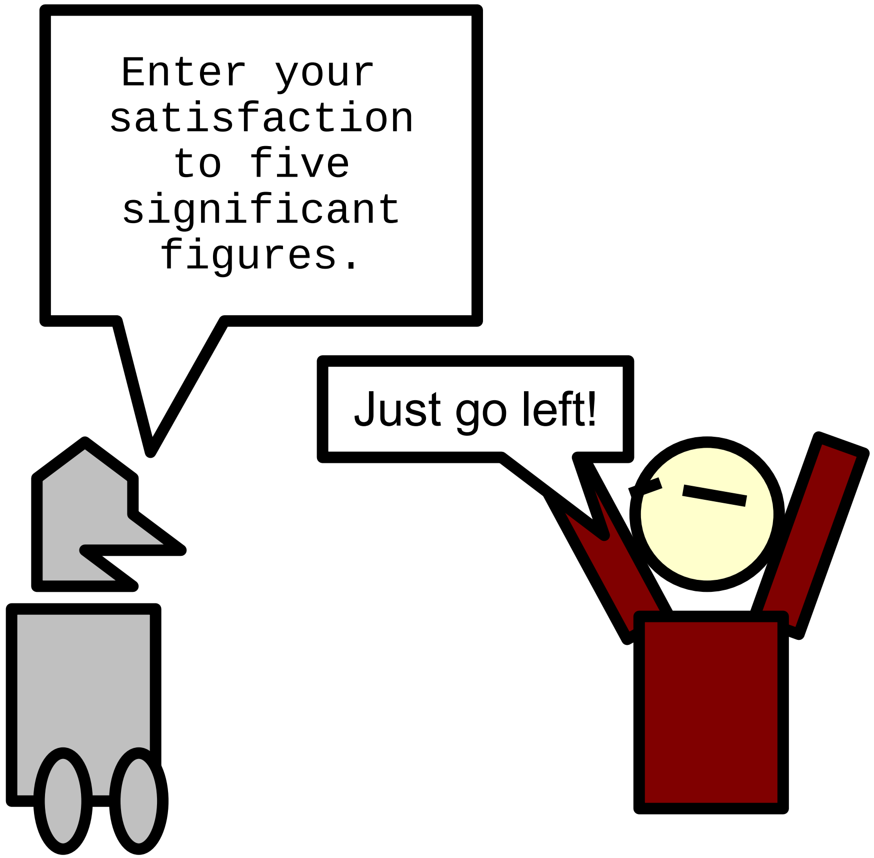
Common issues with RL

- Must make mistakes to learn
- Aside from model/policy convergence, no reasoning about partial information.



Common issues in RL

- Must make mistakes to learn.
- Aside from model/policy convergence, no reasoning about partial information.
- Format of required reinforcement may be unnatural.

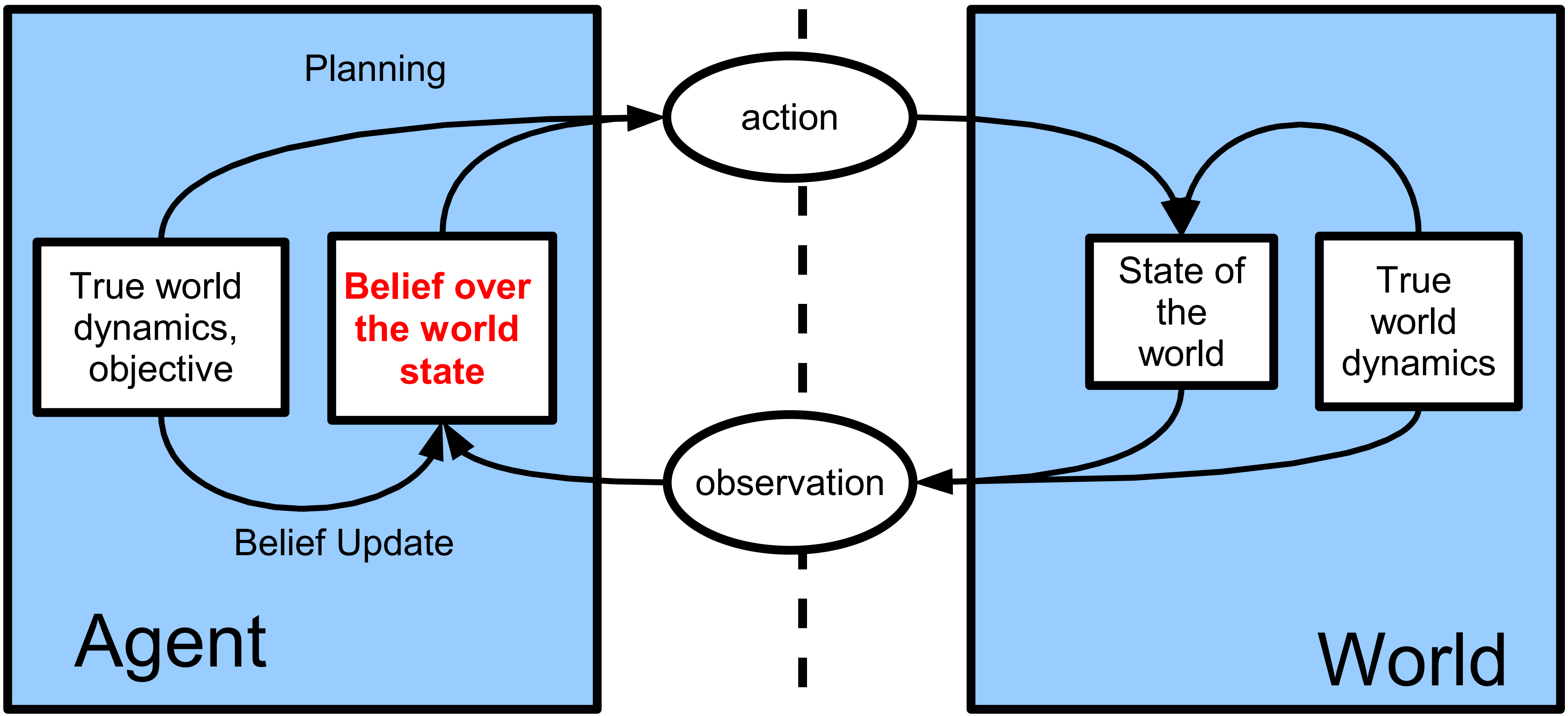


Our Approach

- **POMDP framework:** a Bayesian approach to model uncertainty.
  - The agent is **aware of its model uncertainty**.
- **Meta-queries:** Ask for policy information when confused.
  - The agent can **actively reduce model uncertainty.**
- Resulting approach **combines Bayesian and inverse reinforcement learning**.

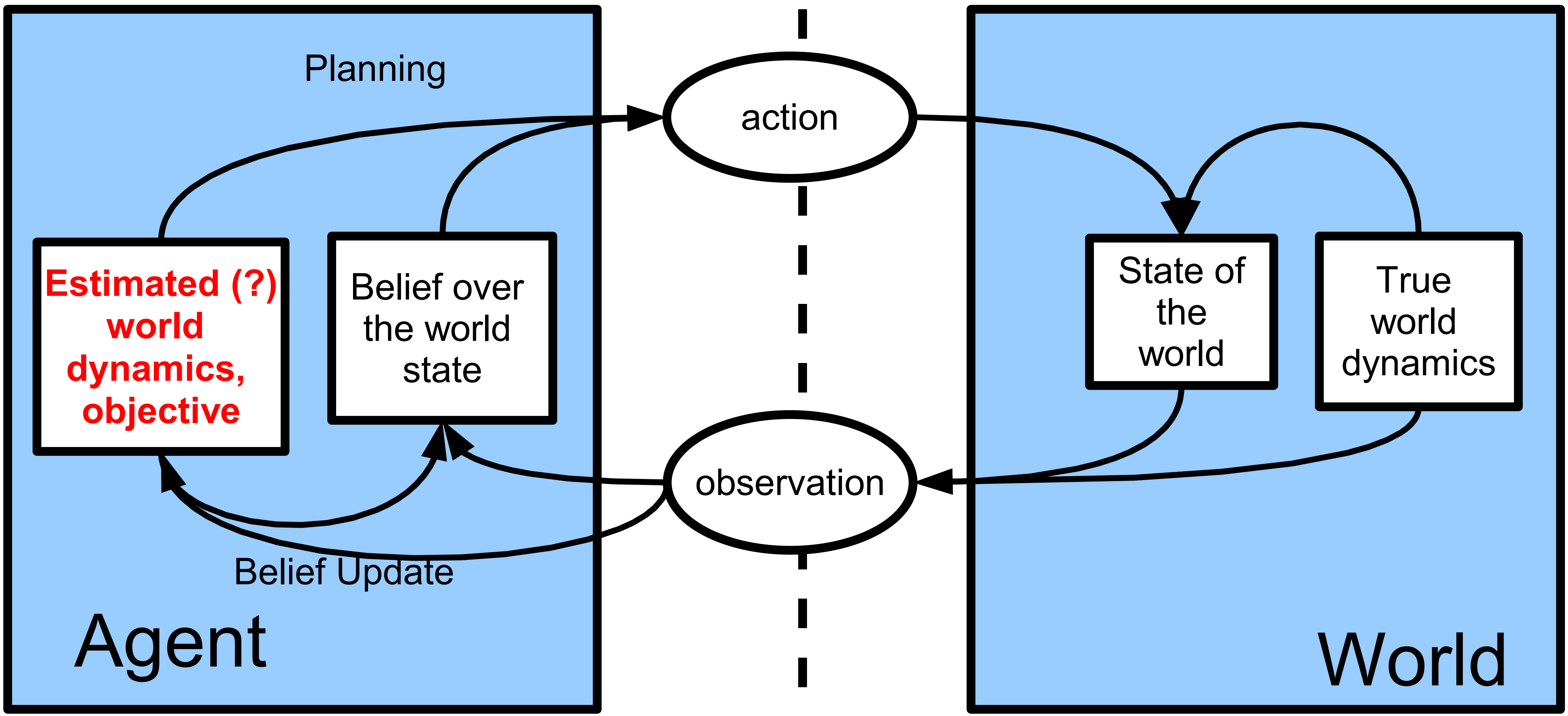
The POMDP Planning Process

When solved, the POMDP optimally trades between gathering information about the state and gathering reward.



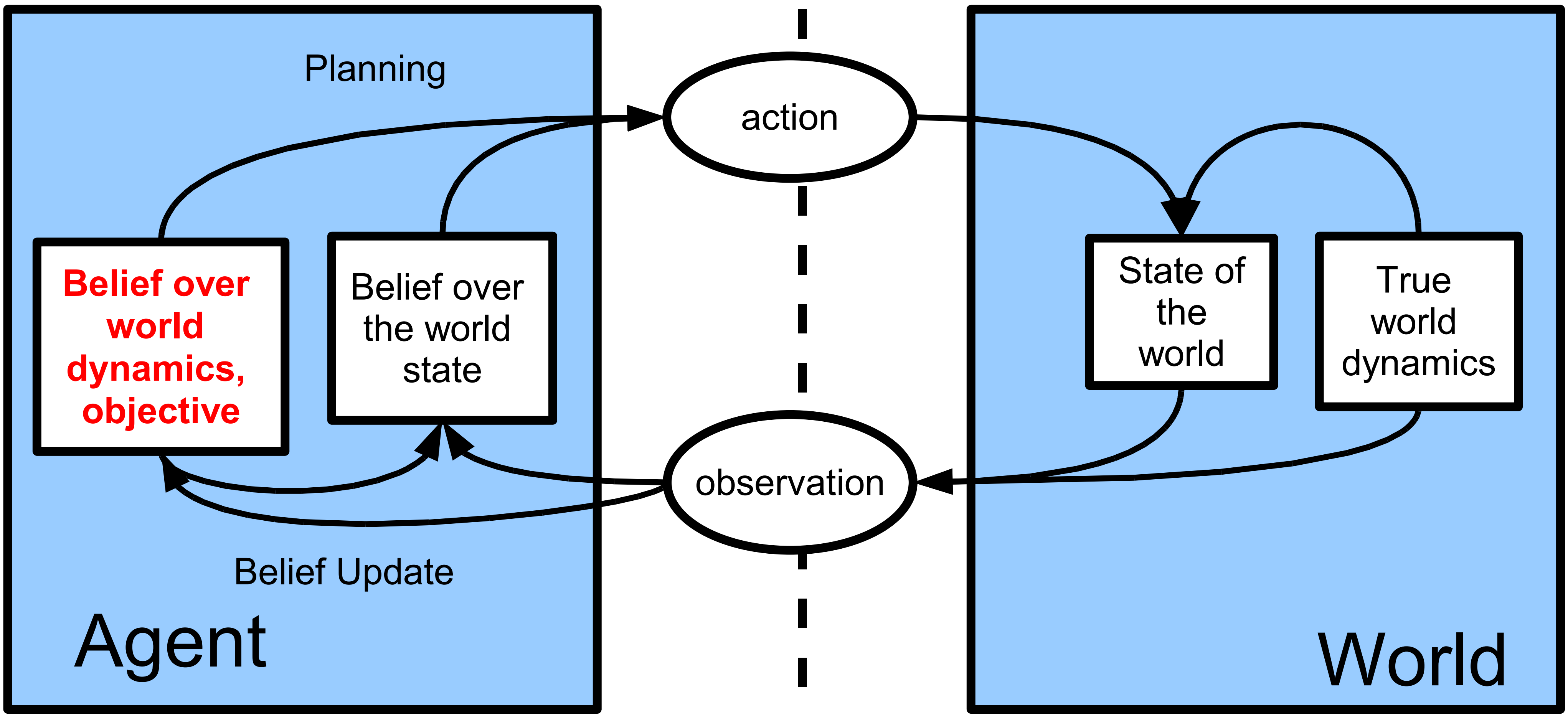
Planning with Uncertain Models

However, models are hard to come by! One option is to keep an estimate of the true model.



Planning with Uncertain Models

But, if we think of the model as hidden state, dealing with model uncertainty is equivalent to dealing with state uncertainty.

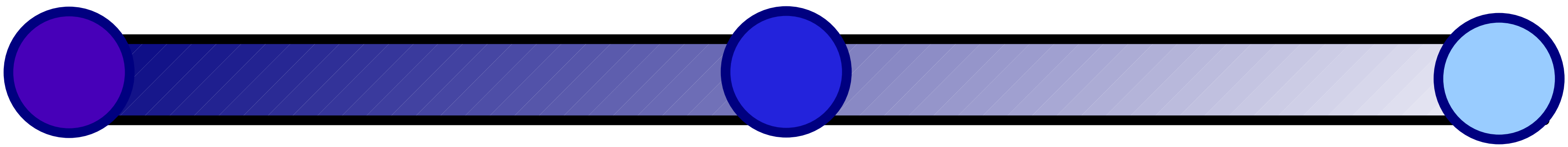


Planning with Uncertain Models

Ignore uncertainty:
fast, not robust

Plan with parameters as hidden state:
optimal but slow

Planning with Uncertain Models

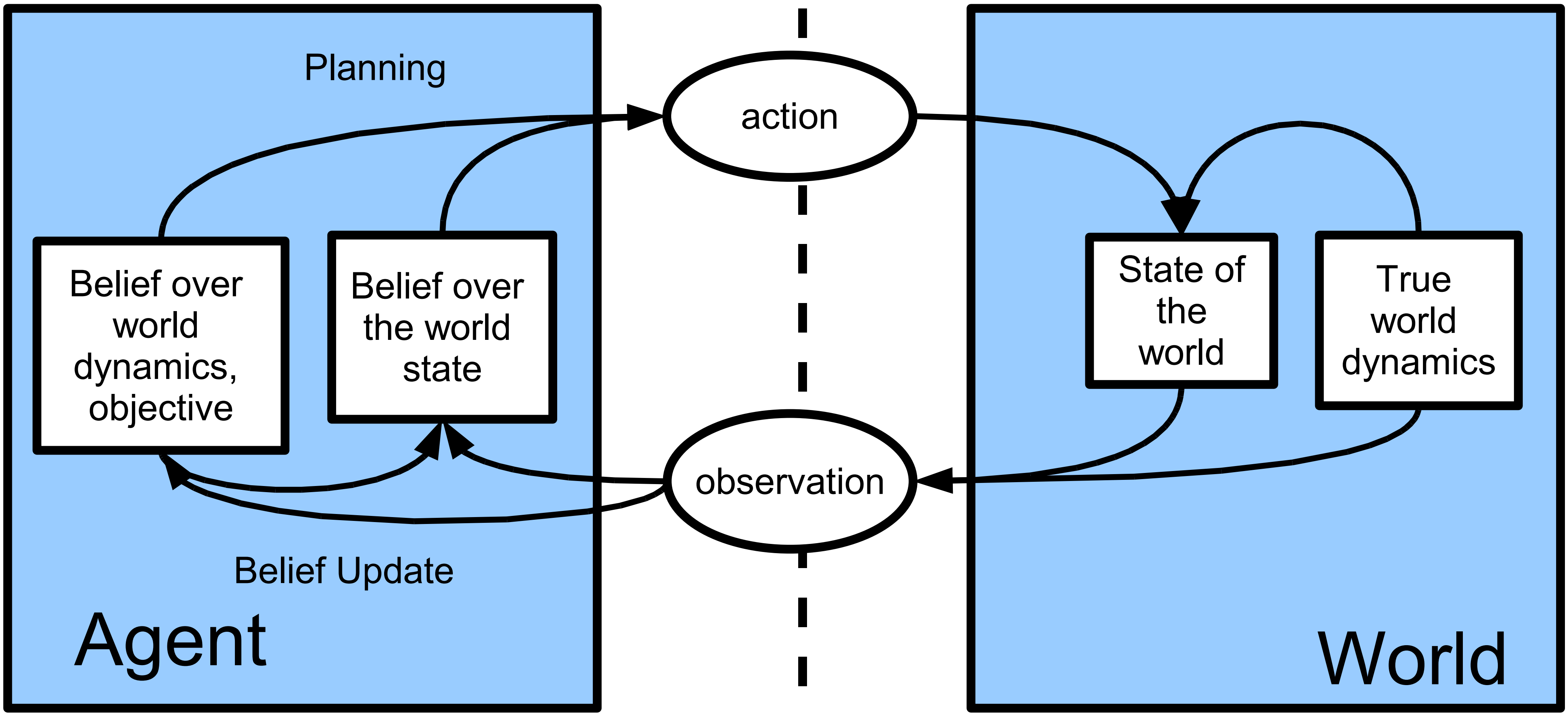


Ignore uncertainty:
fast, not robust

Approximate planning with Bayes risk, policy queries

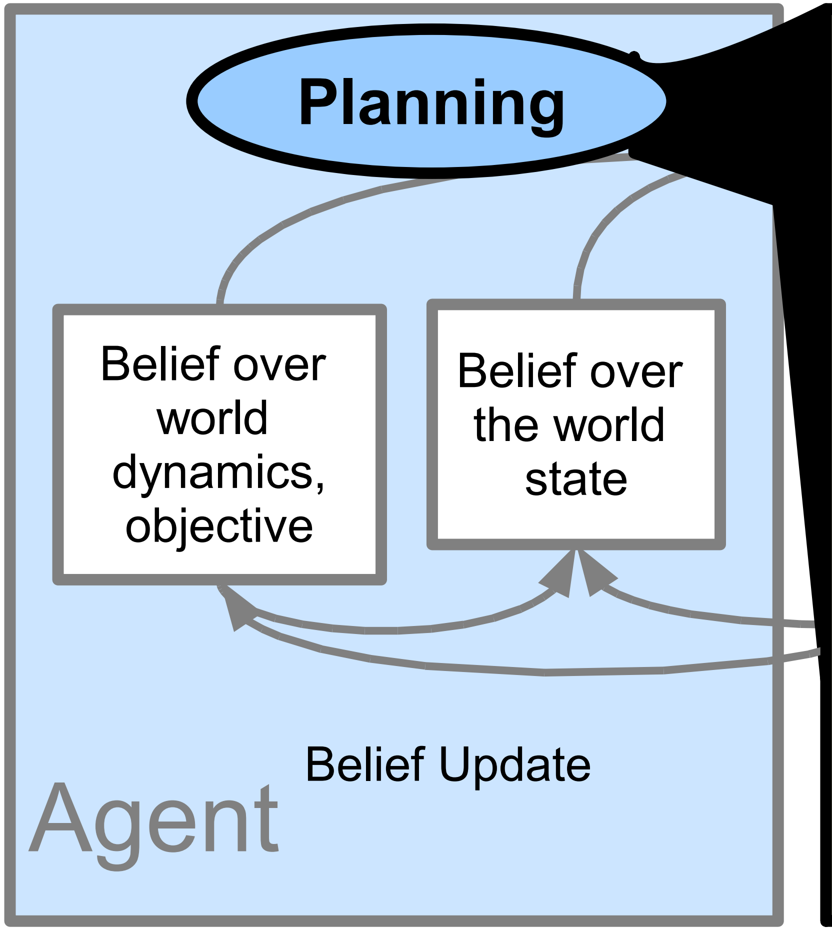
Plan with parameters as hidden state:
optimal but slow

The Model-Uncertainty POMDP



CSAIL

Action Selection



- Use Bayes Risk to find the safest action.

Action Selection with Bayes Risk

- Find the action with the minimal risk:

$$a = argmin_{a \in A} \int_M (Q_m(b_m, a) - Q_m(b_m, a_m')) \, p(m) \, dm$$

- Evaluate the Bayes Risk integral approximately using sampled POMDPs:

$$a = argmin_{a \in A} \sum_i (Q_i(b_i, a) - Q_i(b_i, a_i')) \, w_i$$

(We can bound the approximation error.)

CSAIL

Action Selection with Bayes Risk

- Find the action with the minimal risk:

$$a = \operatorname{argmin}_{a \in A} \int_M \left(Q_m(b_m, a) - Q_m(b_m, a_m') \right) p(m)\, dm$$

- Evaluate the Bayes Risk integral approximately using sampled POMDPs:

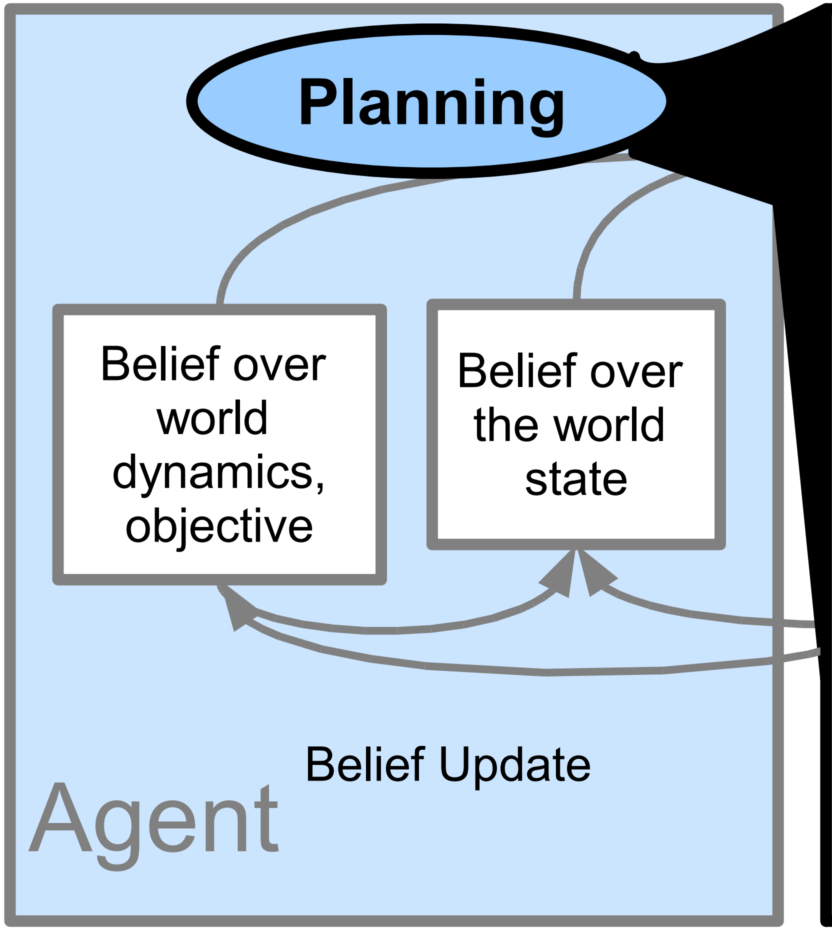
$$a = \operatorname{argmin}_{a \in A} \sum_i \left(Q_i(b_i, a) - Q_i(b_i, a_i') \right) w_i$$

(We can bound the approximation error.)

But what if the safest action is still pretty risky?

CSAIL

Action Selection



- Use Bayes Risk to find the safest action.
- If this action is too risky, ask for help.

Asking for Help: Policy Queries

But what if the risk is too large? Ask for help so

- Agent does not need to take large risks to determine that a particular decision may be poor.
- User only needs to provide reinforcement when the agent is sufficiently confused.

... plus allows us to provide bounds on performance throughout the learning process!

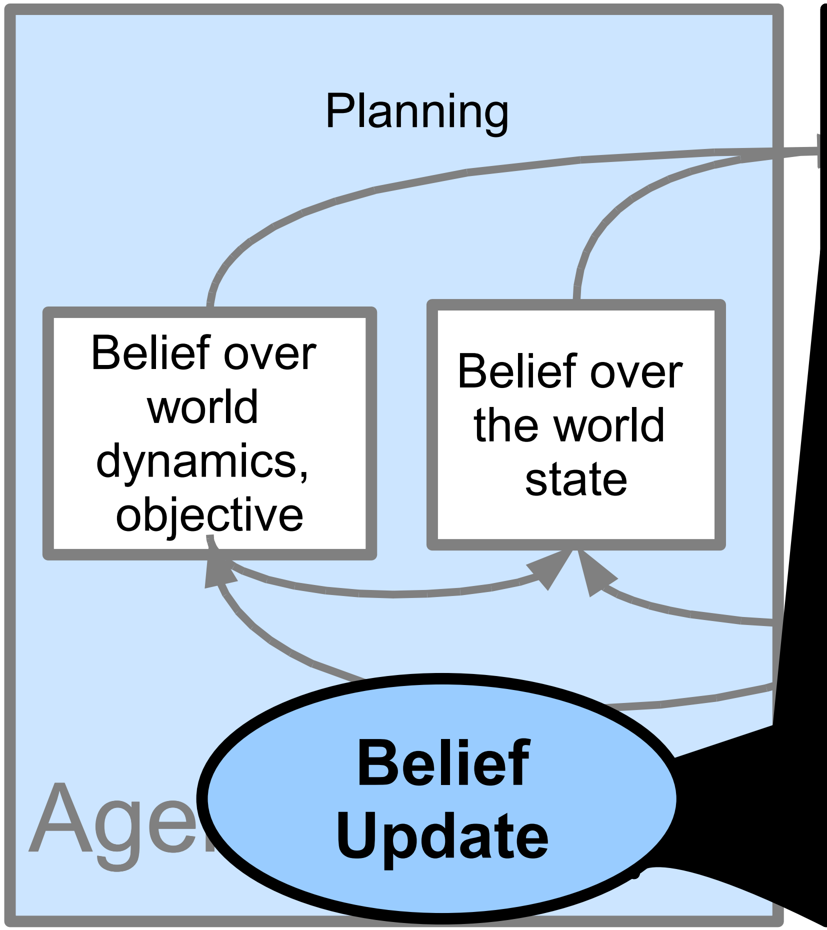
Asking for Help: Implementation

Questions of the form:

- “I think you **might** want to go to the printer. Should I go to the printer?”
- “I'm **certain** you want to go to the printer. Should I go to the printer?”
- “Instead, should I ask for you to confirm your location?”

Ask these questions to determine the correct action; thus a **query results in discovering the optimal action**.

Belief Update



We need to incorporate two sources of information:

- h: Most recent history of actions and observations
- Q: Set of (query, response) pairs that we have asked

Belief Update

- Need to maintain a posterior over models:

$$p(m|h,Q) \propto p(Q|m)\,p(h|m)\,p(m)$$

- History information can be updated in closed form (via an online EM-like update), but query information cannot.

Dirichlet

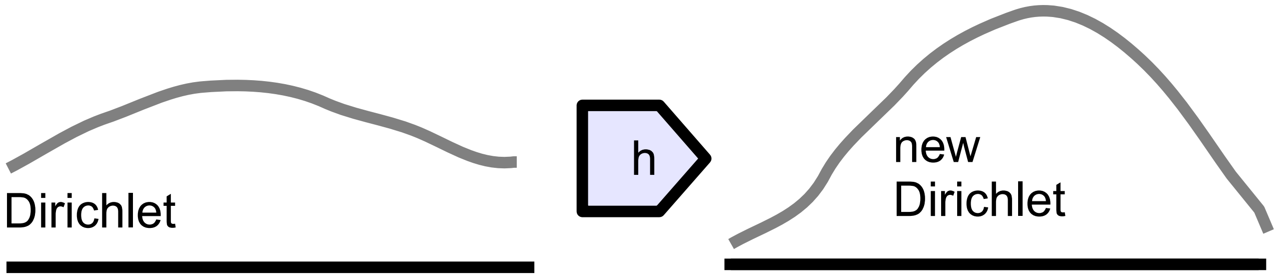
$$p(m)$$

Belief Update

- Need to maintain a posterior over models:

$$p(m|h, Q) \propto p(Q|m)\, p(h|m)\, p(m)$$

- History information can be updated in closed form (via an online EM-like update), but query information cannot.



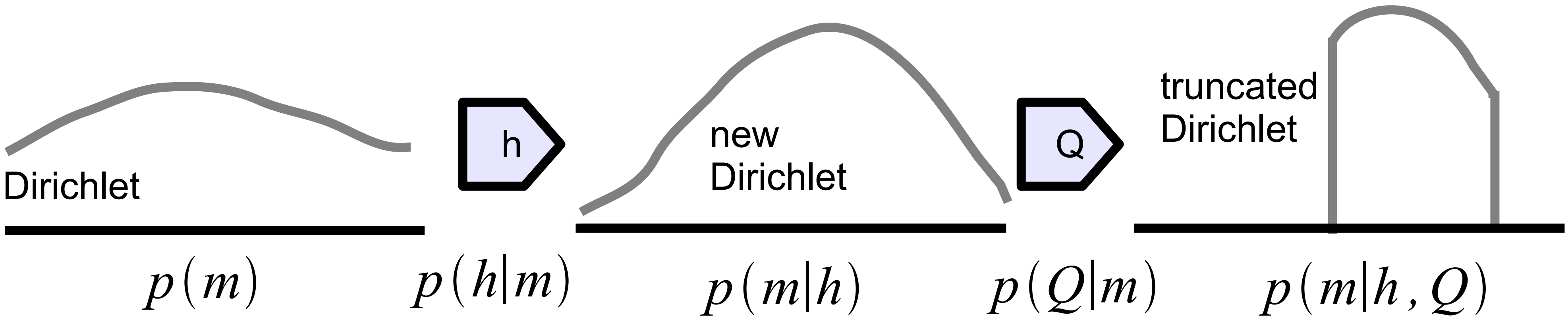
$p(m)$ $\quad p(h|m)$ $\quad p(m|h)$

Belief Update

- Need to maintain a posterior over models:

$$p(m|h, Q) \propto p(Q|m)\, p(h|m)\, p(m)$$

- History information can be updated in closed form (via an online EM-like update), but query information cannot.



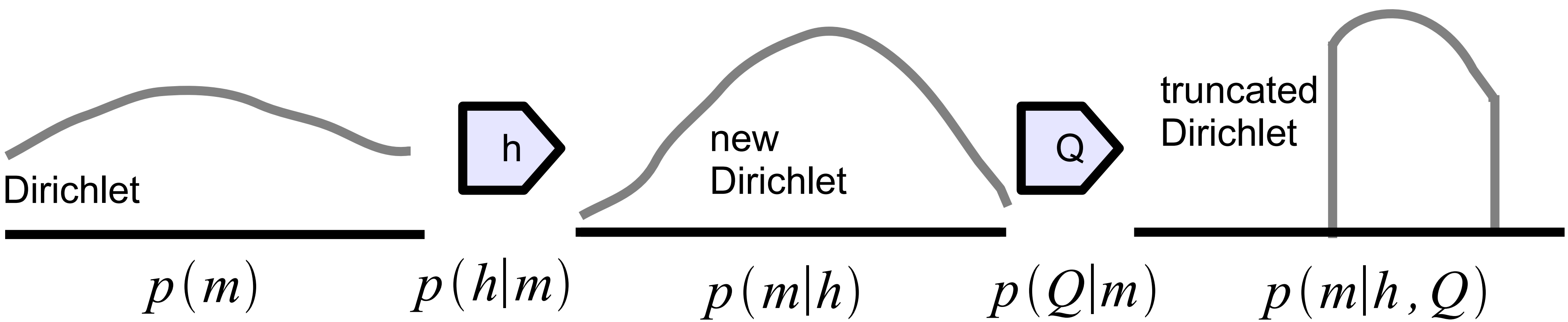
- Therefore we use a set of samples to represent the posterior and apply particle filtering.

Belief Update

- Need to maintain a posterior over models:∝

$$p(m|h,Q) \propto p(Q|m)\,p(h|m)\,p(m)$$

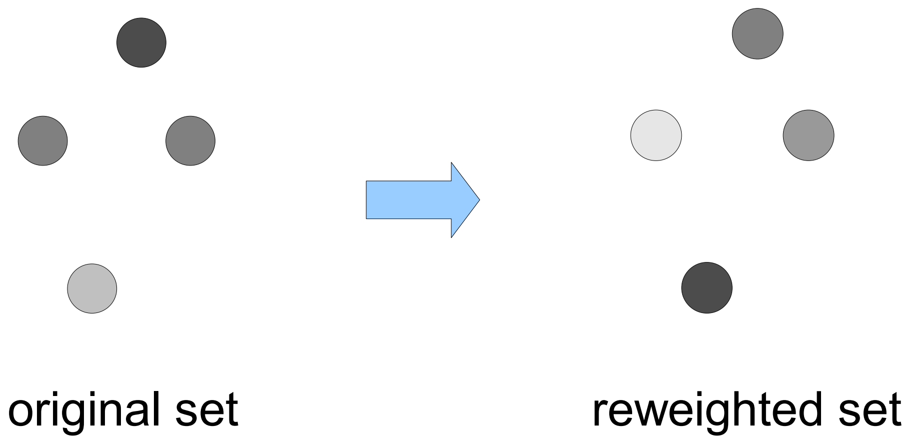
- History information can be updated in closed form (via an online EM-like update), but query information cannot.



- Therefore we use a set of samples to represent the posterior and apply particle filtering.

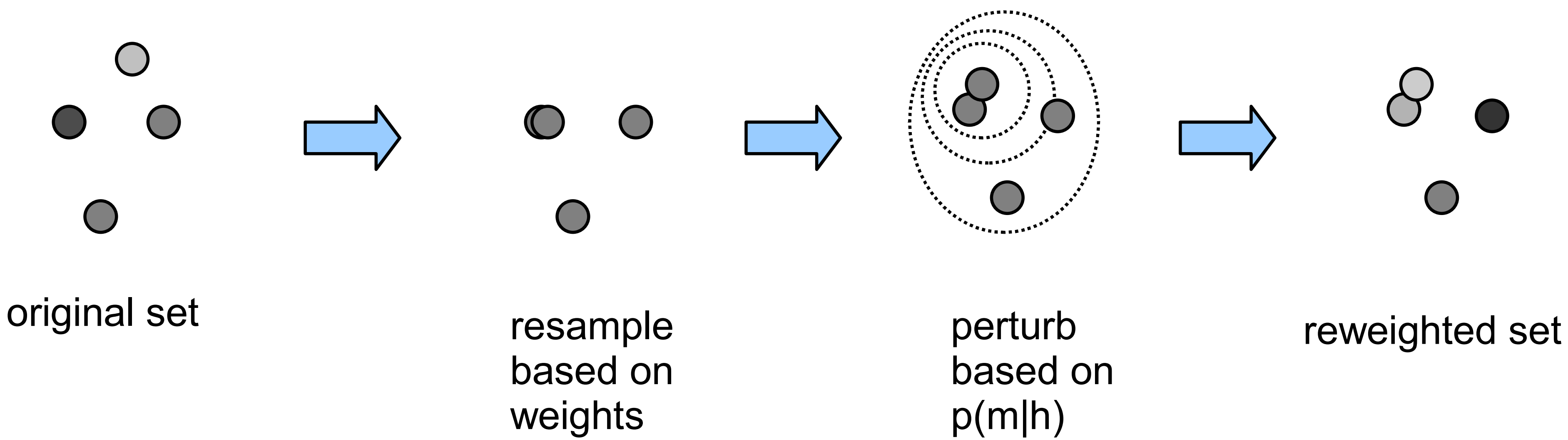
Belief Update: During a Trial

- We want to update the posterior quickly, without solving additional POMDPs, so we don't sample new models, only reweight models based on query information.
- In theory, $p(Q|m)$ should be 0-1, but we use a binomial function to account for approximations in the solver.



Belief Update: Between Trials

- Resample POMDPs.
- Use a transition kernel that incorporates history information by either replacing or perturbing current samples with samples from p(m|h).



Performance Guarantees

- We can lower bound the performance of our approach *in expectation* with respect to the optimal policy:

$$V' > \eta\left(V - \frac{\xi}{(1-\gamma)}\right) + (1-\eta)\left(\frac{R_{min}}{(1-\gamma)}\right), \eta = \frac{(1-\gamma)(1-\delta)}{1-\gamma(1-\delta)}$$

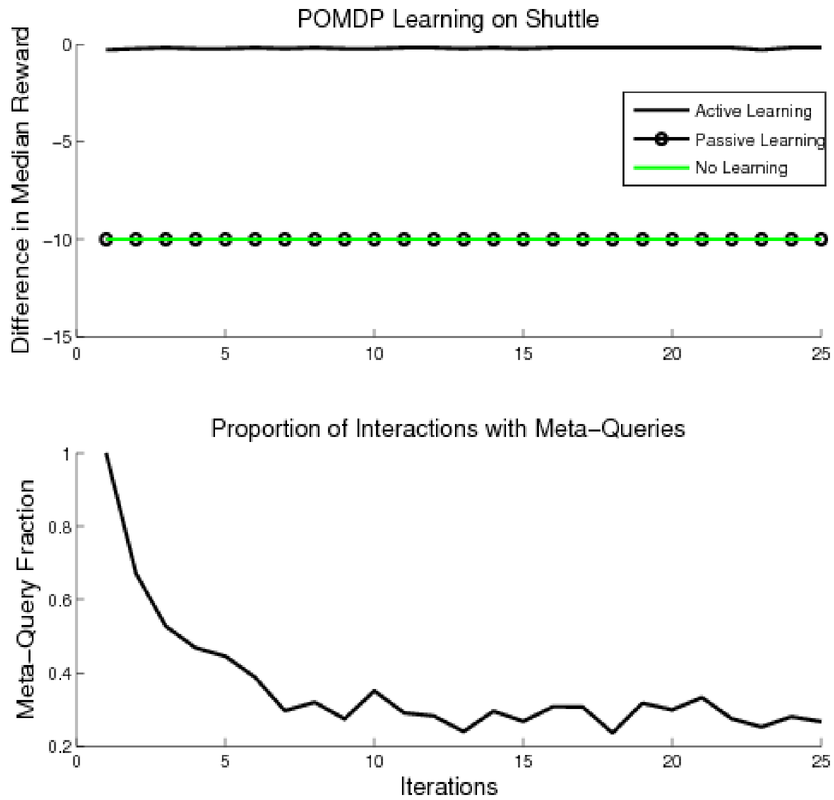
- We will eventually converge to a transition, observation, and reward model.

CSAIL

Results

Results: Standard POMDP Problems

The active learner maintains good performance from the start.



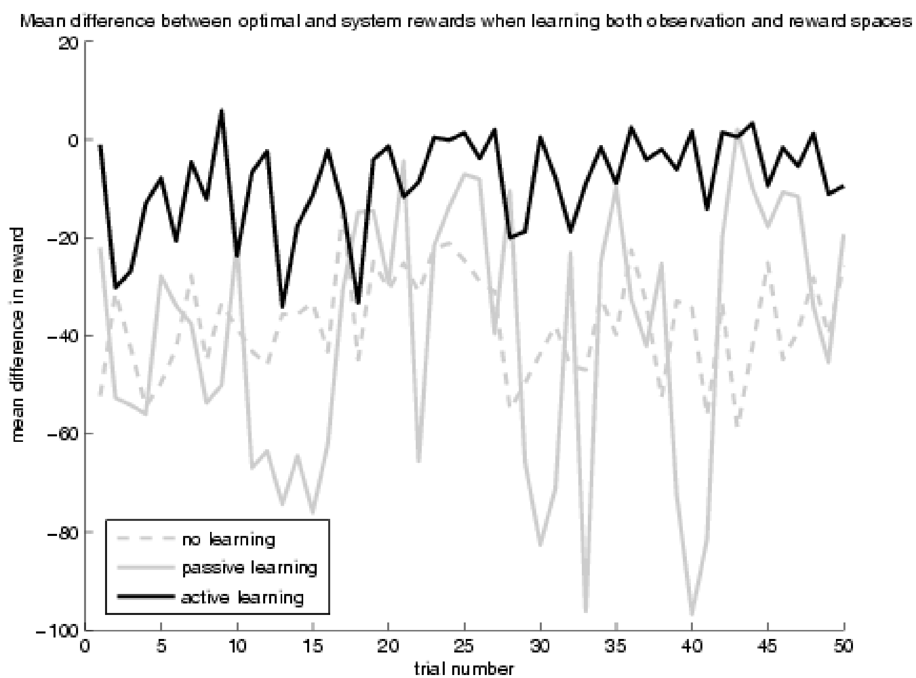
Mean difference between obtained reward and optimal reward over 50 trials (smaller is better)

Problem	States	Control	Passive	Active
Tiger	2	46.5	50.7	33.3
Shuttle	8	10.0	10.0	2.0
Gridworld-5	26	33.1	102	21.4
Hallway	57	1.0	1.0	0.08

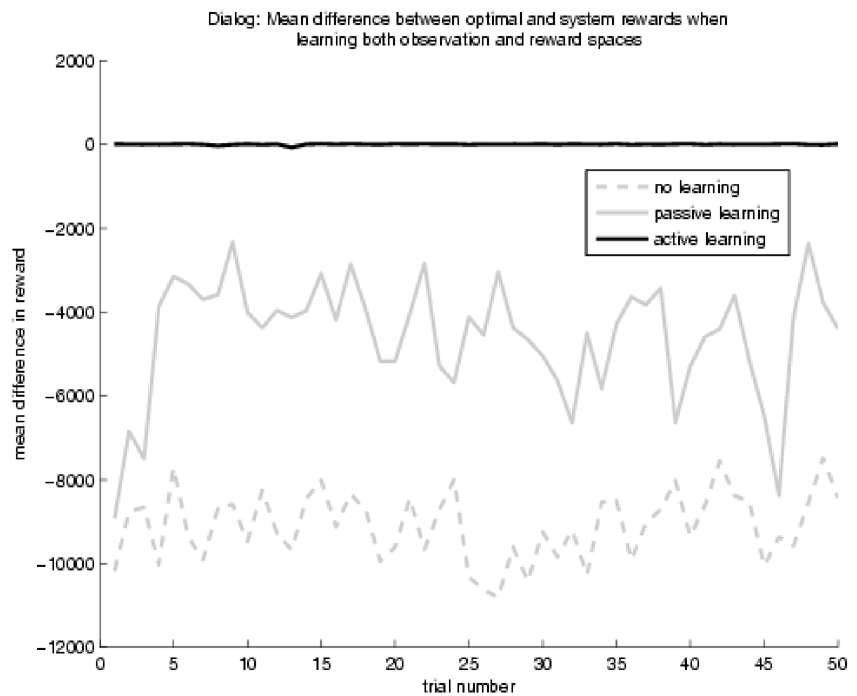
CSAIL

Results: Simulated Dialog Domain

The active learner performs well without informative priors.



Reasonable Prior



Non-informative Prior

Results: Short User Dialog

Early Conversation:
User: Give me the forecast.
Robot: I'm confused. What action should I take now?
<User indicates that the robot should provide the weather forecast>
Robot: Showers

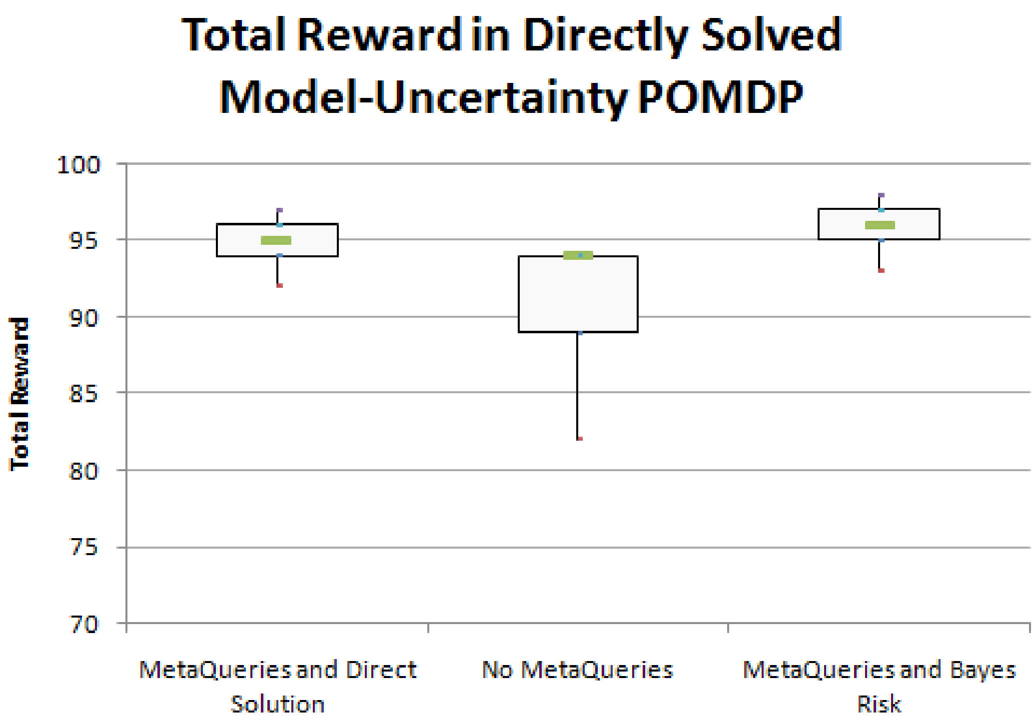
Later Conversation:
User: What's the forecast for today?
Robot: Do you want the weather?
User: Yup.
Robot: Showers

Conclusions and Future Work

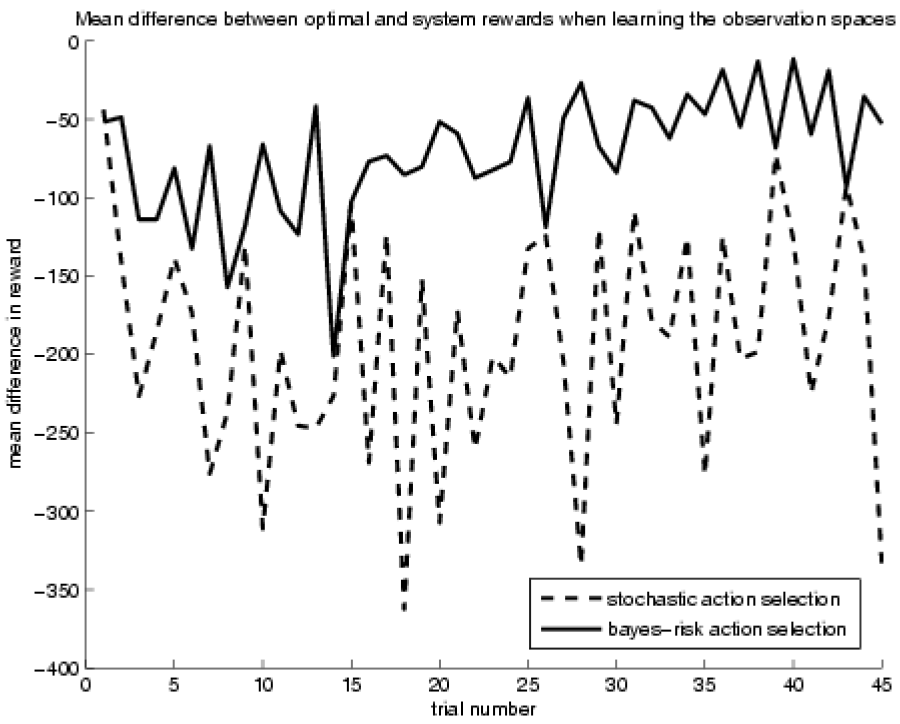
- Although POMDP models require many parameters, we can learn those parameters online.
  - **Bayes risk action** selection lets the agent act robustly in the face of model uncertainty
  - The learning process can be further improved by incorporating **policy queries**.
- Extensions
  - Improve POMDP sampling techniques (or find a closed-form approximation to the POMDP posterior).
  - Approaches to finding policy information from humans (improved meta-queries).

Thank-you!

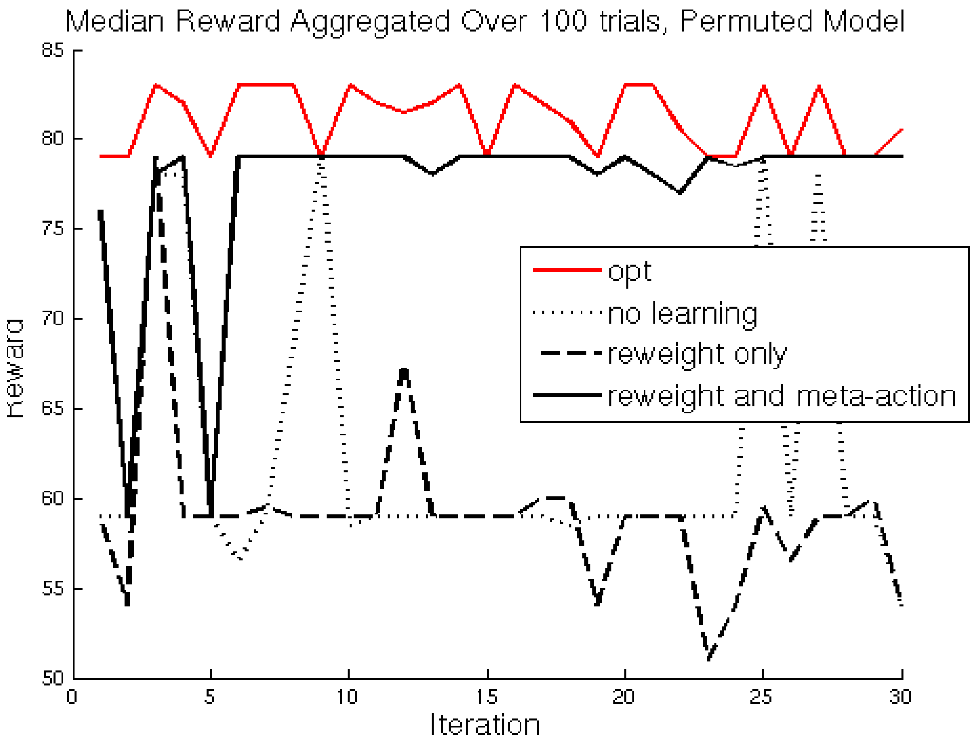
Results: Direct Solution



BR Action Selection (Continuous Models)



Observation Learning



Reward Learning

CSAIL

User Test Results: Example Dialog 2

Early Conversation:

User: I want some tea from the kitchen.
Robot: How can I help you?
User: The tea is in the kitchen near the coffee machine. *<The system hears ``copy'' instead of ``coffee'' and is confused about "tea" and "kitchen.'' >*
Robot: How can I help you?
User: Go to the kitchen.
Robot: I'm confused. What action should I take now?
<The user indicates ``Confirm the coffee machine area'' is the correct action. >
Robot: Do you want to go to the coffee machine area?
User: Yup.
...
<The system asks a few more (non-meta) questions>
Robot: Do you want to go to the coffee machine area?
User: Yup.
Robot: Going to the coffee machine area.

Later Conversation:

User: Take me to the kitchen. *<The system hears "Nick" and ``kitchen.''>*
Robot: How can I help you?
User: I want some coffee. *<The system hears ``copy'' instead of ``coffee.''>*
Robot: Do you want to go to the coffee machine area?
User: Yup.
Robot: Going to the coffee machine area.

Related Work in Bayesian Model Learning

- Dearden et. al.: Bayesian MDP model learning
- Beetle (Poupart et. al.): frame unknown MDP as a continuous state POMDP
- Medusa (Jaulmes et. al.): sample from a distribution over POMDPs; use the sample for action selection

Solving a known POMDP Model

Value of a belief

Value of belief, action pair

$$V_n(b) = max_a Q_n(b, a)$$

$$Q_n(b, a) = R(b, a) + \gamma \sum_{b' \in B} T(b' | b, a) V_{n-1}(b')$$

$$Q_n(b, a) = R(b, a) + \gamma \sum_{o \in O} O(o | b, a) V_{n-1}(b_a^o)$$

Current reward

Future Reward

Error in Approximating Bayes Risk

- If we want to estimate if the Bayes Risk is greater than ζ with confidence δ, two error sources exist:
  - Error due to approximating risk from samples:

$$n_m = \frac{(R_{max} - min(\zeta, R_{min}))^2}{2(1-\gamma)^2 \epsilon_m^2} \log \frac{1}{\delta}$$

  - Error due to approximate POMDP solutions:

$$\epsilon_{pb} = 2\delta_b \frac{(R_{max} - R_{min})}{(1-\gamma)^2}$$

- Noting that $\zeta = \varepsilon_m + \varepsilon_{pb}$, set ε_m and ε_{pb} to trade between the number of belief samples and model samples.

Details for Particle Filter

- In general, given a kernel K(m,m'), particles are sampled and assigned weights according to:

$$m_t \sim K(m_{t-1}, m_t)$$

$$w_t = w_{t-1} \frac{p_{M,t}(m_t)}{p_{M,t-1}(m_{t-1}) K(m_{t-1}, m_t)}$$

- During trials: K(m,m') = δ(m,m')

$$w_t = w_{t-1} p(Q_t | m)$$

- Between trials: Sample m'' from p(m|h), replace m with m'' with probability p; else K(m,m') = am + (1-a)m''.

$$w_t = \frac{p(Q|m') p_{M|h}(m')}{p(Q|m) p_M(m) K(m, m')}$$

Termination Procedure

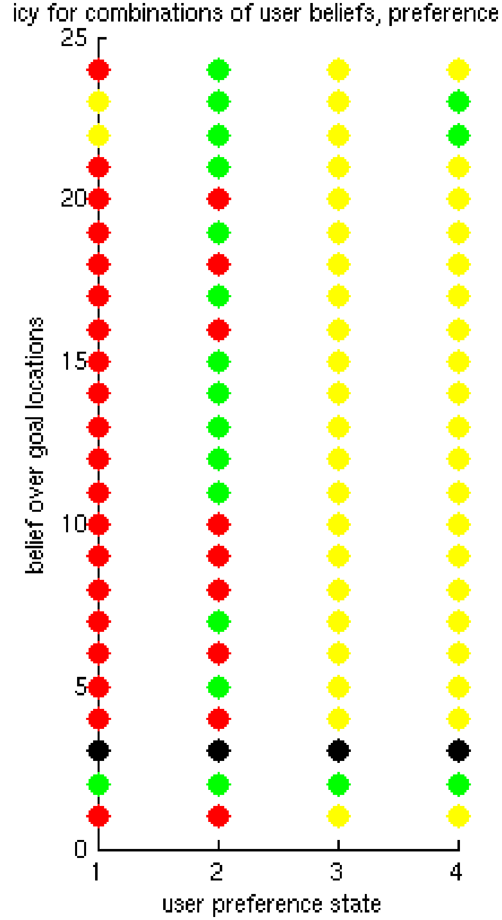
- To estimate if the probability of asking a meta-query after n more interactions is greater than ζ with confidence δ, we can:
  - Compute “worst posterier” by assigning interaction counts to make a flat Dirichlet posterior.
  - Sample POMDPs from the posterior.
  - Sample beliefs from the POMDPs.
  - Reject if $f(\zeta)$-proportion beliefs require meta-queries.
- We can set the number of POMDP, belief samples required, as well as $f(\zeta)$, based on our desired confidence.

Discrete Models: Why few policies?

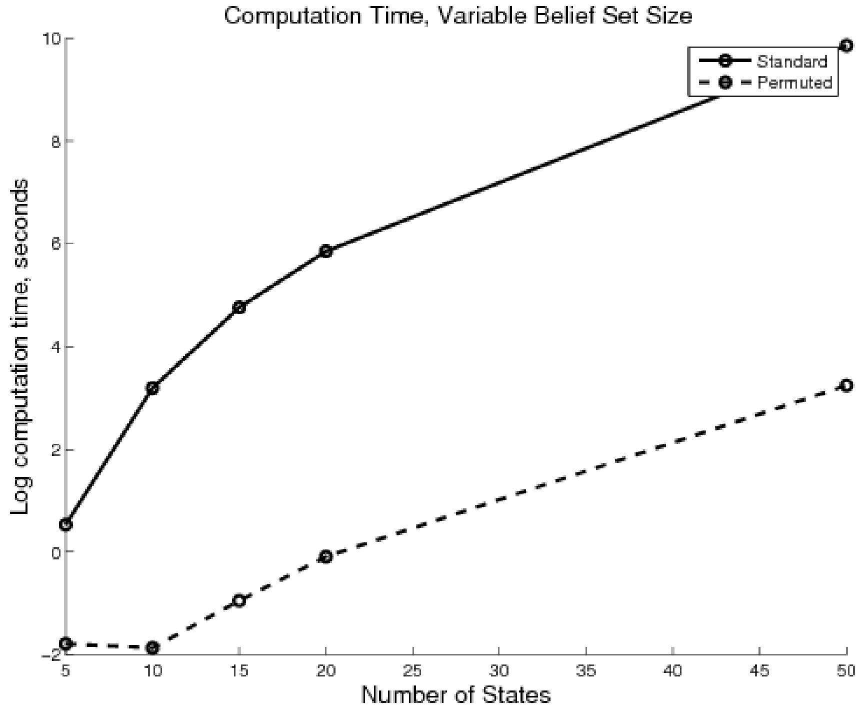
In the special case where:

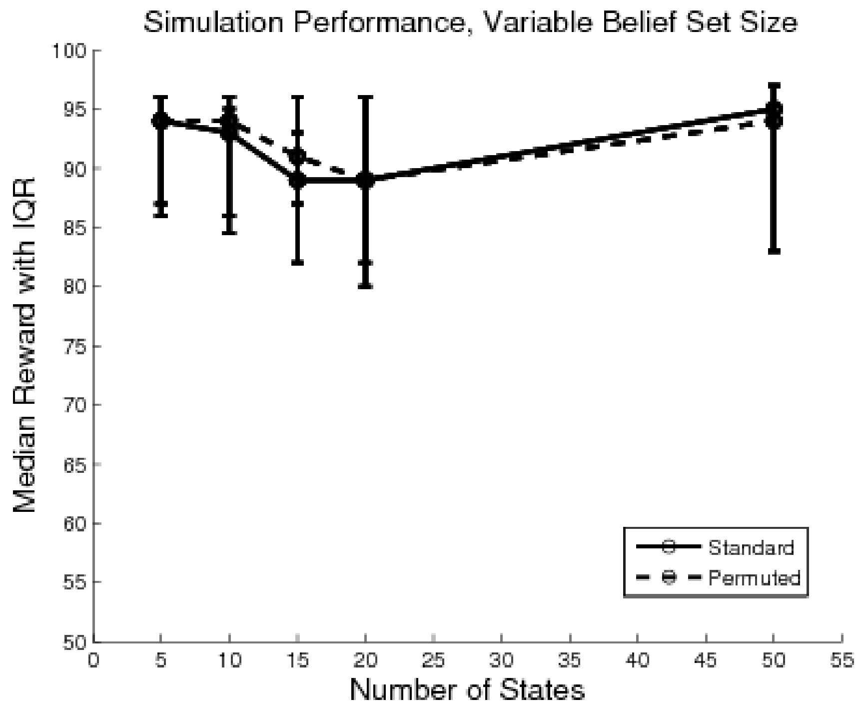
- Only rewards are unknown
- Simple dialog model

The *policies* for a variety of parameter values are similar; the main degree of freedom is how certain we must be before acting, which translates to how many times to confirm a choice.



Sometimes, POMDPs can be solved quickly





Doshi and Roy, AAMAS 2008