

Dialogue management: Tabular approaches to policy optimisation

Milica Gašić

Dialogue Systems and Machine Learning Group,
Heinrich Heine University Düsseldorf

Dialogue optimisation as a reinforcement learning task

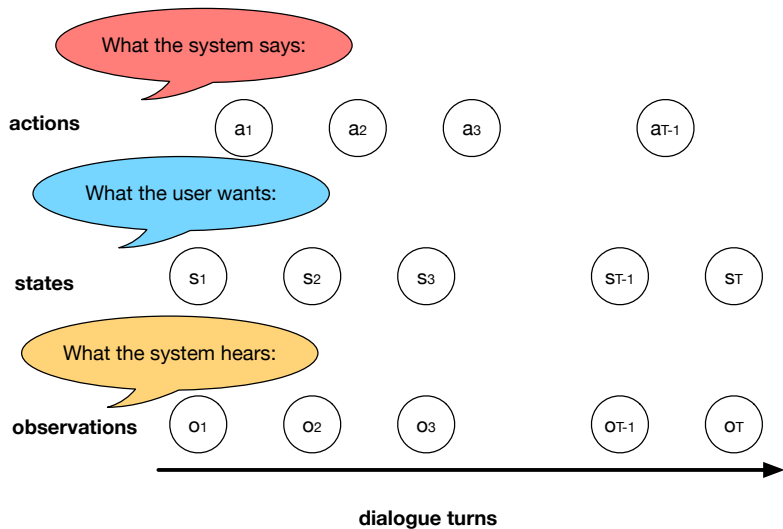
Dialogue management as a continuous space Markov decision process

Summary space

Simulated user

RL algorithms for dialogue management

Elements of dialogue management



Dialogue decision making as an RL problem [Levin et al., 2000]

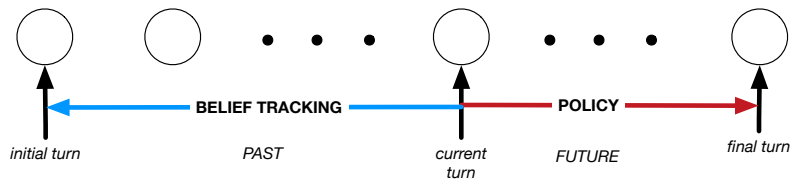
Input the distribution over possible states – belief state, the output of the belief tracker

Control actions that the system takes – what the system says to the user

Feedback signal the estimate of dialogue quality

Aim automatically optimise system actions – dialogue policy

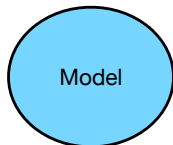
Belief tracking vs policy optimisation



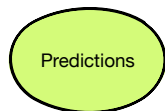
Dialogue as a partially observable Markov decision process



- ▶ Noisy observations
- ▶ Reward – a measure of dialogue quality



- ▶ Partially observable Markov decision process



- ▶ Optimal system actions in noisy environment

Theory: Partially observable Markov decision process

s_t dialogue states

o_t noisy observations

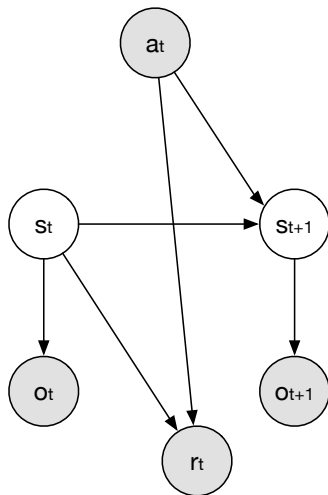
a_t system actions

r_t rewards

$p(s_{t+1}|s_t, a_t)$ transition
probability

$p(o_{t+1}|s_{t+1})$ observation
probability

$b(s_t)$ distribution over
possible states



Optimising POMDP policy

- ▶ Finding optimal policy tractable only for very simple cases [Kaelbling et al., 1998]
- ▶ Alternative view: discrete space POMDPs can be viewed as a continuous space MDP with states as belief states $b_t = b(s_t)$

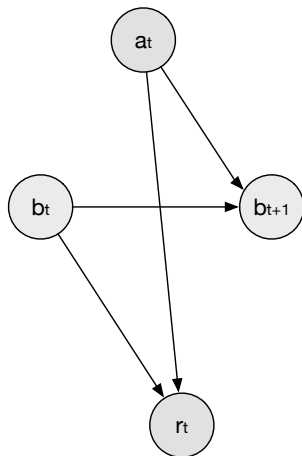
Theory: Markov decision process

b_t belief states from
tracker

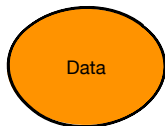
a_t system actions

r_t rewards

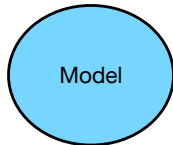
$p(b_{t+1}|b_t, a_t)$ transition
probability



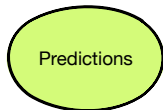
Dialogue management as a continuous space Markov decision process



- ▶ belief states (from belief tracker)
- ▶ Reward – a measure of dialogue quality



- ▶ Markov decision process and reinforcement learning



- ▶ Optimal system actions

Problems

Size of the optimisation problem

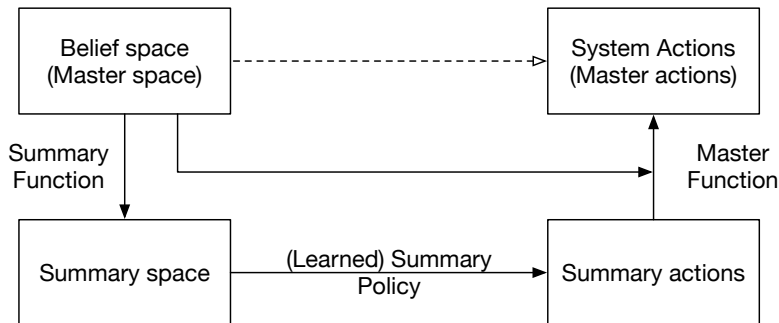
- ▶ Belief state is large and continuous
- ▶ Set of system actions also large

Knowledge of the environment, in this case the user

- ▶ We do not have transition probabilities
- ▶ Where do rewards come from?

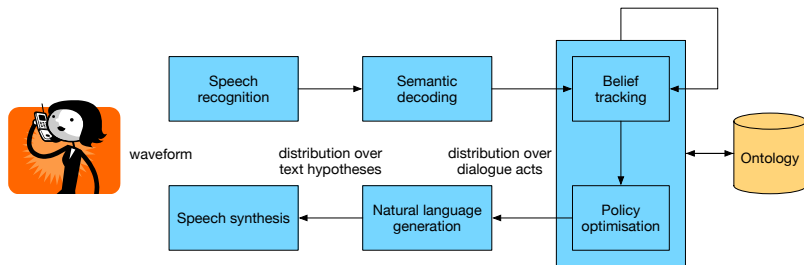
Problem: large belief state and action space

Solution: perform optimisation in a reduced space – summary space built according to the heuristics



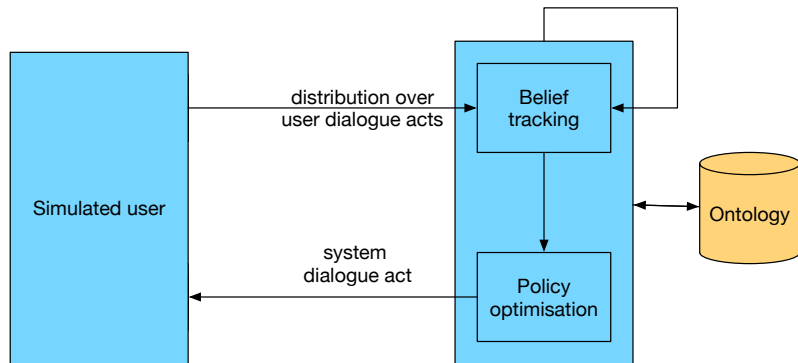
Problem: Where do the transition probability and the reward come from?

Solution: learn from real users.

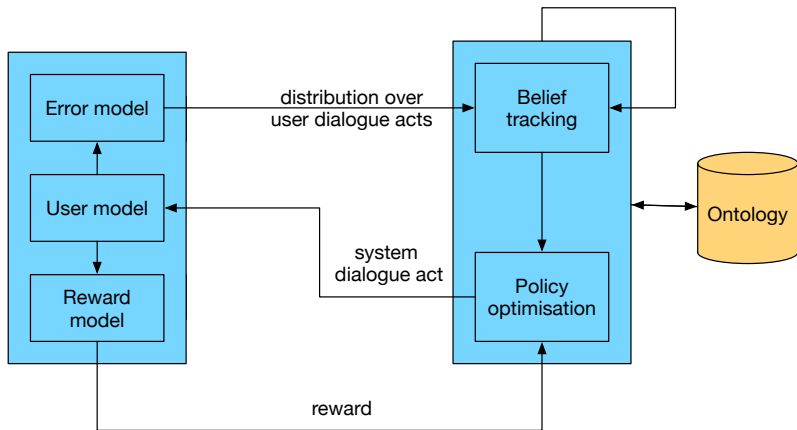


Problem: Where do the transition probability and the reward come from?

Solution: learn from a simulated user.



Elements of the simulated user



Evaluation metrics/optimisation criteria

Candidate	Issues
User satisfaction	How to measure in a real scenario?
Task completion	Task is hidden.
Dialogue length	Hang up on user?
Channel accuracy	Endless confirmations
Repeat usage	Not always makes sense.
Financial benefit	Maybe in industry but not in research.

Williams, Spoken dialogue system: challenges and opportunities for research, ASRU (invited talk), 2009

Theory: reinforcement learning [Sutton and Barto, 2018]

The agent in reinforcement learning

- ▶ Learns from **interaction** with the environment
- ▶ Needs to perform **planning**
- ▶ Has a **goal** or a number of sub-goals
- ▶ Must deal with **uncertain** environments
- ▶ Learns from **experience**

Main elements in reinforcement learning

policy defines behaviour of the agent

reward defines the goal of the agent

value function is a prediction of the future reward for a given state, defines how "good" it is to be in a particular state

model explains how the environment behaves:

- ▶ What are the transition probabilities for different (belief) states?
- ▶ What is the reward for any given state and action?

The agent-environment interface

- ▶ (belief) state $\mathbf{b} \in \mathcal{B}$
- ▶ action $a \in \mathcal{A}$
- ▶ reward $r \in \mathbb{R}$
- ▶ policy $\pi_t(a | \mathbf{b}) = p(a_t = a | b_t = \mathbf{b})$

Return

Episodic tasks: $R_t = r_{t+1} + r_{t+2} + \dots + r_T$, where T is the final time step

Continuing tasks: $R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \dots$ where γ is the discount factor, $0 \leq \gamma < 1$

- ▶ when $\gamma = 0$ the agent is *myopic*
- ▶ when $\gamma \rightarrow 1$ the agent is *farsighted*

Unified notation: $R_t = \sum_{k=0}^{T-t-1} \gamma^k r_{t+k+1}$

Value function

Value function How good is it for the system to be in a particular belief state when following policy π ?

$$V_{\pi}(\mathbf{b}) = E_{\pi} [R_t | b_t = \mathbf{b}]$$

Q-function How good it is to take action a in belief state \mathbf{b} and then follow a policy π ?

$$Q_{\pi}(\mathbf{b}, a) = E_{\pi} [R_t | b_t = \mathbf{b}, a_t = a]$$

Optimal value functions

Policy π is better or equal to policy π' iff $V_\pi(\mathbf{b}) \geq V_{\pi'}(\mathbf{b})$ for every $\mathbf{b} \in \mathcal{B}$.

$$V_*(\mathbf{b}) = \max_{\pi} V_\pi(\mathbf{b})$$

$$Q_*(\mathbf{b}, a) = \max_{\pi} Q_\pi(\mathbf{b}, a)$$

$$= E[r_{t+1} + \gamma V_*(b_{t+1}) \mid b_t = \mathbf{b}, a_t = a]$$

Bellman optimality equation

Bellman optimality equation for Value function:

$$V_*(\mathbf{b}) = \max_a \sum_{\mathbf{b}', r} p(\mathbf{b}', r | \mathbf{b}, a) [r + \gamma V_*(\mathbf{b}')]]$$

Bellman optimality equation for Q function:

$$Q_*(\mathbf{b}, a) = \sum_{\mathbf{b}', r} p(\mathbf{b}', r | \mathbf{b}, a) [r + \gamma \max_{a'} Q_*(\mathbf{b}', a')]]$$

Once we have the optimal Q function we can retrieve the optimal policy:

$$\pi_*(\mathbf{b}) = \arg \max_a Q_*(\mathbf{b}, a)$$

1. Value-based vs policy-based learning

Value-based learning Optimising the Value function or the Q function is equivalent to optimising the policy. Value-based reinforcement learning finds the optimal Value function or the Q-function and from there derive the policy.

Policy-based learning We could directly optimise the policy without needing to optimise first the value function.

Actor-critic methods We can optimise the policy (actor) and the value (critic) jointly. Learning one can aid learning another.

2. Model-free vs model-based learning

Model-free RL The agent has no knowledge of the underlying dynamics of the environment and learns solely from trial and error in interaction with the environment.

Model-based RL The knowledge of the model (transition probabilities and the reward function) is used for planning and no interaction is necessary.

Dyna-Q framework learn from the interaction but use that data to also learn the model.

Offline RL Learn only from (a fixed set of) interactions from a dataset.

3. On-policy vs off-policy methods

In reinforcement learning we distinguish between

On-policy methods which attempt to evaluate or improve the policy that is used to make decisions when interacting with environment

Off-policy methods which evaluate or improve a policy different from that used to interact with the environment (*behavioural policy*).

4. Exploration and exploitation dilemma

When interacting with the environment, the agent must simultaneously:

- ▶ **exploit** current knowledge
- ▶ **explore** new actions

The agent must try a variety of actions and progressively favour those that appear to be best. Examples include:

$$\epsilon\text{-greedy } \pi(\mathbf{b}) = \begin{cases} \arg \max_a Q(\mathbf{b}, a) & \text{with } 1 - \epsilon \text{ probability} \\ \text{random action} & \text{with } \epsilon \text{ probability} \end{cases},$$

where ϵ is the exploration rate

Boltzmann policy $\pi(a|\mathbf{b}) = \frac{\exp(\alpha Q(\mathbf{b}, a))}{\sum_a \exp(\alpha Q(\mathbf{b}, a))}$, where $\alpha > 0$ is the learning rate

Thompson sampling If π or Q function are stochastic we can sample from these to directly deal with exploration/exploitation dilemma.

5. Variance vs bias

- ▶ The prime objective in RL is to maximise the return R in expectation
- ▶ We can make samples and simply observe at the end of each interaction what is the (empirical) return, averaging these over time would give us the estimate. These estimates have high variance (but no bias).
- ▶ Alternatively, we can observe from the Bellman equations that the difference between estimates in the current belief state and the next belief state is the immediate reward. Therefore we can base estimate in the current state on the immediate reward and the (previous) estimate in the next (belief) state. These estimates have high bias (but low variance).

Tabular reinforcement learning

For discrete state spaces standard RL approaches can be used to estimate optimal Value function, Q -function or policy π

Dynamic programming model-based learning and update of the estimates are based on the previous estimates

Monte-Carlo methods model-free learning and update of estimates is based on raw experience (empirical return)

Temporal-difference methods model-free learning and update of the estimates are based on the previous estimates

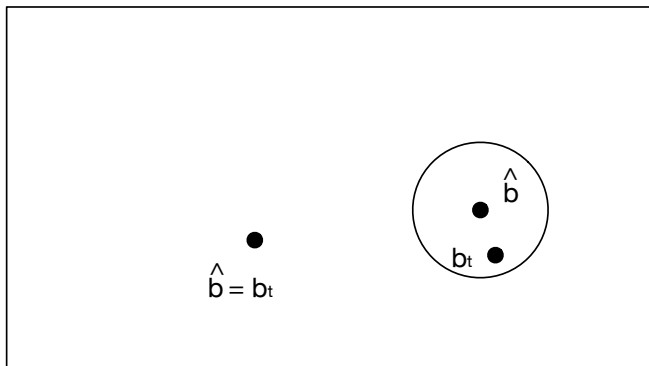
Reinforcement learning for dialogue management

Options

- ▶ **Discretise** the belief state/summary space into a grid and apply tabular RL algorithms to estimate Value function, Q-function or policy π (for example Monte-Carlo Control in the practical)
- ▶ Apply **parametric function approximation** to Value function, Q-function or policy π and find optimal parameters using gradient methods (for example Deep Q network or ACER in the practical)
- ▶ Apply **non-parametric function approximation** to Value function, Q-function or policy π (GPSARSA)

Discretisation of the belief state

- ▶ $\hat{\mathcal{B}}$ is a set of representative points
- ▶ \mathbf{b}_t is a belief state at time step t
- ▶ If \mathbf{b}_t is not near any representative point $\hat{\mathbf{b}}$ we create a new representative point $\hat{\mathbf{b}} = \mathbf{b}_t$
- ▶ Function *Nearest* returns the closest representative point



Belief space

Monte Carlo control algorithm

Algorithm 1 Monte Carlo control

- 1: Initialise π and Q arbitrarily
 - 2: Set $N(\cdot, \cdot)$ to 0
 - 3: **repeat**
 - 4: Generate an episode $[\mathbf{b}_0, a_0, r_1 \dots, a_{T-1}, r_T, \mathbf{b}_T]$ using policy π ϵ -greedily
 - 5: $R \leftarrow 0$
 - 6: **for** $t = T - 1$ down-to 0 **do**
 - 7: $R \leftarrow \gamma R + r_{t+1}$
 - 8: $\hat{\mathbf{b}} = \text{Nearest}(\mathbf{b}_t)$
 - 9: $Q(\hat{\mathbf{b}}, a_t) \leftarrow \frac{Q(\hat{\mathbf{b}}, a_t) * N(\hat{\mathbf{b}}, a_t) + R}{N(\hat{\mathbf{b}}, a_t) + 1}$
 - 10: $N(\hat{\mathbf{b}}, a_t) \leftarrow N(\hat{\mathbf{b}}, a_t) + 1$
 - 11: $\pi(\hat{\mathbf{b}}) = \arg \max_a Q(\hat{\mathbf{b}}, a)$
 - 12: **end for**
 - 13: **until** convergence
-

Monte Carlo control algorithm

1. Value-based or policy-based?
2. Model-based or model-free?
3. On-policy or off-policy?
4. Exploration?
5. High variance or high bias?




Monte Carlo control algorithm

- ▶ Value-based method - we are estimating the Q function
- ▶ Model-free method - we do not know the transition probabilities, the system instead interacts with the (simulated) user
- ▶ On-policy method - we use the current best estimate of the policy to interact with the environment
- ▶ ϵ -greedy exploration
- ▶ Raw experience - we only update the Q function at the end of interaction.

Summary

- ▶ Dialogue policy optimisation can be viewed as a reinforcement learning task
- ▶ POMDP can be viewed as a continuous space MDP
- ▶ Belief state space can be summarised to reduce computational complexity
- ▶ Concepts of consideration in RL include: value-based vs policy-based, model-based vs model-free, exploration vs exploitation, on-policy vs off-policy and variance vs bias.
- ▶ Since we work with continuous spaces in dialogue systems to apply RL the space either needs to be discretised or function approximation needs to be applied.

References I

-  Kaelbling, L. P., Littman, M. L., and Cassandra, A. R. (1998). Planning and acting in partially observable stochastic domains. *Artif. Intell.*, 101(1-2):99–134.
-  Levin, E., Pieraccini, R., and Eckert, W. (2000). A stochastic model of human-machine interaction for learning dialog strategies. *IEEE Transactions on Speech and Audio Processing*, 8(1):11–23.
-  Sutton, R. S. and Barto, A. G. (2018). *Reinforcement Learning: An Introduction*. A Bradford Book, Cambridge, MA, USA.