Your Neighbor: RDL2
Robust Decision-Making & Learning Lab

Hanna Kurniawati

hanna.kurniawati@anu.edu.au

http://users.cecs.anu.edu.au/~hannakur/
What we do

Algorithms for robust decision making:
• Large uncertainty
• Complex system dynamics (including multi-agents & human intention)

CSIT N323: Robust Decision-making & Learning Lab (RDL2)
The dream: Where it all begins...

Make me a cup of coffee, 1\textsuperscript{st} time seeing the coffee maker

Enable robots to manipulate objects to accomplish specific tasks when its understanding about the system (e.g., objects, available tools, and its environment) are limited to none
Slowly, slowly catchy monkey (hopefully)
What’s needed?

Usually:

1. Use sensors to collect lots of data & learn best model from data
2. Use deterministic planning, assuming the model is faithful
3. Execute the plan

Just an approach to solve the problem
What’s the problem?

Excluding h/w design, the problem is:

What should robots do now, so that they can get good long term returns (e.g., accomplish a task), despite various types of uncertainty?
The common approach…

To a large extent is based on a dichotomy of planning vs learning (control vs SID):

Relevant initial information about the system

- Little to none
- A lot to all

Imitation / Reinforcement Learning

Deterministic planning
But, the world is usually not that binary...

Relevant initial information about the system

- Little to none
- A lot to all

• Know **something** (not all) about the system **to some extent** (not exact)
I’ll try to argue...

Relevant initial information about the system

Little to none  [ ]  A lot to all

Learning (at least RL) ↔ Planning
The problem

What should robots do now, so that they can get good long term returns (e.g., accomplish a task), despite various types of uncertainty?
Decide the best strategy (often called policy) based on distributions over states (often called beliefs)
Framing the Problem: POMDP Model

• Main components:
  • State space (S)
  • Action space (A)
  • Observation space (O)
Framing the Problem: POMDP Model

- Main components:
  - **State space (S)** (Not known)
  - Action space (A)
  - Observation space (O)
  - Transition function (T)
  - Observation function (Z)
  - Reward function (R)
“Best” policy

• Maps each belief to an action that satisfies the following objective function

\[ V^*(b) = \max_{a \in A} \left( \sum_{s \in S} R(s, a)b(s) + \gamma \sum_{o \in O} P(o|b, a)V^*(b') \right) \]

Expected immediate reward

Expected total future reward

\( b' \): next belief after the system at belief \( b \) performs action \( a \) and observes \( o \)
\( \gamma \): discount factor, \((0,1)\)
How to get the POMDP model?

• Spaces are easy, how about the functions?
• We could embed uncertainty about the POMDP model in the POMDP itself
In POMDP

**MDP Model**
- State space \((S_{MDP})\)
- Action space \((A_{MDP})\)
- Transition function \((T_{MDP})\)
- Reward function \((R_{MDP})\)

**Construct a POMDP**
- Where the states are MDP states \(X\) parameters of the \(T_{MDP} & R_{MDP}\)
- Essentially, partial observability on which MDP model is the right model
- \(A, T, R\) follows from the particular MDP model
- \(O & Z\) are observations & observation function about which MDP model is correct

Not known
But, how to get the POMDP model?

• POMDP is MDP in the belief space
• So, the same concept applies
  • Off course, much more complicated
In POMDP

**MDP Model**
- State space \((S_{\text{MDP}})\)
- Action space \((A_{\text{MDP}})\)
- Transition function \((T_{\text{MDP}})\)
- Reward function \((R_{\text{MDP}})\)

**Construct a POMDP**
- Where the states are MDP states \(X\) parameters of the \(T_{\text{MDP}}\) & \(R_{\text{MDP}}\)
- Essentially, partial observability on which MDP model is the right model
- \(A, T, R\) follows from the particular MDP model
- \(O \& Z\) are observations & observation function about which MDP model is correct

Reinforcement learning (RL)

Bayesian RL

Not known
So, everything is planning…

Just need to solve that huge POMDP problem
Computationally intractable [Papadimitriou & Tsikilis’87, Madani, et.al.’99].
Not all gloom & doom

• Close to optimal solution is often good enough
  → Sampling

• There’s many useful “structures” even in seemingly unstructured problems
  → Perhaps not environmental structures, but uncertainty structures (e.g., correlation, dependencies / independencies, etc.)
  → Inherent properties of the problems (e.g., continuity of motion in robotics)
  → Significantly reduce sampling domain, converge to good solutions faster
Scaling up POMDP solving capability

• Large state space [Kurniawati, et.al. (RSS’08)]

• Large observation space [Kurniawati, et.al. (RSS’11, Auro’12 invited)]

• Long planning horizon [Kurniawati, et.al. (ISRR’09, IJRR’11 invited)]

• Model may change [Kurniawati & Patrikalakis (WAFR’12), Kurniawati & Yadav (ISRR’13)]

• Large action space [Seiler, et.al. (ICRA’15, best paper award finalist), Wang, et.al. (ICAPS’18)]

• Complex transition functions [Hoerger, et.al. (WAFR’16), Hoerger et.al. (submitted to RSS’19)]

Implementation: http://rdl.cecs.anu.edu.au/software
Some Progress

Temizer, et.al. (Lincoln Lab TR’09) Improve safety of TCAS by 20X

Koval, et.al. (Sid Srinivasan’s group)

Horowitz & Burdick (Joel Burdick’s group)

Nikolaidis, et.al. (Julie Shah’s group)

Wang, et.al. (ICAPS’15) Learn interaction model of bees with reduced data

Bandyopadhyay, et.al. (early work leading to nuTonomy)

Hopefully, satisfied users of our POMDP solvers
But …

• Still not enough to consider uncertain model in general
  • To model initially unknown transition of a simple grid navigation where
    • A robot can move in 8 wind direction
    • Assuming transition is the same everywhere
    • The probability value is discretized into 5 bins
  • we’ll need to multiply the number of states by ~390K
    • Also observations
Machine Learning Solutions

• Computing a good policy is viewed as the problem of finding a **mapping** that fits the **data**
  • Mapping from which space to which space?
    • Model-based
    • Model-free
  • Where does the data come from?
    • Someone / something provides examples
    • Trying on a simulator / the system
  • Use optimization (e.g., policy search) to find a mapping that “best” fits the data
  • More recently, frame as a deep learning problem
Embedding & Solving MDP w/o T & R in Neural Net

\[ V^*(s) = \max_{a \in A} \left( R(s, a) + \gamma \sum_{s' \in S} T(s'|s, a)V^*(s') \right) \]

Convolution, T as the kernel (learned weight)

Sum, R as CNN (learn mapping from images to a map of real number)

max-pool

Iteration: RNN, 1 iteration = 1 layer
Train end-to-end, imitation learning

VIN (Tamar, et.al. NIPS’16)
POMDP?

- Propagate belief (Bayes filter)
  - Jonchowski, et.al.: Histogram (NIPS’16), particle (RSS’18)

- Planning:
  - Straightforward extension of MDP Value Iteration use QMDP planner
    QMDP-Net (Karkus, et.al. NIPS’17)
So, everything is learning…

Just need to get those data somehow
Reducing data requirements

• Turns out, non learning-techniques (including planning) helps …
  • Local structures [WAFR’20]
  • POMDP planning [ICAPS’15, best student paper]
Caveat in VIN, QMDP-Net

\[ V^*(s) = \max_{a \in A} \left( R(s, a) + \gamma \sum_{s' \in S} T(s'|s, a)V^*(s') \right) \]

- Convolution, T as the kernel (learned weight)

• That T is assumed to be independent of states…
  • Makes the #learned weight small
  • Reduce data requirement
TransNet

• T depends on local geometry (and action)
## Results

<table>
<thead>
<tr>
<th>Environment</th>
<th>Success Rate</th>
<th>QMDP-Net</th>
<th>LCI-Net</th>
</tr>
</thead>
<tbody>
<tr>
<td>Freiburg Bldg 79</td>
<td>11.3%</td>
<td>56.7%</td>
<td></td>
</tr>
<tr>
<td>(trained 10x10)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Freiburg Bldg 79</td>
<td>50.8%</td>
<td>66.4%</td>
<td></td>
</tr>
<tr>
<td>(trained 20x20)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Intel Labs</td>
<td>7.3%</td>
<td>54.7%</td>
<td></td>
</tr>
<tr>
<td>(trained 10x10)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Intel Labs</td>
<td>46.8%</td>
<td>66.4%</td>
<td></td>
</tr>
<tr>
<td>(trained 20x20)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Environment</th>
<th>Success Rate</th>
<th>QMDP-Net</th>
<th>LCI-Net</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dynmaze 9x9</td>
<td>80.8%</td>
<td>97.8%</td>
<td></td>
</tr>
<tr>
<td>(trained 9x9)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dynmaze 29x29</td>
<td>9.1%</td>
<td>59.0%</td>
<td></td>
</tr>
<tr>
<td>(trained 9x9)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Reducing data requirements

• Turns out, non learning-techniques (including planning) helps …
  ✔ Local structures [WAFR’20]
• POMDP planning [ICAPS’15, best student paper]
POMDP planning to accentuate data

How do they avoid mid-air collision?

How do bees avoid collision?

• Current view:
  • Animal behavior optimizes certain criteria
  • The question is what criteria is being optimized

Wang, et.al. ICAPS’ 15 (Outstanding Student Paper Award)
A Hypothesis Ranking System

Criteria-1 \[\xrightarrow{\text{POMDP model 1}}\] \[\xrightarrow{\text{POMDP solver}}\] Sim \[\xrightarrow{\text{Sim}}\]

Criteria-2 \[\xrightarrow{\text{POMDP model 2}}\] \[\xrightarrow{\text{POMDP solver}}\] Sim \[\xrightarrow{\text{Sim}}\]

\[\vdots\]

Criteria-n \[\xrightarrow{\text{POMDP model n}}\] \[\xrightarrow{\text{POMDP solver}}\] Sim \[\xrightarrow{\text{Sim}}\]

Rank the criteria based on how similar the simulated trajectory is to the (limited) experimental data.
A Hypothesis Ranking System
(from 100 real data)

Correctly rank phototaxis behavior + horizontal centering at the top of the bees’ behaviour
Reducing data requirements

• Turns out, non learning-techniques (including planning) helps …
  ✓ Local structures [WAFR’20]
  ✓ POMDP planning [ICAPS’15, best student paper]
So, it seems...

Relevant initial information about the system

Learning (at least RL) ↔ Planning
Perhaps...

• The Problem: **Robust Autonomy:**
  What should robots do now, so as to accomplish specific tasks well, despite various types of uncertainty

• Framework: MDP, RL (MDP w. missing component), POMDP, ...

• Solution:
  • Planning, learning, & combination
  • The problem is hard, better take anything that can help solve
What we do

Scaling up algorithms for robust autonomy:
• Large uncertainty
• Complex system dynamics (including multi agents & human intention)

Robust Manipulation Planning

Task: Make me a cup of coffee 1st time seeing the coffee maker

How should robots use tools and manipulate objects to accomplish specific tasks when its understanding about the tools, objects, and its environment are limited to none?

Assuring Autonomous Systems

How to automatically find scenarios that can cause catastrophic failures (before it happens)?

CSIT N323: Robust Decision-making & Learning Lab (RDL2)
Thank you

Q&A