Symbolic Dynamic Programming for Continuous State and Action MDPs

Zahra Zamani
Scott Sanner
Cheng Fang
Continuous State and Action MDPs: e.g., Inventory Control

- **Continuous state and actions**
  - **State:** inventory quantities
  - **Action:** how much of each item to reorder

- **Inventory closed-form optimal policy?**
  - Scarf’s solution (1958) for 1D inventory

This work: optimal closed-form policies for multivariate continuous state and action MDPs

First optimal policies for multi-item inventory control on 50+ years!
Where do we start?

Previous Work on Continuous State and Discrete Actions
Hybrid State, Discrete Action MDPs

- Hybrid discrete / continuous state
  \[(\vec{b}, \vec{x}) = (b_1, \ldots, b_n, x_1, \ldots, x_m) \in \{0, 1\}^n \times \mathbb{R}^m\]

- Discrete action set \(a \in \mathcal{A}\)

- DBN factored transition model
  \[P(\vec{b'}, \vec{x'}|\vec{b}, \vec{x}, a) = \left( \prod_{i=1}^n P(b'_i|\vec{b}, \vec{x}, a) \right) \left( \prod_{j=1}^m P(x'_j|\vec{b}, \vec{b'}, \vec{x}, a) \right)\]

- Arbitrary action-dependent reward
  \[R_a(\vec{b}, \vec{x}) = x_1^2 + x_1 x_2\]
Value Iteration for Hybrid MDPs

- Value of policy in state is expected sum of rewards
- Want optimal value $V^{h,*}$ over horizons $h \in 0..H$
  - Implicitly provides optimal horizon-dependent policy
- Compute inductively via Value Iteration for $h \in 0..H$
  - Regression step:
    \[
    Q_a^{h+1}(\vec{b}, \vec{x}) = R_a(\vec{b}, \vec{x}) + \gamma \cdot \sum_{\vec{b}'} \int_{\vec{x}'} \left( \prod_{i=1}^{n} P(b'_i | \vec{b}, \vec{x}, a) \prod_{j=1}^{m} P(x'_j | \vec{b}, \vec{b}', \vec{x}, a) \right) V^h(\vec{b'}, \vec{x'}) d\vec{x'} \]
  - Maximization step:
    \[
    V_{h+1} = \max_{a \in A} Q_a^{h+1}(\vec{b}, \vec{x})
    \]
Exact Solutions to Hybrid MDPs: Domain

- **2-D Navigation**

- **State:** \((x, y) \in \mathbb{R}^2\)

- **Actions:**
  - move-x-2
    - \(x' = x + 2\)
    - \(y' = y\)
  - move-y-2
    - \(x' = x\)
    - \(y' = y + 2\)

- **Reward:**
  - \(R(x, y) = I[ (x > 5) \land (x < 10) \land (y > 2) \land (y < 5) ]\)

**Assumptions:**
1. Continuous transitions are deterministic and linear
2. Discrete transitions can be stochastic
3. Reward is **piecewise rectilinear**

---

Boyan & Littman NIPS-01 extended to n-D by Feng et al, UAI-04
Exact Solutions to Hybrid MDPs: Regression

- Continuous regression is just translation of “pieces”
Exact Solutions to Hybrid MDPs: Maximization

• Q-value maximization yields piecewise rectilinear solution

\[
\begin{align*}
\max_a Q(a,x,y) &= 1 \\
\text{(Orange area)} \\
\max_a Q(a,x,y) &= 0 \\
\text{(Gray area)}
\end{align*}
\]
Previous Work Limitations I

- Exact regression when transitions nonlinear?

**Action move-nonlin:**

- $x' = x^3 y + y^2$
- $y' = y \times \log(x^2 y)$

**How to compute boundary in closed-form?**
Previous Work Limitations II

• $\max(\ldots)$ when reward/value arbitrary piecewise?

\[ V(x, y) = \begin{cases} 1 \\
0 \end{cases} \]

Closed-form representation for max?
A solution to previous limitations:

Symbolic Dynamic Programming (SDP)

n.b., motivated by SDP from Boutilier et al (IJCAI-01) but here continuous instead of relational
Piecewise Functions (Cases)

\[ z = f(x, y) = \begin{cases} 
(x > 3) \land (y \cdot x) : & x + y \\
(x \cdot 3) \lor (y > x) : & x^2 + xy^3 
\end{cases} \]
Case Operations: $\oplus$, $\otimes$

$$\begin{cases}
\phi_1 : f_1 
\oplus 
\psi_1 : g_1 \\
\phi_2 : f_2 
\oplus 
\psi_2 : g_2 
\end{cases} = ?$$
Case Operations: $\oplus$, $\otimes$

\[
\begin{align*}
\phi_1 : f_1 & \quad \oplus \quad \psi_1 : g_1 \\
\phi_2 : f_2 & \quad \otimes \quad \psi_2 : g_2
\end{align*}
\]

$= \begin{cases} 
\phi_1 \land \psi_1 : f_1 + g_1 \\
\phi_1 \land \psi_2 : f_1 + g_2 \\
\phi_2 \land \psi_1 : f_2 + g_1 \\
\phi_2 \land \psi_2 : f_2 + g_2
\end{cases}$

- Similarly for $\otimes$
  - Expressions trivially closed under $+$, $\times$

- What about max?
  - $\text{max}(f_1, g_1)$ not pure arithmetic expression 😞
Case Operations: max

\[
\max \left( \left\{ \begin{array}{c}
\phi_1 : f_1 \\
\phi_2 : f_2
\end{array} \right\}, \left\{ \begin{array}{c}
\psi_1 : g_1 \\
\psi_2 : g_2
\end{array} \right\} \right) = ?
\]
Case Operations: max

\[
\text{max} \left( \left\{ \phi_1 : f_1, \phi_2 : f_2 \right\}, \left\{ \psi_1 : g_1, \psi_2 : g_2 \right\} \right) = \left\{ \begin{array}{l}
\phi_1 \land \psi_1 \land f_1 > g_1 : f_1 \\
\phi_1 \land \psi_1 \land f_1 \cdot g_1 : g_1 \\
\phi_1 \land \psi_2 \land f_1 > g_2 : f_1 \\
\phi_1 \land \psi_2 \land f_1 \cdot g_2 : g_2 \\
\phi_2 \land \psi_1 \land f_2 > g_1 : f_2 \\
\phi_2 \land \psi_1 \land f_2 \cdot g_1 : g_1 \\
\phi_2 \land \psi_2 \land f_2 > g_2 : f_2 \\
\phi_2 \land \psi_2 \land f_2 \cdot g_2 : g_2 
\end{array} \right. \\
\text{Key point: still in case form!} \\
\text{Size blowup? We’ll get to that…}
\]
Symbolic Dynamic Programming

• In a nutshell
  – $R(\cdot), P(\cdot | \cdot)$ defined as case statements
  – Value iteration uses case operations
    • $\oplus, \otimes, \max$
  – If all VI operations maintain case, then
    • $V^h(\cdot)$ is in case form for all horizons $h$!

• Almost there: we still need to define $\int_x$
SDP Regression Step

- Continuous variables $x_j$
  
  $- \int_{x} \delta[x - y] f(x) dx = f(y)$ triggers symbolic substitution

- e.g., $\int_{x'_j} \delta[x'_j - g(\bar{x})] V' dx'_j = V'\{x'_j / g(\bar{x})\}$

\[
\int_{x'_1} \delta[x'_1 - (x_1^2 + 1)] \left( \begin{array}{l}
  x'_1 < 2 : x'_1 \\
  x'_1 \geq 2 : x'_1^2
\end{array} \right) dx'_1 = \left\{ \begin{array}{l}
  x_1^2 + 1 < 2 : x_1^2 + 1 \\
  x_1^2 + 1 \geq 2 : (x_1^2 + 1)^2
\end{array} \right.
\]

- If $g$ is case: need conditional substitution
  
  • see Sanner et al (UAI 2011)
That’s Discrete Action SDP!

- Value Iteration for $h \in 0..H$
  
  - Regression step:
    
    $$Q_{a}^{h+1}(\vec{b}, \vec{x}) = R_a(\vec{b}, \vec{x}) + \gamma \cdot$$
    
    $$\sum_{\vec{b'}} \int_{\vec{x}'} \left( \prod_{i=1}^{n} P(b'_i | \vec{b}, \vec{x}, a) \prod_{j=1}^{m} P(x'_j | \vec{b}, \vec{b'}, \vec{x}, a) \right) V^h(\vec{b'}, \vec{x'}) d\vec{x'}$$

  - Maximization step:
    
    $$V_{h+1} = \max_{a \in A} Q_{a}^{h+1}(\vec{b}, \vec{x})$$
Continuous Actions

- Inventory control
  - Reorder based on stock, future demand
  - Action: \( a(\Delta); \Delta \in \mathbb{R}^{|a|} \)

- Need \( \max_{\Delta} \) in Bellman backup

\[
V_{h+1} = \max_{a \in A} \max_{\Delta} Q_{a}^{h+1}(b, \bar{x}, \Delta)
\]

- How to compute?
Max-out: $\max_x f(x)$

- How to compute for case?

\[
\begin{aligned}
\max_x & \quad \begin{cases} 
\phi_1: & f_1 \\
\vdots: & \vdots \\
\phi_k: & f_k 
\end{cases} \\
= & \max_x \max_{i=1\ldots k} \phi_i \cdot f_i \\
= & \max_{i=1\ldots k} \max_x \phi_i \cdot f_i
\end{aligned}
\]

- Just $\max_x$ case partitions, case-max results!
Example of Partition Max-out

\[ \max_x [\phi_1] \cdot f_1 \]

Consider function \(-\infty\) when constraints do not hold

\[ \phi_1 := [x > -1] \land [x > y - 1] \land [x \cdot z] \land [x \cdot y + 1] \land [y > 0] \]

\[ f_1 := x^2 - xy \]

\[ \max_{x \in \{LB, UB, Der0\}} f_1 \]

What constraints here?
- those independent of \(x\)
- pairwise \(UB > Der0 > LB\)

But how to evaluate?

Now an unconstrained max!
Max-out Case Operation

- \( \max_x \text{case}(x) \)
  - Reduced to partition max
    \( \ldots \max \) w.r.t. critical points

  - LB, UB
  - Der0\(_x\)

  - \( \max( \text{case}(x/LB), \text{case}(x/UB), \text{case}(x/\text{Der0}_x) ) \)

- Can even track substitutions through max to recover function of maximizing assignments!

See UAI 2011 paper for efficient substitutions into cases
Case $\rightarrow$ XADD

SDP needs an efficient data structure for

- compact, minimal case representation
- efficient case operations
XADDs

• Extended ADD representation of case statements

\[
V = \left\{ \begin{array}{ll}
  x_1 + k > 100 \land x_2 + k > 100 : & 0 \\
  x_1 + k > 100 \land x_2 + k \cdot 100 : & x_2 \\
  x_1 + k \cdot 100 \land x_2 + k > 100 : & x_1 \\
  x_1 + x_2 + k > 100 \land x_1 + k \cdot 100 \land x_2 + k \cdot 100 \land x_2 > x_1 : & x_2 \\
  x_1 + x_2 + k > 100 \land x_1 + k \cdot 100 \land x_2 + k \cdot 100 \land x_2 \cdot x_1 : & x_1 \\
  x_1 + x_2 + k \cdot 100 : & x_1 + x_2 \\
  \end{array} \right. 
\]
XADD Maximization

\[ \max( \begin{array}{c} y > 0 \\ y \\ x > 0 \end{array} , \begin{array}{c} x > 0 \\ y \\ x \end{array} ) = \begin{array}{c} y > 0 \\ x > 0 \\ x > 0 \end{array} \]

May introduce new decision tests
Maintaining XADD Orderings I

- Max may get variables out of order

Decision ordering (root→leaf)

\[
\text{max}(y > 0, x > 0) = \begin{cases} 
  y > 0 & \text{if } y > 0 \\
  x > y & \text{if } x > y \\
  x > 0 & \text{if } x > 0 
\end{cases}
\]

Newly introduced node is out of order!
Maintaining XADD Orderings II

• Substitution may get vars out of order

Decision ordering (root→leaf):

• $x > y$
• $y > 0$
• $x > z$

Substituted nodes are now out of order!
Correcting XADD Ordering

• Obtain *ordered* XADD from *unordered* XADD
  – key idea: binary operations maintain orderings

\[
\begin{align*}
z & \text{ is out of order} \\
\text{result will have } z \text{ in order!}
\end{align*}
\]

Inductively assume ID₁ and ID₀ are ordered.

All operands ordered, so applying \( \otimes, \oplus \) produces ordered result!
XADD Pruning

If linear, can detect with feasibility checker of LP solver & prune

Similar to Penberthy & Weld, AAAI-94
Empirical Results
Illustrative Example

\[ V^0(x) \]

\[ V^1(x) \]

\[ V^2(x) \]

Symbolic Value (Symbolic Policy: y=…)

-96 + 20 \cdot x - x \cdot x \cdot x (y = -10)

4 (y = -x)

-96 + 20 \cdot x - x \cdot x \cdot x (y = 10)
Reservoir Control

• Value Functions \textit{(vs level in each reservoir)}

• Policy
  \textit{(time to hold drain vs. reservoir levels)
Open Problems

• Nonlinear constraints
  – Optimal solutions for restricted cases e.g., quadratic, multilinear

• Bounded (interval) approximation
  – This XADD has > 1000 nodes!
Conclusions

• Key novel insights over Sanner et al (UAI 2011):
  – Introduced continuous actions
  – Showed how to compute $\max_x f(x)$ in closed form
  – All operations remains closed for value iteration

• Need compact case, efficient operations
  – Case $\rightarrow$ Extended ADD (XADD)
  – Extend to handle $\max_x f(x)$

• First exact, closed-form solutions to subset of n-D continuous state-action MDPs
  First exact policies for continuous variant of multivariate inventory control… unsolved for 50+ years!