Thresholded Rewards: Acting Optimally in Timed, Zero-Sum Games

Colin McMillen and Manuela Veloso

Presenter: Man Wang

Overview

- Zero-sum Games
- Markov Decision Problems
- Value Iteration Algorithm
- Thresholded Rewards MDP
- TRMDP Conversion
- Solution Extraction
- Heuristic Techniques
- Conclusion
- References

Zero-sum Games

Zero–sum game

- A participant's gains of utility -- Losses of the other participant

Cumulative intermediate reward

- The difference between our score and opponent's score

True reward

- Win, loss or tie
- Determined at the end based on intermediate reward

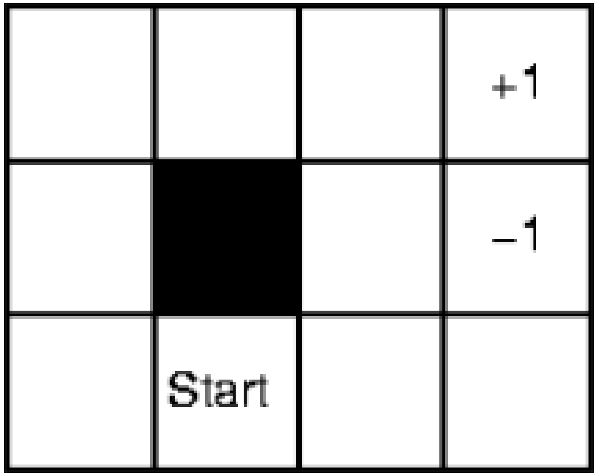
Markov Decision Problem

- Consider a non-perfect system
- Actions are performed with a probability less than 1
- What is the best action for an agent under this constraint?
- Example: A mobile robot does not exactly perform the desired action

Markov Decision Problem

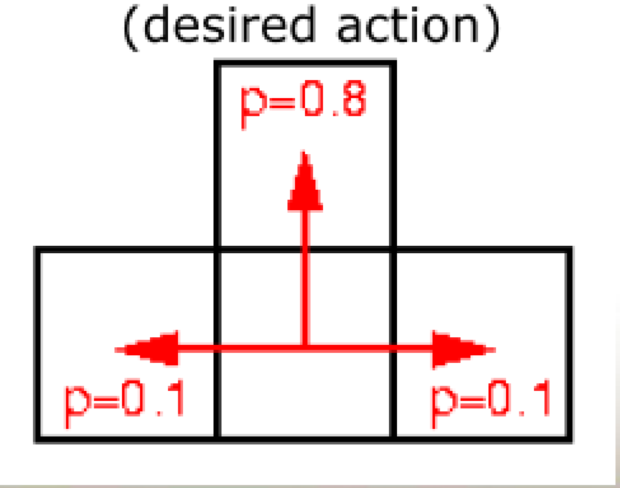
- Sound means of achieving optimal rewards in uncertain domains
- Find a policy maps state S to action A
- Maximize the cumulative long-term rewards

Value Iteration Algorithm



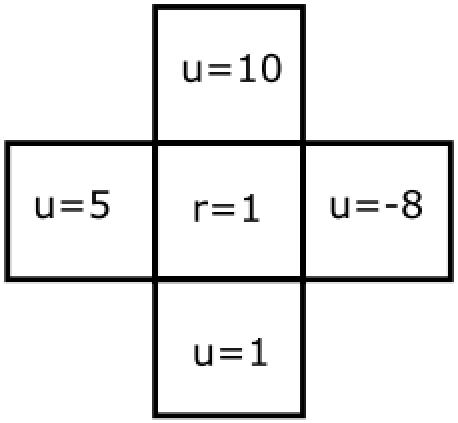
What is the best way to move to +1 without moving into -1?

Consider non-deterministic transition model:

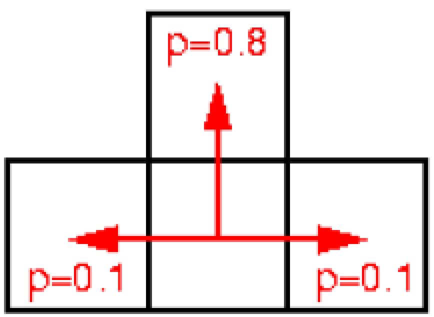


Value Iteration Algorithm

Calculate the utility of the center cell:



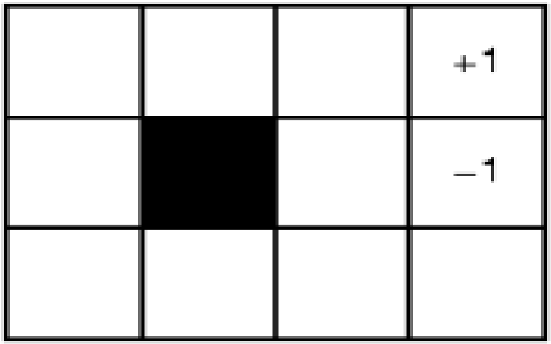
State Space



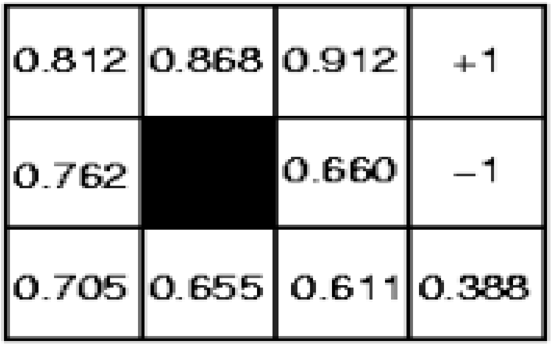
$$U_{t+1}(i) = R(i) + \max_a \sum_j M_{ij}^a \cdot U_t(j)$$

$$\begin{aligned}
= \; & reward + \max\{ \\
& 0.1 \cdot 1 + 0.8 \cdot 5 + 0.1 \cdot 10 \quad (\leftarrow), \\
& 0.1 \cdot 5 + 0.8 \cdot 10 + 0.1 \cdot -8 \quad (\uparrow), \\
& 0.1 \cdot 10 + 0.8 \cdot -8 + 0.1 \cdot 1 \quad (\rightarrow), \\
& 0.1 \cdot -8 + 0.8 \cdot 1 + 0.1 \cdot 5 \quad (\downarrow)\} \\
= \; & 1 + \max\{5.1\,(\leftarrow), 7.7\,(\uparrow), \\
& -5.3\,(\rightarrow), 0.5\,(\downarrow)\} \\
= \; & 1 + 7.7 \\
= \; & 8.7
\end{aligned}$$

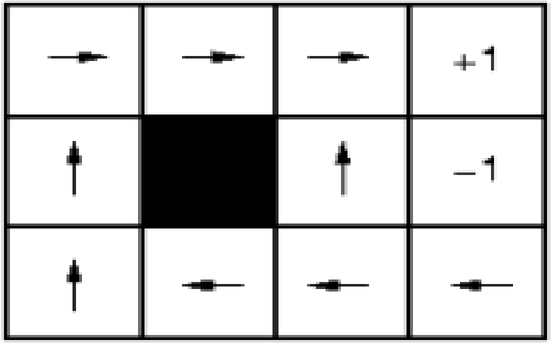
Value Iteration Algorithm



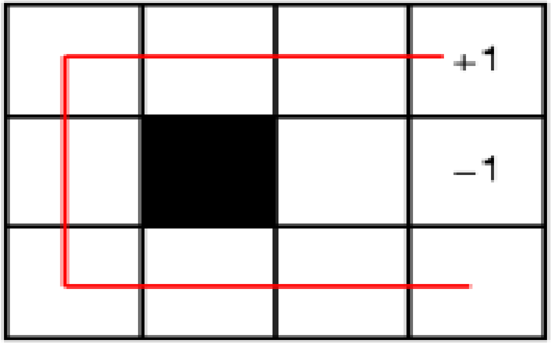
1. The given environment.



2. Calculate Utilities.



3. Extract optimal policy.



4. Execute actions.

Thresholded Rewards MDP

TRMDP (M, f, h):

- **M: MDP(S, A, T, R, s_0)**
- **f : threshold function**
- **$f(r_{intermediate}) = r_{true}$**
- **h : time horizon**

$$r_{true} = \begin{cases} 1 & \text{if } r_{intermediate} > 0 \\ 0 & \text{if } r_{intermediate} = 0 \\ -1 & \text{if } r_{intermediate} < 0. \end{cases}$$

Algorithm 1 Dynamics of a thresholded-rewards MDP.

$s \leftarrow s_0$
$r_{intermediate} \leftarrow 0$
for $t \leftarrow h$ to 1 **do**
  $a \leftarrow \pi(s, t, r_{intermediate})$
  $s \leftarrow s' \sim T(s, a)$
  $r_{intermediate} \leftarrow r_{intermediate} + R(s)$
$r_{true} \leftarrow f(r_{intermediate})$

Thresholded Rewards MDP

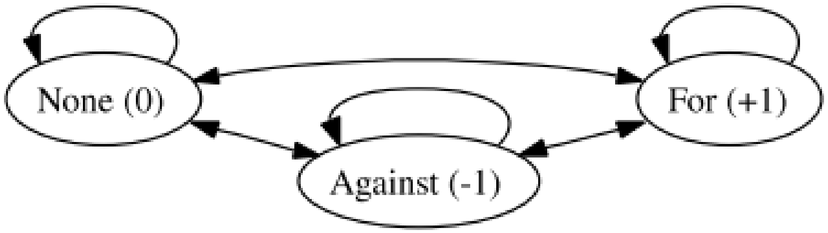
Example:

- **States:**

1. FOR: our team scored (reward +1)
2. AGAINST: opponent scored (reward -1)
3. NONE: no score occurs (reward 0)

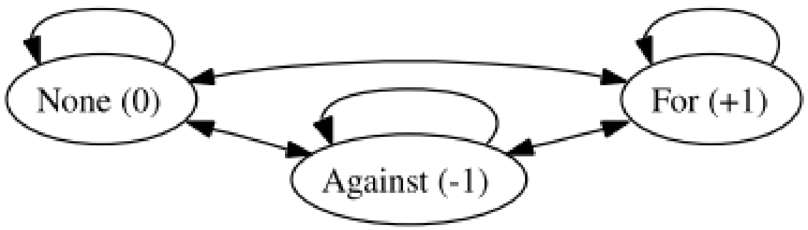
- **Actions:**

1. Balanced
2. Offensive
3. Defensive



a	$T(*, a, \text{FOR})$	$T(*, a, \text{AGAINST})$	$T(*, a, \text{NONE})$
balanced	0.05	0.05	0.9
offensive	0.25	0.5	0.25
defensive	0.01	0.02	0.97

Thresholded Rewards MDP



a	$T(*, a, \text{For})$	$T(*, a, \text{Against})$	$T(*, a, \text{None})$
balanced	0.05	0.05	0.9
offensive	0.25	0.5	0.25
defensive	0.01	0.02	0.97

Expected one step reward:

1. **Balanced: 0 = 0.05*1+0.05*(-1)+0.9*0**
2. **Offensive: -0.25 = 0.25*1+0. 5*(-1)+0.25*0**
3. **Defensive: -0.01 = 0.01*1+0.02*(-1)+0.97*0**

Suboptimal solution, true reward = 0

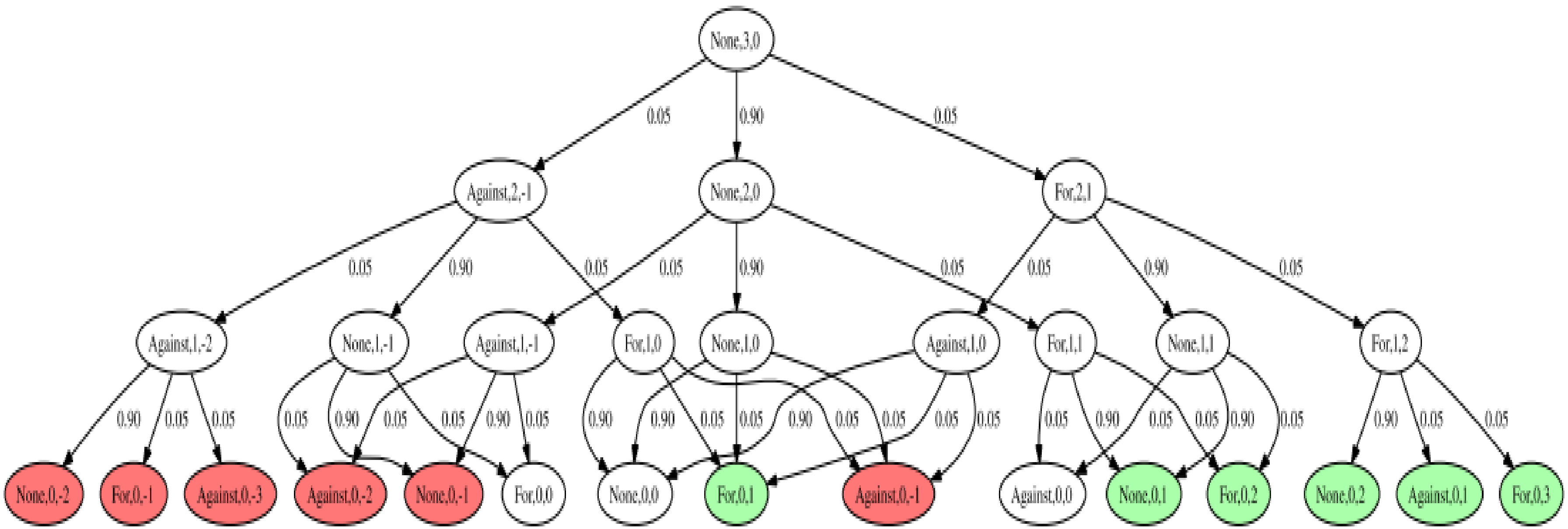
TRMDP Conversion

Algorithm 2 Converts a TRMDP (M, f, h) into an MDP M' suitable for finding the optimal thresholded-rewards policy.

Given: MDP $M = (S, A, T, R, s_0)$, threshold function f, time horizon h

$s'_0 \leftarrow (s_0, h, 0)$
$S' \leftarrow \{s'_0\}$
for $i \leftarrow h$ to 1 **do**
    for all states $s'_1 = (s_1, t, ir) \in S'$ such that $t = i$ **do**
        for all transitions $T(s_1, a, s_2)$ in M **do**
            $s'_2 \leftarrow (s_2, t-1, ir + R(s_2))$
            $S' \leftarrow S' \cup \{s'_2\}$
            $T'(s'_1, a, s'_2) = T(s_1, a, s_2)$
for all states $s' = (s, t, ir)$ in M' **do**
    if $t = 0$ **then**
        $R'(s') \leftarrow f(ir)$
    else
        $R'(s') \leftarrow 0$
return $M' = (S', A, T', R', s'_0)$

TRMDP Conversion



The MDP M’ given MDP M and h=3

Solution Extraction

Two important facts:

- M’ has a layered, feed-forward structure: every layer contains transitions only into the next layer
- At iteration k of value iteration, the only values that change are those for the states s’=(s, t, ir) such that t=k

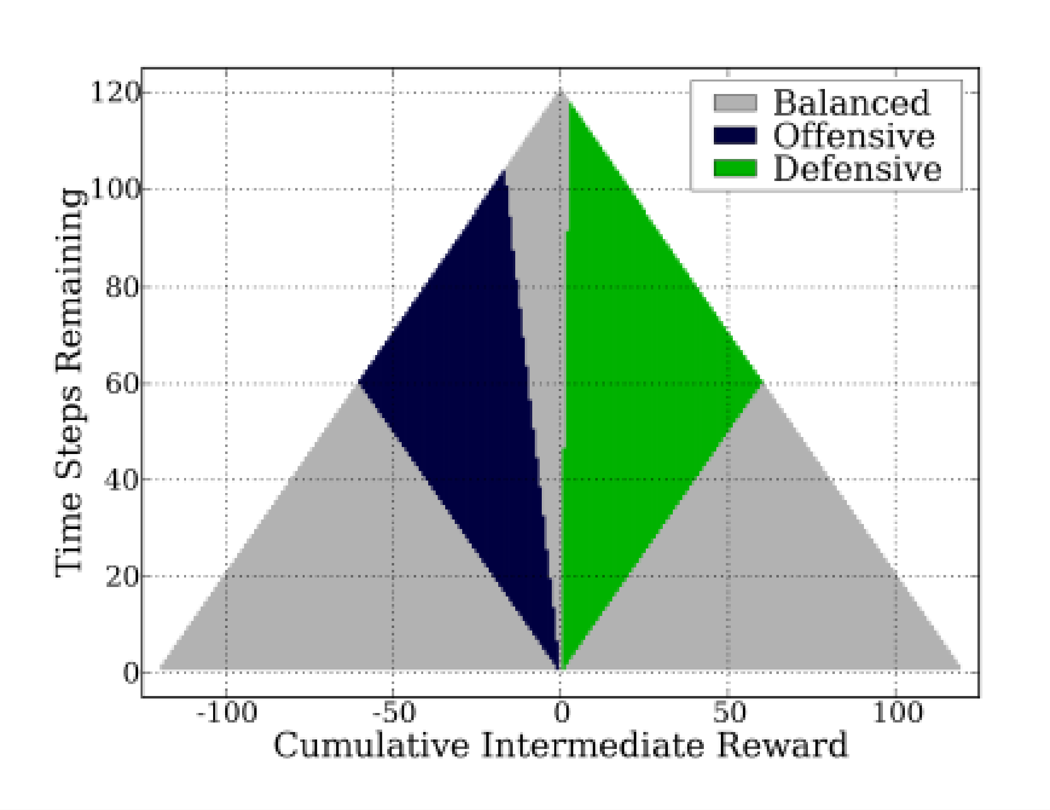
Solution Extraction

Expected reward = 0.1457

Win : 50%

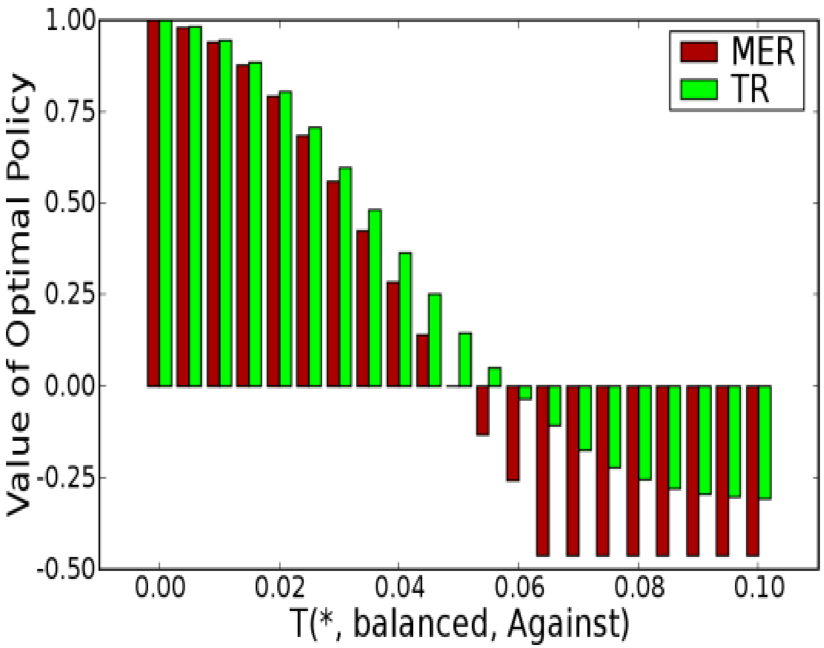
Lose: 35%

Tie : 15%

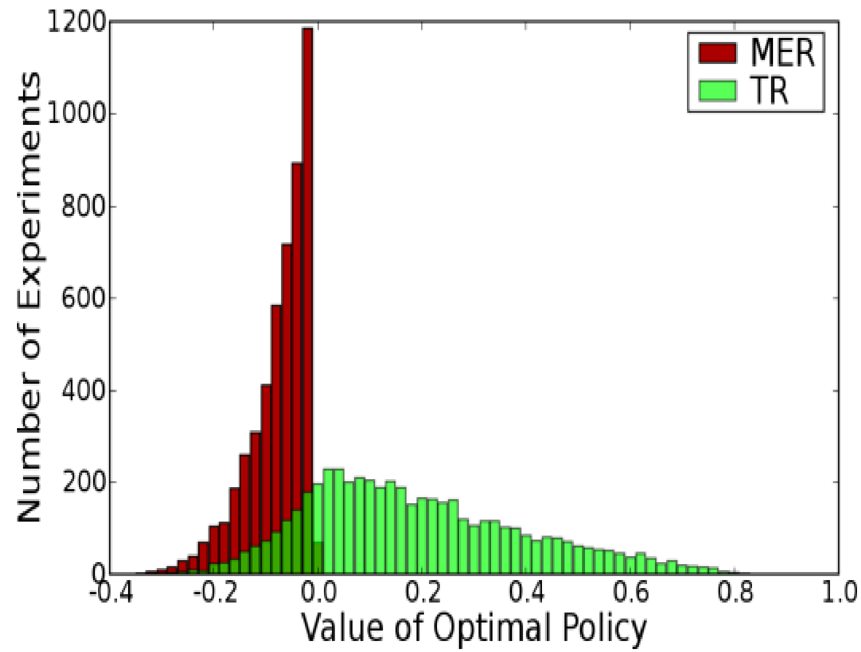


Optimal policy for M and h=120

Solution Extraction



Effect of changing opponent's capabilities



Performance of MER vs TR on 5000 random MDPs

Heuristic Techniques

- **Uniform-k heuristic**
- **Lazy-k heuristic**
- **Logarithmic-k-m heuristic**
- **Experiments**

Uniform-k heuristic

- Adopt non-stationary policy
- Change policy every k time steps
- Compress the time horizon uniformly by factor k
- Solution is suboptimal

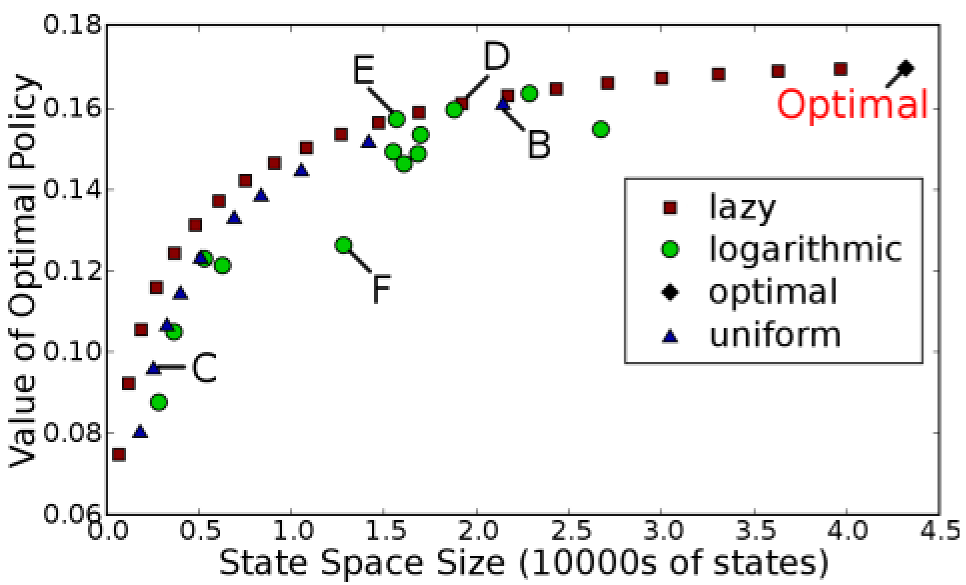
Lazy-k heuristic

- **More than k steps remaining:**
  No reward threshold
- **K steps remaining:**
  Create threshold rewards MDP
  Time horizon k
  Current state as initial state

Logarithmic-k-m heuristic

- Time resolution becomes finer when approaching the time horizon
- k – Number of decisions made before the time resolution increased
- m – The multiple by which the resolution is increased
- For instance, k=10,m=2 means that 10 actions before each increase, time resolution doubles on each increase

Experiment



60 different MDPs randomly chosen from the 5000 MDPs in previous experiment

Uniform-k suffers from large state size

Logarithmic highly depend on parameters

Lazy-k provides high true reward with low number of states

Conclusion

- **Introduced thresholded-rewards problem in finite-horizon environment**
  - **Intermediate rewards**
  - **True reward at the end of horizon**
  - **Maximize the probability of winning**
- **Present an algorithm converts base MDP to expanded MDP**
- **Investigate three heuristic techniques generating approximate solutions**

References

1. Bacchus, F.; Boutilier, C.; and Grove, A. 1996. Rewarding behaviors. In Proc. AAAI-96.
2. Guestrin, C.; Koller, D.; Parr, R.; and Venkataraman, S. 2003. Efficient solution algorithms for factored MDPs. JAIR.
3. Hoey, J.; St-Aubin, R.; Hu, A.; and Boutilier, C. 1999. SPUDD: Stochastic planning using decision diagrams. In Proceedings of Uncertainty in Artificial Intelligence.
4. Kaelbling, L. P.; Littman, M. L.; and Moore, A. W. 1996. Reinforcement learning: A survey. JAIR.
5. Kearns, M. J.; Mansour, Y.; and Ng, A. Y. 2002. A sparse sampling algorithm for near-optimal planning in large Markov decision processes. Machine Learning.

References

6. Li, L.; Walsh, T. J.; and Littman, M. L. 2006. Towards a unified theory of state abstraction for MDPs. In Symposium on Artificial Intelligence and Mathematics.
7. Mahadevan, S. 1996. Average reward reinforcement learning: Foundations, algorithms, and empirical results. Machine Learning 22(1-3):159–195.
8. McMillen, C., and Veloso, M. 2006. Distributed, play-based role assignment for robot teams in dynamic environments. In Proc. Distributed Autonomous Robotic Systems.
9. Puterman, R. L. 1994. Markov Decision Processes: Discrete Stochastic Dynamic Programming. Wiley.
10. Stone, P. 1998. Layered Learning in Multi-Agent Systems. Ph.D. Dissertation, Carnegie Mellon University.