

## Lecture 14

### Artificial Neural Networks (Discussion 2 of 4): Temporal-Difference Reinforcement Learning

Friday, February 18, 2000

William H. Hsu  
Department of Computing and Information Sciences, KSU  
<http://www.cis.ksu.edu/~bhsu>

Readings:  
Sections 20.2-20.7, Russell and Norvig  
Sections 13.5-13.8, Mitchell



CIS 830: Advanced Topics in Artificial Intelligence

Kansas State University  
Department of Computing and Information Sciences

## Lecture Outline

- Readings: 13.1-13.4, Mitchell; 20.2-20.7, Russell and Norvig
- Outside Reading: "Connectionist Learning Procedures", Hinton
- Suggested Exercises: 13.4, Mitchell; 20.11, Russell and Norvig
- **Reinforcement Learning (RL) Concluded**
  - Control policies that choose optimal actions
  - MDP framework, continued
  - Continuing research topics
    - Active learning: experimentation (exploration) strategies
    - Generalization in RL
    - Next: ANNs and GAs for RL
- **Temporal Difference (TD) Learning**
  - Family of dynamic programming algorithms for RL
    - Generalization of Q learning
    - More than one step of lookahead
  - More on TD learning in action



CIS 830: Advanced Topics in Artificial Intelligence

Kansas State University  
Department of Computing and Information Sciences

## Quick Review: Policy Learning Framework



- **Interactive Model**
  - State  $s$  (may be partially observable)
  - Agent selects action  $a$  based upon (current) policy
    - Incremental reward (aka **reinforcement**)  $r(s, a)$  presented to agent
    - Taking action puts agent into new state  $s' = \delta(s, a)$  in environment
  - Agent uses **decision cycle** to estimate  $s'$ , compute outcome distributions, select new actions
- **Reinforcement Learning Problem**
  - Given
    - Observation sequence  $s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} s_2 \xrightarrow{a_2} \dots$
    - **Discount factor**  $\gamma \in [0, 1)$
  - Learn to: choose actions that maximize  $r(t) + \gamma r(t+1) + \gamma^2 r(t+2) + \dots$



CIS 830: Advanced Topics in Artificial Intelligence

Kansas State University  
Department of Computing and Information Sciences

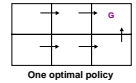
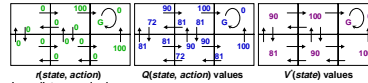
## Quick Review: Q Learning

- **Deterministic World Scenario**
  - "Knowledge-free" (here, **model-free**) search for policy  $\pi$  from policy space  $\Pi$
  - For each possible policy  $\pi \in \Pi$ , can define an evaluation function over states:
 
$$V^\pi(s) \equiv r(t) + \gamma r(t+1) + \gamma^2 r(t+2) + \dots$$

$$\equiv \sum_{i=0}^{\infty} \gamma^i r(t+i)$$
 where  $r(t), r(t+1), r(t+2), \dots$  are generated by following policy  $\pi$  starting at state  $s$
  - Restated, task is to learn optimal policy  $\pi^*$ 

$$\pi^* \equiv \arg \max_{\pi} V^\pi(s), \forall s$$

### Finding Optimal Policy



- **Q-Learning Training Rule**  $\hat{Q}(s, a) \leftarrow r(s, a) + \gamma \max_{a'} \hat{Q}(s', a')$



CIS 830: Advanced Topics in Artificial Intelligence

Kansas State University  
Department of Computing and Information Sciences

## Learning Scenarios

- **First Learning Scenario**
  - **Passive learning in known environment** (Section 20.2, Russell and Norvig)
  - **Intuition** (passive learning in known and unknown environments)
    - Training sequences  $(s_1, s_2, \dots, s_n, r = U(s_n))$
    - Learner has fixed policy  $\pi$ ; determine benefits (expected total reward)
  - **Important note:** *known  $\neq$  accessible  $\neq$  deterministic* (even if transition model known, state may not be directly observable and may be stochastic)
  - Solutions: naïve updating (LMS), dynamic programming, temporal differences
- **Second Learning Scenario**
  - **Passive learning in unknown environment** (Section 20.3, Russell and Norvig)
  - Solutions: LMS, temporal differences; adaptation of dynamic programming
- **Third Learning Scenario**
  - **Active learning in unknown environment** (Sections 20.4-20.6, Russell and Norvig)
  - Policy must be learned (e.g., through application and exploration)
  - Solutions: dynamic programming (Q-learning), temporal differences



CIS 830: Advanced Topics in Artificial Intelligence

Kansas State University  
Department of Computing and Information Sciences

## Reinforcement Learning Methods

- **Solution Approaches**
  - Naïve updating: **least-mean-square (LMS)** utility update
  - **Dynamic programming (DP): solving constraint equations**
    - **Adaptive DP (ADP):** includes value iteration, policy iteration, exact Q-learning
    - **Passive case:** teacher selects sequences (**trajectories** through environment)
    - **Active case:** **exact** Q-learning (recursive exploration)
  - **Method of temporal differences (TD): approximating constraint equations**
    - **Intuitive idea:** use **observed transitions** to adjust  $U(s)$  or  $Q(s, a)$
    - **Active case:** **approximate** Q-learning (TD Q-learning)
- **Passive: Examples**
  - Temporal differences:  $U(s) \leftarrow U(s) + \gamma(R(s) + U(s') - U(s))$
  - **No exploration function**
- **Active: Examples**
  - ADP (value iteration):  $U(s) \leftarrow R(s) + \gamma \max_a (\sum_{s'} M_{s,s'}(a) \cdot U(s'))$
  - Exploration (exact Q-learning):  $\hat{Q}(s, a) \leftarrow r(s, a) + \gamma \max_{a'} \hat{Q}(s', a')$




CIS 830: Advanced Topics in Artificial Intelligence

Kansas State University  
Department of Computing and Information Sciences

## Active Learning and Exploration


- Active Learning Framework**
  - So far: optimal behavior is to choose action with maximum expected utility (MEU), given current estimates
  - Proposed revision: action has two outcomes
    - Gains rewards on current sequence (agent preference: **greed**)
    - Affects percepts → ability of agent to learn → ability of agent to receive future rewards (agent preference: "investment in education", aka **novelty, curiosity**)
  - Tradeoff: comfort (lower risk) reduced payoff versus higher risk, high potential
  - Problem: how to quantify tradeoff, reward latter case?
- Exploration**
  - Define: **exploration function** - e.g.,  $f(u, n) = (n < M) ? R^* : u$ 
    - $u$ : expected utility under **optimistic** estimate;  $f$  increasing in  $u$  (greed)
    - $n = M(s, a)$ : number of trials of action-value pair;  $f$  decreasing in  $n$  (curiosity)
  - Optimistic** utility estimator:  $U^*(s) \leftarrow R(s) + \gamma \max_a f(\sum_{s'} (M_{s,s}(a) \cdot U^*(s')), M(s, a))$
- Key Issues: Generalization (Today); Allocation (CIS 830)**



CIS 830: Advanced Topics in Artificial Intelligence Kansas State University  
Department of Computing and Information Sciences

## Temporal Difference Learning: Rationale and Formula


- Q-Learning**
  - Reduce discrepancy between successive estimates
  - $Q$  estimates
    - One step time difference
    - $Q^{(1)}(s(t), a(t)) = r(t) + \gamma \max_a \hat{Q}(s(t+1), a)$
- Method of Temporal Differences (TD( $\lambda$ )), aka Temporal Differencing**
  - Why not two steps?
    - $Q^{(2)}(s(t), a(t)) = r(t) + \gamma r(t+1) + \gamma^2 \max_a \hat{Q}(s(t+2), a)$
  - Or  $n$  steps?
    - $Q^{(n)}(s(t), a(t)) = r(t) + \gamma r(t+1) + \dots + \gamma^{n-1} r(t+n-1) + \gamma^n \max_a \hat{Q}(s(t+n), a)$
  - TD( $\lambda$ ) formula
    - Blends all of these
    - $Q^\lambda(s(t), a(t)) = (1-\lambda) [ Q^{(1)}(s(t), a(t)) + \lambda Q^{(2)}(s(t), a(t)) + \lambda^2 Q^{(3)}(s(t), a(t)) + \dots ]$
  - Intuitive idea:** use constant  $0 \leq \lambda \leq 1$  to combine estimates from various lookahead distances (note normalization factor  $1-\lambda$ )



CIS 830: Advanced Topics in Artificial Intelligence Kansas State University  
Department of Computing and Information Sciences

## Temporal Difference Learning: TD( $\lambda$ ) Training Rule and Algorithm


- Training Rule: Derivation from Formula**
  - Formula:  $Q^\lambda(s(t), a(t)) = (1-\lambda) [ Q^{(1)}(s(t), a(t)) + \lambda Q^{(2)}(s(t), a(t)) + \lambda^2 Q^{(3)}(s(t), a(t)) + \dots ]$
  - Recurrence equation for  $Q^\lambda(s(t), a(t))$  (recursive definition) defines update rule
    - Select  $a(t+i)$  based on current policy
    - $Q^\lambda(s(t), a(t)) = r(t) + \gamma [ (1-\lambda) \max_a \hat{Q}(s(t+1), a) + \lambda Q^\lambda(s(t+1), a(t+1)) ]$
- Algorithm**
  - Use above training rule
  - Properties
    - Sometimes converges faster than  $Q$  learning
    - Converges for learning  $V^*$  for any  $0 \leq \lambda \leq 1$  [Dayan, 1992]
    - Other results [Sutton, 1988; Peng and Williams, 1994]
  - Application: Tesauro's *TD-Gammon* uses this algorithm [Tesauro, 1995]
  - Recommended book
    - Reinforcement Learning* [Sutton and Barto, 1998]
    - <http://www.cs.umass.edu/~rich/book/the-book.html>



CIS 830: Advanced Topics in Artificial Intelligence Kansas State University  
Department of Computing and Information Sciences

## Applying Results of RL: Models versus Action-Value Functions


- Distinction: Learning Policies with and without Models**
  - Model-theoretic approach
    - Learning: transition function  $\delta$ , utility function  $U$
    - ADP component: value/policy iteration to reconstruct  $U$  from  $R$
    - Putting learning and ADP components together: **decision cycle** (Lecture 17)
    - Function **Active-ADP-Agent**: Figure 20.9, Russell and Norvig
  - Contrast:  $Q$ -learning
    - Produces estimated action-value function
    - No environment model (i.e., no explicit representation of state transitions)
    - NB:** this includes both exact and approximate (e.g., TD)  $Q$ -learning
    - Function **Q-Learning-Agent**: Figure 20.12, Russell and Norvig
- Ramifications: A Debate**
  - Knowledge in model-theoretic approach corresponds to "pseudo-experience" in TD (see: 20.3, Russell and Norvig; **distal supervised learning**; **phantom induction**)
  - Dissenting conjecture: model-free methods "reduce need for knowledge"
  - At issue: *when is it worth while to combine analytical, inductive learning?*



CIS 830: Advanced Topics in Artificial Intelligence Kansas State University  
Department of Computing and Information Sciences

## Applying Results of RL: MDP Decision Cycle Revisited


- Function Decision-Theoretic-Agent (Percept)**
  - Percept: agent's input; collected evidence about world (from sensors)
  - COMPUTE updated **probabilities** for current state based on available evidence, including current percept and previous action (prediction, estimation)
  - COMPUTE **outcome probabilities** for actions, given action descriptions and probabilities of current state (decision model)
  - SELECT **action** with highest expected utility, given probabilities of outcomes and utility functions
  - RETURN **action**
- Situated Decision Cycle**
  - Update percepts, collect rewards
  - Update active model (prediction and estimation; decision model)
  - Update utility function: value iteration
  - Selecting action to maximize expected utility: performance element
- Role of Learning: Acquire State Transition Model, Utility Function**



CIS 830: Advanced Topics in Artificial Intelligence Kansas State University  
Department of Computing and Information Sciences

## Generalization in RL

- Explicit Representation**
  - One output value for each input tuple
  - Assumption: functions represented in **tabular form** for DP
    - Utility  $U$ : state → value,  $U_s$ ; state vector → value
    - Transition  $M$ : state × state × action → probability
    - Reward  $R$ : state → value,  $r$ ; state × action → value
    - Action-value  $Q$ : state × action → value
  - Reasonable for small state spaces, breaks down rapidly with more states
    - ADP convergence, time per iteration becomes unmanageable
    - "Real-world" problems and games: still intractable even for approximate ADP
- Solution Approach: Implicit Representation**
  - Compact representation:** allows calculation of  $U, M, R, Q$
  - e.g., checkers:  $\hat{V}(b) = w_o + w_b p(b) + w_r r p(b) + w_g g k(b) + w_r k(b) + w_b b t(b) + w_g r t(b)$
- Input Generalization**
  - Key benefit of compact representation: **inductive generalization over states**
  - Implicit representation: RL :: representation bias: supervised learning



CIS 830: Advanced Topics in Artificial Intelligence Kansas State University  
Department of Computing and Information Sciences

## Relationship to Dynamic Programming

- **Q-Learning**
  - Exact version closely related to DP-based MDP solvers
  - Typical assumption: perfect knowledge of  $\delta(s, a)$  and  $r(s, a)$
  - NB: remember, does not mean
    - Accessibility (total observability of  $s$ )
    - Determinism of  $\delta, r$
- **Situated Learning**
  - aka *in vivo*, *online*, *lifelong* learning
  - Achieved by moving about, interacting with real environment
  - Opposite: *simulated*, *in vitro* learning
- **Bellman's Equation** [Bellman, 1957]
 
$$(\forall s \in S). V'(s) = E[r(s, \pi(s)) + \gamma V'(\delta(s, \pi(s)))]$$
  - Note very close relationship to definition of optimal policy:
 
$$\pi^* = \arg \max_{\pi} V^{\pi}(s), \forall s$$
  - Result:  $\pi$  satisfies above equation iff  $\pi = \pi^*$

KSU

CIS 830: Advanced Topics in Artificial Intelligence Kansas State University  
Department of Computing and Information Sciences

## Subtle Issues and Continuing Research

- **Current Research Topics**
  - Replace table of  $Q$  estimates with ANN or other generalizer
    - Neural reinforcement learning (next time)
    - Genetic reinforcement learning (next week)
  - Handle case where state only partially observable
    - Estimation problem clear for ADPs (many approaches, e.g., Kalman filtering)
    - How to learn  $Q$  in MDPs?
  - Optimal exploration strategies
  - Extend to continuous action, state
  - Knowledge: incorporate or attempt to discover?
- **Role of Knowledge in Control Learning**
  - Method of incorporating domain knowledge: simulated experiences
    - *Distal supervised learning* [Jordan and Rumelhart, 1992]
    - *Pseudo-experience* [Russell and Norvig, 1995]
    - *Phantom induction* [Brodie and Dejong, 1998]
  - TD Q-learning: knowledge discovery or brute force (or both)?

KSU

CIS 830: Advanced Topics in Artificial Intelligence Kansas State University  
Department of Computing and Information Sciences

## RL Applications: Game Playing

- **Board Games**
  - Checkers
    - Samuel's player [Samuel, 1959]: precursor to temporal difference methods
    - Early case of *multi-agent learning* and *co-evolution*
  - Backgammon
    - Predecessor: *Neurogammon* (backprop-based) [Tesauro and Sejnowski, 1989]
    - *TD-Gammon*: based on TD( $\lambda$ ) [Tesauro, 1992]
- **Robot Games**
  - Soccer
    - *RoboCup* web site: <http://www.robocup.org>
    - Soccer server manual: <http://www.dsv.su.se/~johank/RoboCup/manual/>
  - Air hockey: <http://cyclops.csl.uiuc.edu>
- **Discussions Online (Other Games and Applications)**
  - Sutton and Barto book: <http://www.cs.umass.edu/~rich/book/t1/node1.html>
  - Sheppard's thesis: <http://www.cs.jhu.edu/~sheppard/thesis/node32.html>

KSU

CIS 830: Advanced Topics in Artificial Intelligence Kansas State University  
Department of Computing and Information Sciences

## RL Applications: Control and Optimization

- **Mobile Robot Control: Autonomous Exploration and Navigation**
  - USC Information Sciences Institute (Shen *et al*): <http://www.isi.edu/~shen>
  - Fribourg (Perez): <http://lslwww.epfl.ch/~aperez/robotreinfo.html>
  - Edinburgh (Adams *et al*): <http://www.dai.ed.ac.uk/groups/mrg/MRG.html>
  - CMU (Mitchell *et al*): <http://www.cs.cmu.edu/~rl/>
- **General Robotics: Smart Sensors and Actuators**
  - CMU robotics FAQ: <http://www.frc.ri.cmu.edu/robotics-faq/TOC.html>
  - Colorado State (Anderson *et al*): <http://www.cs.colostate.edu/~anderson/res/rl/>
- **Optimization: General Automation**
  - Planning
    - UM Amherst: <http://eksl-www.cs.umass.edu/planning-resources.html>
    - USC ISI (Knoblock *et al*) <http://www.isi.edu/~knoblock>
  - Scheduling: <http://www.cs.umass.edu/~rich/book/t1/node7.html>

KSU

CIS 830: Advanced Topics in Artificial Intelligence Kansas State University  
Department of Computing and Information Sciences

## Terminology

- **Reinforcement Learning (RL)**
  - Definition: learning policies  $\pi : state \rightarrow action$  from  $\langle\langle state, action \rangle\rangle, reward\rangle$ 
    - Markov decision problems (MDPs): finding control policies to choose optimal actions
    - Q-learning: produces action-value function  $Q : state \times action \rightarrow value$  (expected utility)
  - Active learning: *experimentation (exploration) strategies*
    - Exploration function:  $f(u, n)$
    - Tradeoff: *greed* ( $u$ ) preference versus *novelty* ( $1/n$ ) preference, aka *curiosity*
- **Temporal Difference (TD) Learning**
  - $\lambda$ : constant for blending alternative training estimates from multi-step lookahead
  - TD( $\lambda$ ): algorithm that uses recursive training rule with  $\lambda$ -estimates
- **Generalization in RL**
  - *Explicit representation*: tabular representation of  $U, M, R, Q$
  - *Implicit representation*: compact (aka compressed) representation

KSU

CIS 830: Advanced Topics in Artificial Intelligence Kansas State University  
Department of Computing and Information Sciences

## Summary Points

- **Reinforcement Learning (RL) Concluded**
  - Review: RL framework (learning from  $\langle\langle state, action \rangle\rangle, reward\rangle$ )
  - Continuing research topics
    - Active learning: *experimentation (exploration) strategies*
    - Generalization in RL: made possible by implicit representations
- **Temporal Difference (TD) Learning**
  - Family of algorithms for RL: generalizes Q-learning
  - More than one step of lookahead
  - Many more TD learning results, applications: [Sutton and Barto, 1998]
- **More Discussions Online**
  - Harmon's tutorial: <http://www-anw.cs.umass.edu/~mharmon/rltutorial/>
  - CMU RL Group: <http://www.cs.cmu.edu/Groups/reinforcement/www/>
  - Michigan State RL Repository: <http://www.cse.msu.edu/rl/>
- **Next Time: Presentation on Generative Models (Wake-Sleep Algorithm)**
  - Based on *associative memory* for pattern recognition
  - Related to distal supervised learning (previous), Bayesian networks (next)

KSU

CIS 830: Advanced Topics in Artificial Intelligence Kansas State University  
Department of Computing and Information Sciences