Backgammon

- Ancient board game.
- Race around the board depending on the value on the dice.
- How can we create an expert-level backgammon playing program?

2. Train it with expert examples.
3. Let it learn from playing itself.
Gerry Tesauro of IBM

• Gerry’s The Man.

• His Neurogammon used neural networks to learn to evaluate board positions in a way that matched expert decisions. Played very well. Never better than the experts, though.

• He then created TD-Gammon, which plays millions of games with itself and learns how to win. Best. Program. Ever.

• Go, reinforcement learning!

Where We’re Going

Introduce reinforcement learning

• why I think it’s exciting

Define the problem and current approaches

• challenges of RL in real environments

Focused topics:

• efficient exploration
• rich sensors
• partial observability
• social learning
Understanding Intelligence

Why?

- To shed light on who we are.
- To create artifacts that can make the world a better place.

How?

- Arguments from insufficiency (e.g., paucity of the stimulus): Cognitive Science, Psychology, Philosophy
- Arguments from sufficiency (e.g., demonstrate that an algorithmic approach can achieve intelligent behavior): Computer Science, Artificial Intelligence

Impressive Accomplishment

Honda’s Asimo

- development began in 1999, building on 13 years of engineering experience.
- claimed “most advanced humanoid robot ever created”
- walks 1mph
And Yet…

Asimo is programmed/controlled by people:
- structure of the walk programmed in
- reactions to perturbations programmed in
- directed by technicians and puppeteers during the performance
- no camera-control loop
- static stability

Compare To Kids

Molly
- development began in 1999
- “just an average kid”
- walks 2.5mph even on unfamiliar terrain
- very hard to control
- dynamically stable (sometimes)
Crawl Before Walk

Impressive accomplishment:

- Fastest reported walk/crawl on an Aibo
- Gait pattern optimized automatically

Human "Crawling"

Perhaps our programming isn't for crawling at all, but for the desire for movement!
Reinforcement-Learning Hypothesis

Intelligent behavior arises from the actions of an individual seeking to maximize its received reward signals in a complex and changing world.

Research program:

- identify where reward signals come from,
- develop algorithms that search the space of behaviors to maximize reward signals.

The RL Problem

Input: \(<s_1, a_1, s_2, r_1>, <s_2, a_2, s_3, r_2>, \ldots, s_t>\)

Output: \(a_t\)s to maximize discounted sum of \(r_t\)s.
Find The Ball Task

Learn:
• which way to turn
• to minimize steps
• to see goal (ball)
• from camera input
• given experience.

Problem Formalization: MDP

Most popular formalization: Markov decision process

Assume:
• States/sensations, actions discrete.
• Transitions, rewards stationary and Markov.
• Transition function: $\Pr(s' \mid s, a) = T(s, a, s')$.
• Reward function: $E[r \mid s, a] = R(s, a)$.

Then:
• Optimal policy $\pi^*(s) = \arg\max_a Q^*(s, a)$
• where $Q^*(s, a) = R(s, a) + \gamma \sum_{s'} T(s, a, s') \max_{a'} Q^*(s', a')$. 
Find the Ball: MDP Version

• Actions: rotate left/right

• States: orientation

• Reward: +1 for facing ball, 0 otherwise

It Can Be Done: Q-learning

Since optimal Q function is sufficient, use experience to estimate it (Watkins & Dayan 92).

Given \(<s, a, s', r>:\)

\[ Q(s,a) \leftarrow Q(s,a) + \alpha_t (r + \gamma \max_{a'} Q(s',a') - Q(s,a)) \].

If:

• all \((s,a)\) pairs updated infinitely often

\(\text{Pr}(s'|s,a) = T(s,a,s'), \ E[r|s,a] = R(s,a)\)

• \(\Sigma\alpha_t = \infty, \Sigma\alpha_t^2 < \infty\)

Then: \(Q(s,a) \rightarrow Q^*(s,a).\)
Real-Life Reinforcement Learning

Emphasize learning with real* data.

Q-learning good, but might not be right here...

Mismatches to “Find the Ball” MDP:

- **Efficient exploration**: data is expensive.
- **Rich sensors**: never see the same thing twice.
- **Aliasing**: different states can look similar.
- **Social**: interaction with others important.

* Or, if simulated, from simulators developed outside the AI community.

Efficient Exploration

Limit is nice, but would like something faster.

**Goal**: Policy that’s $\epsilon$ optimal with prob. $1-\delta$ after polynomial amount of experience.

$E^3$ (Kearns & Singh 98):

- Use experience to estimate model ($T$ and $R$).
- Find optimal greedy policy wrt the model.
- Use **model uncertainty** to guide exploration.

Similar to $R_{MAX}$ (Brafman & Tennenholtz 02).
**Exploration/Exploitation**

Two roads diverged in a yellow wood,
And sorry I could not travel both
And be one traveler, long I stood
And looked down one as far as I could
To where it bent in the undergrowth;

- In RL: A system cannot make two decisions and be one decision maker---it can only observe the effects of the actions it *actually chooses to make*. Do the right thing *and* learn.

**Pangloss Assumption**

*We are in the best of all possible worlds.*

Confidence intervals are on model parameters.

Find the *model* that gives maximum reward subject to the constraint that all parameters lie within their confidence intervals.

Choose actions that are best for this model.

In bandit case, this works out to precisely IE.

Very general, but can be intractable.

*Solvable for MDPs.* ([Strehl & Littman 04, 05])
Some (non-real) MBIE Results

Each plot measures cumulative reward by trial.
Varied exploration parameters (6-arms MDP).

Robotic Example

• Compared efficient exploration with naive (random) exploration.
• Task learned much faster (and better).
### Some RL Concepts

- **Rewards**: Define the agent’s task. Tries to maximize (expected) sum.

- **Shaping rewards**: Additional rewards not strictly required for influencing behavior but to provide a “hint” during learning.

- **Model**: The agent’s view of how the world works, based on its experience.

- **Planning**: Use of the learned model to make decisions to maximize rewards.

### Rich Sensors

Can treat experience as state via instance-based view.

With an appropriate similarity function, can make approximate transition model and derive a policy [Ormoneit & Sen 02].

Allows us to combine with MBIE-style approaches.
Learning in a Continuous Space

- **State space**: image location of button, size of button (3 dimensions), and color of button (discrete: red / green).
- **Actions**: Turn L / R, go forward, thrust head.
- **Reward**: Exit the box.
- **Shaping reward**: Pushing button.

When Sensations Not Enough

Robust to weak non-Markovianness.

But, won’t take actions to gain information.

Network repair example ([Littman, Ravi, Fenson, Howard 04]).

- Recover from corrupted network interface config.
- Minimize time to repair.
- **Info. gathering actions**: PluggedIn, PingIp, PingLhost, PingGateway, DnsLookup, …
- **Repair actions**: RenewLease, UseCachedIP, FixIP.

Additional information helps to make the right choice.
Learning Network Troubleshooting

Recovery from corrupted network interface configuration.

Java/Windows XP: Minimize time to repair.

Non-Stationary Environments

Problem: To predict future events in the face of abrupt changes in the environment.

Animal behavior: Match investment to income given multiple options

Observation (Gallistel et al.): Abrupt changes in payoff rates result in abrupt changes in investment rates. Proposed change-detection algorithm.
**Multiagent RL**

What is there to talk about?

- **Nothing**: It’ll just work itself out (other agents are a complex part of the environment).
- **A bit**: Without a boost, learning to work with other agents is just too hard.
- **A lot**: Must be treated directly because it is fundamentally different from other learning.

**Claim**: Multiagent problems addressed via specialized “shaping” rewards.

---

**Shaping Rewards**

We’re smart, but evolution doesn’t trust us to plan all that far ahead.

Evolution programs us to **want** things likely to bring about what we **need**:

- taste/nutrition
- pleasure/procreation
- eye contact/care
- generosity/cooperation
Why Have These Rewards?

Big advantages for (safe) cooperation.

For reciprocal altruism, a species needs:

• repeated interactions
• recognize conspecifics; discriminate against defectors
• incentive towards long-term over short-term gain

Necessary, but not sufficient: Must learn how.

Drives Linked with Altruism

For reciprocal altruism to work:

• want to be around others
• feel obligated to return favors
• feel obligated to punish a defector

Evidence that the reward centers of our brains urge precisely this behavior.
Does Rejection Hurt?

Psychological rejection processed by the same areas of the brain as physical pain.

(Is Eisenberger et al. 03)

Is Cooperation Pleasurable?

- fMRI during repeated Prisoner’s Dilemma
- Payoffs: $3 (tempt), $2 (coop), $1 (defect), $0 (sucker) (Rilling et al. 02)
- Mutual cooperation most common (rational)
- Also triggered brightest signals in those brain areas that respond to desserts, pictures of pretty faces, money, cocaine: “reward processing”
Motivation to Punish?

Ultimatum Game

Proposer is given $10.

Proposer offers $x \in X$ to Responder.

Responder can “take it or leave it”.

- Take it: Responder gets $x$, Proposer gets $10-x$
- Leave it: Both get nothing.

What Should Responder Do?

Fraction of acceptances of $x=2$

<table>
<thead>
<tr>
<th>$X$</th>
<th>one-shot</th>
<th>repeated</th>
<th>human</th>
</tr>
</thead>
<tbody>
<tr>
<td>{2,8}:</td>
<td>100%</td>
<td>34%</td>
<td>70%</td>
</tr>
<tr>
<td>{2,5}:</td>
<td>100%</td>
<td>0%</td>
<td>55%</td>
</tr>
<tr>
<td>{2,2}:</td>
<td>100%</td>
<td>100%</td>
<td>80%</td>
</tr>
<tr>
<td>{2,0}:</td>
<td>100%</td>
<td>100%</td>
<td>90%</td>
</tr>
</tbody>
</table>

Repeated game analysis (Littman & Stone 03)

Human results (Falk et al. 03)
Where We Went

Reinforcement learning: Lots of progress.

• Powerful concepts and algorithms available.
• Helping to understand cognition by creating it.
• Model-based approaches showing great promise
• Some fundamental new ideas needed
  • representation
  • reward
  • reasoning about change and partial observability

Large rewards yet to be found!

Next Time

• Finishing the book: Chapter 9