Towards Real-Life Reinforcement Learning

Michael L. Littman
Rutgers University
Department of Computer Science
Rutgers Laboratory for Real-Life Reinforcement Learning

Where We're Going

Introduce reinforcement learning
- why I think it's exciting

Define the problem and current approaches
- highlight challenges of RL with real data

Current projects in my lab:
- efficient exploration
- rich sensors
- partial observability
- non-stationary environments

What is RL?

Branch of machine learning concerned with sequential behavior:

- tries to remove human activities from the inner loop of the learning process.
- makes systems that improve a performance metric via interaction with their environment.

Some Relevant Applications

Adaptive filtering

- reward for delivering relevant doc
- punishment for delivering irrelevant doc
- learn from (sparse) human feedback

Efficient spam tagging (efficient sorting also)

- spam if fail any of a set of filters
- cost for computation time
- try to run cheap / likely to fail filters first
- non-spam always same cost, can tag spam quickly
- output always same, minimizing (measurable) cost

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

QuickTime™ and a YUV420 codec decompressor are needed to see this picture.

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)
- self motivated

QuickTime™ and a
H.263 decompressor
are needed to see this picture.

Crawl Before Walk

Impressive accomplishment:

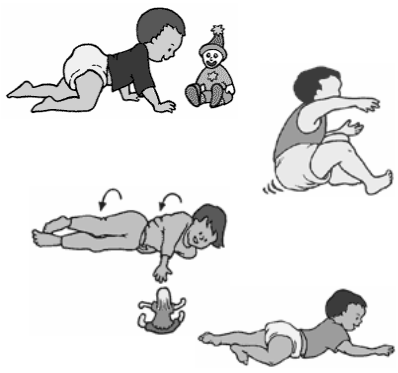
- Fastest reported walk/crawl on an Aibo
- Gait pattern optimized automatically

QuickTime™ and a
YUV420 codec decompressor
are needed to see this picture.

QuickTime™ and a
YUV420 codec decompressor
are needed to see this picture.

Human "Crawling"

QuickTime™ and a
DV/DVCPRO - NTSC decompressor
are needed to see this picture.



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:

- implement appropriate reward signals,
- develop algorithms that search the space of behaviors to maximize reward signals.

Find The Ball Task

Learn:

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

QuickTime™ and a
H.263 decompressor
are needed to see this picture.

Problem Formalization: MDP

Most popular formalization: Markov decision process

Assume:

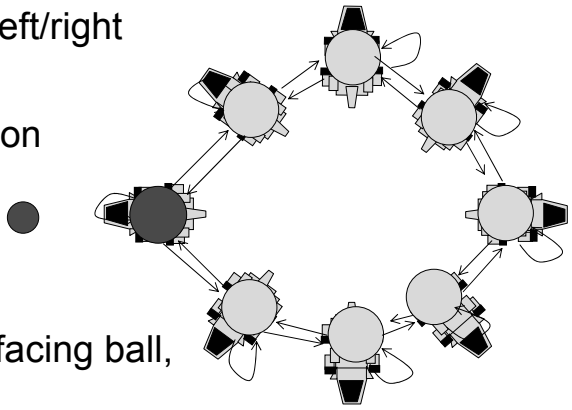
- States/sensations, actions discrete.
- Transitions, rewards stationary and Markov.
- Transition function: $\Pr(s'|s,a) = T(s,a,s')$.
- Reward function: $E[r|s,a] = R(s,a)$.

Then:

- Optimal policy $\pi^*(s) = \text{argmax}_a\ Q^*(s,a)$
- where $Q^*(s,a) = R(s,a) + \gamma\ \Sigma_{s'}\ T(s,a,s')\ \max_{a'}\ Q^*(s',a')$.

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
- $\Pr(s'|s,a) = T(s,a,s')$, $E[r|s,a] = R(s,a)$
- $\Sigma\alpha_t = \infty$, $\Sigma\alpha_t{}^2 < \infty$

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.
- Non-stationarity: details change over time.

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

Convergence of Policy

Logical disconnect:

- Q-learning converges in the limit to $Q^*(s,a)$.
- Best action is greedy: $\pi^*(s) = \text{argmax}_a\, Q^*(s,a)$.
- But, for convergence, can't starve (best) action.

Weak result (Singh, Jaakola, Littman, Szepesvári 00):

- GLIE (greedy in the limit with infinite exploration).
- *Policy* converges to optimal, not just Q values.
- Example: Decaying ε-greedy. Visit $n(s)$ to state s, choose random action with probability $1/n(s)$.

Efficient Exploration

Limit is nice, but would like something faster.

Goal: Policy that's ε optimal with prob. $1\text{-}\delta$ 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).

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.

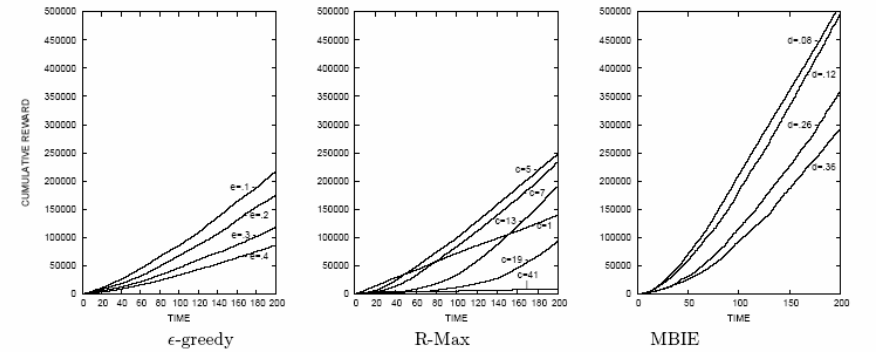
In bandit case, this works out to precisely IE.

Very general, but can be intractable.

Solvable for MDPs. (Strehl & Littman 04)

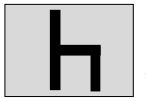
Some (non-real) MBIE Results

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



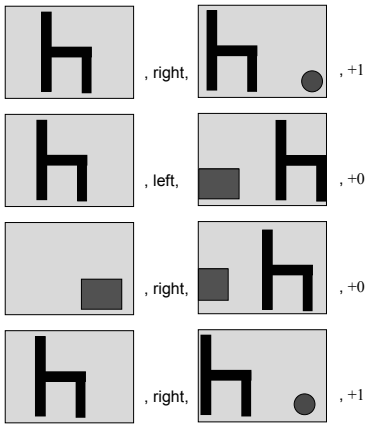
Rich Sensors

Can treat experience as state via instance-based view.

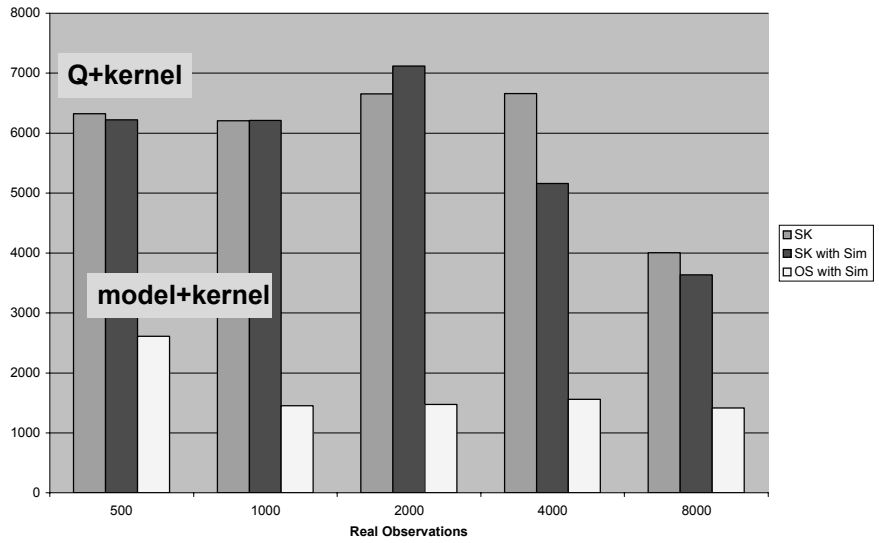

, right

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.



Mountain-Car Experiments



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**.

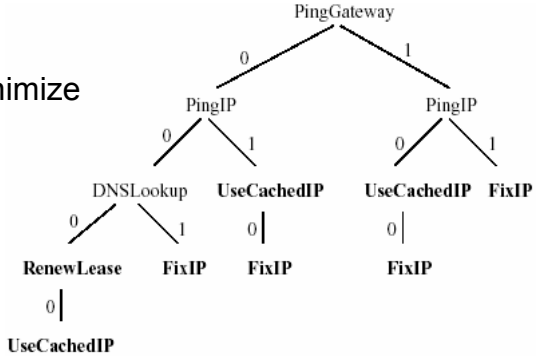
Additional information helps to make the right choice.

Learning Network Troubleshooting

Recovery from corrupted network interface configuration.

Java/Windows XP: Minimize time to repair.

After 95 failure episodes

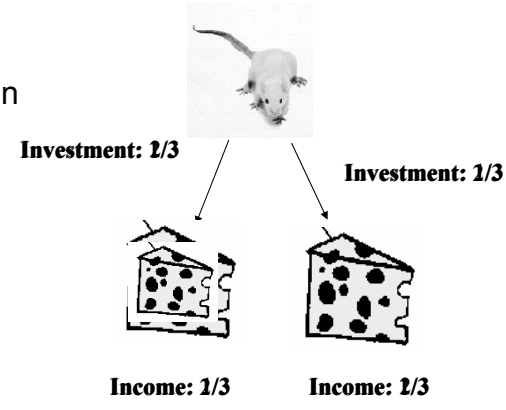


Non-Stationary Environments

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

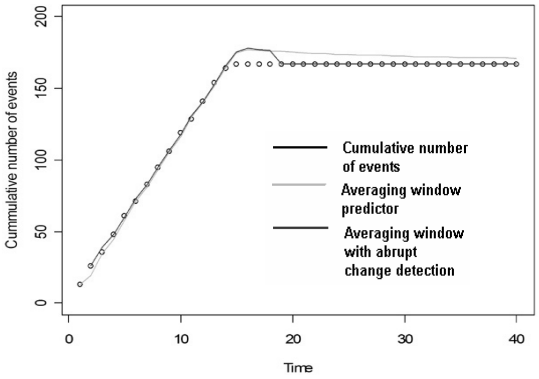
Animal behavior: Match investment to income given multiple options

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



Recognizing Changes in Disk Access

Portable computers use techniques such as disk spin-down/up to conserve energy. Given the history of disk accesses of the user, predicting how long it will be until the next disk access occurs



- Under real usage conditions, abrupt changes between usage modes.
- Detecting abrupt changes to mode can save energy

New Project Highlights

- Main technical idea: ***Use intrinsic motivation*** (computational curiosity) to discover knowledge about the world.
- Demonstration: ***Implement*** our algorithm on Sony Aibo robots exploring a rich playspace.
- Expected Outcome: The robot learns knowledge that is both abstract and grounded, enabling it to be ***instructed*** to ***perform*** high-level tasks in the physical world.

Playroom

Where We Went

Reinforcement learning: Lots of progress.

Let's reconnect learning with real data:

- previous ideas contribute significantly
- 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!