Reinforcement Learning:

The Good, the Bad, and the Ugly

Leslie Pack Kaelbling

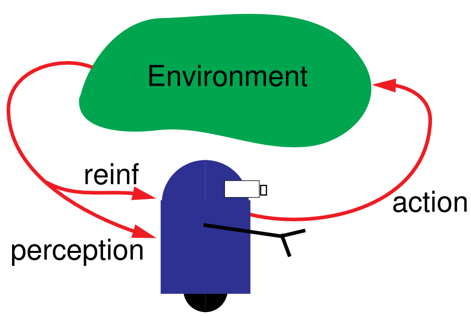
Computer Science Department
Brown University

Plan

- What is reinforcement learning? How does it fit into our understanding of other kinds of machine learning?
- The state of the art in reinforcement learning: the good, the bad, and the ugly
- What role should reinforcement learning play in the operation of general-purpose intelligent agents?

WARNING:
rampant speculation ahead

What is Reinforcement Learning?



A class of problems, not a particular technique.

RL and Other Learning Problems

supervised learning:

- output should be 3.4!

reinforcement learning:

- you're doing pretty well!

unsupervised learning (clustering):

- what do these things have in common?

less guidance from environment

Reinforcement Learning and Supervised Learning

Does reinforcement learning ever reduce to supervised learning? I once said “yes,” when

- there are two actions
- reward is boolean
- there is no noise
- reward is immediate

I was wrong!

- actions affect what situations will occur
- agent has to explore

Two Kinds of Learning

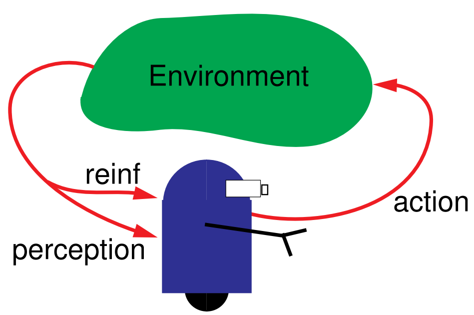
Dietterich applied Newell’s ideas of knowledge level and symbol level to machine learning:

- symbol-level learning: the system already has all of the knowledge, but not in a usable form; learning reformulates the knowledge, typically resulting in faster and more effective performance
- knowledge-level learning: the system gains knowledge from the incoming data that it did not already have

So what’s reinforcement learning?

Knowledge-level Learning

This is our default view of reinforcement learning:



RL for Symbol-Level Learning

When

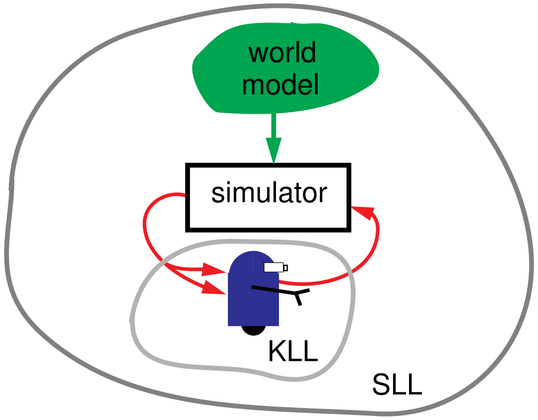
- a model of the environment is known
- theoretically, we could compute the optimal policy using dynamic programming
- practically, it’s hopelessly computationally intractable

Simplify the problem by focusing on the commonly occurring cases:

model $\xrightarrow{\text{simulation}}$ experience $\xrightarrow{\text{RL}}$ policy

TD-Gammon works (kind of) like this.

KLL within SLL



Symbol-Level Learning: Search Control

Zhang & Dietterich solve complex scheduling problems for the space shuttle:

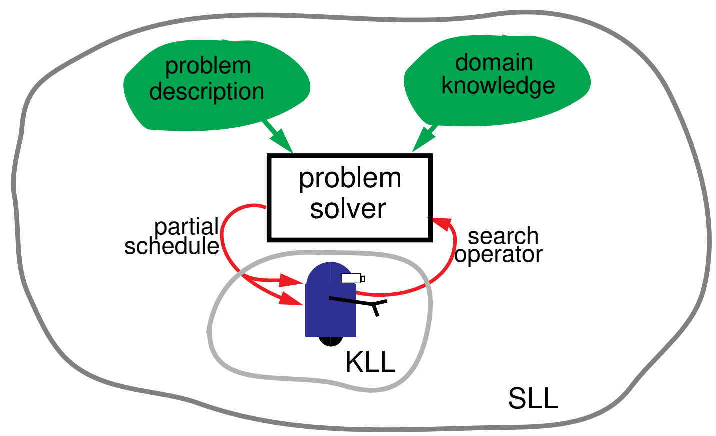
- the system knows all the information needed to derive an optimal schedule
- the solution process is a search in the space of partial schedules
- at each point in the search, there is a set of possible operators; initially, they're chosen according to some fixed heuristics

Learning Search Control

Use RL to learn to choose appropriate search operators based on general features of the current problem-solving situation:

- reward for utility of the final schedule
- penalties for each search step
- state is the current scheduling situation
- if features are well-chosen, between-task transfer results

KLL within SLL for Search Control



Dyna: KLL within SLL within KLL

Sutton's Dyna system

- uses real-world experience to learn a model (KLL)
- derives a policy from the model (SLL)
- uses simulated experience and RL to derive the model (KLL)

data → model → experience → policy

Solving RL Problems

When people talk about RL methods (as opposed to RL problems), they usually refer to methods based on dynamic programming:

- TD, Q
- Dyna, Pri-Sweep

But there are other options:

- gradient descent, recursive least-squares (still based on Bellman error)
- search in policy space: genetic algorithms, other kinds of stochastic search

Maybe there are other useful options.

State of the Art

Good

Temporal credit assignment:

- temporal-difference algorithms are well understood
- useful formal convergence results
- fairly good behavior in practice
- working on making these methods more efficient
  - traces
  - Dyna
  - prioritized sweeping

Ugly

Exploration:

- not formally well-understood
- ad-hoc solutions seem to work reasonably well

Generalization:

- starting to have formal understanding (Baird, Gordon, etc.)
- some good empirical results on input spaces
- very limited work on generalizing actions
- decomposed representations (NNs, Bayes nets)

Bad

Hidden state:

- inescapable for most real problems
- people usually ignore it
- some ideas, none too convincing: recurrent NNs, stochastic policies, belief-state models

Hierarchy:

- most people think this is important
- works okay when levels are hand-defined and trained (e.g., Lin)
- better if it were more flexible

More Bad

Bias: we should all know that it's impossible to learn anything interesting without "knowing" something

- reflexes help agent see interesting part of the state space
- problem decompositions restrict solutions and speed learning
- intermediate reinforcement signals make temporal credit assignment easier
- teaching may be important

Where is the bias going to come from? Human programmers or evolution. I'll bet on the programmers.

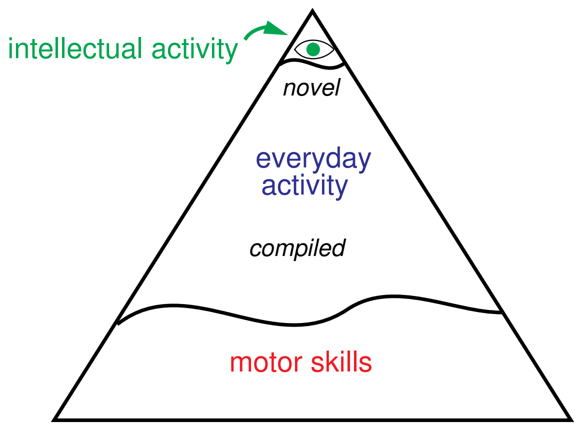
Terrible

No idea how reinforcement learning fits into the overall design of an intelligent agent.

So I'll speculate.....

WARNING:
virtually unfounded speculation ahead

The Pyramid of Intelligence



Motor Behaviors

Examples: walking, driving, hand-eye coordination

- use model-free RL methods
- reinforcement for doing these behaviors must be innate
- reflexes keep the agent out of (too much) trouble while learning

Everyday Activity

Examples: making photocopies, making breakfast, playing Tetris, taking the subway, editing slides

Novel activity:

- learn models
- solve circumscribed problems given dynamic goals
- important question: how to formulate problems?

Compiled activity:

- cache solutions from novel activity
- improve them by learning from experience
- learn when to activate them

Intellectual Activity

Hardly any exists!

- learn logical models?
- perform inference?
- transfer into everyday activity?

Glue

We need lots of glue to make this all work:

- formulating subgoals
- mediating conflicting outputs or strategies
- dealing robustly with breakdowns
- lots (and lots!) of *a priori* bias

Conclusions

Reinforcement learning is an exciting field!

- we've made significant progress
- there's a lot more work to be done
- RL has connections to many other problems in AI
- it's fun to think about building whole agents