Genetic Algorithms and Genetic Programming
Lecture 16

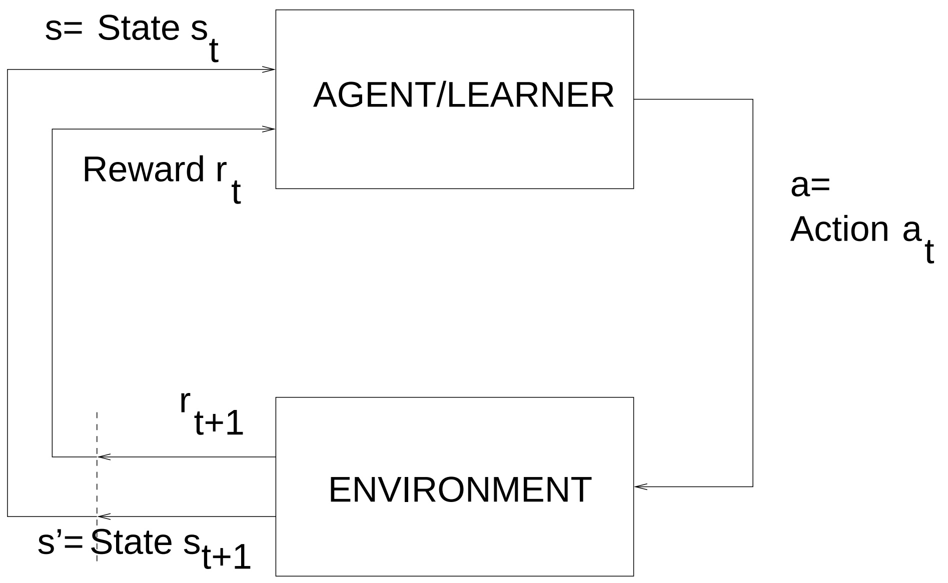
Gillian Hayes

24th November 2006

Interactions Between Learning and Evolution

- Reinforcement Learning
- Choosing the design: transition probabilities and reward function
- Simple agent that evolves its reward function
- Ackley and Littman's AL world
- Baldwin Effect

Reinforcement Learning 1



- The agent has to learn a **policy** π – a mapping from states to actions – what to do in a particular situation: $\pi(s, a) \rightarrow \text{probability } [0, 1]$
- It is given a **reward function**: $R^a_{ss'} \rightarrow$ reward (scalar)

Reinforcement Learning 2

- Use trial and error to choose actions, see what the reward is, adjust the policy so as to maximise your **expected future reward**
- **Learning rule** says how to adjust policy, e.g Q-learning learns the Q-value of state s and action a:
- $Q_{t+1}(s,a) = Q_t(s,a) + \alpha(r_{t+1} + \gamma \max_{a'} Q_t(s',a') - Q_t(s,a))$
- Policy is to choose action with the highest Q and take a random action sometimes ($0 \leq \gamma \leq 1$ discount parameter, $0 \leq \alpha \leq 1$ learning rate).
- Q is a measure of how much reward you can expect to get if you choose that action and continue to follow the policy. High Q – good action – good policy.

The Reward and Transition Functions

- The **reward** r_{t+1} is generated by the **Reward Function** $R^a_{ss'}$. If you choose action a in state s and end up in state s' you'll get reward r – this is part of the problem **design**.

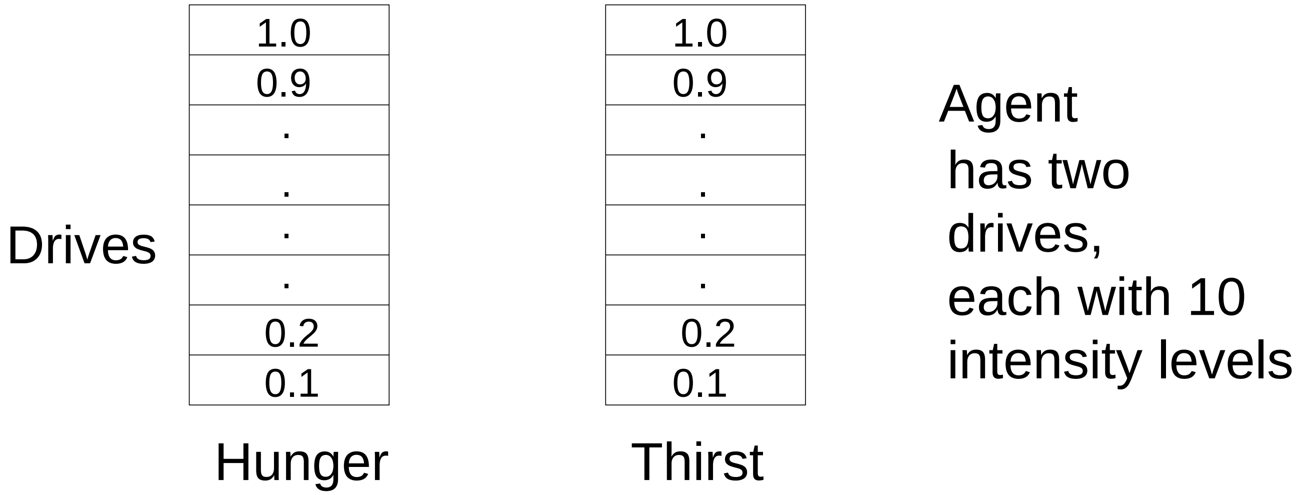
- Also part of the problem design is the **transition function** $P^a_{ss'}$, which gives the probability of transitioning from state s to state s' given that you choose action a.

How should we choose these functions? Can we evolve the reward function?

Choosing $P^a_{ss'}$ and $R^a_{ss'}$

- Make $P^a_{ss'}$ a "natural" consequence of environment, e.g.
  - Agent hungry s
  - Eats food a
  - Less hungry s'
  - With some probability $P^a_{ss'}$
- Actions have real consequences for the agent – e.g. if health drops too low it dies. Need to make it choose food when it's hungry. What $R^a_{ss'}$ will do this?
- Why should one action *a priori* be preferred over another? (Why do you think/feel/know eating food is good for you? Why do you think/feel/know that bashing your thumb with a hammer is bad for you? **Valency**)
- Those agents that prefer the actions with the right consequences survive. Agents whose reward functions allow them to learn good things will survive. **Evolve** the reward function.

Simplest Agent



Actions “+0.1, 0, -0.1”, e.g. eat/drink, do nothing, move

Actions affect directly the drive level of 1 or more drives: increase, decrease, leave same. Perhaps add some noise so action +0.1 in state 0.2 leads to state 0.4 or 0.2 (and not always 0.3).

Evolve Reward Function

Q(s,a) (1 Drive only e.g. food level –– hunger)

Action \ State	0.1	0.2	....
+0.1	0.5	0.55	...
0.0	0.1	0.15	...
−0.1	−0.3	−0.25	...

values at some point during learning

Q−table: Learning updates this table

R(s, s')

start state s \ end state s'	s1	s2	s3	......
s1	6	9	10	
s2	3	4	10	
s3	2	3	5	9
.				
.				
.				

Start in s1, end up in s2, reward is 9.

etc.

Reward function table: Evolve this

Reward function $R^a_{ss'}$ should be a 3D table. But actions are deterministic so condense to 2D table.

Evolve the reward function. Learn the Q table for that reward function. Follow a policy based on that learnt Q.

Task

Evolve reward table such that when:

Use reward table to

Learn Q values and hence

Learn a policy

Follow that policy

The policy is good – agent survives a long time

Fitness function

scores behaviour of this policy

– run the agent with policy that uses the learnt Q-values and see whether it reaches and attains maximum satiation of drive(s) (very time-consuming)

Representation

Chromosome is a 1D (string) representation of the reward table

Single-point crossover + 1% mutation

No crossover + 8% mutation

Inversion 1% probability

Algorithm

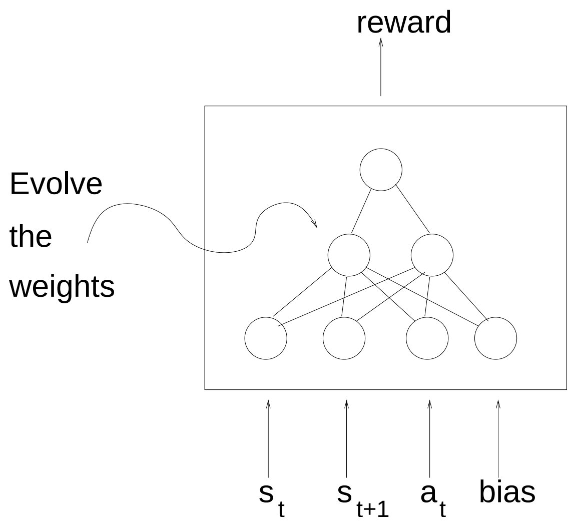
```
Initialise reward table (chromosomes) for N agents
Evaluate each agent:
    Do Q-learning with its reward table to learn Q-values
    When table stable, run agent on evaluation task,
    using policy based on the learnt Q-values
Select, crossover, mutate, form next generation
Stop when acceptable performance or max generations passed
```

This works for simple situations: the agent evolves a reward table that allows it to learn how to keep its food levels topped up to a maximum

So it associates the correct valency with eating – eating allows it to survive (and get a high fitness)

Other Possibilities

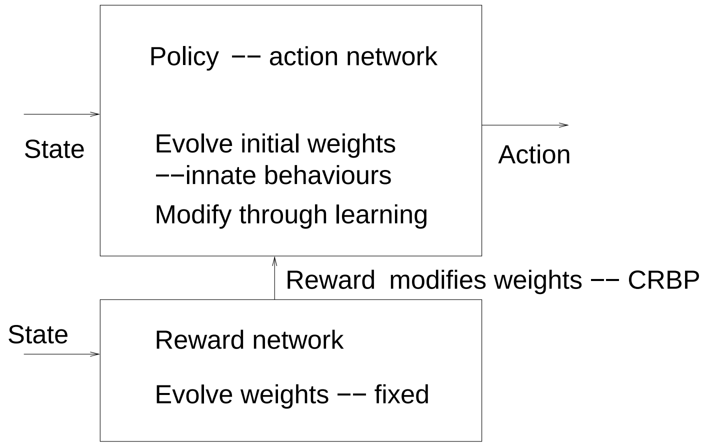
- Could also evolve Q-table directly (rather than going through the indirect route of learning $R^a_{ss'}$ and learning Q on top of that)
- Reward table becomes rather large rather quickly so use a neural net....



See Theodoros Damoulas: Evolving a sense of valency. MSc thesis, Informatics 2004.

Context

Ackley and Littman: can an agent learn given only natural selection as feedback?



CRBP = complementary reinforcement back propagation (a learning method)

Agents run in a landscape with other agents, food, hazards, etc. They can reproduce if (i) they can accumulate enough energy and (ii) there is another agent nearby.

Agents with evolution and learning do better than agents with just evolution: it is easier to evolve a good evaluation function (i.e. reinforcement function) than a good action function (i.e. a policy) (see Ackley and Littman (1991)).

Baldwin Effect 1

Learning can explain [some] evolutionary phenomena (Wikipedia)

• A&L decoded the genome to identify the genes for behaviours. Initially successful behaviours are in the reward function network (so agent can learn them) but eventually show up as innate behaviours in action network (so agent can do them from birth – hard-wired).

e.g. Plants $\Rightarrow$ high reward $\Rightarrow$ choose approach action (learn to do this)

vs.

See plant $\Rightarrow$ Choose approach action (as a reflex behaviour with no evaluation of valency)

• Is this **Lamarckian evolution?**: changes during lifetime transmitted genetically?

Agents initially **learnt** that plants were good, now their offspring "know" this innately!

Baldwin Effect 2

No, not Lamarckian Evolution.

Explain as:

• Darwinian evolution + **Baldwin Effect**

• Those agents who know plants are good (because their reward network assigns high reward to them) will learn to approach them.

They will preferentially survive.

• This provides selection pressure for them to develop action networks with the same behaviour – then they don't have to "waste time" learning this link (plants–good–approach). They can do it from birth and so will survive at the expense of those that are still learning.

Conclude

If learning expensive, tend to acquire innate characteristics.

If everything stays the same, tend to acquire instinctive behaviours and no advantage to being able to learn.

So evolutionary methods do well in stable environments, but need learning (as well) if environment is dynamic.