Reward Bonuses for Efficient, Effective Exploration

(or, KWIK Learners at Play)

Michael Littman

Rutgers University

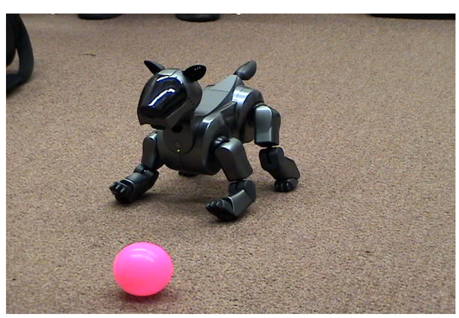
Computer Science

Rutgers University Center for Cognitive Science

Rutgers Laboratory for Real-Life Reinforcement Learning

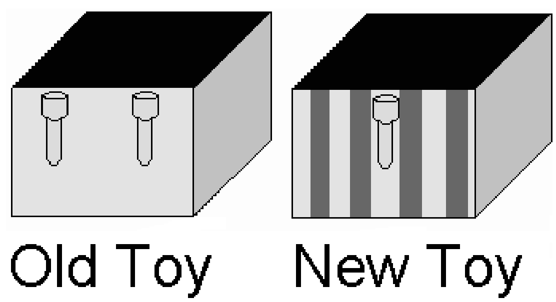
Background (RL^3)

- Creating algorithms that learn to behave.
- Model a learner, design an environment. (Backwards from psychology?)
- Interested in "in principle" learnability.
  - computational and experience complexity



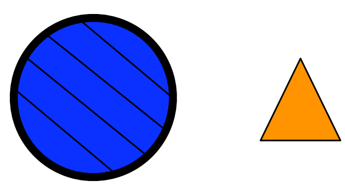
Motivational Data

- Statistics of play sensitive to confounding
- Show kid 2-lever toy (Schulz/Bonawitz 07).
  - Demonstrate both. Kid becomes interested in new toy.
  - Demonstrate them together. Kids stays interested in old toy.
- Experiment design intractable. How can play be computed?



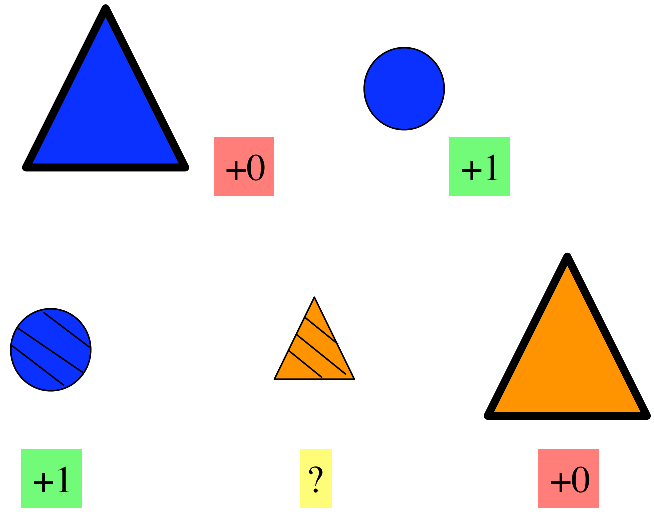
Motivating Example

- Let's imagine the world consists of shapes.
- Four attributes:
  - striped / solid
  - big / little
  - blue / orange
  - circle / triangle



- Each shape is either rewarding +1 or not +0 .
- Critical assumption: *Only one attribute matters.*
- Each round: Select shape from a collection, get the associated reward.

Let's Play



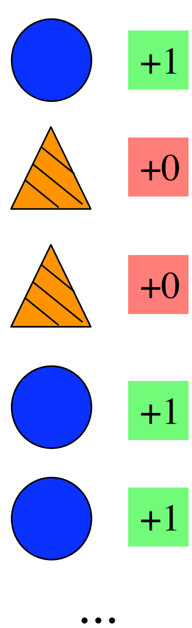
Which to Choose?

- +1 vs. +0 → +1
- +1 vs. ? → +1
- +0 vs. ? → ?
- Like assigning a value of 0.5 to ? and always choosing the highest scoring shape.
- Results in maximum total reward.
- Critical for learner to *know when it knows* the value.
- Contrast with classical machine learning…

PAC Learning

- Classic machine-learning paradigm is PAC (probably approximately correct).
- A learner is efficient in the PAC model if it can make accurate predictions after seeing a small set of labeled examples. Critical assumption: Examples are drawn iid.
- Commits to a single hypothesis based on the statistics of the training examples.
- A vicious adversary can make a PAC learner get very low reward.

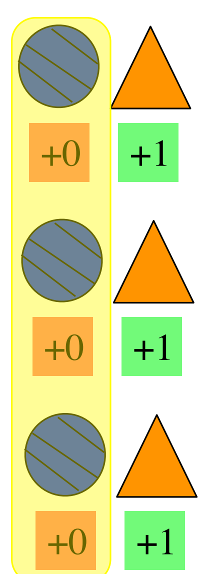
Nasty Example for PAC



PAC learner now picks a hypothesis

Could be:

- blue = +1
- circle = +1
- solid = +1



KWIK Learning

- We devised a different model, which we call KWIK learning (Knows What It Knows).
- A KWIK learner cannot make miscategorization errors, but it can choose not to label some examples.
- However, there is a bound on the number of times the learner can opt out. So, it needs to glean something substantial from each example.
- No training/testing distinction.

Decisions Can Be Hard

- Using a KWIK algorithm and preferences of +1 < ? < +0 leads to optimal decisions.
- Things are not always so simple.
- Let's say each round of the game has an associated reward multiplier.
- Now, it can be better to prefer a ? to a +1 .
- <u>Example</u>: Next trial has a huge multiplier.
- Can formulate optimal behavior in a Bayesian framework–computationally intractable.

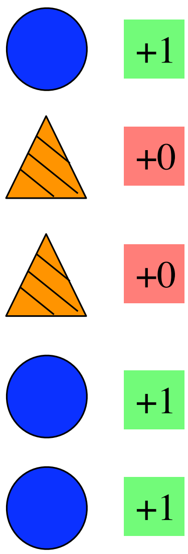
Experience-Efficient Learning

- Decision maker interacts with the world.
- We call an action a "mistake" if it is not a step of a nearly optimal (ϵ-optimal) behavior.
- With high probability ($1-\delta$), the number of mistakes should be small compared to the complexity of the environment.
- Such an approach is "experience efficient".

Implications and Solution

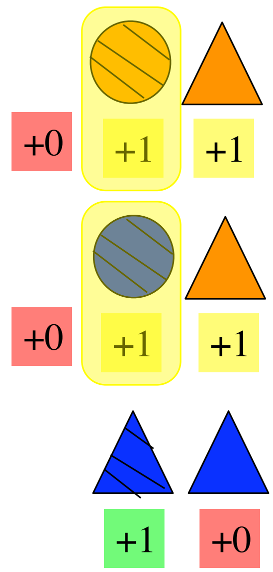
- Easier than if requiring optimal behavior.
- But, still challenging.
- Must balance exploration/exploitation.
- One solution: KWIK learn while behaving assuming ? ≤ +1 < +0 .
  - Adds a reward bonus for exploring.
  - Version of R_{MAX} (Brafman & Tenneholtz)
- If we can KWIK learn it, we can use this information to drive behavior.

RMAX on the Shape Task



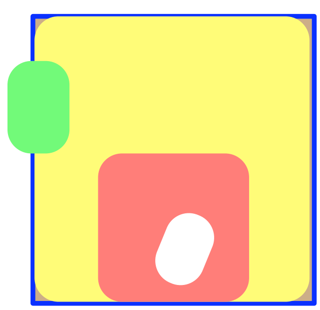
…

KWIK learner doesn't pick a hypothesis, changing over time.



RMAX Makes a KWIK Escape

Task: Exit room using bird's-eye state representation.



schematic

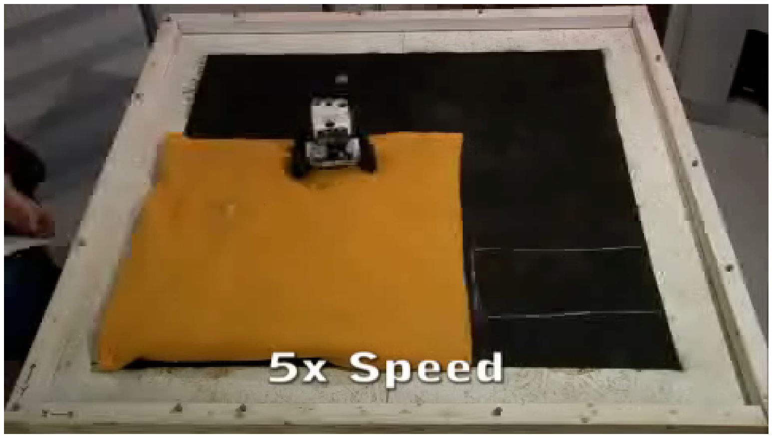
(Nouri)

Details: Discretized 15x15 grid x 18 orientation (4050 states);
6 actions: forward, backward, turn L ,turn R, slide L, slide R.

Prefer unknown to "known bad".

Learn Surface Properties

- Learns the effect of its action on sand and wood. Uses the resulting model to plan shortest path. (Leffler, Edmunds, Littman)



RMAX Observations

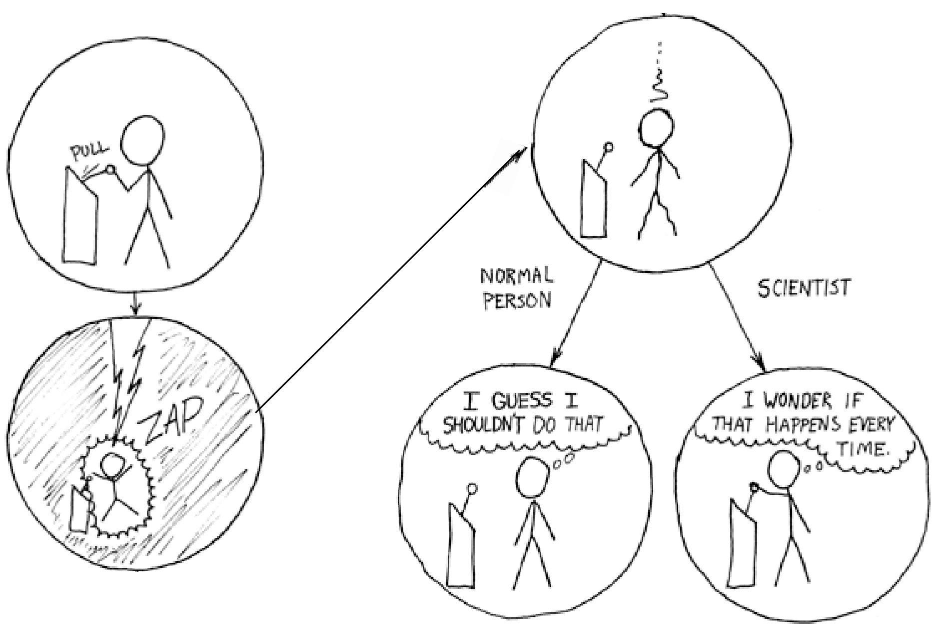
- Actively balance exploration/exploitation.
  - Provable near-optimality, bounded exploration.
- Don't have to make explicit experiments!
  - Difficult, and unnecessary for this objective.
- Works in diverse decision-making algorithms
  - uncertainty put on a value scale
  - decisions driven off of value judgments
- Doesn't "consciously" know if an action is for information or reward gathering.

Final Thoughts

Many hypothesis classes KWIK learnable:

- coin flip probability
- Dynamic Bayes net probabilities given graph
- degree *k* Dynamic Bayes net
- *k* Meteorologist problem
- *k*-CNF
- *k*-depth decision tree
- unions of KWIK-learnable classes

Do People Explore? (xkcd)



Wrap Up

- Goal: Algorithms that learn to behave.
- To provide learning algorithms with guarantees of near-optimality, devised a new learning setting, KWIK.
- The learner can influence exploration/exploitation in a decision maker by adding artificial rewards to unknown states.

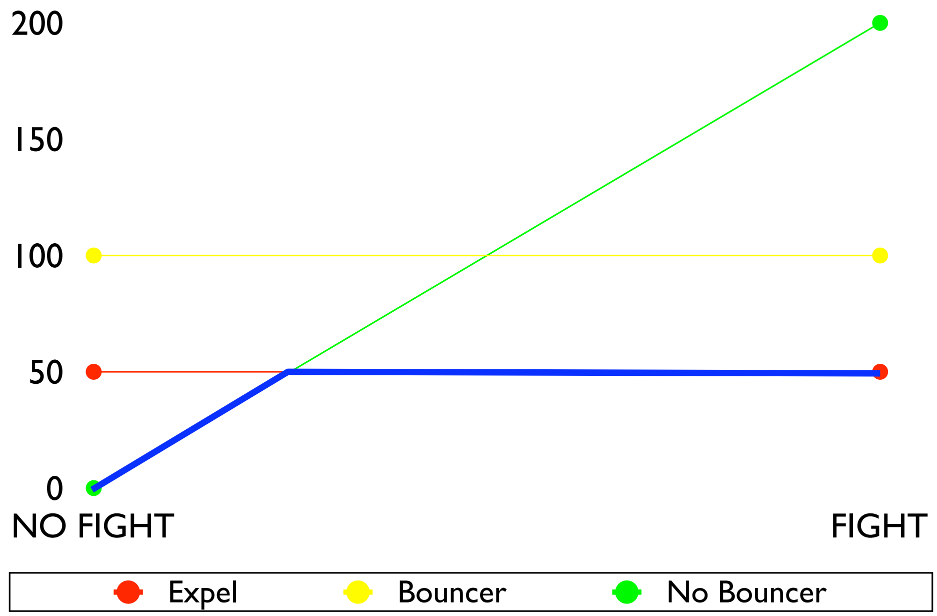
Extended Fanciful Example

We own a bar. There's a collection of n=5 regular customers. One is belligerent (not sure who). One is a peacemaker (not sure who). Each night, we see who comes to the bar. A fight breaks out if the belligerent one is there and the peacemaker is not.

After we see who has arrived, we can:

- Pay $100 for a <u>bouncer</u> who will stop a fight as soon as it breaks out.
- Pay $200 to repair the bar if a fight breaks out and <u>no bouncer</u> was hired.
- Pay $50 in opportunity cost and <u>expel</u> the whole group before they even enter the bar.

Costs of Choices By Outcome



Example Interaction

- [0,2,3]
  - no fight
- [1,3]
  - no fight
- [3]
  - *no fight*
- [0,3,4]
  - fight
- [2,3]
  - *no fight*
- [1,2,4]
  - *no fight*
- [0,4]
  - *fight*
- [0,2,3,4]
  - no fight
- [1,3,4]
  - no fight
- [0,2,4]
  - *no fight*
- [0,1]
  - *fight*
- [1,2,3,4]
  - *no fight*
- [1,4]
  - *no fight*

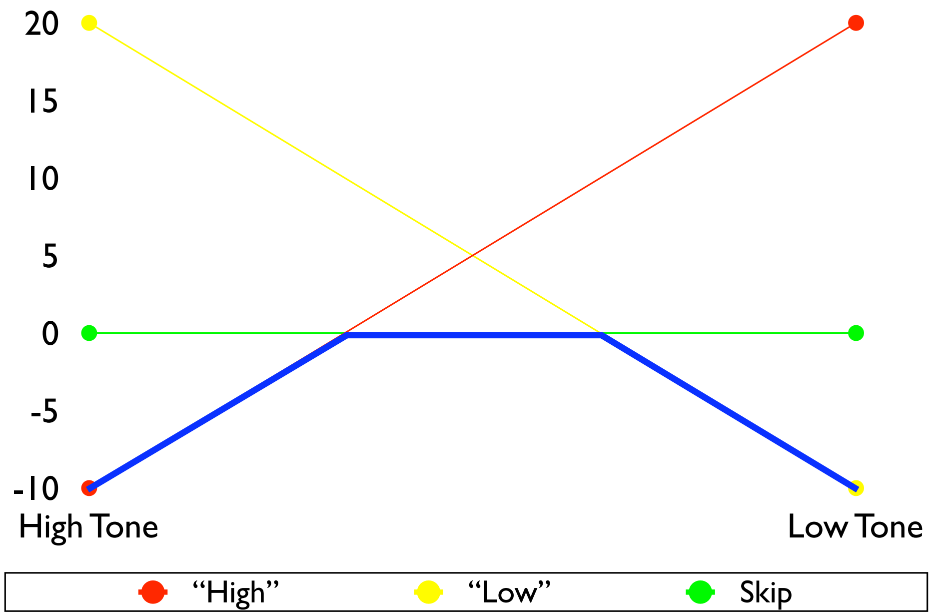
KWIK Bound

- Will pay bouncer ("don't know") no more than $n(n\text{-}1)$ times. Might overpay by $100.
- Every other trial is optimal!
- Same overpay cost as active learning, in spite of the lack of control over the examples.
- Better than mistake bound or PAC approaches.

Some KWIKer Than Others

- Given a classification task with a high cost of being wrong, can a subject learn to choose an "I don't know" response to move on to a new question?
  - Porpoises, spider monkeys, people can do it.
  - Rats* and pigeons haven't.
- Not really the same… "I don't know" is actually the right answer for some inputs. (Not part of learning, per se.)

Costs of Choices By Outcome



Building Block for Learning

- One scenario of general interest follows.
- Imagine learner can perceive a set of candidate causes.
- Imagine it knows that exactly one is responsible for the output.
- Concretely,
  - *n*-bit input, one bit (unknown) controls output
  - one output distribution if bit is on, another if off
  - Learn complex structure by same idea: one hypothesis controls output...

Why So Hard?

- Given the data, you can estimate the probability the output is 1 given any of the input bits. Even the wrong ones.
- Which should we believe?
- One with lowest prediction error.
- If two bits are correlated, prediction error will be similar... Need to notice when they

Exploring Approximations to Models of Exploration

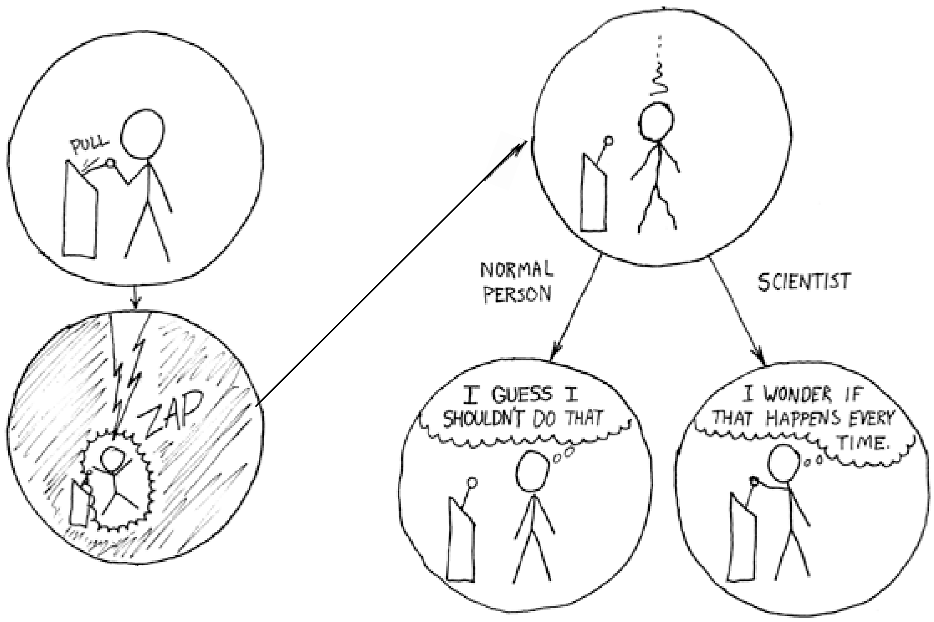
apologies to Adam Sanborn

Michael Littman

Rutgers University

Computer Science
Rutgers University Center for Cognitive Science
Rutgers Laboratory for Real-Life Reinforcement Learning

Do People Explore? (xkcd)



Importance of Exploration

- Exploring is risking low-outcome decisions to obtain high-outcome decisions.
  - Necessary for finding optimal decisions.
  - Probably needed for finding satisficing decisions.
- Perhaps a higher order decision.
- Can interact with representation issues.
- What could happen? What will happen?

Demo

- win the mystery game...

Comparison

- Taxi problem (Dietterich)
- Taxi: How long until optimal behavior?

Exploration style	Algorithm	# of steps
epsilon greedy	Q-learning	47157
count on states	Flat Rmax	4151
count on features	Factored Rmax	1839
count on interaction	Objects	143
whatever people do	People	50

Views of Exploration

- Ad hoc: Trying something crazy now & then.
- Bayesian: Act optimally given current uncertainty and future information.
- PAC-MDP: Act near optimally (w.h.p.) on all but a small number of steps.
- Regret: Converge to optimal, total loss grows slowly.

None quite right.

Demo #2

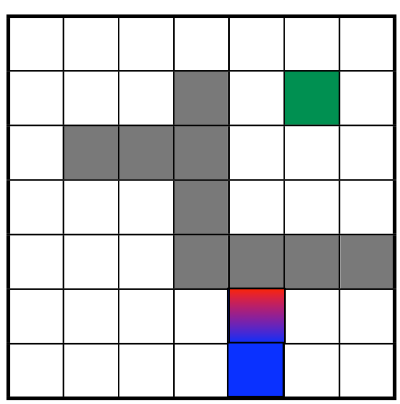
- What do you learn?

PAC-MDP

- Can exploit the structure to learn fast!
  - Knows nothing matters: Rmax.
  - Know some feature matters: RAM-Rmax.
  - Knows which feature matters: RAM-Rmax.
  - Know the wall: No learning needed.
- KWIK learning underlies fast learners.
- *Enthymematic*!
  - Where do these assertions come from?
  - What if they hold only partially?

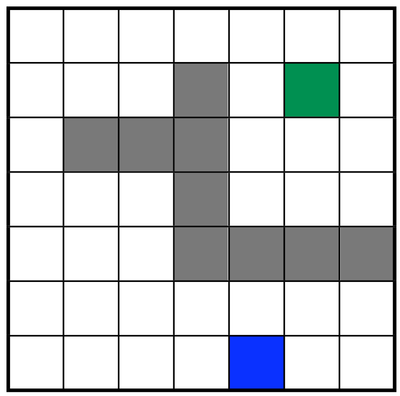
It's Not Easy Getting Creamed

- Standard PAC-MDP algorithms can't say:
  - I know you told me all states independent,
  - but every wall I've seen has been painful.
  - Can I just walk around now, please?



PAC-MDP with Bayesian Priors

- With a prior that all similar colored squares are the same, we can bound the chance generalization will lead to sub-optimality.
- Idea: Don't worry about it if it's small!



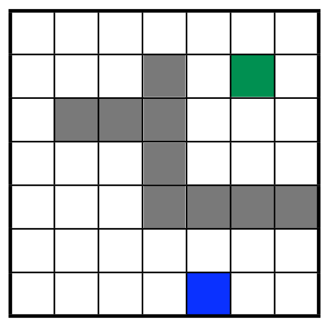
BOSS: Algorithmic Approach

- Maintain a posterior.
- Sample models from the posterior.
- Solve each one.
- Assume the best of sampled set is right.
- Act accordingly until something surprising.

- If set big, guarantee near optimality (whp).

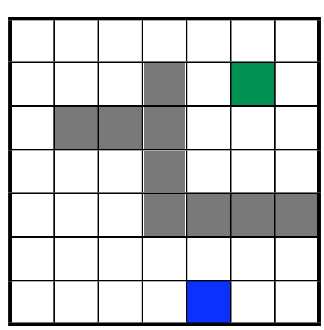
Priors Change Learning Alg

- Gray always wall: No learning
- Each gray independent: Rmax
- Grays always like each other: RAM-Rmax
- Sometimes independent/not: New alg.



Learn Prior: Learn to learn

- These priors themselves can be learned.
- Techniques like those discussed earlier in a "transfer" setting seeing related problems.



Conclusion

- Humans don't solve NP hard problems.
- Observations of impressive behavior imply we've framed the problem wrong.
- Happily "stealing" from cognitive science to create better learning algorithms.