

# **Inductive Agent Modeling in Games**

**Bobby D. Bryant**

Neuroevolution and Behavior Laboratory  
Department of Computer Science and Engineering  
University of Nevada, Reno

# Today's Plan

- Introduction to Inductive Agent Modeling
  - concepts and issues
- In-depth example

# ***Why model agents?***

- Competitive reasons
- Automated content creation

# How complicated is an agent model?

- from the very simple
  - e.g., drama manager sets parameter for whether the player wants to see cat-fights
- to the very complex
  - e.g., controller emulates the Red Baron's use of his airplane and guns in a dog-fight

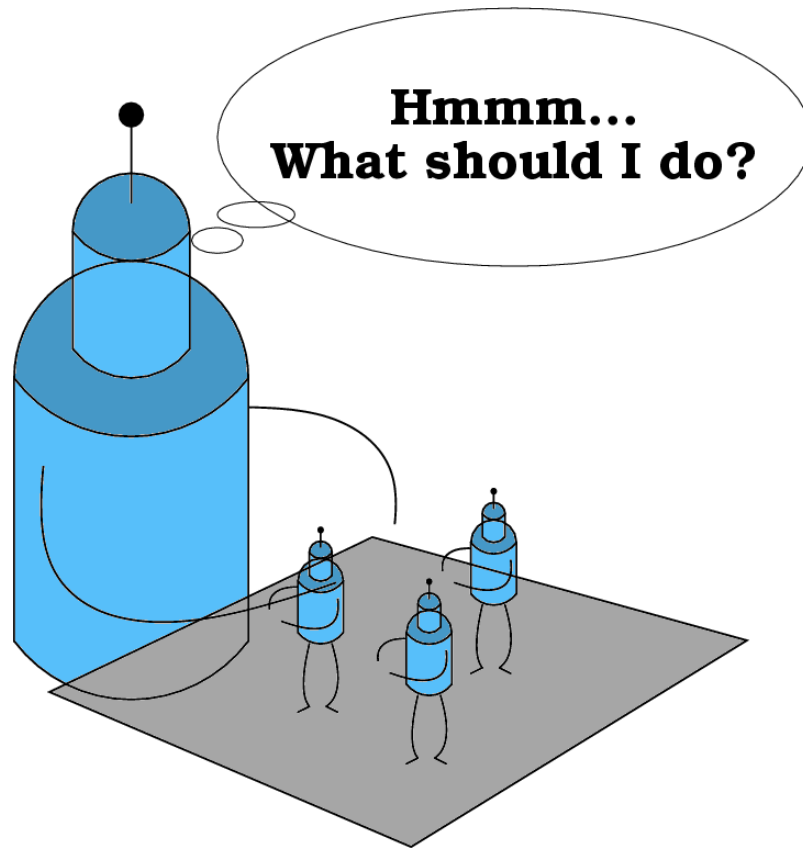
# Relationship of Modeler to Modeled

- Foe (*AKA opponent modeling*)
- Ally (e.g., human teammate)
- Neutral (e.g., NPCs for richer game environment)

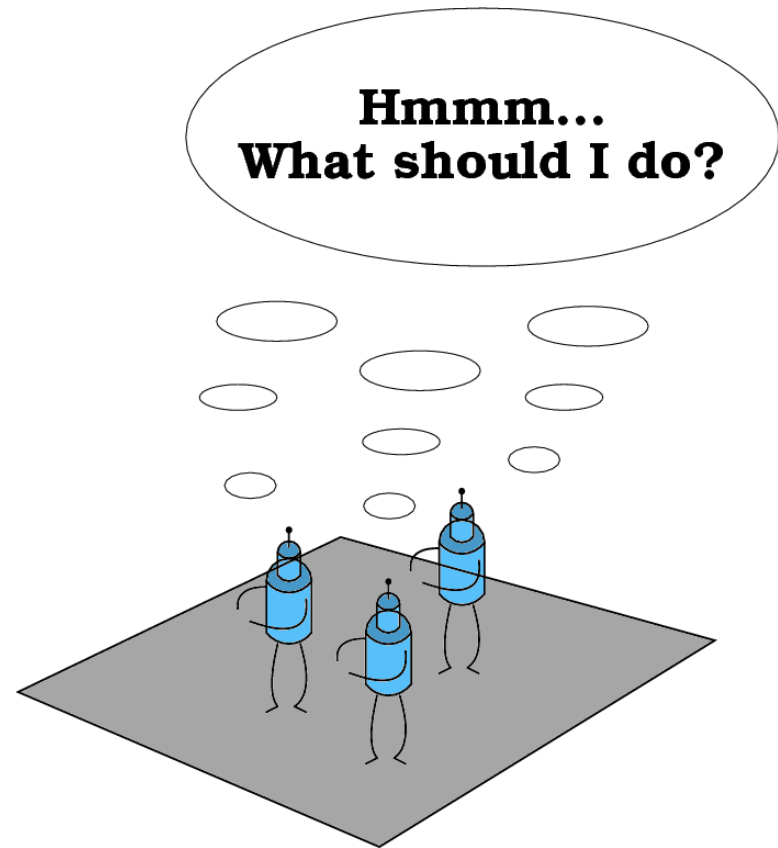
# Who/what do we model?

- Human player
- Game AI
  - Virtual player (e.g., *Ms. Pacman*)
  - Autonomous in-game agent (e.g., NPC in FRPG)

# Autonomous In-Game Agents



Virtual Player



Embedded Agents

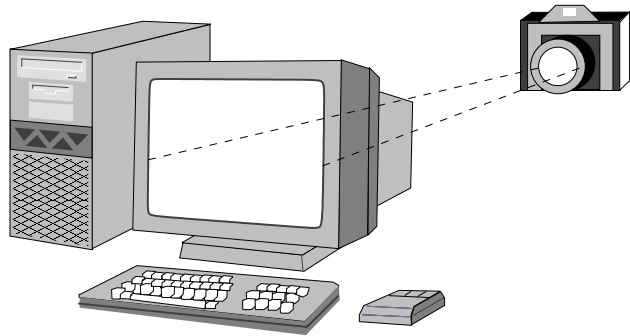
# What if You Want to Model a *Specific Kind* of Behavior?

- E.g., **The Red Baron** (vs. “a fighter pilot”)
- Behavior may not be optimal (in the normal sense)
  - this notion 'challenges' many researchers
- Hand-code the behavior?
- Use RL with a complex/subtle reward function?
- Derive policy from examples?
  - bypasses traditional *knowledge engineering* methods
    - allows indirect, *intuitive* expression of behavior
  - uses subject-matter experts, not technical experts



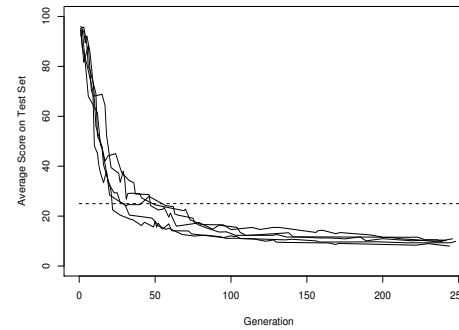
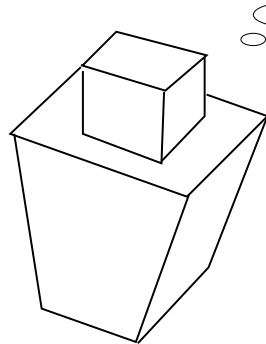
# Now here's the plan...

Human Plays Games...



...While Machine Learning System  
Captures Example Decisions...

*Foolish Human,  
Prepare to Die!*



...And Uses Them to Train  
Game-Playing Agents

# The Inductive Agent Modeling Challenge

- Capture examples
- Create an agent model
  - Conceive as a function:
    - for reactive control -  
$$m_a : observablestate \rightarrow action$$
    - more generally -  
$$m_a : observablestate \times context \rightarrow action$$
  - Derive by inductive machine learning mechanism

# Terminology

- *Exemplar* - the agent creating the examples
- *Observer* - the agency for capturing the examples
- *Learner* - the model that will emulate the exemplar

## On-line vs. Off-line

- On-line: the learner and observer may be the same
- Off-line: the learner and observer are (probably) distinct

# Challenge: Induction

- Deriving a generality from a collection of instances
- A hard problem...
  - known to be a logical problem since Hume (18th C.)
  - no-free-lunch theorems
- But we do it all the time anyway...
  - Quine: *Creatures inveterately wrong in their inductions have a pathetic but praise-worthy tendency to die before reproducing their kind.*
  - works because the universe isn't a random place(?)

# Induction in Machine Learning (i)

## Learning Classifier Systems (LCS)

- Map feature sets onto discrete classifications

$$c : \langle f_1, f_2, f_3, \dots, f_n \rangle \rightarrow class$$

- Learn general rule from examples
- Large body of research
  - we can adopt these methods directly, when the agent to be modeled has discrete action space

# Induction in Machine Learning (ii)

## Artificial Neural Networks (ANN)

- Map input patterns onto (continuous) output patterns

$$n : R^m \rightarrow R^n$$

- Learn general rule from examples
- Large body of research
  - we can adopt these methods directly, when the agent to be modeled has continuous action spaces

## Induction in Machine Learning (iii)

# <Your Favorite Method Goes Here>

- Map ??? onto ???
- Learn general rule from examples
- *Your* body of research
  - you can adopt these methods directly, when the agent to be modeled has appropriate input/output types
- [Discuss!]

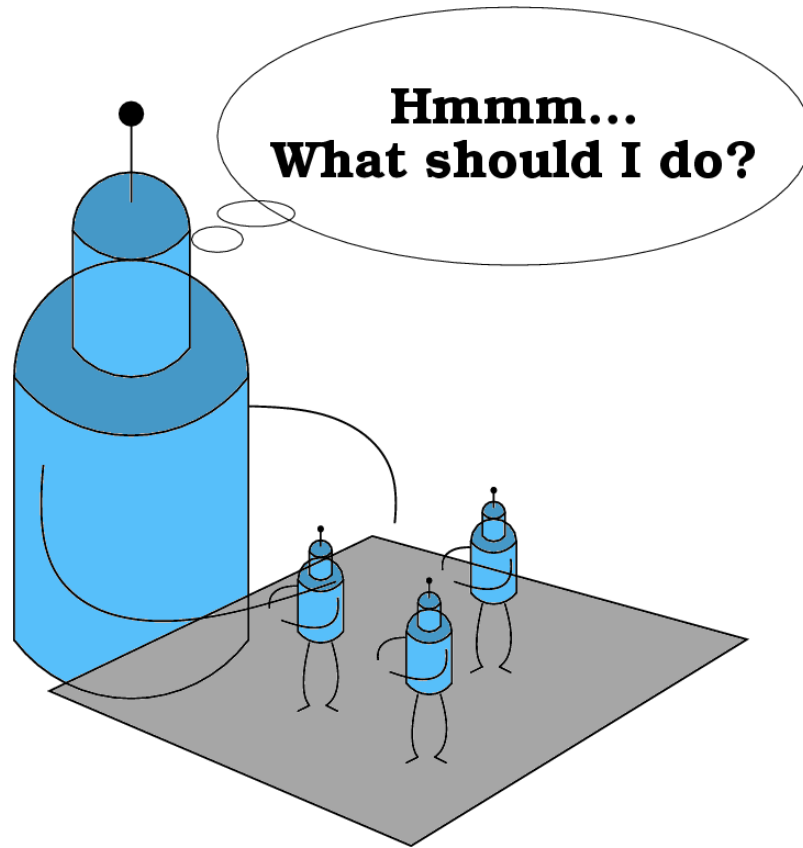
# Challenge: Change in POV (i)

Observer and/or learner may have a different observable state (or view thereof) than the exemplar has

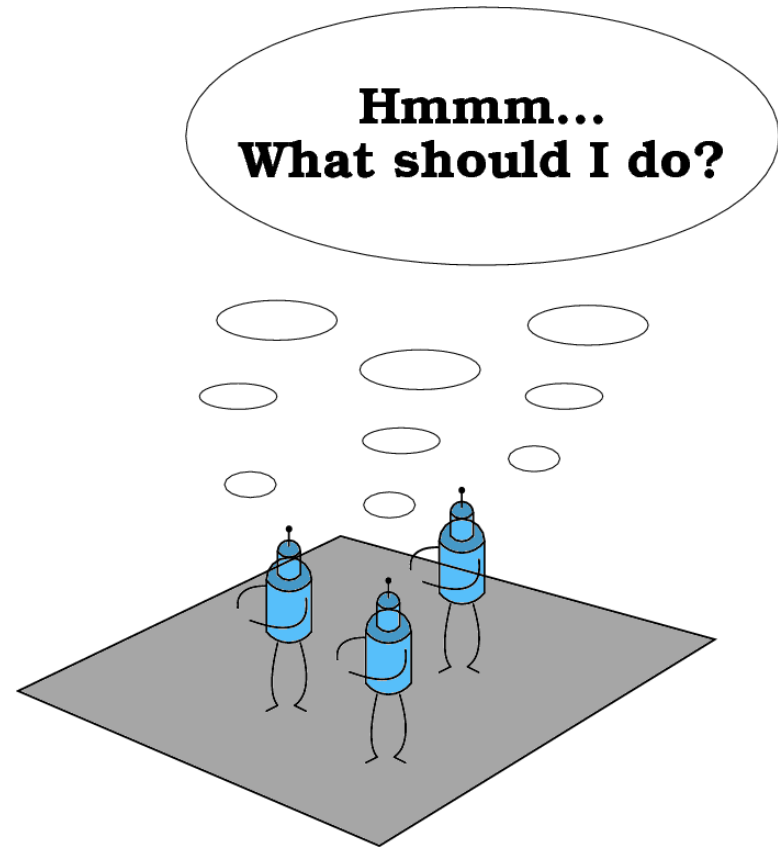
- e.g., *Legion-II* example (later)
  - exemplar is human player with “God’s-eye” view
  - observer/learner is in-game agent with egocentric view
- may make the induction harder
- can make the induction *impossible*,  
if critical state information is not visible
  - e.g., trying to model a driver when observer or learner cannot see street lights



# Challenge: Change in POV (ib)



Human Player



In-Game Agents

# Challenge: Change in POV (ii)

Observer's and/or learner's view of state may have a completely different modality than the exemplar's

- e.g., Parker & Bryant (in press) work on emulating Quake II bot
  - exemplar (bot) has direct access to games's state variables
    - distances, directions, etc.
  - observer/learner has only low-resolution rendered visual input
- presumably makes the induction harder

# Challenge: Measuring Success

- Nix “I don’t think the Red Baron would do it that way.”
- Approach based on Behavior Analysis
  - 2007 workshop
  - anecdote (if time allows)
- Holding back training examples for testing
  - conventional ML technique
  - best suggestion so far

# **Detailed Example Using Lamarckian Neuroevolution**

- But let's talk about this a bit more first...

# References

Bryant, B. D., and Miikkulainen, R. (2007). Acquiring visibly intelligent behavior with example-guided neuroevolution. In *Proceedings of the Twenty-Second National Conference on Artificial Intelligence (AAAI-07)*, 801–808. Menlo Park, CA: AAAI Press.

# **Acquiring Visibly Intelligent Behavior with Example-Guided Neuroevolution**

**Bobby D. Bryant**

Department of Computer Science and Engineering  
University of Nevada, Reno

**Risto Miikkulainen**

Department of Computer Sciences  
The University of Texas at Austin

Based on the AAAI'07 Presentation, July 25, 2007

# Visibly Intelligent Behavior

Focus on agents in environments where -

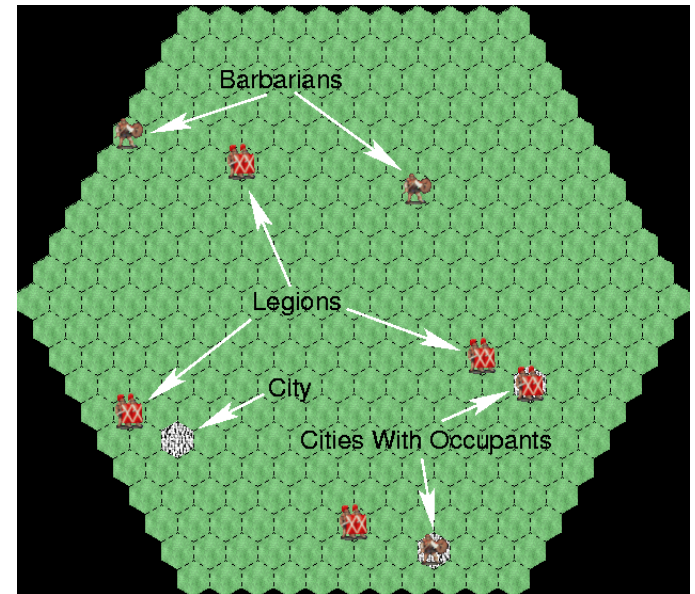
- The agents' behavior is directly observable
- Humans have intuitions about what is and is not intelligent



Obvious examples: games and simulators

# Example Game Environment: *Legion-II*

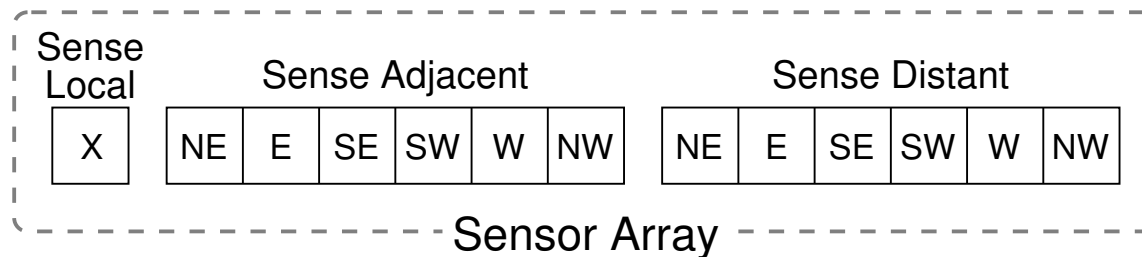
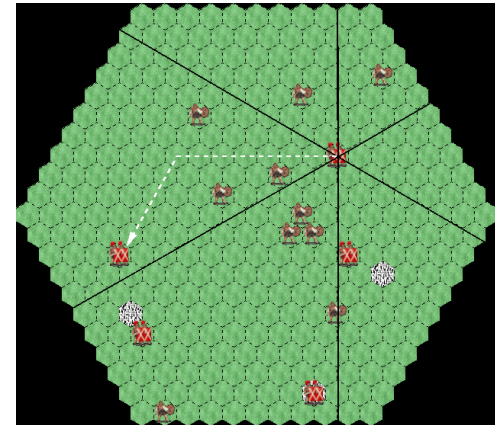
- Discrete-state strategy game with video display
- Pits legions vs. barbarians
- Legions must learn to cooperate to minimize the barbarians' pillage
  - pre-programmed barbarians
  - legions trained by neuroevolution
- Designed as multi-agent testbed
  - complex enough to produce interesting phenomena
  - transparent enough for analysis
  - scalable in complexity





# The Legions' Sensors

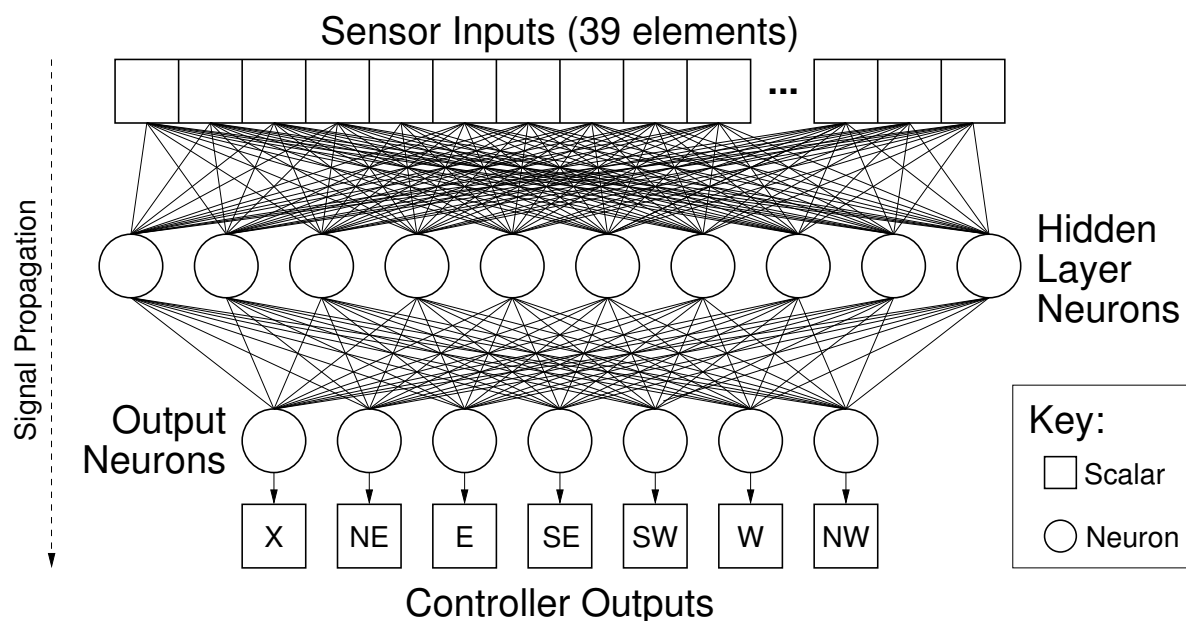
- Three egocentric sensor arrays detect cities, barbarians, and other legions
- Sub-arrays provide range and direction information for objects in six  $60^\circ$  “pie slices”:



- Each ranged element computed as  $\sum_i \frac{1}{d_i}$
- $3 \times (1 + 6 + 6) = 39$  FP sensor elements total
- Still only provides a fuzzy view of the map

# The Legions' Controllers

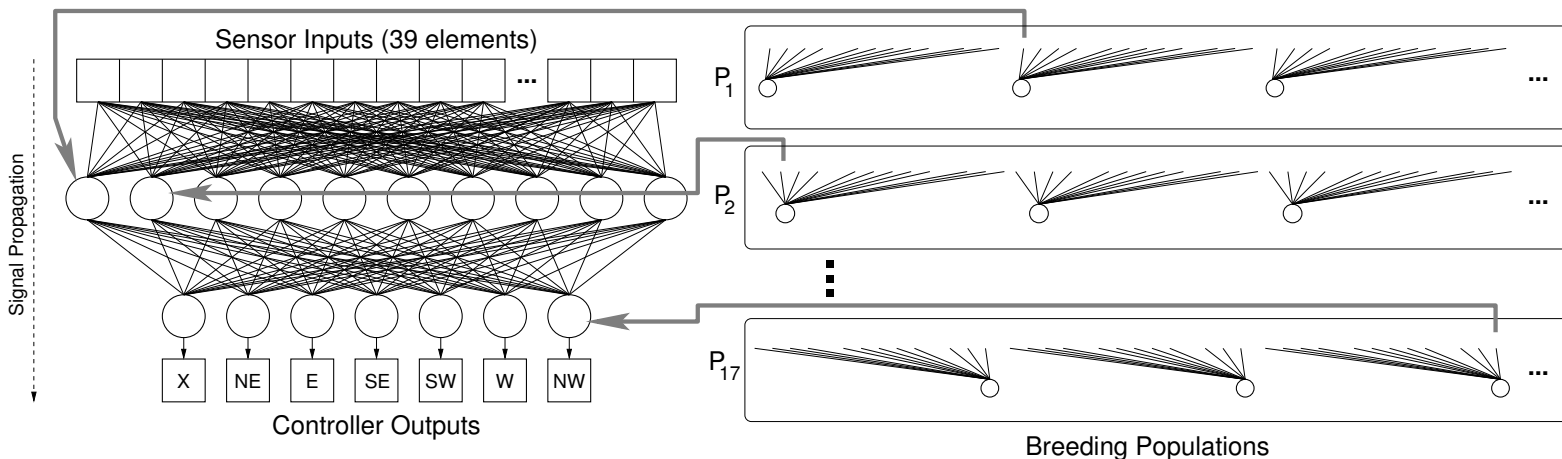
- Sensor activations are propagated through an artificial neural network:



- Then the network's output activations are decoded to choose an action
- The network must be trained to the task

# Neuroevolution with Enforced Sub-Populations (ESP)

(Gomez 2003)



- Direct-encoding neuroevolution mechanism
- Separate breeding population for each neuron
  - populations co-evolve to produce network
- Use game scores for fitness function
  - score = pillage rate (lower is better)
  - evaluation noise reduced by averaging

# How well does it work?

- Quantitative results – see prior publications

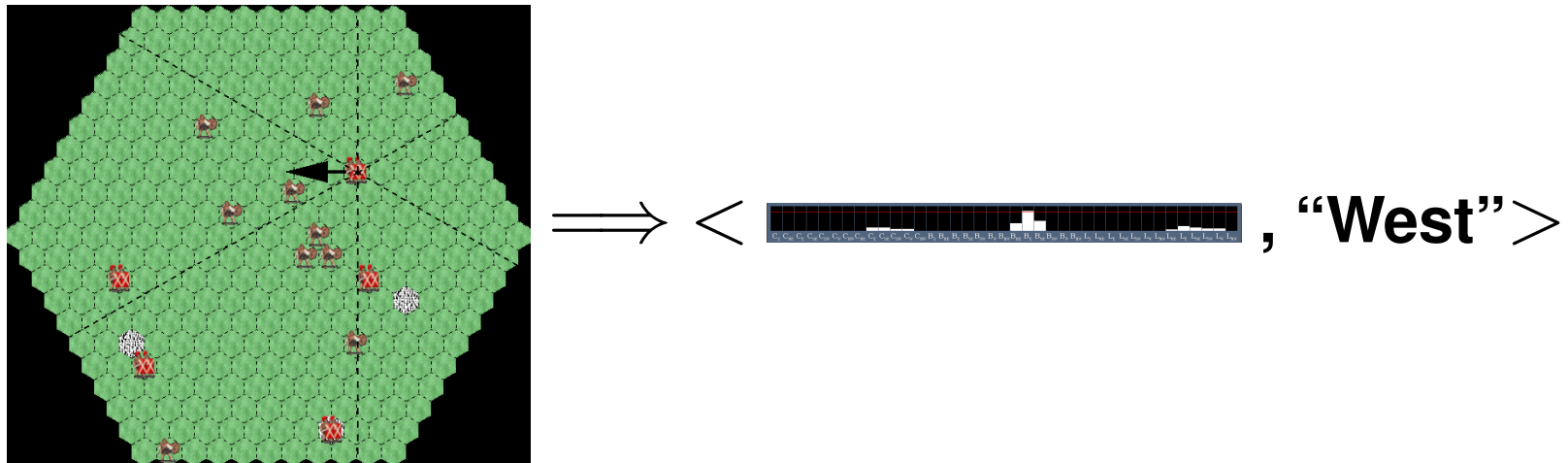
- visit

<http://www.cse.unr.edu/~bdbryant/#ref-research-publications>

- Qualitative results – see movie

# Learning from Examples

- Play a dozen games
- Capture examples as  $\langle state, action \rangle$  pairs
  - use egocentric sensor readings for *state*
  - use the human's choice of move for *action*

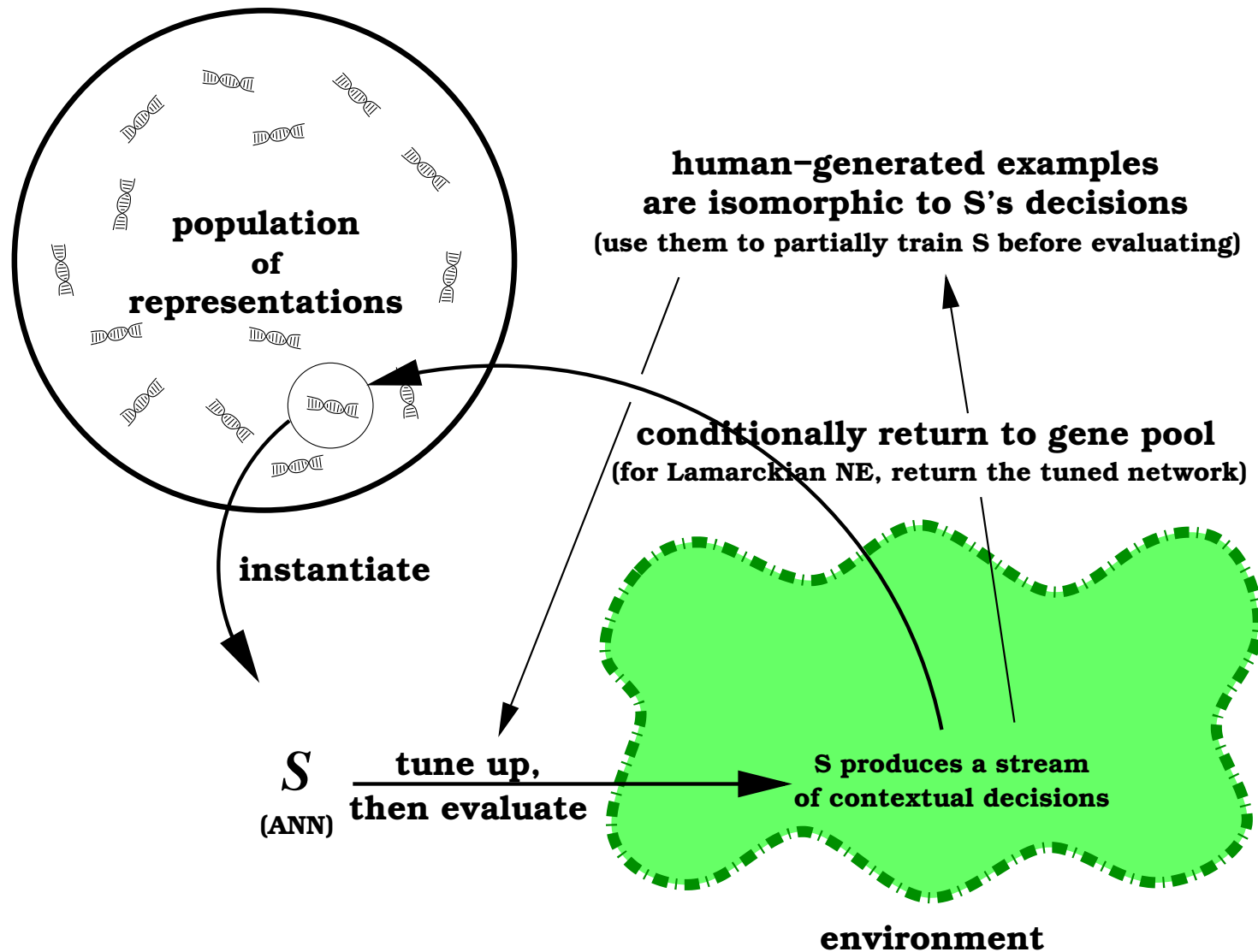


# Target Policies

(Used for Example Generation)

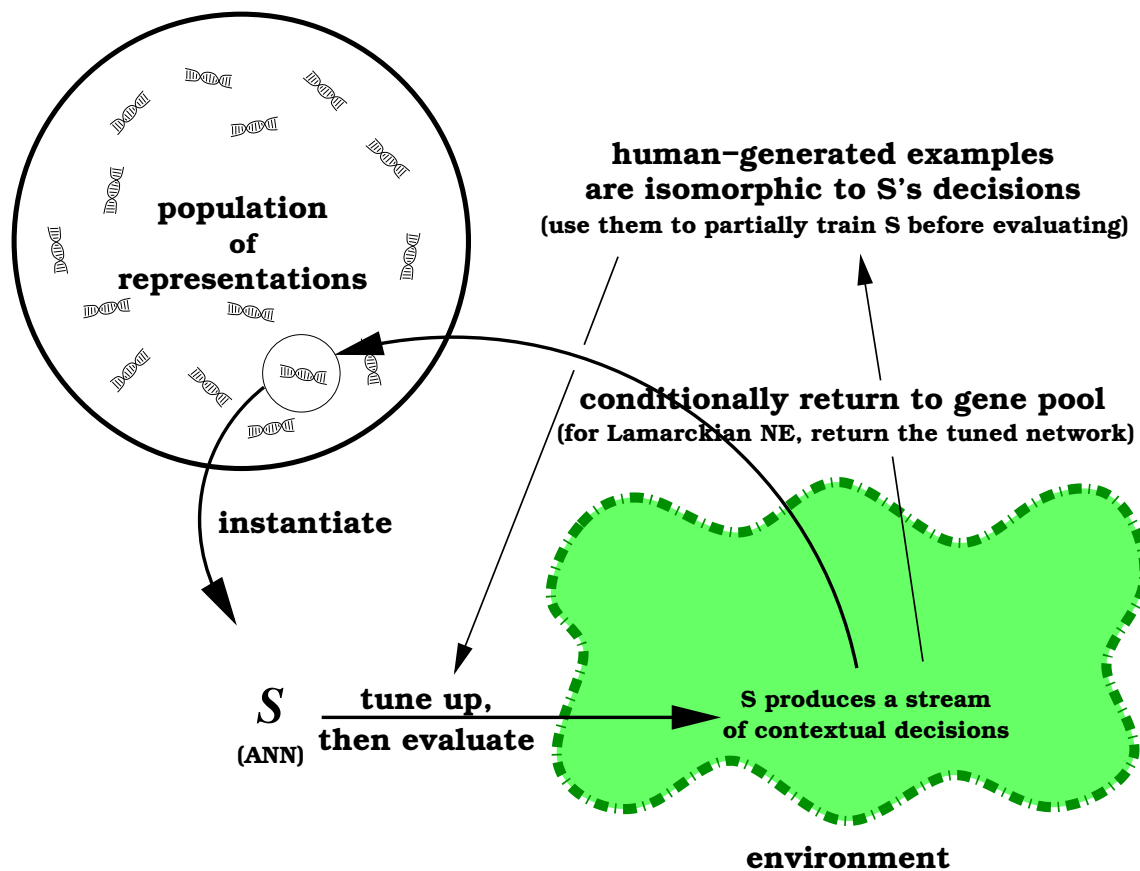
- Policy family  $L_d$ , where  $d$  is an integer distance
  - garrison may not move  $> d$  from city
  - must return to city when no barbarians within  $d$
- Safety condition
  - garrison may not end with barbarian equally near city
  - must move directly to city if unavoidable
  - side effect: two barbarians can lock a garrison in
- Examined only  $d \in 0, 1$  (limited sensor resolution)
  - notice that  $L_0$  is degenerate (trivial)
- No constraints on 'rovers'

# Lamarckian Neuroevolution



# Lamarckian Neuroevolution

## With ESP!



- Tuning is done with backpropagation
- ESP and Lamarckian mechanisms are orthogonal

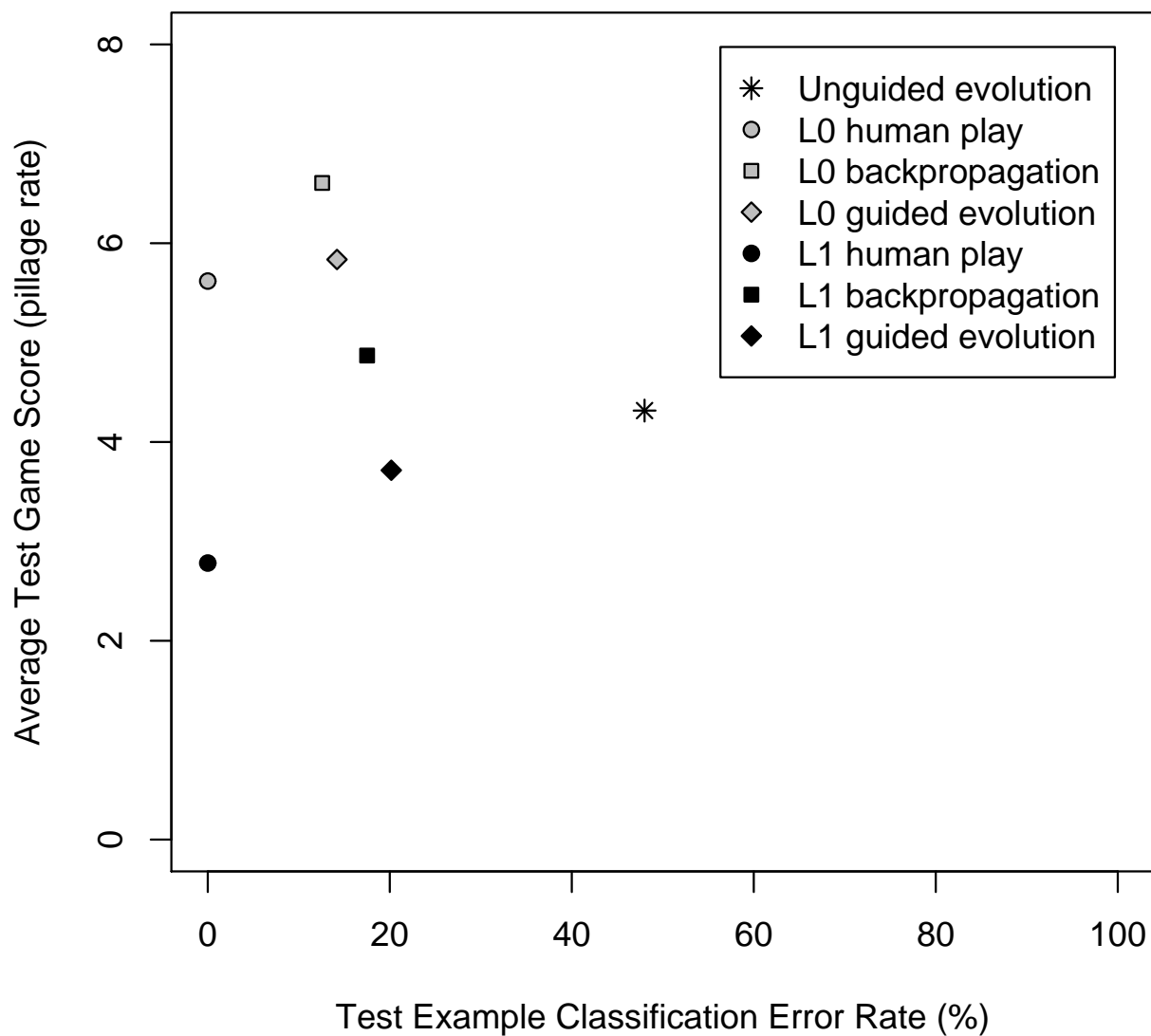


# Comparanda

- Unguided evolution (as before)
  - 5,000 generations
  - fitness = 1 / pillage rate
- Lamarckian neuroevolution
  - 5,000 generations
  - various amounts of training per generation
    - sample sizes: 5, 10, 20, 50, 100, 200, ... 10,000
    - only report results for 5,000 in the paper
    - in general, more examples  $\longrightarrow$  better results, higher cost
- Backpropagation
  - 20,000 epochs
  - full 11,000 examples per epoch

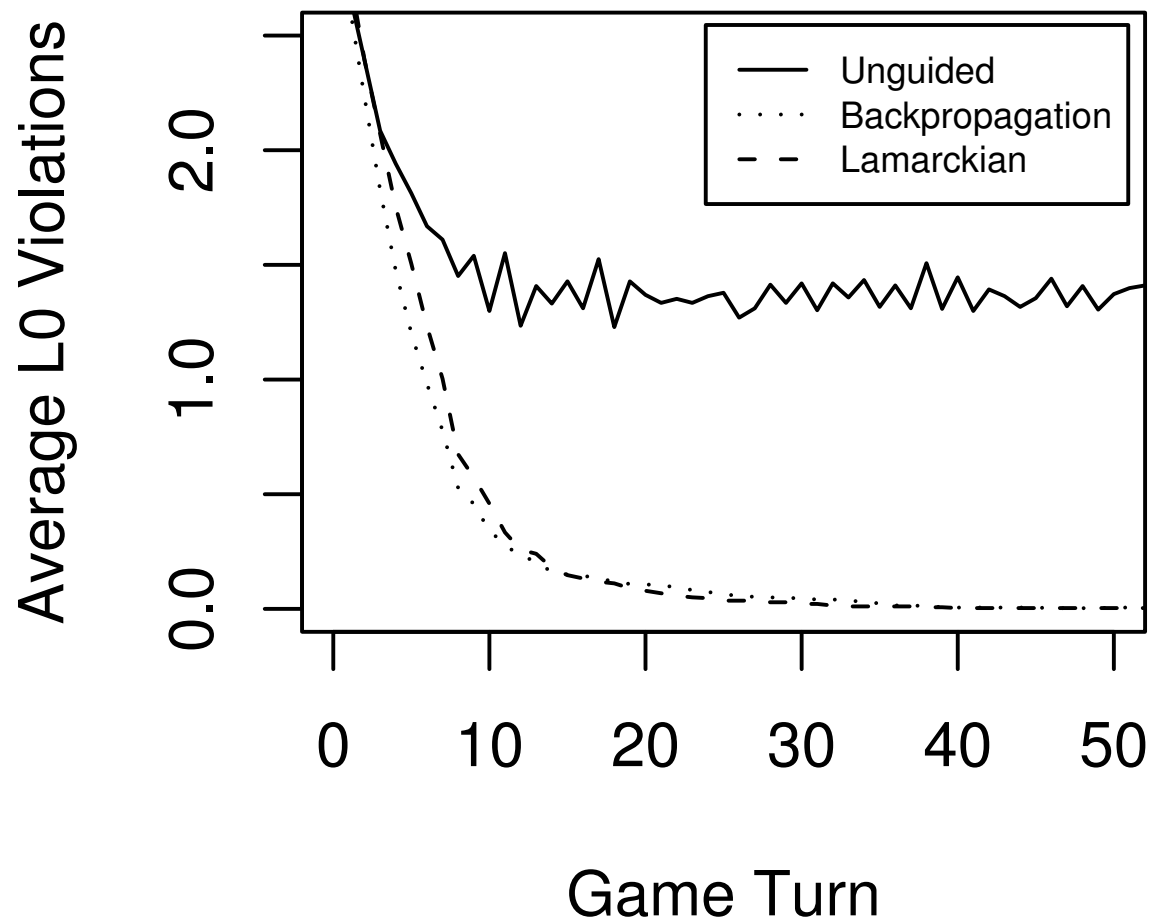
# Results (i)

## Coarse Behavior Metrics



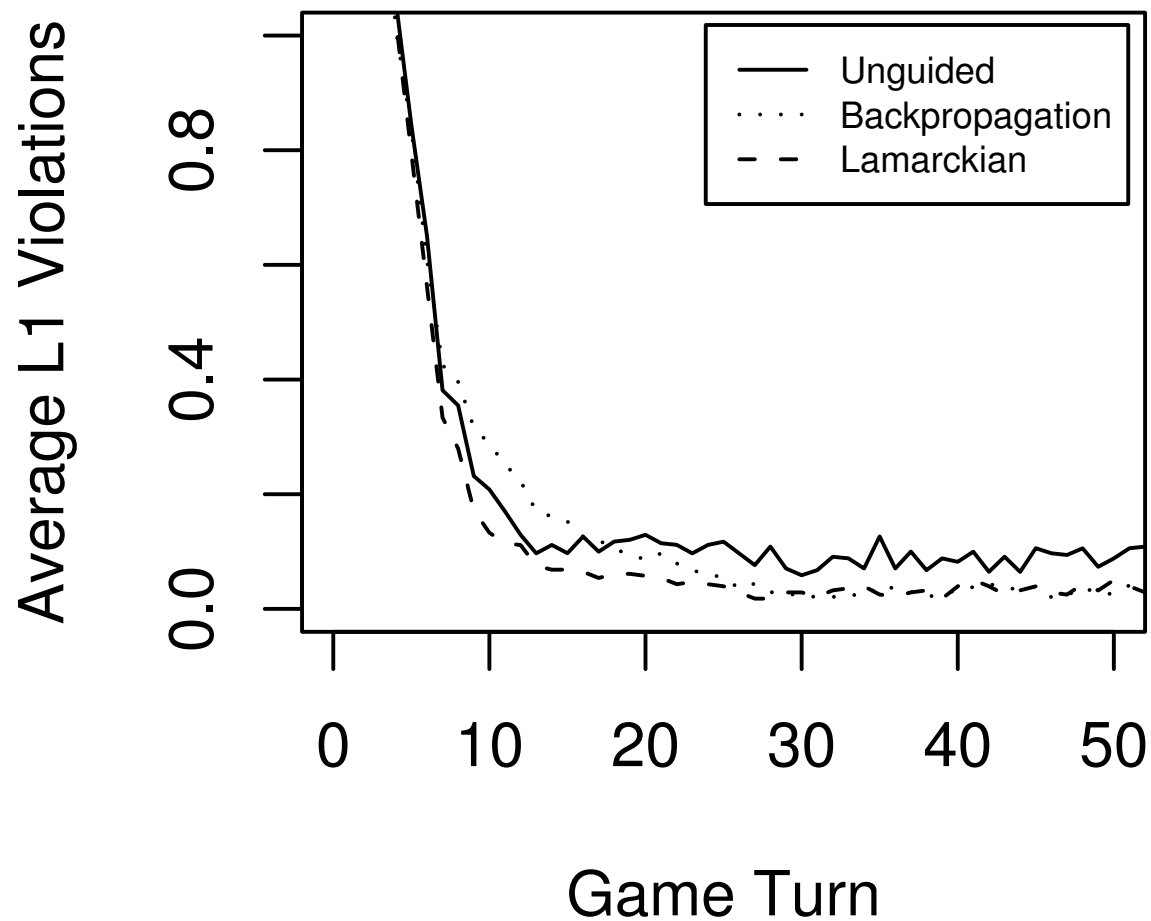
# Results (ii)

## Rule Induction – $L_0$ Results



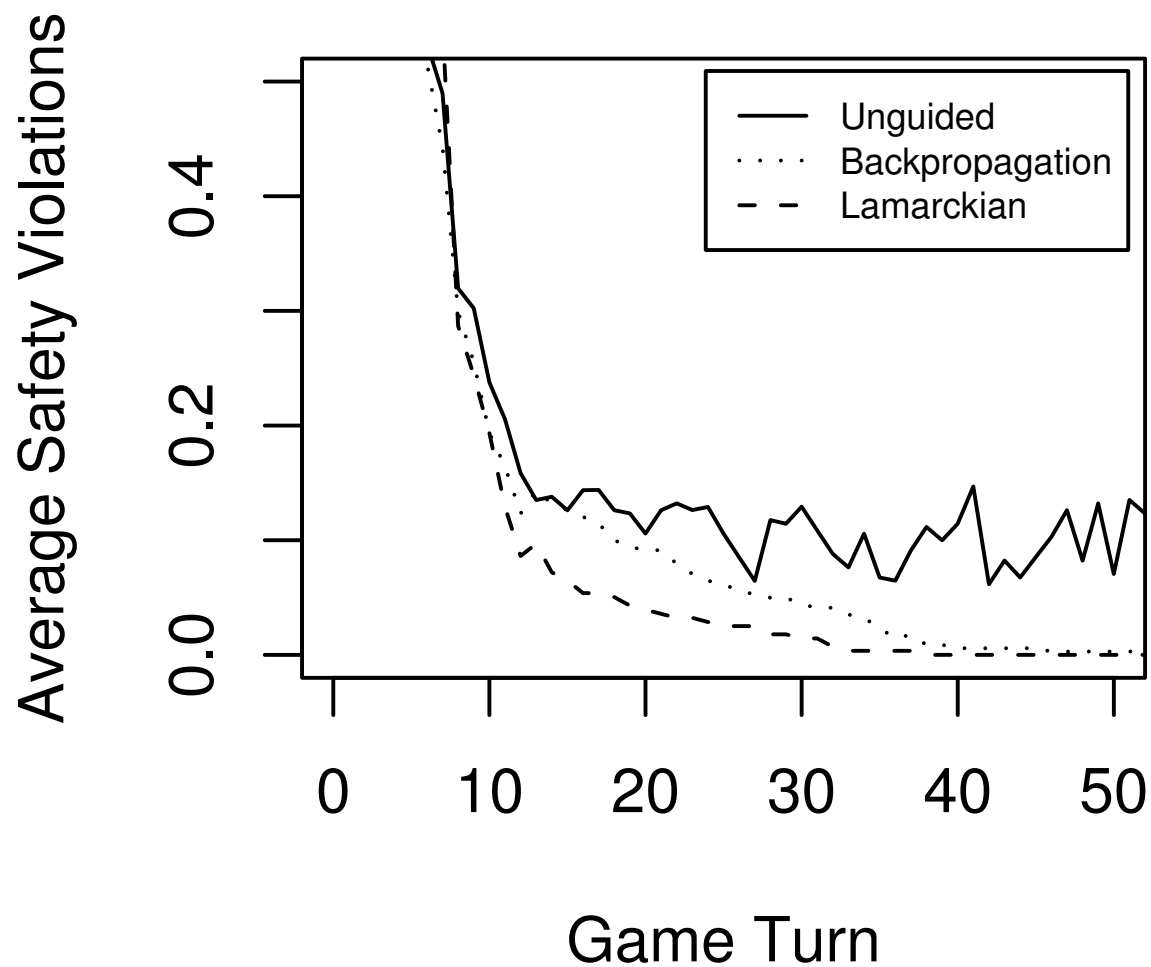
# Results (iii)

## Rule Induction – $L_1$ Results



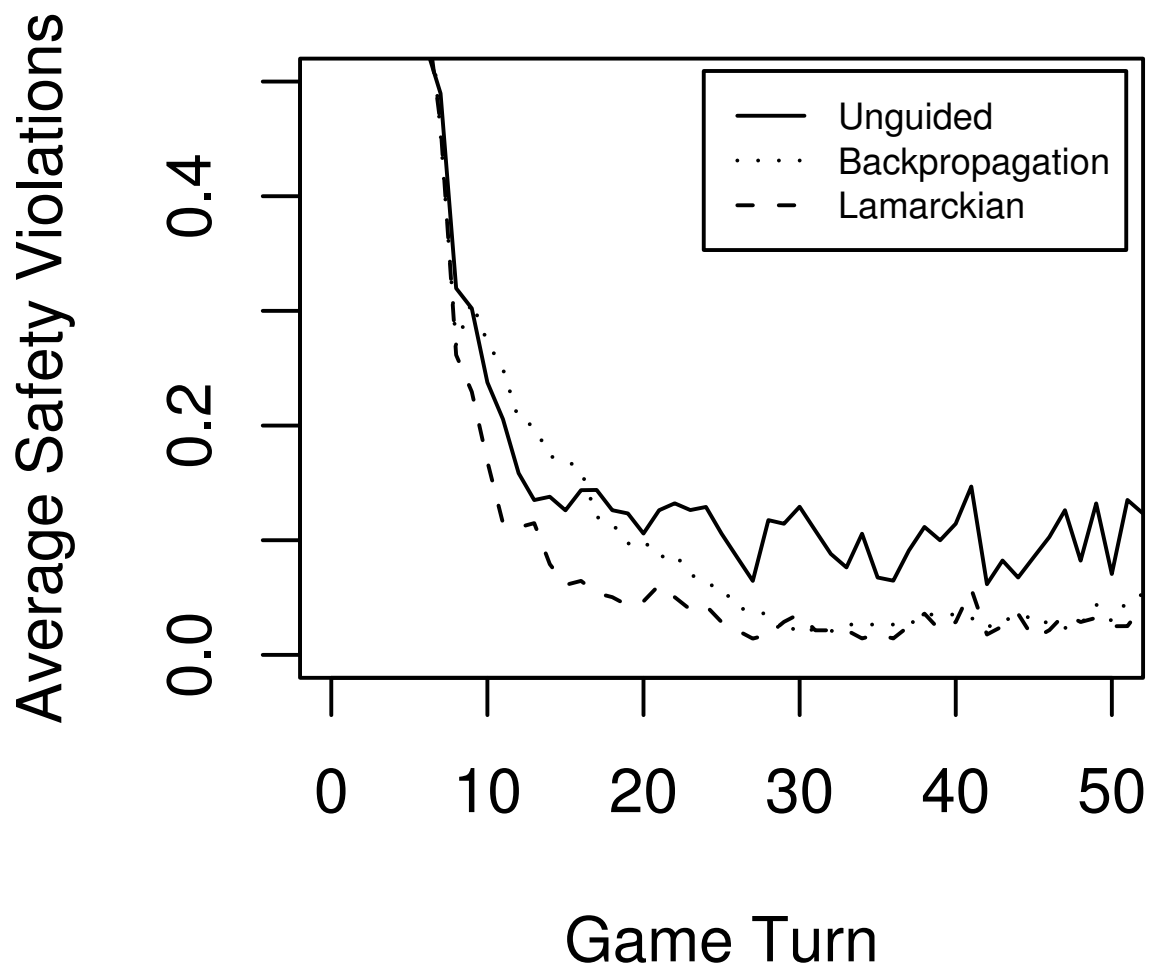
# Results (iv)

## Rule Induction – $L_0$ Results



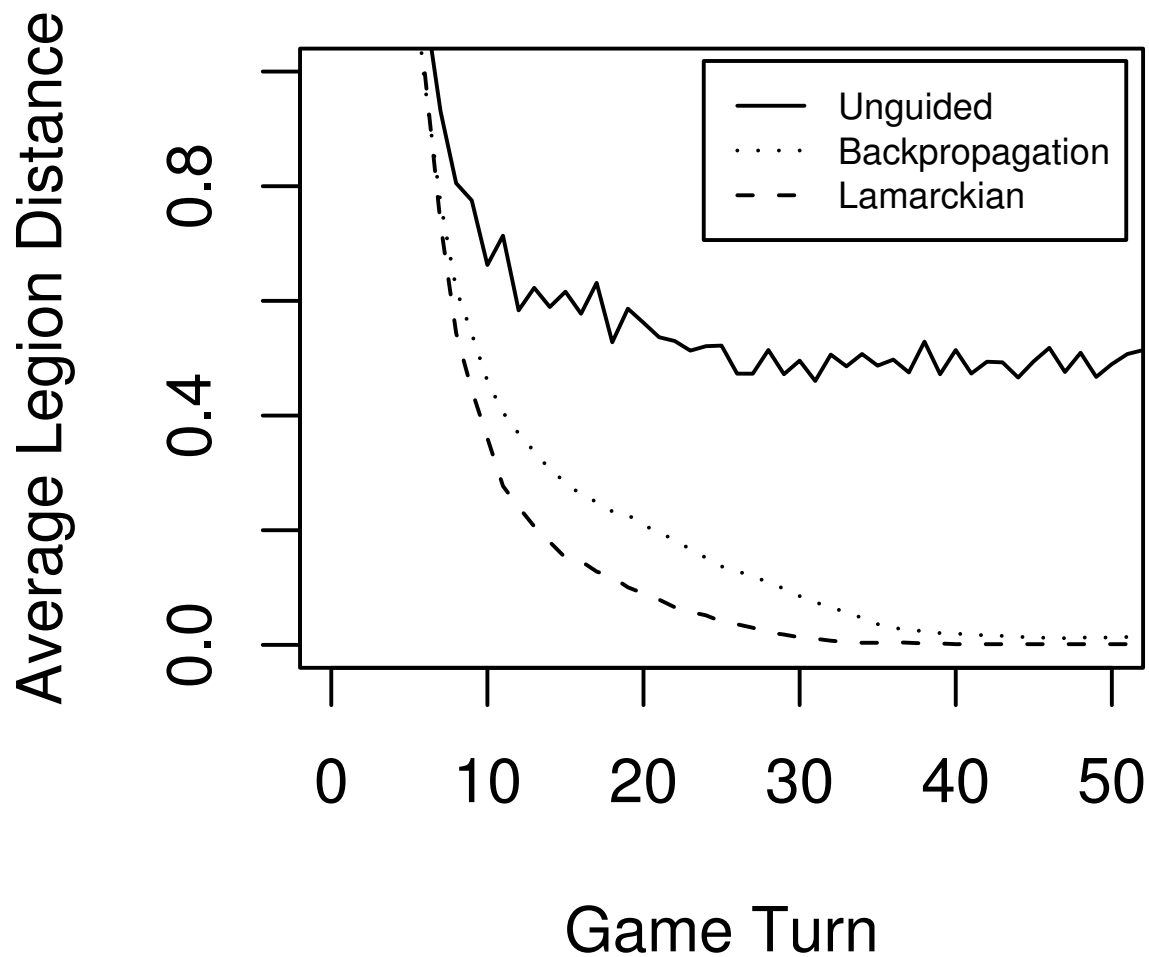
# Results (v)

## Rule Induction – $L_1$ Results



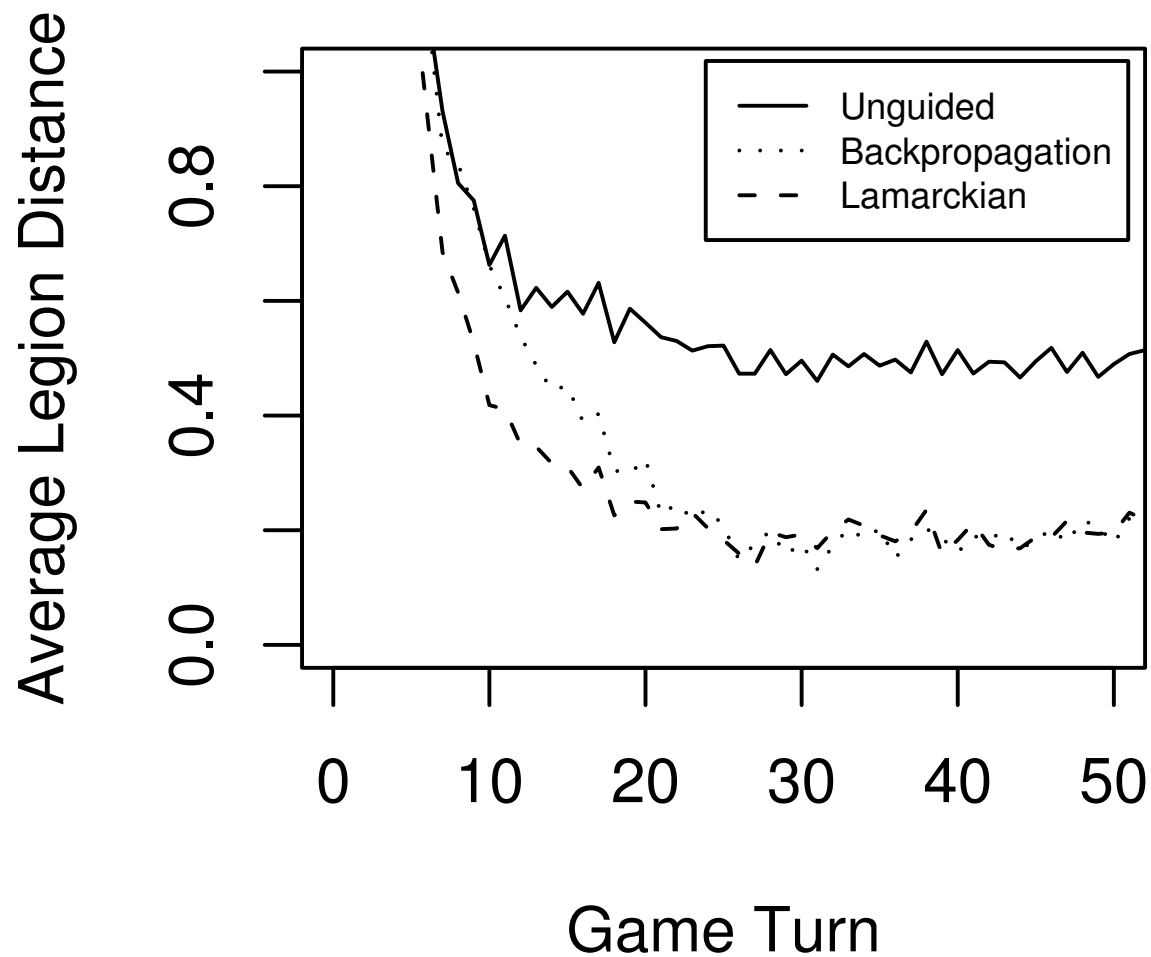
# Results (vi)

## Rule Induction – $L_0$ Results



# Results (vii)

## Rule Induction – $L_1$ Results





# Related Work

- Policy induction with rule-based systems
  - behavioral cloning (Sammut et al. 1992)
  - KnoMic (van Lent and Laird 2001)
  - Style Machines (Brand and Hertzmann 2000)
- Social robotics / mimetic algorithms
  - surveyed in Nicolescu (2003)
- Advice systems
  - RL *advice unit* (Kuhlmann et al. 2004)
  - appliqué networks (Yong et al. 2005)
- Inverse reinforcement learning (Ng and Russell 2000)
- User modeling
  - adaptive user interfaces
  - drama management for interactive fiction (AIIDE'07)

# Conclusions

- Some applications require intuitively correct behaviors
- Policy induction can simplify creating such behaviors
- Lamarckian neuroevolution can implement PI for some applications
  - beats backpropagation on enforcing rule conformance
  - more power and efficiency are needed
  - raises questions about how success can be measured

Related papers and movies can be found at

[www.cse.unr.edu/~bdbryant/#ref-research-publications](http://www.cse.unr.edu/~bdbryant/#ref-research-publications)

# Acknowledgments

This research was sponsored in part by the **Digital Media Collaboratory** at the IC<sup>2</sup> Institute at the University of Texas at Austin.

It builds on earlier research supported in part by the **National Science Foundation** under grant IIS-0083776 and the **Texas Higher Education Coordinating Board** under grant ARP-003658-476-2001.

CPU time for the experiments was made possible by NSF grant EIA-0303609, using the **Mastodon** cluster at UT-Austin.

Some of the images are taken from the open-source game *Freeciv*.

# References

- Gomez, F. (2003). *Robust Non-Linear Control Through Neuroevolution*. PhD thesis, Department of Computer Sciences, The University of Texas at Austin.
- Kuhlmann, G., Stone, P., Mooney, R., and Shavlik, J. (2004). Guiding a reinforcement learner with natural language advice: Initial results in RoboCup soccer. In *The AAAI-2004 Workshop on Supervisory Control of Learning and Adaptive Systems*.
- Ng, A. Y., and Russell, S. (2000). Algorithms for inverse reinforcement learning. In *Proceedings of the 17th International Conference on Machine Learning*, 663–670. San Francisco: Morgan Kaufmann.
- Nicolescu, M. N. (2003). *A Framework for Learning From Demonstration, Generalization and Practiced in Human-Robot Domains*. PhD thesis, University of Southern California.
- Sammut, C., Hurst, S., Kedzier, D., and Michie, D. (1992). Learning to fly. In *Proceed-*

## References

*ings of the Ninth International Conference on Machine Learning*, 385–393. Aberdeen, UK: Morgan Kaufmann.

van Lent, M., and Laird, J. E. (2001). Learning procedural knowledge through observation. In *Proceedings of the International Conference on Knowledge Capture*, 179–186. New York: ACM.

Yong, C. H., Stanley, K. O., and Miikkulainen, R. (2005). Incorporating advice into evolution of neural networks. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2005): Late Breaking Papers*.