



Neuroevolution of Combat Bots

Artificial Intelligence for
Interactive Media and Games

Professor Charles Rich
Computer Science Department
rich@wpi.edu

IMGD 400X (B 08)

1

Paper at AIIDE'08

- Constructing Complex NPC Behavior via Multi-Objective Neuroevolution

- Jacob Schrum
- Risto Miikkulainen



- University of Texas at Austin, CS Dept.

 WPI IMGD 400X (B 08)

2

Outline

- Machine Learning
- Neural Nets
- Genetic Algorithms
- Neuroevolution of Combat Bots

Machine Learning

- algorithms for improving performance based on experience
 - avoids “manual” programming labor
 - adapts to changing environment

Machine Learning

- algorithms for improving *performance* based on experience

“outputs of the system”

- recognizing speech
- diagnosing diseases
- controlling a combat bot

Machine Learning

- algorithms for improving performance based on *experience*

“input data”

- sound waves
- medical symptoms and test results
- actions of bot and player in game

Machine Learning

- algorithms for *improving* performance based on experience

“measure of performance”

- what the person actually said
- disease the patient actually has
- related to game design
 - how much damage bot inflicts on player
 - how much damage bot receives
 - how much fun the player has (harder to evaluate)

Machine Learning

- *algorithms* for improving performance based on experience

*“it’s all **search** (in very large spaces)”*

- reinforcement
- Bayesian
- simulated evolution (genetic algorithms)
- etc., etc.
- *issues*: efficiency, convergence, etc., etc.

Machine Learning

- *algorithms* for improving performance based on experience

“it’s all function approximation”

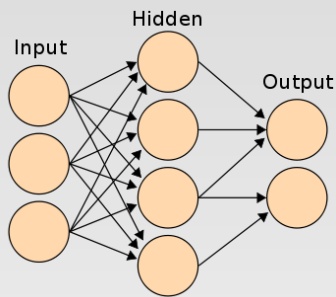
- given input/output pairs (“training set”)
 - each with evaluation of good the performance is
 - may be mix of good and bad performances
- induce a function which will produce good output for any input (“test set”)

Machine Learning

- supervised vs. unsupervised
 - *supervised*: system is given (by “teacher”) a planned sequence of experiences (training set), which will lead to efficient learning
 - *unsupervised*: system generates experiences by itself, e.g., by interacting with environment

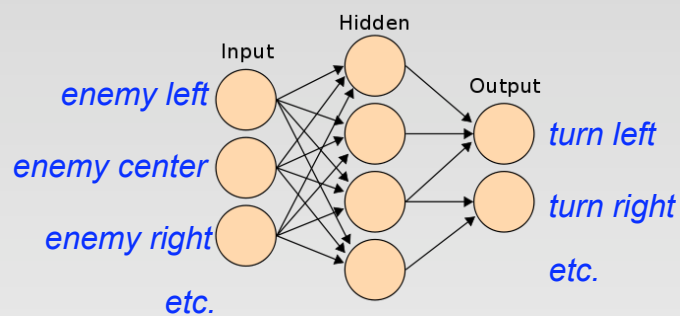
Neural Nets

an interconnected network of nodes, inspired by the network of neurons in the brain



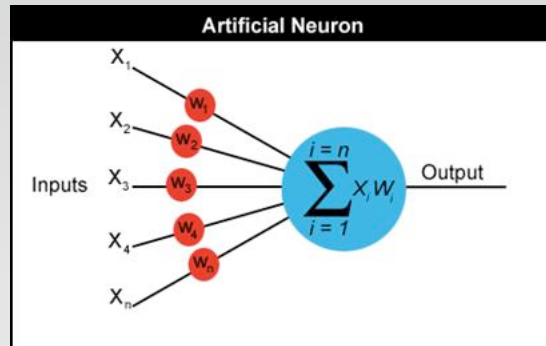
<http://www.ai-junkie.com/ann/evolved/nnt1.html>

Neural Net to Control Combat Bot



(NB: cannot sense other bots)

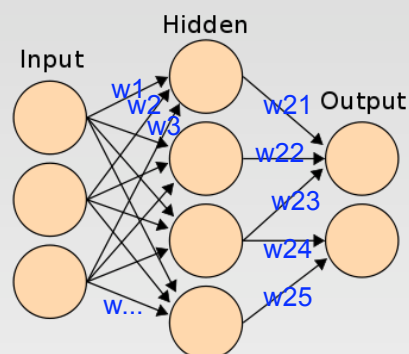
Neural Net Weights



$$a = x_1 w_1 + x_2 w_2 + x_3 w_3 \dots + x_n w_n$$

Neural Nets

- The “knowledge” is in the *structure* of the node connections and the *weights*



Neural Net Learning

- Initialize all weights to random numbers
- Typical supervised learning
 - start with totally connected network of given depth (hidden layers)
 - apply positive (negative) input/output training pairs
 - iteratively improve weights by *backpropagation* algorithm
- Neuroevolution
 - mutate weights and connections
 - use *genetic algorithm* for selection

Genetic Algorithms

inspired by natural evolution

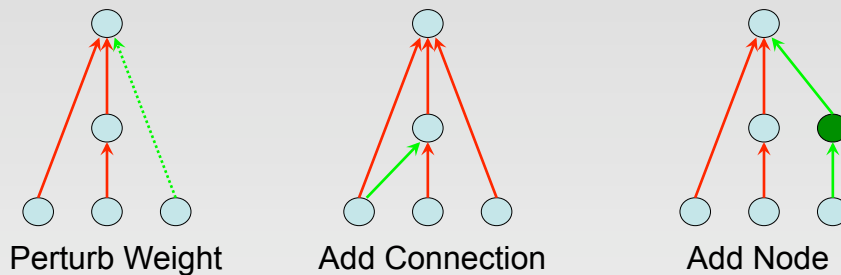
- **Given:**
 - a genetic representation each solution, e.g.,
 - DNA sequence
 - array of bits
 - neural net
 - a fitness function
 - applied to a genetic representation
 - relative to an “environment” (problem)
 - typically a numerical “score” (higher is better)

Combat Game Rules

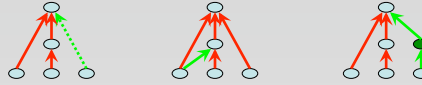
- player swings bat (weapon)
 - if player hits bot with bat
 - bot is knocked back
 - and incurs 10 points damage
- if bot hits player (attacks with body)
 - player is knocked back
 - and incurs 10 points damage
 - player cannot swing bat while being knocked back
 - afterwards, is always facing direction of bot that hit it

Neuroevolution of Combat Bots

- Genetic representation is neural net
- Three types of mutations (no crossover used)



Neuroevolution of Combat Bots



- **Breeding**
 - each “parent” bot creates a clone (copy) of itself
 - clone is mutated with some small probability
 - each mutation type has different fixed probability
- **“Elitist” Selection**
 - combined population plays against simulated player
 - best scoring (most fit) half of combined population
 - become “parents” of next generation

Player Simulation

Progression of three strategies (in order)

1. **Spinning**
 - player spins in place while swinging bat
 - to defeat this strategy, bots must learn to
 - wait until player’s back is turned,
 - then rush in and retreat
2. **Alternating**
 - player alternates between spinning and advancing
3. **Chasing**
 - player turns and moves toward closest bot
 - player and bots have same maximum speed

Player Simulation (cont'd)

- Player progressed to next strategy when all of the following satisfied
 - average amount of damage *received* from bots (as a *group*) is consistently over 100
 - average amount of damage *inflicted* upon *single* bot was consistently less than 20
 - average time alive per bot was consistently over 850

Contradictory Bot Objectives

1. maximize total damage to player (by the group)
 - requires coordination between bots (e.g., sacrifices)
2. minimize damage to self
 - the longer you live, the more chance you have to attack player
 - but you cannot just run away and stay safe

Three Fitness Measures

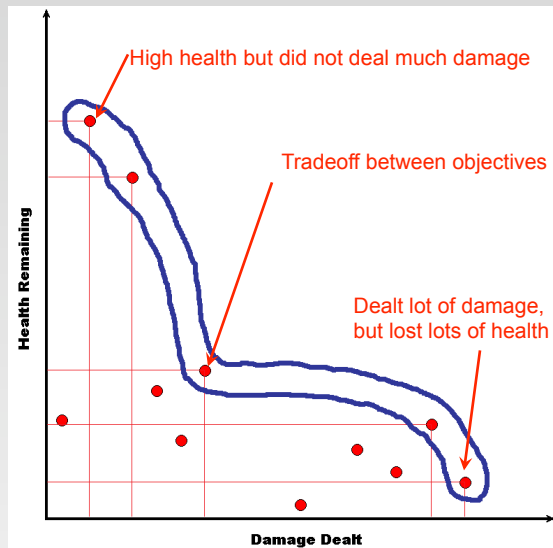
- **Attack Score**
 - all bots within small radius receive 10 points each time player is hit
 - bot that actually did the hit gets extra point
- **Damage Received**
 - negative 10 points for each hit received from bat
 - bot starts with 50 points (dead at zero)
- **Time Alive**
 - score is number of simulation time steps (0 – 900)
- *How to combine??*

Combining Fitness Measures

- **Compared two approaches**
 - single-objective
 - use single weighted score combining three measures
 - multi-objective
 - choose next parent population using “Pareto front”
- **Multi-objective approach worked much better**
 - evolves complex cooperative behaviors

Multi-Objective Optimization

- imagine a game with two objectives
- A dominates B iff A is strictly better in one objective and at least as good in others
- population of points not dominated are best: **Pareto Front**



Experimental Method

- simulation run 30 times
- each run consisted of 3 populations of 15 bots (total population of 45 bots)
- 300 generations
- each generation consisted of 5 evaluations over which the fitness scores were averaged

Results

- Complex, successful populations evolved in which the following two behaviors were mixed:
 - *baiting* – one bot takes a risk in front so that rest can attack from the back and sides (“evolved altruism” ??) [see video]
 - *charging* – keep knocking player back before player can recover to swing bat [see video]

“Multi-objective evolution has found a good balance between objectives, in that bots are willing to risk a little damage in exchange for a higher assist bonus”

Future Directions

- evolve against humans
 - takes a long time (many generations)
 - maybe can “snapshot” old evolutionary states and switch between them?
- evolve against scripted behaviors to find weaknesses