Neuroevolution Decision Strategies

- Input variables describe the state
- Output variables describe actions
- Network between input and output
  - Hidden nodes
  - Weighted connections
- Execution:
  - Numerical activation of input
  - Nonlinear weighted sums
- Performs a nonlinear mapping
  - Memory in recurrent connections
- Connection weights and structure evolved

Conventional Neuroevolution (CNE)

- Evolving connection weights in a population of networks
- Chromosomes are strings of weights (bits or real)
  - E.g. 10010110101100101111001
  - Usually fully connected, fixed topology
  - Initially random
Conventional Neuroevolution (2)

- Each NN evaluated in the task
  - Good NN reproduce through crossover, mutation
  - Bad thrown away
  - Over time, NNs evolve that solve the task
- Natural mapping between genotype and phenotype
- GA and NN are a good match!

Problems with CNE

- Evolution converges the population (as usual with EAs)
  - Diversity is lost; progress stagnates
- Competing conventions
  - Different, incompatible encodings for the same solution
- Too many parameters to be optimized simultaneously
  - Thousands of weight values at once

Advanced NE 1: Evolving Neurons

- Evolving individual neurons to cooperate in networks\textsuperscript{1,2,24} (Agogino GECCO’05)
- E.g. Enforced Sub-Populations (ESP? )
  - Each (hidden) neuron in a separate subpopulation
  - Fully connected; weights of each neuron evolved
  - Populations learn compatible subtasks

Evolving Neurons with ESP

- Evolution encourages diversity automatically
  - Good networks require different kinds of neurons
- Evolution discourages competing conventions
  - Neurons optimized for compatible roles
- Large search space divided into subtasks
  - Optimize compatible neurons
Advanced NE 2: Evolutionary Strategies

- Evolving complete networks with ES (CMA-ES\textsuperscript{15})
- Small populations, no crossover
- Instead, intelligent mutations
  - Adapt covariance matrix of mutation distribution
  - Take into account correlations between weights
- Smaller space, less convergence, fewer conventions

How Can Crossover be Implemented?

- Problem: Structures do not match
- Solution: Utilize historical markings

Advanced NE 3: Evolving Topologies

- Optimizing connection weights and network topology\textsuperscript{11,40}
- E.g. Neuroevolution of Augmenting Topologies (NEAT\textsuperscript{27,29})
- Based on Complexification
- Of networks:
  - Mutations to add nodes and connections
- Of behavior:
  - Elaborates on earlier behaviors

How can Innovation Survive?

- Problem: Innovations have initially low fitness
- Solution: Speciate the population
  - Innovations have time to optimize
  - Mitigates competing conventions
  - Promotes diversity
How Can We Search in Large Spaces?

- Need to optimize not just weights but also topologies

- Solution: Start with minimal structure and complexify
  - Hidden nodes, connections, input features (Whiteson GECCO’05)

Extending NE to Applications

- Evolving composite decision makers
- Evolving teams of agents
- Utilizing coevolution
- Real-time neuroevolution
- Combining human knowledge with evolution

Further NE Techniques

- Incremental evolution
- Utilizing population culture
- Evolving ensembles of NNs (Pardoe GECCO’05)
- Evolving neural modules
- Evolving transfer functions and learning rules
- Combining learning and evolution

Applications to Control

- Pole-balancing benchmark
  - Originates from the 1960s
  - Original 1-pole version too easy
  - Several extensions: acrobat, jointed, 2-pole, particle chasing
- Good surrogate for other control tasks
  - Vehicles and other physical devices
  - Process control
Competitive Coevolution

- Evolution requires an opponent to beat
- Such opponents are not always available
- Co-evolve two populations to outdo each other
- How to maintain an arms race?

Competitive Coevolution with NEAT

- Complexification elaborates instead of alters
  - Adding more complexity to existing behaviors
- Can establish a coevolutionary arms race
  - Two populations continually outdo each other
  - Absolute progress, not just tricks

Robot Duel Domain

- Two Khepera-like robots forage, pursue, evade
  - Collect food to gain energy
  - Win by crashing to a weaker robot

Early Strategies

- Crash when higher energy
- Collect food by accident
- DEMO
**Mature Strategies**

- Collect food to gain energy
- Avoid moving to lose energy
- Standoff: Difficult to predict outcome
- DEMO

**Sophisticated Strategy**

- “Fake” a move up, force away from last piece
- Win by making a dash to last piece
- Complexification → arms race
- DEMO

**Applications to Games**

- Good research platform
  - Controlled domains, clear performance, safe
  - Economically important; training games possible

- Board games: beyond limits of search
  - Evaluation functions in checkers, chess
  - Filtering information in go, othello

**Discovering Novel Strategies in Othello**

- Players take turns placing pieces
- Each move must flank opponent’s piece
- Surrounded pieces are flipped
- Player with most pieces wins
Strategies in Othello

- **Positional**
  - Number of pieces and their positions
  - Typical novice strategy
- **Mobility**
  - Number of available moves: force a bad move
  - Much more powerful, but counterintuitive
  - Discovered in 1970’s in Japan

Evolving Against a Random Player

- Network sees the board, suggests moves by ranking
- Networks maximize piece counts throughout the game
- A positional strategy emerges
- Achieved 97% winning percentage

Evolving Against an \(\alpha - \beta\) Program

- Iago’s positional strategy destroyed networks at first
- Evolution turned low piece count into an advantage
- Mobility strategy emerged!
- Achieved 70% winning percentage

Example game

- Black’s positions strong, but mobility weak
- White (the network) moves to f2
- Black’s available moves b2, g2, and g7 each will surrender a corner
- The network wins by forcing a bad move
Discovering Novel Strategies

- Neuroevolution discovered a strategy novel to us
- "Evolution works by tinkering"
  - So does neuroevolution
  - Initial disadvantage turns into novel advantage

Future Challenge: Utilizing Knowledge

- Given a problem, NE discovers a solution by exploring
  - Sometimes you already know (roughly) what works
  - Sometimes random initial behavior is not acceptable
- How can domain knowledge be utilized?
  - By incorporating rules (Yong GECCO’05)
  - By learning from examples

Numerous Other Applications

- Creating art, music
- Theorem proving
- Time-series prediction
- Computer system optimization
- Manufacturing optimization
- Process control optimization
- Etc.

Conclusion

- NE is a powerful technology for sequential decision tasks
  - Evolutionary computation and neural nets are a good match
  - Lends itself to many extensions
  - Powerful in applications
- Easy to adapt to applications
  - Control, robotics, optimization
  - Artificial life, biology
  - Gaming: entertainment, training
- Lots of future work opportunities
  - Theory not well developed
  - Indirect encodings
  - Learning and evolution
  - Knowledge and interaction
References


