

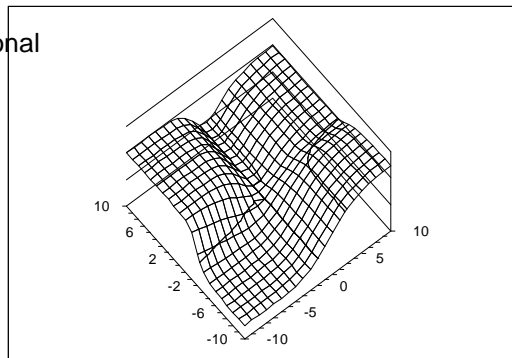
Learning Objectives

At the end of this lecture students will understand

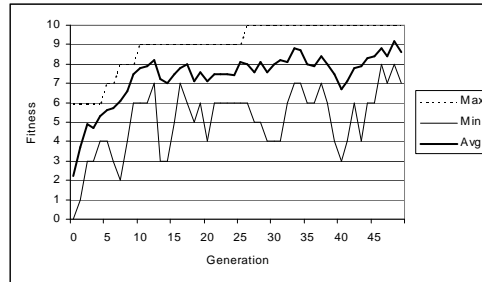
- What is an evolutionary algorithm?
- Effects of evolutionary operators
- **The course of computational evolution**
- Interactions between evolution and...

Fitness landscapes

- Wright (1932): for a given set of genes each possible combination of gene values (“alleles”) could be assigned a fitness value for a particular set of conditions
- Entire genotype space can then be visualized as a landscape, with genotypes of high fitness occupying peaks and those of low fitness forming troughs
- Generally very high-dimensional



The course of evolution in silico



This EA has a chromosome length of 10 bits and a population of 10 individuals. The fitness function is simply a count of the number of 1s in the chromosome; maximum fitness is therefore 10. The EA uses elitism, where the fittest individual in each generation is retained. Elitism ensures that a good solution, once found, is never lost, and means that the maximum fitness in the population always increases

Computational evolution

- Fitness initially random
- Increases over time
- Faster at first
- Eventually converges to a local optimum
- Not necessarily the global optimum
- Stochastic, so “evolution” works iteratively
- Can be time consuming
- Can produce good solutions that work unexpectedly

Schema Theorem

- Holland, 1975
- “short, low-order, above-average schemata receive exponentially increasing trials in subsequent generations”
- If the chromosome is a bit string, a schema is a set of building blocks described by a template consisting of ones, zeros and asterisks
- Template 10*001*1 can be
 - 10100111
 - 10100101
 - 10000101
 - 10000111

Schema theorem

- an evolutionary algorithm proceeds by identifying short schemas of high fitness in different individuals, and recombining them using crossover in order to produce longer schemas of higher fitness, and eventually entire individuals having high fitness
- attractive because it suggests that schemas can be identified and the effects of mutation and crossover upon schemas in a population of a given size can be calculated exactly
- mathematical tractability would potentially provide useful insights into the way in which an EA functions

Schema theorem

- Schemata: $H_1=011*1^{**}$ and $H_2=1*1^{****}$ have many instances, e.g. 0111111 and 1010010 resp.
- The order of H : $o(H)$, is the number of specified bits, e.g. $o(H_1)=4$.
- The defining length of H : $\delta(H)$, is the difference between the last and first position of specified bits, e.g. $\delta(H_1)=5-1=4$.
- A_i represent the i th bit string in A , and its fitness is f_i .
- We are interested in m : $m(H,t)$ the number of examples that are instances of a particular H , at time t (which refers to a particular population).

Scenario: Only selection (no crossover or mutation)

String A_i is selected with probability $f_i / \sum f_k$

$$m(H, t+1) = m(H, t) \cdot n \cdot \frac{f(H)}{\sum f_k}$$

where $f(H)$ refers to the average fitness of all H 's instances, and n the number of instances in the population.

$$\bar{f} = \sum f_k / n \qquad m(H, t+1) = m(H, t) \frac{f(H)}{\bar{f}}$$

The equation states that any schema H grows in numbers proportional to the ratio of the average fitness of H to the average fitness of the population as a whole. Put another way, schemata with fitness values above the population average increase in numbers in the next generation and those with below average fitness decrease in numbers

Premature convergence

- In most EAs the entire population eventually reaches a single peak and tends to stay there
- If this peak is not the global maximum, the EA is considered to have converged prematurely
- Premature convergence occurs when the population loses the genetic variability which is essential to continued evolution
- This almost complete loss of genetic diversity is never observed in biological populations

Hill Climbing

- *implicit parallelization*; by maintaining a population of candidate solutions which are modified by mutation and/or crossover, the algorithm is, in effect, exploring different regions of its search space in parallel
- simplest alternative to a population based EA is a *hill climber*, an algorithm which has a population of one individual, and performs a strictly local search using mutation
- The parallel nature of an EA provides no obvious advantages over multiple random restarts of a hill climber in terms of the number of solution evaluations performed

When is an EA better?

1. the action of the genetic operators used in the EA provided advantages over local search, which would, indeed, be the case if the schema theorem was acting as described, with useful partial solutions discovered by different individuals being recombined to produce fitter individuals more rapidly than could be done by mutation alone; or
2. the structure of the fitness landscape was such that the “implicit memory” of a population-based algorithm (i.e. the memory encoded into the structure of the population itself as a result of evolution) allowed it to concentrate its search in areas of high fitness in a manner that would not be possible for a hill climber
 - In practice, hill climbers with multiple restarts often perform as efficiently as or better than population-based algorithms

Learning Objectives

At the end of this lecture students will understand

- What is an evolutionary algorithm?
- Effects of evolutionary operators
- The course of computational evolution
- **Interactions between evolution and...**

Co-evolution

- The interactions between two or more species as they evolve
- Kauffman's rubber sheet: evolution by one species modifies the fitness landscape for both species; the co-evolving species is thus given a spur to further evolution, as its environment changes
- Fitness landscape is constantly changing
- Powerful strategy for avoiding premature convergence in evolutionary algorithms; is less chance of the population converging to a local minimum, since local (and global) minima are constantly forming and dissolving as the fitness landscape changes

Using co-evolution

- Samuels' checker players (1963)
 - hill climber, in which two programs played against each other
 - In the course of the game one program modified its parameter settings, while the other remained static
 - If the modified copy won the game, it was accepted, otherwise the original was retained
 - eventually played checkers at the level of a human champion
- Fogel (2001) still using evolution to develop checkers players (Blondie21)

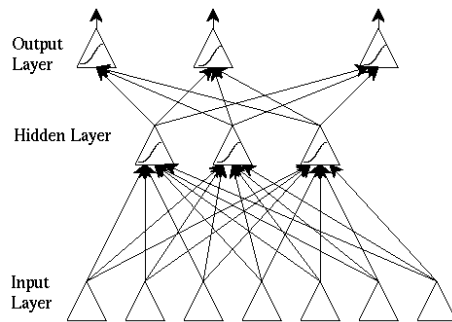
Learning and evolution

- Neural networks may be evolved: architecture, connection weights, or both
- Baldwin Effect (Baldwin, 1896): learning on the part of individuals could guide the course of evolution in the population as a whole
- A particular trait may be learned, or it may be innate
- A learned trait has the advantage of providing flexibility, but the disadvantage of being slow to acquire; an innate trait is present from birth, but inflexible
- Traits which are initially learned may become, over time, encoded into the genotype of the population

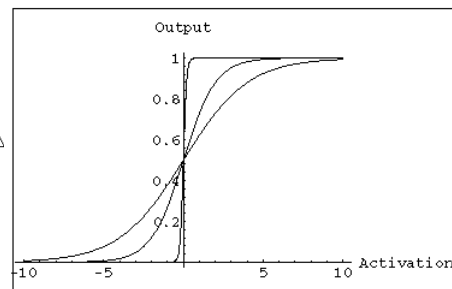
Learning and Evolution

- Neural networks may be evolved
 - architecture
 - connection weights
 - architecture
 - learning rules
 - input features
- Individual learns; population evolves

Neural networks



$$y_i = \left(\sum_{j=1}^n w_{ij} x_j - \phi_j \right)$$



<http://www.cs.stir.ac.uk/~lss/NNIntro/InvSlides.html#what>

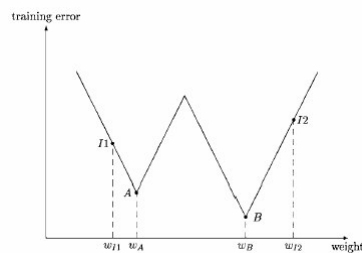
Evolving connection weights

1. Decode each individual (genotype) in the current generation into a set of connection weights and construct a corresponding ANN with the weights
2. Evaluate each ANN by computing its total mean square error between actual and target outputs. The higher the error, the lower the fitness. Large weights may be penalized
3. Select parents for reproduction based on their fitness
4. Apply genetic operators such as crossover and mutation to generate offspring

Yao (1999)

Pros and cons

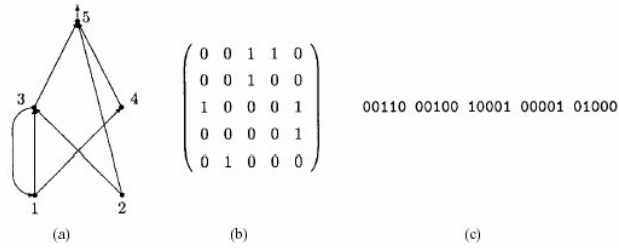
- EC training may be faster - depends upon problem and specific algorithms used
- May combine global and local search
 - e.g. use EA to find good initial connection weights, then use BP to fine-tune them



Evolving architectures

1. Decode each individual in the current generation into an architecture. If an indirect coding scheme is used further details may be provided by a developmental process
2. Train each ANN using a predefined learning rule and different initial weights and parameter values
3. Compute the fitness of the ANN - may incorporate penalties for architectural complexity
4. Select parents based upon their fitness
5. Apply genetic operators to produce next generation
 - Can evolve weights and architecture together

Encoding an architecture

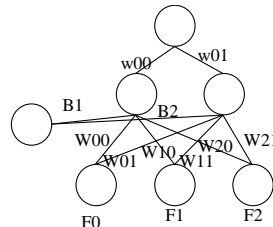


Evolution of transfer functions

- Can add new nodes with different types of function chosen at random (sigmoidal, Gaussian, etc)
- Or use a given function and encode parameter values in the genome

An example

Neural Network



Conceptual Organization

F0	W00	W01	F1	W10	W11	F2	W20	W21	w00	w01	B1	B2
----	-----	-----	----	-----	-----	----	-----	-----	-----	-----	----	----

Physical Implementation

010010101111010101000101010100100101000001010101011011010100101001010100101100101010

The Baldwin Effect

- Mark James Baldwin, 1896: learning on the part of individuals could guide the course of evolution in the population as a whole
- A particular trait may be learned, or it may be innate
- A learned trait has the advantage of providing flexibility, but the disadvantage of being slow to acquire; an innate trait is present from birth, but inflexible
- Traits which are initially learned may become, over time, encoded into the genotype of the population
- Sounds Lamarckian!

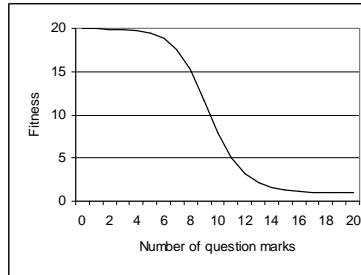
The Baldwin effect

- Can the learning that occurs during an individual's lifetime guide evolution of that individual's species?
- According to Darwin's theory a genetically encoded trait that is beneficial to the survival of the individual will be selected and transferred to the population.
- Let's say that an individual - under its lifetime - acquires (by learning) a trait. The trait itself is not genetically encoded, merely, the ability to acquire it...
- Now, let's further assume that this acquired trait is beneficial to the individual's survival. That means that - over time - the population becomes dominated by the individuals that have the ability to acquire the trait, i.e. learn the trait.
- Moreover, selection will prefer those individuals that quickly acquire the trait. Hence, the population will soon be dominated by those individuals that learn quickly. Indirectly, the trait becomes (more or less) genetically encoded. This is the Baldwin effect.

The Baldwin effect

- Hinton and Nowlan (1987) provided a clear (and simple) demonstration of the Baldwin effect in a computational simulation.
- An individual had a brain with 20 neuronal connections. According to the genome of the creature, each connection was present (1), not present (0) or learnable(?). There was only one configuration that was "good" (11111111111111111111).
- Hence, the search was like looking for a needle in a haystack.
- However, if a connection was learnable, it could acquire the presence of the connection (1) after some attempts of learning it (each individual has a set number of "guesses" in its life)
- Consequently, a genome specifying a single connection as not being present (0) was doomed.
- A genome with some learnable connections could (with a little luck in Hinton and Nowlan's simulation) acquire the necessary connections (switching ? to 1 in the phenotype). Fitness was based on the time it took to acquire the right phenotype.
- As a result of this added flexibility the population soon genetically encoded a majority of the connections as present with a few as being learnable.
- All individuals in the population quickly acquired the necessary connections after conception.
- Without the "learnable" connections the population would never (unless extremely lucky) find the needle...

Fitness landscape



$$f = 1 + \frac{(L-1)(G-g)}{G}$$

N = 1,000

L = 20

G = 50

Course of evolution

No mutation

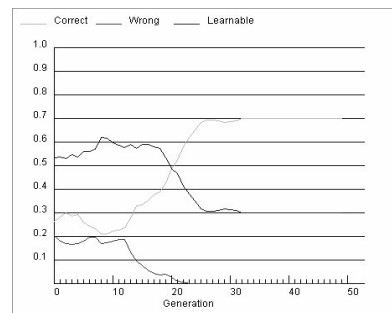
Crossover 1.0

Roulette wheel selection

Initial proportions

1 / 0 / ?

0.25 / 0.25 / 0.50



The Baldwin effect

- Two preconditions must be met
 1. The trait in question (which may be a behaviour or a physical trait) must be influenced by several interacting genes, so that a mutation in one of these genes will make the phenotypic expression of the trait more likely; and
 2. an individual bearing such a mutation can learn to express the trait
- learning acts to provide “partial credit” for a mutation
- An individual carrying a mutation that predisposes it towards an advantageous phenotype will learn the trait more easily than its less fortunately genetically endowed conspecifics, and thus will survive and pass on more copies of that allele to the next generation. Over time, multiple mutations will accumulate in the genes for the desirable trait, which will thus become innate in the population

Conclusions

- Evolutionary computation can be used to
 - Model biological evolution
 - Optimize arbitrary functions
 - Evolve artificial intelligence?
- EC is a very simplified version of real evolution
- It is important to understand what these simplifications involve
- Can be combined with many other approaches e.g. neural networks
- EC can be used to solve problems which are otherwise intractable

Learning Objectives

At the end of this lecture students will understand

- What is an evolutionary algorithm?
 - Population-based, adaptive, optimization, not necessarily global optimum
- Effects of evolutionary operators
 - Mutation, crossover, selection
- The course of computational evolution
 - Fitness increase, variability, premature convergence, etc
- Interactions between evolution and...
 - Coevolution, evolution and learning

More Learning Objectives

- **What is evolutionary robotics?**
 - **What can be evolved?**
 - **What are the advantages of evolving robotic components?**
- What is embodied evolution?
 - What are the implications for computer models?
 - What are the implications for real systems

Evolutionary Robotics

- Early 1990s
- Automatic creation of autonomous robots
- Robots as autonomous artificial organisms that develop their own skills in close interaction with the environment without human intervention
- Makes use of tools like neural networks, genetic algorithms, dynamic systems, and bio-morphic engineering
- Can evolve
 - robot control: totally in simulation or tested in hardware
 - perform evolution within a population of robots
 - robots themselves

Why?

- Save human time and effort
- Evolve solutions that a human may not have thought of
- Problems
 - Computational time
 - Differences between simulation and embodiment - transference issues
 - Power delivery
 - Robustness
 - Fitness function (an issue with all EC)

Evolving robot control - Evorobot

- Stefano Nolfi
- Written in C / C++
- Run simulations or test in hardware
- *Embodied trials*
- Khepera robot (Laboratory of Microprocessors and Interfaces, Swiss Federal Institute of Technology in Lausanne)



Khepera

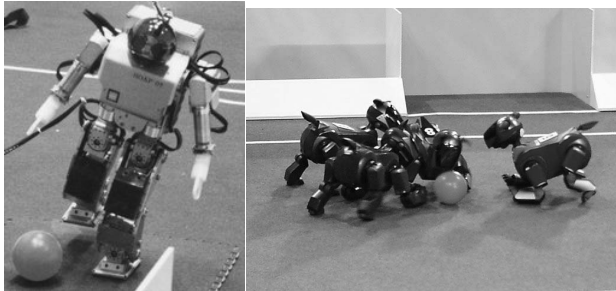
Element	Technical Information
Processor	Motorola 68331
RAM	256K
ROM	256 or 512K
Motion	2 DC motors with incremental encoder (about 10 pulses per mm of advancement of the robot)
Sensors	8 Infra-red proximity and light sensors
Power	Rechargeable NiCd Batteries or external
Autonomy	30 minutes (basic configuration with maximal activity)
Extension bus	The robot can be expanded by modules added on the K-Extension bus.
Size	Diameter 55mm
	Height 30mm
Weight	About 70g

Robocup

- By the year 2050, develop a team of fully autonomous humanoid robots that can win against the human world soccer champion team

- 5 categories

- Simulation
- Small size
- Middle size
- 4 legged
- Humanoid



RoboRoos (UQ)

- Grand finalists 1998
- Quarter finalists 1999
- Semi finalists 2000
- “The playing team consists of five field robots and one specialised goal keeper robot. The robots have been designed with simplicity and agility in mind. Each robot is driven by a pair of DC motors in a wheelchair configuration. The centre of gravity and turning moment of the robot have been kept minimal to maximise agility. Each wheel carries a high resolution rotary encoder for velocity control. A custom built 68332 controller board controls the robots. The controller performs the servo loops, and plans paths based on the strategic commands and vision information received from the PC connected to the overhead camera. The robots receive messages from the PC using 418/433 MHz RF modules. In addition the robots have IR reflective sensors that enable high bandwidth, short range measurement of the ball offset for kicking”



Learning Objectives

By the end of this lecture students will understand:

- What is evolutionary robotics?
 - What can be evolved?
 - What are the advantages of evolving robotic components?
- **What is embodied evolution?**
 - **What are the implications for computer models?**
 - **What are the implications for real systems**

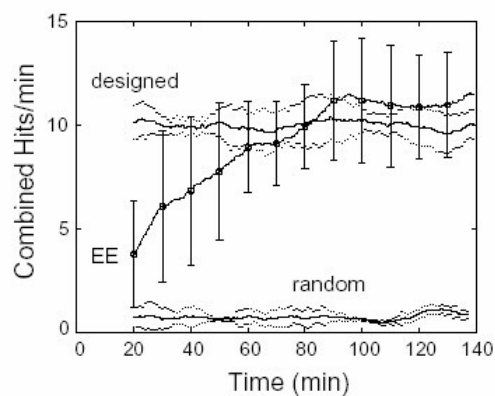
Embodied Evolution (EE)

- Evolution occurs within a population of robots
- Evaluation, selection and reproduction
- Evaluation
 - performed by each robot autonomously (e.g. maintaining power, number of objects collected)
- Reproduction
 - Cannot create new robots
 - Comes down to exchange of genetic information with other robots
- Selection
 - Each robot broadcasts a mutated version of one of its genes with a frequency proportional to its current power level, and resists ditto

Example task

- Pen with light in the middle; robot task is to reach light
- When robot reaches the light its energy is increased by a fixed amount
- When it broadcasts a gene it is decremented a set amount
- Broadcasts therefore decrease exponentially with time since last visit to light
- Behaviour controlled by evolving neural network
- Evolved robots outperform hand-crafted network
- Solutions are different

Results



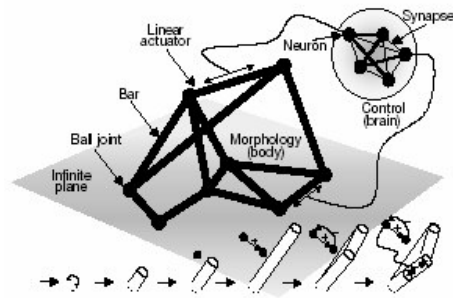
Why use EE?

- When the task domain cannot be simulated e.g. multiple agents in a complex environment computationally infeasible
- When a centralized coordinator for machine learning is not available; communications may be bottleneck
- When agents must learn in the field e.g. Mars
- When we do not have *a priori* knowledge of task decomposition - speciation

Evolving robots

- EC for design
- Additive fabrication for construction
- evolutionary process operates on a population of candidate robots, each composed of some repertoire of building blocks
- The evolutionary process iteratively selects fitter machines, creates offspring by adding, modifying and removing building blocks using a set of operators, and replaces them into the population
- design space consists of bars and actuators as building blocks of structure and artificial neurons as building blocks of control
- Aim: locomotion over the infinite plane

Evolving...



Embodiment

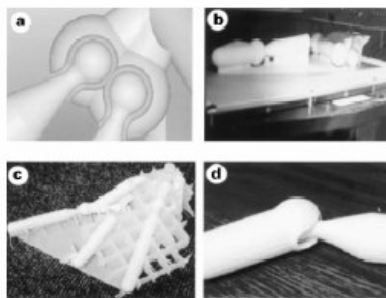
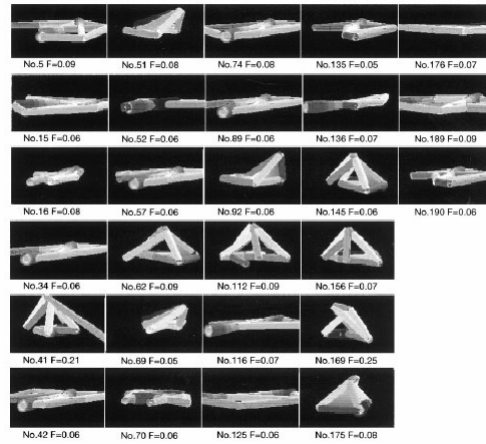
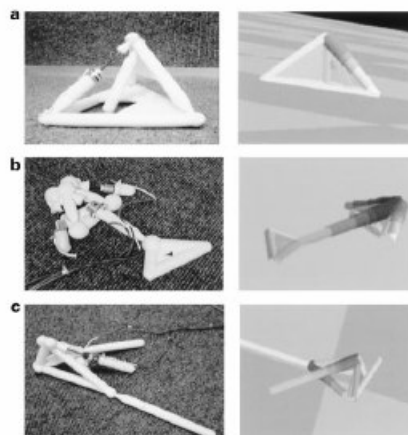


Figure 4 Physical embodiment process. **a**, Automatically "fleshed" joints in virtual space; **b**, a physical replication process in a rapid prototyping machine that builds the three-dimensional morphology layer after layer; **c**, pre-assembled body in mid print with discardable support structure; **d**, a close-up image of a joint printed as a single unit. The ball is printed inside the socket.

A generation of robots



Real robots



Summary

- EC can be used to develop solutions to a wide variety of problems
- Should be used wherever
 - A mathematical solution is infeasible
 - A “good enough” solution is acceptable
 - Computational time is less important than human time
- Important issues
 - Problem representation
 - Parameterization