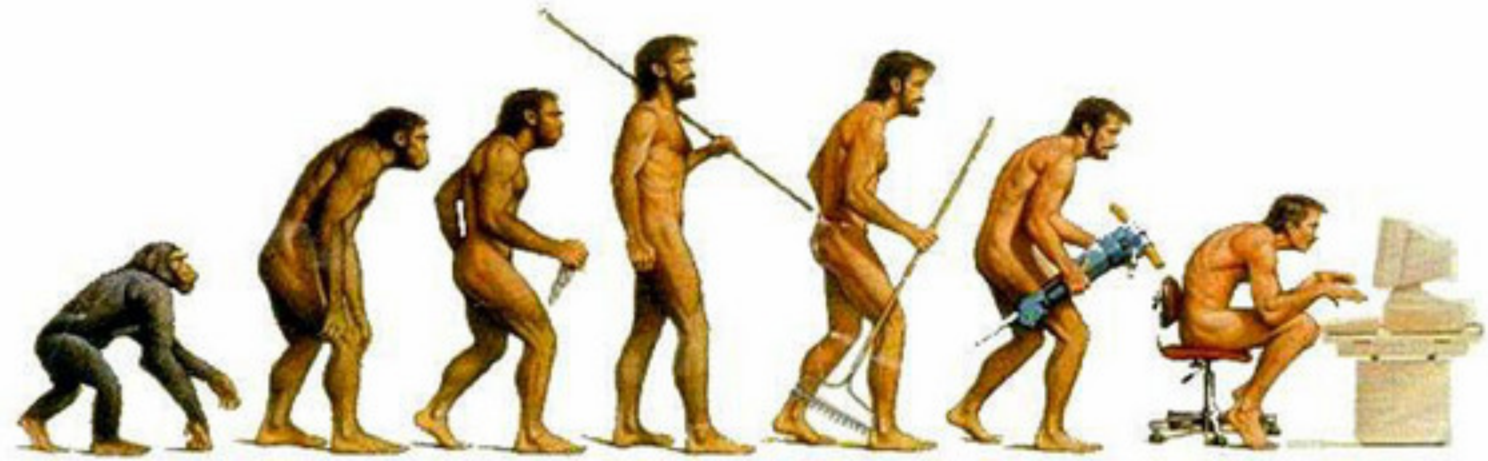
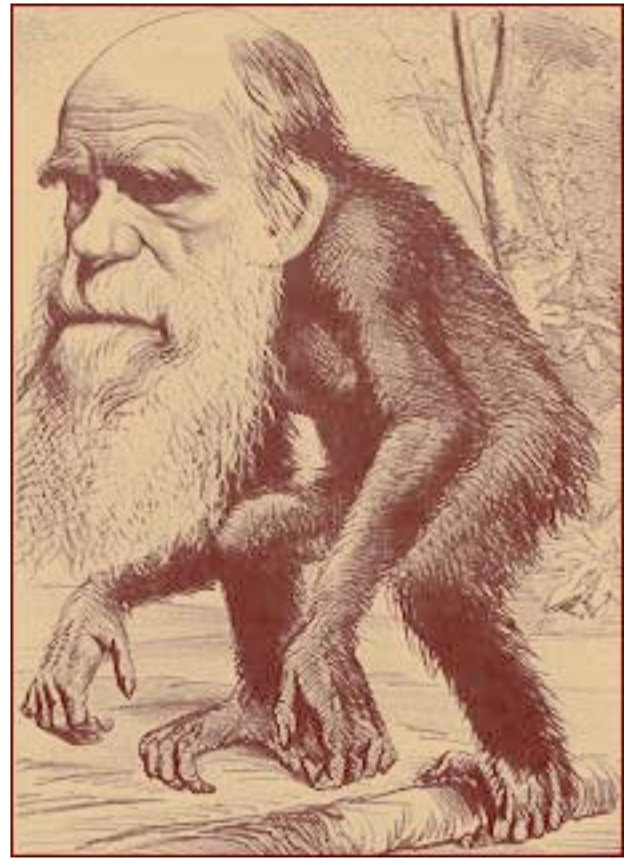
Evolutionary Computing



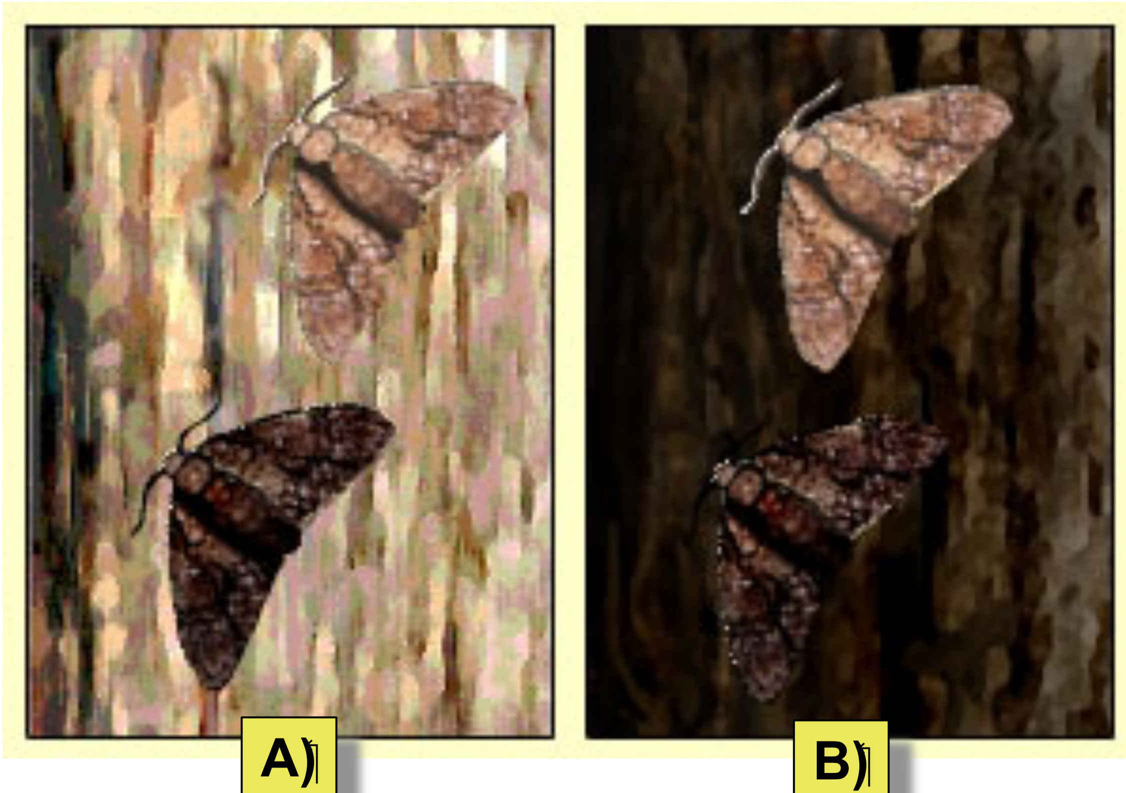
Darwinian Evolution

- A species consists of a number of individuals who live in an environment.
- The environment may include various pressures on the population.
  - *<u>External pressures:</u> Predators, harsh climate (cold, arid, ...).*
  - *<u>Internal pressures:</u> Intra-species competition for limited resources.*
- Individuals more suited to the environment have an increased chance of survival, and hence have more time to reproduce, and pass their adaptations to their children.



- **“Survival of the fittest”, “Diversity drives change”**

Example: The Peppered Moth

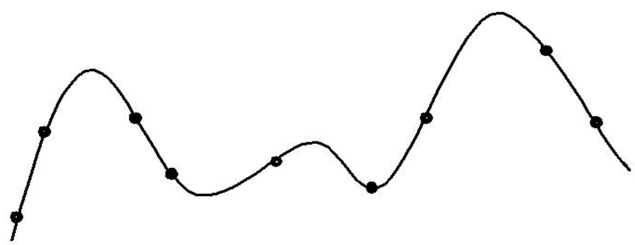


- The darker variety of the species was once an anomaly, at a disadvantage to the more prevalent pale variety (A).

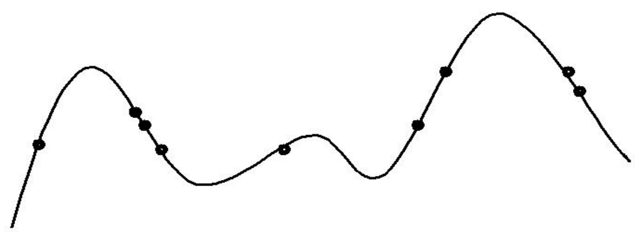
- With the Industrial Revolution in Britain in the 19th Century, its habitat (Birch tree) became darkened with pollution (B), and its mutation became an asset for camouflage and therefore survival.

Typical Behaviour of Evolution

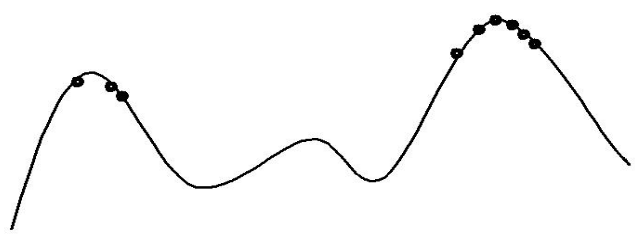
- Phases in optimising a 1-dimensional fitness landscape:



- <u>Early Phase</u>:
  Quasi-random population distribution



- <u>Mid-Phase</u>:
  Population around/on hills



- <u>Late-Phase</u>:
  Population concentrated high on hills

Evolutionary Computing

- Biologically inspired, like neural networks.
  - Turing, 1948: “genetic or evolutionary search”.
  - Bremermann, 1962: “[optimisation] through evolution and recombination”.
  - Holland, 1975: Introduces Genetic Algorithms
  - Koza, 1992: Introduces Genetic Programming

- Explore complex space where analytical solutions are not available:
  - Discrete optimisation
  - Discrete NP-hard problems (e.g., traveling salesman, job scheduling, etc.)

- Attributed features:
  - Not too fast, nor always optimal.
  - Good heuristic for combinatorial problems.
  - Many variants, e.g., reproduction models, operators.

Genetic Algorithm (GA)

- evaluation (fitness function) of individuals
- selection (of parents)
- reproduction (crossover)
- mutation

Caveat

*"GAs do not find the **best** solution,*
*they find the **first** solution"*

Representation of Population

- Basic idea: To find a solution to a problem, throw a bunch of *potential* solutions at it, and create new guesses based on the best of those attempts.

- A population, π, is a set of individuals, π_i, for $i = 1..N$.
  - Each individual is a candidate solution.

- Individuals are encoded by genes Γ_j, for $j = 1..G$.
  - Every possible solution can be encoded.

- Ex. Find x that maximizes

$$\frac{\delta\left(\frac{1}{1+e^{0.8x-0.2}}+2^x \sinh\left(\sqrt{2x^3}+20\right)\right)}{\delta x}$$

  - Phenotype (an individual) x = 412895
    Genotype (its genetic code (G=20)) = 01100100110011011111

Fitness Function

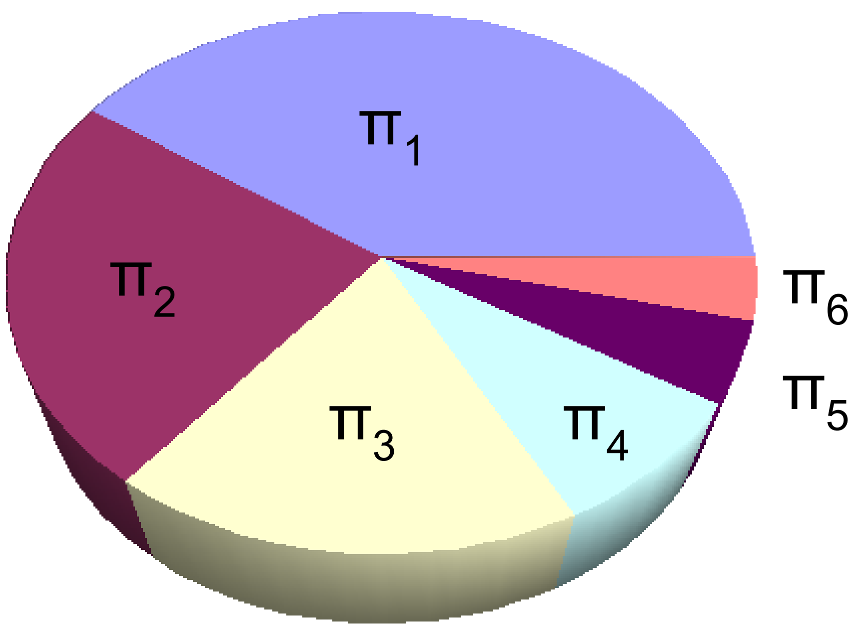
- All members of a population are evaluated for 'fitness' according to some function, $\Phi: \pi_i \rightarrow \mathbb{R}$.
  - Φ is often just the function we're trying to maximise.
  - Otherwise, it can be a heuristic approximator to the problem.
  - Theoretically, Φ expresses how well an individual is adapted to an environment.

Selection

- Giving the most-fit agents an increased chance of reproduction yields a stronger population in subsequent generations.
- At each iteration of the algorithm, we select and combine agents based on Φ, producing a new generation of *N* agents.
- Reproduction:
  - usually (but not always) involves 2 distinct parents
  - usually fatal to the parents (they do not survive to the next iteration)
  - exception: **Elitism**: The strongest *n* individuals survive to the next generation

Selection by Roulette Wheel

- For $|\pi| = N$ individuals, π_i has probability $\dfrac{\phi(\pi_i)}{\sum \phi(\pi_j)}$ of being selected



- Ex:

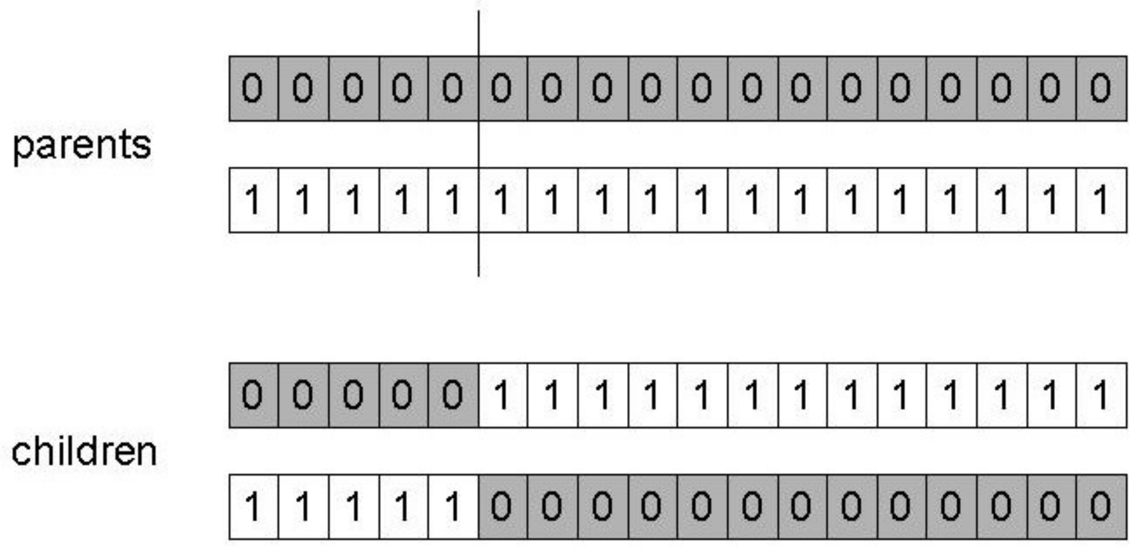
$\Phi(\pi_1) = 103$
$\Phi(\pi_2) = 61$
$\Phi(\pi_3) = 48$
$\Phi(\pi_4) = 22$
$\Phi(\pi_5) = 13$
$\Phi(\pi_6) = 9$

- TOTAL FITNESS = 256

π_4 has a $22/256 \approx 8.6\%$ chance of being selected

Crossover

- Combine chunks of parental DNA to produce the new infant.
- For species with G genes, Γ_g , and 1-point crossover:
  - Randomly select $0 < j < G$ as the crossover point
  - Split parent genotypes at j.
  - Create children by crossing tails over around Γ_j:



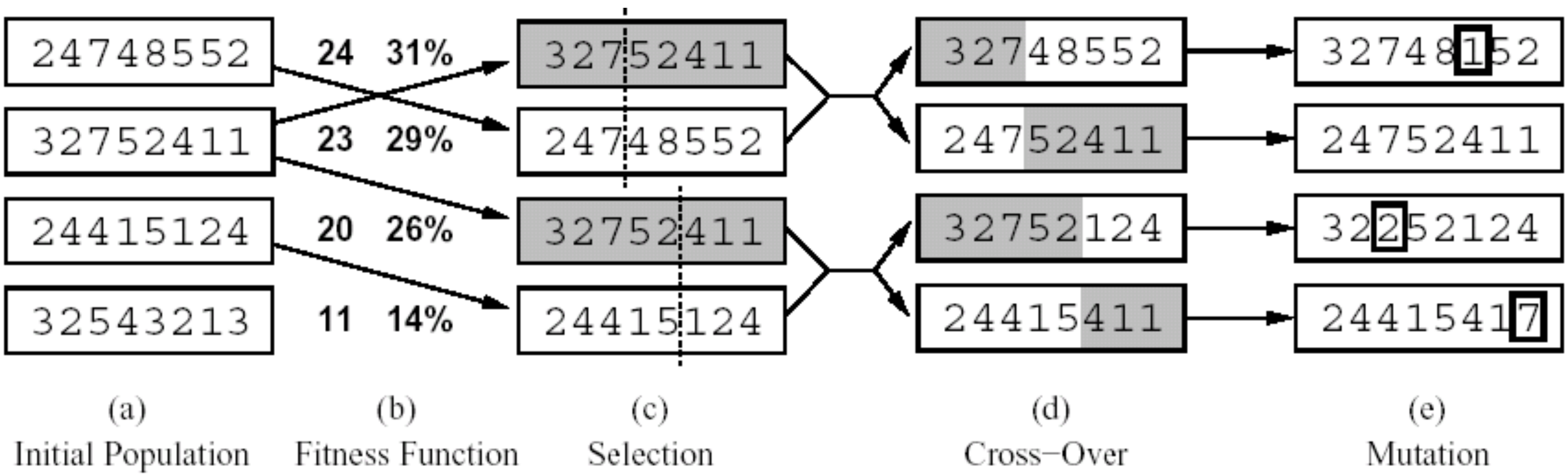
Mutation

- For each new child, alter each gene, Γ_g, with mutation rate, p_m.
  - p_m usually small, and proportional to (1 / | π |) or (1 / G).
- For genotypes of bitstrings, this involves merely switching the bit
  - $\Gamma_g = (\Gamma_g + 1) \textbf{ mod } 2$
- This process drives change and diversity, and helps to avoid local maxima in the process.

Original offspring 1 1101111000011110
Original offspring 2 0101000100110110

Mutated offspring 1 1100111000011110
Mutated offspring 2 0101001100110100

GA Summary


24748552
32752411
24415124
32543213
24 31%
23 29%
20 26%
11 14%
32752411
24748552
32752411
24415124
32748552
24752411
32752124
24415411
32748152
24752411
32252124
24415417
(a)
Initial Population
(b)
Fitness Function
(c)
Selection
(d)
Cross−Over
(e)
Mutation

A Genetic Algorithm(*)

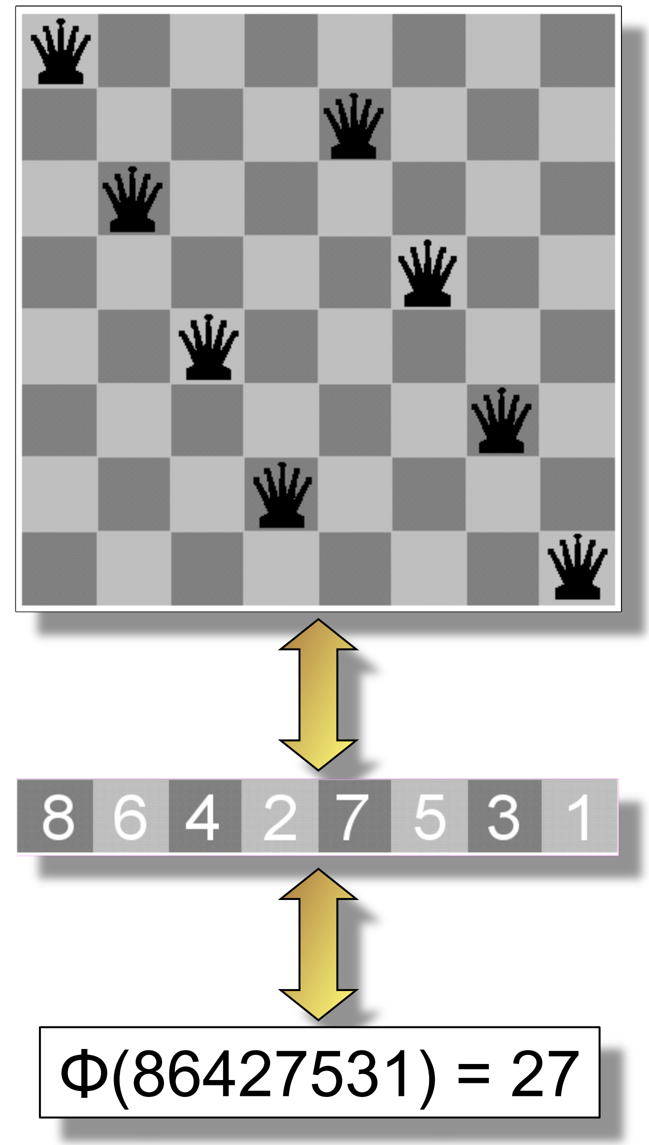
```
function Genetic-Algorithm( π, Φ ) returns individual σ
    inputs: π; a population of N individual solutions
            Φ; a fitness-function for individuals

    repeat
        π ' ← ∅
        for i from 1 to (N / 2) do
            (x, y) ← Random-Select-Parents( π, Φ )
            (w, z)  ← Reproduce( x, y )
            if (small random probability) then (w, z) ← Mutate(w, z) )
            add (w, z) to π '
        π  ← π '
        σ ← Best( π, Φ )
    until Φ( σ ) > some threshold, or enough time has elapsed
    return σ
```

(*) : There are many variations. This is Simplified GA (SGA), popularised by (Holland, 73)

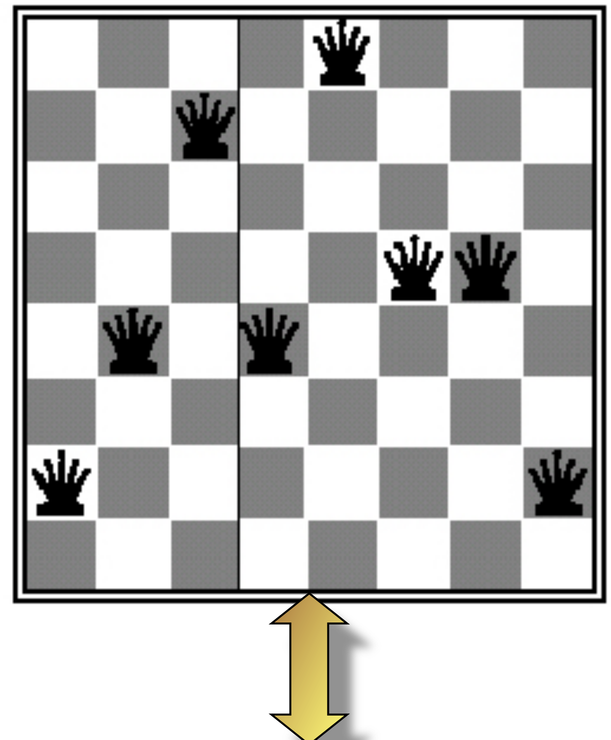
Example: 8 Queens

- Representation:
  - 8 genes, where $\Gamma_i = j \leftrightarrow (i,j)$ has a queen.
  - $8^8 \approx 17$ million *possible* individuals.
- Fitness Function:
  - Number of non-attacking pairs.
  - Best fitness: $\binom{8}{2} = \frac{8!}{2!(8-2)!} = 28$



8 Queens: Initialisation

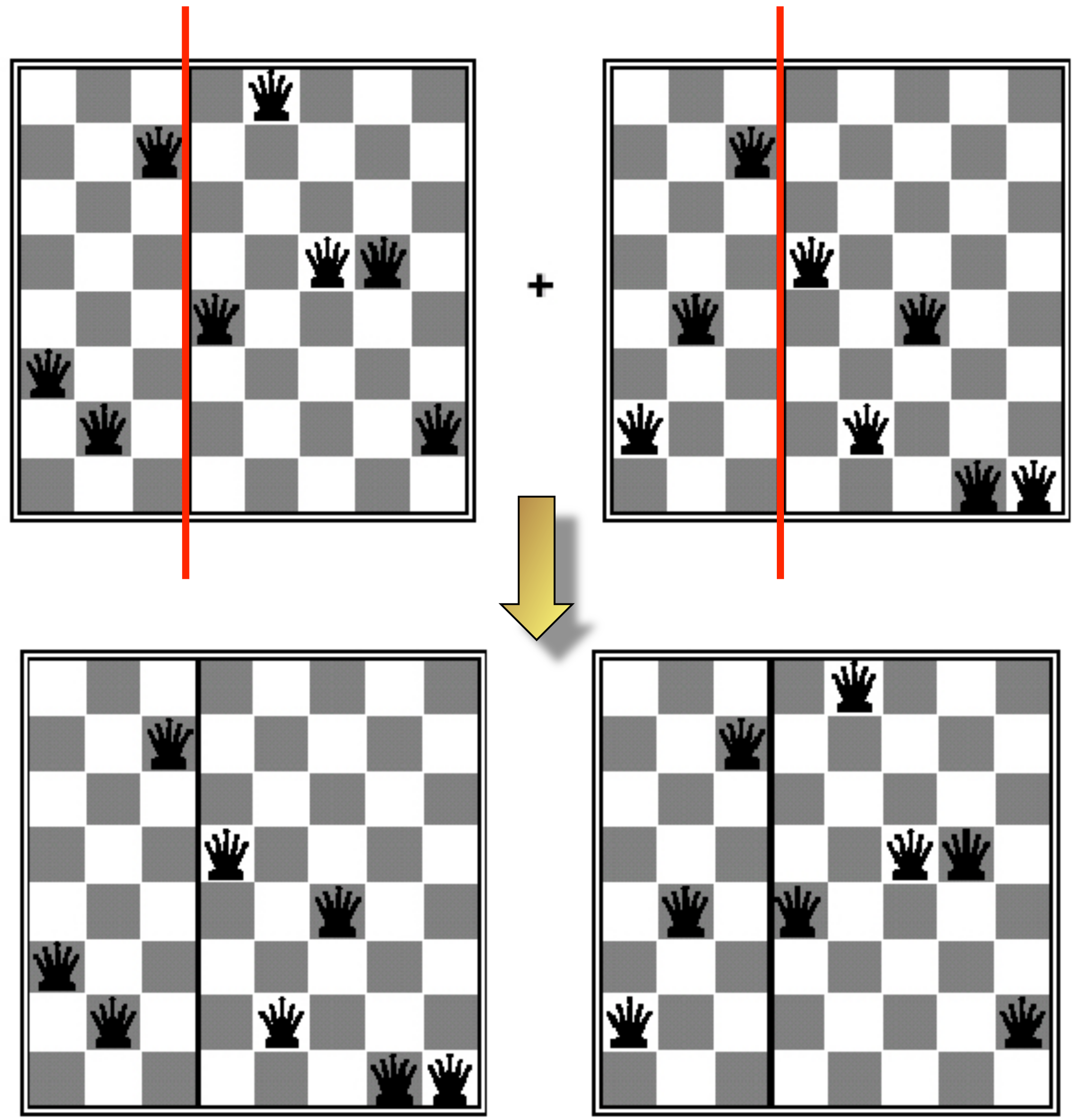
- <u>Initialise population</u>:
  - π_1 = 24748552, π_2 = 32752411,
    π_3 = 24415124, π_4 = 32543213.
- <u>Determine fitness</u>:
  - $\Phi(\pi_1) = 24$, $\Phi(\pi_2) = 23$, $\Phi(\pi_3) = 20$, $\Phi(\pi_4) = 11$.
  - Total fitness = 24+23+20+11 = 78
- <u>Selection</u>:
  - Assuming "Roulette Wheel" process...
  - $P(\pi_1) = 24/78 = 0.31 \rightarrow$ Range: [0, .31)
    $P(\pi_2) = 23/78 = 0.29 \rightarrow$ Range: [.31, .61)
    $P(\pi_3) = 20/78 = 0.26 \rightarrow$ Range: [.61, .87)
    $P(\pi_4) = 11/78 = 0.13 \rightarrow$ Range: [.87, 1.0)



$\Phi(24748552) = 24$

8 Queens: Reproduction

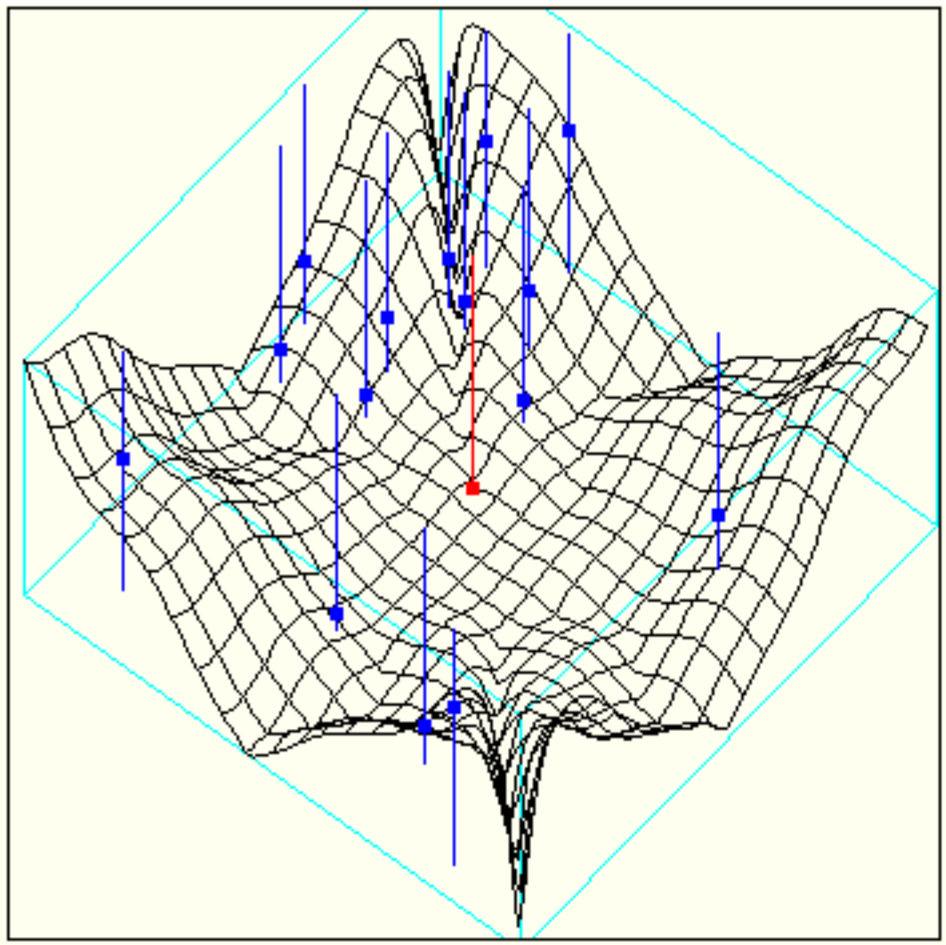
- **Select Parents**:
  - Given
    $(\pi_1) \rightarrow [0, .31)$
    $(\pi_2) \rightarrow [.31, .61)$
    $(\pi_3) \rightarrow [.61, .87)$
    $(\pi_4) \rightarrow [.87, 1.0)$
  - Randomly draw: 0.4012, 0.1486, 0.5973 and 0.8129.
  - Parents: $(\pi_2 + \pi_1)$ and $(\pi_2 + \pi_3)$ paired off randomly.
- **Crossover**:
  - Single random crossover point.
  - ex. `Crossover`$(\pi_2, \pi_1) = 3$



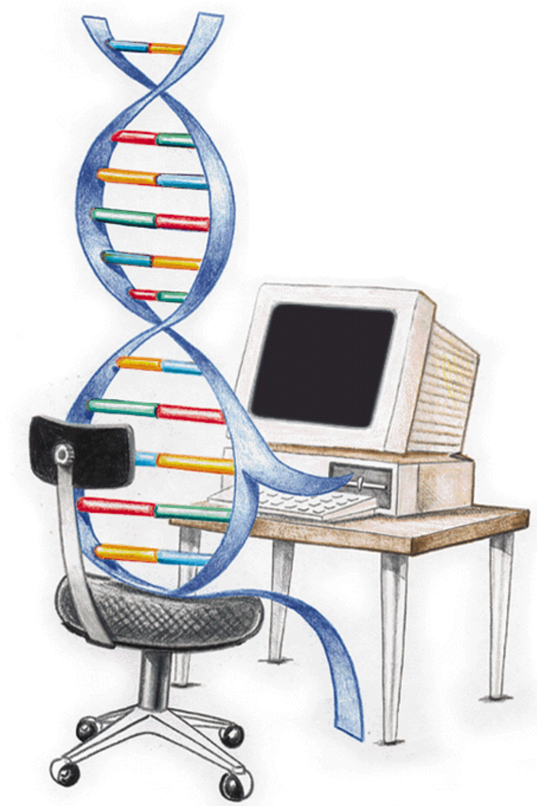
Example: Optimisation

- Find the values for *x and y* that minimize

$$f(x,y) = x\cos(x) + x\,y\tanh(x\,y) + y\sin(y)$$



Part II: Genetic Programming

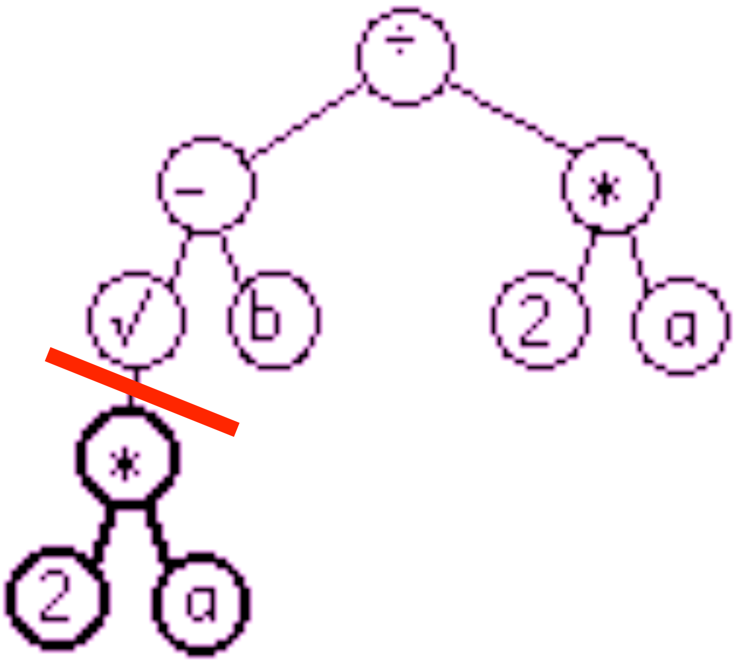


Intro: λ-Calculus and LISP

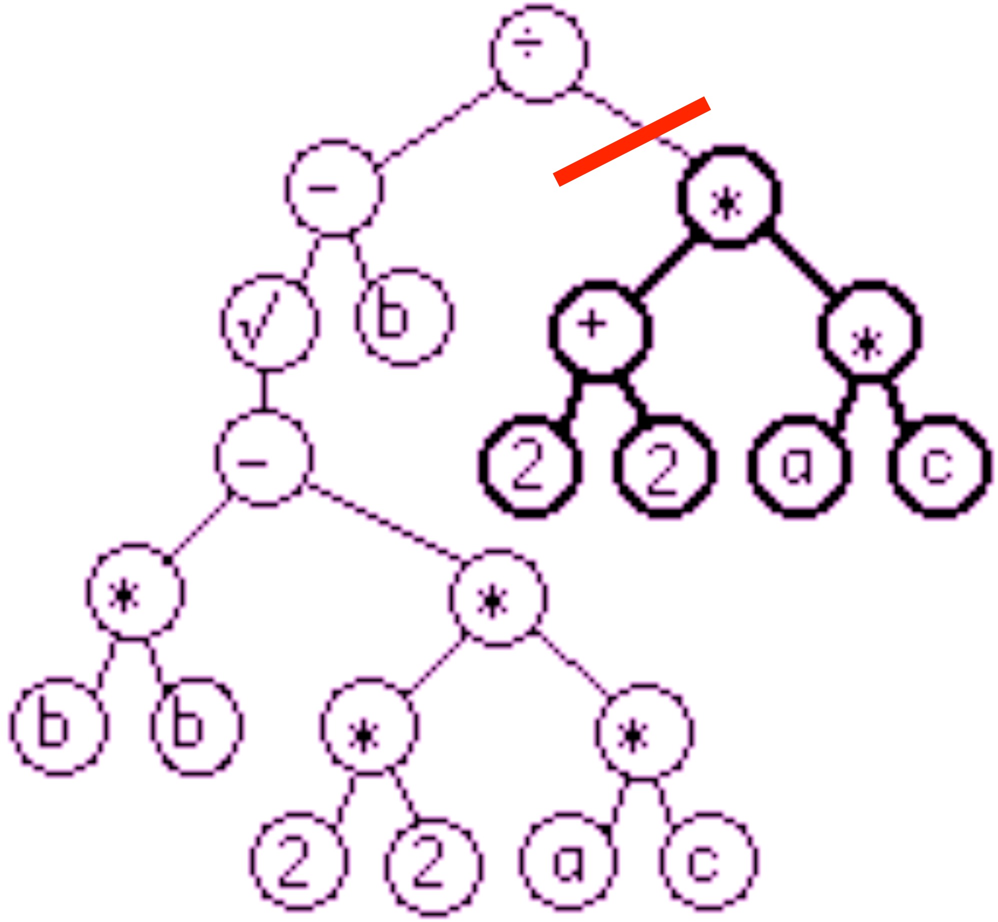
- λ-Calculus (Church & Kleene, 1936
  - formal system describing all computable functions
  - ***Functions and data are interchangeable.***
- LISP: *functional* programming language implementing λ-Calculus
  - Introduced by John McCarthy in an MIT memo in 1958.
  - S-expressions represent program code in LISP:
    (<operator> <param1> <param2> ...)

```
(defun factorial (n)
  (if (<= n 1)
    1
    (* n (factorial (- n 1)))))
```

Why λ-Calculus for EC?




```
(÷  (–  (√  (*  2  a))  b)
    (*  2  a  ))
```

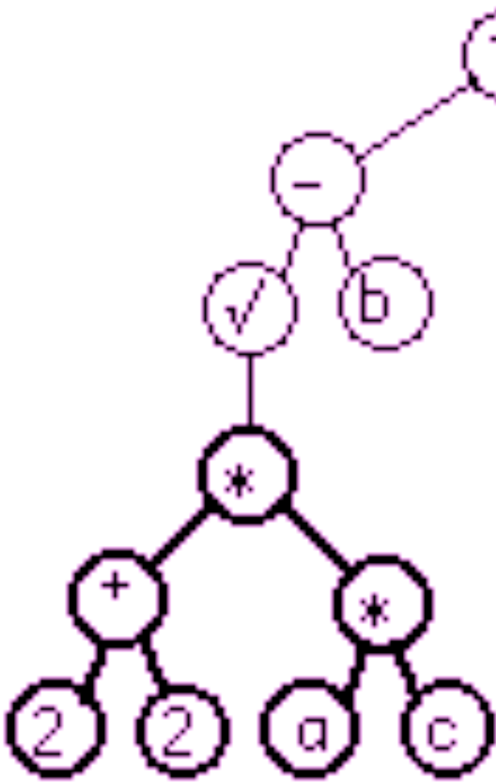


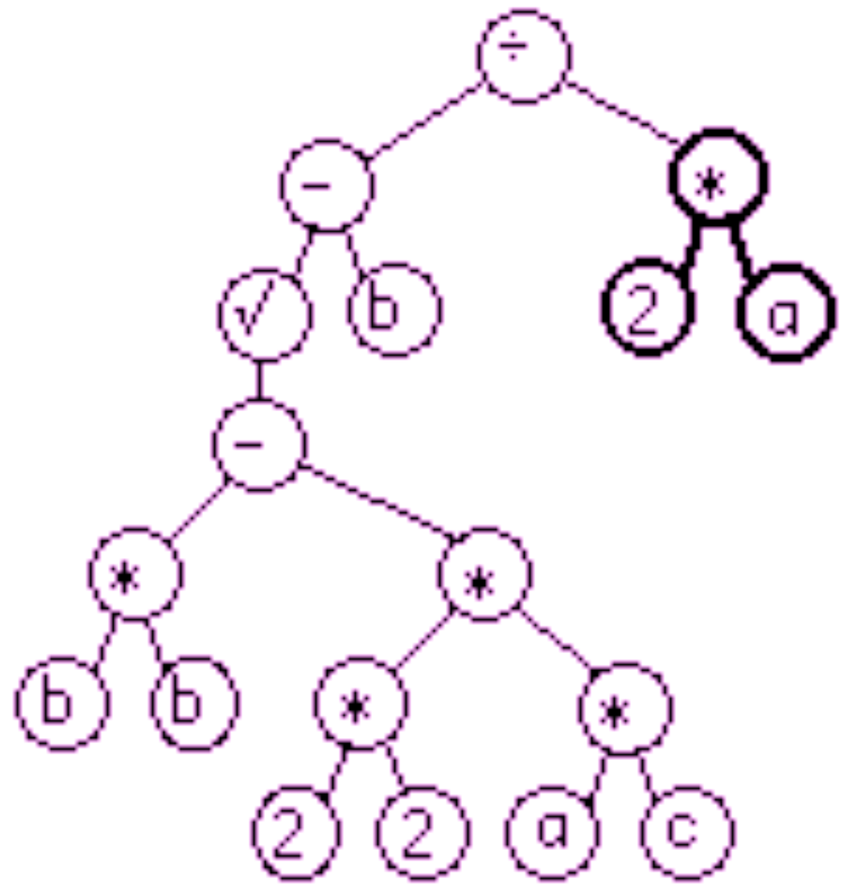
$$\frac{\sqrt{2\mathrm{a}}-b}{2\mathrm{a}}$$

$$\frac{\sqrt{b^2-2^2ac}-b}{(2+2)ac}$$

Why λ-Calculus for EC?

- Individuals are effectively program trees that can be arbitrarily clipped and recombined, and evaluated to directly infer fitness.
- Child nodes from the parents in the previous slide:



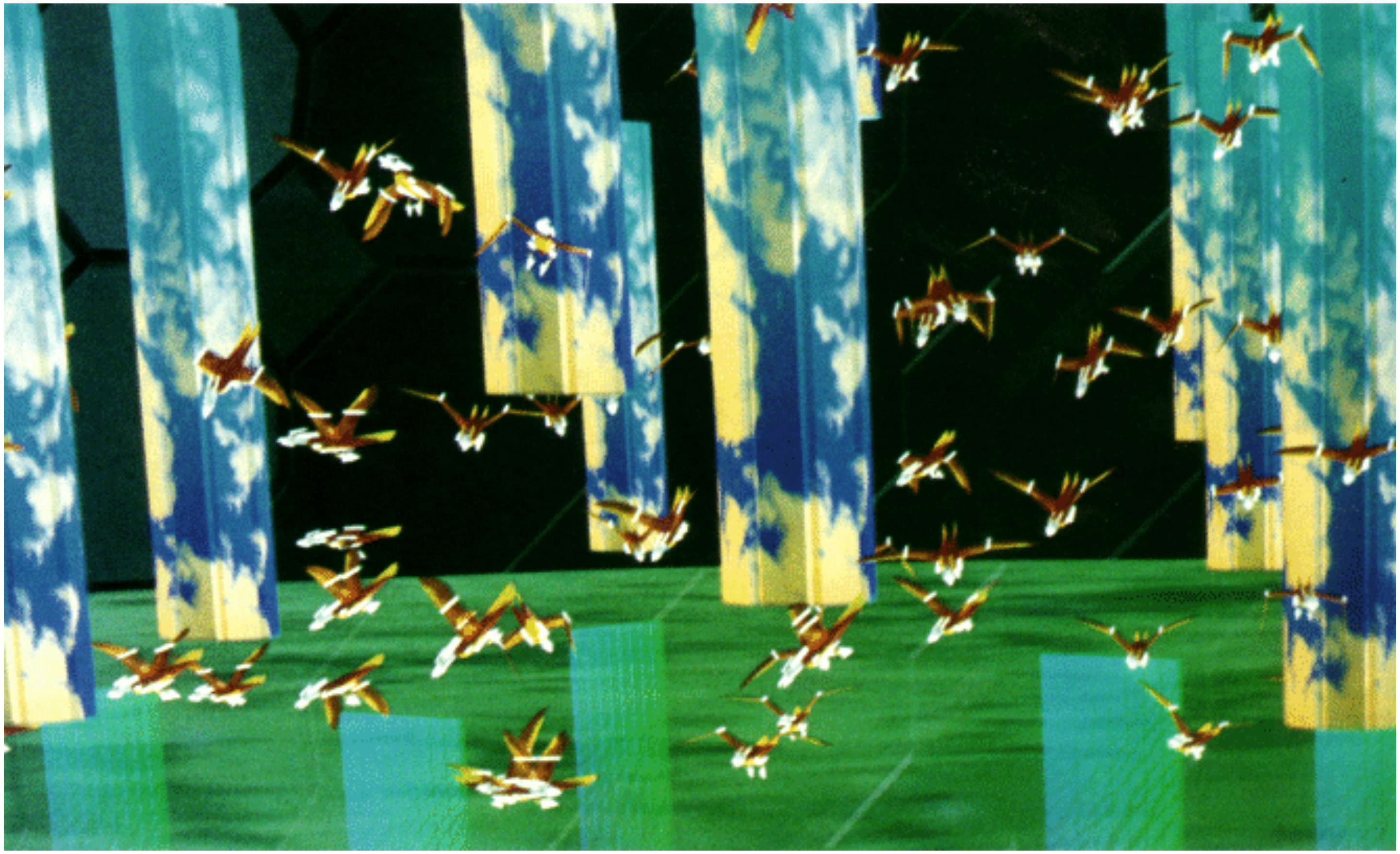


Artificial Life

- **Definition:** The process of creating synthetic biology on computers to study, simulate and understand living systems.

Boids
[Reynolds 87]

- Bat swarms and penguin flocks from "Batman Returns"



Go Fish
[Tu 94]

- Physically based animation combined with simple behavioural models

https://www.youtube.com/watch?v=aP1_XkCdAaE
https://www.youtube.com/watch?v=VpZ93n5QQuQ

Cognitive Modeling
[Funge 99]

- Language for describing character’s intentions, abilities, and reasoning process
- Allows for automated animation

Philip K. Dick Android
[Hanson, 2005]

- Putting together robotics, NLU, and the Turing Test