Lecture 06 - Genetic Algorithms

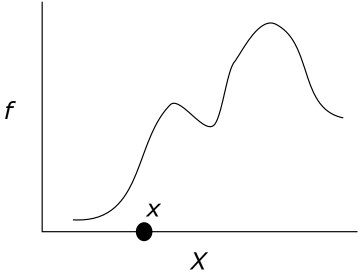
Optimisation CT5141

James McDermott

University of Galway

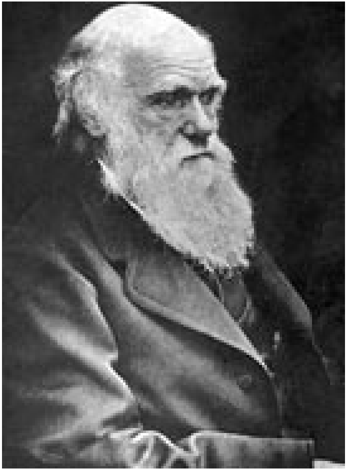
Overview

Reminder: How to escape local optima



1. Allow **larger jumps** in the `nbr` function
2. **Restart** (**iterated hill-climbing**)
3. Allow **disimproving moves** (**simulated annealing** and **late-acceptance hill-climbing**)
4. Use **multiple search points** with **information transfer** between them (**genetic algorithm** or GA).

Darwinian evolution

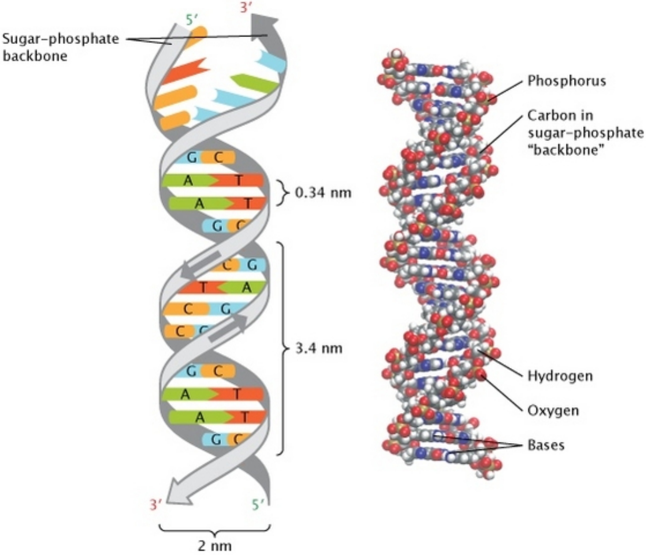


"If I were to give an award for the single best idea anyone ever had, I'd give it to Darwin." – Dennett (1995) "Darwin's Dangerous Idea."

Source: CNN

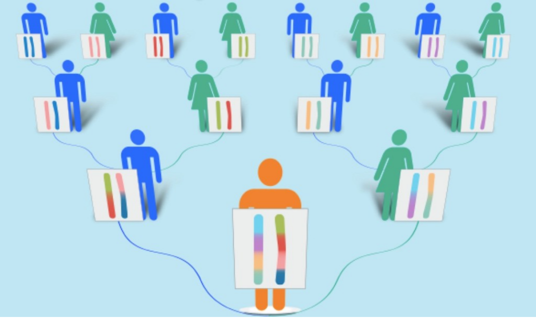
- Population of organisms
- Competition within and between species
- Only the fittest survive and reproduce
- Reproduction: mating and combination of genes
- Mutation of genes
- Inheritance (but no inheritance of acquired traits)

Genes



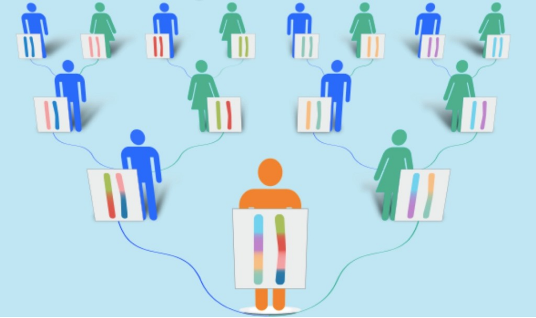
Nature Education 2013

Inheritance and mixing of genes



From regenerationnet

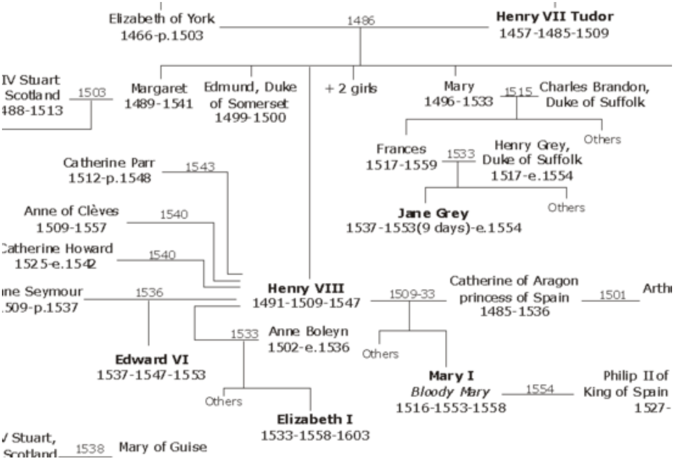
Inheritance and mixing of genes



From regenerationnet

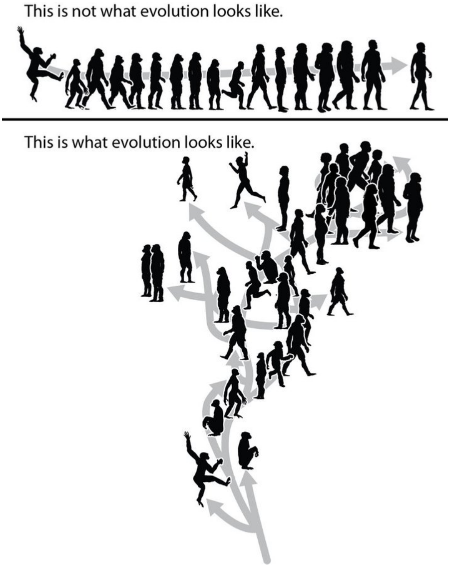
How would this tree look if we went back another 100 generations?

Inheritance and mixing of genes

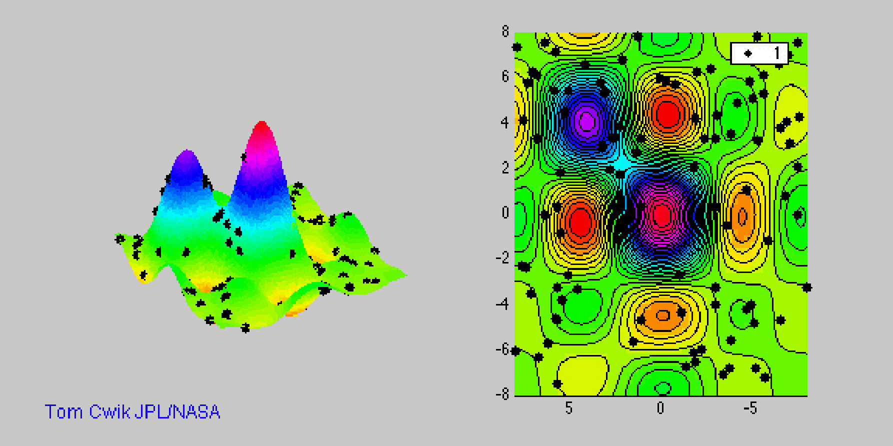


Evolution is not a linear progression

From the perspective of a species evolving/speciating over time:



Population change over time



Evolution = optimisation?

In nature, there is no explicit objective function! No-one is trying to optimise anything.

But there is an **implicit** objective function – ability to survive, compete, and reproduce.

This has the **effect** of optimisation – cheetahs gradually get faster.

Evolutionary algorithms

Evolutionary algorithms are a **family** of black-box metaheuristic optimisation algorithms **inspired by** the main ideas of Darwinian evolution.

Evolutionary algorithms

Evolutionary algorithms are a **family** of black-box metaheuristic optimisation algorithms **inspired by** the main ideas of Darwinian evolution.

- There is always a **population** – not just one current point
- There is crossover/recombination – an operator that takes in **two** parent solutions and outputs one or two **offspring**.
- There are **not** two sexes in the population – any individual can be recombined with any other.

Terminology

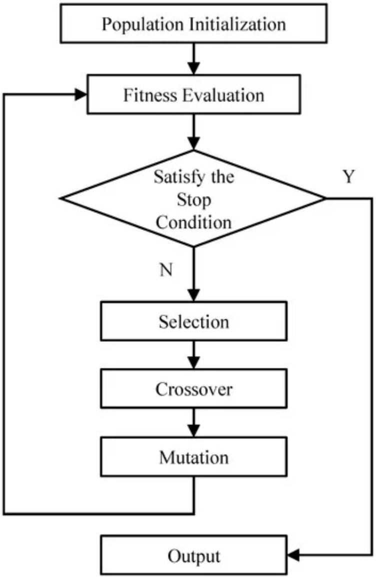
Kirkpatrick

- **Evolutionary Algorithms** or **Evolutionary Computation** is an umbrella term
- The **Genetic Algorithm** (GA) is the central algorithm
- But really, the GA is a huge **family** of variant algorithms
- Sometimes with stupid names like **Intelligent Water-Drop Algorithm**

Terminology

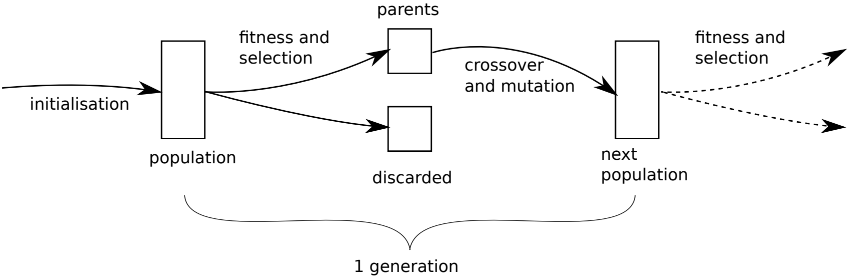
Other optimisation	Evolutionary
Decision variable	Gene
Point/Solution	Genotype/Individual
Multiple points	Population
Objective	Fitness
Neighbour	Mutation
Combination	Crossover
Iteration	Generation

GA flowchart



From Xiang et al

GA loop


parents
fitness and
selection
initialisation
population
discarded
crossover
and mutation
next
population
fitness and
selection
1 generation

Overview

Operators

GAs use a few main **operators** (functions):

- Genetic operators
  - Initialisation
  - Mutation
  - Crossover
- Selection
- (Elitism)

Operators

GAs use a few main **operators** (functions):

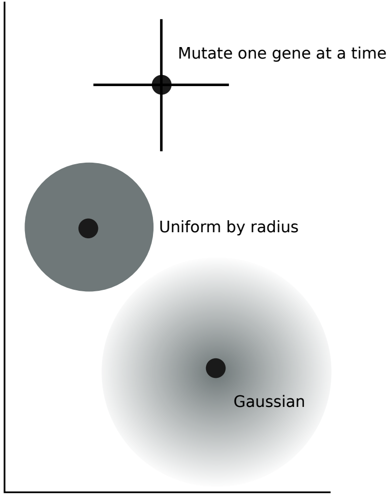
- Genetic operators
  - Initialisation
  - Mutation
  - Crossover
- Selection
- (Elitism)

For each of these, there are many possible ways to design and implement the operator.

Initialisation

- We create a population of *n* individuals
- Usually **uniformly distributed** in the search space
- `[init() for i in range(n)]`

Mutation



- **Mutation = neighbour**
- May be many possible mutation functions for a given search space, with different properties
- E.g. some more explorative, some more exploitative
- E.g. Gaussian mutation versus uniform mutation
- E.g. mutate one gene chosen randomly, versus mutate all genes with a certain low probability, versus mutate all genes with a small step-size.

Crossover

Crossover is the means of **information transfer** between individuals.

Crossover (and use of a population) are the main features that distinguishes a GA from a HC method.

Crossover is usually seen as **more important** than mutation. Mutation may be omitted or applied to a small proportion of the population.

Crossover

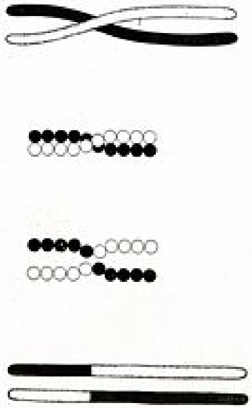


Fig. 64. Scheme to illustrate a method of crossing over of the chromosomes.

Thomas Hunt Morgan, 1916, taken from Wiki

- Each individual's **genome** is a **vector** of DVs.

Crossover

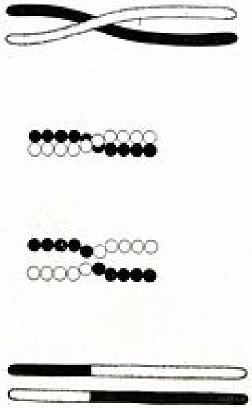
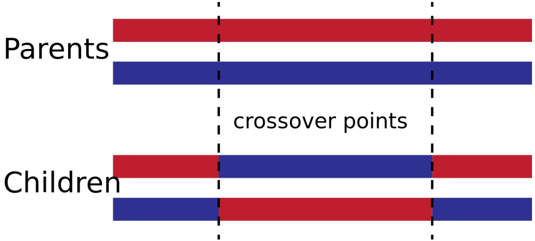


Fig. 64. Scheme to illustrate a method of crossing over of the chromosomes.

Thomas Hunt Morgan, 1916, taken from Wiki

- Each individual's **genome** is a **vector** of DVs.
- The scheme shown here is called **one-point** crossover because we choose one (random) split-point.

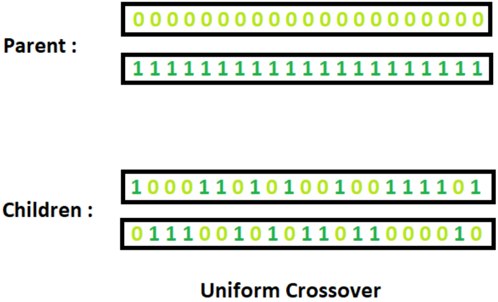
Two-point crossover



This is **two-point** because we choose two split-points at random locations as shown.

Uniform crossover

Every position gets a value chosen from the same position in one parent or the other



Geeks for geeks

Uniform crossover

```
def uniform_crossover(x, y):
    c, d = [], []
    for xi, yi in zip(x, y):
        if random.random() < 0.5:
            c.append(xi); d.append(yi)
        else:
            c.append(yi); d.append(xi)
    return c, d
```

Uniform crossover

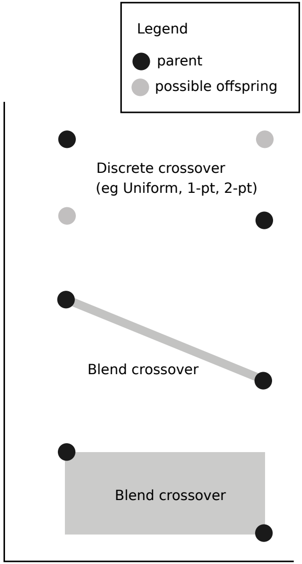
```
def uniform_crossover(x, y):
    c, d = [], []
    for xi, yi in zip(x, y):
        if random.random() < 0.5:
            c.append(xi); d.append(yi)
        else:
            c.append(yi); d.append(xi)
    return c, d
```

You can easily implement any of the crossover operators we have seen.

Crossover offspring

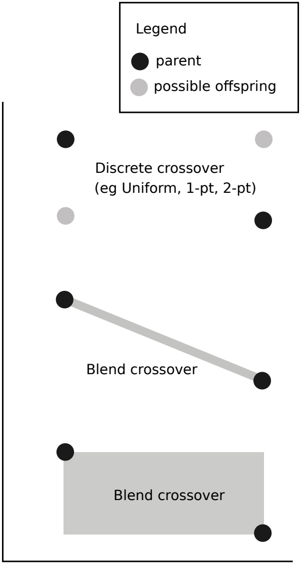
Some operators are defined to return just one offspring; some return two.

Crossover: where do offspring go?



- Offspring are usually somehow **intermediate** to their parents.

Crossover: where do offspring go?

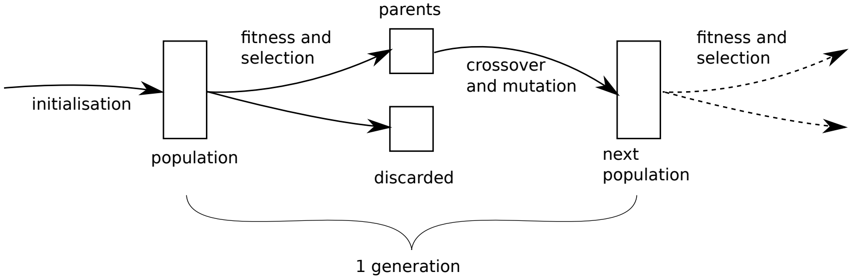


- Offspring are usually somehow **intermediate** to their parents.
- Implication: crossover can only search within the **convex hull** of the population. A little mutation is needed to go outside it.

Genetic operators' desirable properties

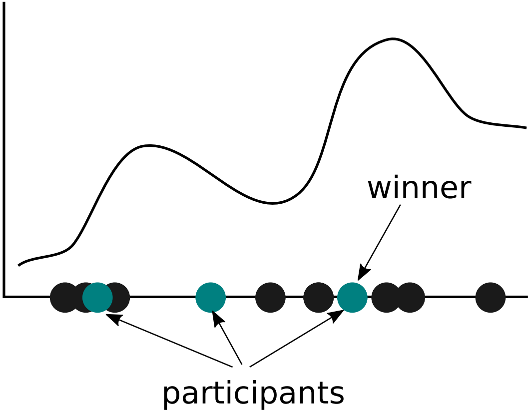
- Initialisation:
  - Uniform on the space
- Mutation:
  - Mutated genotype is close to original
  - **Ergodic**: capable of eventually reaching any point
  - Undirected
- Crossover:
  - Offspring are intermediate to parents
  - Undirected

GA loop


initialisation
population
fitness and
selection
parents
discarded
crossover
and mutation
next
population
fitness and
selection
1 generation

Tournament selection

```
def tournament_select(pop, size):
    return max(random.sample(pop, size), key=f)
```



Tournament selection

Notice that the winner is not the very best individual in the population! (But maybe it will win another tournament later in the same generation.) We don't want the very best individual to win always – it would be too **exploitative** and would cause the algorithm to **converge** immediately.

Tournament selection

Notice that the winner is not the very best individual in the population! (But maybe it will win another tournament later in the same generation.) We don't want the very best individual to win always – it would be too **exploitative** and would cause the algorithm to **converge** immediately.

Larger tournament sizes give a **higher selection pressure**, thus make the algorithm **more exploitative**.

Tournament selection

Notice that the winner is not the very best individual in the population! (But maybe it will win another tournament later in the same generation.) We don't want the very best individual to win always – it would be too **exploitative** and would cause the algorithm to **converge** immediately.

Larger tournament sizes give a **higher selection pressure**, thus make the algorithm **more exploitative**.

What would happen if the tournament size equalled the population size?

Tournament selection

Tournament selection is nice because it is robust and tunable (exploration versus exploitation). Some other methods are less robust, e.g. the **roulette wheel** can be too exploitative early on and too explorative later in the search.

Why do GAs work?

"Consider everything. Keep the good. Avoid evil whenever you notice it." (1 Thess. 5:21-22)

(quoted on https://www.mat.univie.ac.at/~neum/glopt.html)

Why do GAs work?

"Consider everything. Keep the good. Avoid evil whenever you notice it." (1 Thess. 5:21-22)

(quoted on https://www.mat.univie.ac.at/~neum/glopt.html)

Just like with hill-climbing, the **genetic operators** (mutation and crossover) are **undirected**, unaffected by fitness. It is **selection** and the **ratchet** which "drive" evolution.

Why do GAs work?

"Consider everything. Keep the good. Avoid evil whenever you notice it." (1 Thess. 5:21-22)

(quoted on https://www.mat.univie.ac.at/~neum/glopt.html)

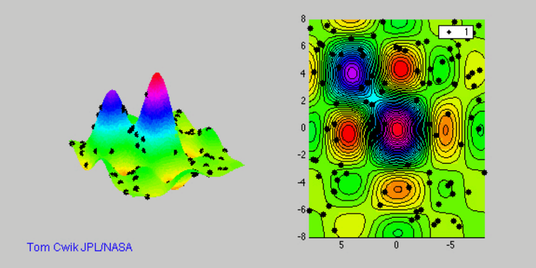
Just like with hill-climbing, the **genetic operators** (mutation and crossover) are **undirected**, unaffected by fitness. It is **selection** and the **ratchet** which "drive" evolution.

Recall the **ratchet** in hill-climbing. Can you see how to rewrite hill-climbing using **selection** in place of the `if`-statement?

Elitism

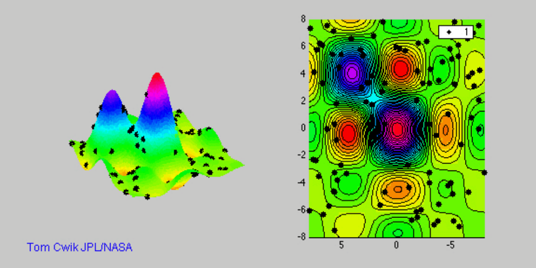
- In hill-climbing, we allow disimproving moves to help escape local optima
- In the GA, we don't have to discard our best individual to escape
- No real need to allow a disimprovement in the **best** objective value from one generation to the next
- **Elitism** means: we copy the best individual in the population directly to the next generation
- (sometimes more than one).

Convergence



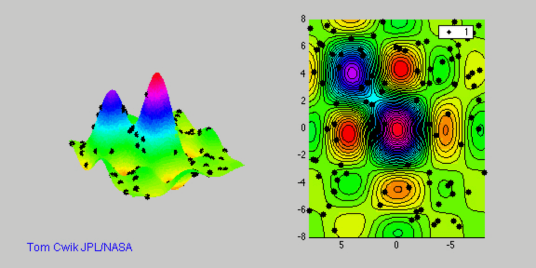
- We can measure genetic diversity in the population

Convergence



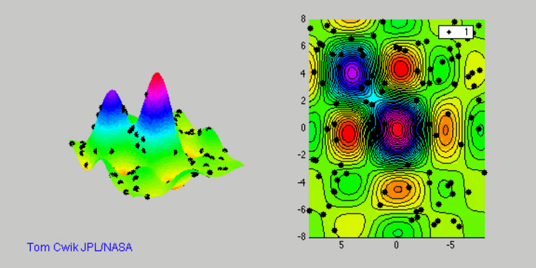
- We can measure genetic diversity in the population
- Or measure variance of objective values in the population

Convergence



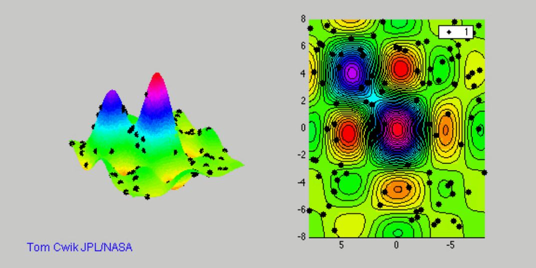
- We can measure genetic diversity in the population
- Or measure variance of objective values in the population
- Diversity **decreases over time**

Convergence



- We can measure genetic diversity in the population
- Or measure variance of objective values in the population
- Diversity **decreases over time**
- When the population has **converged**, there will be no more improvement…

Convergence



- We can measure genetic diversity in the population
- Or measure variance of objective values in the population
- Diversity **decreases over time**
- When the population has **converged**, there will be no more improvement…
- … because offspring will be almost identical to parents.

GA assumptions

GAs make the same sorts of assumptions as smart hill-climbing:

- the objective may be a bit rugged, but not totally uninformative or deceptive;

GA assumptions

GAs make the same sorts of assumptions as smart hill-climbing:

- the objective may be a bit rugged, but not totally uninformative or deceptive;
- we don't know the gradient;

GA assumptions

GAs make the same sorts of assumptions as smart hill-climbing:

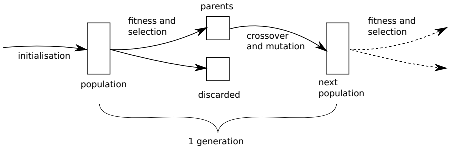
- the objective may be a bit rugged, but not totally uninformative or deceptive;
- we don't know the gradient;
- neighbours (created by mutation) usually have similar fitness;

GA assumptions

GAs make the same sorts of assumptions as smart hill-climbing:

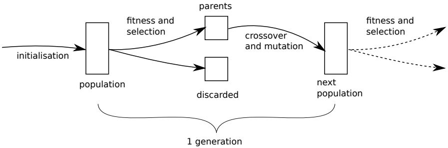
- the objective may be a bit rugged, but not totally uninformative or deceptive;
- we don’t know the gradient;
- neighbours (created by mutation) usually have similar fitness;
- children (created by crossover) usually have fitness similar to parents.

GA: alternative loops



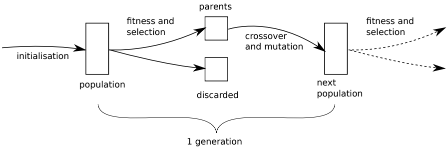
1. E.g. apply crossover to produce some individuals, and mutation to produce others

GA: alternative loops



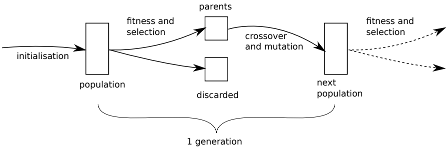
1. E.g. apply crossover to produce some individuals, and mutation to produce others
2. E.g. use entire population to produce offspring, then run fitness/selection to **discard from** (population + offspring)

GA: alternative loops



1. E.g. apply crossover to produce some individuals, and mutation to produce others
2. E.g. use entire population to produce offspring, then run fitness/selection to **discard from** (population + offspring)
3. E.g. "steady-state GA" where we just produce 1 or 2 offspring at a time from any parents, and use fitness/selection to decide which individuals to **discard**. No generations.

GA: alternative loops



1. E.g. apply crossover to produce some individuals, and mutation to produce others
2. E.g. use entire population to produce offspring, then run fitness/selection to **discard from** (population + offspring)
3. E.g. “steady-state GA” where we just produce 1 or 2 offspring at a time from any parents, and use fitness/selection to decide which individuals to **discard**. No generations.
4. E.g. “memetic algorithm”, a GA with an inner loop doing local search (e.g. mutation only).

Memetic algorithm

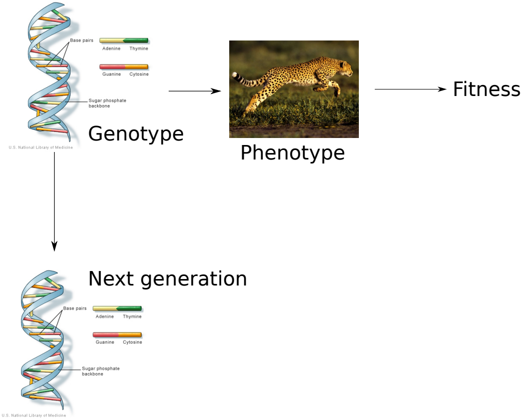
In *The Selfish Gene* (1978), Dawkins proposed that ideas evolve in a way similar to genes, and nicknamed them **memes**.

A **memetic algorithm** is a GA with an inner loop doing local search (e.g. mutation only). This is **not a good name** for the algorithm.


HIPSTER DAWKINS
LIKED MEMES
BEFORE THEY WERE COOL
imgflip.com

Overview

Genotypes and phenotypes



- Mapping from genes (**genotype**) to organism (**phenotype**)
- Only the organism is “evaluated” by nature for its “fitness”; only the genes are passed on to offspring.

Motivation

- Q. **Why** does nature do this?

Motivation

- Q. **Why** does nature do this?
- A. It's impossible to mutate a cheetah, or to crossover two cheetahs. Mutation and crossover work on the underlying DNA, not on the animal.

Example

A car is defined by:

- Shape (8 floats, 1 per vertex)
- Wheel size (2 floats, 1 per wheel)
- Wheel position (2 ints, 1 per wheel)
- Wheel density (2 floats, 1 per wheel) darker means denser
- Chassis density (1 float) darker means denser

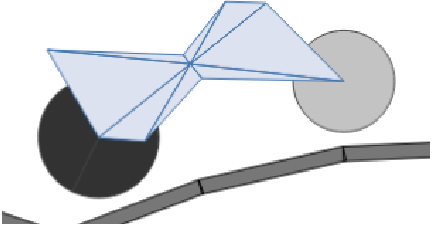


Image from rednuht.org

Phenotype: the car.

Example

A car is defined by:

- Shape (8 floats, 1 per vertex)
- Wheel size (2 floats, 1 per wheel)
- Wheel position (2 ints, 1 per wheel)
- Wheel density (2 floats, 1 per wheel) darker means denser
- Chassis density (1 float) darker means denser

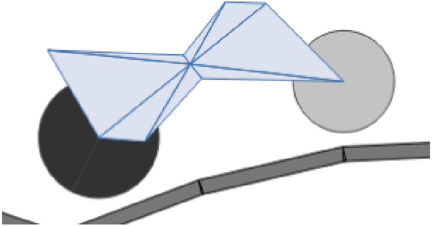


Image from rednuht.org

Phenotype: the car.

Genotype (15 genes): e.g. [.3 .4 .6 .1 .3 .4 .6 .1 | .6 .7 | 3 6 | .3 .8 | .4]

(Full details of genotype-phenotype mapping here.)

Objective: distance travelled in the simulation.

Search space and solution space

The same idea is often used outside Evolutionary Algorithms also, but with different terminology: the **search space** (genotype space) and the **solution space** (phenotype space).

What we actually want is a **solution**, but we run our search in the **search** space, and whenever we look at a search point we map it to a solution.

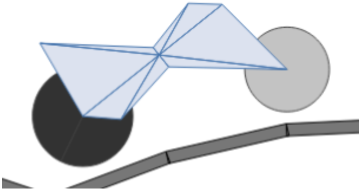
Terminology

Other optimisation	Evolutionary
Decision variable	Gene
Point	Genotype
Search space	Genotype space
Solution space	Phenotype space

Motivation

- Q. **Why** do we do this?
- A. Just like cheetahs, it is not possible to mutate or crossover 2D cars. But it is easy to define these operations on the underlying data structure.

Simulation fitness



rednuht.org

The car **phenotype** has to “live” in a simulated world. If it succeeds there, its **genotype** (parameter values) are passed on, i.e. are varied for use in the next iteration.

Explicit and implicit fitness

Explicit fitness:

- As we saw in LP/IP
- A **formula** or function that gives a **number**

Simulation fitness:

- As in 2D Boxcar
- The solution is put in a simulation, and we somehow measure performance as a **number**

Implicit fitness:

- As in biological evolution
- There is **no number**!
- Research fields: Coevolution and Artificial Life