Genetic Algorithms and Genetic Programming

Lecture 2: The Genetic Code (29/9/09)

School of informatics

Michael Herrmann

michael.herrmann@ed.ac.uk, phone: 0131 6 517177, Informatics Forum 1.42

Problem Solving as Optimization

Choosing the best option from some set of available alternatives

- Minimize energy, time, cost, risk, …
- Maximize gains, acceptance, turnover, …
- Discrete cost:
  - admissible goal state: maximal gain
  - anything else: no gain
- Secondary costs for:
  - acquisition of domain knowledge
  - testing alternatives
  - doing nothing
  - determining costs

A Simple Example

Consider the Tutor Allocation Problem

Jobs: $\text{Job}_1, \text{Job}_2, \ldots, \text{Job}_m$

Job_i is a single tutorial to be taught:

- subject, e.g. Java, GAGP
- slot, e.g. Thu 2–3
- place, e.g. A10, 5 Forrest Hill
- knowledge, skills required, e.g. strong at Java, some knowledge of AI techniques useful

A Simple Example

One tutor teaches each tutorial.

We have a pool of tutors to choose from:

Tutors:

Tutor A, Tutor B, Tutor C, …

Properties of tutors:

- knowledge/skills
- cost per hour
- time preferences
- room preferences
- optimal number of jobs

Solutions

A **solution** is an allocation of tutors to jobs:

Job:	1	2	3	4	5	6	7	8	9	10
Tutor:	A	B	C	D	E	F	G	H	I	J

Each job-tutor pairing can be given a **score**, based on how good the knowledge/skills match is:

Tutor A: some C++, strong at AI
Job 1: strong Java, some AI useful

– a reasonable match, though not perfect

A function $f_s(\text{job}, \text{tutor})$ calculates a numerical score for us for any pairing.

The **whole** solution can be given a score, based on:

- scores for job-tutor pairings
- total cost of solution
- hard constraints
- tutor preferences

The total score will be calculated from the scores for the individual parts.

The **problem** is to find the solution with the **best** score.

Possible Methods

Use exhaustive search?

- 5 tutors, 10 jobs $= 9.8 * 10^6$ solutions
- 10 tutors, 20 jobs $= 1.0 * 10^{20}$ solutions
- 15 tutors, 30 jobs $= 1.92 * 10^{35}$ solutions
- . . .

Possible Methods

Use greedy search?

Job 1 – find best tutor

Job 2 – find best tutor to give best combined score with the choice for Job 1

Job 3 – etc.

Almost certain to be sub-optimal since it commits to choices too early.

Possible Methods

Use Hillclimbing Local Search?

	1	2	3	4	5	6	7	8	9	10	Job
Solution$_i$:	A	B	E	A	B	B	$\boxed{\text{D}}$	C	E	D	Tutor

Suppose $\boxed{\text{D}}$ is the worst scorer. Try A, B, C, E

	1	2	3	4	5	6	7	8	9	10	Job
Solution$_{i+1}$:	A	B	E	A	B	B	$\boxed{\text{A}}$	C	E	D	Tutor

Continue until no improvement possible.

Prone to local maxima.

Genetic Algorithms

How about trying a biologically inspired solution based on genetics?

1. Generate a **population** of solutions:

Generation$_i$:

Solution1:	A	B	C	A	B	C	D	D	E	E
Solution2:	B	C	E	A	B	D	E	C	A	D
⋮										
Solution$n$:	E	D	A	C	C	D	A	D	B	A

2. Give each solution a score, called a **fitness**.

3. Create a **new generation** of solutions by:
(a) selecting fit solutions
(b) breeding new solutions from old ones and add to generation$_{i+1}$.

4. When a sufficiently good solution has been found, stop.

A Simple Genetic Algorithm

- Selection (out of *n* solutions, greedy type):
  - Calculate $\Sigma_i f_S(\mathrm{Job}_i, \mathrm{Tutor}_i)$ for each solution *S*
  - Rank solutions
  - Choose the *k* best scorers ($1 \leq k \leq n$)
- Breeding (Mixing good solutions):
  - take a few of the good solutions as parents
  - cut in halves, cross, and re-glue (see next slide)
- Mutation:
  - generate copies of the mixed solutions with very few modifications
  - e.g. for $k=n/2$: two “children” for each of them

Recombination and Mutation

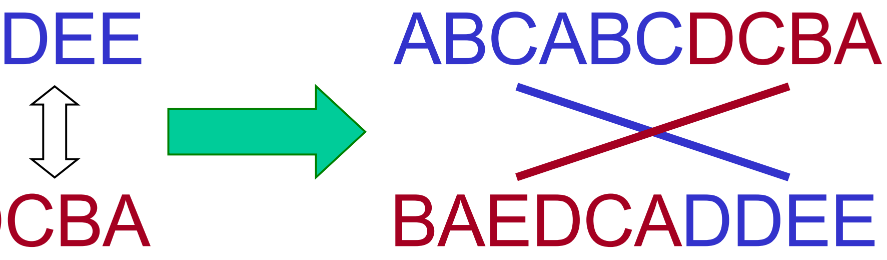
How does breeding work?

1. Reproduction:

Copy solution$_i$ unchanged into the next generation.

2. Crossover:

Parent1: ABCABC DDEE



Parent2: BAEDCA DCBA

Exchange of genetic material to form children.

3. Mutation:

(a) change one value in a solution to a random new value:

AEBCABDDCE $\Longrightarrow$ AEBCABDCCE

$\Uparrow$

(b) swap two values:

AEBCABDDCE $\Longrightarrow$ AEBDABDCCE

(c) lots of others!

Mutation is usually done after reproduction/crossover, with low probability (1%).

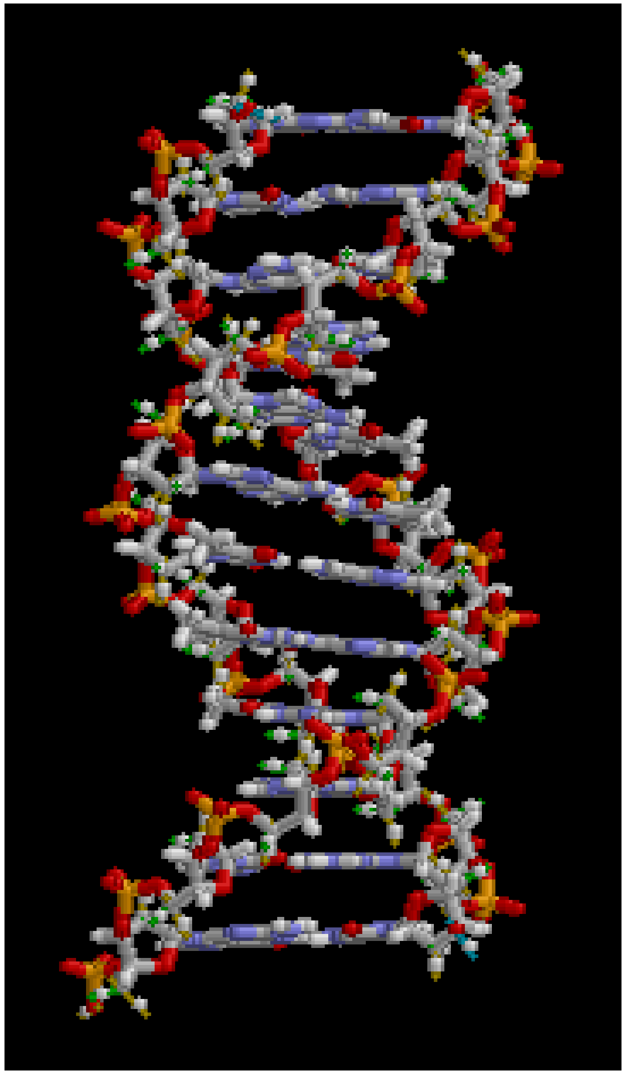
How Well Does This Work?

- small problems: optimal solutions
- larger problems: optimal or near optimal given enough time
- anytime behaviour
- runs on parallel machines
- adding constraints is very easy
- used in a multitude of real applications
- wide applicability to problems in search, optimisation, machine learning, automatic programming, A-life, . . .

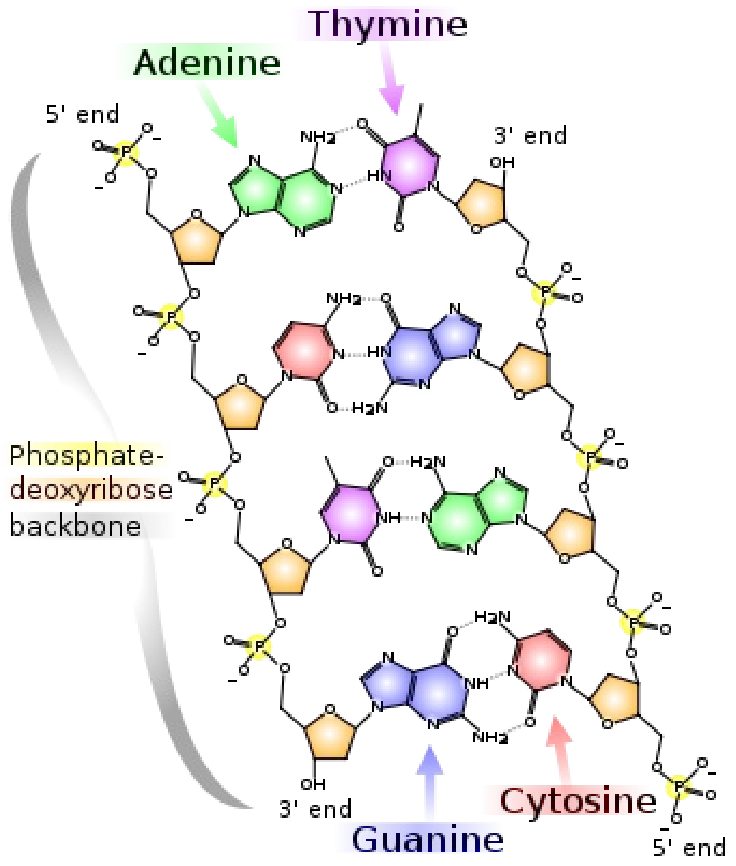
Overview

- Introduction: History
- The genetic code
- The canonical genetic algorithm
- Examples & practical issues
- The schema theorem

2. The Genetic code



James Watson, Francis Crick, Maurice Wilkins and Rosalind Franklin: DNA structure (hypothesis 1953, Nobel Prize 1962)



George Gamow: triplets [wrongly assumed ambiguity: GGAC=GGA+GAC (please do not memorize!)]

Genes

Sections of chromosome which encode a protein (i.e. tell you the order in which to connect amino acids together) are often called a **gene**. (Plus sections to encode when the gene will be activated, i.e. when/where the protein is produced.)

Other meanings of gene:

- genetic material which encodes a trait (e.g. eye colour)
- a long-lived inheritable genetic unit

The 46 chromosomes in every cell build proteins which make a human body.

In fact, we could use only 23: the 1a and 1b etc. chromosomes are alternative solutions to the same problems:

1a: . . . | Eyes are blue | . . .   1b: . . . | Eyes are brown | . . .

Sometimes one gene is dominant, the other recessive: in this case, eyes are brown. The “eyes are blue” gene is called an **allele** – a rival value. (Actually, there is more than one gene for eye colour.)

Sometimes the trait will be somewhere between the 2 values.

Sometimes (usually...) it’s more complicated!

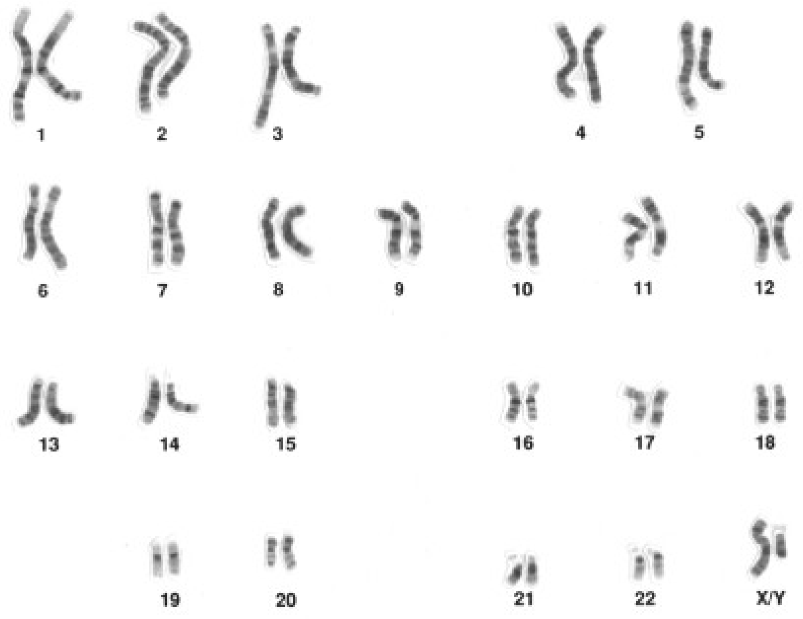
E.g.: body height, ADD, …

Chromosomes

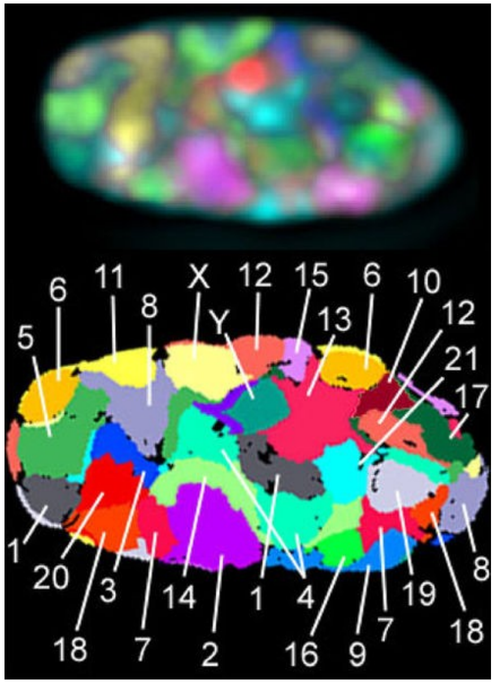
Inside every human cell (apart from gametes and red blood cells) are 46 strands of DNA called **chromosomes**. There are 23 pairs (diploid):

Mother:	1a	2a	3a	4a	. . . 23a
Father:	1b	2b	3b	4b	. . . 23b

The 46 chromosomes are the instructions for making an entire human being. (This depends on a lot of other things happening just at the right time.)



Karyogram of a human male 22+X+Y



The 24 human chromosome territories during prometaphase in fibroblast cells.

The Genetic Code

DNA = deoxyribonucleic acid

DNA is made up of a chain of simple molecular units. Each unit comprises a base, a sugar and a phosphate. The sugars and phosphates in many units link together in a chain with the bases sticking out. The bases in two chains attract one another resulting in a double helix structure.

There are just 4 kinds of **base** in DNA, labelled A, C, G and T (adenine, cytosine, guanine, thymine). C and G pair up, as do A and T.

. . . GATTACCA . . .
. . . CTAATGGT . . .

Reading: Wikipedia articles on “DNA”, “Introduction to genetics”

Central Dogma of Molecular Biology

- Enunciated by Francis Crick in 1958 (*Nature* 1970)
- "Information cannot be transferred back from protein to either protein or nucleic acid."
- In other words, 'once information gets into protein, it can't flow back to nucleic acid.'

From: To:	DNA	RNA	Protein
DNA	replication	reverse transcription	?
RNA	transcription	RNA replication	?
Protein	direct translation	translation	prions?

general transfer

special transfer

unknown transfer

Encoding Proteins

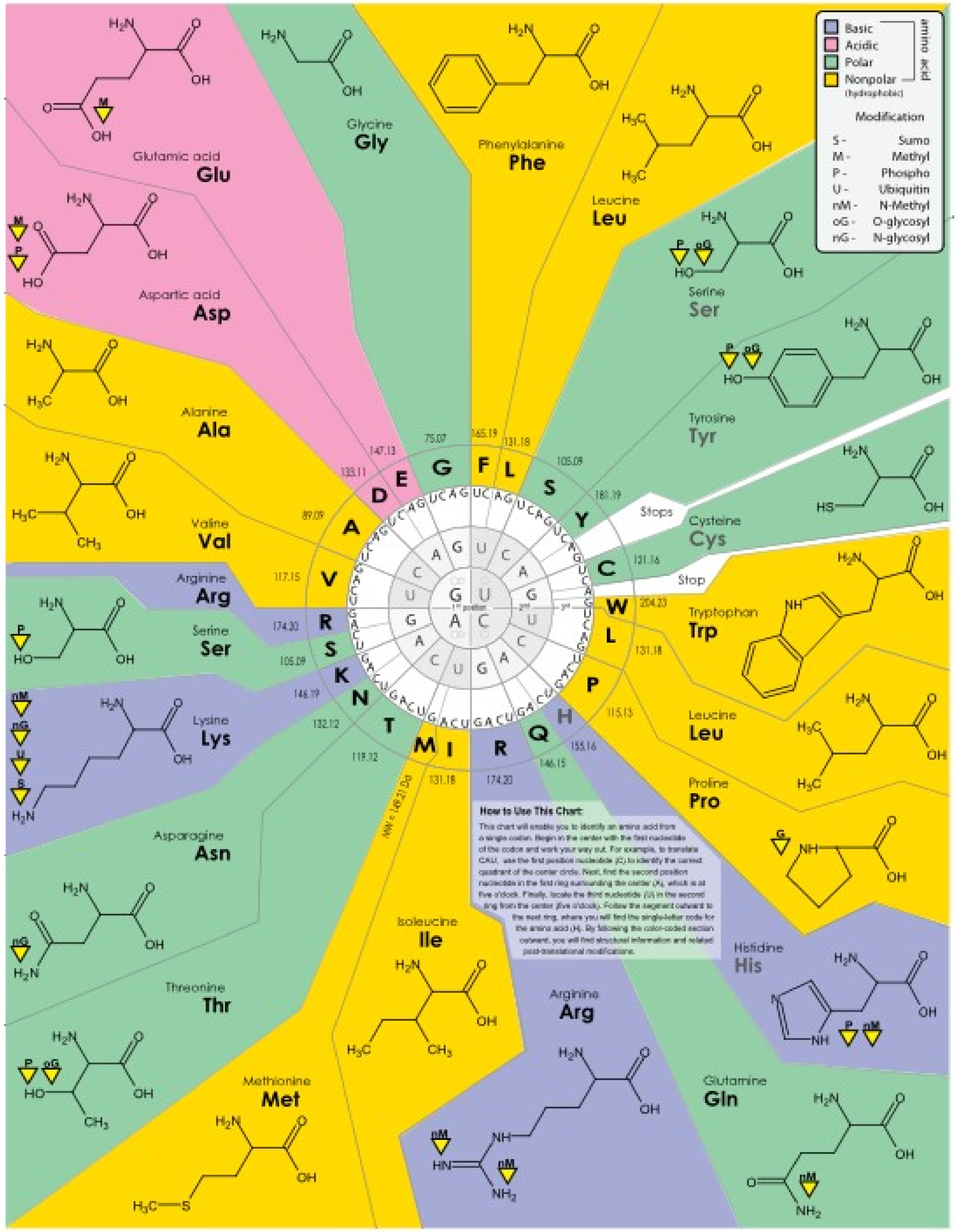
How does this work?

Sections of chromosome contain the instructions for building chains of amino acids – proteins. The proteins are the building blocks, regulation units and manufacturing units of the body:

e.g. lactase (enzyme), collagen (structure), haemoglobin (oxygen transport), actin (muscle contractions), CLOCK protein (circadian rhythm regulation).

Encoding:	3 DNA bases	$\rightarrow$	1 amino acid	AAA = lysine
	64 combinations	$\rightarrow$	20 amino acids	CCC = proline
	– some redundancy			

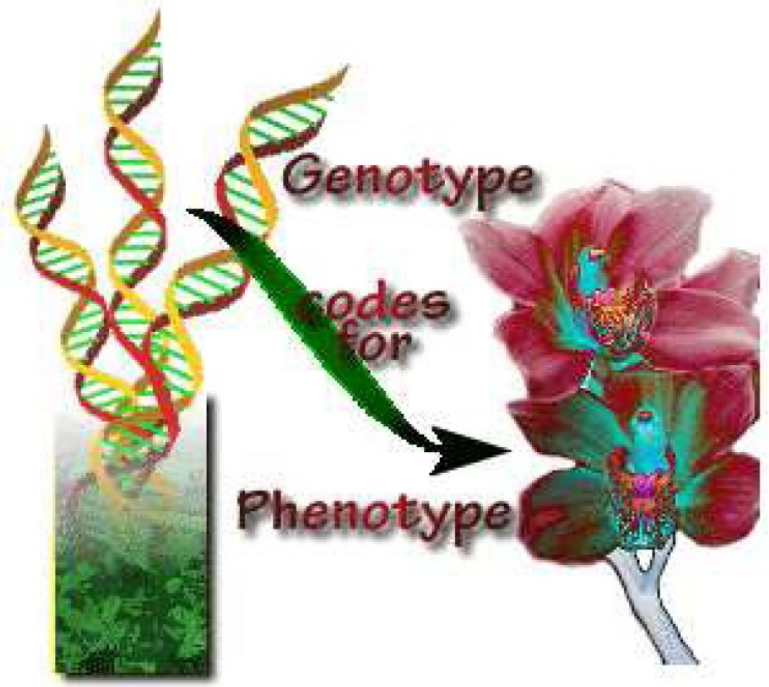
A protein is made up of many amino acids strung together and folded up.



How to Use This Chart:

This chart will enable you to identify an amino acid from a single codon. Begin in the center with the first nucleotide of the codon and work your way out. For example, to translate CAU, use the first position nucleotide (C) to identify the correct quadrant of the center circle. Next, find the second position nucleotide in the first ring surrounding the center (A), which is at five o'clock. Finally, locate the third nucleotide (U) in the second ring from the center (five o'clock). Follow the segment outward to the next ring, where you will find the single-letter code for the amino acid (H). By following the color-coded section outward, you will find structural information and related post-translational modifications.

Genotype and Phenotype



DNA code:	$\rightarrow$	human being
genotype	$\rightarrow$	phenotype

The genotype is the genetic coding for an individual – so you can pass it from one individual to another, replicate it ($\sim$ digital information).

The phenotype is what the genotype decodes into – the physical instantiation ($\sim$ analogue).

A genotype may decode into different phenotypes depending on environment. And the same phenotype may result from different genotypes.

Selection – Survival of the Fittest?

When individuals exist in populations, they **compete** for resources.

The fitter ones live, the less fit die.

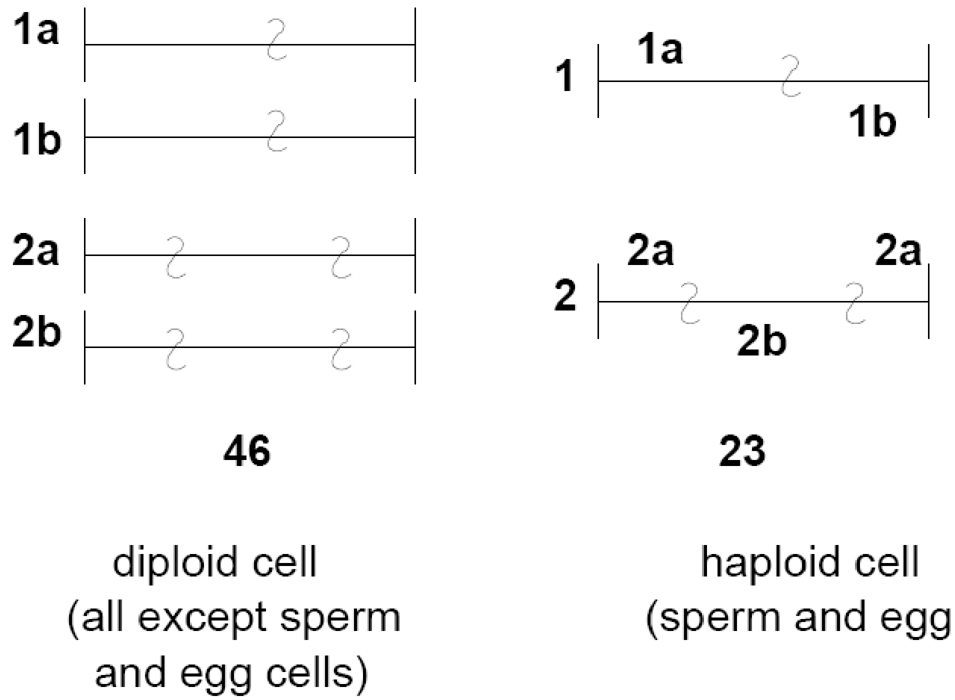
The survivors choose mates and produce offspring. The offspring share similarities with, but are not direct copies of, the parents:

- children inherit traits from parents
- offspring vary in their physical properties and behaviour

How can we explain this?

- sexual reproduction
- mutation
- crossover
- inversion

Crossover



Every haploid cell is a result of **different** recombinations of the 46 chromosomes.

When sperm and egg fuse, the 23 from the male and the 23 from the female are simply collected together to form a new collection of 46.

Mutation

During crossover, sometimes copying errors are made, with low probability:

. . . CGTATTCATGG . . .
. . . CGTA**C**TCATGG . . .

Also, sometimes strands of DNA become detached and flip end-over-end before reattaching – **inversion**:

. . . CGTATTCATGG . . .
. . . CGTA**ACTT**TGG . . .

So different amino acids are coded for. Crossover can be disruptive – it favours smaller units of inheritance (Dawkins).

Some (very large) parts of DNA seem to be there to protect genes from crossover (junk DNA, introns, bloat). Free radicals can also disrupt DNA.

Selection – Fitness Landscapes

Selection governs which organisms live long enough to pass on their genes.

Selection pressure defines a fitness landscape which favours one type of creature over others. E.g. in landscapes that provide sparse food, creatures that store fat efficiently are more likely to survive.

$\rightarrow$ optimisation

Populations can also adapt to changes in their environment. E.g. if populations start farming rather than hunting/gathering they develop enzymes that can digest grains.

$\rightarrow$ adaptation

Genetics and Evolution $\rightarrow$ AI?

Evolution shows how complexity and solutions to the problem of survival arise from populations, selection and recombination.

In our search for AI, such ideas are appealing:

– intelligence as an emergent property

– optimisation of behaviours

– adaptation to changes

– learning of new behaviours

– searching for optimal solutions (or better than the rest?)

– Artificial Life

Using These Ideas

How does all this inspire us to build clever computer programs?

– See Lecture 1!

- find a hard problem (local maxima)
- encode solutions as a genotype
- map genotypes to phenotypes (optional)
- evaluate phenotype for fitness
- mate fit genotypes using crossover
- make a few mutations
- continue until you have your solution (or you are convinced that no improvement is possible)

The Main Issues

- How do I represent a solution?
- How should I rate a solution for fitness?
- How large should the population of solutions be?
- How much selection pressure should I apply?
- What form of crossover should I use?
- what form of mutation should I use?

Next lecture: The Canonical GA

DNA figure: Access Excellence Graphics Library.
Genotype–phenotype figure: Blamire's Science at a Distance.