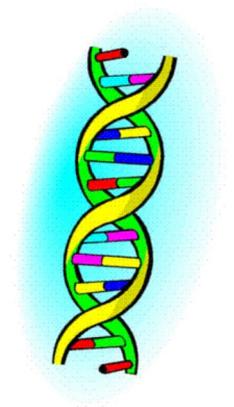
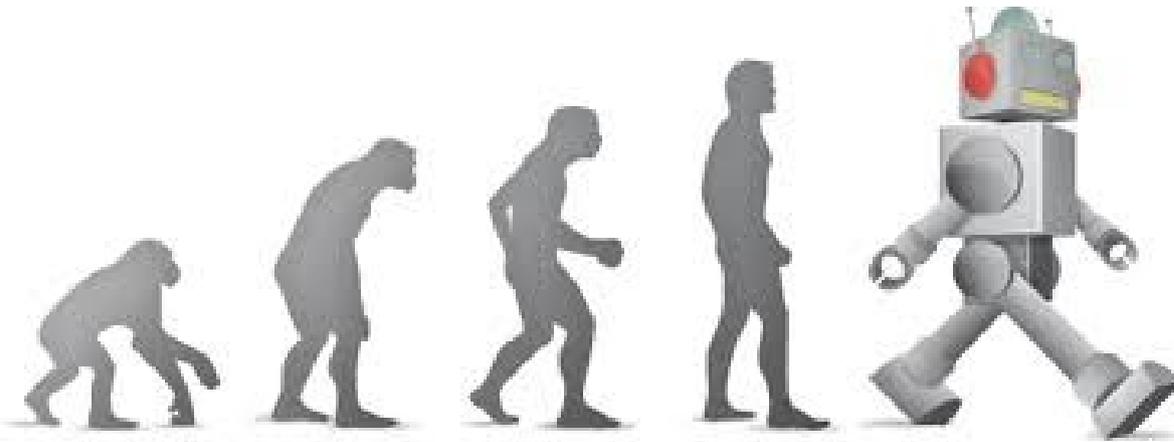


Genetic Algorithms

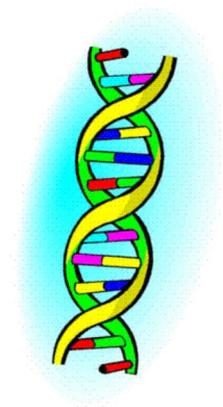


Genetic Algorithms

- Invented by John Holland in the 1960s



1929 - 2015

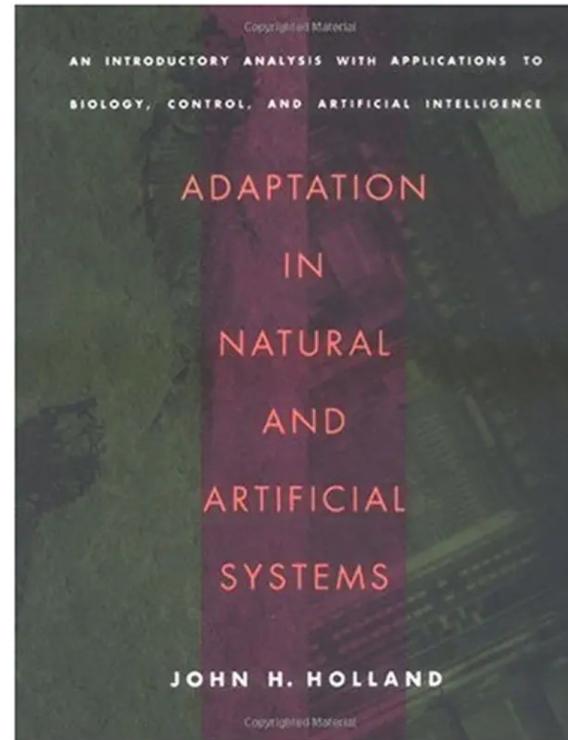


Genetic Algorithms

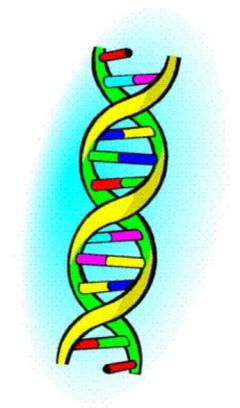
- Invented by John Holland in the 1960s



1929 - 2015



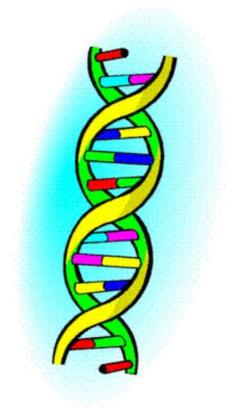
1975



Genetic Algorithms

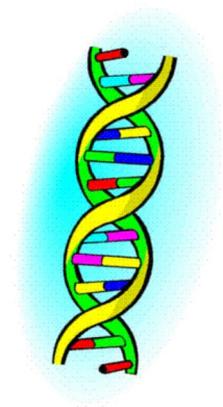
- **Genome**

- An encoded representation of a candidate solution to the problem (typically a sequence of numbers or bits)



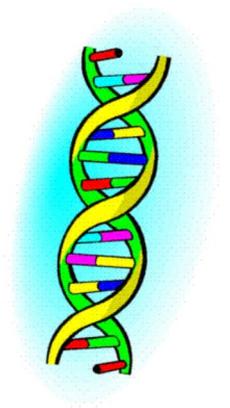
Genetic Algorithms

- **Genome**
 - An encoded representation of a candidate solution to the problem (typically a sequence of numbers or bits)
- **Population of genomes**
 - A pool of candidate solutions, initially generated at random



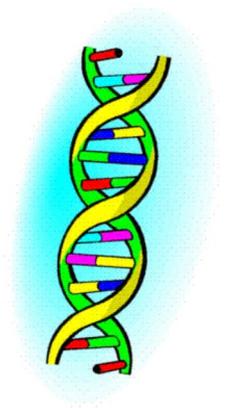
Genetic Algorithms

- **Genome**
 - An encoded representation of a candidate solution to the problem (typically a sequence of numbers or bits)
- **Population of genomes**
 - A pool of candidate solutions, initially generated at random
- **Fitness function**
 - $f(\text{genome}) \Rightarrow$ numerical estimate of quality



Genetic Algorithms

- **Genome**
 - An encoded representation of a candidate solution to the problem (typically a sequence of numbers or bits)
- **Population of genomes**
 - A pool of candidate solutions, initially generated at random
- **Fitness function**
 - $f(\text{genome}) \Rightarrow$ numerical estimate of quality
- **Operators**
 - *Selection*: survival of the fittest
 - *Crossover*: genetic recombination
 - *Mutation*: random variation



Outline of a Genetic Algorithm

1. Create a population of random genomes

Outline of a Genetic Algorithm

1. Create a population of random genomes

110010011110011
000011101001100
100110100110100
111101011111101
00000100100010
101001101010101
000010000101011
110100100010001

Outline of a Genetic Algorithm

1. Create a population of random genomes
2. Evaluate the fitness of each genome in the population

110010011110011
000011101001100
100110100110100
111101011111101
00000100100010
101001101010101
000010000101011
110100100010001

Outline of a Genetic Algorithm

1. Create a population of random genomes
2. Evaluate the fitness of each genome in the population

$$\begin{aligned} f(110010011110011) &= 9 & f(000011101001100) &= 6 & f(100110100110100) &= 7 \\ f(111101011111101) &= 12 & f(00000100100010) &= 3 & f(101001101010101) &= 8 \\ f(000010000101011) &= 5 & f(110100100010001) &= 6 & & \end{aligned}$$

Outline of a Genetic Algorithm

1. Create a population of random genomes
2. Evaluate the fitness of each genome in the population
3. Build a new population of genomes:
 - (a) **select** 2 genomes probabilistically, based on fitness

$$\begin{array}{l} f(110010011110011) = 9 \\ f(000011101001100) = 6 \\ f(100110100110100) = 7 \\ f(111101011111101) = 12 \\ f(00000100100010) = 3 \\ f(101001101010101) = 8 \\ f(000010000101011) = 5 \\ f(110100100010001) = 6 \end{array}$$

Outline of a Genetic Algorithm

1. Create a population of random genomes
2. Evaluate the fitness of each genome in the population
3. Build a new population of genomes:
 - (a) **select** 2 genomes probabilistically, based on fitness

$$f(110010011110011) = 9$$
$$f(000011101001100) = 6$$
$$f(100110100110100) = 7$$
$$f(111101011111101) = 12$$
$$f(00000100100010) = 3$$
$$f(101001101010101) = 8$$
$$f(000010000101011) = 5$$
$$f(110100100010001) = 6$$

Outline of a Genetic Algorithm

1. Create a population of random genomes
2. Evaluate the fitness of each genome in the population
3. Build a new population of genomes:
 - (a) **select** 2 genomes probabilistically, based on fitness

100110100110100

111101011111101

Outline of a Genetic Algorithm

1. Create a population of random genomes
2. Evaluate the fitness of each genome in the population
3. Build a new population of genomes:
 - (a) **select** 2 genomes probabilistically, based on fitness
 - (b) create 2 new offspring from them, using **crossover**

100110100110100

111101011111101

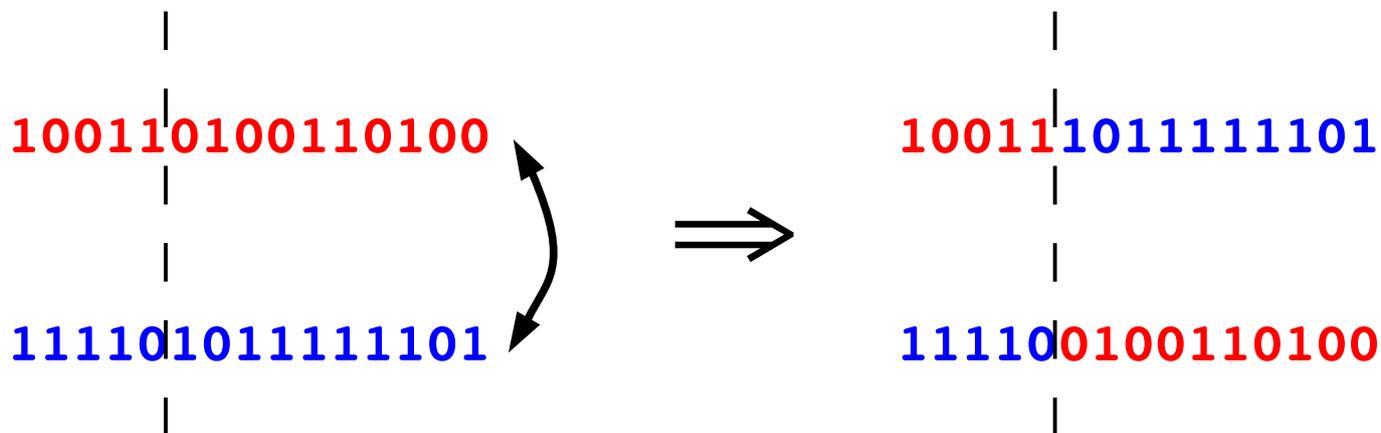
Outline of a Genetic Algorithm

1. Create a population of random genomes
2. Evaluate the fitness of each genome in the population
3. Build a new population of genomes:
 - (a) **select** 2 genomes probabilistically, based on fitness
 - (b) create 2 new offspring from them, using **crossover**

|
|
100110100110100
|
|
11110101111101
|

Outline of a Genetic Algorithm

1. Create a population of random genomes
2. Evaluate the fitness of each genome in the population
3. Build a new population of genomes:
 - (a) **select** 2 genomes probabilistically, based on fitness
 - (b) create 2 new offspring from them, using **crossover**



Outline of a Genetic Algorithm

1. Create a population of random genomes
2. Evaluate the fitness of each genome in the population
3. Build a new population of genomes:
 - (a) **select** 2 genomes probabilistically, based on fitness
 - (b) create 2 new offspring from them, using **crossover**

100111011111101

111100100110100

Outline of a Genetic Algorithm

1. Create a population of random genomes
2. Evaluate the fitness of each genome in the population
3. Build a new population of genomes:
 - (a) **select** 2 genomes probabilistically, based on fitness
 - (b) create 2 new offspring from them, using **crossover**
 - (c) **mutate** each offspring with some small probability

100111**0**111111**0**1

111100100110100

Outline of a Genetic Algorithm

1. Create a population of random genomes
2. Evaluate the fitness of each genome in the population
3. Build a new population of genomes:
 - (a) **select** 2 genomes probabilistically, based on fitness
 - (b) create 2 new offspring from them, using **crossover**
 - (c) **mutate** each offspring with some small probability

100111**1**1111111**1**1

011100100110100

Outline of a Genetic Algorithm

1. Create a population of random genomes
2. Evaluate the fitness of each genome in the population
3. Build a new population of genomes:
 - (a) **select** 2 genomes probabilistically, based on fitness
 - (b) create 2 new offspring from them, using **crossover**
 - (c) **mutate** each offspring with some small probability
 - (d) add the offspring to the new population

1001111111111111

011100100110100

Outline of a Genetic Algorithm

1. Create a population of random genomes
 2. Evaluate the fitness of each genome in the population
 3. Build a new population of genomes:
 - (a) **select** 2 genomes probabilistically, based on fitness
 - (b) create 2 new offspring from them, using **crossover**
 - (c) **mutate** each offspring with some small probability
 - (d) add the offspring to the new population

continue steps (a) - (d)
-

Outline of a Genetic Algorithm

1. Create a population of random genomes
 2. Evaluate the fitness of each genome in the population
 3. Build a new population of genomes:
 - (a) **select** 2 genomes probabilistically, based on fitness
 - (b) create 2 new offspring from them, using **crossover**
 - (c) **mutate** each offspring with some small probability
 - (d) add the offspring to the new population

continue steps (a) - (d)
 4. When the new population has reached the same size as the current population, replace the current population by the new population
-

Outline of a Genetic Algorithm

1. Create a population of random genomes



2. Evaluate the fitness of each genome in the population

3. Build a new population of genomes:

(a) **select** 2 genomes probabilistically, based on fitness

(b) create 2 new offspring from them, using **crossover**

(c) **mutate** each offspring with some small probability

(d) add the offspring to the new population

continue steps (a) - (d)

4. When the new population has reached the same size as the current population, replace the current population by the new population ... **and repeat**

Outline of a Genetic Algorithm

- Over time, the **average fitness** of the population will increase

Outline of a Genetic Algorithm

- Over time, the **average fitness** of the population will increase
- Even the **best-fit** individuals are not guaranteed to survive to the next generation

Outline of a Genetic Algorithm

- Over time, the **average fitness** of the population will increase
- Even the **best-fit** individuals are not guaranteed to survive to the next generation
- Even the **worst-fit** individuals have some (small) probability of surviving

Outline of a Genetic Algorithm

- Over time, the **average fitness** of the population will increase
- Even the **best-fit** individuals are not guaranteed to survive to the next generation
- Even the **worst-fit** individuals have some (small) probability of surviving
- Some GAs use **elitism** to ensure that the best individuals do survive

Outline of a Genetic Algorithm

- Over time, the **average fitness** of the population will increase
- Even the **best-fit** individuals are not guaranteed to survive to the next generation
- Even the **worst-fit** individuals have some (small) probability of surviving
- Some GAs use **elitism** to ensure that the best individuals do survive
- Many ways of probabilistically selecting individuals
 - **fitness-proportionate** selection (“roulette-wheel sampling”)
 - **rank** selection
 - **tournament** selection

A Simple Example

	Genome	Fitness	New population
A:	00000110	2	
B:	11101110	6	
C:	00100000	1	
D:	00110100	3	

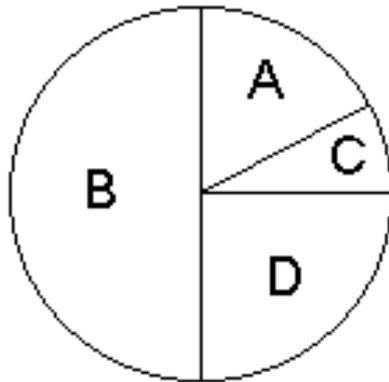
A Simple Example

	Genome	Fitness	New population
A:	00000110	2	
B:	11101110	6	
C:	00100000	1	
D:	00110100	3	

Average fitness of current population = $12 / 4 = 3.0$

A Simple Example

Genome	Fitness	New population
A: 00000110	2	
B: 11101110	6	
C: 00100000	1	
D: 00110100	3	



Fitness-proportionate selection
("roulette-wheel sampling")

B: 11101110 and C: 00100000 are selected

A Simple Example

	Genome	Fitness	New population
A:	00000110	2	
B:	11101110	6	
C:	00100000	1	
D:	00110100	3	

No crossover

B: 11101110

C: 00100000

A Simple Example

	Genome	Fitness	New population
A:	00000110	2	
B:	11101110	6	
C:	00100000	1	
D:	00110100	3	

No crossover

B is mutated

B: **1**1101110 → E: **0**1101110

C: 00100000

A Simple Example

	Genome	Fitness	New population
A:	00000110	2	E: 01101110
B:	11101110	6	C: 00100000
C:	00100000	1	
D:	00110100	3	

No crossover

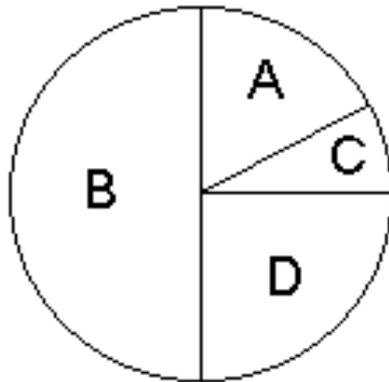
B is mutated

B: **1**1101110 → E: **0**1101110

C: 00100000

A Simple Example

Genome	Fitness	New population
A: 00000110	2	E: 01101110
B: 11101110	6	C: 00100000
C: 00100000	1	
D: 00110100	3	



Fitness-proportionate selection
("roulette-wheel sampling")

B: 11101110 and D: 00110100 are selected

A Simple Example

Genome	Fitness	New population
A: 00000110	2	E: 01101110
B: 11101110	6	C: 00100000
C: 00100000	1	
D: 00110100	3	

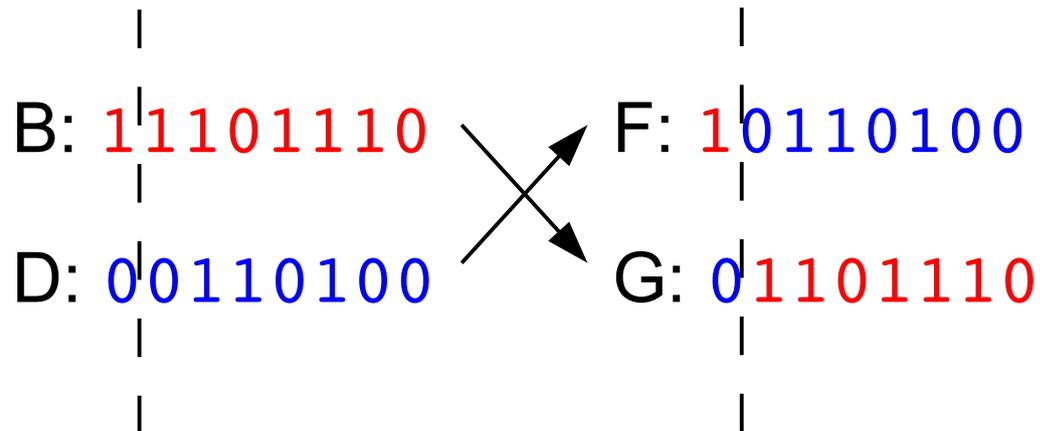
Crossover occurs

|
B: 11101110
|
D: 00110100
|
|

A Simple Example

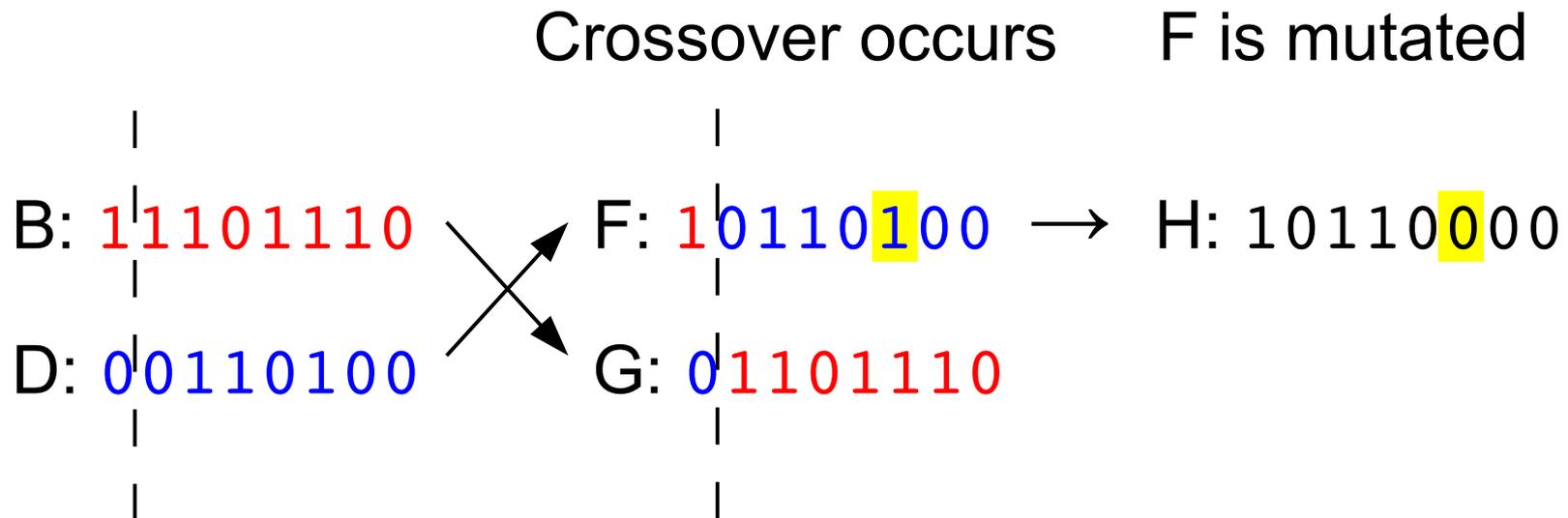
Genome	Fitness	New population
A: 00000110	2	E: 01101110
B: 11101110	6	C: 00100000
C: 00100000	1	
D: 00110100	3	

Crossover occurs



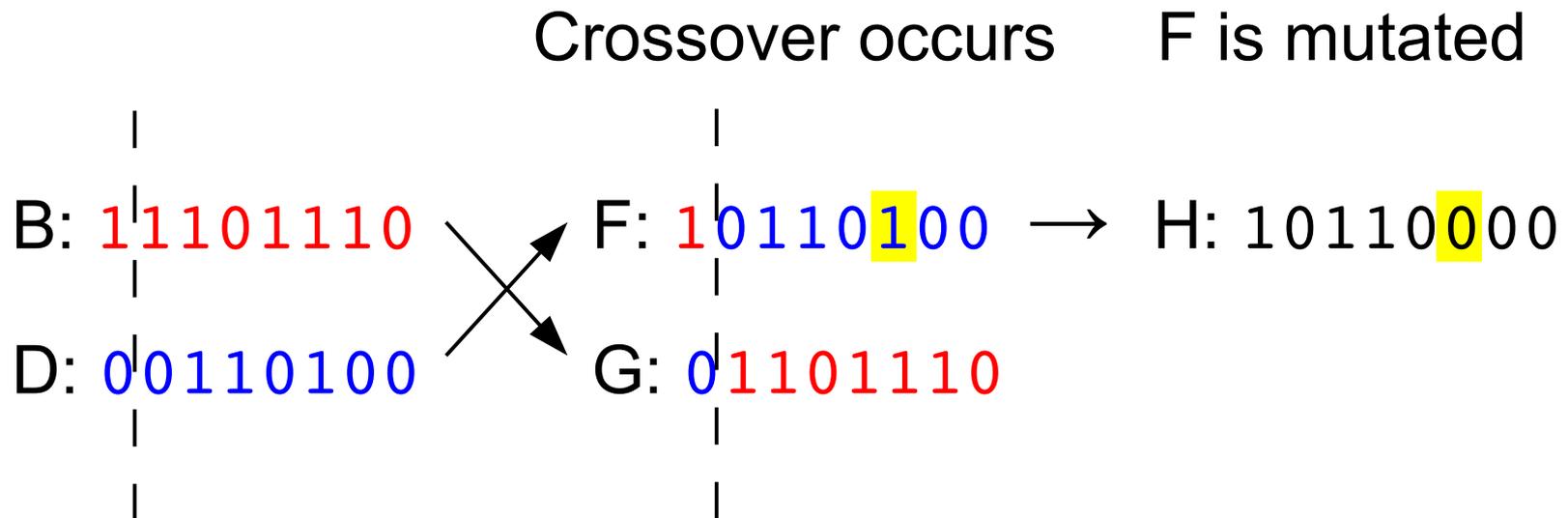
A Simple Example

Genome	Fitness	New population
A: 00000110	2	E: 01101110
B: 11101110	6	C: 00100000
C: 00100000	1	
D: 00110100	3	



A Simple Example

Genome	Fitness	New population
A: 00000110	2	E: 01101110
B: 11101110	6	C: 00100000
C: 00100000	1	H: 10110000
D: 00110100	3	G: 01101110



A Simple Example

Genome	Fitness	New population	Fitness
A: 00000110	2	E: 01101110	5
B: 11101110	6	C: 00100000	1
C: 00100000	1	H: 10110000	3
D: 00110100	3	G: 01101110	5

A Simple Example

Genome	Fitness	New population	Fitness
A: 00000110	2	E: 01101110	5
B: 11101110	6	C: 00100000	1
C: 00100000	1	H: 10110000	3
D: 00110100	3	G: 01101110	5

Average fitness of new population = $14 / 4 = 3.5$

Best-fit genome from previous population was lost

Demo: Evolving an English Phrase

- Genome
a sequence of letters representing an English phrase

Demo: Evolving an English Phrase

- Genome
a sequence of letters representing an English phrase
- Fitness function
the number of letters in the genome that match
“the rain in spain stays mainly in the plain”

Demo: Evolving an English Phrase

- Genome
a sequence of letters representing an English phrase
- Fitness function
the number of letters in the genome that match
“the rain in spain stays mainly in the plain”
- Example
“the yain in szbin stays mainly ik the ploin”
fitness = 38

Demo: Evolving an English Phrase

- Genome
a sequence of letters representing an English phrase
- Fitness function
the number of letters in the genome that match
“the rain in spain stays mainly in the plain”
- Example
“the yain in szbin stays mainly ik the ploin”
fitness = 38
- GA parameters
population size: 100
crossover probability: 0.75
mutation probability: 0.005 per locus
elitism: 0.5

Key GA Concepts

- Search space
- Fitness landscape
- Local minimum / maximum
- Global minimum / maximum
- Hill-climbing search
- Population-based search

