Genetic Algorithms

Sources:

Melanie Mitchell - *Complexity: A Guided Tour*, Oxford University Press (2009)

Jon Sumner (Physics) - https://compute.dawsoncollege.qc.ca/course-modules/optimization/robby-robot/

Let’s say:

- You are trying to optimize a problem with many possible discrete solutions
  - Discrete = separate and distinct
    - Not related to any previous move
  - Optimize = find the best solution
    - Maximize profits = Max (Revenues – costs)
- Example: given a city grid, where should parking kiosks be placed?
  - Minimize costs = few kiosks
  - Maximize revenues = easy to pay, many kiosks

Example

- Which blocks should have a kiosk?
- Exhaustive search – go through every permutation:
- 4 blocks – how many permutations?

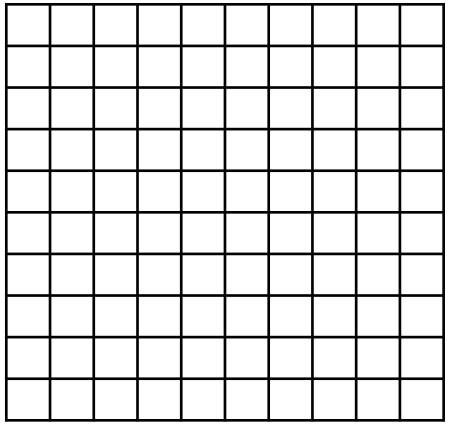
Example

- Which blocks should have a kiosk?
- Exhaustive search – go through every permutation:
- 4 blocks – how many permutations?

- Answer = 2 options each block => 2^4

What if

- Exhaustive search for N = 100
- 2^{100}
- If you need to evaluate a function for each permutation that takes 0.6 ms



→ 2.4×10^{16} millennia

Age of universe = 1.4×10^{7} millennia

Other options

- Trial and error
- Algorithm
  - a series of steps by which an input is transformed to an output
  - Input = permutations
  - Output = optimized solution

Evolutionary computation

- Inspired by nature
- Evolution and adaptation
  - Individuals better adapted to the environment thrive and reproduce

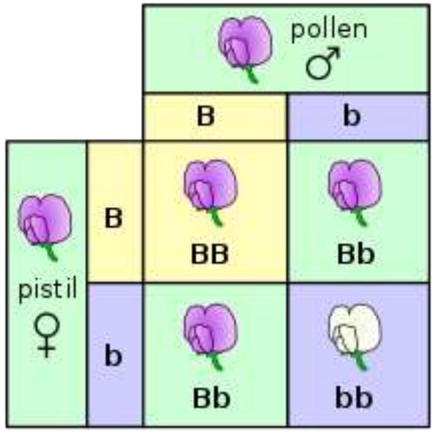
  -> nature's optimization algorithm!

Genetic algorithm (GA)

- Example of an evolutionary computational algorithm
- The input to the GA
  - a **population** of candidate solutions
  - a **fitness** function that takes a candidate solutions and assigns to it a value that measures how well it solves the task
- Candidate solutions can be represented as a series of bits (if there are two options) or ints/enum (if there are many)
  - E.g. 0001 means the candidate solution to the parking kiosk problem is to put the kiosk in the 4^{th} block
- If we run the fitness function on every possible candidate solution (i.e., the entire population) -> same as exhaustive search

Genetics analogy

- inheritance in organisms occurs by passing genes, from parents to offspring
- different, discrete versions of the same gene are called alleles



- Chromosones contains genes
- Chromosonal crossover is how genes are recombined during reproduction
- Mutations occur randomly when some alleles change

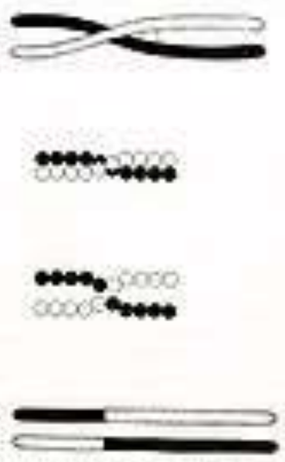


Fig. 64. Scheme to illustrate a method of crossing over of the chromosomes.

Genetics analogy

Population	Set of all solutions -> set of all possible chromosones
Chromosone	A particular solution, represented by a sequence of bits/numbers/enums/strings
Gene	A specific bit/number/string in the chromosone
Allele	Allowed values of the gene (e.g., 0 or 1, etc…)
Reproduction	Mix of 2 chromosones to form a third (and fourth)
Mutation	Random change of a Gene’s value (change of an allele)
Generation	Iteration

Coding a GA

1. Generate an initial population of candidate solutions.
   - generate a bunch of random chromosones, called “individuals.
2. Calculate the fitness of each individual in the current population.
3. Select some random individuals with high fitness to be the parents of the next generation.
4. Pair up the selected parents.
   - Each pair produces offspring by recombining parts of the parents, with some chance of random mutations, and the offspring enter the new population.
   - Continue selecting paraents and procreating until the population is full
   - The new population now becomes the current population (i.e., next generation replaces the current)
5. Go to step 2.

- Repeat this for many generations

Sounds weird, but it works!

- Genetic algorithms have been used by GE to automate design of some aircraft parts
- John Deere to automate assembly line scheduling
- Texas Instruments for computer chip design
- The Lord of the Rings to generate realistic CGI horses
- London Stock Exchange to detect fraudulent trades
- First Quadrant to optimize investment portfolios
- NASA to develop novel antenna designs

Source: Complexity: A Guided Tour by Melanie Mitchell

Encoding solutions in chromosones

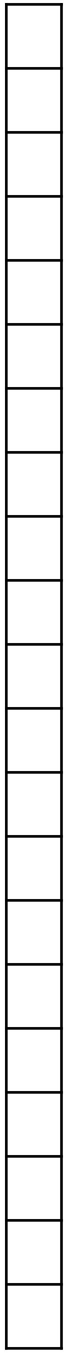
- Example: parking kiosks

1	1	0	0	1	0	0	0	1	0
0	0	0	1	0	0	0	0	0	1
0	1	1	0	0	1	1	0	1	0
1	1	0	0	1	0	0	0	1	0
0	0	0	1	0	0	0	0	0	1
0	1	1	0	0	1	1	0	1	0
1	1	0	0	1	0	0	0	1	0
0	0	0	1	0	0	0	0	0	1
0	1	1	0	0	1	1	0	1	0
1	1	0	0	0	1	0	1	0	0

- chromosome with 100 genes
- each gene has 2 possible alleles (0 or 1)

=> Abstraction = array of 100 bits

- population is 100 random chromosones
- Apply the fitness function and find the best parents

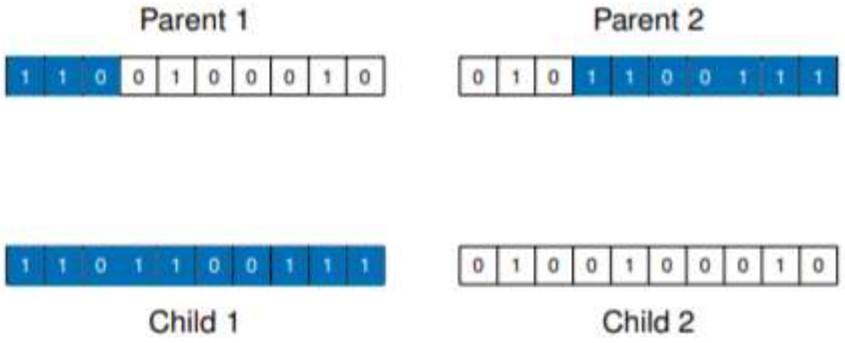


Choosing the parents

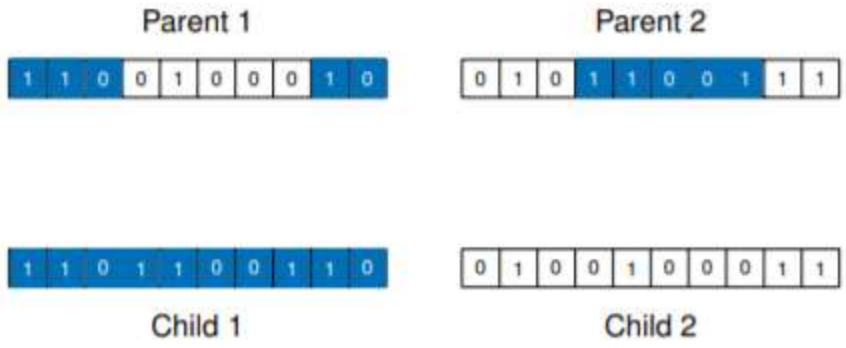
- Parents are randomly chosen with greater weight to the best solutions
- Elitism: also clone the best solutions (keep them in the population for the next generation
    - parameter - what percent is elite?

Reproduction

- Choose best solution (highest fitness) to be parents
- Options (strategies) for reproduction:
- Single point crossover:

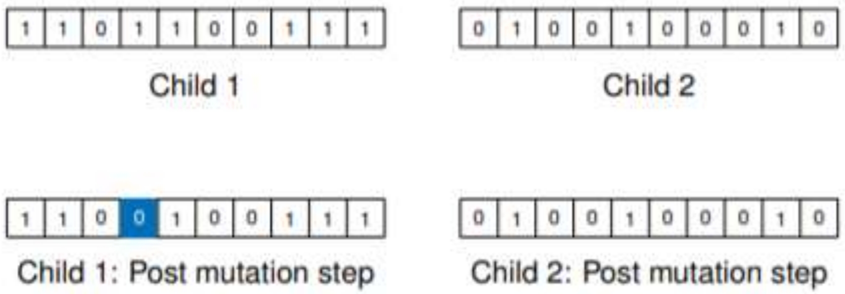


- Two-point crossover



Mutation

- After reproduction, for each gene there is a small, non-zer probability of changing the allele
- Sometimes no mutation will occur, other times multiple mutations on the same individual
- Parameter – mutation rate for any given gene

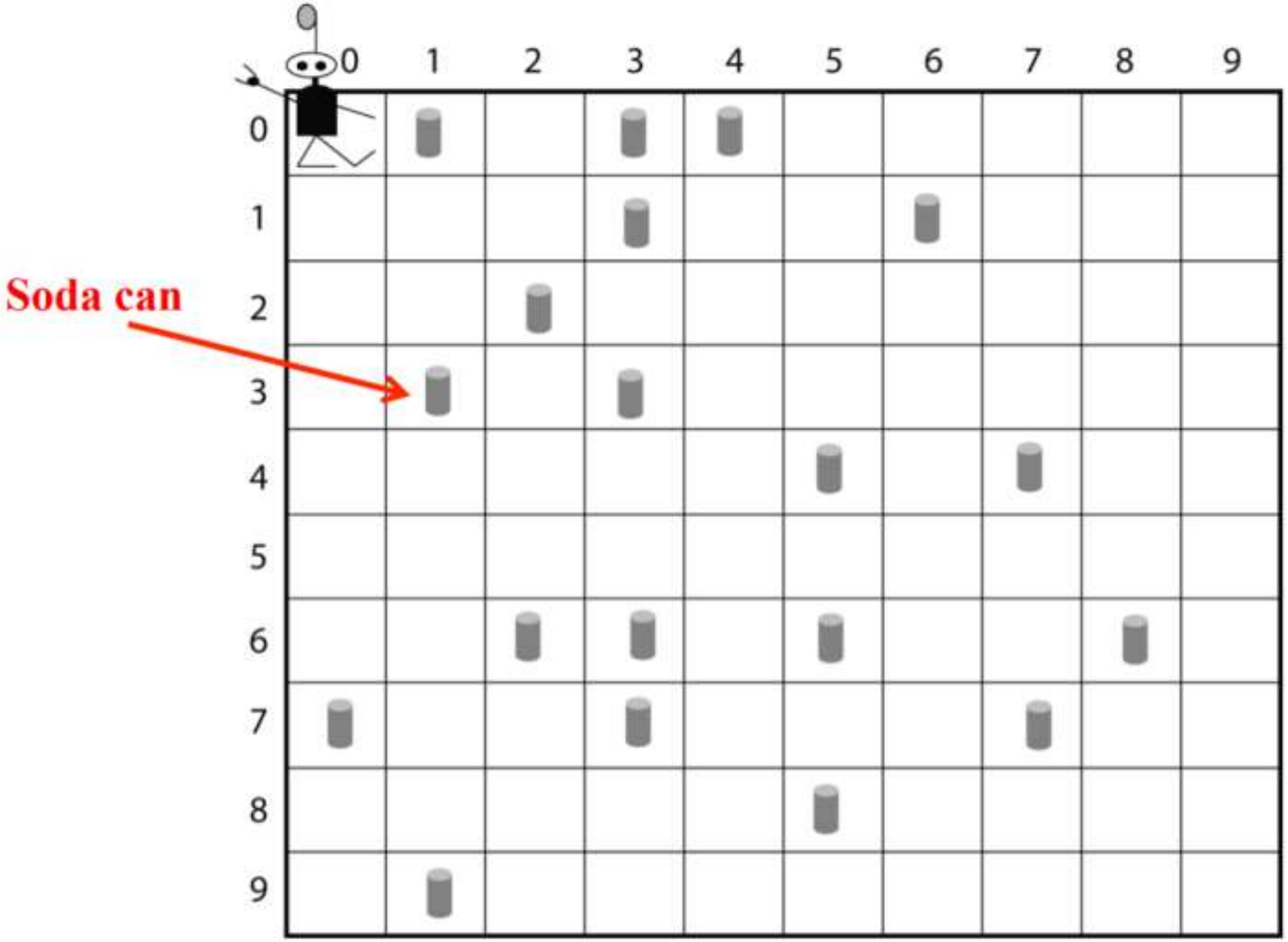


Application: Robby the Robot

- Robby lives in a two-dimensional world that is randomly strewn with empty soda cans
- Use a genetic algorithm to evolve a strategy for Robby to collect the most cans possible
- Robby cannot see well: from wherever he is, he can see the contents of one adjacent site in the north, south, east, and west directions, as well as the contents of the site he occupies. A site can be empty, contain a can, or be a wall.
- Robby has no memory of any previous move; he doesn't know he was going north or east; he just know what is around him in the adjacent blocks and current block

Source: Complexity: A Guided Tour by Melanie Mitchell

Robby’s world



- 100 squares in a 10 x 10 grid
- Wall around the area
- No more than 1 can/square
- At Robby’s current position, he sees:
  - his current site is empty,
  - North and West are walls,
  - South is empty,
  - East has a can

Fitness calculations

- Robby performs 200 actions that lead to points:
- Action choices:
  - Move North
  - Move South
  - Move East
  - Move West
  - Stay
  - Pick a can
  - Random move
- Fitness points => Reinforcement
  - 10 points for every can Robby picks up
  - -1 point if Robby tries to pick up a can that is not there
  - -5 points for crashing into a wall (stays in the same place)

  (why can negative points happen? Because we are testing solutions of 200 actions! Some will keep bumping into walls. Solutions are “hard-coded”, there is no logic in the solutions)

  Score = sum of all rewards and penalties

What does a possible solution (chromosome) look like?

- In general, a solution is a set of rules that gives, for any situation, the action you should take in that situation
- In how many situations can Robby be in? (how many permutations)?
- He can see the content of the positions that are North, South, East, West and Current
- Each position has 3 possible values: empty, can, wall

- 3 values, 5 positions => 3^5 or 243 total permutations of situations

(actually it is less since some permutations are impossible, such as current being a wall, or N and S both being walls, etc…)

All possible 243 situations -> 243 genes

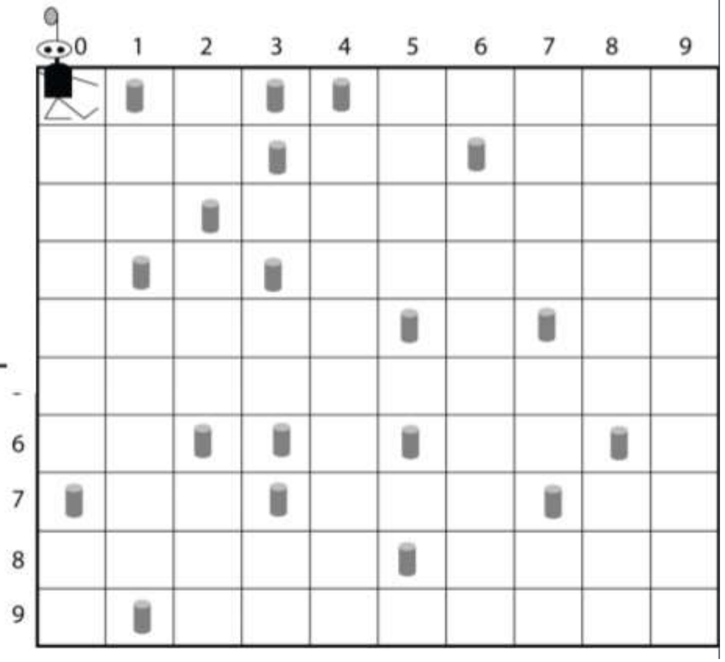
	Situation				
	North	*South*	*East*	*West*	*Current Site*
1	Empty	Empty	Empty	Empty	Empty
2	Empty	Empty	Empty	Empty	Can
3	Empty	Empty	Empty	Empty	Wall
4	Empty	Empty	Empty	Can	Empty
.	⋮	⋮	⋮	⋮	⋮
.	Wall	Empty	Can	Wall	Empty
.	⋮	⋮	⋮	⋮	⋮
243	Wall	Wall	Wall	Wall	Wall

What does a possible solution (chromosome) look like?

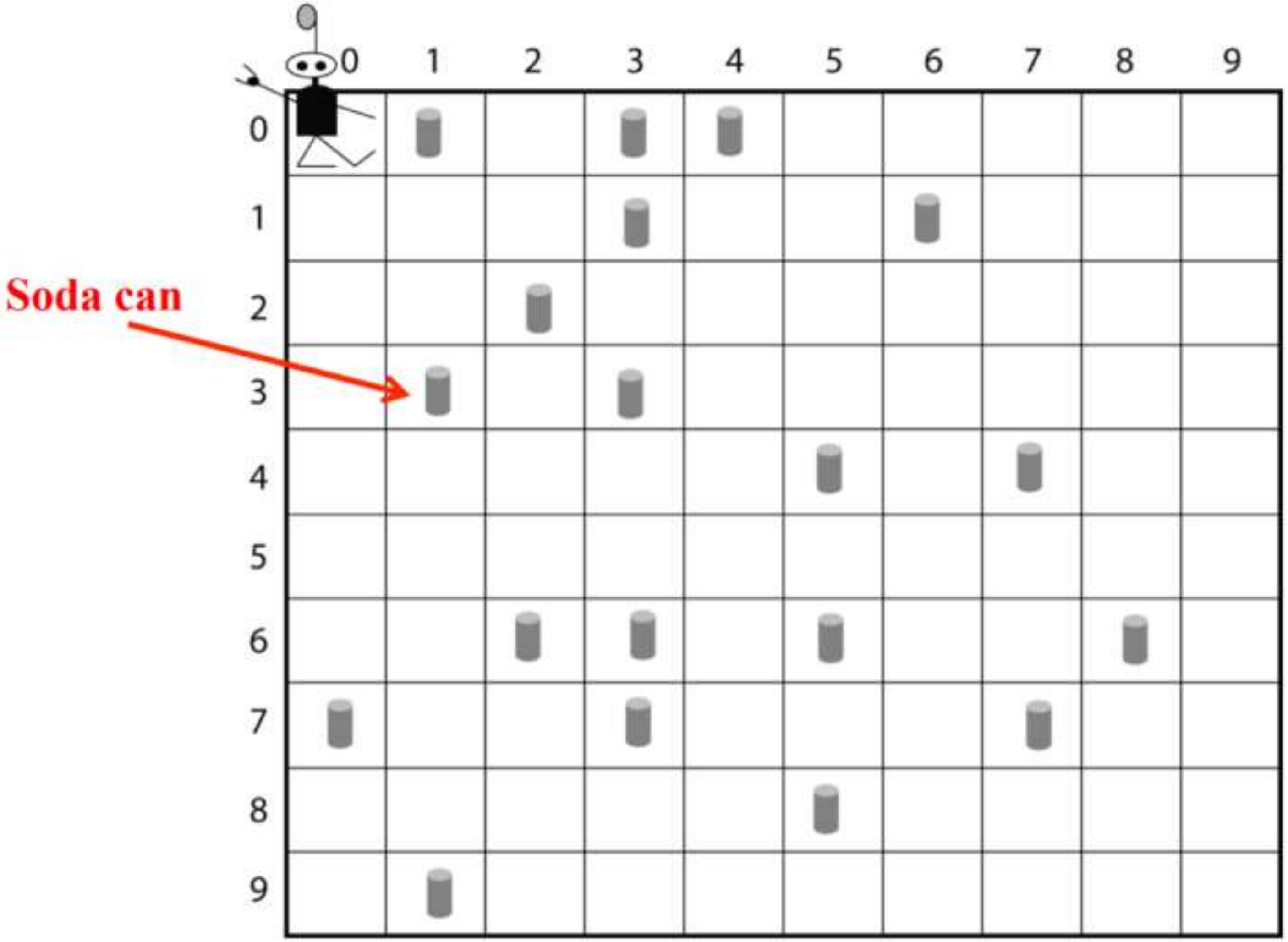
- 243 permutations of situations
- For any given situation what should Robby do?
- 7 choices -> Alleles for each gene -> use an enum to encode
  - North,
  - South,
  - East,
  - West,
  - Nothing,
  - PickUp,
  - Random

Chromosone – Possible solution

	Situation					Action
	North	*South*	*East*	*West*	*Current Site*	
1	Empty	Empty	Empty	Empty	Empty	MoveNorth
2	Empty	Empty	Empty	Empty	Can	MoveEast
3	Empty	Empty	Empty	Empty	Wall	MoveRandom
4	Empty	Empty	Empty	Can	Empty	PickUpCan
.	⋮	⋮	⋮	⋮	⋮	⋮
.	Wall	Empty	Can	Wall	Empty	MoveWest
.	⋮	⋮	⋮	⋮	⋮	⋮
243	Wall	Wall	Wall	Wall	Wall	StayPut



Example – Robby's situation



North	South	East	West	Current
Wall	Empty	Can	Wall	Empty

- What should Robby do?
- He looks up the step to take for this permutation
- This step is encoded in his chromosone

Encode

- 243 genes in the Chromosone -> array of 243 enum
  - Index indicates the situation
- Alleles:

  North

  South

  East

  West

  Nothing

  PickUp

  Random

How many possible chromosones?

- 243 values with 7 possibilities each = 7^{243} possible solutions! Impossible to check each one!

1. Generate 200 random solutions (i.e., programs for controlling Robby that tell him what to do in every solution)
2. For each solution, calculate fitness
   - average reward minus penalties earned on random environments
   - Not hardcoding to work in only 1 environment!!
3. Solutions mate and create offspring via crossover with random mutations
   - the fitter the parents, the more offspring they create.
4. Keep going back to step 2 for a set number of generations

1 – Random initial population of 200

Individual 1

South,PickUp,West,Nothing,PickUp,Random,North,North,North,East,Nothing,East,Random,PickUp,North,Nothing,North,West,South,Random,PickUp….

Indiv 2:

PickUp,West,Random,East,South,Random,Random,West,South,South,PickUp,East,Random,East,East,West,Nothing,East,West,Nothing,

2- Calculate fitness

- Fitness in 1 grid = final score after 200 moves

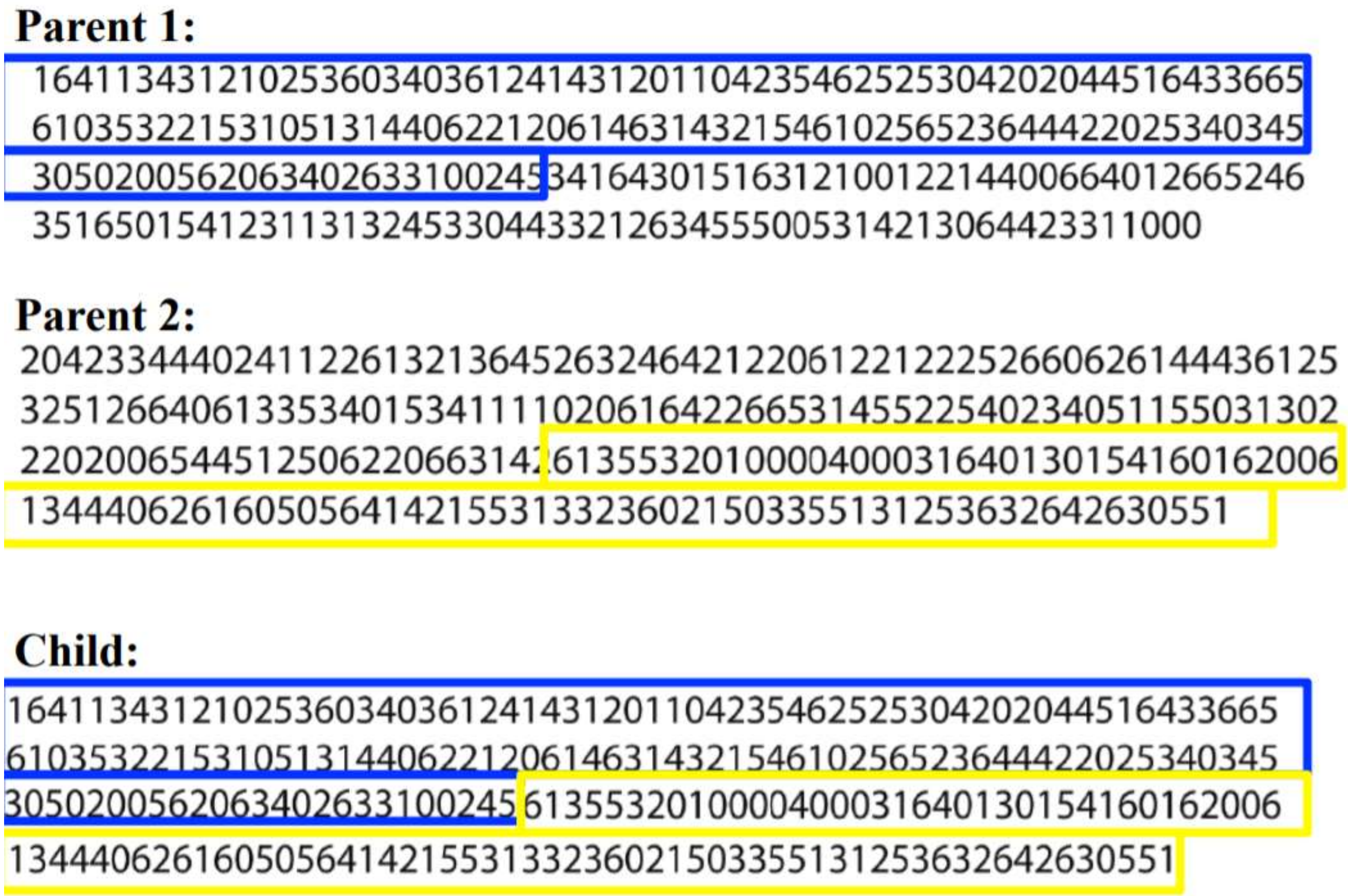
- To ensure that the solution works over many grids: generate 100 random 10 x 10 grids
  - 50 sites have a can (randomly placed)
- Calculate average fitness over 100 random grids for each solution

3 – Select parents

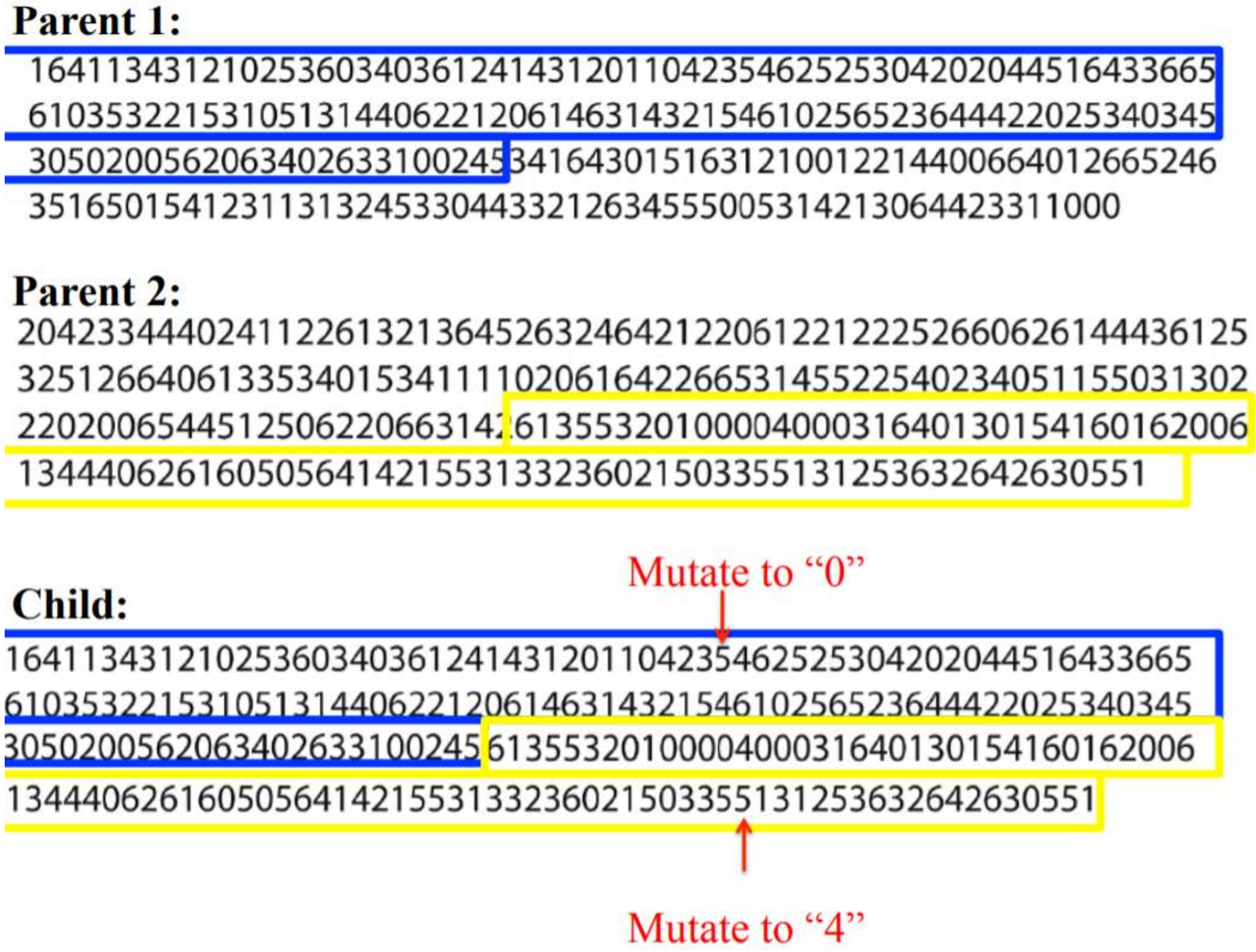
- Choose two parent individuals from the current population probabilistically based on fitness. That is, the higher an individual's fitness, the more chance it has to be chosen as a parent.

4 – Reproduce with crossover

- Random crossover point



And Random mutation



5- Offspring and Elite are placed in the new population and the old population dies.

:’(