Multi-objective Optimization Methods:

Meta-Heuristic Methods

Azarakhsh Rafiee
a.rafiee@tudelft.nl

Meta-Heuristics

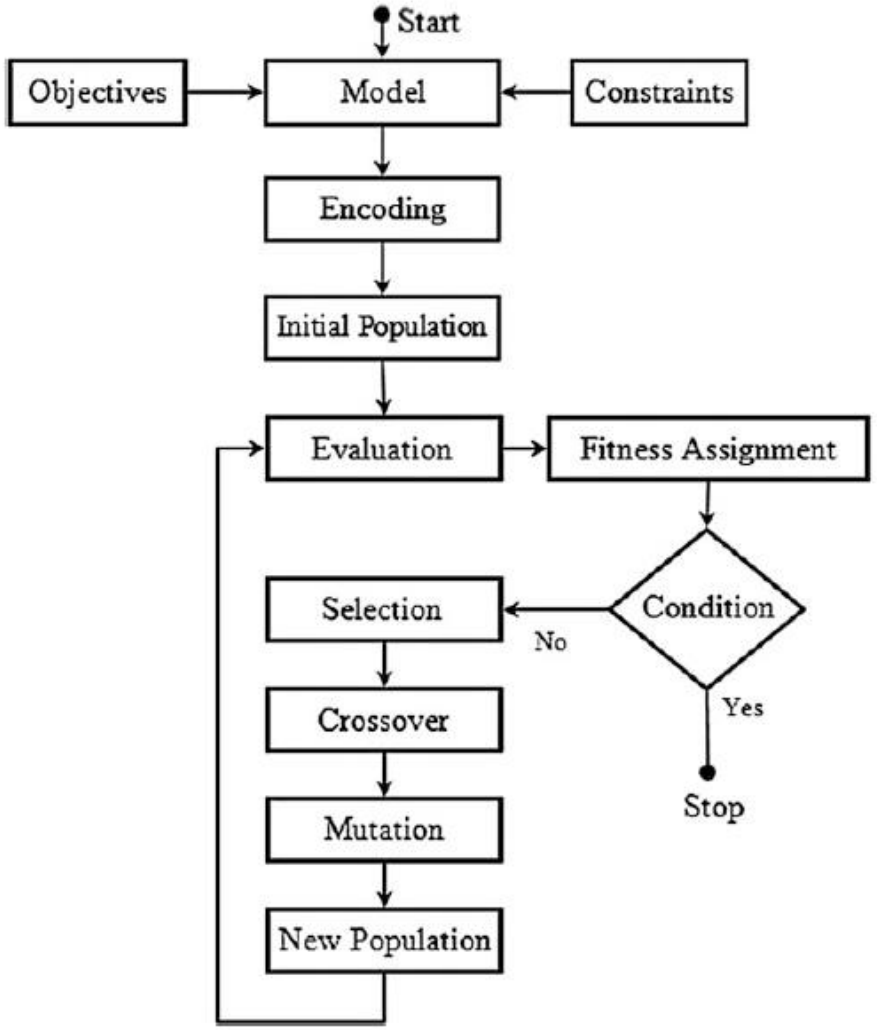
Common methods in Multi-objective optimization for GIS applications:

- Genetic Algorithms
- Simulated Annealing
- Swarm Intelligence
  - Ant Colony Optimization
  - Particle Swarm Optimization

Genetic Algorithms

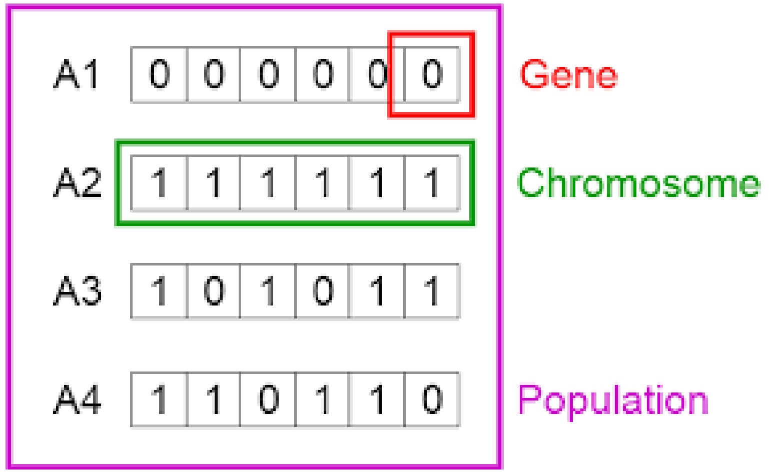
- Evolutionary algorithms inspired by biological principles of natural selection and survival of the fittest
- Operate on a population of individuals (potential solutions) instead of single solutions (metaheuristic search is performed in a parallel manner)
- The algorithms produce a set of improved solutions at each generation (iteration)
- Evolution of populations of individuals that are better suited to their environment (optimization problem) than the individuals that they were created from.

Genetic Algorithms
flowchart of genetic procedure

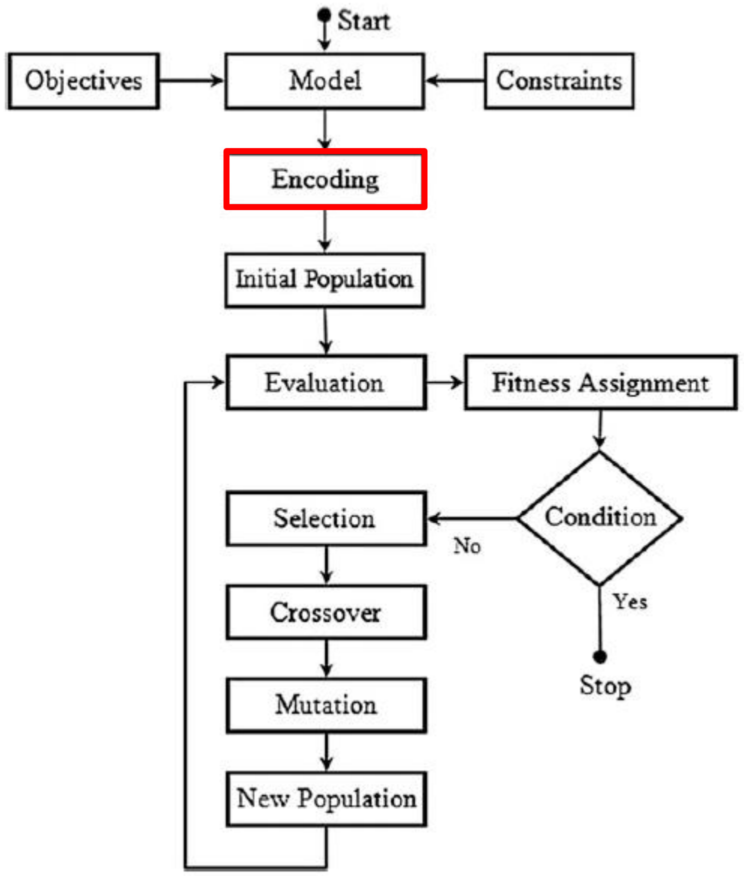


Genetic Algorithms

- Chromosome or an individual: a solution vector of decision variables
- Chromosomes are made of discrete units. These units are called genes



Genetic Algorithms Encoding

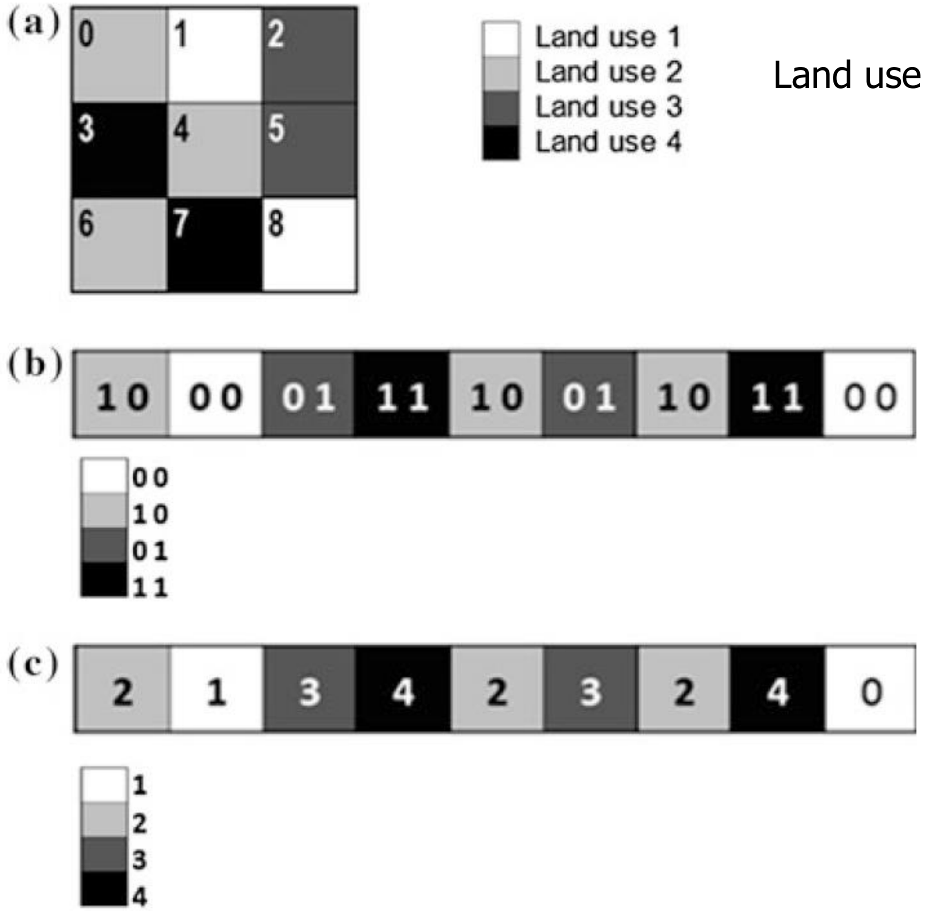


Encoding

- Common GIS-based genetic algorithm encodings:
  - binary representation
  - discrete representation
  - order- (or permutation) based representation
- The choice of the encoding strategy is problem dependent

Example: Encoding potential land use plan

The number of possible land uses defines the length of the gene. For example, if four land uses are considered (e.g., residential, commercial, industrial, and recreational), then a gene of two bits is required to represent all four types of land use; that is, 00, 10, 01, and 11.



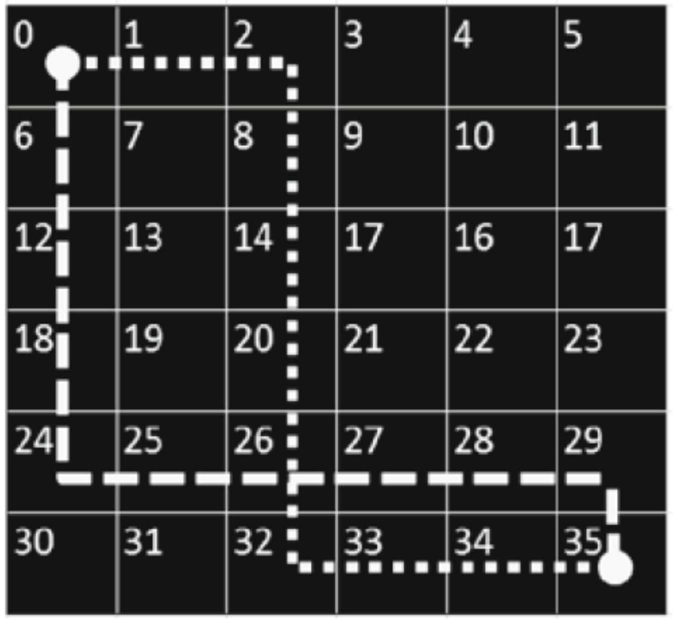
Land use pattern (ID = 0, 1, 2, …, 8)

binary encoding

integer encoding

Example: Encoding for a routing problem

(a)



Two alternative corridors (routes) between location (raster) 0 and location 35 in the geographical space

▪▪▪▪ Corridor *A*

▭▭▭ Corridor *B*

(b)

0	1	2	8	14	20	26	32	33	34	35

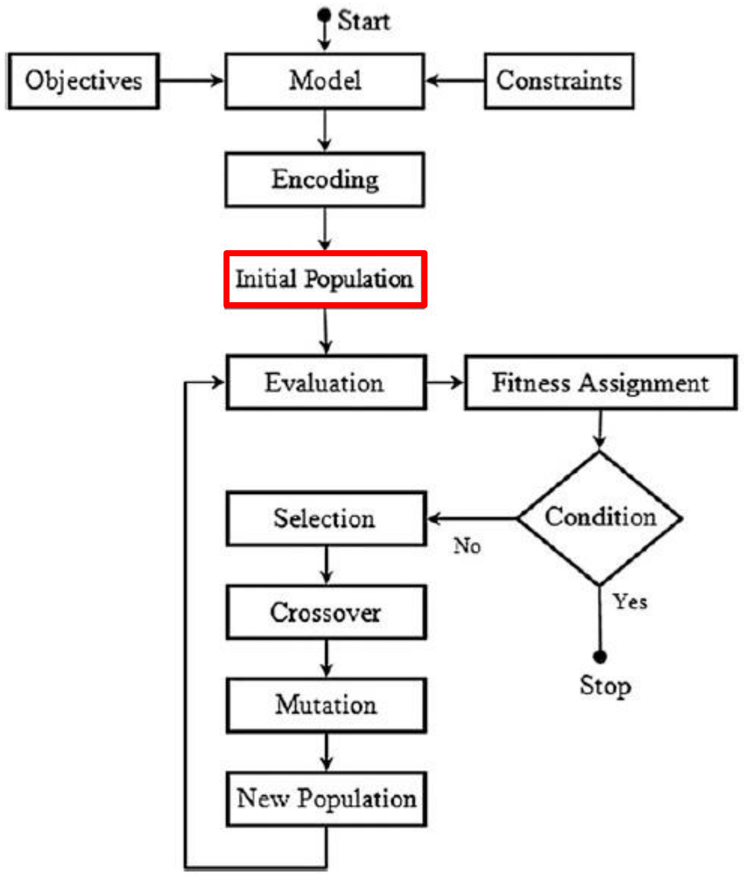
chromosome representation of corridor A

(c)

0	6	12	18	24	25	26	27	28	29	35

chromosome representation of corridor B

Genetic Algorithms
Initialization



Initialization

- The quality of results generated by a genetic algorithm depends on the size of the initial population and the way it is constructed.

- The main consideration in the process of constructing the initial population is **diversification**

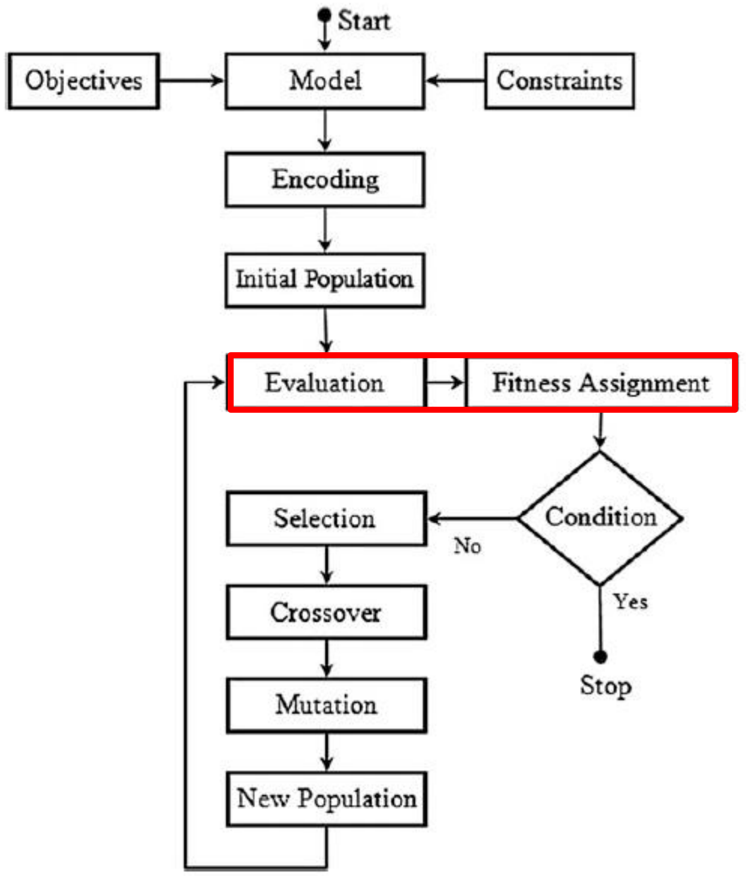
- If the population is not well diversified, a premature convergence (a convergence toward a suboptimal result or local optimum) can occur.

Initialization

Common methods to construct the initial population:

- random generation
- sequential diversification
- parallel diversification
- heuristics

Genetic Algorithms
Evaluation and Fitness Assignment



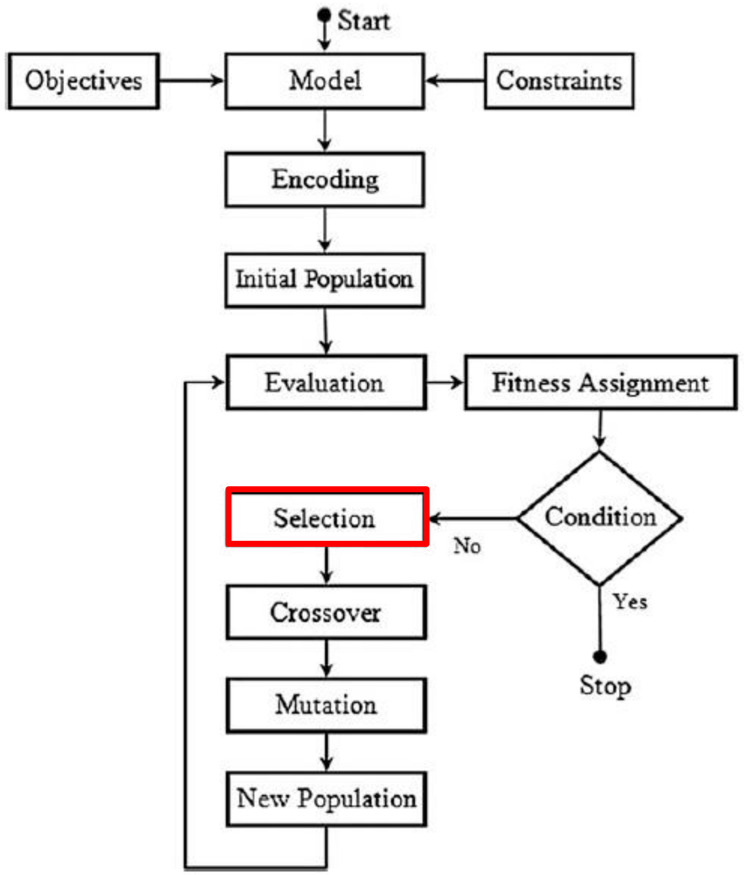
Evaluation and Fitness Assignment

- Assign a fitness value to measure the quality of each solution
- In general, the techniques for fitness assignment are based on the concepts and methods of the conventional multiobjective optimization procedures: example: weighted-sum approach
- The weighted-sum approach transforms the multicriteria decision problem into a single criterion one.

$$\text{maximize } F(x) = \{\Sigma\, w_k\, v(f_k(x))\},$$

$$\text{subject to: } x \in X,\ w_k \geq 0 \ \text{for } k = 1, 2, \ldots, n.$$

Genetic Algorithms
Selection



Selection

- Survival-of-the-fittest principle to choose “high quality” individuals (solutions) from the current population

- At each successive generation: the higher the fitness of the individual, the higher the probability of it being selected into a mating pool (temporary population)

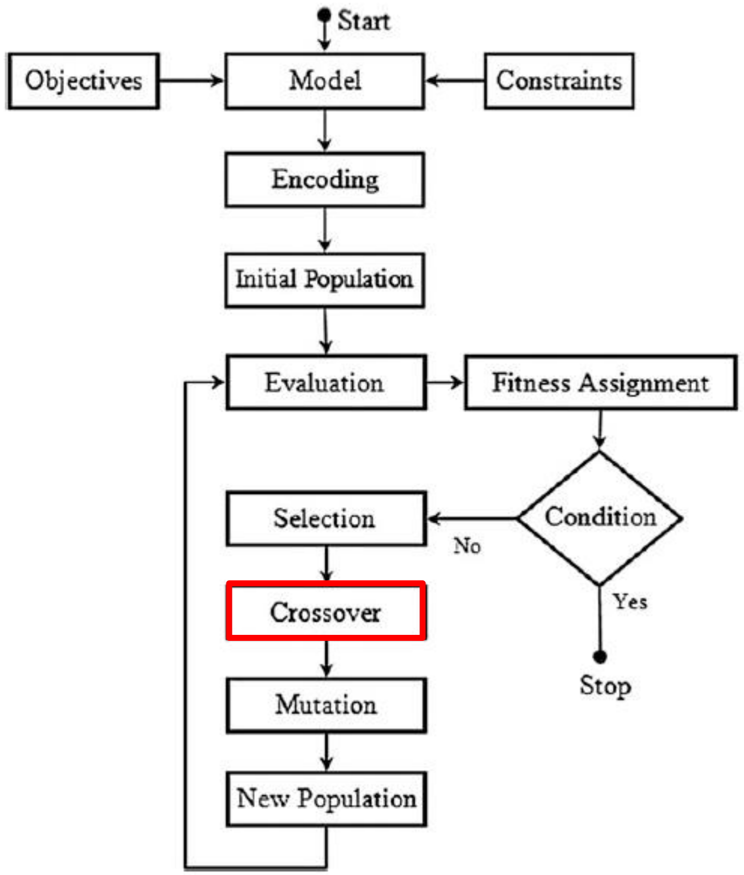
Selection

selection procedures can be grouped into two categories:

- fitness proportionate selection (e.g. roulette-wheel selection)
  - chromosome is selected from the population with a probability proportional to its fitness
  - It could reduce the search space if there are outliers (individuals with exceptionally higher fitness values) in the population
  - Nearly equal chance for each solution to be selected, if the interval of fitness function values of the solutions is relatively small

- rank-based selection (e.g. tournament selection methods)
  - a random selection of s individuals for a tournament against each other

Genetic Algorithms
Crossover

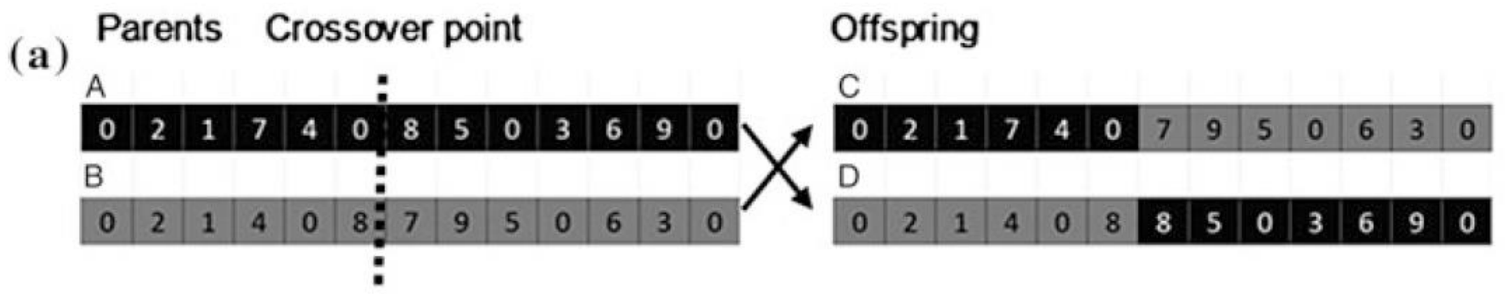


Crossover

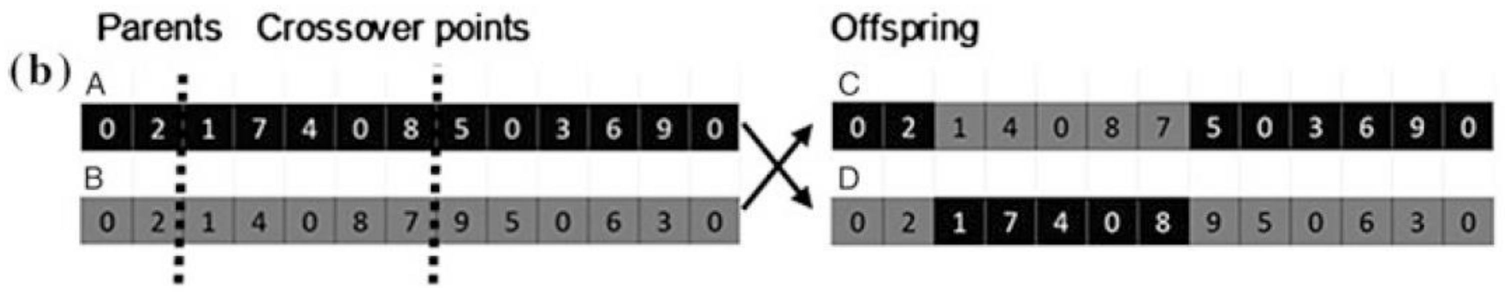
- Individuals are recombined (or crossover) to make new offspring

- Common crossover methods for GIS-based applications:
  - one-point crossover
  - two-point crossover
  - uniform crossover

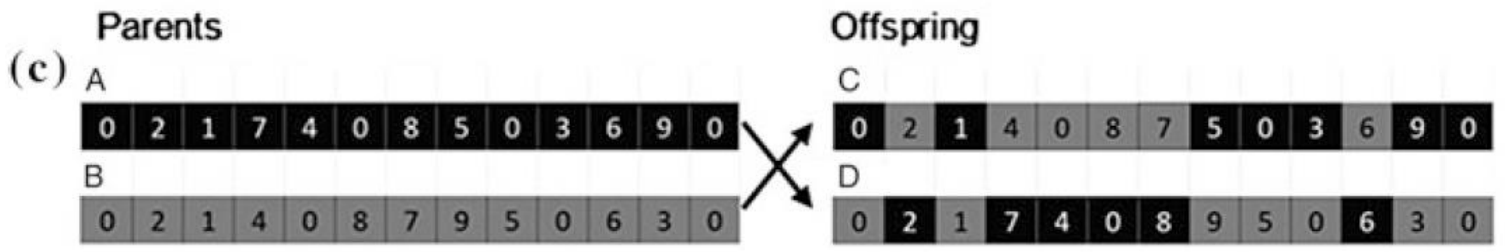
Common crossover methods



one-point crossover

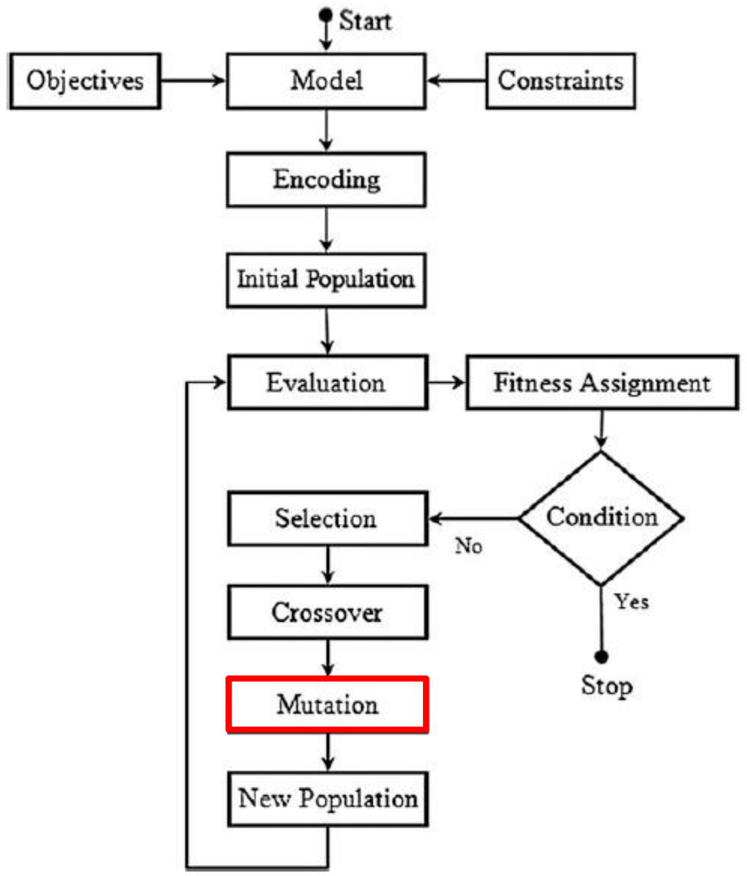


two-point crossover



uniform crossover

Genetic Algorithms
Mutation



Mutation

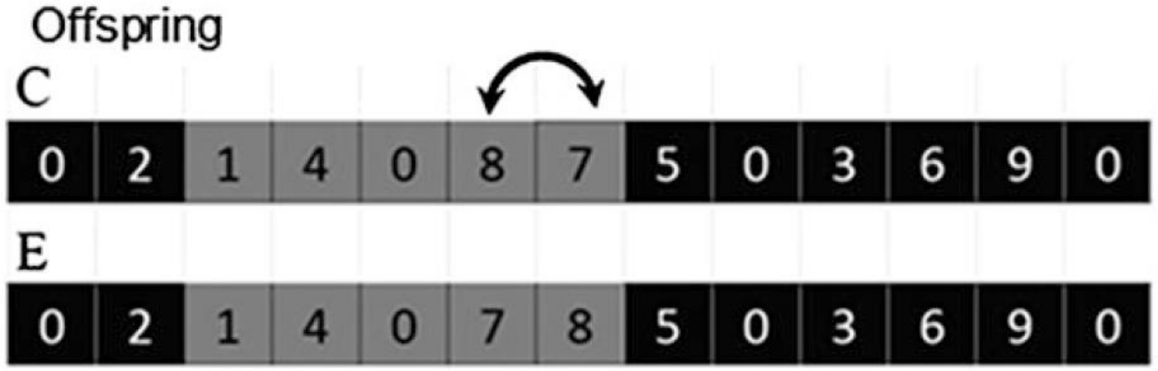
- Mutation procedures operate on a single solution (offspring)
- It aims at **diversifying** the search (that is, maintaining genetic diversity from one generation to the next)
- preventing premature convergence of the genetic algorithm: preventing all solutions in a population to fall into a local optimum)
- Altering one or more gene values of individuals in the current generation to create new solutions

Mutation

mutation probability: P_m recomendation:$0.001 < P_m < 0.01$

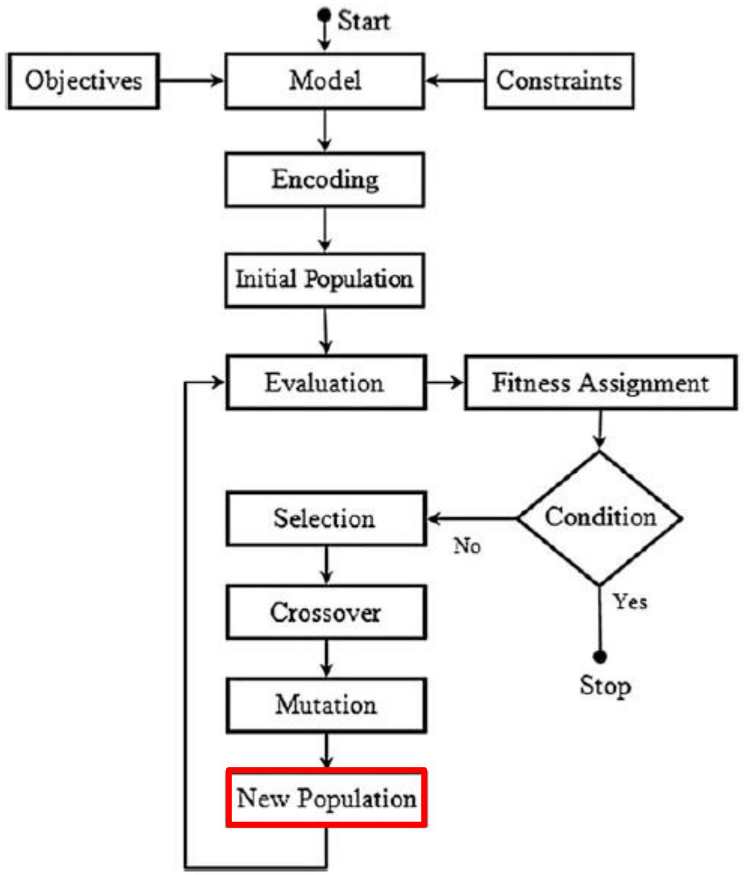
For instance:

P_m =0,01→% of the genes of an offspring solution (e.g. land use patterns) are randomly changed to a new value



Mutation: swapping two randomly selected nodes

Genetic Algorithms New Population



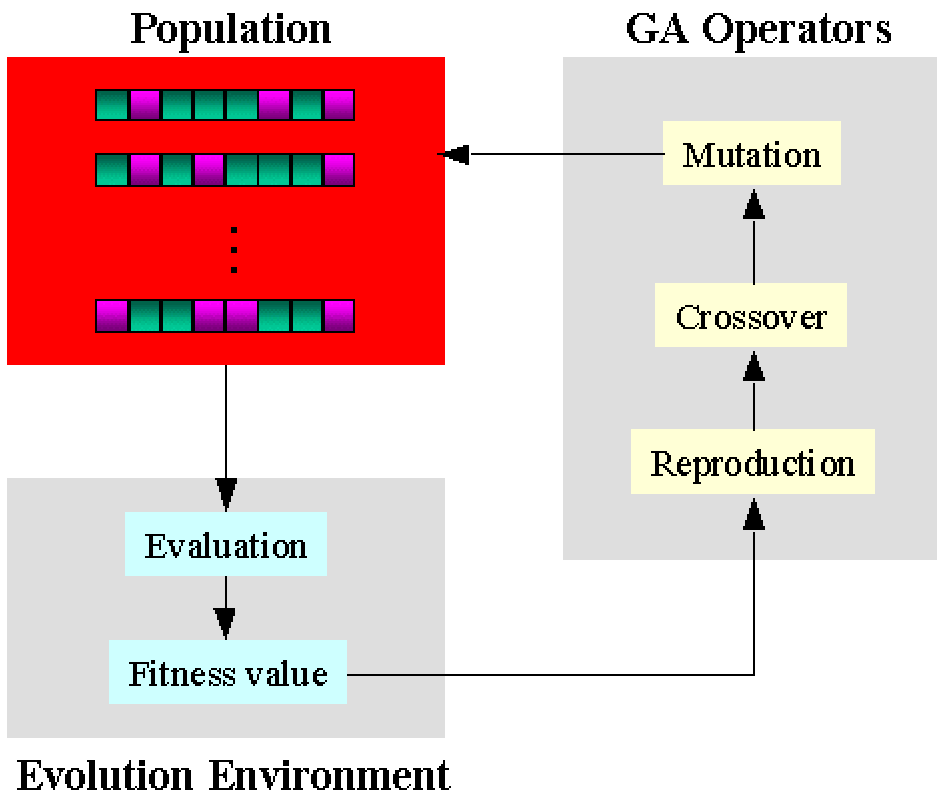
New Population

Replacement strategies:

- delete-all approach
  - parent population is eliminated and replaced by the offspring population

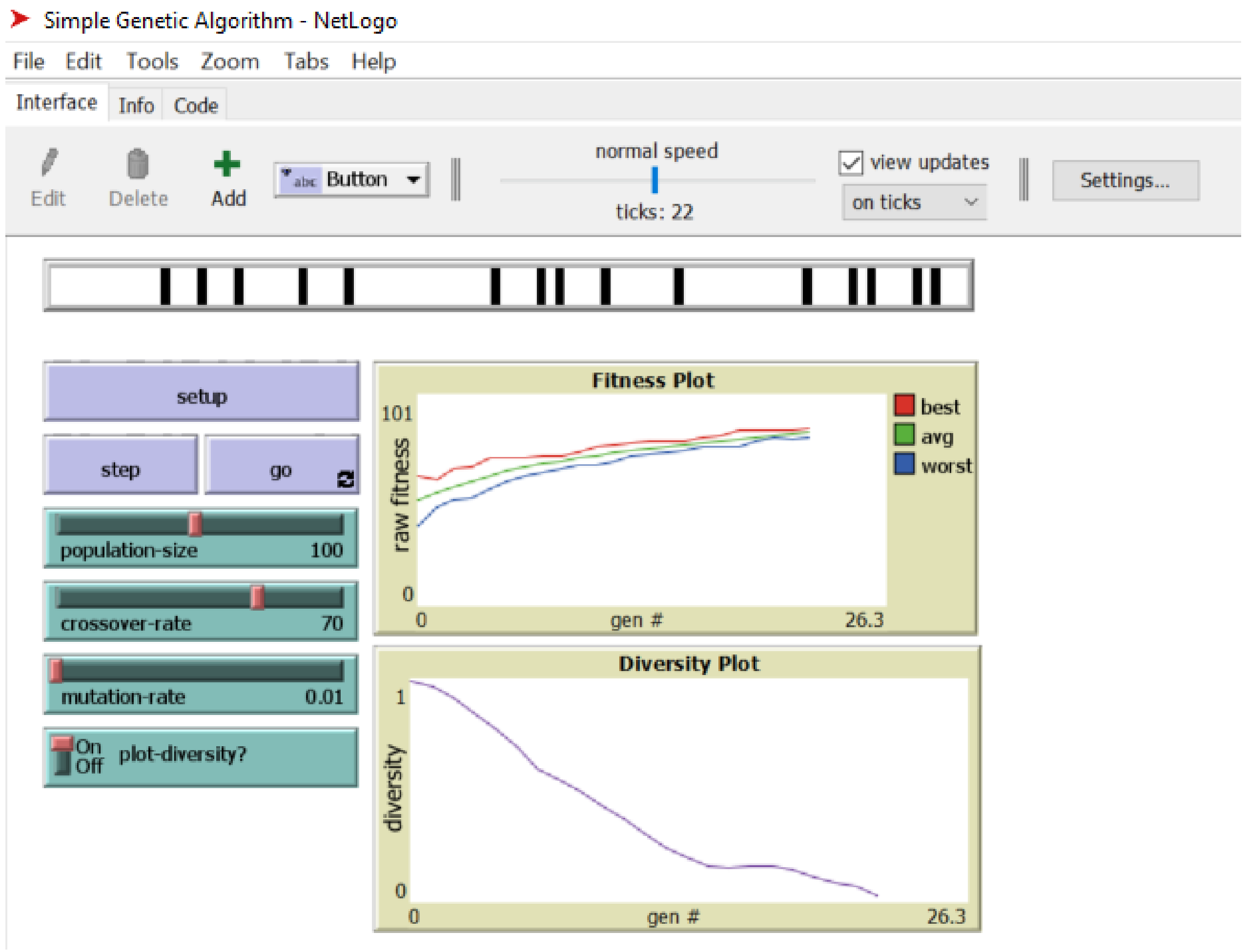
- steady-state strategy
  - a portion of parent solutions is deleted and replaced by the offspring individuals.

Genetic algorithm evolution flow



Genetic Algorithm Evolution Flow

Netlogo Simple Genetic Algorithm

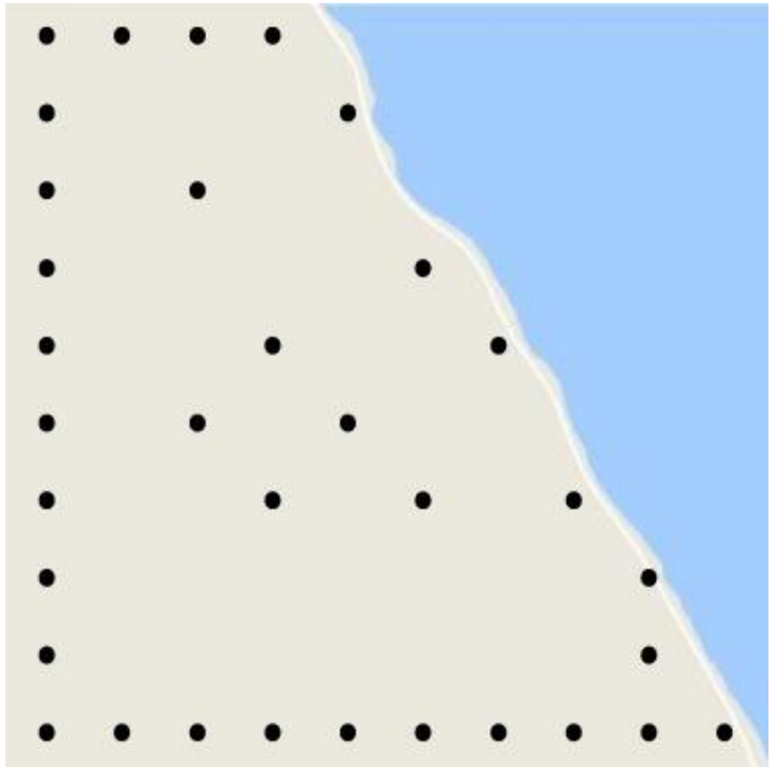


population-size 100

crossover-rate 70

mutation-rate 0.01

Optimization of wind turbines siting in a wind farm using genetic algorithm

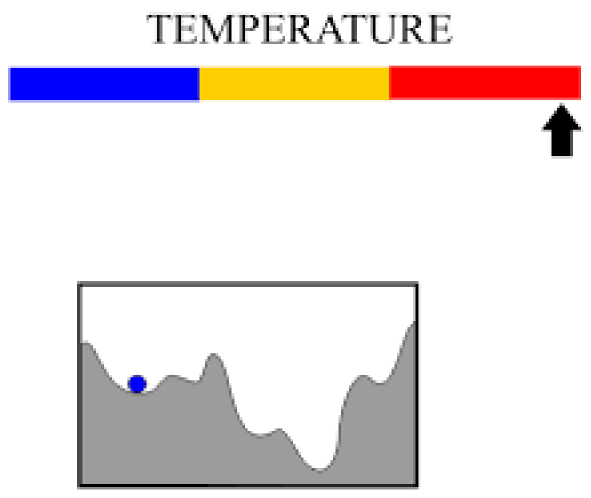


Source: Abdelsalam, A. M., & El-Shorbagy, M. A. (2018). Optimization of wind turbines siting in a wind farm using genetic algorithm based local search. Renewable energy, 123, 748-755.

Demo: Landuse modelling through "Structure from randomness 2‟ in Netlogo

- Not Genetic algorithm

Simulated Annealing (SA)



- Mimics the process of arranging atoms when a material (steel or glass) is heated and then slowly cooled
- During the process of crystallization, the temperature controls the arrangement of atoms in their lowest-energy configuration
- Eventually, when the material approaches zero temperature, the atoms approach their minimum energy state

Simulated Annealing (SA)

- SA algorithm imitates the annealing process to solve an optimization problem
- The goal is to bring the system, from an arbitrary *initial state*, to a state with the minimum possible energy.
- Generates moves randomly in the solution space searching for a solution that minimizes the value of an objective function.
- The algorithm always accepts moves that decrease the value of the objective function.
- However, changes that increase the value of the objective function are accepted with a small probability that depends on a control parameter called the temperature.

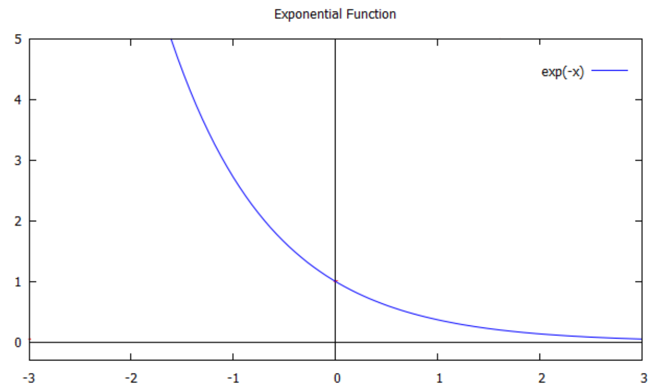
Simulated Annealing (SA)

- probability of accepting a worse point:

$$\exp(-\Delta E/T)$$

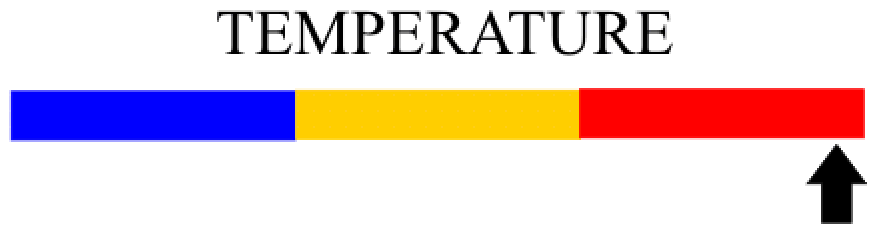
energy difference:
(the difference in the value of objective function between the current and
next point)

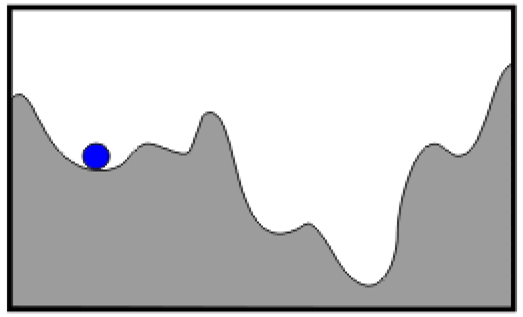
Temperature (Control parameter)



As the SA algorithm progresses, the probability that such moves are accepted decreases, giving worse solutions a lesser chance of replacing the current solution. This results in convergence to an optimal point.

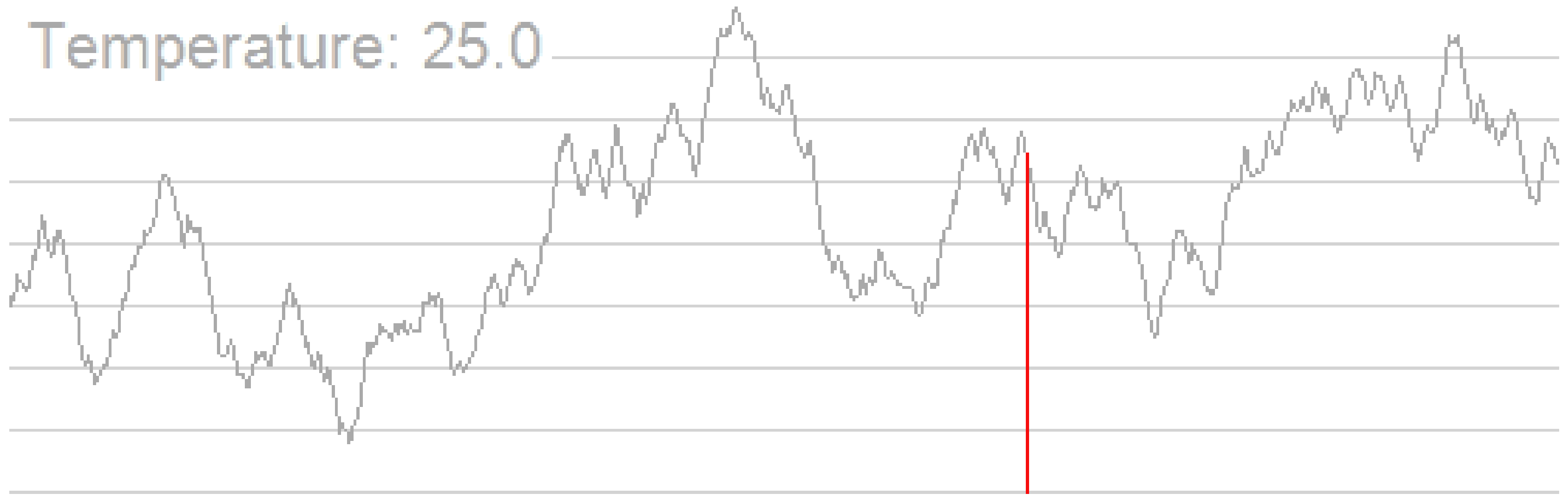
Simulated Annealing



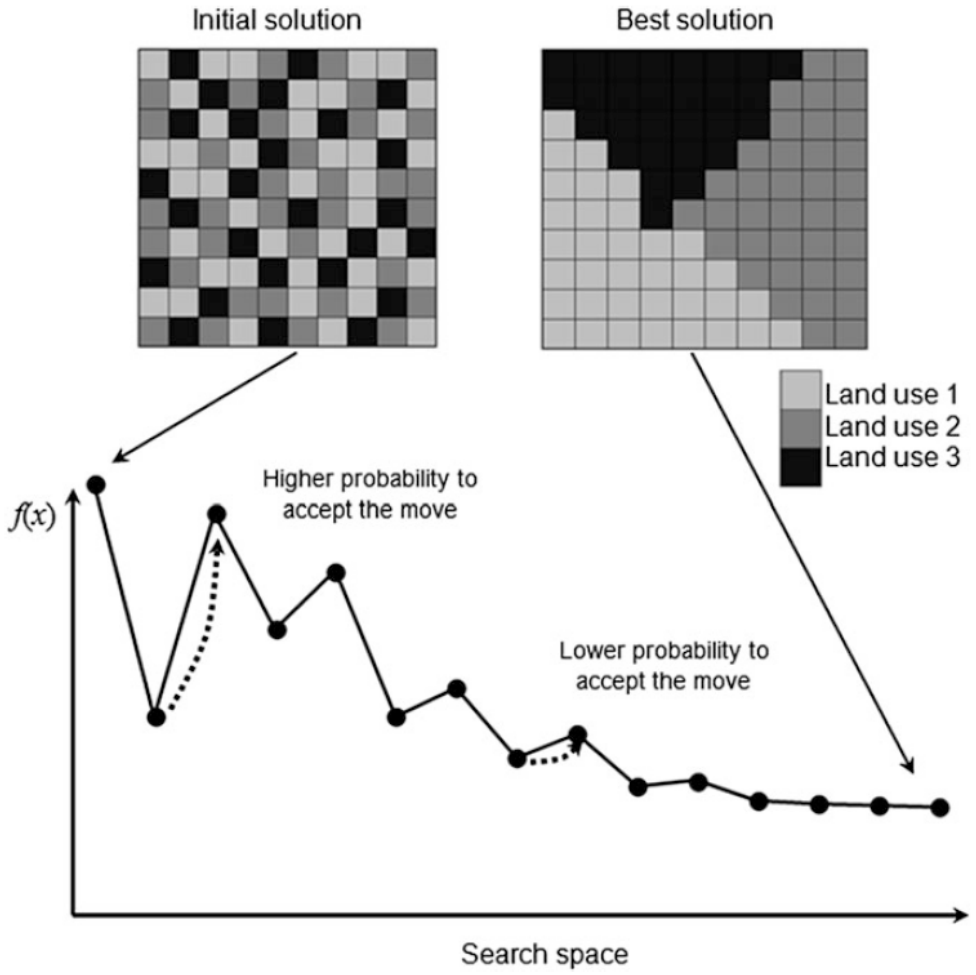


Simulated Annealing

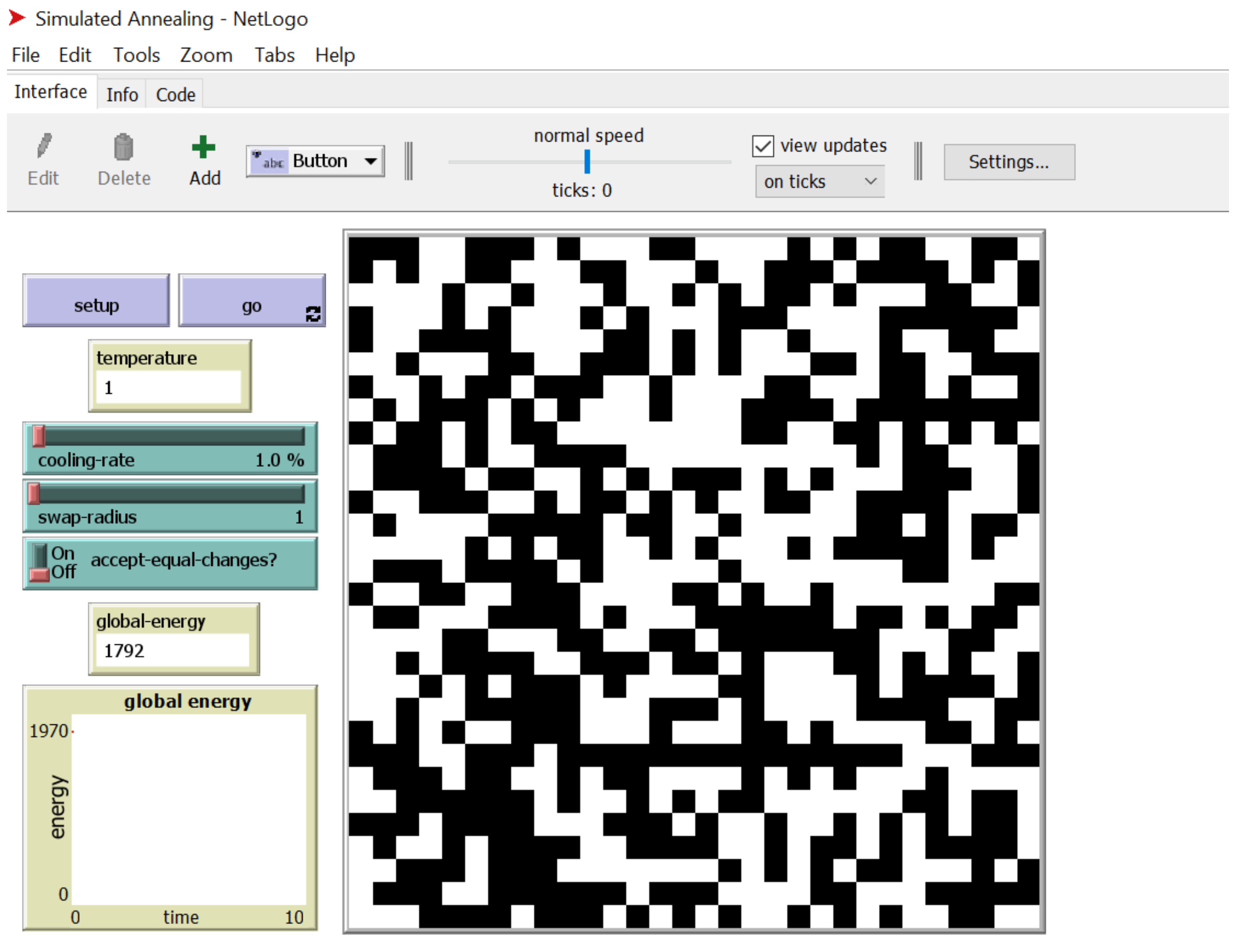
Simulated annealing searching for a maximum. The objective here is to get to the highest point.



Simulated Annealing: example of land use pattern



Simulated Annealing: NetLogo Demo



Simulated Annealing - NetLogo

File Edit Tools Zoom Tabs Help

Interface Info Code

temperature
1

cooling-rate 1.0 %

swap-radius 1

On Off accept-equal-changes?

global-energy
1792

Swarm Intelligence

- Swarm intelligence optimization methods are inspired by social behaviours in flocks of birds, schools of fish, herds of buffalo, colonies of ants, and so forth.
- A colony or swarm is a self-organized, multi-agent system in which the individuals (agents) cooperate to accomplish complex tasks.
- This cooperation is distributed among the entire population, without any centralized control.

Swarm Intelligence

- The global pattern of agents emerges from local interactions between agents, which occur through direct (agent-to-agent) or indirect (via the environment) communication.

- Each agent follows a set of rules influenced by locally available information.

(Common) swarm intelligence inspired optimization algorithms

- **Ant Colony Optimization**
- **Particle Swarm Optimization**

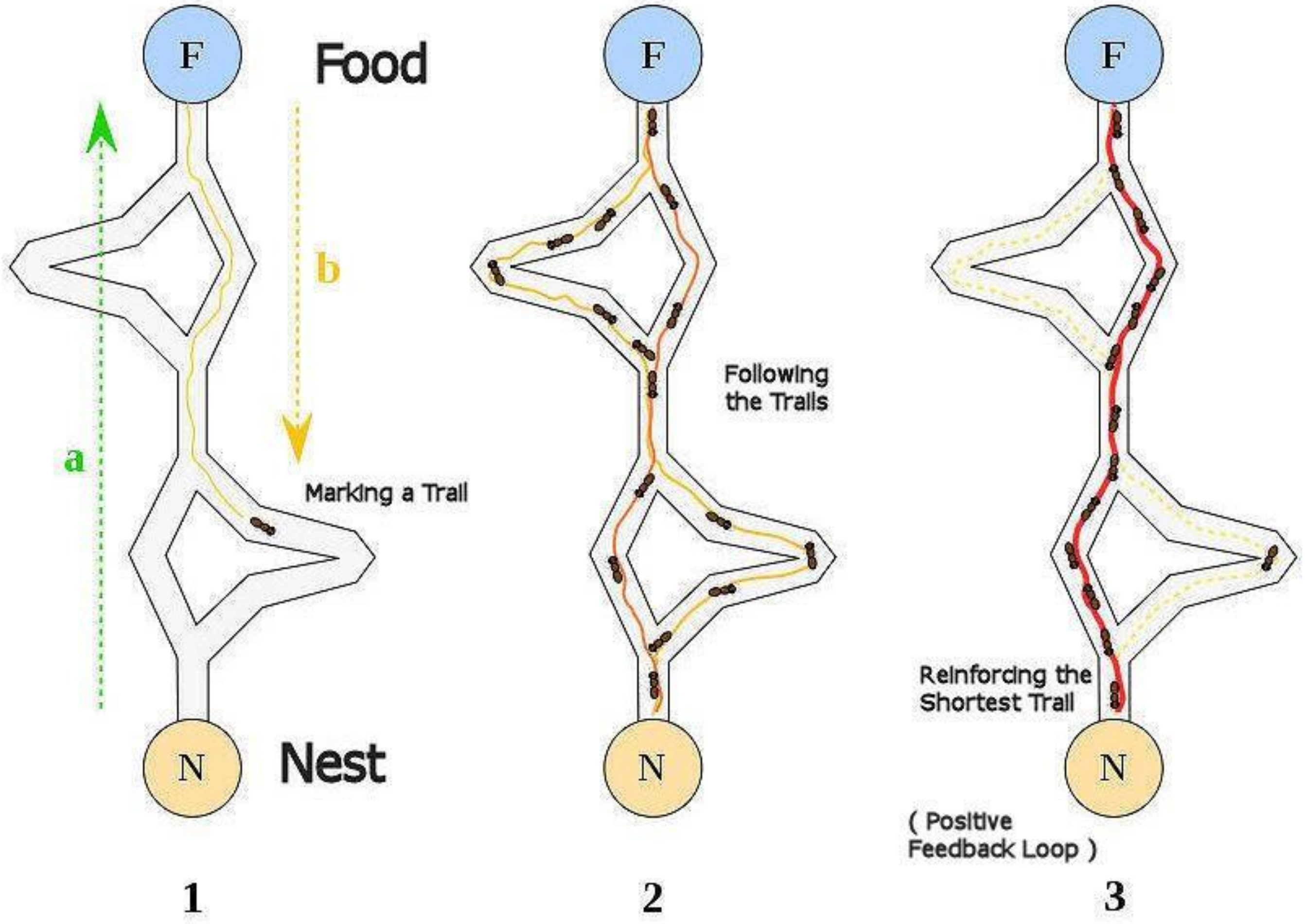
Ant Colony Optimization (ACO)

- Based on an imitation of the behaviour of ants in their search for food
- Although it seems that each ant in a colony has its own agenda, the colony (population) behaves as a self-organizing system

Ant Colony Optimization (ACO)

- Ants form and maintain a line to their food source by laying a trail of pheromone (a chemical to which other ants are sensitive).
- They deposit a certain amount of pheromone while walking, and each ant prefers to follow a direction marked by high concentration of pheromone.
- This enables the colony of ants to quickly find the **shortest route**.

Ant Colony Optimization

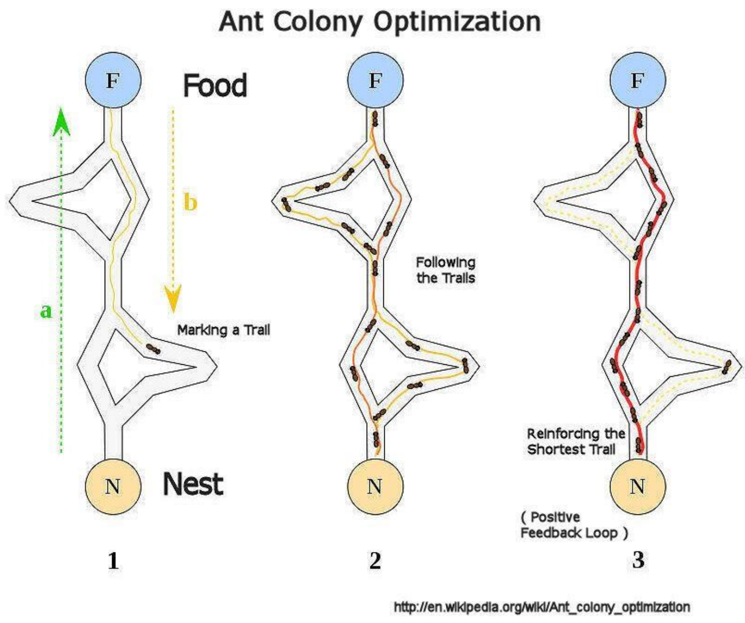


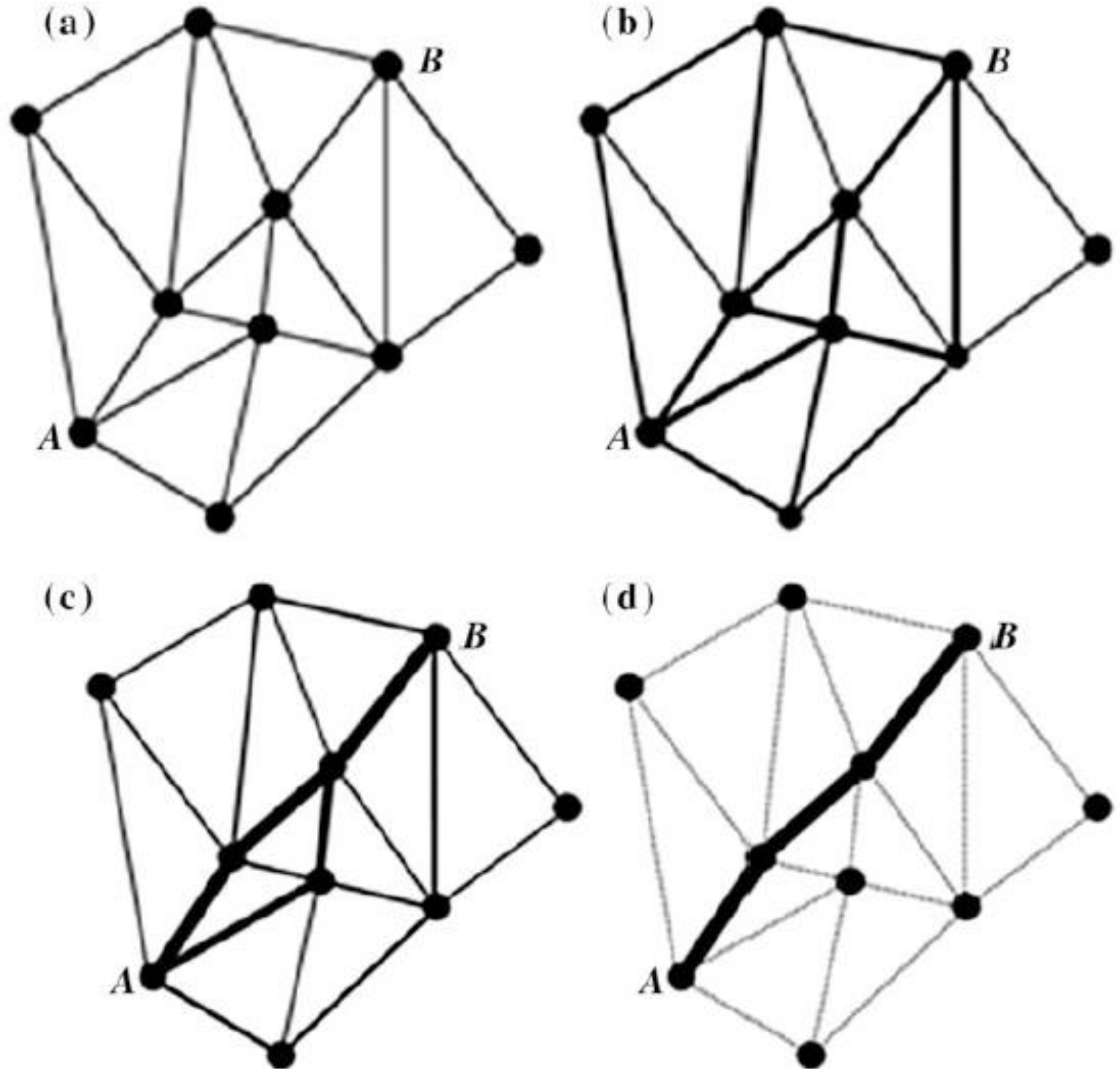
http://en.wikipedia.org/wiki/Ant_colony_optimization

- At the beginning of the searching process, the ants explore all the paths or routes.

- Gradually the shortest path is marked with more pheromone than other routes because the self-regulating mechanism attracts more and more ants to the route characterized the highest level of pheromone.

- Sooner or later, nearly all the ants are choosing the shortest path


Ant Colony Optimization
F
Food
F
F
a
b
Marking a Trail
Following the Trails
Reinforcing the Shortest Trail
N
Nest
N
N
(Positive Feedback Loop)
1
2
3
http://en.wikipedia.org/wiki/Ant_colony_optimization


(a)
A
B
(b)
A
B
(c)
A
B
(d)
A
B

Ant Colony Optimization (ACO)

- Main idea behind the ACO meta-heuristics: use repeated simulations of a set of software agents (mobile agents inspired by real ant behaviour) to generate solutions to a given problem.

- At each iteration, the agents collect relevant information, which is used in subsequent iterations to **direct their search** for the best solution.

Ant Colony Optimization (ACO)

- The information for permutation problems (such as shortest path, vehicle routing, and traveling salesman problems) is usually encoded in an n by n **pheromone matrix** $[\tau_{ij}]$, *i*, *j* = 1, 2, …, n.

- In any given iteration, an ant chooses its route **probabilistically**.

- The choice is based on the **pheromone level** and the **distances** between two nodes, *i* and *j*.

Ant Colony Optimization (ACO)

Probability:

$$p_{ij} = \frac{\tau_{ij}^{\alpha}\eta_{ij}^{\beta}}{\sum_{p\in S}\tau_{ip}^{\alpha}\eta_{ip}^{\beta}}, \quad \text{for } j \in S;$$

Ant Colony Optimization (ACO)

Probability:

$$p_{ij} = \frac{\tau_{ij}^{\alpha}\eta_{ij}^{\beta}}{\sum_{p\in S}\tau_{ip}^{\alpha}\eta_{ip}^{\beta}}, \quad \text{for } j \in S;$$

pheromone concentration on the route between i and j

desirability of the route between nodes i and j (the value of ηij is typically inversely proportional to the distance between i and j; that is, ηij = 1/dij)

Parameter determining the relative influence of the pheromone concentration

Parameter determining the relative influence of the route desirability

Ant Colony Optimization (ACO): pheromone concentration

- pheromone concentration can change with time due to the process of evaporation.
- pheromone update→ increase the pheromone values associated with good solutions, and to decrease those that are associated with bad ones
- Two steps:

1. Pheromone reduction: $\tau_{ij} = (1-\rho)\tau_{ij} \quad \text{for } i = 1, 2, \ldots, n;$
    - reduction rate, ρ ∈ (0,1):

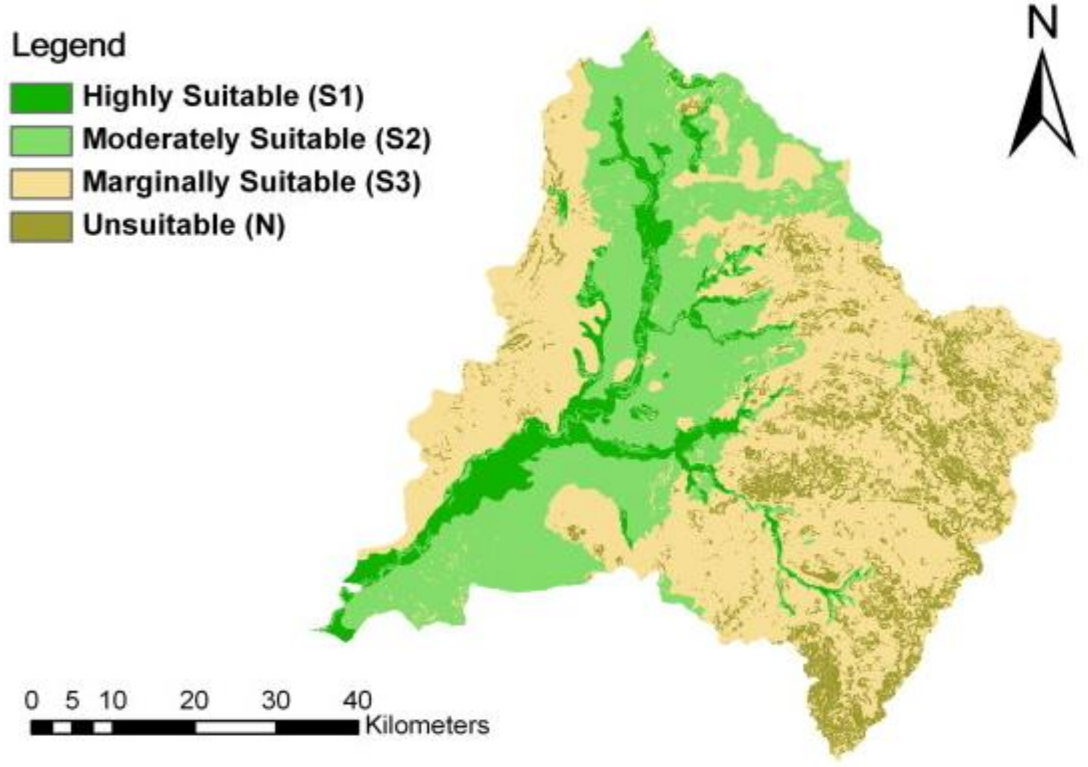
- 2. Pheromone update associated with the best solution:

$$\tau_{i\pi(i)} = \tau_{i\pi(i)} + \Delta, \text{ for } i = 1, 2, \ldots, n;$$

Ant Colony Optimization application in GIS Example:

- site selection problems
- path-finding problems
- problems of zoning design
- searching for the best location of public facilities
- land-use suitability assessment
- land use allocation problem
- etc.

Example: ArcGIS-based ACO for multicriteria land-use suitability assessment



Source: Yu, J., Chen, Y., & Wu, J. (2011). Modeling and implementation of classification rule discovery by ant colony optimisation for spatial land-use suitability assessment. Computers, Environment and Urban Systems, 35, 308–319.

Particle Swarm Optimization (PSO)

Particle Swarm Optimization (PSO)

- Particle swarm optimization (PSO) is inspired by the social behaviour in flocks of birds, schools of fish, and swarms of insects such as termites, bees, and wasps.

- An individual (e.g., a bird, fish) is referred to as a particle (or an agent)

- Each individual in a swarm behaves according a **combination** of its own intelligence and the collective intelligence of the population.

- In PSO, individual particles of a swarm represent potential solutions (e.g., spatial patterns of land uses).

Particle Swarm Optimization (PSO)

- Each particle has its own velocity and position (location) in a multi-dimensional search space
- The particles move through the space seeking a good solution
- The particles communicate their current locations to neighbouring particles.
- The position of each particle is adjusted according to its velocity (i.e., rate of change) and the difference between its current position and the best position found by its neighbours, and the best position it has found so far

Particle Swarm Optimization (PSO)

Velocity is defined by three elements:

- ‘inertia’ based on the current velocity value of the element
- ‘personal influence’ based on the solution element of the particle’s own best solution so far (called local best)
- ‘social influence’ based on the solution element of the best particle found in the population during the search process (called global best)

As the model is iterated, the swarm focuses more and more on an area of the search space containing high-quality solutions

Particle Swarm Optimization (PSO)

$$V_i^{t+1} = W.V_i^t + c_1 U_1^t\left(P_{b_1}^t - P_i^t\right) + c_2 U_2^t\left(g_b^t - P_i^t\right)$$

V_i: Velocity of the particle or agent

A: Population of agents

W: Inertia weight

C_1: cognitive constant

C_2: social constant

U_1 & U_2 : random numbers

P_i: Position of the particle or agent

P_b: Personal Best

g_b: global Best

Particle Swarm Optimization (PSO)

Diversification: Searches new solutions

Intensification: Explores the previous solution

$$V_i^{t+1} = \overbrace{\boxed{W.V_i^t}}^{\text{Diversification}} + \overbrace{\boxed{c_1 U_1^t \left(P_{b_1}^t - P_i^t\right)} + \boxed{c_2 U_2^t \left(g_b^t - P_i^t\right)}}^{\text{Intensification}}$$

Intertia: Makes the particle move in the same direction and with the same velocity

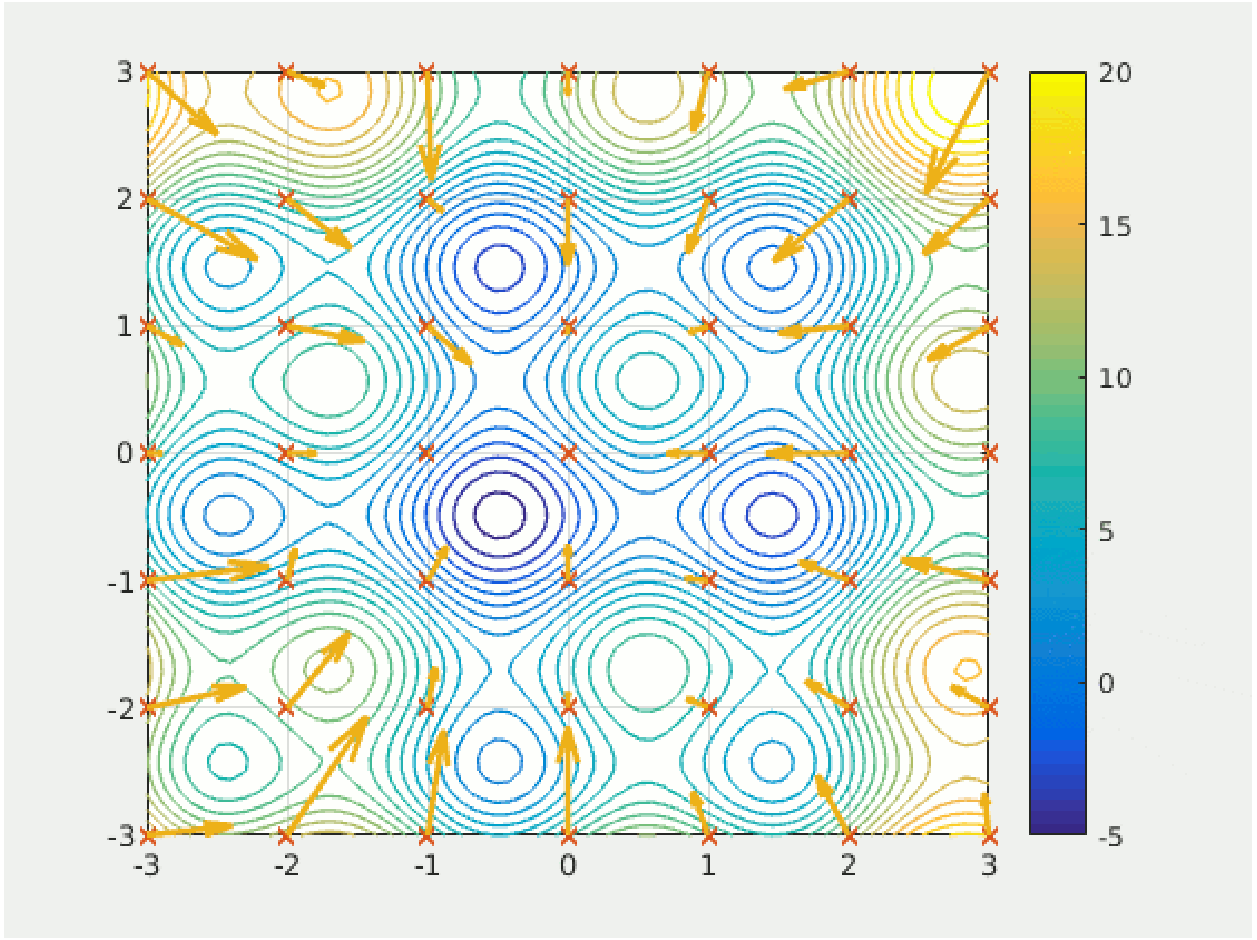
Personal Influence: Improves the individual. Makes the particle return to a previous position, better than the current.

Social Influence: Makes the particle follow the best neighbouring direction

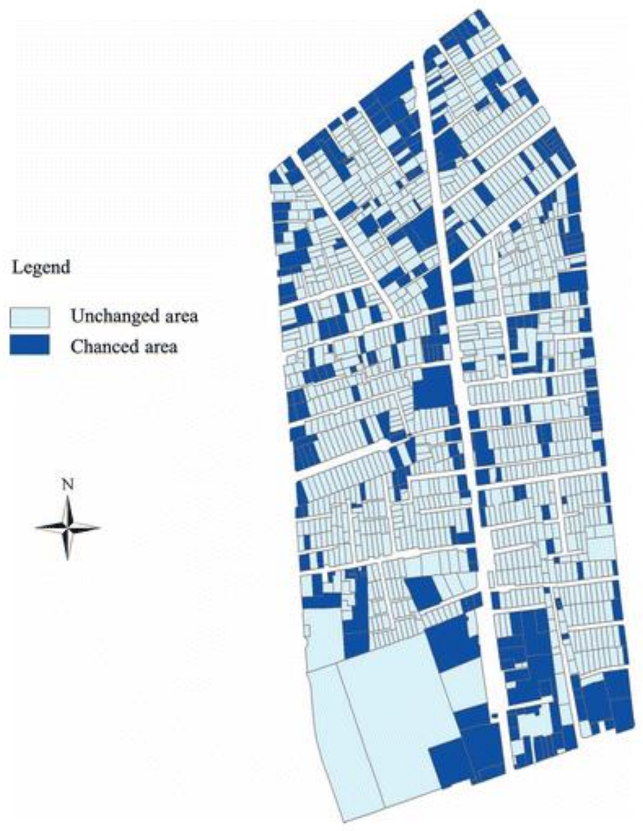
Particle Swarm Optimization (PSO)

1. Create a ‘population’ of agents (particles)
2. Evaluate each particle’s position considering the objective function
3. If a particle’s present position is better than its previous best position, update it.
4. Find the best particle (according to the particle’s last best places).
5. Update particles’ velocities: $V_i^{t+1} = W.V_i^t + c_1 U_1^t (P_{b_1}^t - P_i^t) + c_2 U_2^t (g_b^t - P_i^t)$
6. Move particles to their new positions: $P_i^{t+1} = P_i^t + v_i^{t+1}$
7. Go to step 2 until the stopping criteria are satisfied.

Particle Swarm Optimization (PSO)

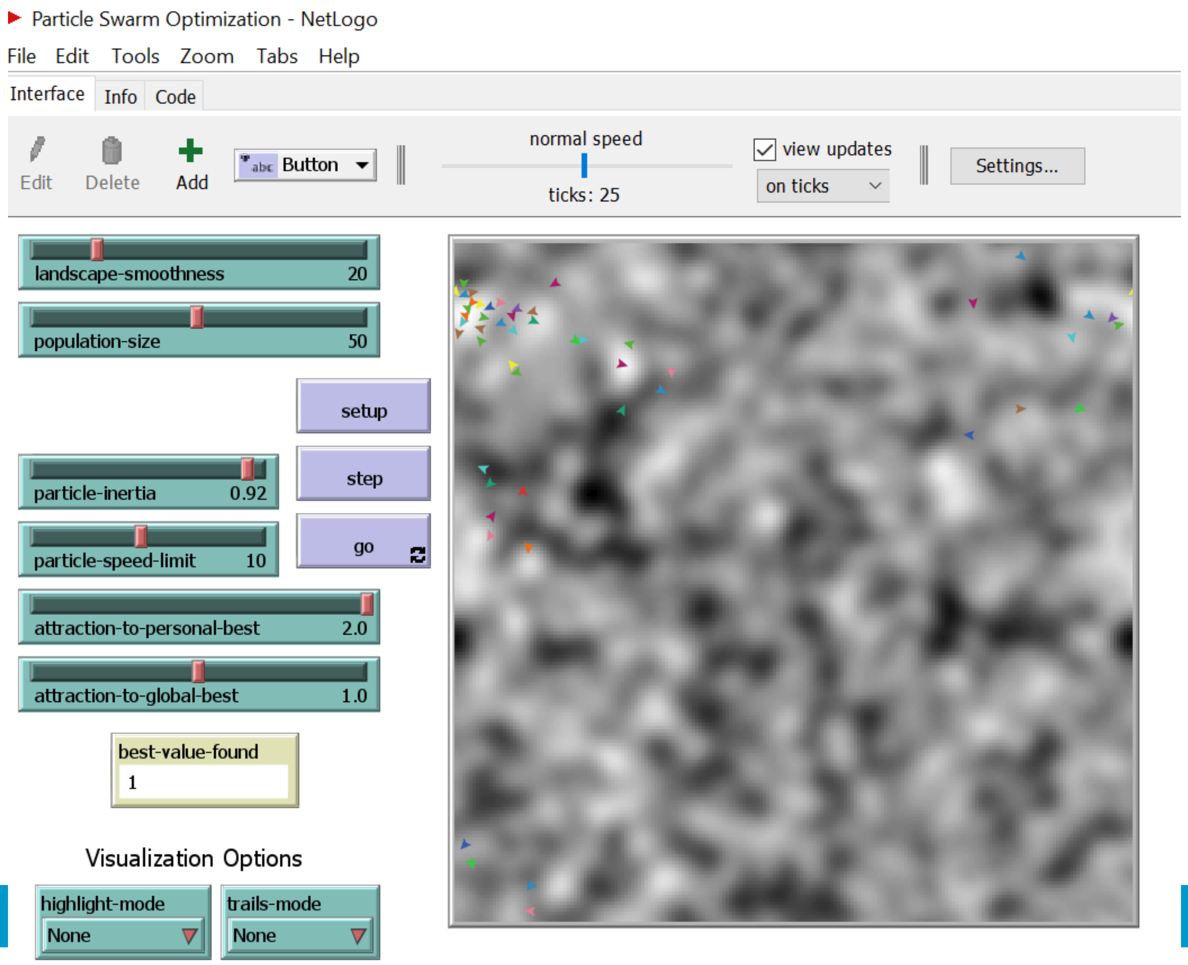


Example: GIS-based PSO for land use allocation problem



Source: Masoomi, Z., Mesgari, M. S., & Hamrah, M. (2013). Allocation of urban land uses by multiobjective particle swarm optimization algorithm. International Journal of Geographical Information Science, 27(3), 542–566.

Particle Swarm Optimization (PSO) NetLogo demo



Particle Swarm Optimization - NetLogo

File Edit Tools Zoom Tabs Help

Interface Info Code

landscape-smoothness 20

population-size 50

particle-inertia 0.92

particle-speed-limit 10

attraction-to-personal-best 2.0

attraction-to-global-best 1.0

best-value-found 1

highlight-mode None

trails-mode None