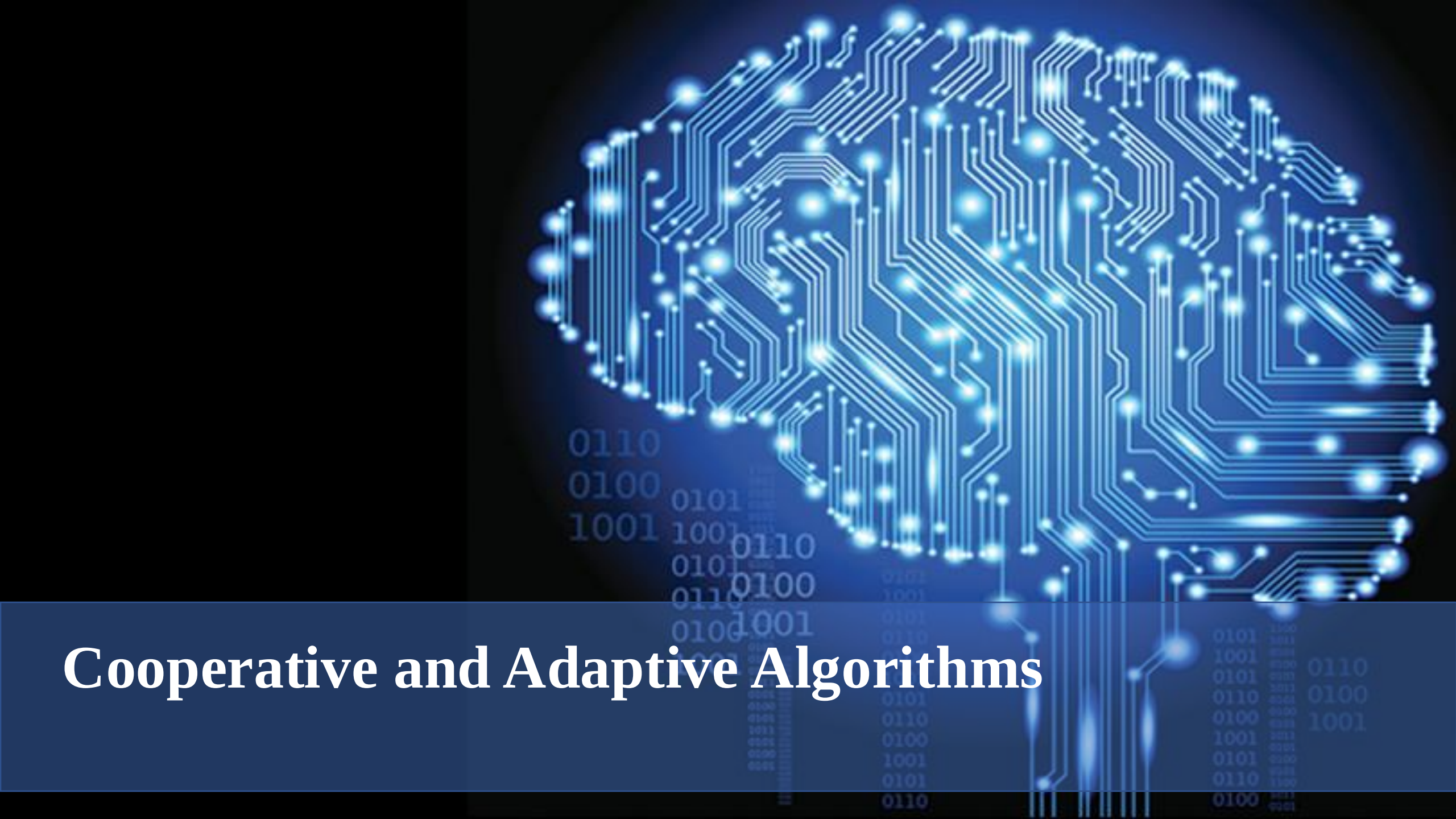


Cooperative and Adaptive Algorithms

Genetic Algorithms

Adaptive GAs

ADAPTIVE GAS

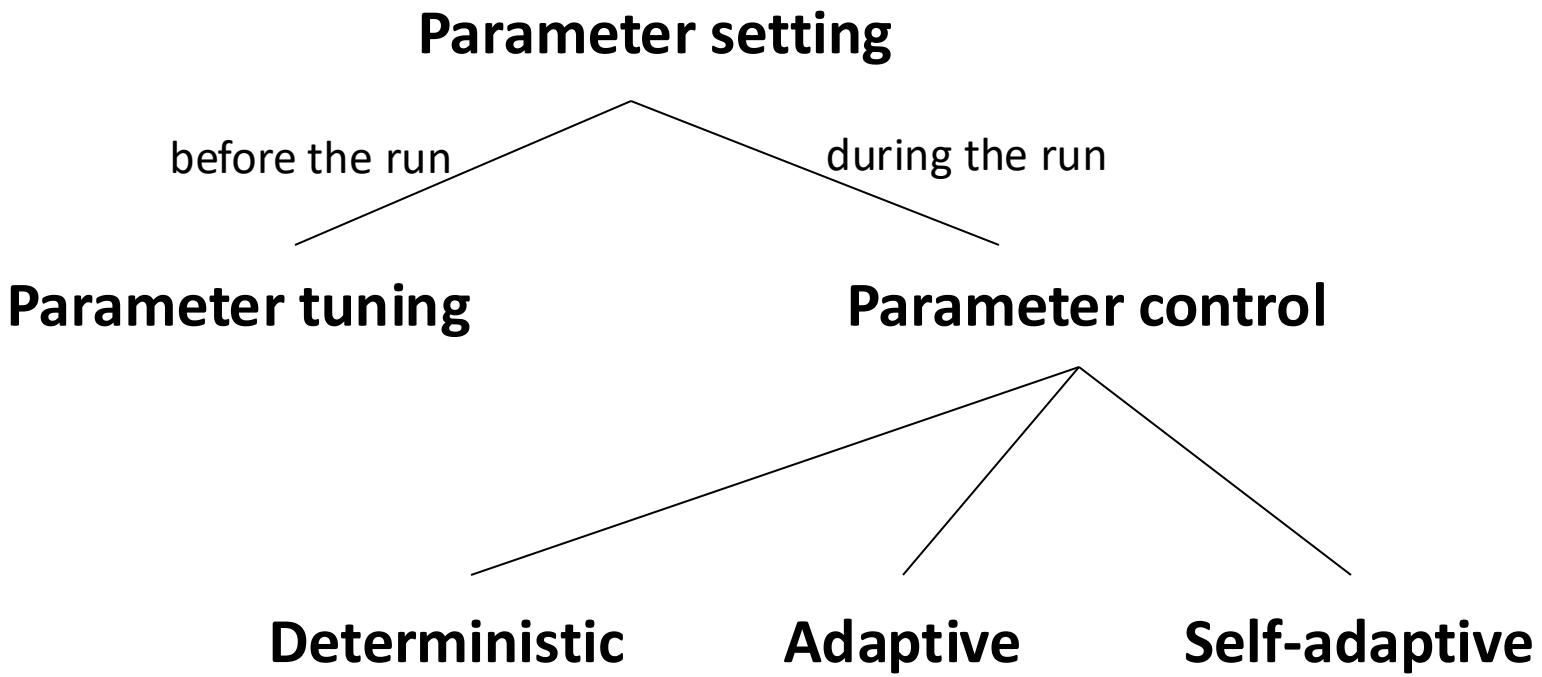
- One of the main disadvantages of GAs is the problem of *parameter tuning*

- ***Parameter tuning*** refers to the problem of finding a suitable values for the different algorithm parameters before the algorithm is run

- Disadvantages of parameter tuning:
  - ***Time consuming*** process,
  - Parameters were tuned one-by-one, which may lead to sub-optimal choices because the parameters are ***not independent***,
  - Simultaneous tuning of parameters may lead to an ***enormous*** amount of experiments,
  - The best parameters settings are ***problem specific***,
  - ***Different values*** may be optimal at different stages of the search process.

ADAPTIVE GAS

- Adaptive techniques have been proposed in GAs to set the values of the different algorithm parameters on-the-fly,
- These algorithms are more efficient and perform better on a wider range of problems.



ADAPTIVE GAS

- ***Deterministic*** parameter control is characterized by replacing any parameter p with a function p(t), where t is the generation number
  - This approach does not take the progress of the search process into account (or the current state of the search).

◎ ***Adaptive*** parameter control takes feedback from the current state of the search and heuristically control the parameter values,

◎ ***Self-adaptive*** approaches incorporate the parameter values into the chromosomes, in this case the parameter control process will be driven through selection, crossover and mutation.

ADAPTIVE GAs: A MUTATION EXAMPLE

- Recall the Gaussian mutation defined as:

$$x_i^{'} = x_i + N(0,\sigma)$$

- We can adaptively set the value of σ during the run.

- The ***deterministic*** approach could set the value of σ as:

$$\sigma(t) = 1 - 0.9\frac{t}{T}$$

where *t* is the generation number ranging from 0 to T (maximum generation number).

ADAPTIVE GAS: A MUTATION EXAMPLE

- In this case, σ is changed from 1 at the beginning of the run to 0.1 at the end
- Helps in enhancing the fine tuning ability of the search towards the end
- The values of σ are known and determined for every generation.

ADAPTIVE GAS: A MUTATION EXAMPLE

- The ***adaptive*** approach incorporates feedback from the search process,
  - A well-known method is the *Rechenberg's* ***'1/5 success rule'***

◎ The rule states that the shown percentage should hold:

$$\frac{\#\text{successful mutations}}{\#\text{mutations}} = \frac{1}{5}$$

◎ A successful mutation is a mutation that ***produces a better individual***

◎ The values of $\sigma(t)$ are selected as non-deterministic based on the ratio

- If the ratio is greater than 1/5, the mutation step size should be increased, and if the ratio is less than 1/5, the step size should be decreased

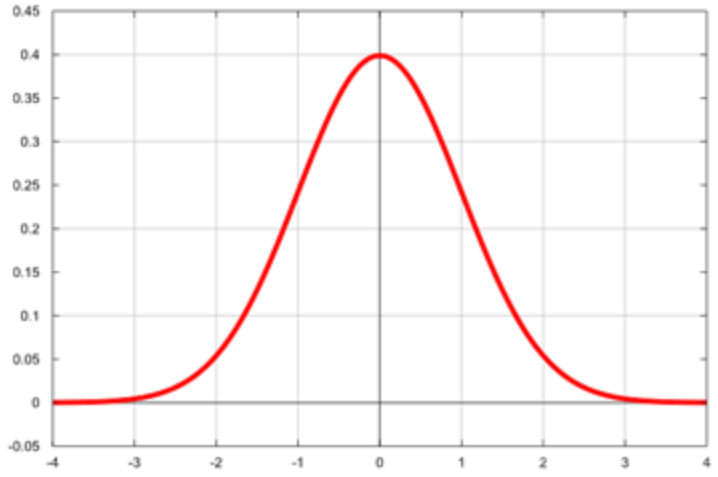
ADAPTIVE GAS: A MUTATION EXAMPLE

- Motivation behind the rule:
  - If the moves are not very successful then we should proceed in smaller steps,
  - If the moves are very successful then we should try larger steps in order to improve the efficiency of the search.

- The update rule is applied every n generations as follows:

$$\sigma = \begin{cases} \sigma / c, & if \ \ p_s > 1/5 \\ \sigma . c, & if \ \ p_s < 1/5 \\ \sigma, & if \ \ p_s = 1/5 \end{cases}$$

where p_s is the calculated percentage and 0.82 < c < 1.



ADAPTIVE GAS: A MUTATION EXAMPLE

- The ***self-adaptive*** approach adds the mutation step size to the individual:

$$x = \langle x_1, x_2, ..., x_n, \sigma \rangle$$

- The value of σ is evolved with the individual

- Each individual will have a different mutation step size.
- Mutating the individual is usually on the form:

$$\sigma' = \sigma . e^{N(0,\tau_0)}$$

$$x_i' = x_i + N(0, \sigma^i)$$

where τ_0 is a parameter of the method referred to as the *learning rate*.

ADAPTIVE GAS: A CROSSOVER EXAMPLE

- One can also adapt the crossover operator
- This operator could be adapted on three levels:
  - **Top level:** adapting the operator itself
  - **Medium level:** adapting the probability of crossover
  - **Bottom level:** adapting the crossing position or the swapping probability.
- One approach is to adapt the swapping probability
- Instead of uniform crossover, where the probability at each gene is always 0.5, this probability is adapted and different across genes.

ADAPTIVE GAS: A CROSSOVER EXAMPLE

- A real–valued $f_1(t)$ vector holding the frequencies of the 1-bits at each gene is calculated after every generation t

- Then, a vector holding the probabilities of swapping the different genes is calculated $p_s(t)$

- The swapping probability vector (referred to as the triangular rule):

$$p_s(t) = P_{\max} - 2\left|f(t) - 0.5\right|\left(P_{\max} - P_{\min}\right)$$

- where |.| is the absolute function and P_{max} and P_{min} are the minimum and maximum allowable probabilities.

ADAPTIVE GAS: A CROSSOVER EXAMPLE

- Motivation behind the rule:
  - As the GA progresses, the genes that are 1-intrinsic (or 0-intrinsic) will converge towards these values
  - Hence, their frequencies will approach 1 (or 0),
  - Consequently, their swapping probabilities will be very low.

Genetic Algorithms

Parallel GAs

PARALLEL GAS

- Different approaches have been proposed to study parallelization in GAs:

  - **Master-slave approaches,**
  - **Fine-grained GAs,**
  - **Coarse-grained GAs,**
  - Hierarchal parallel GAs.

PARALLEL GAS

- In the ***master-slave*** approach:
  - ***Selection and mating*** are handled by the master processor and applied on the entire population
  - The fitness evaluation process is distributed among different slave processors
  - Also known as ***global parallel GAs***.

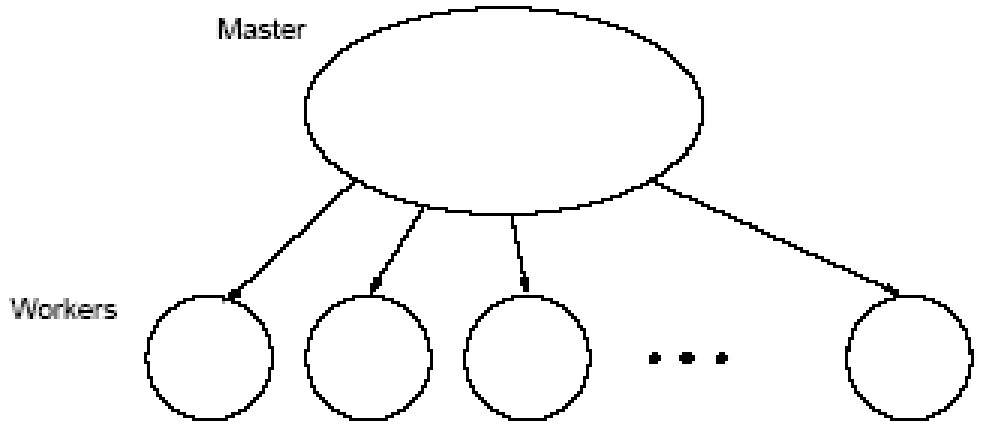


Figure 1. *A schematic of a master-slave parallel GA. The master stores the population, executes GA operations, and distributes individuals to the slaves. The slaves only evaluate the fitness of the individuals.*

PARALLEL GAS

- In the ***fine-grained*** approach:
  - Selection and mating is ***restricted to local neighbourhoods*** (may overlap),
  - Suitable for massively parallel computers,
  - The ideal case is to have one individual per processor.

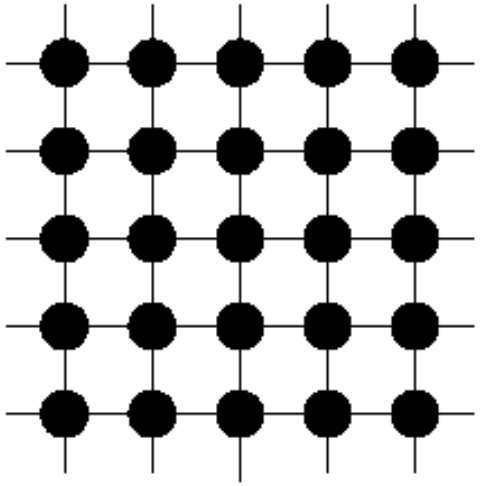
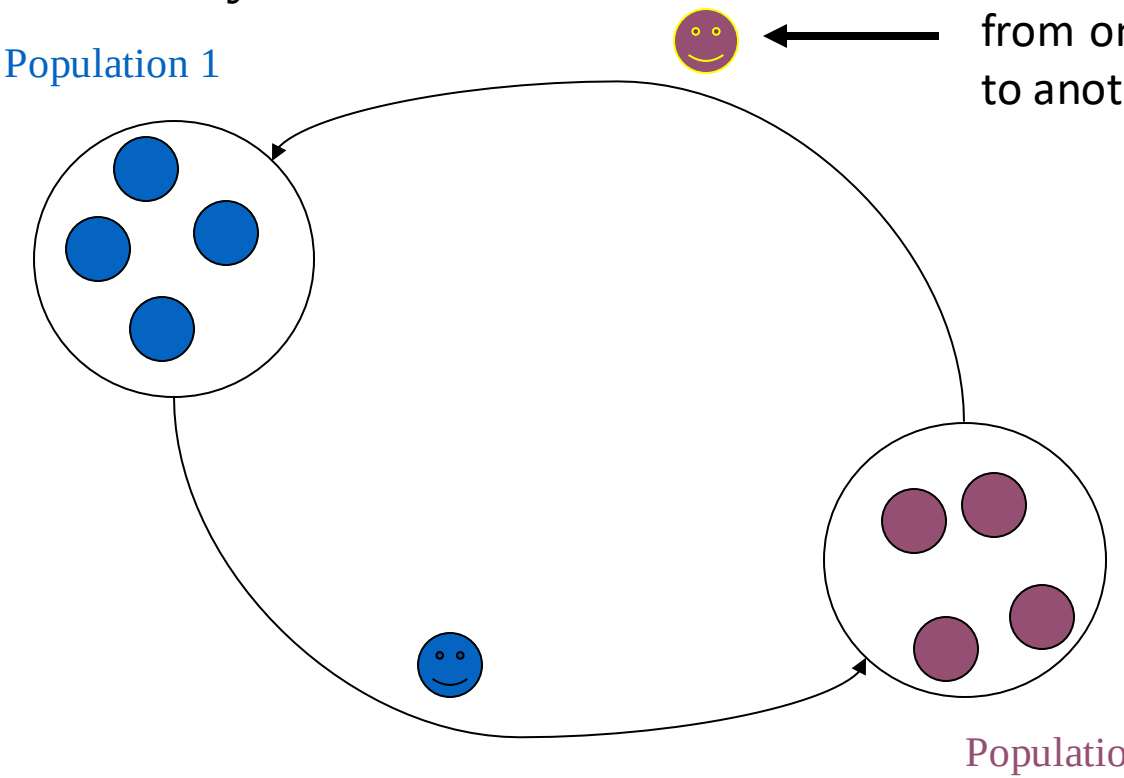


Figure 2. *A schematic of a fine-grained parallel GA. This class of parallel GAs has one spatially-distributed population, and it can be implemented very efficiently on massively parallel computers.*

PARALLEL GAS

- In the ***coarse-grained*** approach:
  - Multiple populations evolving in parallel,
  - Selection and mating is restricted within a population,
  - The populations exchange individuals every now and then,
  - Also referred to as:
    - ***multiple-population GAs***,
    - ***multiple-deme GAs***,
    - ***distributed GAs***
    - ***"island" parallel GAs***.
  - deme: local population



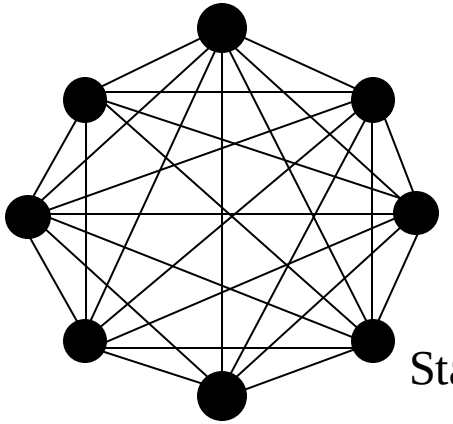
MULTIPLE-DEME - FACTORS

- There are a lot of different issues to consider in multiple-deme GAs:
  - Number and sizes of populations,
  - Different topologies,
  - Different migration polices,
  - Different migration frequencies,
  - Different migration rates.

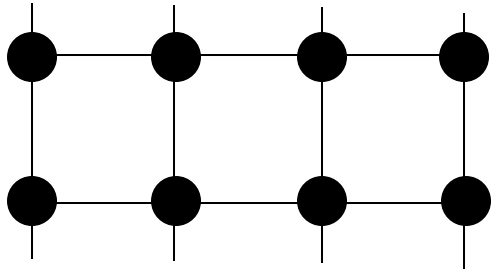
MULTIPLE-DEME - TOPOLOGY

- The *topology* factor is important when evolving more than two populations,
- It states which populations are connected to each other,
- Individuals migrate between connected populations only,
- The communication topology also affects the cost of migration.

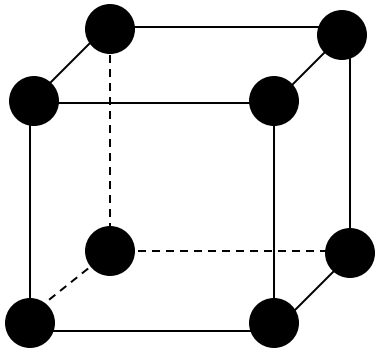
MULTIPLE-DEME - TOPOLOGY



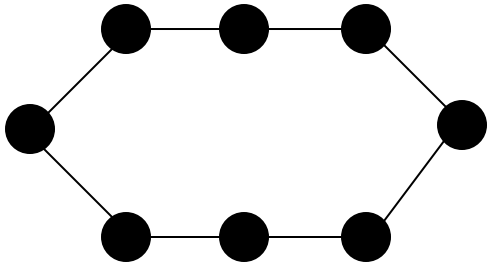
Star topology



Grid topology



Hyper-cube topology



Ring topology

MULTIPLE-DEME - TOPOLOGY

- In dense topologies, good solutions spread faster and quickly take over the populations,

- In sparse topologies:
  - Solutions spread slower causing the demes to be more isolated,
  - Allow different solutions to appear,
  - These solutions may come together towards the end and recombine to produce even better individuals.

- *Static* vs. *Dynamic* topologies.

MULTIPLE-DEME – MIGRATION POLICY

- The ***migration policy*** dictates how the migrating individuals are being selected and how are they being handled at the receiving population,

- Four different exchange approaches could be adopted:
  - Best-Worst,
  - Best-Random,
  - Random-Random,
  - Random-Worst.

MULTIPLE-DEME – MIGRATION FREQUENCY

- The migration frequency controls when the communication is being carried out,
- Two types of communication occur:
  - Synchronous communication, migration happens every predetermined number of generations,
  - Asynchronous communication, migration happens when a certain event occur (e.g. populations have converged).

MULTIPLE-DEME – MIGRATION FREQUENCY

- Assuming synchronous communication, an interesting question is “When is the right time to migrate?”,

- Communication in the early stages may lead the populations to sub-optimal solutions as well as causing high communication costs.

MULTIPLE-DEME – MIGRATION RATE

- The *migration rate* is the ***number of individuals*** that are being migrated from one population to another at every communication step,
- Low migration rates may cause the demes to behave independently, the migrants don’t have a significant effect,
- High migration rates may cause a fast convergence to sub-optimal solutions.

PARALLEL GAS

- Another form of parallelization is what is known as ***Cooperative GAs***,
- Having multiple populations where the fitness of any individual in any population depends on the individuals of the other populations,
- Proposed by Potter el. al. in 1994.

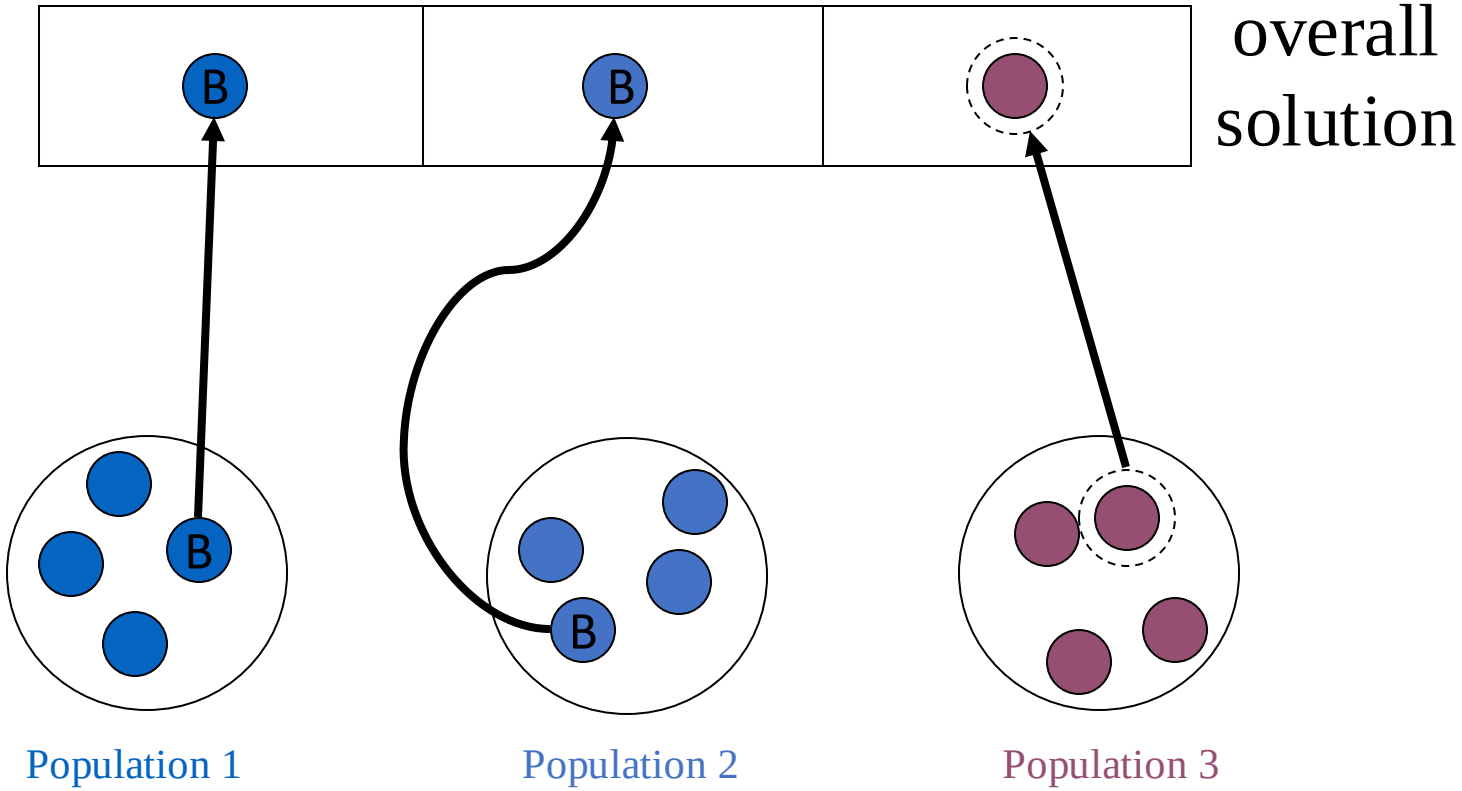
PARALLEL GAS

- The general idea is to have different populations optimizing different variables of the problem (different dimensions of the solution),

- The fitness of any individual is determined by its value and the value of the best individuals in all the other populations,

- Performs best If the problem variables are independent.

PARALLEL GAS



EVOLUTION STRATEGIES

EVOLUTION STRATEGIES

Pioneered by Schwefel and Rechenberg in the late 1960s.

Evolution Strategies:

Represtation:	Real-valued vectors
Recombination:	none / discrete or intermediate recombination
Mutation:	Gaussian perturbation
Parent selection:	Uniform random
Survivor selection:	Generational (μ, λ) or steady-state $(\mu + \lambda)$

REPRESENTATION

Representation

Genotype: Real-valued vector $x = (x_1, ..., x_n) \in R^n$ of n variables:

x_1	x_2	x_3	x_4	$\cdots$	x_n

e.g.

1.3	−0.5	0.32	11.23	6

Typically, evolutionary strategies are used for continuous parameter optimization, meaning that the problem at hand can be given as an objective function $R^n \mapsto R$. In this case, the genotype space and phenotype space are identical.

Self-Adaptation: In ES, it's common practice to include parameters controlling the mutation directly into each individual.

$$x = (\underbrace{x_1, x_2, \ldots, x_n}_{\text{candidate}}, \underbrace{\sigma_1, \sigma_2, \ldots, \sigma_k}_{\text{step size}}, \underbrace{\alpha_1, \alpha_2, \ldots, \alpha_l}_{\text{'interaction'}})$$

Common mutation parameters include

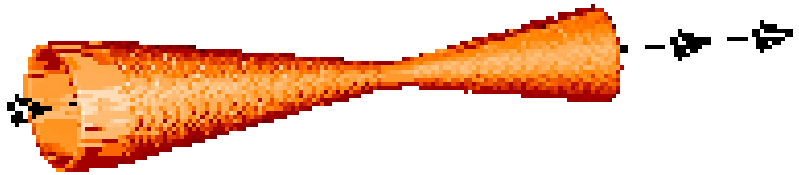
- either $k = 1$ (one value σ for all variables x_i) or $k = n$ (individual value σ_i for each variable x_i) parameters controlling the mutation step sizes
- (optionally) l parameters controlling the 'interaction' between the step sizes of different variables.

INTRODUCTORY EXAMPLE

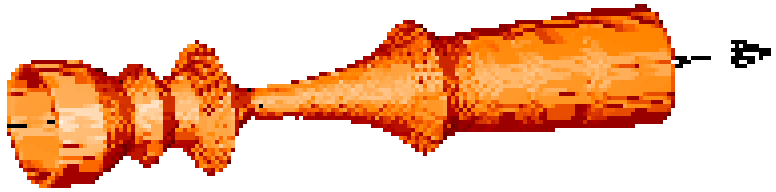
- Task: **minimimise** $f : R^n \rightarrow R$
- Algorithm: “two-membered ES” using
  - Vectors from R^n directly as chromosomes
  - Population size 1
  - Only mutation creating one child
  - Greedy selection

ANOTHER HISTORICAL EXAMPLE: THE JET NOZZLE EXPERIMENT

Task: to optimize the shape of a jet nozzle
Approach: random mutations to shape + selection

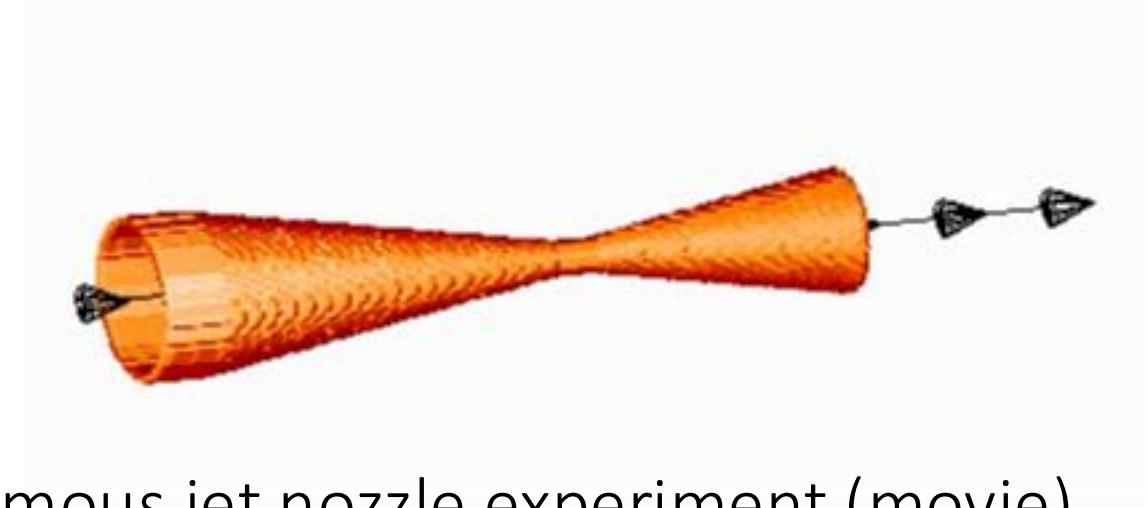


Initial shape



Final shape

ANOTHER HISTORICAL EXAMPLE: THE JET NOZZLE EXPERIMENT CONT'D



The famous jet nozzle experiment (movie)

Jet nozzle: the movie

http://ls11-www.cs.uni-dortmund.de/people/schwefel/EADemos/

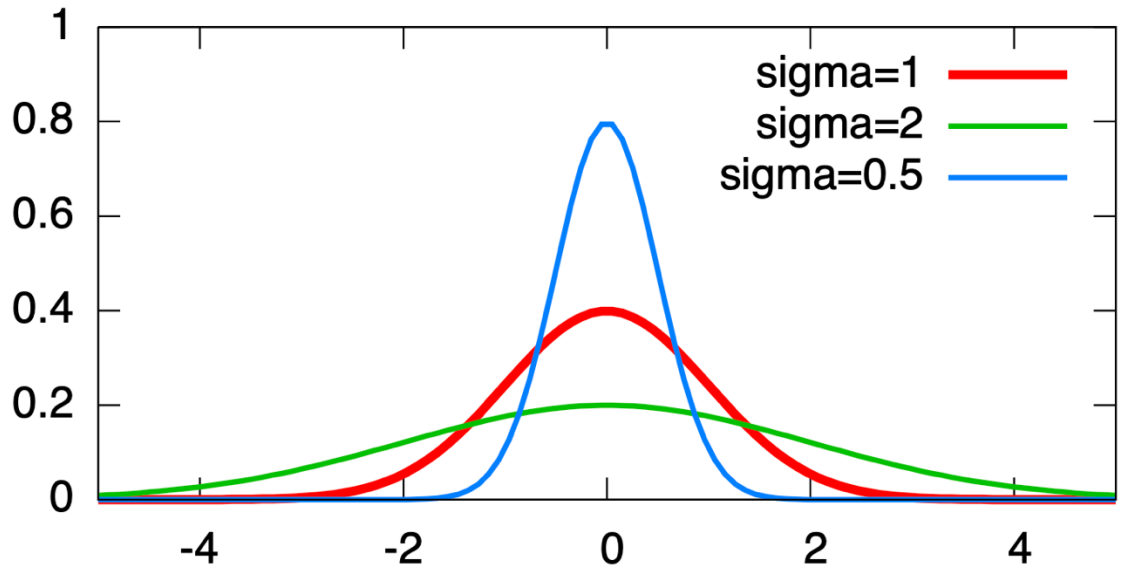
MUTATION

Mutation is realized by adding a random number Δx_i to each of the values x_i:

$$x_i' = x_i + \Delta x_i.$$

The delta-terms are usually drawn from a zero-centered normal distribution $\mathcal{N}(0, \sigma)$ with mean $\mu = 0$ and standard deviation σ. In this case, the probability density function is

$$p(\Delta x_i) = \frac{1}{\sigma\sqrt{2\pi}} \cdot e^{-\frac{(\Delta x_i - \mu)^2}{2\sigma^2}}$$



INTRODUCTORY EXAMPLE: PSEUDOCDE

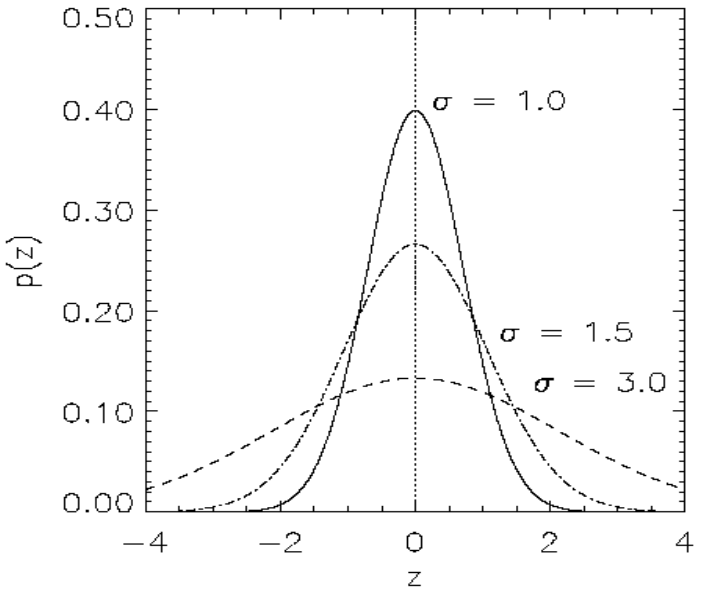
- Set $t = 0$
- Create initial point $x^t = \langle x_1^t, \ldots, x_n^t \rangle$
- REPEAT UNTIL (*TERMIN.COND* satisfied) DO
- Draw z_i from a normal distr. for all $i = 1, \ldots, n$
- $y_i^t = x_i^t + z_i$
- IF $f(x^t) < f(y^t)$ THEN $x^{t+1} = x^t$
  - ELSE $x^{t+1} = y^t$
  - FI
  - Set $t = t+1$
- OD

MUTATION

Regarding the mutation operator, there are two choices to make:

A) whether to use a constant step size σ during the whole evolutionary process or to use a (self-adapted) variable step size

B) whether to use the same step size σ for all variables x_i or to use an individual σ_i for each variable x_i.



Adapting a Step Size with the $\frac{1}{5}$-rule:

- Rechenberg: ideally, 1/5 of the mutations should be 'successful' (fitter than parents)
- Adapt step size σ every k generations according to measured proportion of successful mutations ($0.8 \leq c \leq 1$):

$$\sigma = \begin{cases} \sigma/c & \text{if } p_s > 1/5 \\ \sigma \cdot c & \text{if } p_s < 1/5 \\ \sigma & \text{if } p_s = 1/5. \end{cases}$$

MUTATION: CASE 1, EQUAL STEP SIZE

Adapting a Step Size using Self-Adaptation:

- In the most simple form include one step-size in the individual:

$$(x_1, \ldots, x_n, \sigma)$$

- σ undergoes same variation as the other values x_i
- Important: First mutate σ, then use the new σ' to produce offsprings

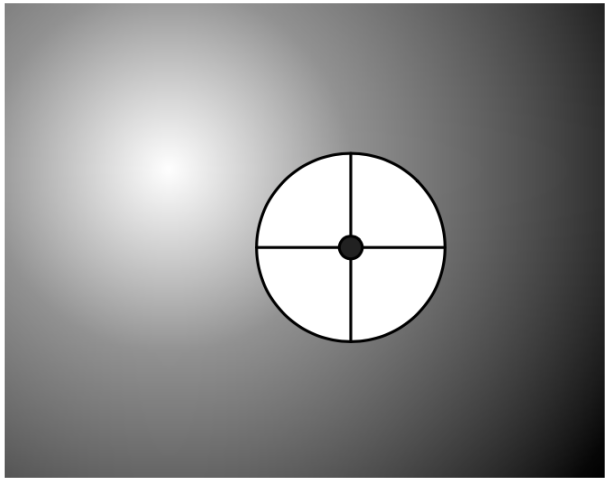
Uncorrelated Mutation with One Step Size:

- Individual $(x_1, \ldots, x_n, \sigma)$ is mutated according to

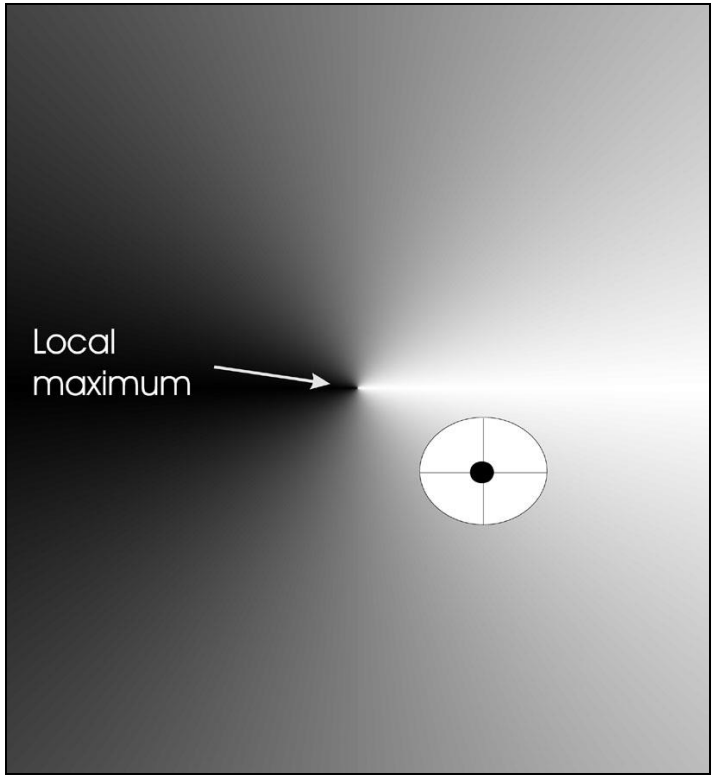
$$\sigma' = \sigma \cdot e^{\mathcal{N}(0,\tau)}$$

$$x_i' = x_i + \mathcal{N}_i(0, \sigma')$$

- Two parameters: 'learning rate' $\tau \propto 1/\sqrt{n}$ and 'boundary rule': $\sigma' < \epsilon_0 \Rightarrow \sigma' = \epsilon_0$



MUTANTS WITH EQUAL LIKELIHOOD



Circle: mutants having the same chance to be created

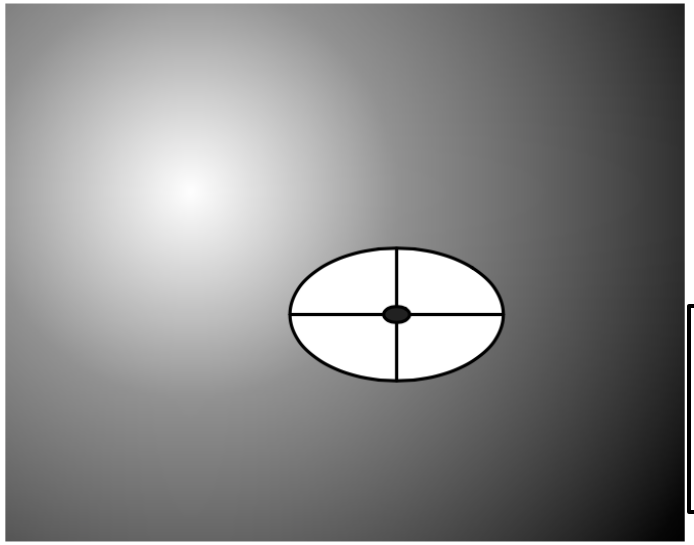
CASE 2: UNCORRELATED MUTATIONS WITH N STEP SIZES

- Observation: fitness landscape can have different slopes in different directions
- Idea: individual $(x_1, \ldots, x_n, \ \sigma_1, \ldots \sigma_n)$ with one individual step size σ_i for each dimension x_i
- Mutation mechanism:

$$\sigma_i' = \sigma_i \cdot e^{\mathcal{N}(0,\tau') + \mathcal{N}_i(0,\tau)}$$

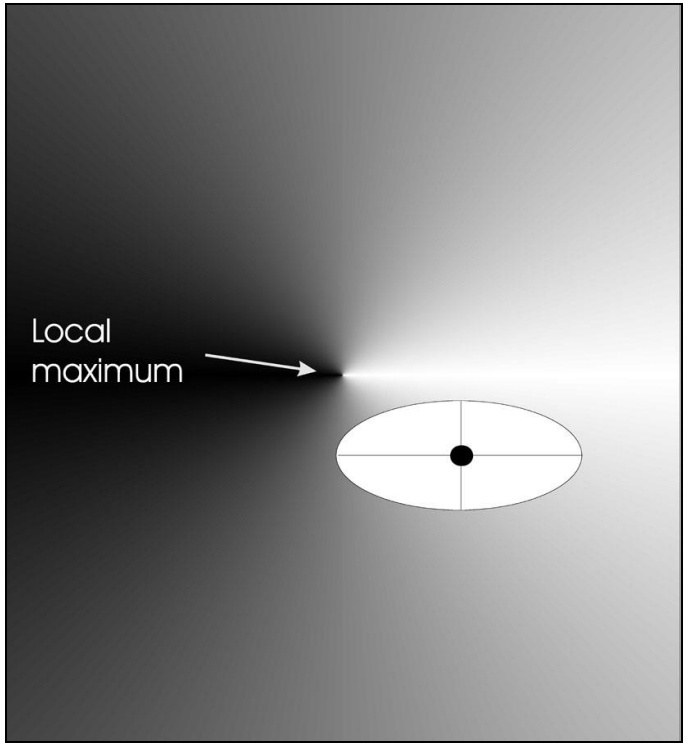
$$x_i' = x_i + \mathcal{N}_i(0, \sigma_i')$$

- Three parameters:
  overall change of mutability:
  $\tau' \propto 1/\sqrt{2n}$
  coordinate-wise change: $\tau \propto 1/\sqrt{2\sqrt{n}}$
  and 'boundary rule': $\sigma' < \epsilon_0 \Rightarrow \sigma' = \epsilon_0$



The two learning rate parameters:
τ' overall learning rate
τ coordinate wise learning rate

MUTANTS WITH EQUAL LIKELIHOOD



Ellipse: mutants having the same chance to be created

MUTATION CASE 3: CORRELATED MUTATIONS

Correlated Mutation:

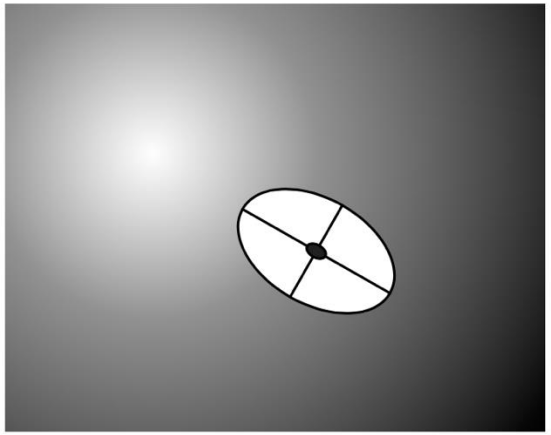
- Observation: ellipses in previous scheme were orthogonal to x-axis
- Idea: individual $(x_1, \ldots, x_n,\ \sigma_1, \ldots \sigma_n,\ \alpha_1, \ldots \alpha_l)$ $(l = n(n-1)/2)$ with correlated step-sizes using a 'covariance' matrix C with

$$
\begin{aligned}
c_{ii} &= \sigma_i^2 \\
c_{ij, i \neq j} &= 0 \text{ iff i and j are not correlated} \\
c_{ij, i \neq j} &= \frac{1}{2}(\sigma_i^2 - \sigma_j^2)\tan(2\alpha_{ij}) \text{ iff i and j are correlated}
\end{aligned}
$$

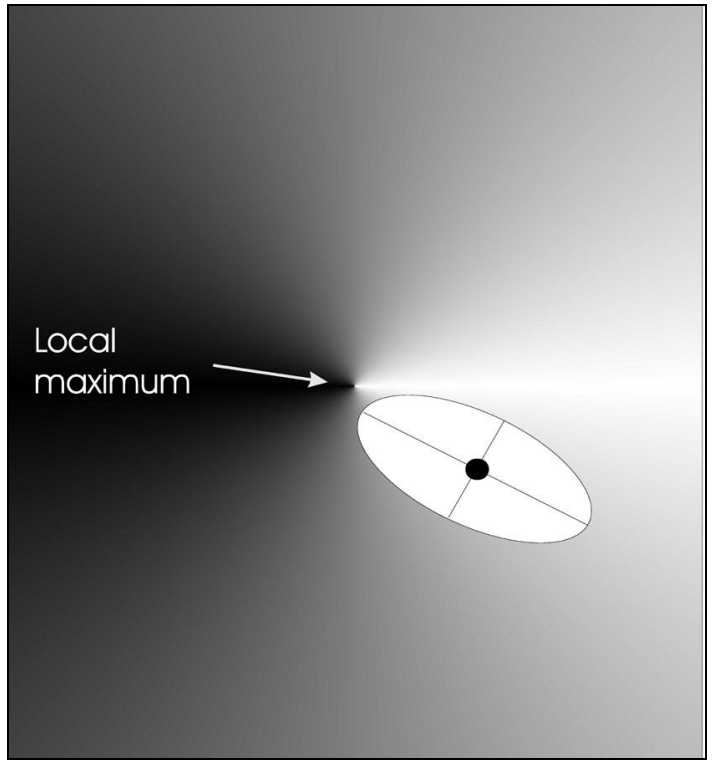
- Mutation mechanism:

$$
\begin{aligned}
\sigma_i' &= \sigma_i \cdot e^{\mathcal{N}(0,\tau') + \mathcal{N}_i(0,\tau)} \\
\alpha_j' &= \alpha_j + \mathcal{N}_j(0, \beta) \\
x' &= x + \mathcal{N}(0, C')
\end{aligned}
$$

where $\Delta x = \mathcal{N}(0, C')$ is a vector drawn from a n-dimensional normal distribution with covariance matrix C'.



MUTANTS WITH EQUAL LIKELIHOOD



Ellipse: mutants having the same chance to be created

RECOMBINATION

If recombination is used, the most common theme in ES is to combine two randomly selected parents that form one offspring. For the parent vectors x and y, one child z is created:

$$z_i = \begin{cases} (x_i + y_i)/2 & \text{intermediary recombination} \\ x_i \text{ or } y_i \text{ chosen randomly} & \text{discrete recombination} \end{cases}$$

Usually, discrete recombination is used for the variables representing the candidate solution and intermediary recombination is used for the search-strategy part.

The procedure is repeated λ times in order to produce λ offsprings.

Variant: selecting different parents x and y for each variable z_i individually ('global recombination').

SELECTION

Parent Selection

In ES not biased by fitness values, parents are drawn randomly with uniform distribution from the population of μ individuals.

Survivor Selection

After creating λ offsprings, μ survivors are selected:

- (μ, λ) selection: μ survivors from the λ offsprings only
- $(\mu + \lambda)$ selection: μ survivors from the union of parents and offsprings.

In both variants the selection is deterministic and rank based.

Practical Considerations:

(μ, λ) selection is often preferred for the following reasons:

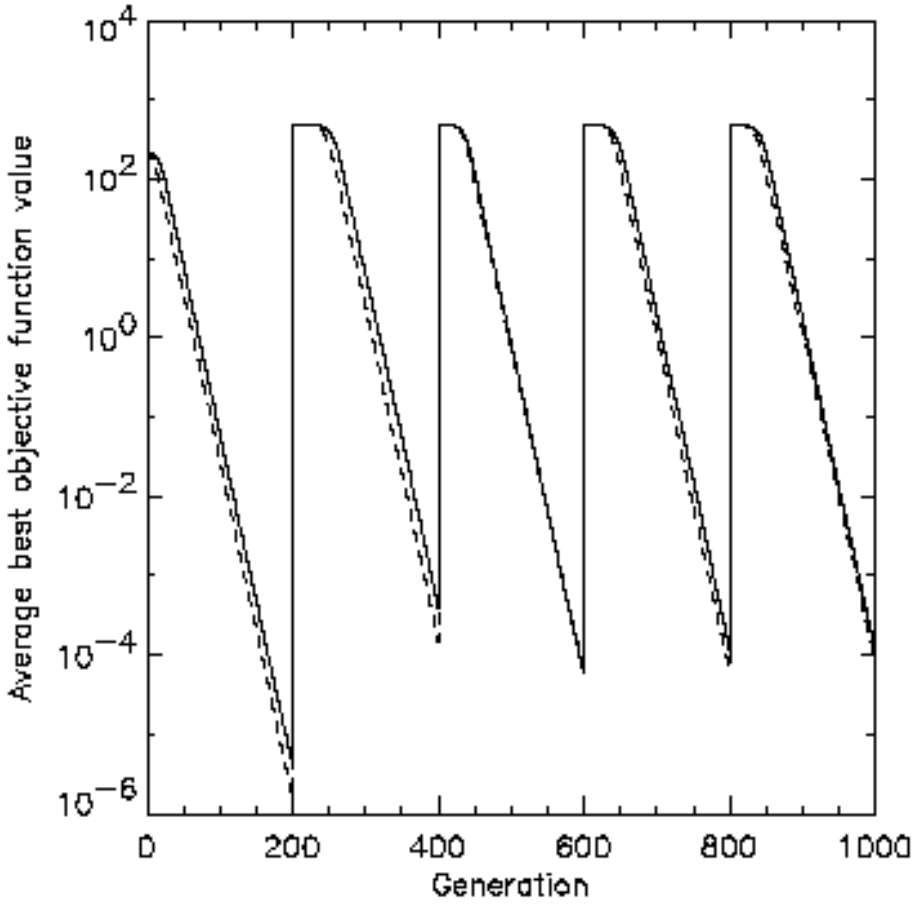
1. replaces all parents, thus able to leave local optima,
2. its better suited to follow non-stationary (moving) optima,
3. in $(\mu + \lambda)$ better candidate solutions dominate better search strategies.

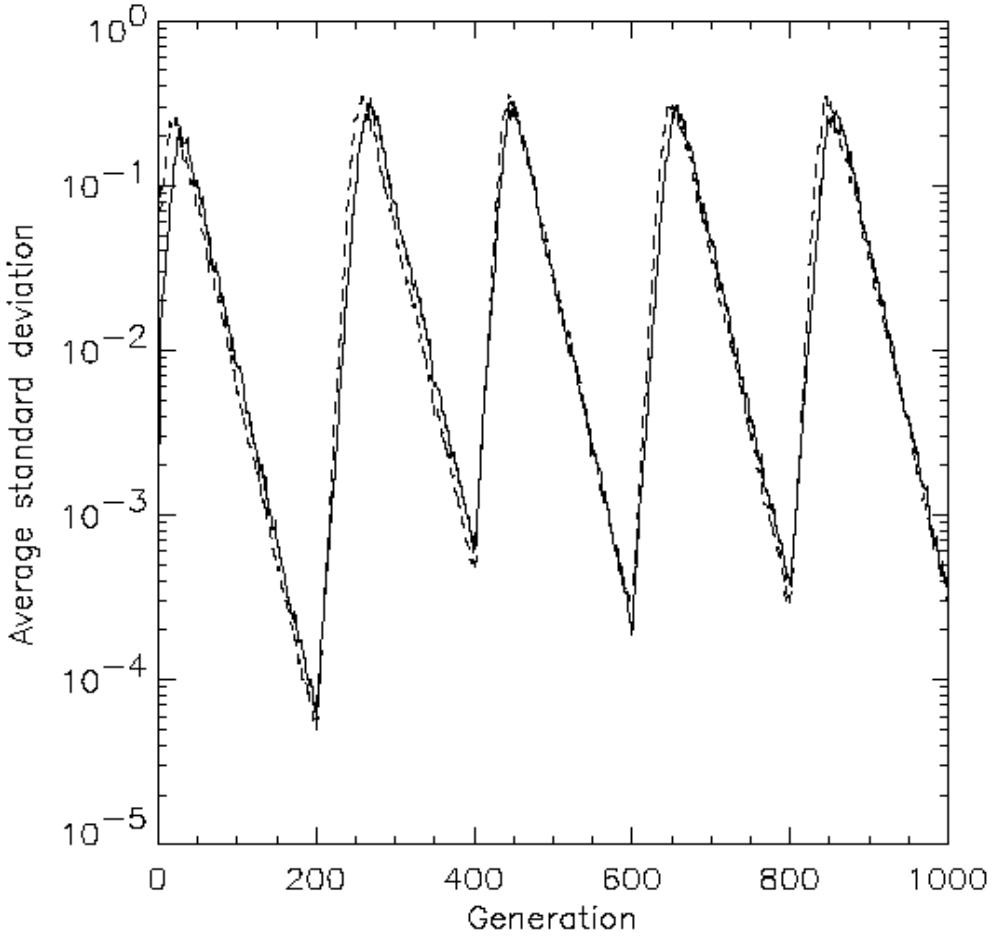
Usually, selection pressure in ES is very high. Common setting: $\lambda = 7 \cdot \mu$

SELF-ADAPTATION ILLUSTRATED

- Given a dynamically changing fitness landscape (optimum location shifted every 200 generations)
- Self-adaptive ES is able to
  - follow the optimum and
  - adjust the mutation step size after every shift !

SELF-ADAPTATION ILLUSTRATED CONT'D





Changes in the fitness values (left) and the mutation step sizes (right)

EXAMPLE APPLICATION: THE CHERRY BRANDY EXPERIMENT

- Task to create a colour mix yielding a target colour (that of a well known cherry brandy)
- Ingredients: water + red, yellow, blue dye
- Representation: ⟨ w, r, y ,b ⟩ no self-adaptation!
- Values scaled to give a predefined total volume (30 ml)
- Mutation: lo / med / hi σ values used with equal chance
- Selection: (1,8) strategy

EXAMPLE APPLICATION: CHERRY BRANDY EXPERIMENT CONT'D

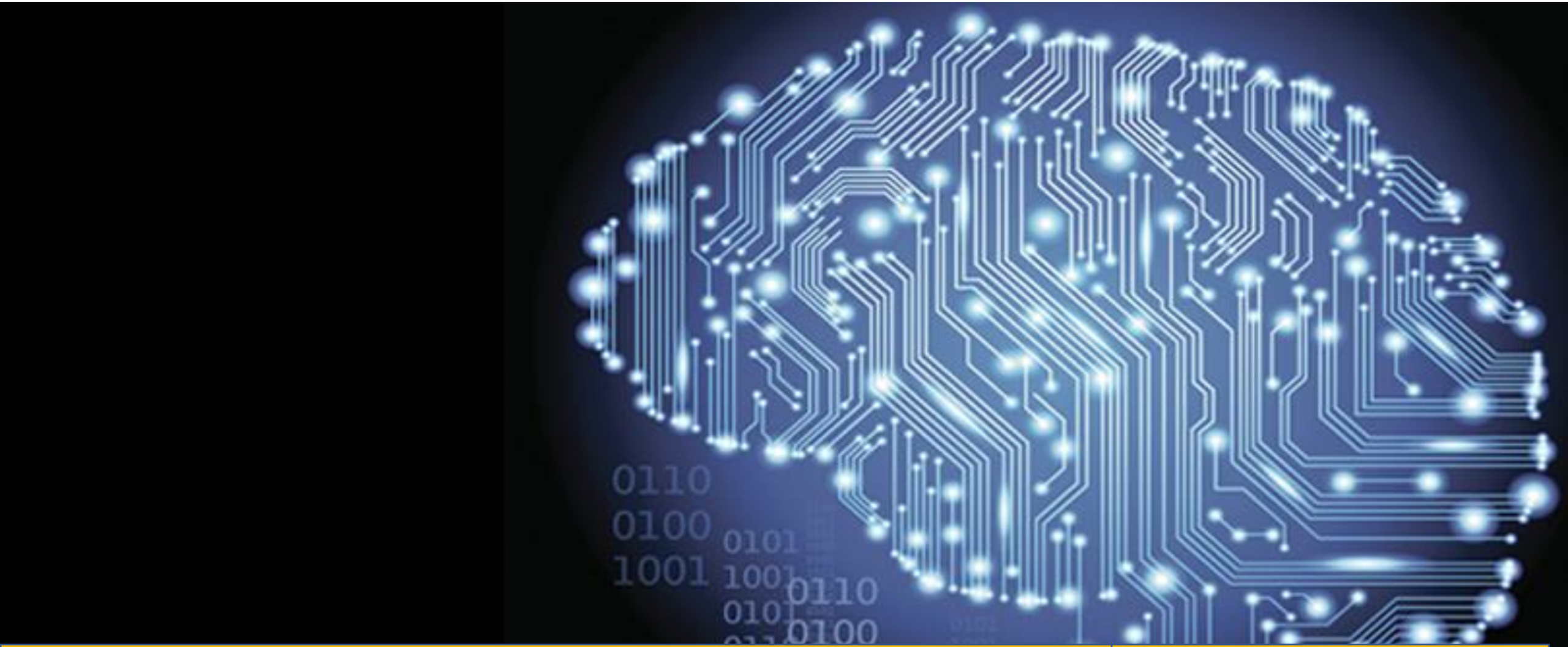
- Fitness: students effectively making the mix and comparing it with target colour
- Termination criterion: student satisfied with mixed colour
- Solution is found mostly within 20 generations
- Accuracy is very good

EXAMPLE APPLICATION: THE ACKLEY FUNCTION (BÄCK ET AL '93)

- The Ackley function (here used with n =30):

$$f(x) = -20 \cdot \exp\left(-0.2\sqrt{\frac{1}{n} \cdot \sum_{i=1}^{n} x_i^2} \right) - \exp\left(\frac{1}{n} \sum_{i=1}^{n} \cos(2\pi x_i) \right) + 20 + e$$

- Evolution strategy:
  - Representation:
    - $-30 < x_i < 30$
    - 30 step sizes
  - (30,200) selection
  - Termination : after 200000 fitness evaluations
  - Results: average best solution is $7.48 \cdot 10^{-8}$ (very good)



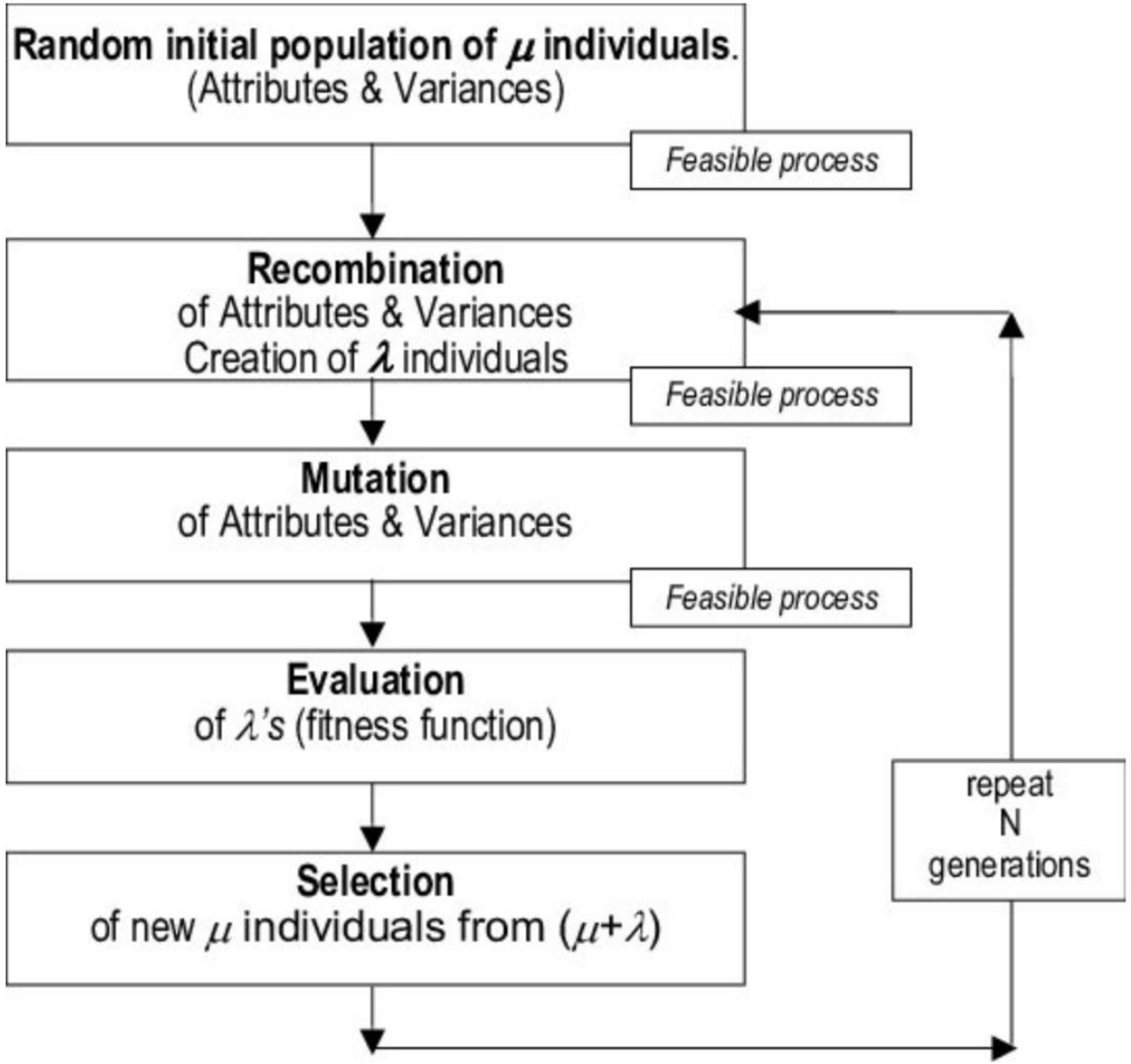
Cooperative and Adaptive Algorithms

Evolution Strategies
CONT

WHAT IS ES

- It is a technique that searches for the **optimum** solution in a **search-space**
- Evolution Strategies belong to the family of Evolutionary Computation
- Evolution strategy steps:
  1. **Generate** a population of candidate solutions
  2. **Evaluate** every individual in the population
  3. **Select** parents from the fittest individuals
  4. **Reproduce** offspring of the next generation (Recombination & mutation)
  5. **Repeat** until a *termination criterion* is met

REVOLUTION STRATEGIES



EVOLUTION STRATEGIES VS GENETIC ALGORITHMS

	ES	GA
Initial Population	Random mutations of the initial guess	Random or seeded
Evaluation	Objective Function	Fitness (Evaluation) Function
Selection	Truncation Selection	Different methods
Reproduction	Recombination + Mutation	Crossover + Mutation
Termination	Almost similar stop conditions	

THE BASIC EVOLUTION STRATEGY

- The basic evolution strategy is defined by:

$$(\mu/\rho, \lambda)\text{-ES and } (\mu/\rho+ \lambda)\text{-ES}$$

Where:

μ	The number of selected individuals per generation	
ρ	The number of parents (selected from μ) involved in recombination ($\leq \mu$)	
λ	The number of individuals per generation (population size)	
,	Comma Selection	→ μ parents are selected from the λ individuals
+	Plus Selection	→ μ parents are selected from the λ individuals + the current ρ parents

THE STRUCTURE OF AN INDIVIDUAL

$$y = 5.x_1^2 + 10.x_2 - 15.z + 7$$

$$Y = \{x_1, x_2, z\}$$

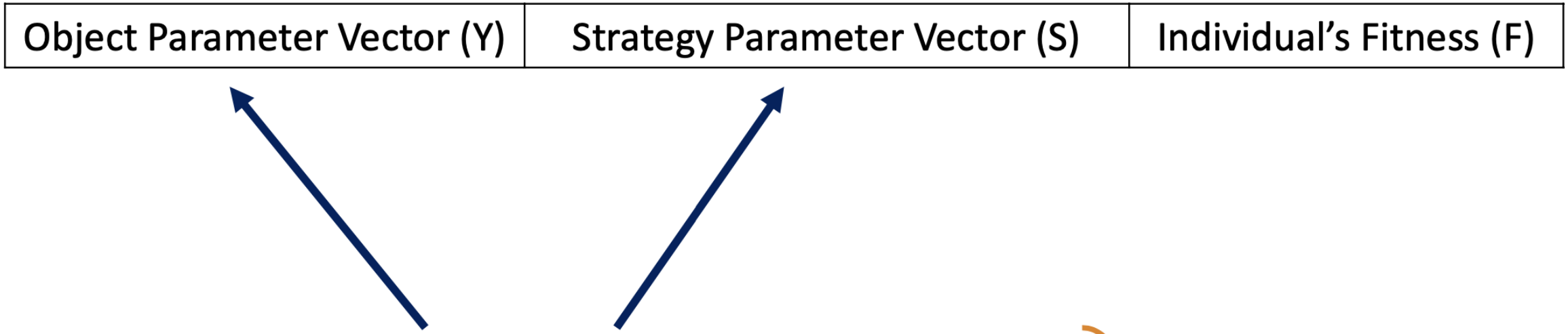
Object Parameter Vector (Y)	Strategy Parameter Vector (S)	Individual's Fitness (F)

Y The candidate solution of the problem (e.g. (x, y) point)

S The parameters used by the strategy (e.g. mutation strength)

F The fitness of the candidate solution y as measured by the fitness function (i.e. the value of the objective function)

THE STRUCTURE OF AN INDIVIDUAL



- Evolution strategies search for the optimum:
  1. Solution: The highest fitness
  2. Strategy Parameters: The fastest improvement

Two search spaces

THE BASIC EVOLUTION STRATEGY

- An initial guess, should be as close as possible to the expected solution

1- Initial Solution

2 - Initial Population

3 - Evaluation

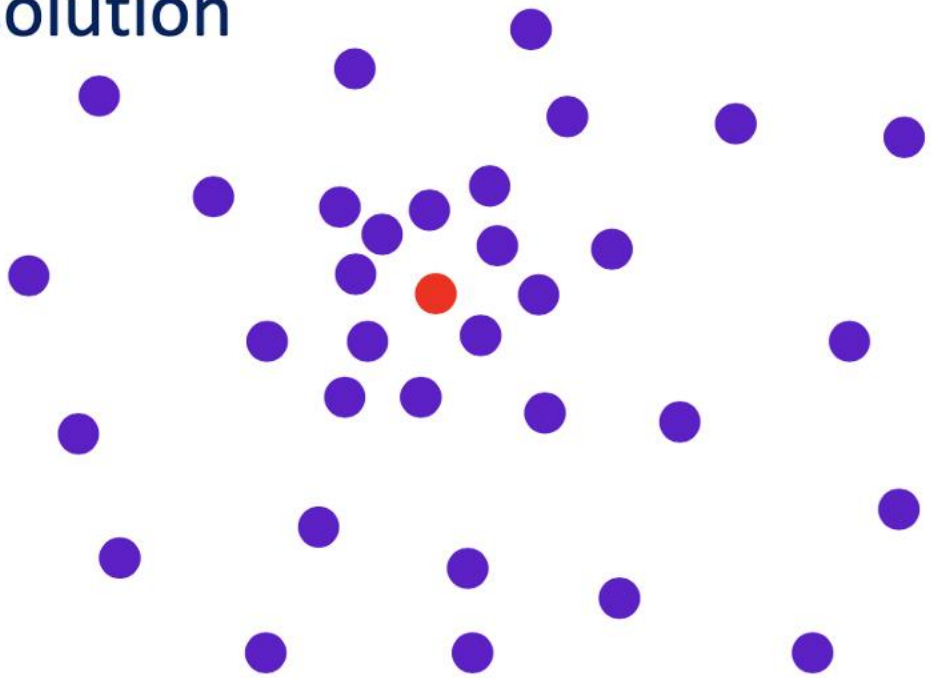
4 - Selection

5 - Reproduction

6 - Termination

THE BASIC EVOLUTION STRATEGY

- The intial population is generated by mutating the initial solution



1- Initial Solution

2 - Initial Population

3 - Evaluation

4 - Selection

5 - Reproduction

6 - Termination

THE BASIC EVOLUTION STRATEGY

- Every individual is evaluated by the objective function

$$fitness = e^{2.x} + y^{-\frac{1}{2}} + 10.\log(z) - 99.823$$

Best Fitness = 0

1- Initial Solution

2 - Initial Population

3 - Evaluation

4 - Selection

5 - Reproduction

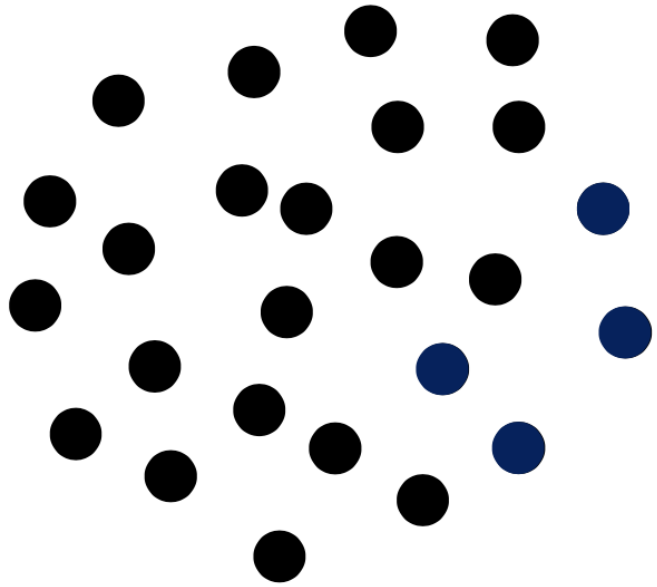
6 - Termination

THE BASIC EVOLUTION STRATEGY

- Truncation Selection is used

  Select the fittest μ individuals

  Drop the other individuals



1- Initial Solution

2 - Initial Population

3 - Evaluation

4 - Selection

5 - Reproduction

6 - Termination

THE BASIC EVOLUTION STRATEGY

Reproduction

Recombination

Combining two or more parents to produce a *mean* for the new generation

Mutation

Adding normally-distributed random vectors to the new *mean*

1- Initial Solution

2 - Initial Population

3 - Evaluation

4 - Selection

5 - Reproduction

6 - Termination

RECOMBINATION

- Creates one child
- Acts per variable / position by either
  - Averaging parental values, or
  - Selecting one of the parental values
- From two or more parents by either:
  - Using two selected parents to make a child
  - Selecting two parents for each position anew

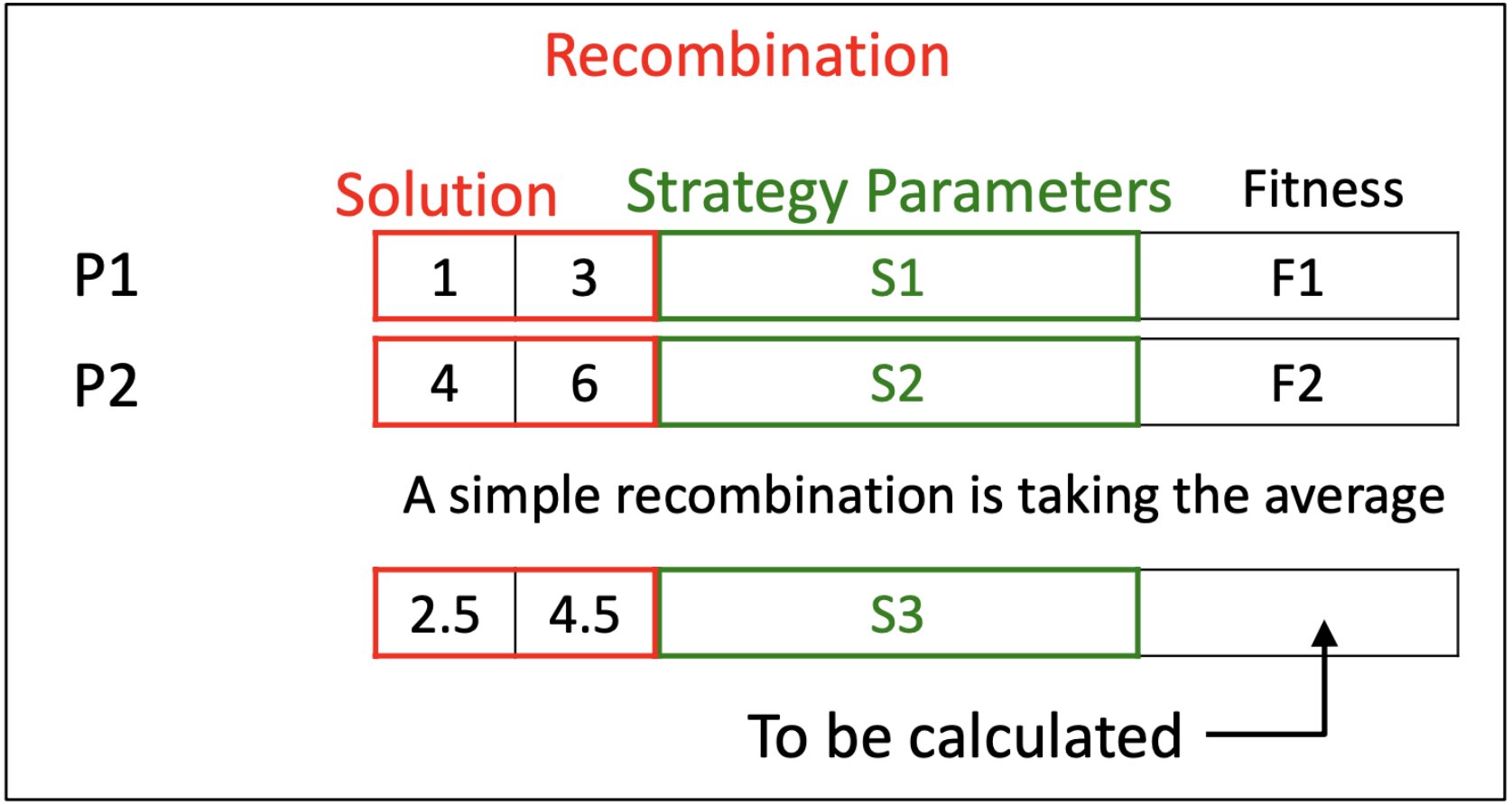
Names of recombinations

	Two fixed parents	Two parents selected for each i
$z_i = (x_i + y_i)/2$	Local intermediary	Global intermediary
z_i is x_i or y_i chosen randomly	Local discrete	Global discrete

PARENT SELECTION

- Parents are selected by uniform random distribution whenever an operator needs one/some
- Thus: ES parent selection is unbiased - every individual has the same probability to be selected
- Note that in ES “parent” means a population (in GA’s parent is a member of the population)

THE BASIC EVOLUTION STRATEGY



1- Initial Solution

2 - Initial Population

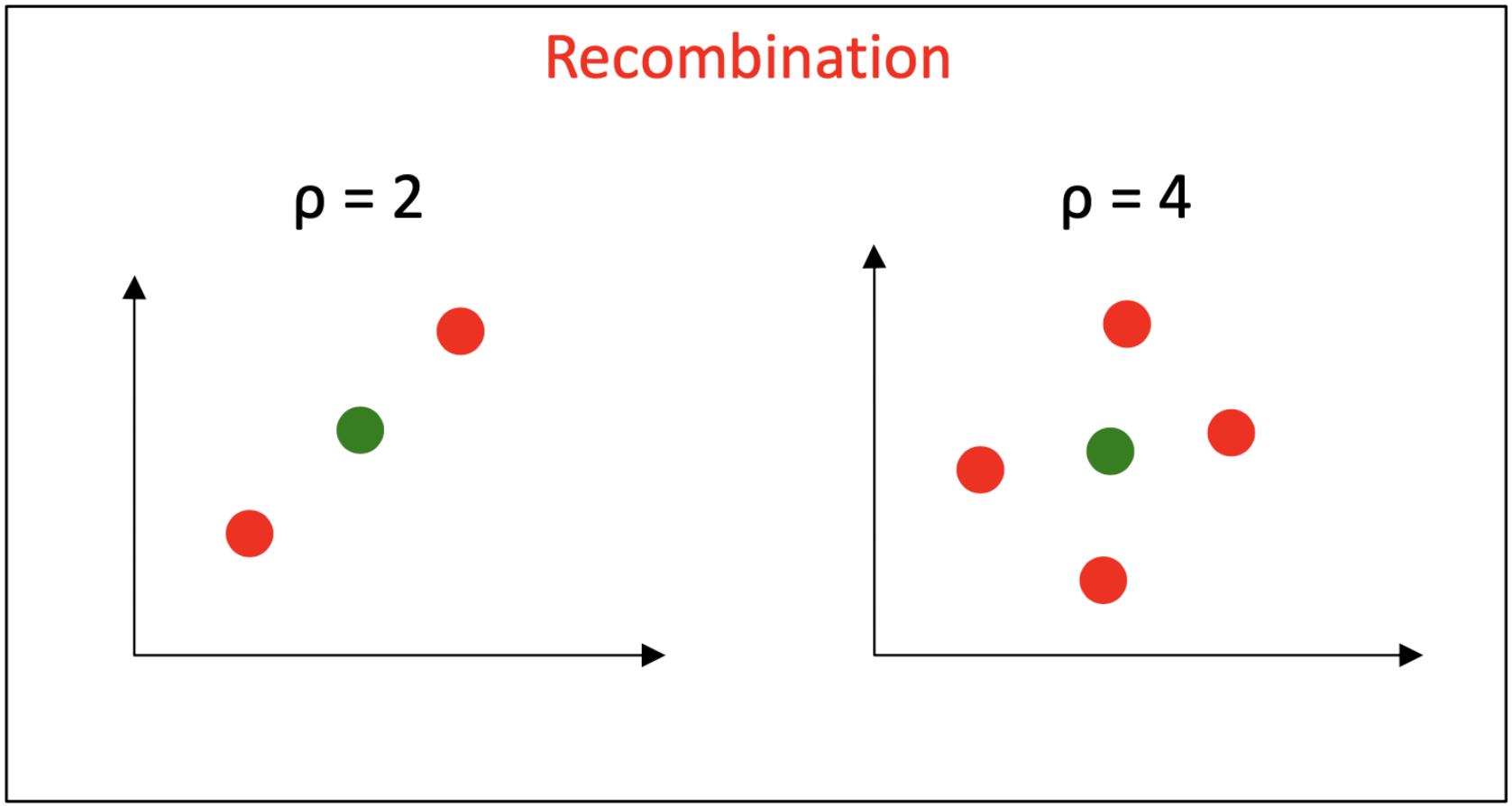
3 - Evaluation

4 - Selection

5 - Reproduction

6 - Termination

THE BASIC EVOLUTION STRATEGY



1- Initial Solution

2 - Initial Population

3 - Evaluation

4 - Selection

5 - Reproduction

6 - Termination

THE BASIC EVOLUTION STRATEGY

Mutation

Parent

5.5	8.0

Generate λ *normally-distributed* random vectors

RX1	RY1
RX2	RY2
RX3	RY3

Add each of the λ mutating vectors to the initial solution

5.5 + RX1	8.0 + RY1
5.5 + RX2	8.0 + RY2
5.5 + RX3	8.0 + RY3

1- Initial Solution

2 - Initial Population

3 - Evaluation

4 - Selection

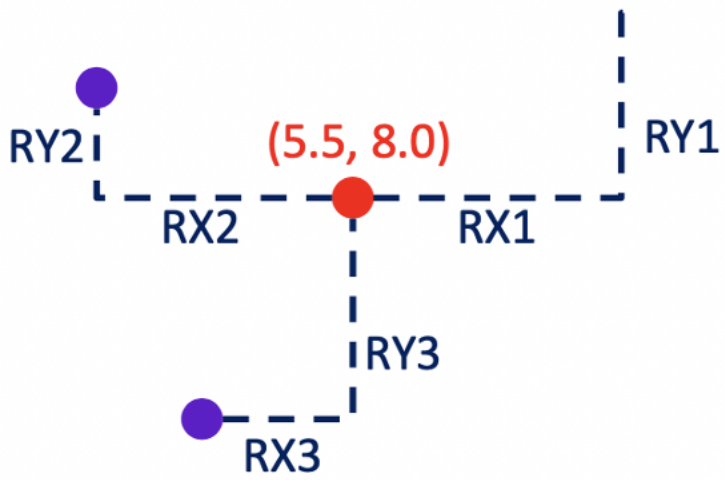
5 - Reproduction

Recombination

6 - Termination

THE BASIC EVOLUTION STRATEGY

Mutation



5.5	8.0

RX1	RY1
RX2	RY2
RX3	RY3

5.5 + RX1	8.0 + RY1
5.5 + RX2	8.0 + RY2
5.5 + RX3	8.0 + RY3

1- Initial Solution

2 - Initial Populatior

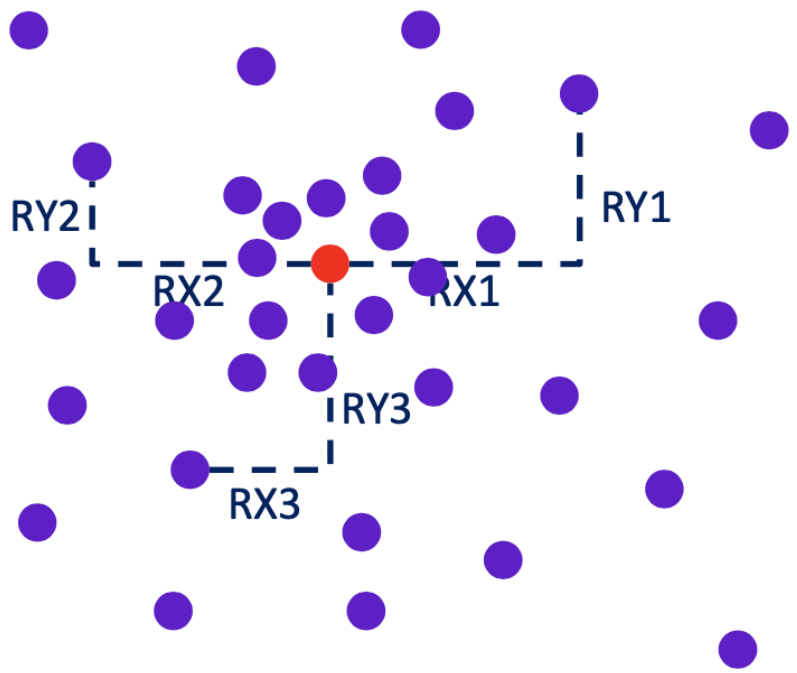
3 - Evaluation

4 - Selection

5 - Reproduction

6 - Termination

Mutation



5.5	8.0

RX1	RY1
RX2	RY2
RX3	RY3

5.5 + RX1	8.0 + RY1
5.5 + RX2	8.0 + RY2
5.5 + RX3	8.0 + RY3

1- Initial Solution

2 - Initial Population

3 - Evaluation

4 - Selection

5 - Reproduction

Recombination

6 - Termination

GAUSSIAN DISTRIBUTION

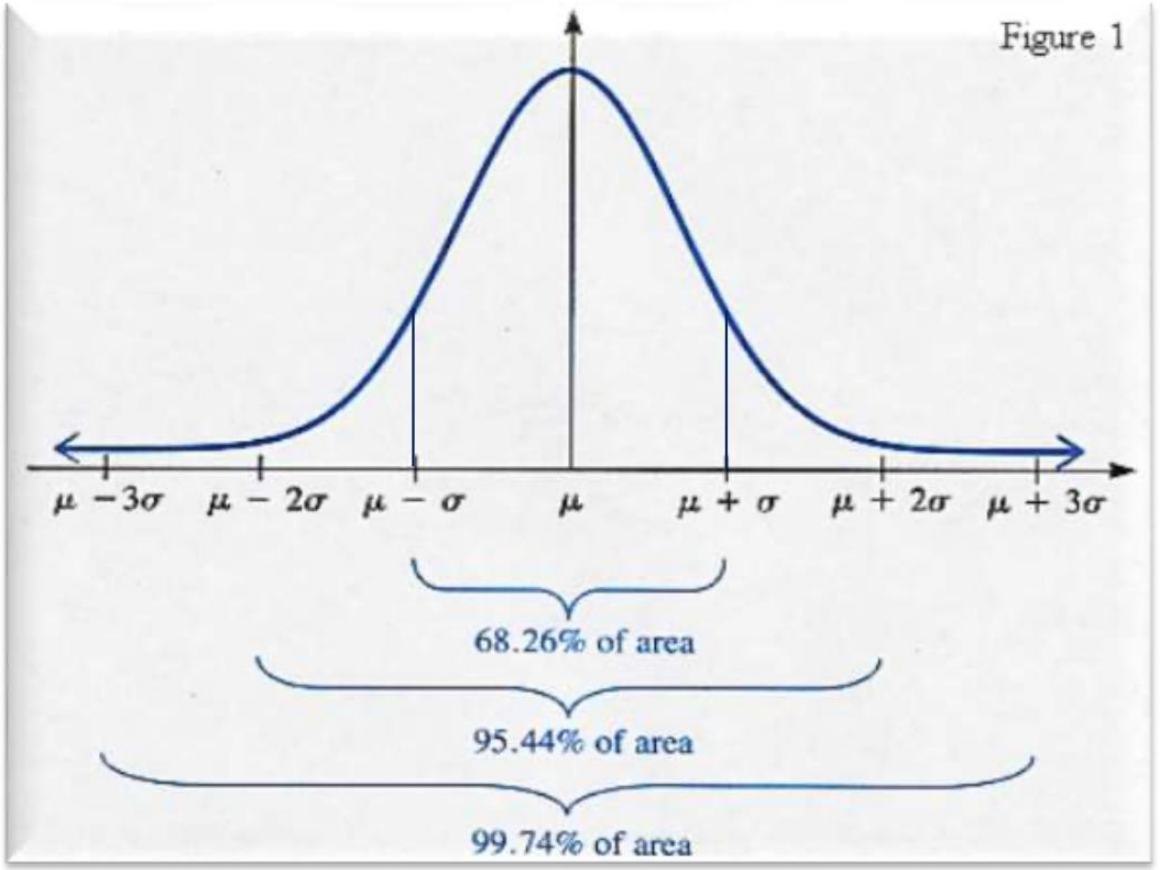
$$N(\mu, \sigma^2) \equiv \mu + \sigma.N(0, 1)$$

$$\mu + \sigma.N(0, I)$$

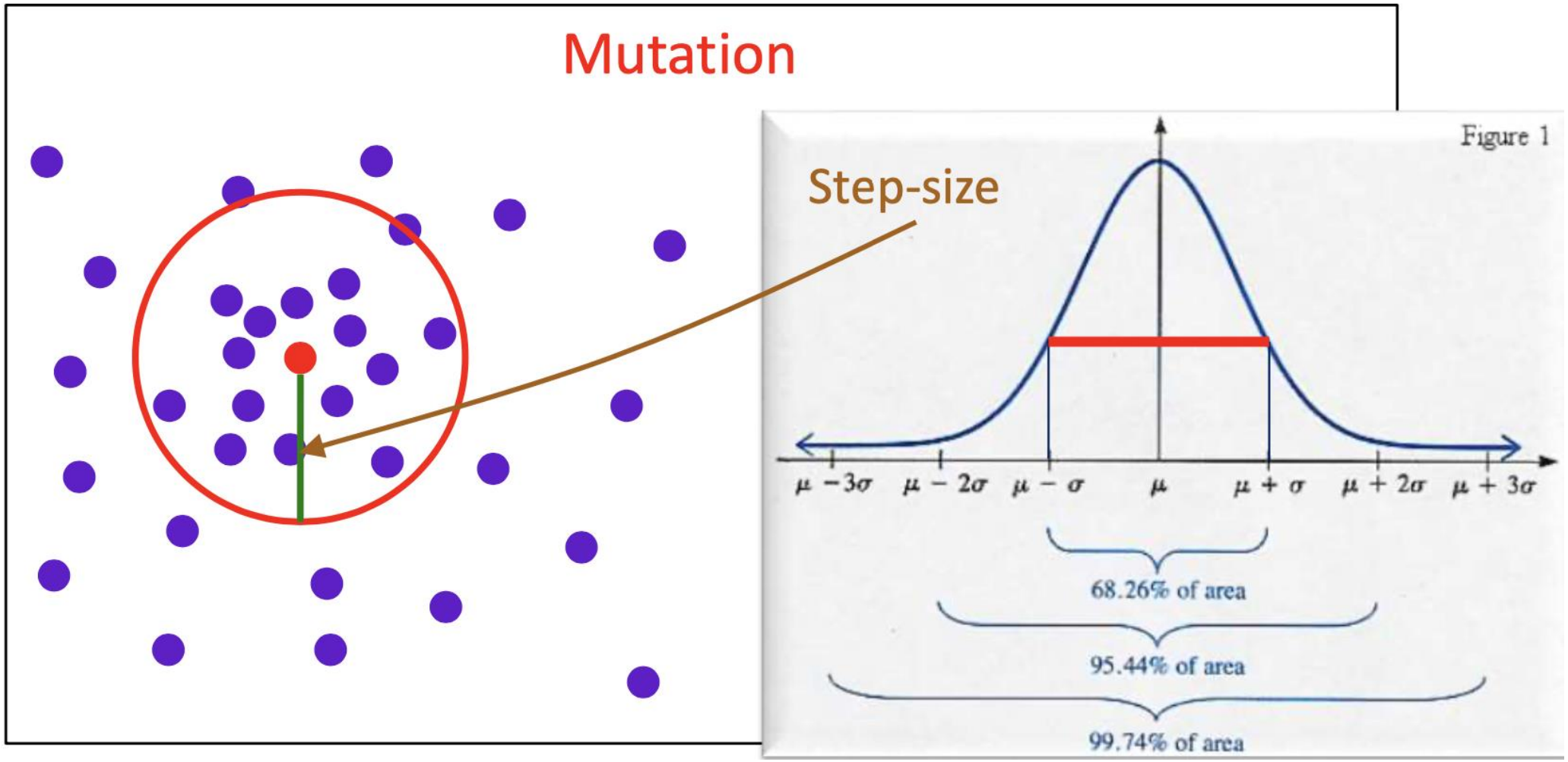
Multivariate

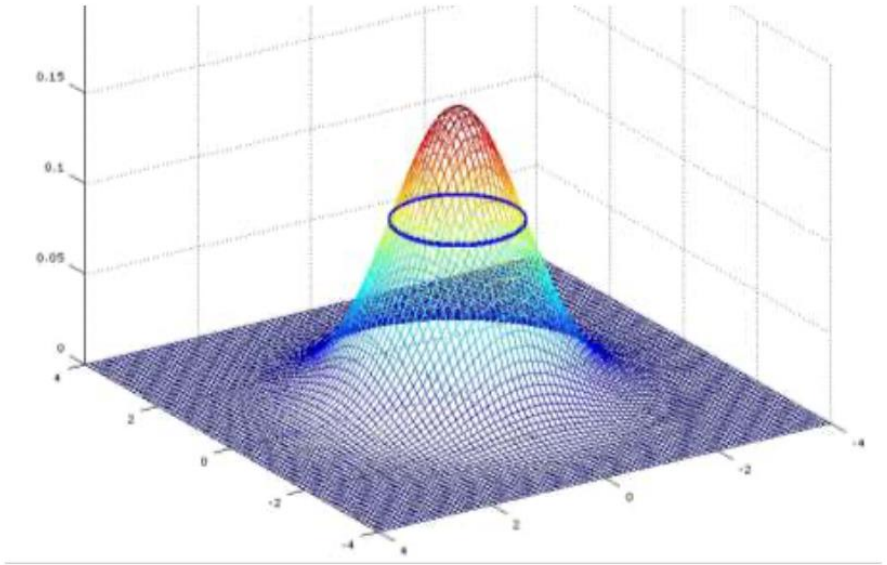
Mean

(Standard Deviation)^2 = Variance

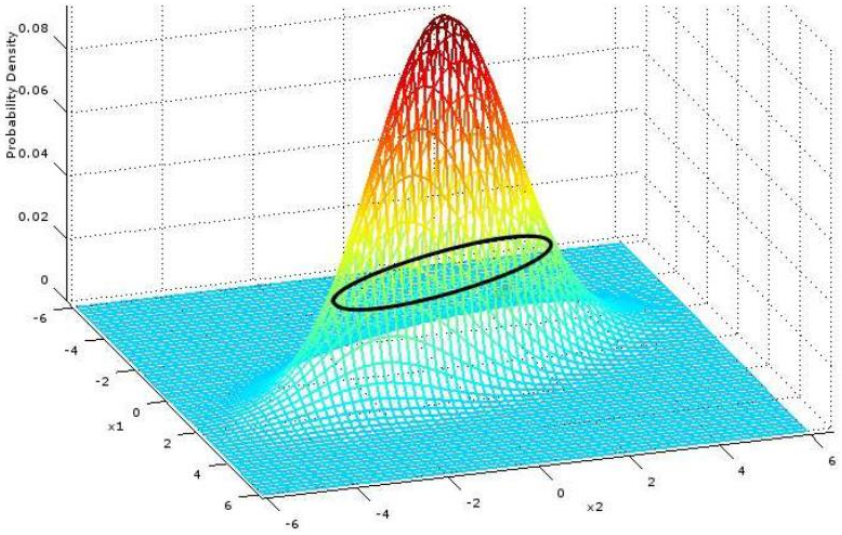


Mutation


Step-size
Figure 1
$\mu - 3\sigma$
$\mu - 2\sigma$
$\mu - \sigma$
μ
$\mu + \sigma$
$\mu + 2\sigma$
$\mu + 3\sigma$
68.26% of area
95.44% of area
99.74% of area



VS



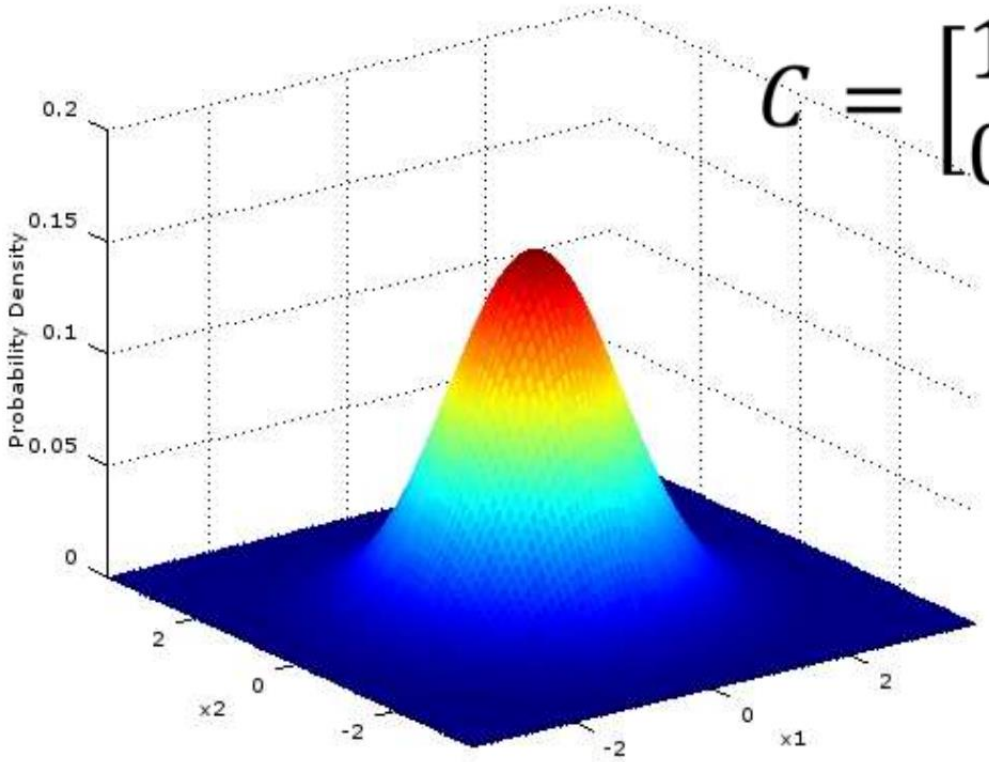
$N(0, I)$

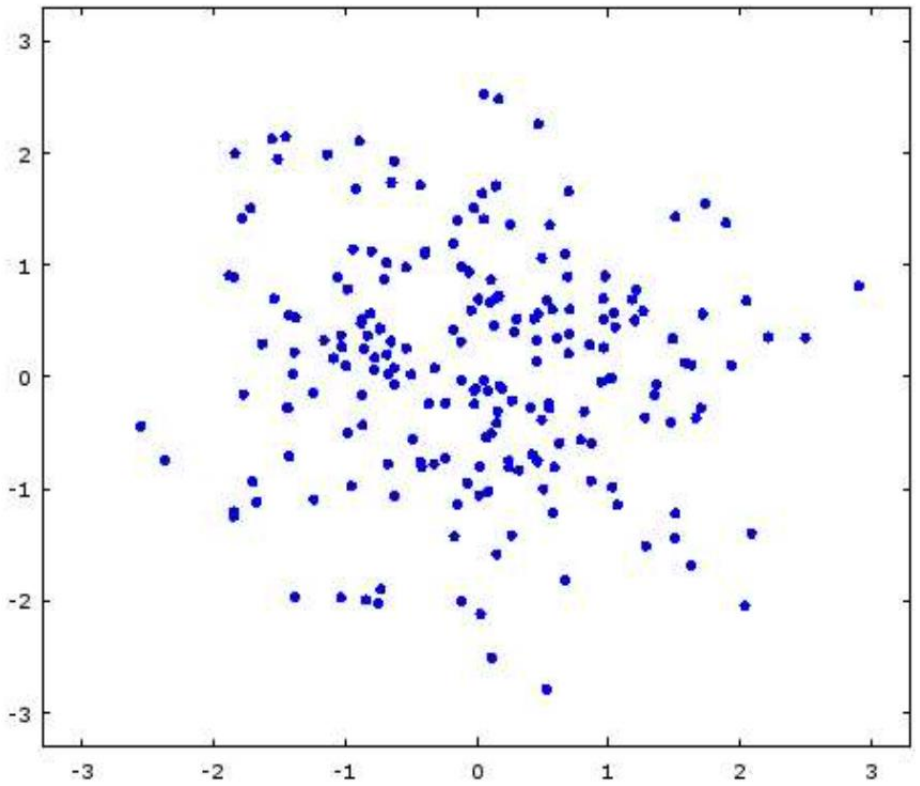
$N(0, C)$

Identity Matrix

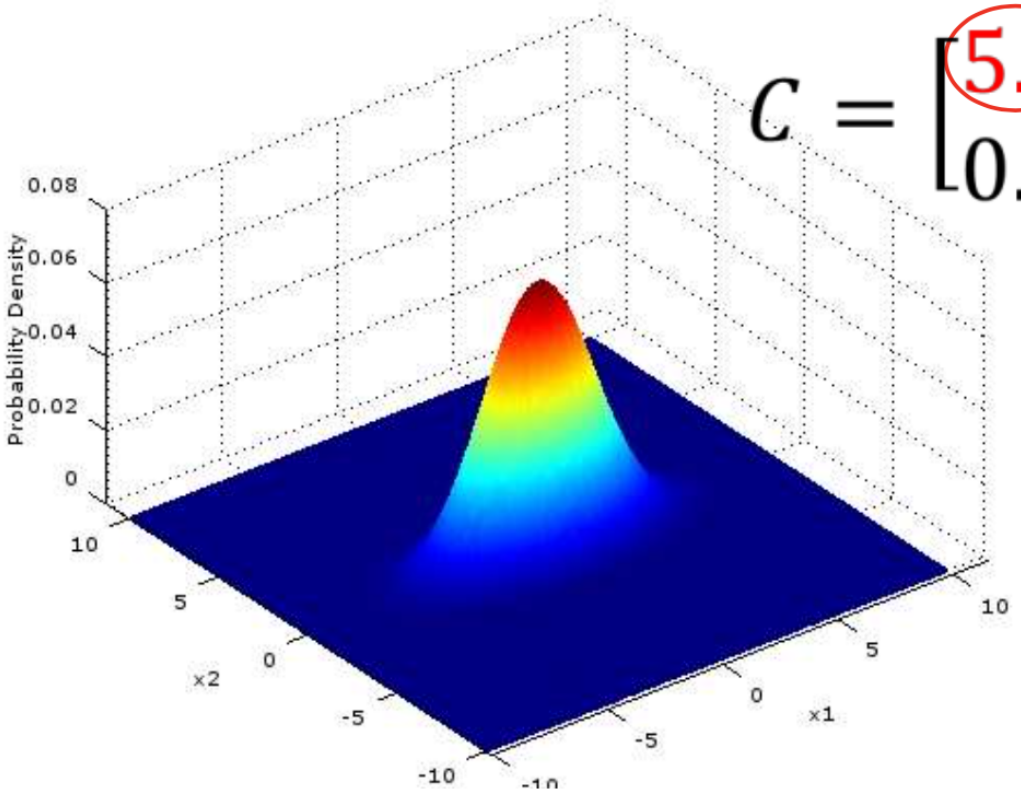
Adapt the **Covariance Matrix**

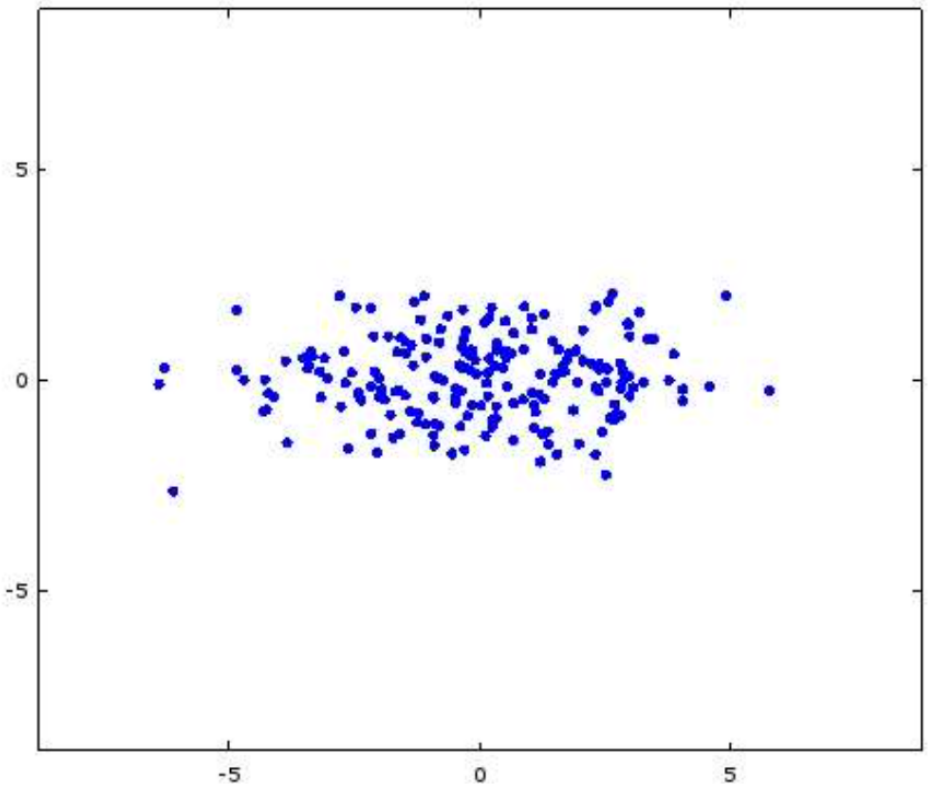
$$C = \begin{bmatrix} 1.0 & 0.0 \\ 0.0 & 1.0 \end{bmatrix}$$



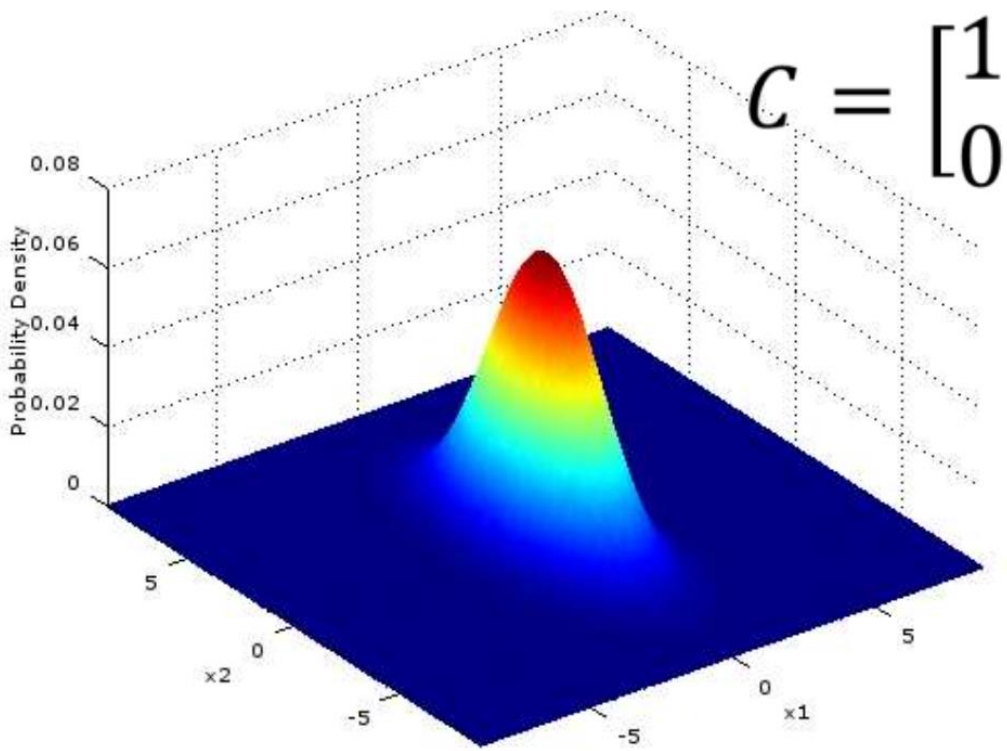


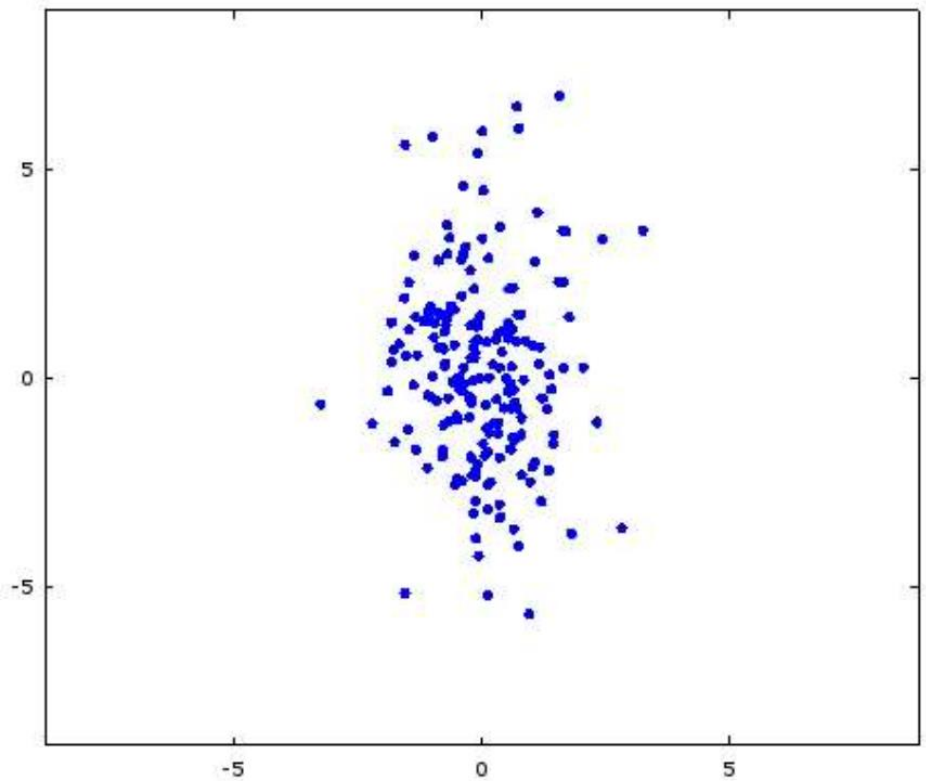
$$C = \begin{bmatrix} 5.0 & 0.0 \\ 0.0 & 1.0 \end{bmatrix}$$



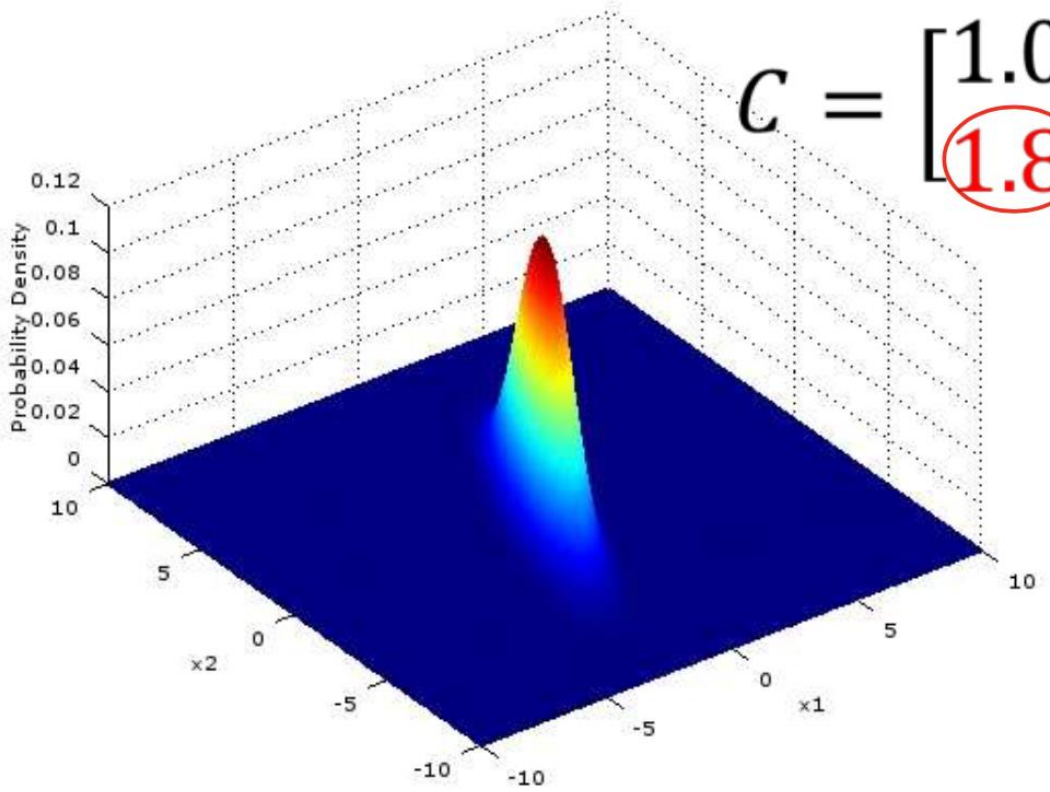


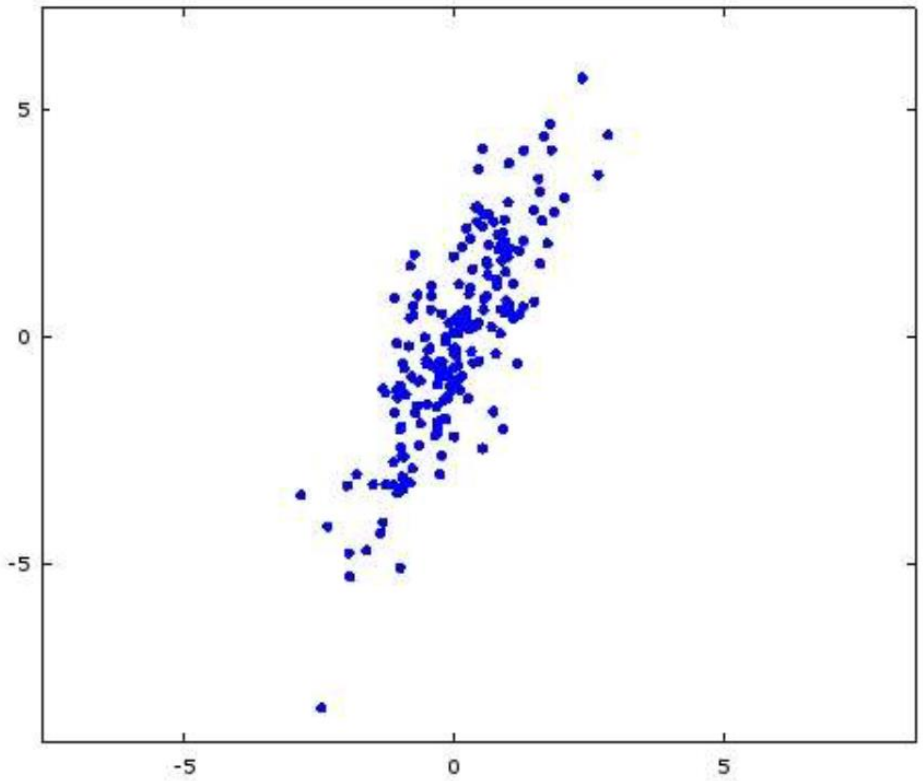
$$C = \begin{bmatrix} 1.0 & 0.0 \\ 0.0 & 5.0 \end{bmatrix}$$



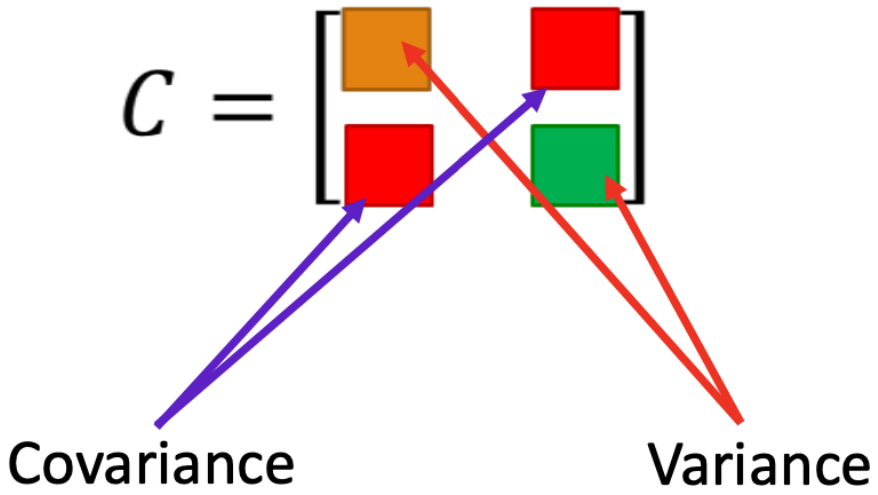


$$C = \begin{bmatrix} 1.0 & 1.8 \\ 1.8 & 5.0 \end{bmatrix}$$





COVARIANCE-MATRIX ADAPTATION (CMA)



SURVIVOR SELECTION

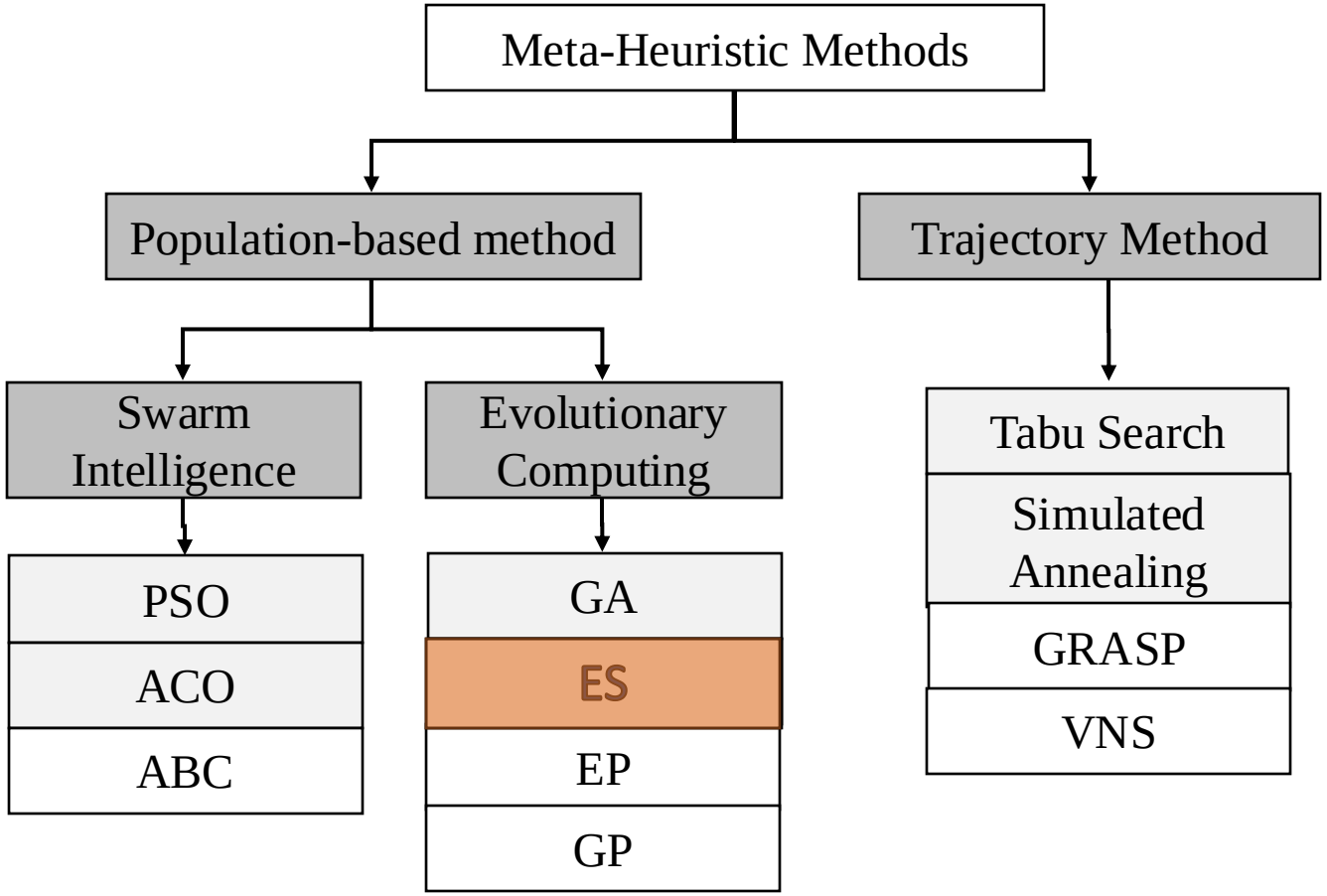
- Applied after creating λ children from the μ parents by mutation and recombination
- Deterministically chops off the "bad stuff"
- Basis of selection is either:
  - The set of children only: (μ,λ)-selection
  - The set of parents and children: $(\mu+\lambda)$-selection

- $(\mu+\lambda)$-selection is an elitist strategy
- (μ,λ)-selection can "forget"
- Often (μ,λ)-selection is preferred for:
  - Better in leaving local optima
  - Better in following moving optima
  - Using the + strategy bad σ values can survive in $\langle x,\sigma \rangle$ too long if their host x is very fit
- Selective pressure in ES is very high ($\lambda \approx 7 \cdot \mu$ is the common setting)

Swarm Intelligence

Introduction

META-HEURISTICS



SWARM INTELLIGENCE

A recent computational metaphor that is inspired by the collective behaviour of social insects, vertebrates, animals, bacteria.

INSPIRATION FROM NATURE

Swarm: A group of agents which communicate with each other by acting on their local environment and their interaction results in distributive collective problem solving strategy.

SWARM INTELLIGENCE

- The interaction between the agents results in distributive collective problem solving strategy.

- This complex behavior is not a property of ***any single individual*** (or agent) of the swarm but rather ***emerges from their interaction***.

- Interaction in biological systems happens in a number of ways:
  - Direct: physical contact, or by visual, audio or chemical perceptual inputs,
  - Indirect: via local change to the environment, ***Stigmergy***

STIGMERGY

- The word *Stigmergy* is due to Pierre-Paul Grasse who was studying the behavior of termites,

- He observed that the insects react to signals that activate a ***genetically encoded reaction***,

- The effect of these reactions serve as new signals for both:
  - The insect that produced them,
  - Other insects in the colony.

EXAMPLES FROM NATURE: HOW TERMITES BUILD THEIR NEST

Nest building by termites, bees, ants

https://www.youtube.com/watch?v=zhy27gMTMTg

EXAMPLES FROM NATURE

Ants’ agricultural behaviour (© Alex Wild)

Wasps’ nest building (© Dominique Snyers)

EMERGENT BEHAVIOR

- **Emergent behavior has been observed in other non-biological systems.**

i. **Stock market** is a good example of collective behavior.
   - Each investor has limited knowledge and affects the market in a limited way
   - Interactions create a market behavior

i. Another example is **traffic patterns** and self organizing behavior that result in traffic.

ii. A third example is the **spatial structure of galaxies** is an emergent property of distribution of energy and matter in the universe.

PROPERTIES

These behaviour are inspiring as they have some interesting properties:

- **Flexibility:** the system performance is adaptive with respect to internal or external changes,
- **Robustness:** the system always perform even if some individuals fail,
- **Decentralization:** the control is distributed among the individuals,
- **Self-organization:** global behaviours appear out of local individual interactions.

BIRD FLOCKING

Mystery: how the starling flies in murmurations, or flocks, without colliding.

This short film was shot in the Netherlands, and it captures the birds gathering at dusk, just about to start their "performance.

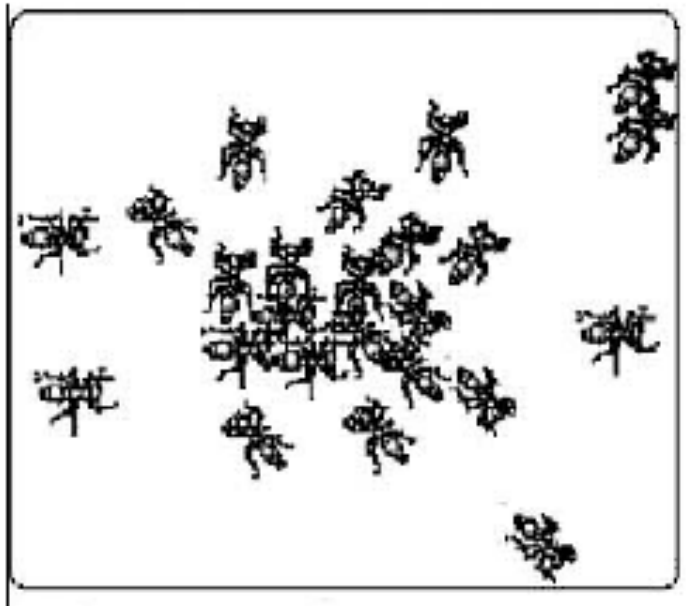
https://www.youtube.com/watch?v=KGegmMWsgu8

MODELS OF BEHAVIOUR

- Couzin et. al, identified different behavioural models:
  - Swarm,
  - Torus,
  - Dynamic parallel group,
  - Highly parallel group.

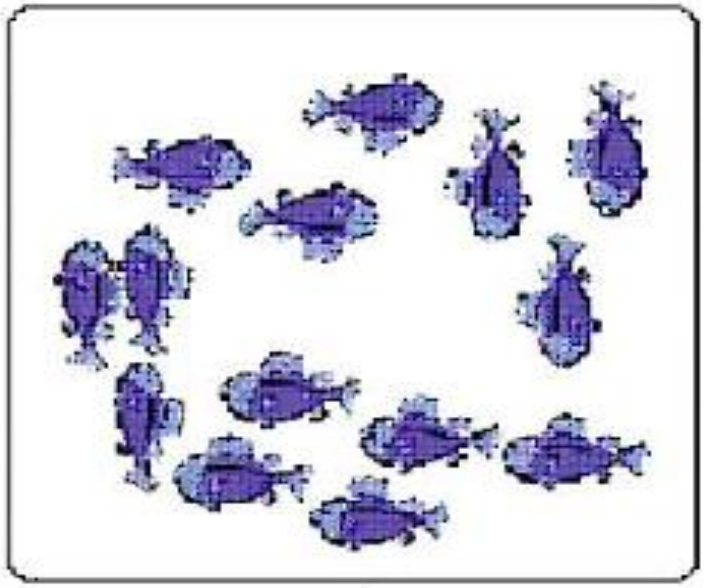
ID. Couzin, J. Krause, R. James, GD. Ruxton, NR. Franks.”Collective Memory and Spatial Sorting in Animal Groups”. Journal of Theoretical Biology, 218, pp. 1–11, 2002.

MODELS OF BEHAVIOUR



Swarm

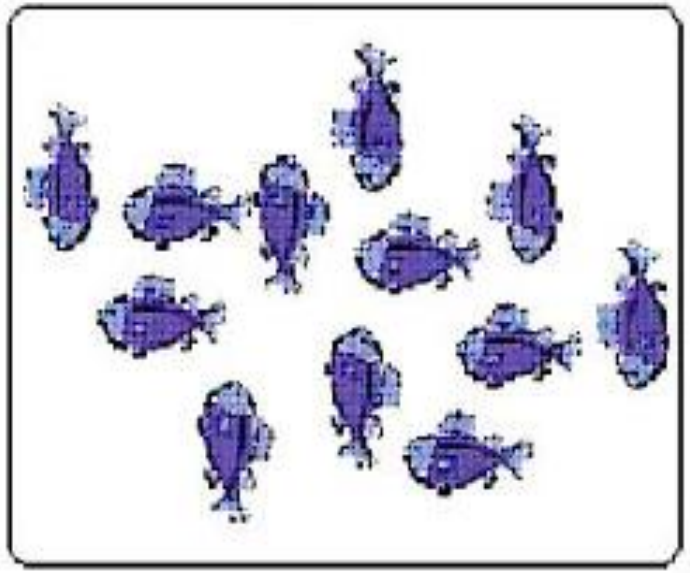
Coherent group with low level of polarization (parallel alignment),



Torus

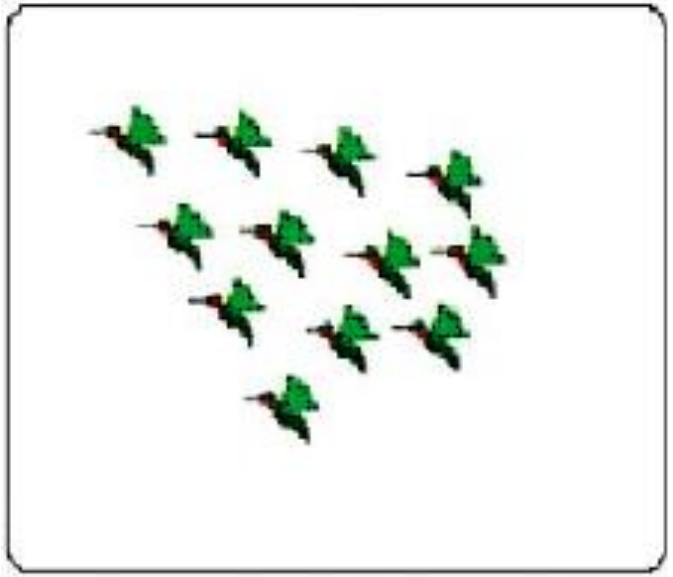
Entities rotate around an empty core with a random direction of rotation

MODELS OF BEHAVIOUR



Dynamic parallel group

Entities are polarized and move as a coherent group, but they can move through out the group and density and group form fluctuates



Highly parallel group

Same as previous one but with minimum fluctuations

COMPUTATIONAL SWARM INTELLIGENCE

- Building computational models to solve complex problems based on models of behaviors in biological swarms.

- Design issues involve:
  i. modeling the agent (or individual),
  ii. the interaction process,
  iii. adaptation and cooperation.

- Ant Colony Optimization (**ACO**) and Particle Swarm Optimization (**PSO**) are two examples of computational swarm intelligence models.

ACO AND PSO

- ACO models the very simple behavior of pheromone trail following of ants.
- PSO models 2 simple behaviors: moving towards its best closest neighbor and moving back to its own best state.
- Communication is different in each. PSO uses neighbors broadcast (direct communication), ACO uses *stigmergy* through pheromone perception.
- The agent in both is very simple and doesn't learn at the individual level.

A SUCCESSFUL SI TOOL

There are five basic principles that should be present:

- **Proximity principle:** The basic units of a swarm should be capable of giving the respond back to to environmental variance triggered by interactions among agents. However, some fundamental behaviors are shared such as living-resource searching and nest-building.
- **Quality:** A swarm should be able to respond to quality factors such as determining the safety of a location.
- **Principle of diverse response:** Resources should not be concentrated in a narrow region. The distribution should be designed so that each agent will be maximally protected facing environmental fluctuations.
- **Principle of stability:** The population should not change its mode of behavior every time the environment changes.
- **Principle of adaptability:** The swarm is sensitive to the changes in the environment that result in different swarm behaviour

**MM. Millonas .“Swarms, phase transitions, and collective intelligence”. In CG. Langton Ed., Artificial Life III, Addison Wesley, Reading, MA, 1994.

Ant Colony Optimization

Ants: An Introduction

INSPIRATION FROM NATURE

ANTS

- There are about 10^{18} living insects
- About 2% of all insects are social
- About 50% of all social insects are ants
- The total weight of ants is about the total weight of humans
- Ant colony size can be as few as 30 ants and as large as million ants
- Ants colonize the world since 100,000,000 years, humans only since 50,000 years

ANTS

- Ants use Stigmergy that was discovered by Pierre-Paul Grasse who was studying termites.

- Properties of Stigmergy
  - indirect agent interaction modification of the environment
  - environmental modification serves as external memory
  - work can be continued by any individual

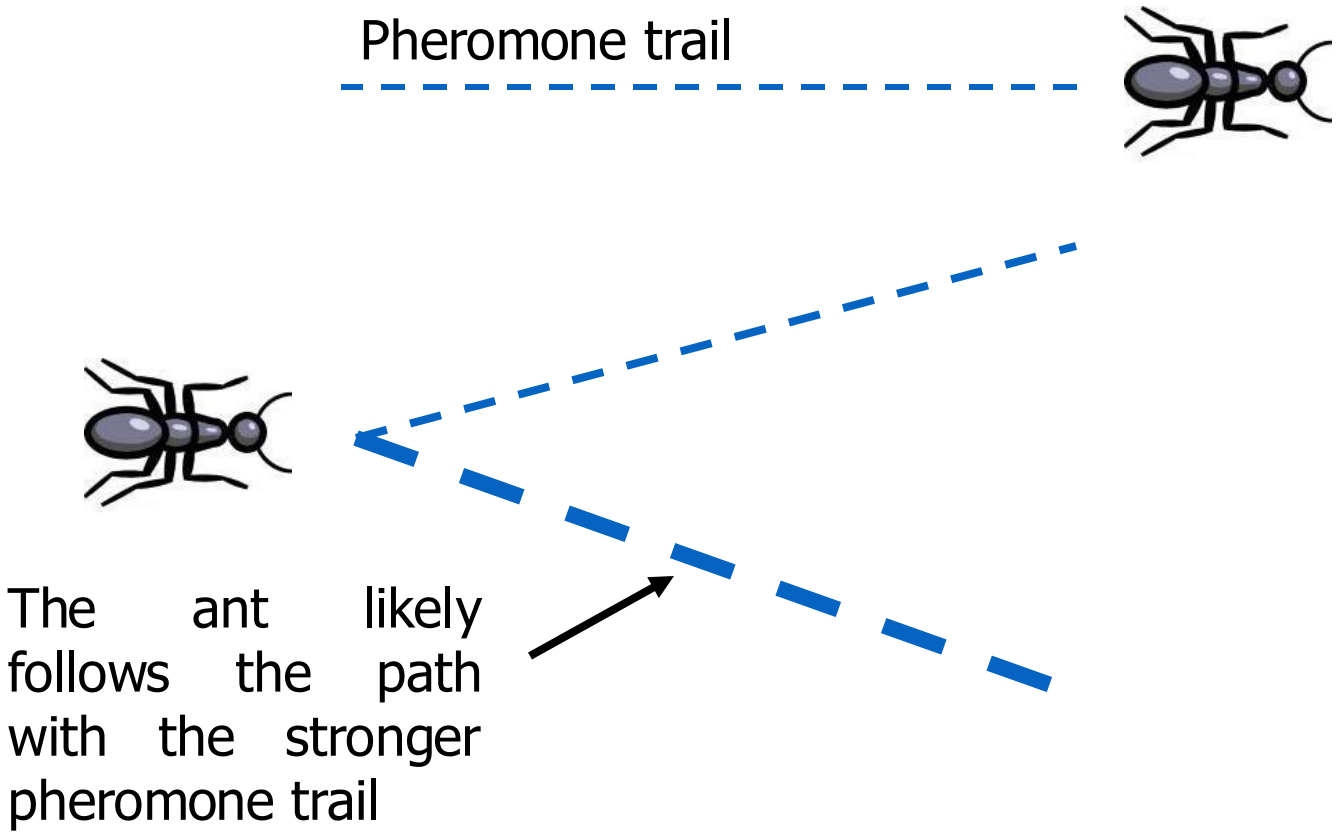
- Ants walking, to or from, a food source deposit a chemical substance on its way,
  - This substance is referred to as the ***pheromone***.

- P. P. Grasse."Recherches sur la biologie des termites champignonnistes (*Macrotermitina*)". Ann. Sc. Nat., Zool. Biol. anim., 6, 97, 1944.
- P. P. Grasse."La reconstruction du nid et les coordinations interindividu- elles chez *bellicositermes natalensis* et *cubitermes* sp. La theorie de la stig- mergie: essai d'interpretation du comportement des termites constructeurs". Insectes Sociaux, 6, 41, 1959.

PHEROMONE

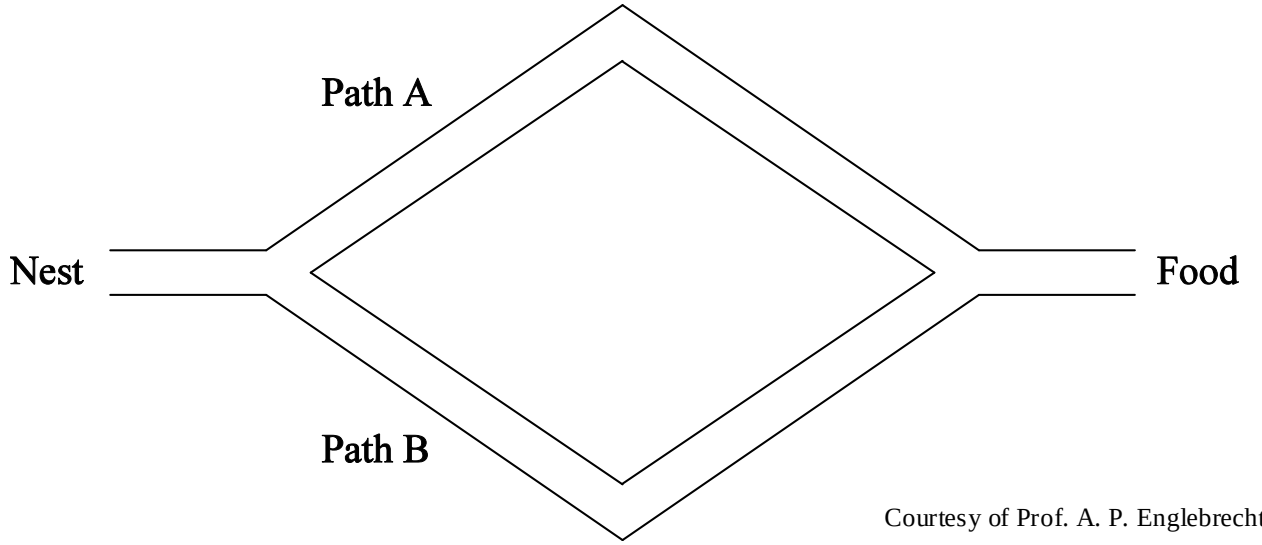
- Other ants can sense this pheromone,
- Its presence will influence the ants' choice of a certain path,
- Ants tend to follow strong pheromone concentrations.
- ***Pheromone trails*****,** is formed by the pheromone deposited on the ground,
- Pheromone trails help the ants to reach good food sources that have been previously identified by other ants.

PHEROMONE


Pheromone trail
The ant likely follows the path with the stronger pheromone trail

BINARY BRIDGE EXPERIMENT

- A ***binary bridge experiment*** was done by Deneubourg et al.,
- The ants nest was connected to a food source through two equal length bridges,
- The ants behaviour was observed until they converge towards the use of the same bridge.



Courtesy of Prof. A. P. Englebrecht, author of text

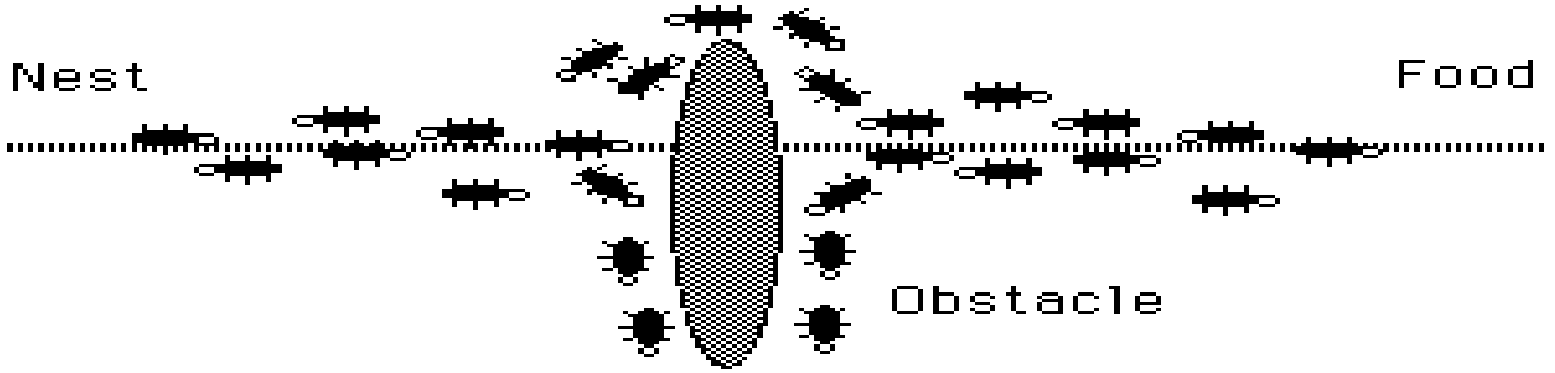
J. L. Deneubourg, S. Aron, S. Goss and J. M. Pasteels."The self- organizing exploratory pattern of the Argentine ant". Journal of Insect Behaviour, 3, 159, 1990.

BINARY BRIDGE EXPERIMENT

- Initially, the ants randomly choose the bridge to follow,
- After a while, the ants will tend to follow the bridge with ***stronger pheromone trails***,
- The selection of one bridge is due to ***random fluctuations*** causing it to have higher pheromone concentration.
- Pheromone trail acts as a ***collective memory*** for the ants to communicate through by sensing and recording their foraging experience
- Pheromone ***evaporates*** over time introducing some changes in the environment.

SHORTEST PATH AROUND AN OBSTACLE

Initial movements



After deposit

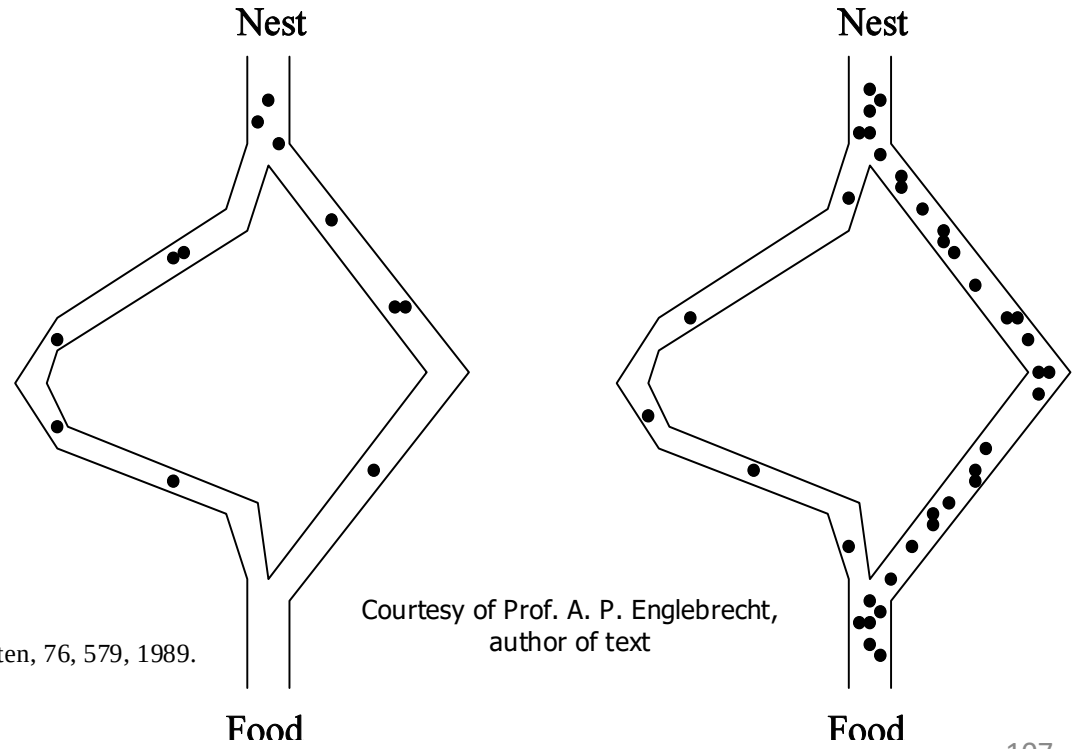
BRIDGES WITH NON-EQUAL LENGTH

- Another experiment was carried by Goss et al. [4] were the bridges were not of equal length. One bridge was significantly shorter than the other,

- In this case, the ants following the shorter bridge by chance will be the first to reach the food source.

- They will be also the first to reach the nest as they will take the same path home; since it will have more pheromone,

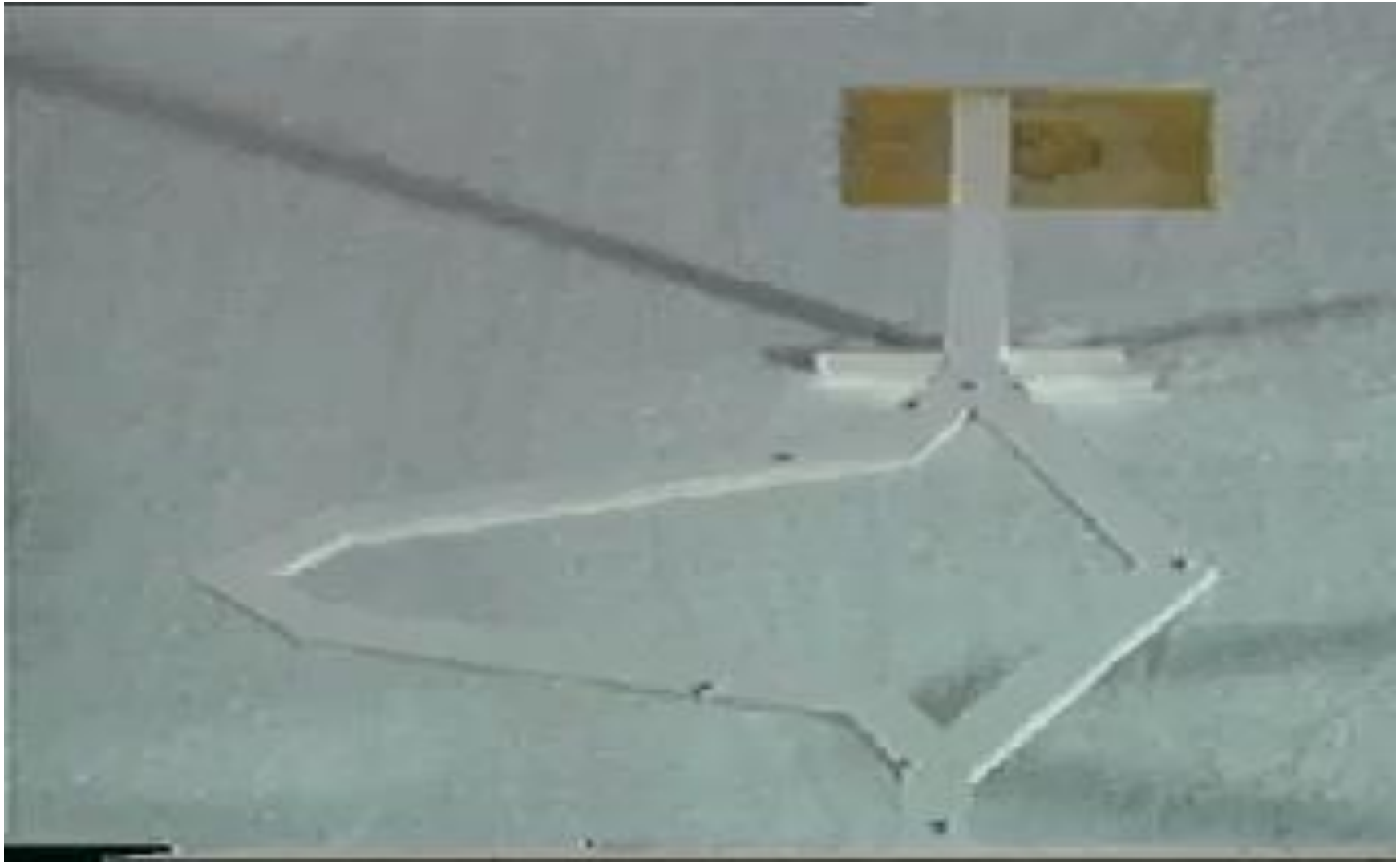
- The ants finally converged to the shorter path due to the pheromone depositing mechanism.



Courtesy of Prof. A. P. Englebrecht, author of text

S. Goss, S. Aron, J. L. Deneubourg and J. M. Pasteels."Self-organized shortcuts in the Argentine ant". Naturwissenschaften, 76, 579, 1989.

BRIDGES WITH NON-EQUAL LENGTH



Ant Colony Optimization

Algorithm

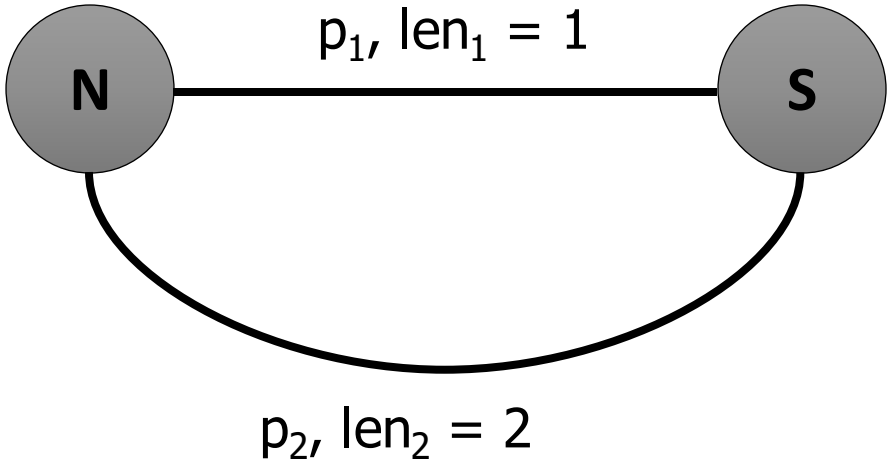
ACO - INTRODUCTION

- Introduced by M. Dorigo in 1992,
- Inspired by the foraging behaviour or real ants,
- Successfully applied in many application including:
  - Traveling Salesman Problem (TSP),
  - Packet Routing in Network Telecommunications,
  - Scheduling Problems,
  - Vehicle Routing Problems.

M. Dorigo.”Optimization, Learning and Natural Algorithms”. Ph.D. thesis, DEI, Politecnico di Milano, Italy, pp. 140, 1992.

ACO - SIMULATION

- To simulate the behaviour of ants: assume we have a nest and a food source connected through two paths with different lengths.



- Assign initial artificial pheromone values for every path, τ_1 and τ_2,

- Initially, both values are equal:

$$\tau_1 = \tau_2 = c > 0$$

ACO - SIMULATION

- Place *m* ants at the nest,

- For each ant *k*,
  - Traverses path 1 with a probability:

$$p_1 = \frac{\tau_1}{\tau_1 + \tau_2}$$

  - Traverses path 2 with a probability:

$$p_2 = 1 - p_1$$

ACO - SIMULATION

- An evaporation phase is applied:
  - To simulate the evaporation of real pheromone,
  - To avoid quick convergence to sub-optimal paths.

$$\tau_i = (1-\rho) \times \tau_i, \quad \rho \in (0,1)$$

ρ specifies the rate of evaporation, when equal to 1, the move becomes random.

- Each ant leaves more pheromone on its traversed path:

$$\tau_i = \tau_i + \frac{1}{len_i}$$

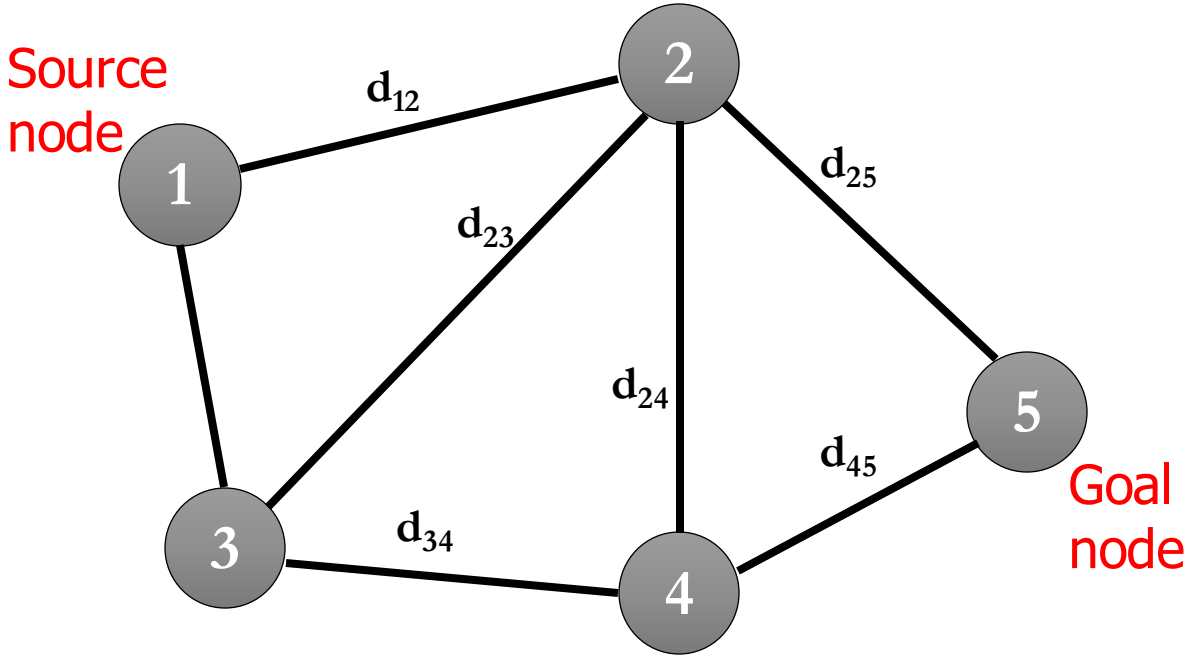
REAL AC VS. ARTIFICIAL AC

Real AC		Artificial AC
• Real ant and pheromone	⟷	• Artificial ant (agent) and pheromone (value)
• Food	⟷	• Solution
• Continuous	⟷	• Discrete
• Pheromone update while moving	⟷	• Pheromone update after traverse
• Solutions are evaluated implicitly	⟷	• Explicit function to evaluate solutions

ACO - ALGORITHM

- Let ***G=(N,E)*** denote a graph, where ***N*** is the set of nodes (vertices) and ***E*** is the set of arcs (edges),

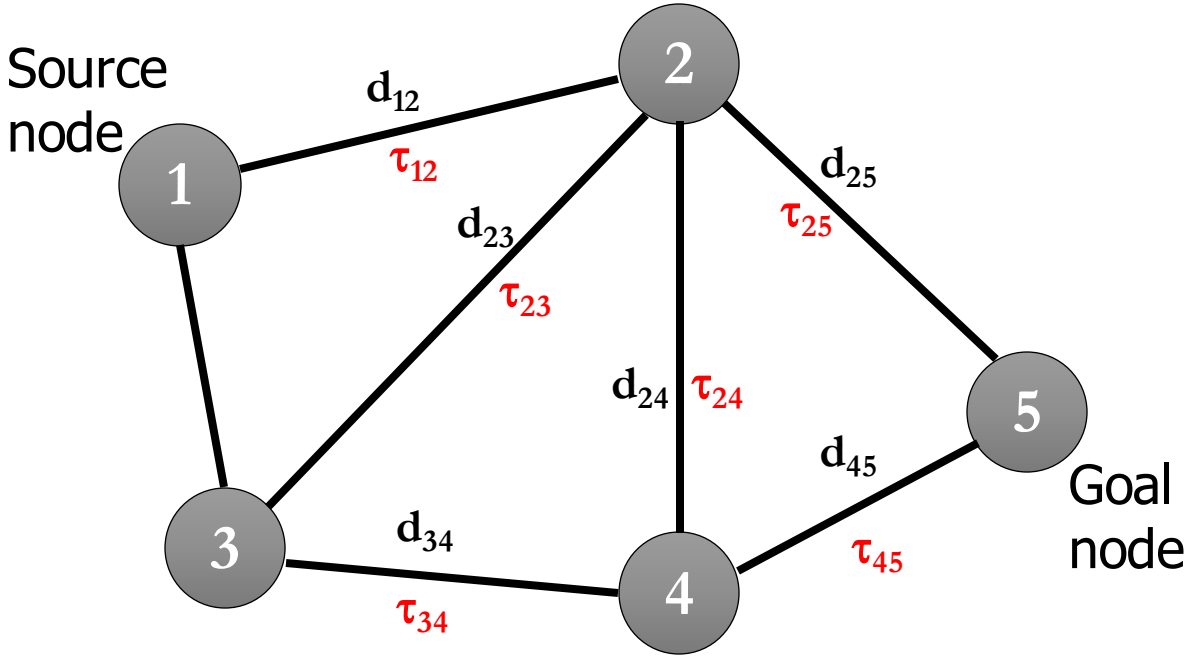
- Each arc ***(i, j)*** is associated with a value d_{ij} denoting the ***distance*** between nodes ***i*** and ***j***,



- A simple ant algorithm could be used to find the shortest path between two given nodes in the graph.

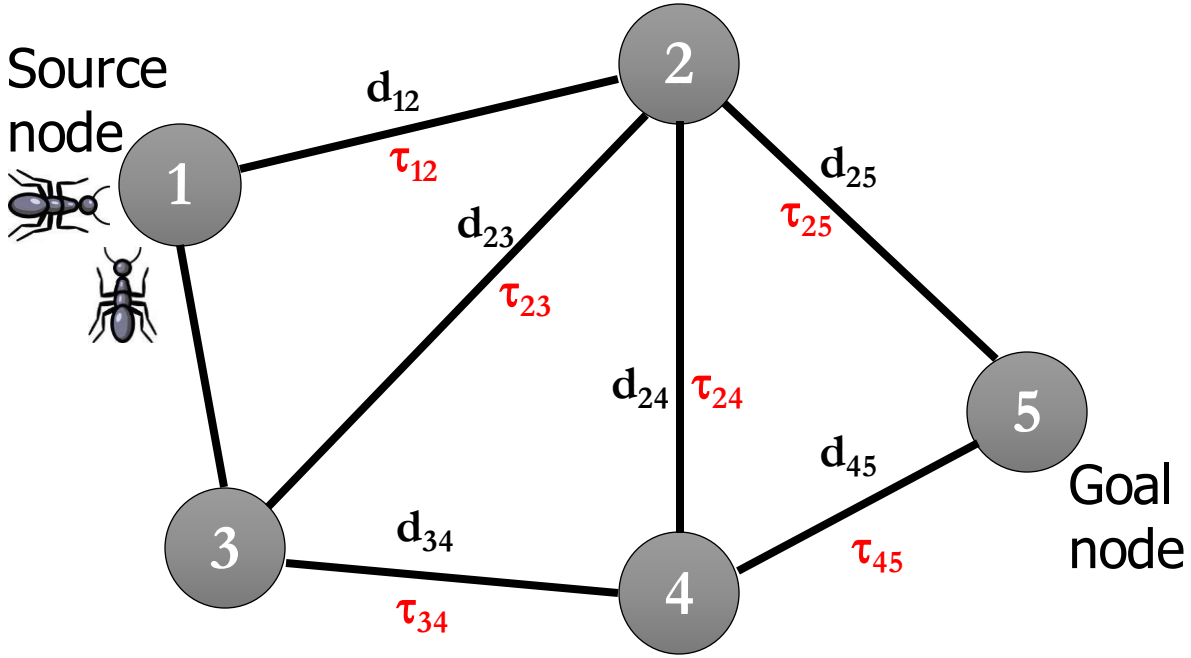
ACO - ALGORITHM

- Let each arc (i, j) be associated with a value τ_{ij} called the ***artificial pheromone***,
- At the beginning, the same small amount of artificial pheromone is placed on all arcs,



ACO - ALGORITHM

- Place a group of ***m*** ants at the source node.



ACO - ALGORITHM

- At each node ***i***, the ant has a choice to move to any of the ***j*** nodes connected to it,

- Each node $j \in N_i$ connected to ***i*** has a probability to be selected by ant ***K*** equal to:

$$p_{ij}^{k} = \begin{cases} \dfrac{\tau_{ij}^{\alpha} / d_{ij}^{\beta}}{\sum_{n \in N_i} \tau_{in}^{\alpha} / d_{in}^{\beta}} & if \quad j \in N_i \\ 0 & if \quad j \notin N_i \end{cases}$$

- α and β are chosen to balance the ***local*** vs. the ***global*** search ability of the ant.

ACO - ALGORITHM

- Evaporate the artificial pheromone:

$$\tau_i = (1-\rho)\times\tau_i, \quad \rho \in (0,1]$$

- The ant deposits extra pheromone on the arc it chooses:

$$\tau_{ij} = \tau_{ij} + \Delta\tau$$

- This will increase the probability of a subsequent ant choosing the same arc and referred to as ***online step_by_step*** pheromone update.

ACO - ALGORITHM

- Different approaches exist for choosing the value of $\Delta\tau$:
  - ***Ant density model***, adding ***Q***, a constant value, hence the final pheromone added to the edge will be proportional to the number of ants choosing it. This doesn't take the edge length into account,
  - ***Ant quantity model***, adding Q/d_{ij}, taking the edge length into account, hence enforcing the ant local search ability.
  - ***Online delayed*** pheromone update, sometimes referred to as the ***ant cycle model***,
    - After the ant builds the solution, it traces the path backward and updates the pheromone trails on the visited arcs according to the solution quality.

ACO - ALGORITHM

- The online delayed pheromone update is done by adding:

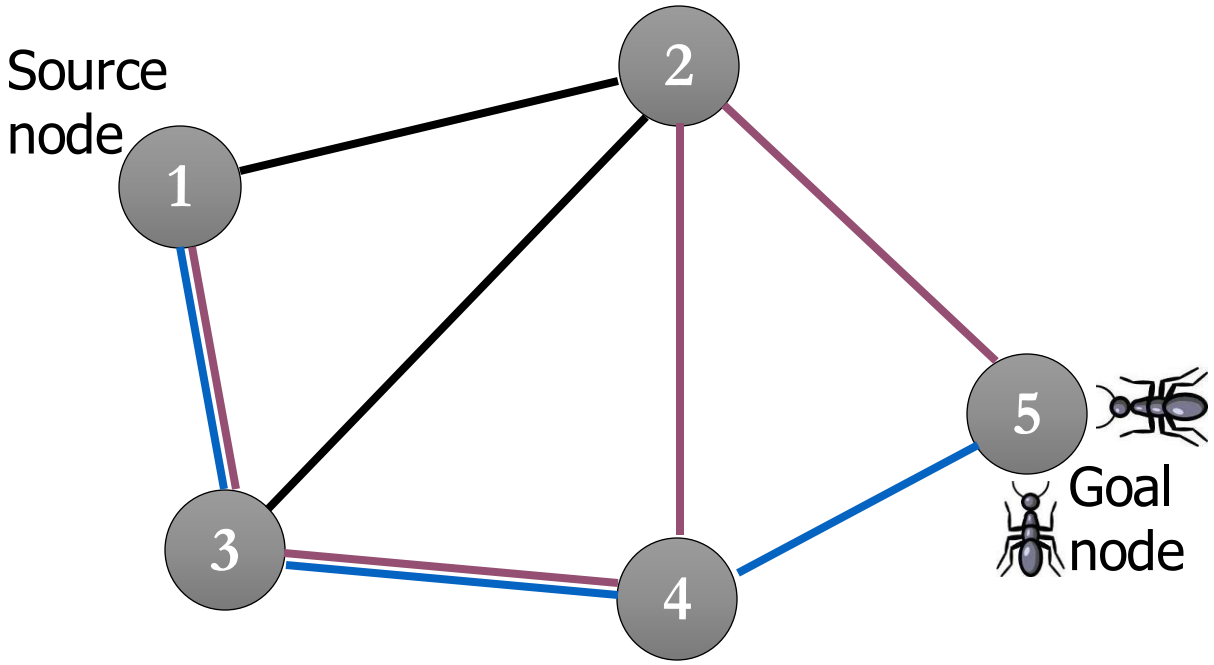
$$\Delta \tau_{ij} = \frac{Q}{L^k}$$

for every arc (i, j) on the path, where $\boldsymbol{L^k}$ is the length of the path found by ant $\boldsymbol{k}$.

- Done after the evaporation phase

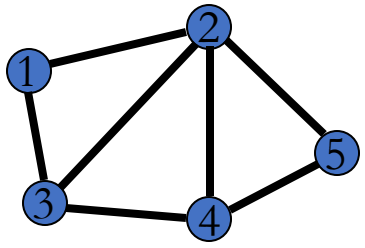
- Repeat the process for different initialization of pheromone

ACO - ALGORITHM

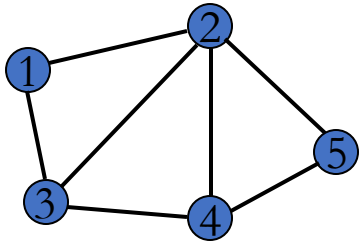


ACO - ALGORITHM

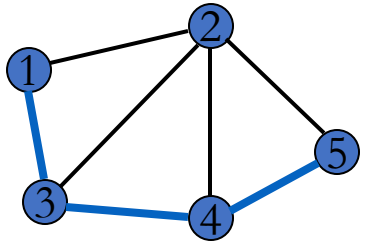
Start

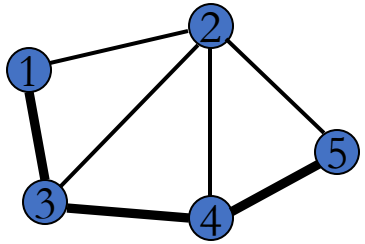


Evaporation

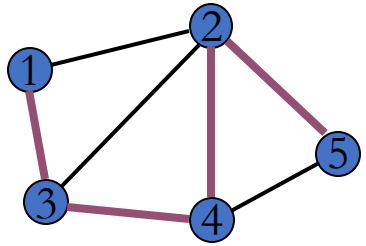


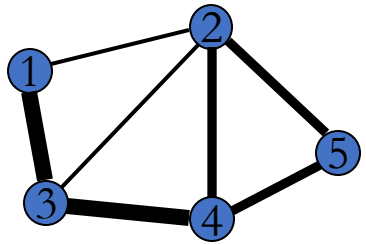
Solution 1





Solution 2





ACO METAHEURISTIC

- Algorithm Ant colony optimization metaheuristic
- Set parameters, initialize pheromone trails
- while termination conditions not met do
  - Construct Ant Solutions
  - Apply Local Search (optional)
  - Update Pheromones
- end while

ACO ALGORITHM

- Termination conditions:
  - Max number of iterations reached
  - Acceptable solution reached
  - All ants (or most of them) follow the same path, i.e stagnation.

PARAMETERS

- **Number of ants**:
  - More ants more computations, but also more exploration.
- **Max number of iterations**: has to be enough to allow convergence.
- **Initial pheromon**e: constant, random, max value, small value.
- **Pheromone decay parameter** ρ

COMPONENTS

- **Transition rule:** probability of selection for the ant
- **Pheromone evaporation rule**
- **Pheromone update rule**
- **Problem heuristic if used**
- **Quality of solution measure**
- **Memory or list for constraints (Tabu list)**
- **Termination Criteria**

ANT SYSTEM ALGORITHM

- ***Ant System Algorithm (AS):***

  - Proposed by Dorigo et al.,
  - Uses the same basic steps outlined in ACO,
  - The online delayed pheromone update is adopted using all the solutions of the current iteration.

$$\Delta\tau_{ij} = \frac{Q}{L^k}$$

- M. Dorigo."Optimization, Learning and Natural Algorithms". Ph.D. thesis, DEI, Politecnico di Milano, Italy, pp. 140, 1992.
- M. Dorigo, V. Maniezzo and A. Colorni." Ant System: Optimization by a Colony of Cooperating Agents". IEEE Trans. Syst., Man, and Cybern. Part B, vol. 26, no. 1, 1996.

ANT COLONY SYSTEM ALGORITHM

- ***Ant Colony System Algorithm (ACS):***

  - Proposed by Gambardella and Dorigo,

  - Based on AS but is different in:
    - Transition rule based on Elitist strategy (balances exploitation vs exploration)
    - Pheromone update rule
    - Local pheromone update
    - Candidate list

M. Dorigo and L. M. Gambardella.”Ant Colony System: a cooperative learning

ACO PARAMETERS

τ_{ij} : Pheromone trail of combination (i,j)

η_{ij} : Local heuristic of combination (i,j)

P_{ij} : Transition probability of combination (i,j)

α : Relative importance of pheromone trail

β : Relative importance of local heuristic

q_0 : Determines the relative importance of exploitation versus exploration

ρ : Trail persistence

ACS

- If $q < q_o$ (q is a random number, q_o is in(0,1)):

$$j = arg\ max_{j \in N_i^k} \left\{ \tau_{ij}(t) \eta_{ij}^{\beta} \right\}$$

- Else use:

$$p_{ij}^{k}(t) = \begin{cases} \dfrac{[\tau_{ij}(t)]^{\alpha} [\eta_{ij}]^{\beta}}{\sum_{k \in allowed_k} [\tau_{ik}(t)]^{\alpha} [\eta_{ik}]^{\beta}} & \text{if } j \in \text{allowed}_k \\ 0 & \text{otherwise} \end{cases}$$

- J element of N^k_i is a randomly selected node give probability P^k_{ij}

- This creates bias towards choices of better quality, q_o controls this bias

ACS

- Uses the offline pheromone update, where all the ants update their last traversed edge only,

$$\tau_{ij} = (1-\rho_2)\tau_{ij} + \rho_2\tau_0$$

- Introduced a new pheromone update rule (***using the best solution (ant) only***).

$$\tau_{ij}(t+1) = (1-\rho_1)\tau_{ij}(t) + \rho_1\Delta\tau_{ij}^{best}(t)$$

- Best here can be ***iteration best or global best***

MAX-MIN ANT SYSTEM ALGORITHM

- ***Max-Min Ant System Algorithm (MMAS):***

  - Proposed by Stutzle and Hoos to overcome stagnation.

  - The update is done using the best solution (best ant) in the current iteration or the best solution over all, also decay in the update of the pheromone

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \rho\Delta\tau_{ij}^{best}(t)$$

  - The values of the pheromone are restricted between τ_{min} and τ_{max}, allows high exploration in the beginning (τ_{max}) and more intensification later.

  - The values of τ_{min} and τ_{max} are chosen experimentally, although they could be calculated analytically if the optimal solution is known,

  - Improved the performance significantly over AS.

T. Stutzle and H. H. Hoos."*MAX-MIN* Ant System". *Future Generation Comput. Syst.*, vol. 16, no 8, 2000.

Ant Colony Optimization

Applications

ACO APPLICATIONS

- Travelling Salesman Problem (TSP),
- Assembly Line Balancing (ALB),
- Cell Assignment in PCS Networks (CA).

- Other

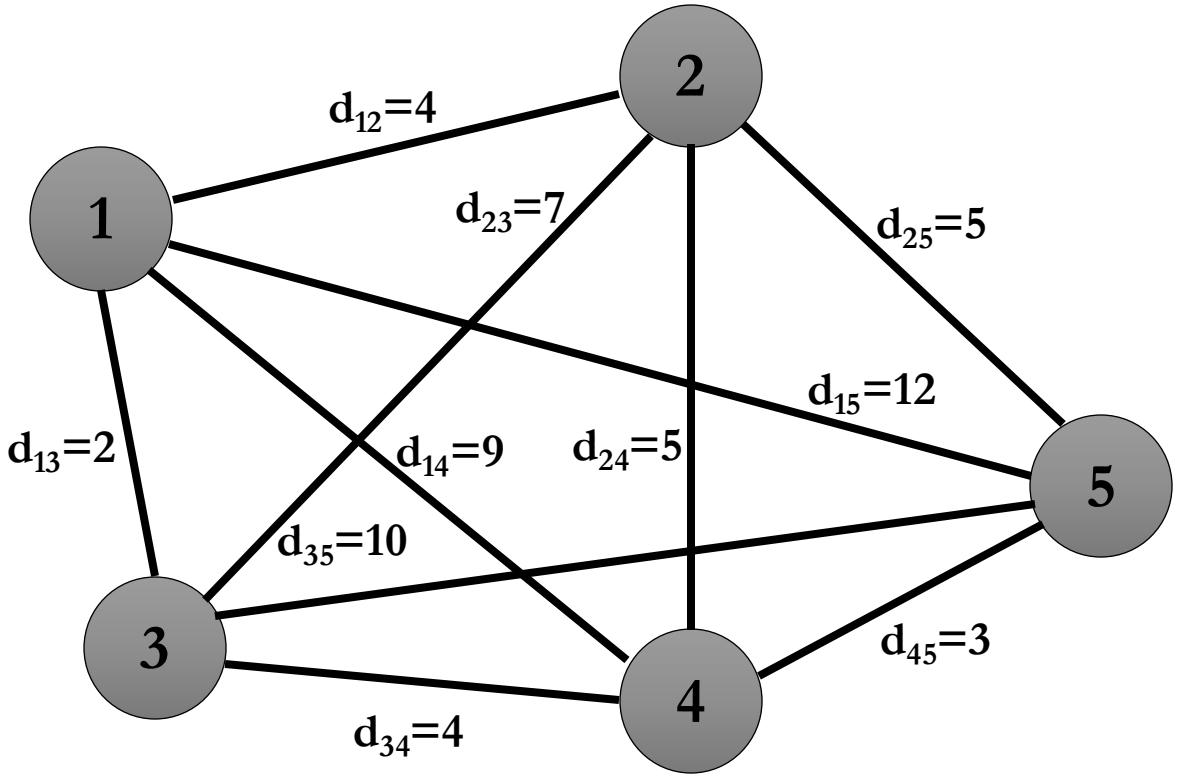
ACO APPLICATIONS

- Travelling Salesman Problem (TSP),
- Cell Assignment in PCS Networks (CA),
- Timetabling Problems.

ACO Applications

Travelling Salesman Problem

ACO FOR TSP - EXAMPLE



ACO FOR TSP - EXAMPLE

- A small artificial pheromone value is placed on all the edges,
- ***M*** ants are placed on the graph divided among all the nodes,
- ***M*** (number of ants) roundtrips are created in ***n*** (number of cities) steps.

ACO ALGORITHM OVERVIEW

- Initialize the pheromone trails,
- While termination condition not met
  - Construct Ant Solutions,
  - Apply Local Search *(Optional)*,
  - Update Pheromones,
- end

ACO FOR TSP - EXAMPLE

- The Construct Ant Solutions step:

- For each ant:
  - A new node is randomly chosen according to the selection probability,
  - This node is added to the ants' ***tabu list*** (so as not to be re-selected because every city is visited once).
  - After the completion of the roundtrips, the tabu lists are emptied,

ACO ALGORITHM OVERVIEW

- Initialize the pheromone trails,
- While termination condition not met
  - ConstructAntSolutions,
  - ApplyLocalSearch *(Optional),*
  - UpdatePheromones,
- end

ACO FOR TSP - EXAMPLE

- Usually, the online step-by-step update rule is ***not adopted***,
- The ApplyLocalSearch step:
  - These are optional actions referred to as ***daemon actions***,
  - One action is to apply a local search method in order to improve the constructed solution,
  - Another action is to add extra pheromone (***delayed update***) based on the collection of some global information.

ACO ALGORITHM OVERVIEW

- Initialize the pheromone trails,
- While termination condition not met
  - ConstructAntSolutions,
  - ApplyLocalSearch *(Optional)*,
  - UpdatePheromones,
- end

ACO FOR TSP - EXAMPLE

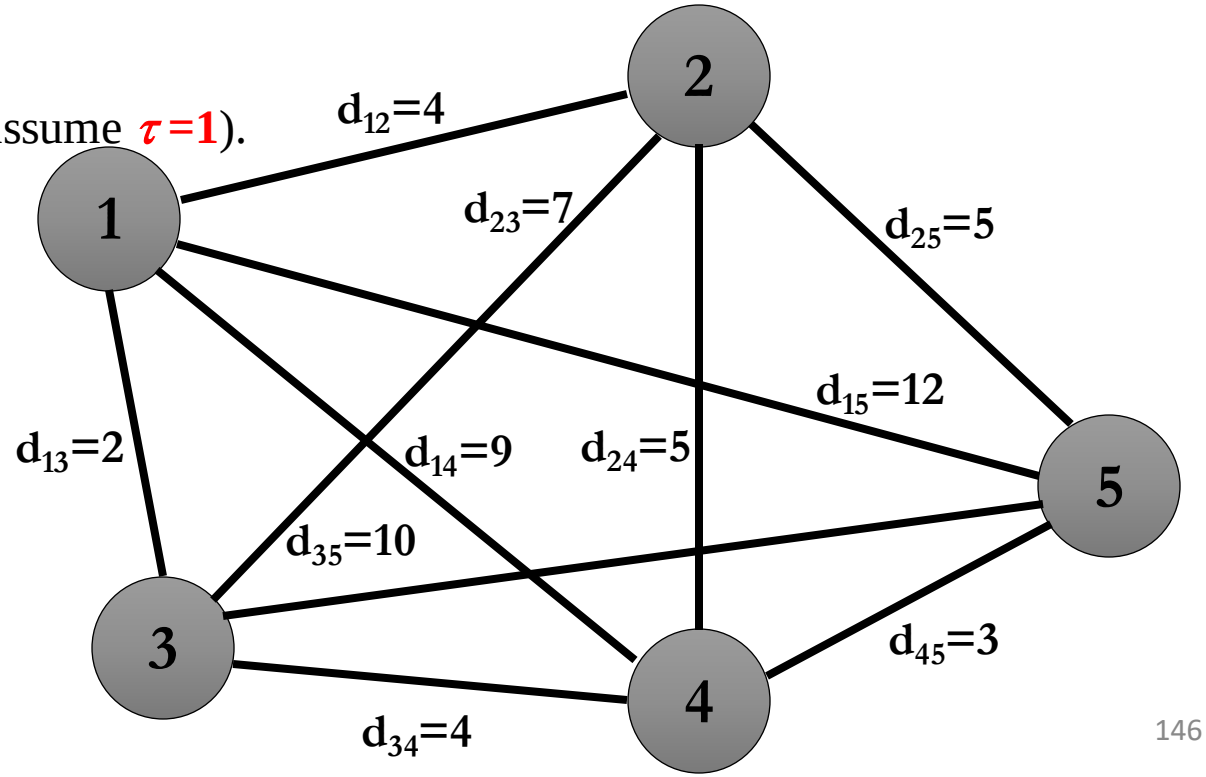
- The UpdatePheromones step:
  - Apply pheromone evaporation,
  - Update the pheromone trails using a chosen set of good solutions, different approaches use (***Ant cycle model***):
    - All the solutions found in the current iteration,
    - The best solution found in the current iteration,
    - The best solution found so far,

ACO FOR TSP - EXAMPLE

- A simple iteration:
  - Assume that an ant ***k*** was placed at node 1,
  - The neighbours of node 1 are :

$$N_1 = \{2, 3, 4, 5\}$$

  - Initially, all the edges have the same pheromone value (assume $\tau = 1$).



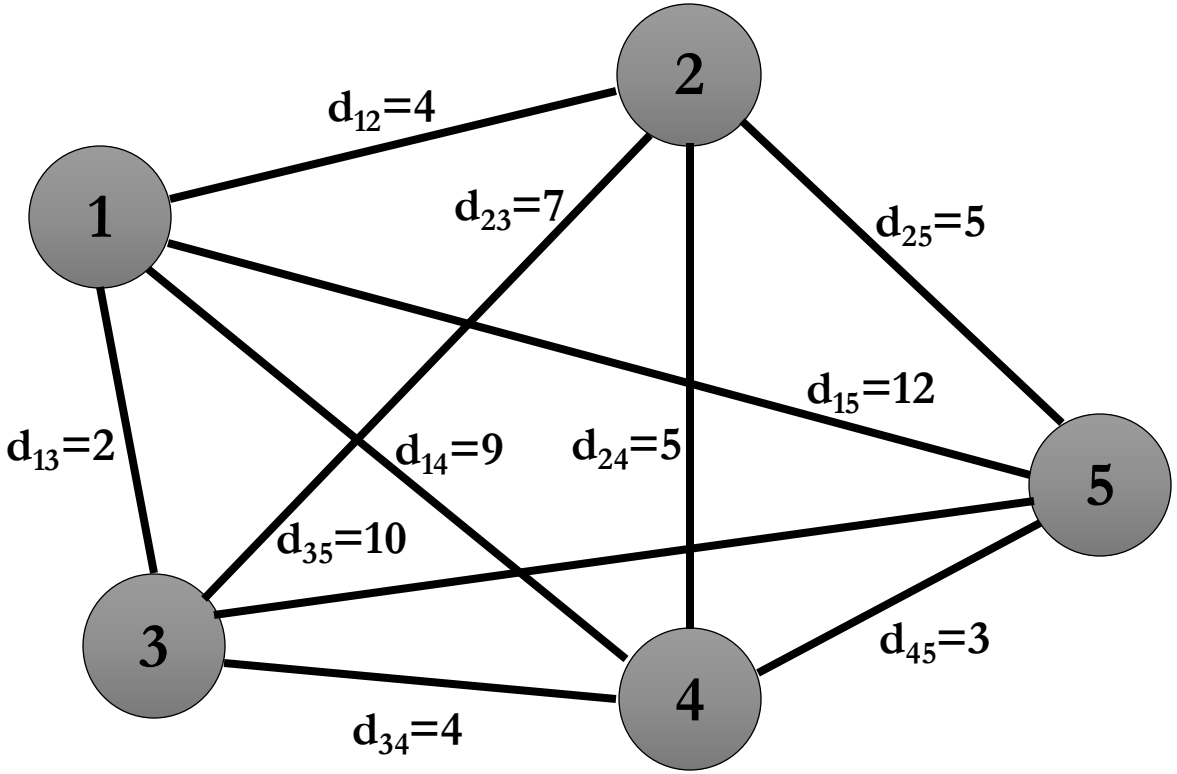
ACO FOR TSP - EXAMPLE

- Assuming α and β are both equal to 1,
- Then:

$$\sum_{n \in N_1} \tau_{1n}^{\alpha} / d_{1n}^{\beta} = \frac{1}{4} + \frac{1}{2} + \frac{1}{9} + \frac{1}{12} \cong 0.95$$

$$p_{12}^{k} = \frac{0.25}{0.95} \cong 0.26, p_{13}^{k} = \frac{0.5}{0.95} \cong 0.53$$

$$p_{14}^{k} = \frac{0.11}{0.95} \cong 0.12, p_{15}^{k} = \frac{0.08}{0.95} \cong 0.08$$



ACO FOR TSP - EXAMPLE

- Node 3 has the highest probability of being selected,

- If node 3 gets selected, ant ***k*** moves to that node and continues with the next iteration where:

$$N_3 = \{2, 4, 5\}$$

ACO FOR TSP - EXAMPLE

- Hence:

$$\sum_{n \in N_3} \tau_{1n}^{\alpha} / d_{1n}^{\beta} = \frac{1}{7} + \frac{1}{4} + \frac{1}{10} \cong 0.49$$

$$p_{32}^{k} = \frac{0.14}{0.49} \cong 0.29, p_{34}^{k} = \frac{0.25}{0.49} \cong 0.51$$

$$p_{35}^{k} = \frac{0.1}{0.49} \cong 0.20$$

- Another node is selected, and so on …

ACO FOR TSP - EXAMPLE

- If ant *k* completes the tour:

$$Tour = \{1, 3, 2, 4, 5\}$$

$$Cost = 29$$

- First, all the pheromone trails get evaporated,

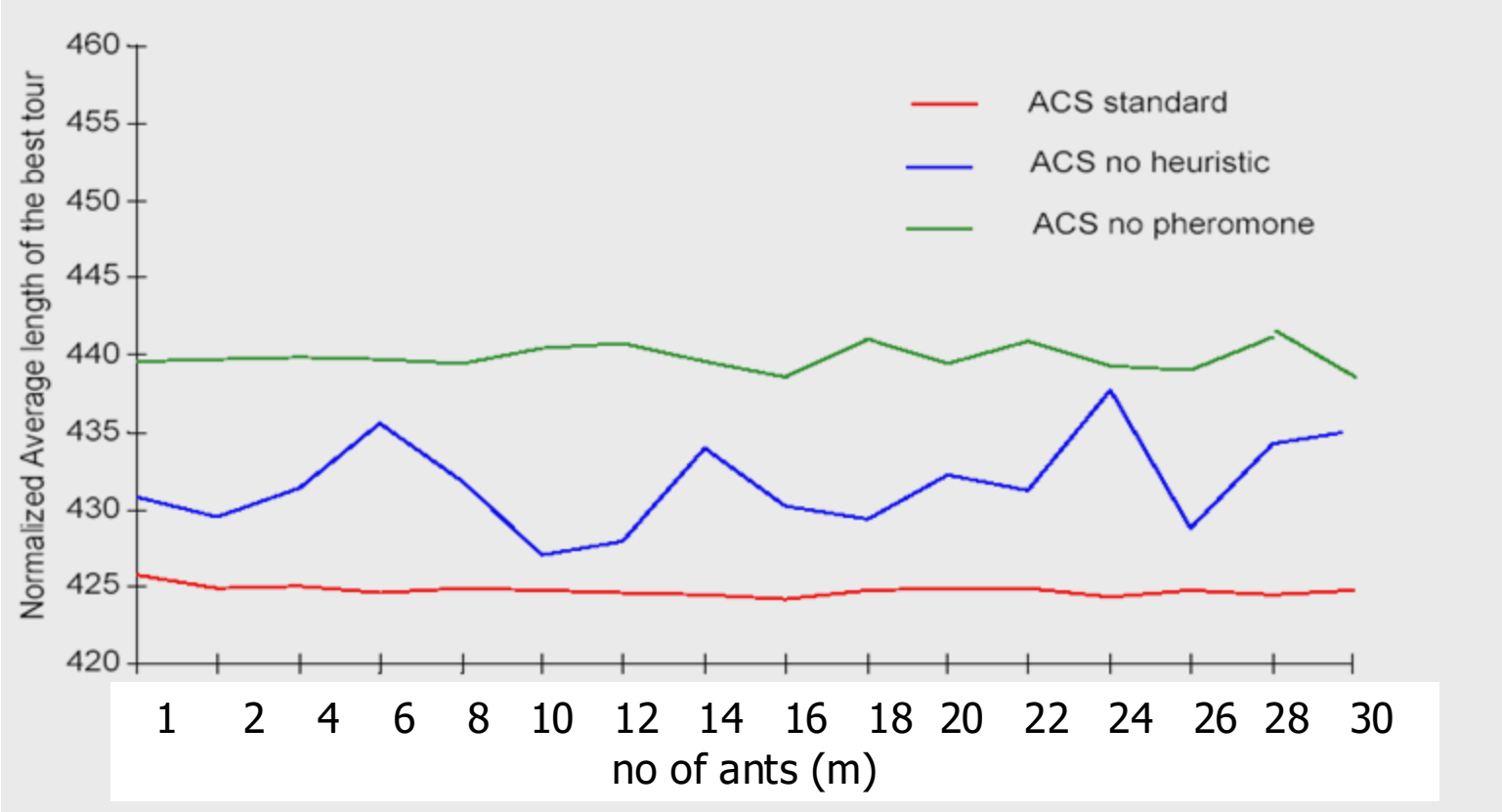
- And then, if this ant is selected to update the pheromone trails, it enforces the edges

$$\{1, 3\}, \{3, 2\}, \{2, 4\},$$
$$\{4, 5\}, \{5, 1\}$$

with the value Q/29.

PHEROMONE AND THE HEURISTIC FUNCTION

The importance of the pheromone and the heuristic function [8]



Comparison between ACS standard, ACS with no heuristic, and ACS in which ants neither sense nor deposit pheromone. Problem: Oliver30. Averaged over 30 trials, 10,000/*m* iterations per trial.

M. Dorigo and L. M. Gambardella."Ant Colony System: a cooperative learning approach to the traveling salesman problem". IEEE Trans. Evolutionary Computation, vol. 1, no. 1, 1997.

PHEROMONE AND THE HEURISTIC FUNCTION

- The results show that not using pheromone deteriorates the performance,
- ACS without heuristics performs better than ACS without pheromone. This may be due that ACS with pheromone and no heuristics is still guided by the global update rule (reflecting the importance of the solution).
- ACS without pheromone reduces to a stochastic multi-greedy algorithm
- ACS with both is better confirming the role of cooperation.

M. Dorigo and L. M. Gambardella.”Ant Colony System: a cooperative learning approach to the traveling salesman problem”. IEEE Trans. Evolutionary Computation, vol. 1, no. 1, 1997.

COMPARISON RESULTS OF ACO SYSTEMS

Number of ants = n

Number of iterations = 10000

Benchmark	Optimal	AS	MMAS	ACS
eil51	**426**	437.3	**427.6**	428.1
kroa100	**21282**	22471.4	**21320.3**	21420.0
d198	**15780**	16702.1	**15972.5**	16054.0

T. Stutzle and H. H. Hoos."*MAX-MIN* Ant System". *Future Generation Comput. Syst.*, vol. 16, no 8, 2000.

COMPARISON RESULTS: ACO AND GENETIC ALGORITHMS

- Comparison of ACS with the genetic algorithm (GA), evolutionary programming (EP), simulated annealing (SA), and the annealing-genetic algorithm (AG), a combination of genetic algorithm and simulated annealing.

- The best integer tour length, the best real tour length (in parentheses) and the number of tours required to find the best integer tour length (in square brackets).

- The best result for each problem is in boldface.

M. Dorigo and L. M. Gambardella.”Ant Colonies for the travelling salesman problem”, BioSystems, 1997.

COMPARISON RESULTS: ACO AND GENETIC ALGORITHMS

Problem name	ACS	GA	EP	SA	AG	Optimum
Oliver30	**420**	421	**420**	424	**420**	**420**
(30-city problem)	**(423.74)**	(N/A)	**(423.74)**	(N/A)	(N/A)	**(423.74)**
	[830]	[3,200]	[40,000]	[24,617]	[12,620]	
Eil50	**425**	428	426	443	436	**425**
(50-city problem)	(427.96)	(N/A)	**(427.86)**	(N/A)	(N/A)	(N/A)
	[1,830]	[25,000]	[100,000]	[68,512]	[28,111]	
Eil75	**535**	545	**542**	580	561	**535**
(75-city problem)	**(542.31)**	(N/A)	(549.18)	(N/A)	(N/A)	(N/A)
	[3,480]	[80,000]	[325,000]	[173,250]	[95,506]	
KroA100	**21,282**	21,761	N/A	N/A	N/A	**21,282**
(100-city problem)	**(21,285.44)**	(N/A)	(N/A)	(N/A)	(N/A)	(N/A)
	[4,820]	[103,000]	[N/A]	[N/A]	[N/A]	

It is clear that ACS and EP greatly outperform GA, SA, and AG.
N/A means not available in the literature

M. Dorigo and L. M. Gambardella."Ant Colonies for the travelling salesman problem", BioSystems, 1997.

ACS ON LARGER TSP PROBLEMS

ACS performance for some bigger geometric problems (over 15 trials). We report the integer length of the shortest tour found, the number of tours required to find it, the average integer length, the standard deviation, the optimal solution (for fl1577 we give, in square brackets, the known lower and upper bounds, given that the optimal solution is not known), and the relative error of ACS.

Problem name	ACS best integer length (1)	ACS number of tours generated to best	ACS average integer length	Standard deviation	Optimum (2)	Relative error $\frac{(1)-(2)}{(2)} * 100$
d198 (198-city problem)	15,888	585,000	16,054	71	15,780	0.68 %
pcb442 (442-city problem)	51,268	595,000	51,690	188	50,779	0.96 %
att532 (532-city problem)	28,147	830,658	28,523	275	27,686	1.67 %
rat783 (783-city problem)	9,015	991,276	9,066	28	8,806	2.37 %
fl1577 (1577-city problem)	22,977	942,000	23,163	116	[22,204 – 22,249]	3.27÷3.48 %

ACO Applications

Cell Assignment in PCS Networks

CELL ASSIGNMENT IN PCS NETWORKS (CA)

- The cell assignment problem is a challenging problem in PCS (Personal Communication Services) networks,
- In PCS networks, each cell has an antenna that is used to communicate with subscribers over some pre-assigned radio frequencies,
- Groups of cells are connected to a switch, through which the cells are then routed to the satellite networks.

S. J. Shyu, B. M. T. Lin and T. S. Hsiao.“An Ant Algorithm for Cell Assignment in PCS Netwroks”. IEEE International Conference on Networking, Sensing and Control, pp. 1081-1086, 2004.
J. R. L. Fournier and S. Pierre.“Assigning Cells to Switches in Mobile Networks using an Ant Colony Optimization Heuristic”.Computer Communications, vol. 28, pp. 65-73, 2005.

CELL ASSIGNMENT IN PCS NETWORKS (CA)

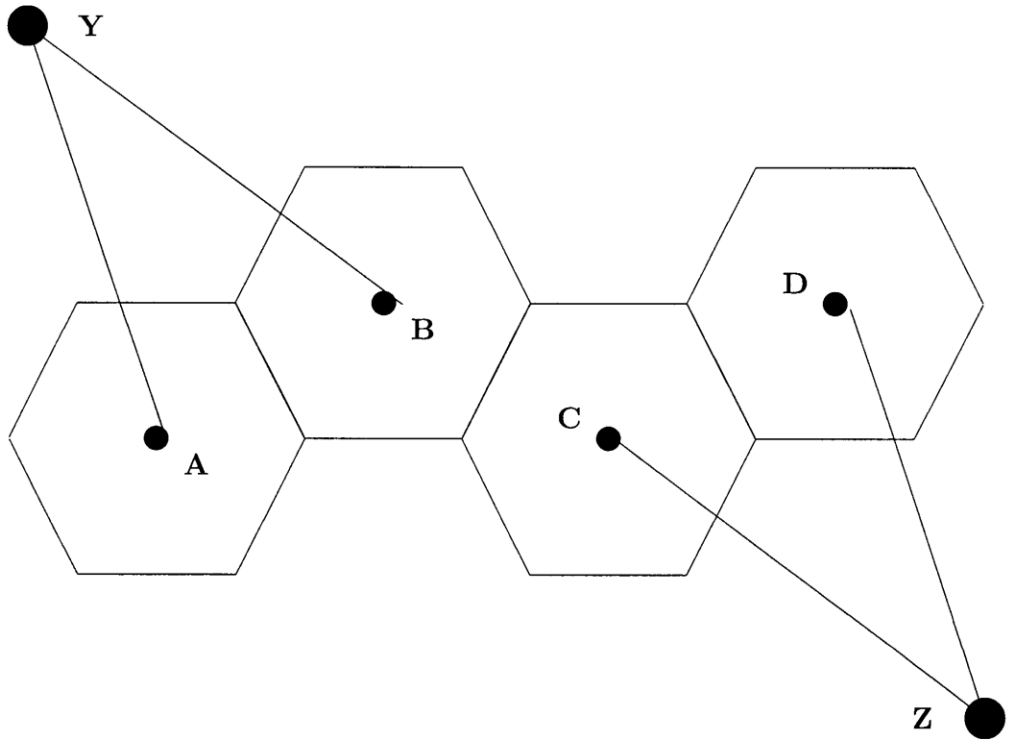
- Assignment of channels to cells
  - Assign frequency channels to cells so as to minimize interference (close cells shouldn't have close frequency ranges) and maximize utilization of channels (reuse)

- Given a set of n cells to be assigned to m switches, each switch k has a capacity M_k. The objective is to have an assignment that minimizes the cost (link between cells and switches and handoff cost)

ACO FOR CA

- In this problem, each ant has to make two different decisions per iteration:
  - Choosing the next cell to be assigned,
  - Choosing the switch that the chosen cell to be assigned to.
  - ➢A pheromone trail τ_{ij} is associated with every cell i and switch j possible assignment.
- Select max number of iterations
- Select number of ants
- Decide on transition rule
- Decide on pheromone update rules

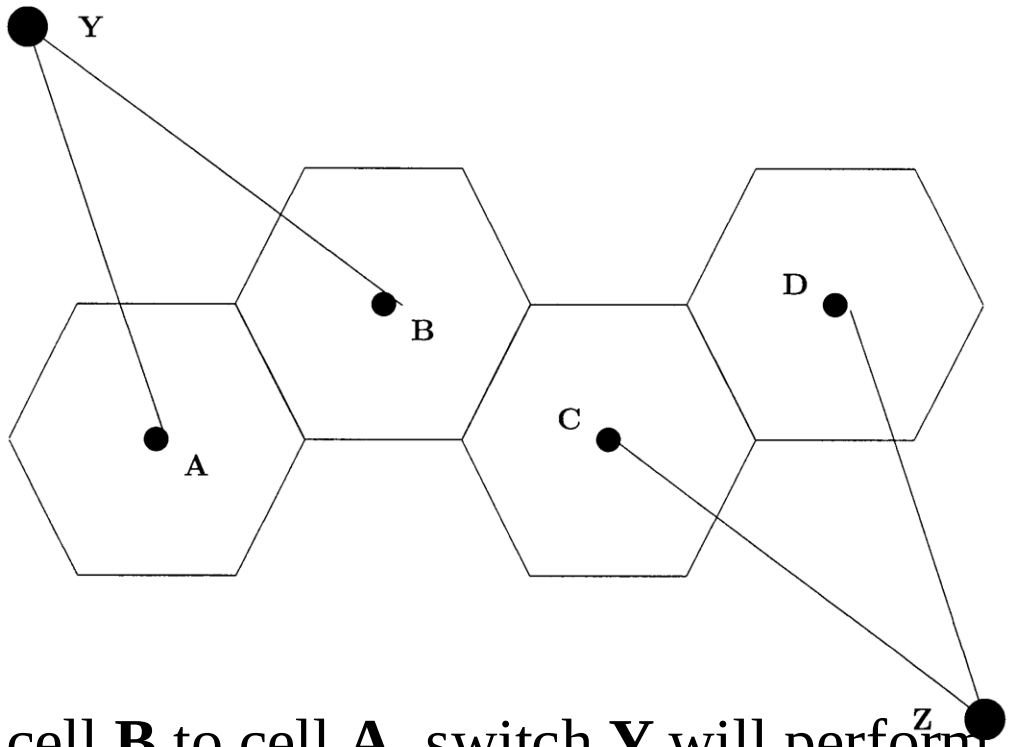
CELL ASSIGNMENT IN PCS NETWORKS (CA)



For instance:

- Assuming a situation where cells **A** and **B** are assigned to switch **Y** and cells **C** and **D** are assigned to switch **Z**,
- Suppose that a subscriber is currently talking to someone and this call is transmitted through cell **B** and switch **Y**,

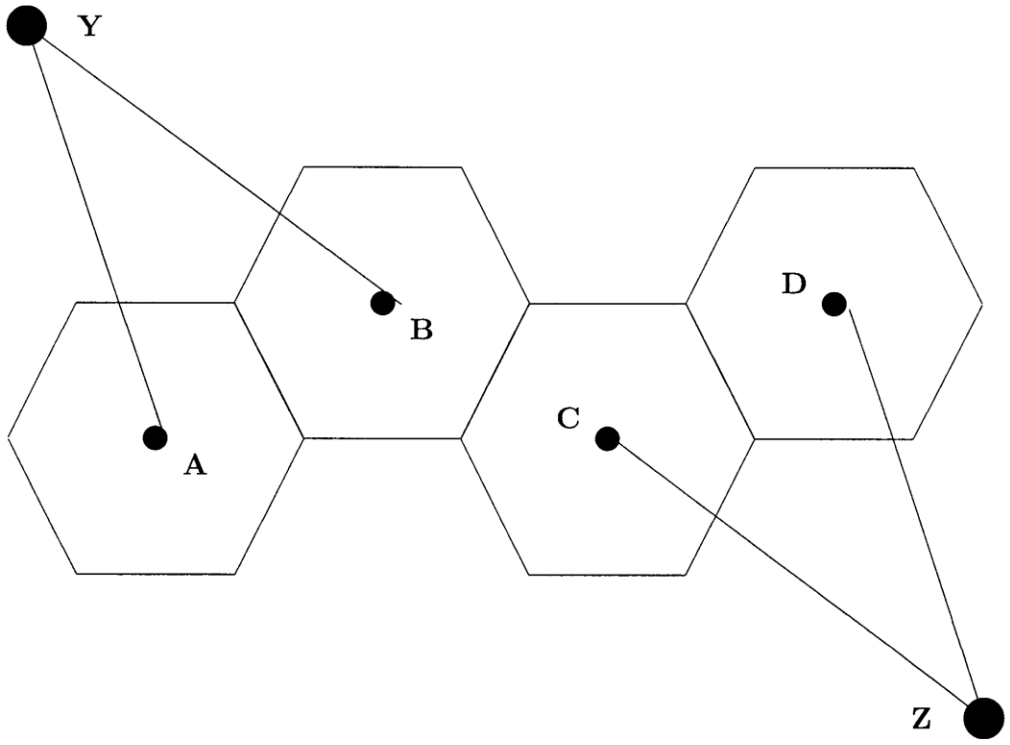
CELL ASSIGNMENT IN PCS NETWORKS (CA)



If the subscriber moves from cell **B** to cell **A,** switch **Y** will perform a *handoff* for the call:

- This call does not trigger any location update in the database that records the position of the subscriber,
- The handoff does not entail any network entity other than switch **Y**.

CELL ASSIGNMENT IN PCS NETWORKS (CA)



On the other hand, if the subscriber moves from cell **B** to cell **C**, then the *handoff* involves:

- The modification of the location of the subscriber in the database,
- The execution of a fairly complicated protocol between switches **Y** and **Z**.

CELL ASSIGNMENT IN PCS NETWORKS (CA)

- Costs associated with the assignment include:
  - Cable (link) costs between cells and switches,
  - Handoff costs between different cells:
    - Simple or no costs (involving only one switch),
    - Complex costs (involving two switches).
- Each switch has a certain capacity (in terms of calls volume) that should not be exceeded.
- The objective is to have an assignment that minimizes the cost.
- NP-hard problem

MATHEMATICAL FORMULATION

- Let c_{ik} be the cost of the link between cell i and switch k, $i=1,\ldots,n$; $k=1,\ldots,m$
- Let x_{ik} is 1 if cell i assigned to switch k, 0 otherwise;
- Let y_{ij} = 1 if cell i and cell j are assigned to the same switch.
- Let M_k be the capacity of switch k
- Let h_{ij} is handoff cost between cell i and j when they are assigned to different switches;
- Let d_i is the number of calls allowed for cell i

MATHEMATICAL FORMULATION

- The objective is

Minimize $$f = \sum_{i=1}^{n}\sum_{k=1}^{m} c_{ik}x_{ik} + \sum_{i=1}^{n}\sum_{j=1,i\neq j}^{n} h_{ij}(1-y_{ij})$$

subject to $$\sum_{i=1}^{n} d_i x_{ik} \leq M_k, k = 1,\ldots,m$$

ACO-ALGORITHM

- Select max no of iterations, itmax
- Select number of Ants, antmax
- While number of iterations <itmax
  - While current ant number< antmax
    - Initialize ant parameters (pheromone, update,..etc)
    - While (number of assigned cells is less than n)
      - Select next cell to be assigned using transition rule.
      - check capacity constraint of the selected switch,
      - Update problem data (capacity, cost,..)
    - End
    - Update Pheromone trail using update rules and evaporation rule
    - Evaluate quality of the Solution using the objective function
  - Retain best solution of all ants
  - Update pheromone trail based on best solution (if desired)
- Return best solution

ACO FOR CA

- Transition rules

- Let T be the set of switches not assigned yet

  - Simple ratio

$$p_{ij} = \begin{cases} \dfrac{\tau_{ij}^{\alpha}}{\sum_{j \in T} \tau_{ij}^{\alpha}} & if \quad j \in T \\ 0 & if \quad j \notin T \end{cases}$$

  - Using heuristics

$$p_{ij} = \begin{cases} \dfrac{[\tau_{ij}]^{\alpha}[\eta_{ij}]^{\beta}}{\sum_{l \in T}[\tau_{lk}]^{\alpha}[\eta_{lk}]^{\beta}} & \text{if} \quad j \in T \\ 0 & \text{otherwise} \end{cases}$$

ACO FOR CA

- Heuristic can be related to utilization

$$\eta_{ij} = (1/\text{cost of assigning cell i to switch j})$$

- Pheromone update and evaporation can be simple

$$\tau_{ij} = (1-\rho)\tau_{ij} + \Delta\tau_{ij}$$

- Or

$$\tau_{ij} = (1-\rho)\tau_{ij} + \rho\Delta\tau_{ij}$$

- Where $\Delta\tau_{ij}$ can be density, quantity (using local cost) or online delayed using solution quality

ACO FOR CA

- Or as in ACS, two updates

$$\tau_{ij} = (1-\rho_2)\tau_{ij} + \rho_2\tau_0$$

and

$$\tau_{ij} = (1-\rho_1)\tau_{ij} + \rho_1\Delta\tau_{ij}^{best}$$

REFERENCES

Text: Fundamentals of Computational Swarm Intelligence by A. P. Engelbrecht, Wiely, 2005. Chapter 1, 22, and 23.

Papers:

- ID. Couzin, J. Krause, R. James, GD. Ruxton, NR. Franks.”Collective Memory and Spatial Sorting in Animal Groups”. Journal of Theoretical Biology, 218, pp. 1–11, 2002.

- C. Grosan, A. Abraham and C. Monica.”Swarm Intelligence in Data Mining”. In Swarm Intelligence in Data Mining, A. Abraham, C. Grosan and V. Ramos (Eds), Springer, pp. 1–16, 2006.

- MM. Millonas .“Swarms, phase transitions, and collective intelligence”. In CG. Langton Ed., Artificial Life III, Addison Wesley, Reading, MA, 1994.

REFERENCES

- P. P. Grasse.“Recherches sur la biologie des termites champignonnistes (*Macrotermitina*)”. Ann. Sc. Nat., Zool. Biol. anim., 6, 97, 1944.

- P. P. Grasse.”La reconstruction du nid et les coordinations interindividu- elles chez *bellicositermes natalensis* et *cubitermes* sp. La theorie de la stig- mergie: essai d'interpretation du comportement des termites constructeurs”. Insectes Sociaux, 6, 41, 1959.

- J. L. Deneubourg, S. Aron, S. Goss and J. M. Pasteels.”The self- organizing exploratory pattern of the Argentine ant”. Journal of Insect Behaviour, 3, 159, 1990.

- S. Goss, S. Aron, J. L. Deneubourg and J. M. Pasteels.”Self-organized shortcuts in the Argentine ant”. Naturwissenschaften, 76, 579, 1989.

- M. Dorigo.”Optimization, Learning and Natural Algorithms”. Ph.D. thesis, DEI, Politecnico di Milano, Italy, pp. 140, 1992.

REFERENCES

- M. Dorigo, V. Maniezzo and A. Colorni.” Ant System: Optimization by a Colony of Cooperating Agents”. IEEE Trans. Syst., Man, and Cybern. Part B, vol. 26, no. 1, 1996.

- T. Stutzle and H. H. Hoos.”*MAX-MIN* Ant System”. *Future Generation Comput. Syst.*, vol. 16, no 8, 2000.

- M. Dorigo and L. M. Gambardella.”Ant Colony System: a cooperative learning approach to the traveling salesman problem”. IEEE Trans. Evolutionary Computation, vol. 1, no. 1, 1997.

- J. Bautista and J. Pereira. “Ant Algorithms for A Time and Space Constrained Assembly Line Balancing Problem”. European Journal of Operation Research, 177, pp. 2016-2032, 2002.

- S. J. Shyu, B. M. T. Lin and T. S. Hsiao.“An Ant Algorithm for Cell Assignment in PCS Netwroks”. IEEE International Conference on Networking, Sensing and Control, pp. 1081-1086, 2004.