Metaheuristics

Nicolas Durand

ENAC

2021

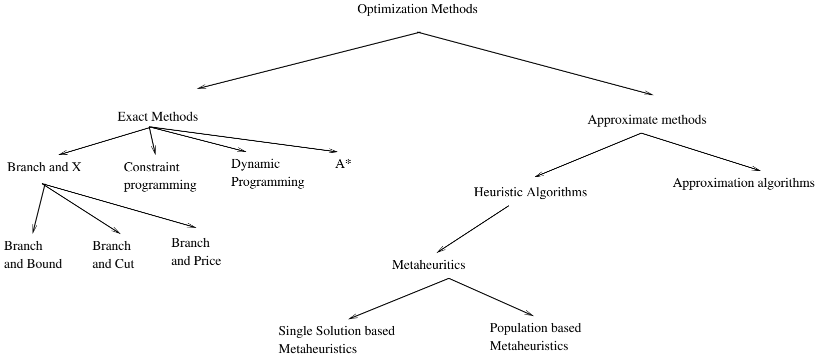
A Classification of Discrete Optimization Algorithms

- Deterministic : example Branch and Bound
- Probabilistic : example Monte Carlo Algorithms

Stochastic Optimization and Global Optimization

- Definition: a global optimization algorithm is an optimization algorithm that employs measures that prevent convergence to local optima and search for a global optimum.
- Stochastic Global Optimization algorithms discover good elements with higher probability than elements with bad characteristics.
- The success of optimization depends very much on the way the search is conducted.
- It also depends on the time the optimizer allowed to use.
- Hill climbing algorithms are no global optimization algorithms since they have no means of preventing getting stuck at local optima.

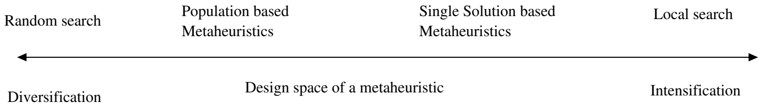
Optimization Methods



Definitions

- Heuristic: a part of an optimization algorithm that uses the information currently gathered by the algorithm to help to decide which solution candidate should be tested next. Heuristics are usually problem class dependent.
- Metaheuristic: a top-level general strategy which guides other heuristics to search for solving a general class of problems where the task is hard.

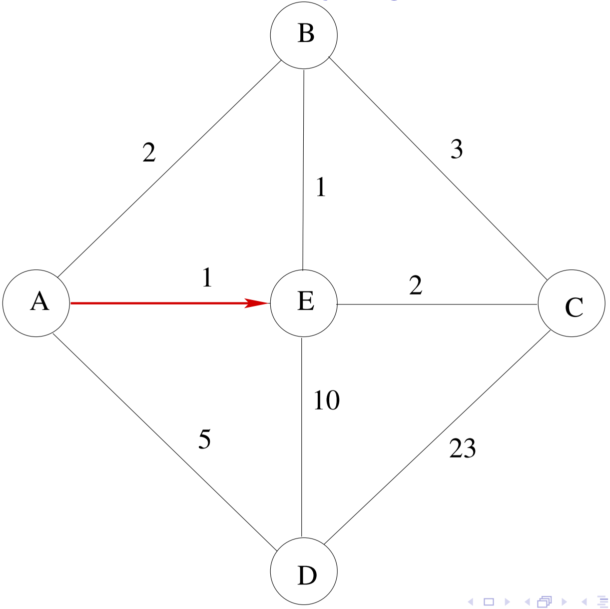
Single versus Population based Metaheuristics



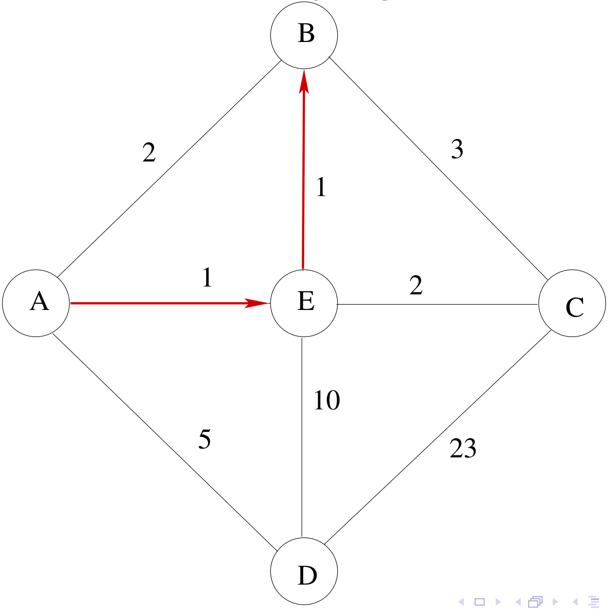
Iterative versus Greedy Algorithms

- Iterative algorithm start with a complete solution and transforms it at each iteration
- Greedy algorithm start from empty solution and at each step a decision variable is assigned to build a complete solution.
- Most metaheuristics are iterative algorithms

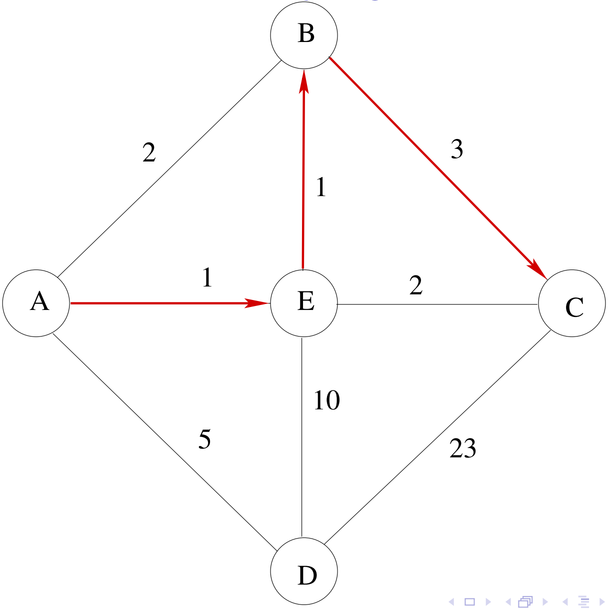
Example of Greedy Algorithm: TSP



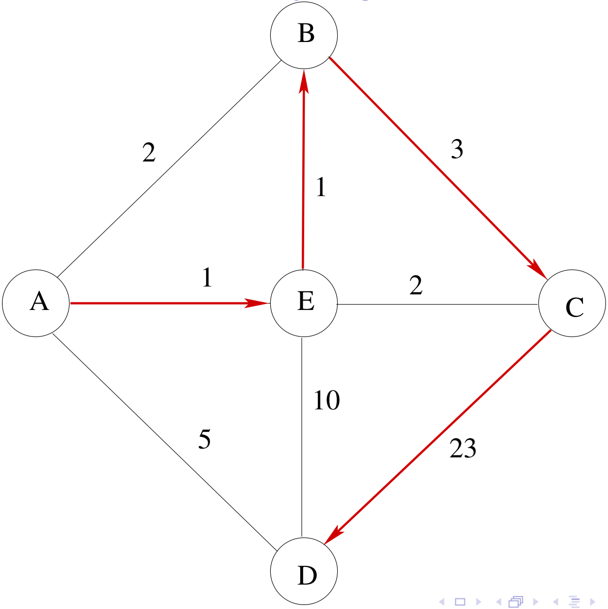
Example of Greedy Algorithm: TSP



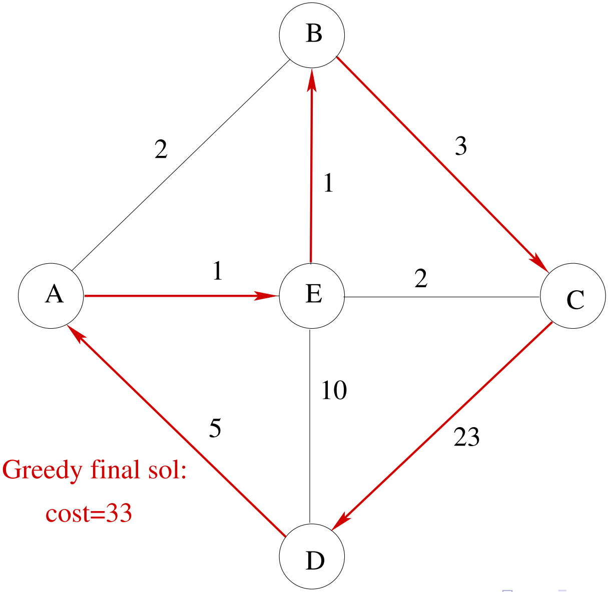
Example of Greedy Algorithm: TSP



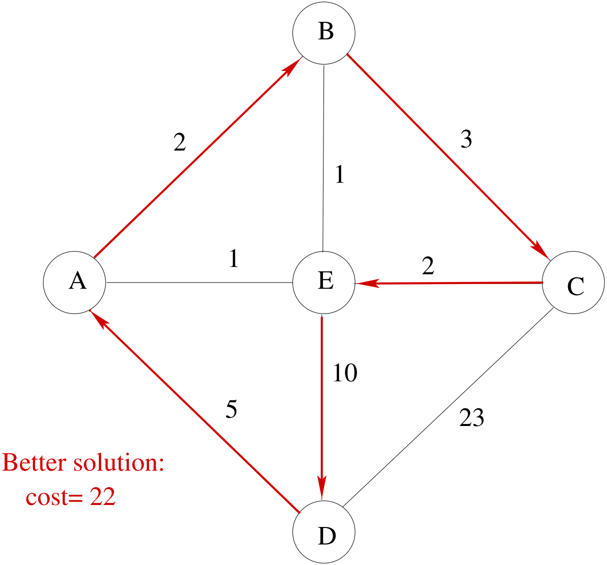
Example of Greedy Algorithm: TSP



Example of Greedy Algorithm: TSP



Example of Greedy Algorithm: TSP



Definitions

- Evaluations number: the number of solution candidates for which the set of objective functions F has been evaluated.
- Iterations: an iteration refers to one round in a loop of an algorithm. It is one repetition of a specific sequence of instruction inside an algorithm.
- Termination Criterion: decides the end of the optimum search process:
  - maximum computation time
  - total number of iterations t or individual evaluations
  - no improvement in the solution quality
  - required precision or good enough solution obtained

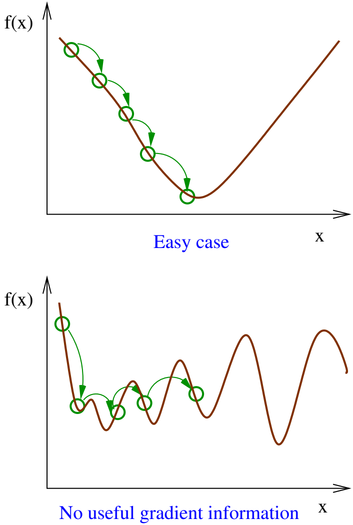
Definitions

- An optimization algorithm has converged if it keeps on producing solutions from a small subset of the problem space, possibly neglecting better solutions (in terms of fitness) in other areas.
- An optimization process has prematurely converged to a local optimum if it is no longer able to explore other parts of the search space than the currently examined area and there exists another region that contains a solution superior to the currently exploited one.

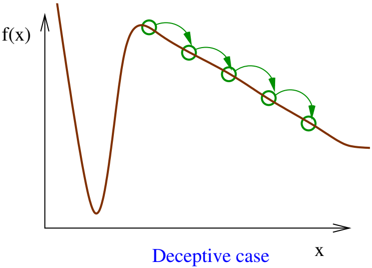
Definitions

- An objective function F is multimodal, if it has multiple local or global optima
- Exploration in terms of optimization means finding new points in areas of the search space which have not been investigated yet.
- Exploitation means trying to improve the currently known solution(s) by performing small changes which lead to new individuals very close to them.

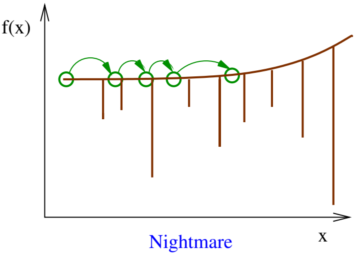
Ruggedness


f(x)
x
Easy case
f(x)
x
Multiple local optima
f(x)
x
No useful gradient information
f(x)
x
Needle In A Haystack

Ruggedness



Deceptive case



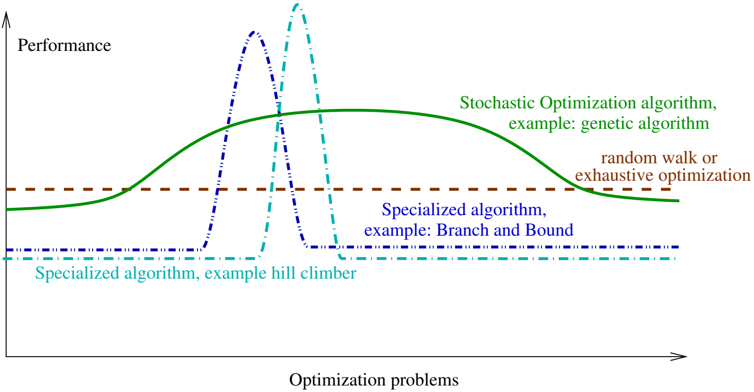
Nightmare

Causality

- The principle of strong causality (locality) (proposed by Rechenberg) states that small changes in an object lead to small changes in its behavior.
- In general, the lower the causality of an optimization problem, the more rugged its fitness landscape will be, and hence, the worse will optimizers perform.

No free lunch Theorem

- it is impossible for any optimization algorithm to outperform all random walks or exhaustive enumerations on all possible problems. For every problem where a given method leads to good results, we can construct a problem where the same method has exactly the opposite effect.



Stochastic Optimization Algorithms

- Simulated Annealing
- Tabu Search
- Genetic Algorithms
- Evolutionary Algorithms
- Evolutionnary and Genetic Programming
- Ant Colony Optimization

Simulated Annealing

- 1953, Metropolis: algorithm for the simulation of the evolution of a solid to thermal equilibrium
- 30 years later, Kirkpatrick 82 and Cerny 85: analogy with minimizing the cost function of a combinatorial optimization problem
- Given the current state of a solid by the position of its particles, operate a small displacement of a random particule
- If the difference in Energy, ΔE between the current and perturbed state is negative then the process is continued
- If $\Delta E \geq 0$, then the probability of acceptance of the new state is $exp(\frac{-\Delta E}{k_B T})$
- The system eventually evolves to a thermical equilibrium

Simulated Annealing

- The cost function f takes the role of energy
- The control parameter c replaces the temperature.
- The SA algorithm can be viewed as a sequence of Metropolis Algorithms evaluated at a sequence of decreasing values of c
- INITIALIZE, $M = 0$
    - PERTURB (Config i $\Rightarrow$ Config j)
    - if $\Delta C_{ij} \leq 0$ then ACCEPT
    - else if $exp(-\frac{\Delta C_{ij}}{c}) > random[0, 1]$ then ACCEPT
    - if ACCEPT then UPDATE (conf j)
- until equilibrium is approached sufficiently closely
- $C_{M+1} = T(C_M)$, $M = M + 1$

Tabu Search

- 1986 and 1989, Fred W. Glover: Local search starting from a point and check neighbors to improve the current solution.
- Different from hill climbing.
- Worsening moves can be accepted if no improving move is available
- Prohibitions are introduced to discourage the search from coming back to previously visited solutions.

Tabu Search: ingredients

- Definition of local moves
- Tabu list: short-term set of solutions that have been recently visited
- Different types of memories:
- Short-term memory: recently visited states
- Mid-term memory: Intensification Rules to bias the search toward promising areas
- Long-Term memory: Diversification Rules that drive the search into new regions

Tabu Search: Pseudocode

1. $x_{best} \leftarrow x_0$
2. $x_{prov} \leftarrow x_0$
3. $tabulist \leftarrow []$
4. While not StoppingCondition(x_{best})
5. $\quad Curr_{neighborhood} \leftarrow get_{neighbors}(x_{prov})$
6. $\quad$ for $x \in Curr_{neighborhood}$
7. $\qquad$ if $x \notin tabulist$ and $f(x) > f(x_{xprov})$ then $x_{prov} \leftarrow x$
8. $\quad$ if $f(x_{prov}) > f(x_b)$ then $x_b \leftarrow x_{prov}$
9. $\quad tabulist \leftarrow tabulist :: x_{prov}$
10. $\quad$ if $tabulist.size > maxsize$ then
11. $\qquad tabulist \leftarrow tabulist.removefirst()$
12. return x_b

Evolutionnary Computation

- Genetic Algorithms. Holland (1975). Goldberg (1989)
- Evolution Startegies. Rechenberg et Schwefel (1965). Schwefel (1981 - 1995)
- Evolutionnary Programming. L.J. Fogel (1966). D.B. Fogel (1991 - 1995)
- Genetic programming. Koza (1992 - 1994)

Canonical genetic algorithm

- A population of n chromosoms or elements $X_i = (b_1, ..b_n) \in E = \{0, 1\}^n$
- Each population elements codes a point of the research set
- An initial population is randomly chosen
- The Objective function or Fitness function defines which elements will be selected and reproduced for the next generation (the best ones).
- Crossover and Mutation Operators are applied to a part of the population
- An ending criteria decides to stop the algorithm

Roulette wheel selection

- Each chromosom X_i is reproduced with a probability

$$\frac{F(X_i)}{\sum_{j=1}^{P} F(X_j)}$$

- Example :

$$\begin{aligned} F(X_1) &= 50 \qquad & P &\rightarrow 50\% \\ F(X_2) &= 15 \qquad & P &\rightarrow 15\% \\ F(X_3) &= 25 \qquad & P &\rightarrow 25\% \\ F(X_4) &= 10 \qquad & P &\rightarrow 10\% \end{aligned}$$

Crossover: bit exchange between 2 parents ($p_c \approx 0.25 - 0.75$)

- 1 point crossover (Holland - 75): random position: $l \in [1, N]$

$$\frac{b_1, b_2, .., b_N}{c_1, c_2, .., c_N} \quad \rightarrow \quad \frac{b_1, .., b_l, c_{l+1}, .., c_N}{c_1, .., c_l, b_{l+1}, .., b_N}$$

- 2 points crossover (Dejong - 75): 2 random positions: $(l, m) \in [1, N]^2$

$$\frac{b_1, b_2, .., b_N}{c_1, c_2, .., c_N} \quad \rightarrow \quad \frac{b_1, .., b_l, c_{l+1}, ..c_m, b_{m+1}, .., b_N}{c_1, .., c_l, b_{l+1}, ..b_m, c_{m+1}, .., c_N}$$

- n points crossover (Syswerda - 89): each bit is randomly chosen from both parents

Mutation: $p_m \approx 0.001 - 0.1$

- For one population element, choose a random position $l \in [1, N]$ with a uniform probability:

$$b_1, ..b_l, .., b_N \quad \rightarrow \quad b_1, ..\bar{b}_l, .., b_N$$

- with:

$$\begin{aligned} b_l = 0 \quad &\rightarrow \quad \bar{b}_l = 1 \\ b_l = 1 \quad &\rightarrow \quad \bar{b}_l = 0 \end{aligned}$$

Example: $f(x) = 4\,x\,(1 - x)$

sequ	10111010	11011110	00011010	01101100
val	0.7265625	0.8671875	0.1015625	0.4218750
f(x)	0.794678	0.460693	0.364990	0.975586
%/tot	$\frac{0.79}{2.59} = 0.31$	$\frac{0.46}{2.59} = 0.18$	$\frac{0.36}{2.59} = 0.14$	$\frac{0.97}{2.59} = 0.37$
cumul	0.31	0.49	0.63	1.00
reprod	11011110	10111010	01101100	01101100

- For reproduction, randomly chose 4 numbers between 0 and 1: 0.47, 0.18, 0.89, $0, 75$

Crossover: $p_c = 0.5$

- 2 population elements chosen randomly: 1 and 3
- Position chosen: 5
- After crossover:

$$\frac{11011 - 110}{01101 - 100} \rightarrow \frac{11011 - 100}{01101 - 110}$$

- New population after crossover:

sequ	01101110	10111010	11011100	01101100
val	0.4296875	0.7265625	0.8593750	0.4218750
f(x)	0.980225	0.794678	0.483398	0.975586

Evolution Strategies (ES)

- Historically $E = \Re^n$, (1+1):
- $x \in E$
- Mutation: $y = x + N(0, \sigma)$
- If $F(y) \geq F(x)$ then $x = y$
- Theoretical results on simple functions: The proportion π of successful mutations must be close to $\frac{1}{5}$. If $\pi > \frac{1}{5}$, increase σ, else decrease it.
- Other strategies:
  - $(1, 1)$: random walk
  - $(\mu + 1)$: eliminate the worse
  - $(1, \lambda)$: 1 parent, several children
  - $(1 + \lambda)$: idem, plus elitism

Evolutionnary Programming, Genetic Programming:

- Optimize a program, a fonction or a more complex structure
  - Example: find a function estimator knowing inputs and outputs
- The p parents are reproduced once
- Only one operator: mutation
- The mutation amplitude depends on the performance (small for the best, large for the less performant)
- Genetic Programming uses crossover and mutation
- Efficiency of the crossover contested

Theoretical Results exist but are useless in practice

- A general result of Global Convergence, in terms of weak convergence of probability measures. (Zhigljavsky 1990)
- Partial results based on a Markov chain modeling of EAs in the population space (Davis et Principe, 1991 - Nix et Vose, 1992 - Fogel 1992)
- A Global Strong Convergence Result for GAs based on Freidlin Wentzell's Stochastic Perturbations Theory (Cerf 1993)
- Global Convergence results with convergence rates for $(1+1)$ and $(1, \lambda)$ ES but only on convex functions

GAs Convergence (Cerf 1993)

- Freidlin Wentzell's Theory Application to stochastic perturbations of dynamic systems
- Process: Markov chain, irreductible aperiodical and homogeneous of finished space states
- Unperturbed System: no mutation, no crossover, caricatural selection
- Pertubation: proportional selection, mutation ans crossover (perturbations converge to 0 simultaneously)
- Then there exist a population size P^* (computable using F and the perturbations definitions) such that if $P > P^*$:

$$\forall x \in E^P lim_{t \to \infty} P(G_t \subset F^* / G_0 = x) = 1$$

$$P^* \leq \frac{a\, R + c\,(R-1)\,\Delta}{min(a,\, \frac{b}{2},\, c\,\delta)}$$

a and b caracterize the transition speeds of the irreductibuility kernel of the mutation and crossover

operators

Bibliography

- Ancestors
  - L.J. Fogel and A.J. Owens and M.J. Walsh, *Artificial Intelligence Through Simulated Evolution*, John Wiley & Sons, 1966
  - J. Holland, *Adaptation in natural and artificial systems* University of Michigan Press, Ann Arbor, 1975
  - H.P. Schwefel, *Numerical Optimization of Computer Models*, John Wiley & Sons, 1981 (Nelle édition 1995).

Bibliography

- Moderns
  - D.E. Goldberg, *Genetic algorithms in search, optimization and machine learning* Addison Wesley, 1989
  - Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, Springer Verlag, 1992
  - J. Koza, *Genetic Programming, I & II*, MIT Press, 1992 & 1994
  - D. B. Fogel, *Evolutionary Computation, Toward a New Philosophy of Machine Intelligence*, IEEE Press, 1995.
  - T. Back, *Evolutionary Algorithms in theory and practice*, New-York : Oxford University Press, 1995.

GAs ⇒ making them work

1. Data coding
2. Dealing with constraints
3. Selection modes
4. Scaling
5. Elitism
6. Sharing
7. Example of test functions
8. Parallelization

Data Coding

- Problem: With bit string coding, data which are close can have very different encodings.
- Example: 01111111 and 10000000 represent consecutive numbers
- Solution: Gray coding
- Or: adapt the coding to the problem
  - Float coding
  - Integer coding
  - Mixed coding

Adapted operators

- Arithmetic crossover:

$$\begin{aligned} C_i &= \alpha A_i + (1-\alpha) B_i \\ D_i &= (1-\alpha) A_i + \alpha B_i \end{aligned}$$

with $\alpha \in [-0.5, 1.5]$ randomly chosen for each variable or equal for the whole crossover process

- Mutation: add a random noise

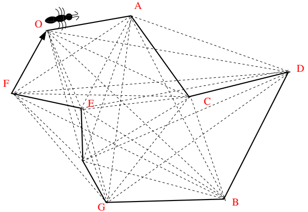
$$C_i = A_i + N(0, \sigma)$$

- Operators depend on coding

Dealing with constraints

- If possible: only create data that respect constraint
- Example: interval constraint
  - create data inside the interval
  - after crossover and mutation, check that the data created remain inside the interval.
- $Max(\Pi sin(x_i)) \,/\, (x_i - x_j)^2 > d^2$
  - Correct each new datum using a local optimization algorithm: difficult and time consuming
  - Include the constraint respect in the fitness function

Example Traveling Salesman Problem



Data coding:

- Integer list: the list must not contain twice the same number
- Use the following property: any bijection can be decomposed in a sequence of transpositions
  - advantage: no constraint to check
  - drawback: how to define effective crossover and mutation operators?

TSP: Dantzig-Fulkerson-Johnson formulation

Label the cities with the numbers $1 \dots n$ and define:

$$x_{ij} = \begin{cases} 1 \text{ if the path goes from city } i \text{ to city } j \\ 0 \text{ otherwise} \end{cases}$$

Let c_{ij} be the distance from city i to city j. Then TSP can be written as the following integer linear programming problem:

$$\begin{aligned}
\min \sum_{i=1}^{n} \sum_{j \neq i, j=1}^{n} c_{ij} x_{ij} & \\
0 \leq x_{ij} \leq 1 & \qquad i, j = 1, \dots, n; \\
\sum_{i=1, i \neq j}^{n} x_{ij} = 1 & \qquad j = 1, \dots, n; \\
\sum_{j=1, j \neq i}^{n} x_{ij} = 1 & \qquad i = 1, \dots, n; \\
\sum_{i \in Q} \sum_{j \in Q} x_{ij} \leq |Q| - 1 & \qquad \forall Q \subsetneq \{1, \dots, n\}, |Q| \geq 2
\end{aligned}$$

Explain the role of every constraint.

How many constraints does the problem have?

TSP: Miller-Tucker-Zemlin formulation

Label the cities with the numbers $1 \ldots n$ and define:

$$x_{ij} = \begin{cases} 1 \text{ if the path goes from city } i \text{ to city } j \\ 0 \text{ otherwise} \end{cases}$$

Let c_{ij} be the distance from city i to city j.
For $i = 1 \ldots n$, let u_i be a dummy variable
Then TSP can be written as the following integer linear programming problem:

$$\begin{aligned}
\min \sum_{i=1}^{n} \sum_{j \neq i, j=1}^{n} & c_{ij} x_{ij} & \\
x_{ij} &\in \{0, 1\} & i, j = 1, \ldots, n; \\
u_i &\in \mathbf{Z} & i = 1, \ldots, n; \\
\sum_{i=1, i \neq j}^{n} x_{ij} &= 1 & j = 1, \ldots, n; \\
\sum_{j=1, j \neq i}^{n} x_{ij} &= 1 & i = 1, \ldots, n; \\
u_i - u_j + n x_{ij} &\leq n - 1 & 2 \leq i \neq j \leq n; \\
0 \leq u_i &\leq n - 1 & 2 \leq i \leq n.
\end{aligned}$$

Prove that the last two constraints are necessary and sufficient to guarantee a single loop.
How many constraints do we have?
How can we convert u_i into binary constraints?

Selection modes

- Roulette wheel selection:
  when the sizes of the populations are too small, it can introduce a biasis, because the law of large numbers is not satisfied
- $F(X_1) = 22$, $F(X_2) = 11$, $F(X_3) = 11$, $F(X_4) = 6$
  selection: X_1, X_1, X_2, X_3, *but a good element can be lost*
- Stochastic remainder without replacement:
  Each element is reproduced $\frac{F(x_i)}{\bar{F}}$ *times*
  The roulette wheel selection is applied on the rest:
  $F(X_i) - \frac{F(x_i)}{\bar{F}}$

Stochastic remainder without replacement

i	1	2	3	4	5	6	7	8	9	10
élt	1	2	3	4	5	6	7	8	9	10
fit	0.6	0.9	0.5	0.2	0.2	0.4	0.1	0.2	0.6	0.1

$moy = 0.38$

i	1	2	3	4	5	6	7	8	9	10
res	0.22	0.14	0.12	0.2	0.2	0.02	0.1	0.2	0.22	0.1
élt	1	2	2	3	6	9				

i	1	2	3	4	5	6	7	8	9	10
élt	1	2	2	3	6	9	1	4	8	5

Scaling

- Goal: increase or decrease the selection speed
- Method: scale the function values
  - Sigma truncation scaling

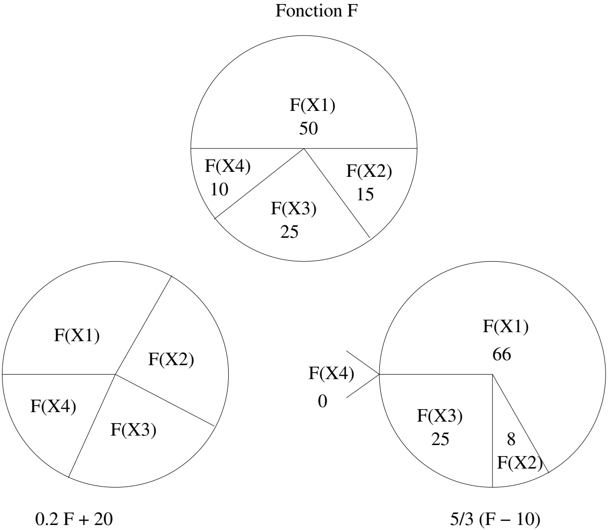
$$F_{scal}(X) = F(X) - (moy - C\dot{\sigma})$$
$$Si\ F_{scal}(X) < 0,\ F_{scal}(X) = 0$$

  - Power law scaling :

$$F_{scal}(X) \quad = \quad F(X)^{l(gen)}$$

$l(gen)$ increases with time

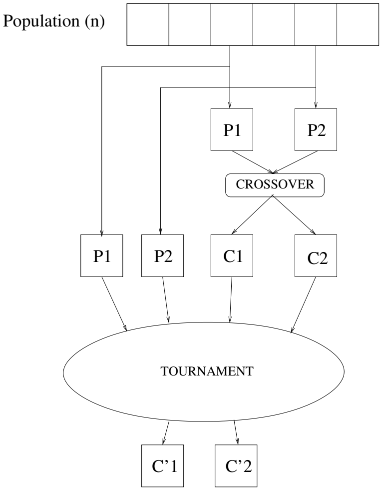
Selection pressure



Elitism

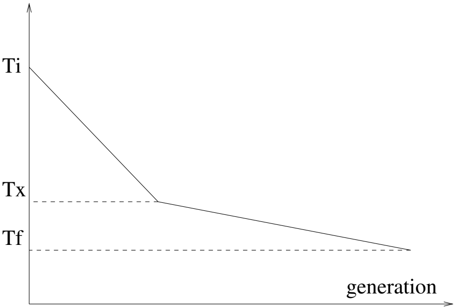
- Goal: do not lose the best elements during the crossover and mutation process
- How: protect the best elements from crossover and mutation
- Drawback: if the size of the population is small, not enough elements are left for crossover and mutation

Elitism and crossover



Crossover, mutation and simulated annealing

- Goal: improve the fitness of the elements after the crossover and the mutation processes
- How: a tournament between parents and children is done
  - if the child is better, than it replaces the parent
  - if not, it replaces the parent with a probability $|F_P - F_E| < e^{-kT}$
- T varies like in a Simulated Annealing process



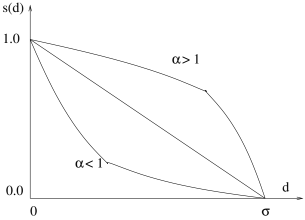
Sharing

- Goal: avoid over concentrations of elements in some part of the space research
- Penalize the fitness function of an individual according to the number of its neighbours

$$F_s(X_i) = \frac{F(X_i)}{\sum s[d(X_i, X_j)]}$$

$$s(d) = 1 - (\frac{d}{\sigma})^\alpha \text{ if } d < \sigma$$

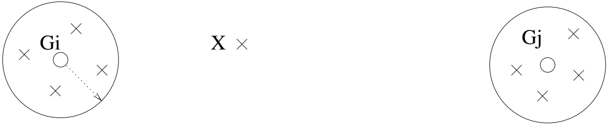
$$s(d) = 0 \text{ if } d \geq \sigma$$



- Drawback: Complexity in n^2 and choice of σ

Sharing: clustering

- Goal: decrease the Sharing Complexity $n\,log(n)$
- How: create clusters by aggregating data



  - if $\forall i, d(X_n, X_{gi}) > d_{max}$ then create a new cluster, else aggregate the closest cluster
  - if $d(X_{gi}, X_{gj}) < d_{min}$ then aggregate clusters i and j
- Penalize proportinally to the number of cluster elements and inversly proportionnaly to the distance to the cluster barycenter
- Drawback:
  - not an exact method
  - Difficult to chose d_{min} and d_{max}

Sharing: clustering

$$F_{clus}(X_i) = \frac{F(X_i)}{T_i}$$

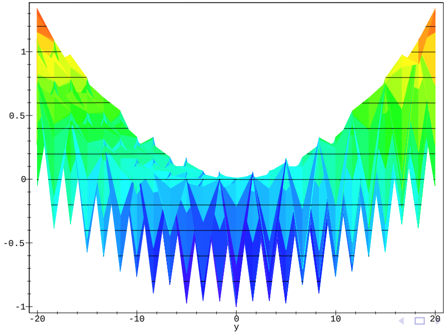
If X_i belongs to cluster j:

$$\begin{aligned} d &= dis(X_i, G_j) \\ T_i &= N_j\,[1 - (\frac{d}{d_{max}})^a] \end{aligned}$$

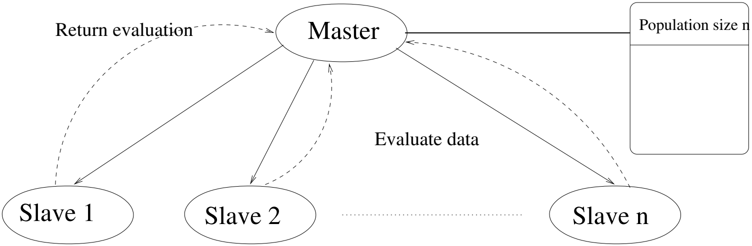
Griewank's function

$$\forall (x_1, .., x_{10}) \in [-10000, 10000]^{10}$$

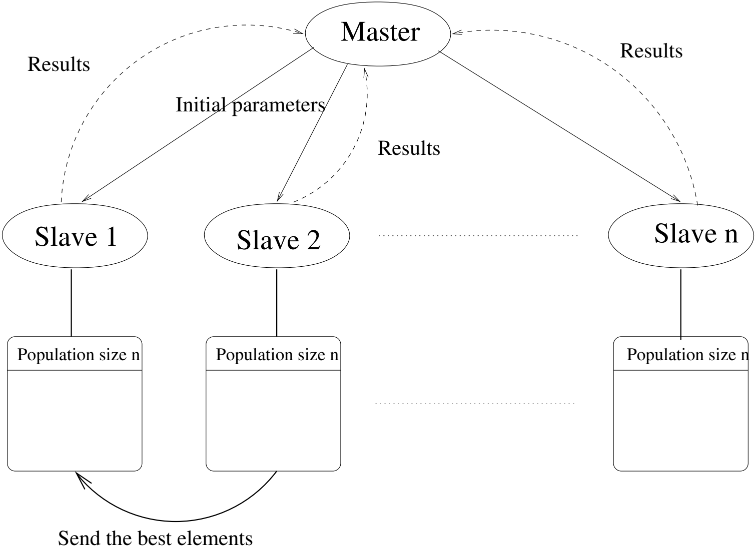
$$F(x_1, .., x_{10}) = \frac{1}{4000} \sum_{i=1}^{10} x_i^2 - \prod_{i=1}^{10} \cos\left(\frac{x_i}{\sqrt{i}}\right)$$



Master Slave parallel GAs

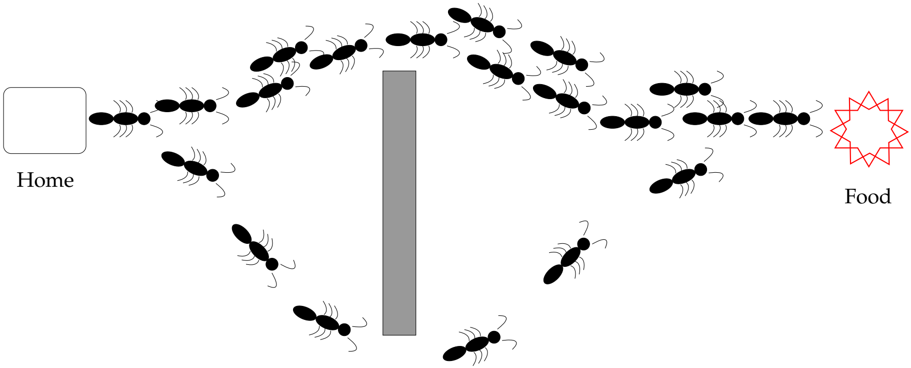


Multiple population parallel GAs



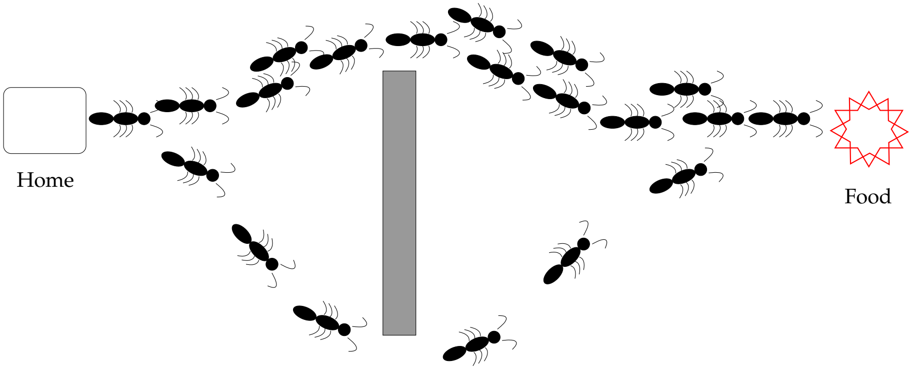
ACO principles

- Use the environment as a medium of communication



ACO principles

- Use the environment as a medium of communication
- Mimic the ants trying to find the shortest path from their colony to food



ACO algorithm principle

- Ants deposite pheromones according to the quality of the path they find

ACO algorithm principle

- Ants deposite pheromones according to the quality of the path they find
- Ants more likely to follow paths with the most pheromones

ACO algorithm principle

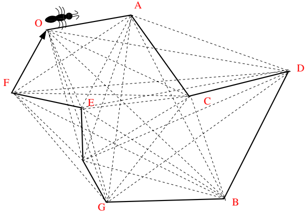
- Ants deposite pheromones according to the quality of the path they find
- Ants more likely to follow paths with the most pheromones
- Add evaporation process to prevent algorithm from local convergence

ACO algorithm principle

- Ants deposite pheromones according to the quality of the path they find
- Ants more likely to follow paths with the most pheromones
- Add evaporation process to prevent algorithm from local convergence
- Stop when no more improvement

ACO for the Traveling Salesman Problem

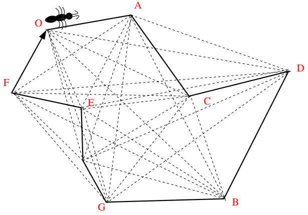
- Ants sent on graph. Each ant builds complete path. Choice of next city influenced by pheromone quantity on paths.



Video

ACO for the Traveling Salesman Problem

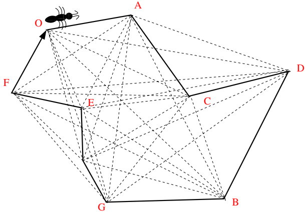
- Ants sent on graph. Each ant builds complete path. Choice of next city influenced by pheromone quantity on paths.
- Ants deposite pheromones on the path chosen: $\Delta\tau_{ij}(t) \propto \frac{1}{\sum L_{ij}}$



Video

ACO for the Traveling Salesman Problem

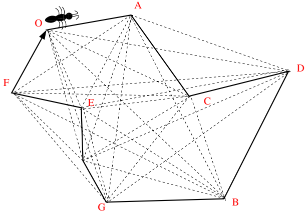
- Ants sent on graph. Each ant builds complete path. Choice of next city influenced by pheromone quantity on paths.
- Ants deposite pheromones on the path chosen: $\Delta\tau_{ij}(t) \propto \frac{1}{\sum L_{ij}}$
- At each iteration, evaporate trails: $\tau_{ij} \leftarrow \rho \cdot \tau_{ij}$



Video

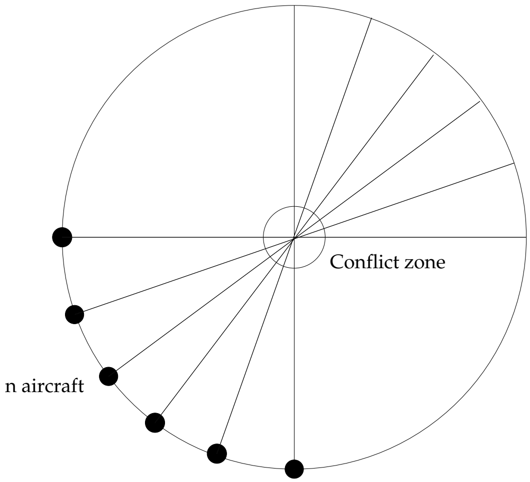
ACO for the Traveling Salesman Problem

- Ants sent on graph. Each ant builds complete path. Choice of next city influenced by pheromone quantity on paths.
- Ants deposite pheromones on the path chosen: $\Delta\tau_{ij}(t) \propto \frac{1}{\sum L_{ij}}$
- At each iteration, evaporate trails: $\tau_{ij} \leftarrow \rho \cdot \tau_{ij}$
- Stop when no more improvement

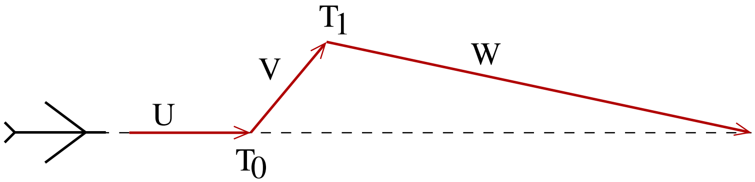


Video

n aircraft conflict example



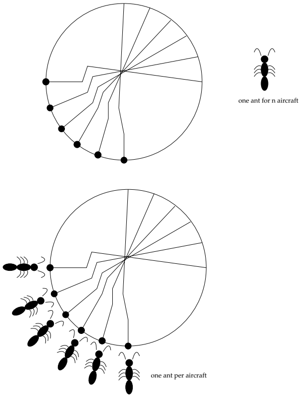
Maneuver modeling



Discretize time into *timesteps*
3 possible angles: 10, 20 or 30 degrees

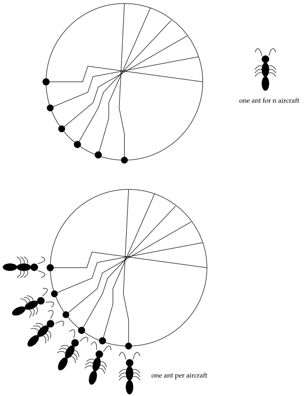
One ant per cluster or one ant per aircraft

- one ant → one cluster



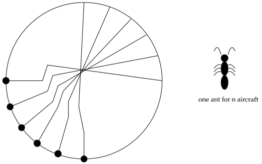
One ant per cluster or one ant per aircraft

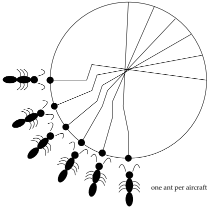
- one ant $\rightarrow$ one cluster
- for n aircraft and t timesteps: $(1 + 3t + 3t^2)^n$ trails.



One ant per cluster or one ant per aircraft

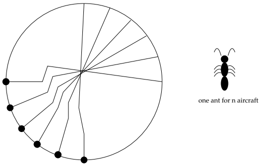
- one ant $\rightarrow$ one cluster
- for n aircraft and t timesteps: $(1+3t+3t^2)^n$ trails.
- For $n=5$ and $t=10$: more than 10^{12} trails

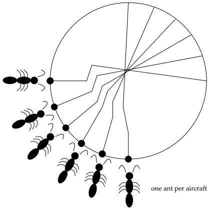




One ant per cluster or one ant per aircraft

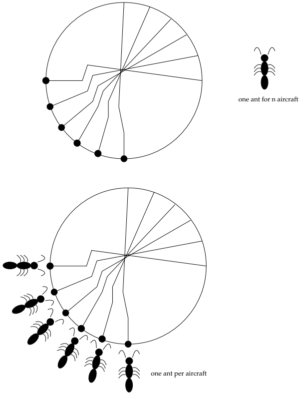
- one ant → one cluster
- for n aircraft and t timesteps: $(1+3t+3t^2)^n$ trails.
- For $n = 5$ and $t = 10$: more than 10^{12} trails
- one ant → one aircraft





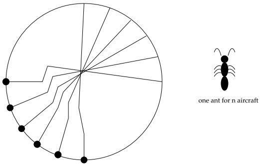
One ant per cluster or one ant per aircraft

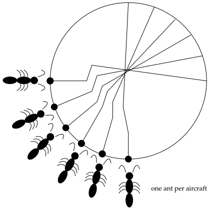
- one ant $\rightarrow$ one cluster
- for n aircraft and t timesteps: $(1 + 3t + 3t^2)^n$ trails.
- For $n = 5$ and $t = 10$: more than 10^{12} trails
- one ant $\rightarrow$ one aircraft
- for n aircraft and t timesteps: $n\,(1 + 3t + 3t^2)$ trails.



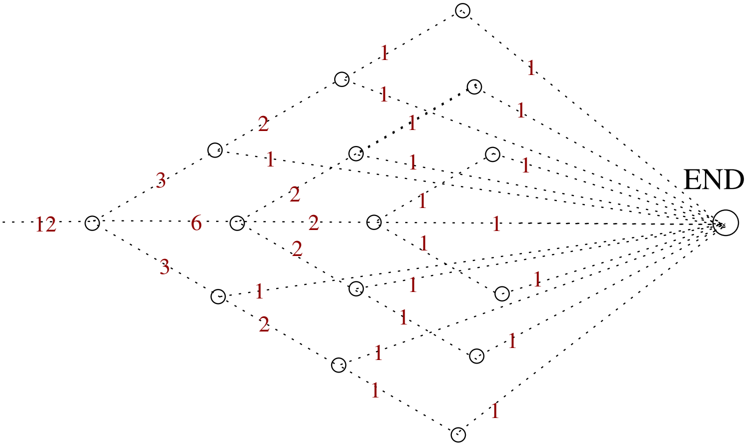
One ant per cluster or one ant per aircraft

- one ant $\rightarrow$ one cluster
- for n aircraft and t timesteps: $(1 + 3t + 3t^2)^n$ trails.
- For $n = 5$ and $t = 10$: more than 10^{12} trails
- one ant $\rightarrow$ one aircraft
- for n aircraft and t timesteps: $n\,(1 + 3t + 3t^2)$ trails.
- For $n = 30$ and $t = 20$: more than 37830 trails instead of 10^{93}

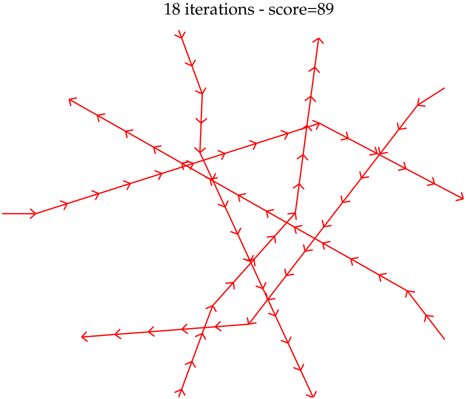




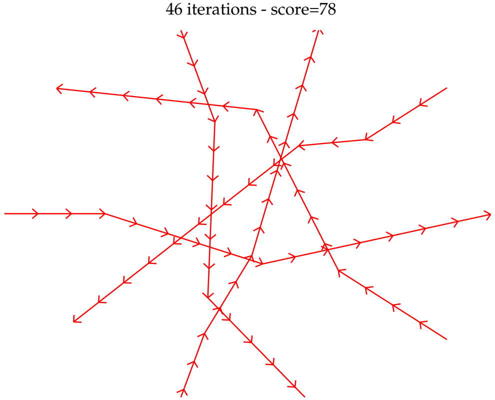
Initial amount of pheromones on the graph



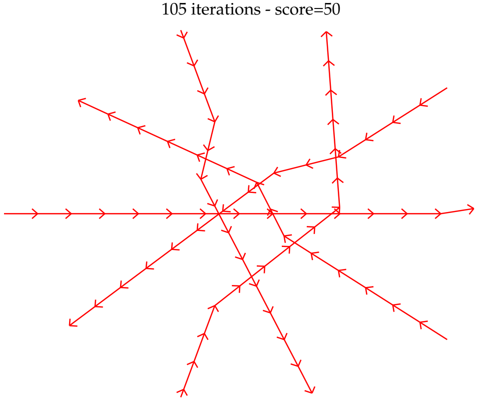
Example of 5 aircraft conflict resolution



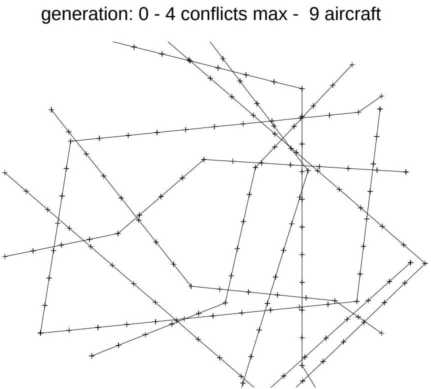
Example of 5 aircraft conflict resolution



Example of 5 aircraft conflict resolution

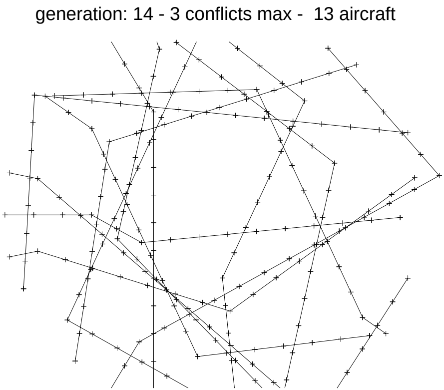


Example of 30 aircraft conflict resolution



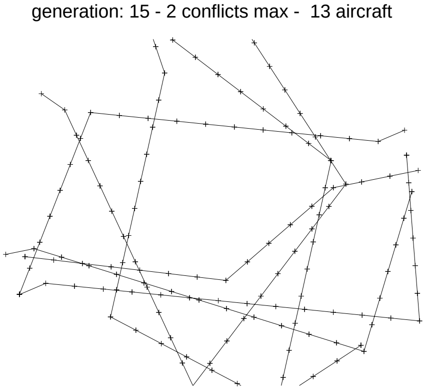
Video

Example of 30 aircraft conflict resolution



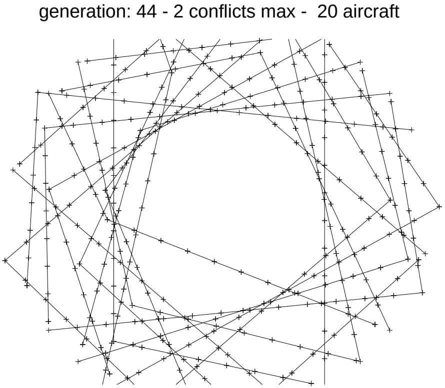
Video

Example of 30 aircraft conflict resolution



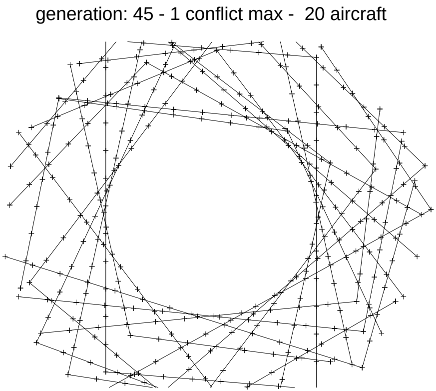
Video

Example of 30 aircraft conflict resolution



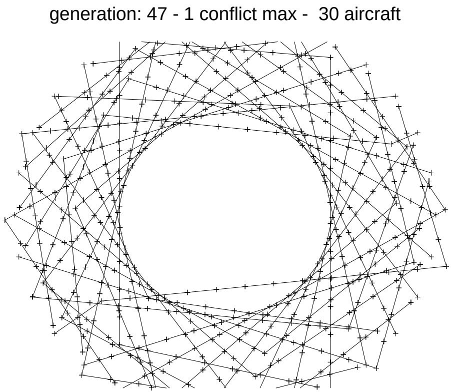
Video

Example of 30 aircraft conflict resolution



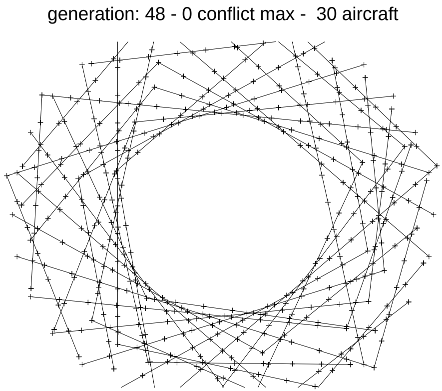
Video

Example of 30 aircraft conflict resolution



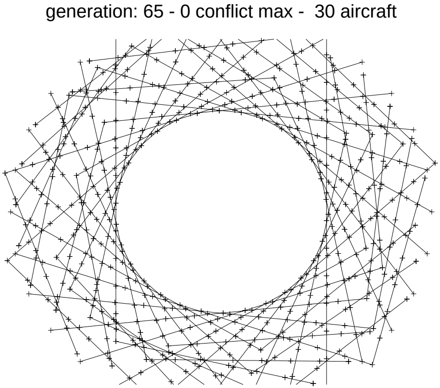
Video

Example of 30 aircraft conflict resolution



Video

Example of 30 aircraft conflict resolution



Video