Stochastic optimization -

Mono-agent methods :
Recall & LAHC/SA

Recalls

Metaheuristics

Methods that are :

- global ;
- stochastic ;
- generic they depend on the context (continuous/discrete) ;
- does not guarantee optimality ;
- based on analogies with biology, physic, animals behaviours...

Elements & strategies

Usually, they are composed of :

- one or many initial candidate solutions (population(s)) ;
- a random-based neighbor creation strategy ;
- a new candidate acceptance criteria ;
- coefficients parameters to control algorithm behavior ;
- information sharing, a memory, a selection strategy, … ;
- a convergence criteria.

Searching...

Exploration

VS

Exploitation

Neighborhood

- Definition
  - Discrete variables : shift/permutation of data,
  - Continuous variables : hyper(sphere|cube) centered around the data.
- Strategies
  - geographic neighborhood,
  - social neighborhood,
  - random neighborhood.
- Gradient, Newton-Raphson, dichotomy methods, Nelder-Mead polytope...
- Hybridation local search methods.

Late Acceptance Hill Climbing

Late Acceptance Hill Climbing (LAHC)

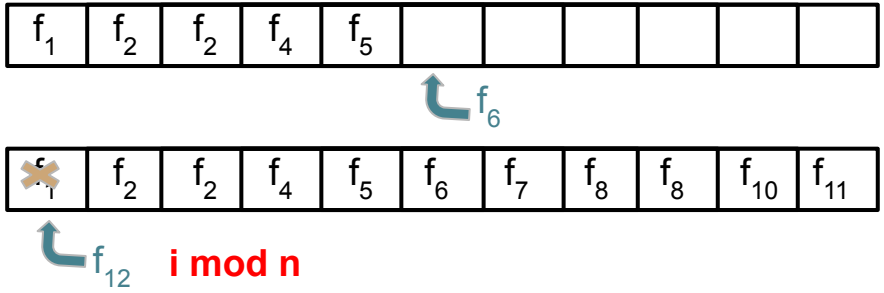
- Proposed by Edmond K. Burke & Youri Bykov (2008) ;
- Mono-agent, adapted to discrete problems, algorithm ;
- Use of **scores memory** of visited candidates ;
- Based on Greedy Hill Climbing heuristic : “Accept a new candidate if it is better than current” ;
- “Accept a new candidate if it is better than a previous one (from a fixed number of iterations)” ;
- A unique parameter : memory size.

LAHC : principle

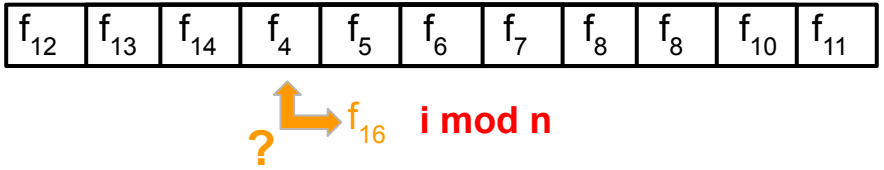
- Current candidate generates a new neighbor ;
- Current evaluation is added at the end of evaluations memory (modulo *n*) ;
- Compare the new evaluation to the performance of n^{th} past candidate (*n* : memory size) ;
- If better, the neighbor becomes the current candidate.

LAHC : illustration

- Current candidate score is added to memory ;



- The comparison is made between the neighbor evaluation and the n^{th} previous evaluation.



LAHC algorithm (minimizing)

```
X, a candidate, f(X) its evaluation
f_min <- f(X), X_min <- X, tabMem <- [f_min|...|f_min]
n <- 1
While not stoppingCriterion()
     X_vois <- generateNeighbor(X)
     f_mem <- tabMem[n % memSize]
     If f_mem ≥ f(X_vois) then
          X <- X_vois
     end if
     tabMem[n % memSize] <- f(X)
     if f_min > f(X_vois) then
          X_min <- X_vois
          f_min <- f(X_vois)
     end if
     n++
End while
```

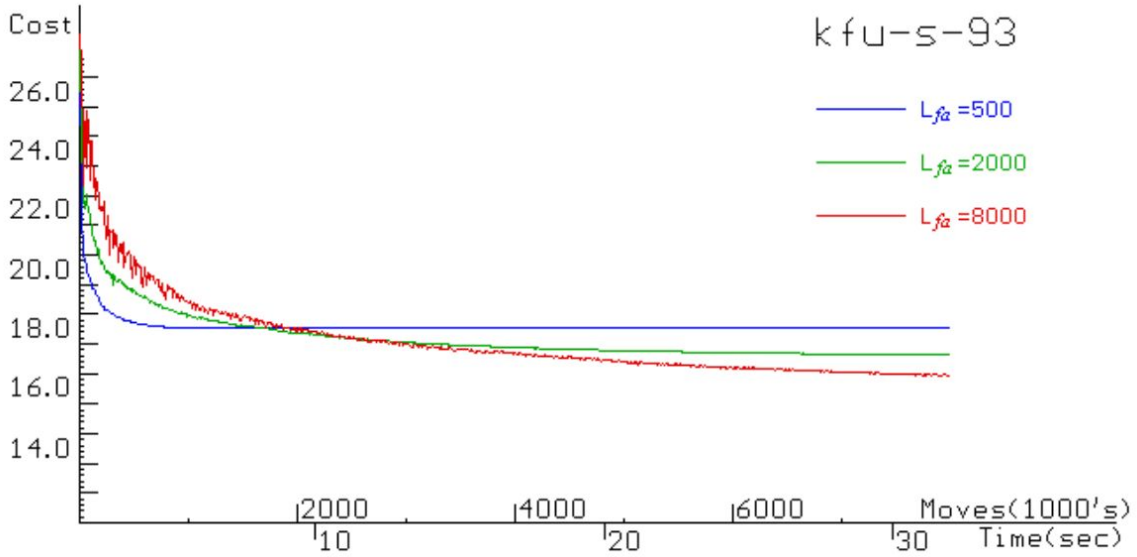
LAHC : memory management

The memory of previous candidate scores is managed as :

- Array of *n* evaluations of visited candidates ;
- The memory size is a **critical** parameter :
  - *n* small : few values recorded, behavior similar to hill-climbing heuristic = premature convergence,
  - *n* big : many random moves favours diversity but slow down convergence.
- At initializing, all values are initialized to the first one. It prevents from too random moves during the first iterations.

LAHC : memory management

Influence of memory size of the convergence behavior in a case of scheduling problem :



LAHC : improvements

Comparison to a previous candidate :

- Depends on memory size (original version) ;
- Compare to memory best score ;
- Compare to a random value of memory ;
- Compare to a mean/median of memory values.

LAHC : improvements

Accepting criterion is :

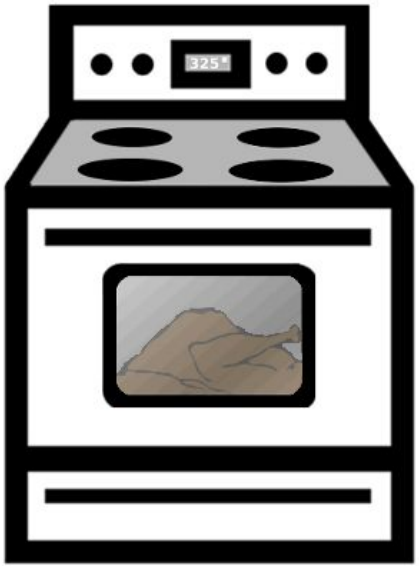
- Deterministic :
  - evaluation less or equal to n^{th} previous memory score (by default),
  - evaluation strictly less than n^{th} previous memory score,
  - evaluation less or equal to n^{th} previous memory score, or current one,
  - ...
- Probabilistic :
  - Metropolis-like : $p = e^{-(c_{vois} - c_{cur}) / (mem[i] - c_{cur})}$,
  - linear : $p = 1 - (c_{vois} - c_{cur}) / [2(mem[i] - c_{cur})]$.

LAHC : improvements

Memorize score according to :

- Descending-only:
  - replace in memory if c < mem[i] ;
- Cooling strategy :
  - after convergence, increment all memory stored scores of *p%* (p=20).

Simulated annealing



Simulated annealing

- Metaheuristic, variant from Metropolis-Hastings algorithm (see example here) ;
- Proposed by Kirkpatrick, Gelatt & Vecchi (1983) and by Cerny (1985) ;
- First metaheuristic ;
- Adapted to discrete problems (originally configure electronic composants disposition on a printed circuit).

Criterion of ....

```
critereMetropolis(Δf,T) :  //minimizing
    If Δf ≤ 0 then
        return TRUE
    else
        return random(0,1) < e^(-Δf/T)
```

- $\Delta f \leq 0$, neighbor is accepted ;
- A small offset towards a less good neighboras more chance to be accepted than a huge one ;
- **Stochastique** function.

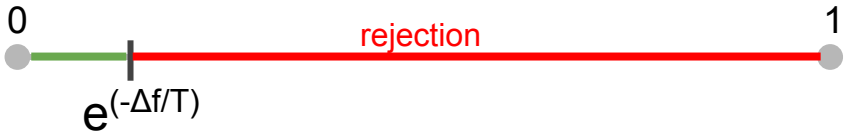
Algorithm (minimization)

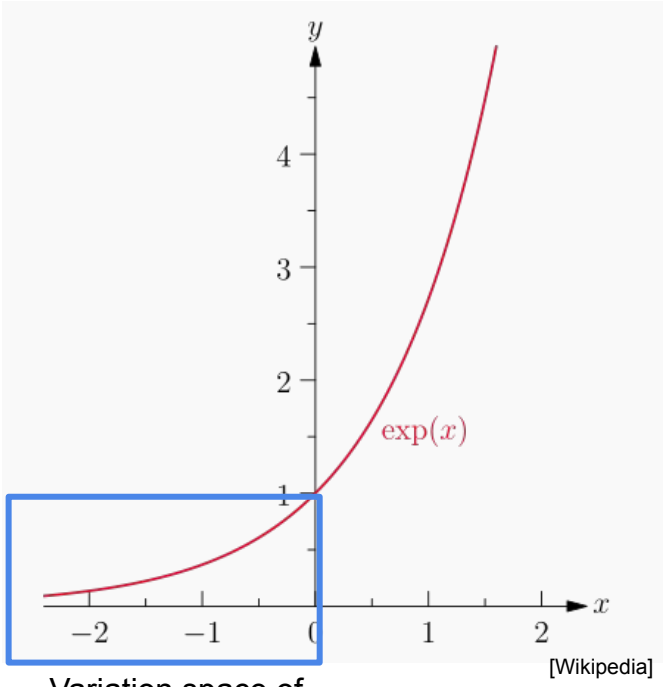
```
X, a random candidate, f_X <- f(X) score, T Initial temperature
X_min <- X, f_min <- f(X)
While T > T_min and not stoppingCriterion()
     While not thermodynamicEquilibrium()
          X_vois <- generateNeighbor(X)
          fX_vois <- f(X_vois)
          Δf = fX_vois - f_X
          If metropolisCriterion(Δf,T) then
               X <- X_vois
               f_X <- fX_vois
               If fX_vois < f_min then
                    f_min <- fX_vois
                    X_min <- X_vois
               End if
          End if
     End while
     T <- coooooooooooooling(T)
End while
```

Coooooooooling

- High temperature, Metropolis criterion value is close to 1, most of worst candidates are accepted :

- Low temperature, Metropolis criterion value is close to 0, most of worst candidates are rejected :





[Wikipedia]

Variation space of
Metropolis criterion values

Coooooooooling

Differents schemes of temperature cooling :

- Geometric : $T_{k+1} = \alpha.T_k$, the mostly used ;
- Logarithmic : $T_k = \mu / log(1+k)$, where k : nb of thermodynamic steps & μ constant. expensivze in computation time, rarely used ;
- Exponential : $T_k = T_0.exp(-k/\tau)$, où k : nb of steps & τ constant ;
- esotéric ....

Question :

“How to profit from a distribution framework to improve an optimization algorithm”

?