Presentation of Some Metaheuristics for the Optimization of Complex Systems

Fatma Tangour[1, 2] **and Pierre Borne**[1]

[1]LARA, École Nationale d'Ingénieurs de Tunis, BP 37, Le Belvédère 1002 Tunis, Tunisie

[2]LAGIS, École Centrale de Lille, Cité scientifique, BP 48, 59651 Villeneuve d'Ascq Cedex, France

fatma.tangour@laposte.net, PIERRE.BORNE@EC-LILLE.FR

Abstract: Some metaheuristics for the optimization of complex systems are proposed in this paper. The metaheuristic approaches can be separate in two classes: the local search techniques and the global ones. An important difficulty which appears in complex optimization problems is the existence of constraints which can be strict and inviolable or soft. To resolve these problems, some hybrid approaches are considered.

Keywords: Optimization, complex systems, metaheuristics, tabu search, simulated annealing, genetic algorithms, ant colony optimization, particle swarm optimization, tunneling Algorithms.

Fatma Tangour graduated from Ecole Normale Superieure de l'Enseignement Technique of Tunis, received the Master degree of automatic and signal treatment in 2004 and Doctoral degree from the Ecole Nationale d'Ingénieur de Tunis and the Ecole Centrale de Lille in 2007. She is currently researcher in LARA Automatique at the Ecole Nationale d'Ingénieur de Tunis and working with the higher institute of sciences applied and technology of Kairouan, Tunisia. Her research interests are in the area of optimization methods for discrete events systems, computer science and operational research.

Pierre Borne received the Master degree of Physics in 1967, the Masters of Electronics, of Mechanics and of Applied Mathematics in 1968. The same year he obtained the Diploma of "Ingénieur IDN" (French "Grande Ecole"). He obtained the PhD in Automatic Control of the University of Lille in 1970 and the DSc of the same University in 1976. He became Doctor Honoris Causa of the Moscow Institute of Electronics and Mathematics (Russia) in 1999, of the University of Waterloo (Canada) in 2006 and of the Polytechnic University of Bucarest (Romania). He is author or co-author of about 200 Journal articles and book chapters, and of 34 plenary lectures and of more than 250 communications in international conferences. He has been the supervisor of 68 PhD thesis and is author of 20 books. He is Fellow of IEEE and has been President of the IEEE/SMC society in 2000 and 2001. He is presently Professor "de classe exceptionnelle" at the Ecole Centrale de Lille and director of the french pluriformations national group of research in Automatic Control

1. Introduction

Many optimization problems such as combinatorial optimization [3, 31] ones are usually N-P hard problems which prevent the implementation of exact used solving methodologies. It is the reason why engineers prefer to use metaheuristics which are able to produce good solutions in a reasonable computation time. The metaheuristic approaches can be separate in two classes: the local search techniques and the global ones. Among the local search techniques the Tabu search [14] is the more known. The other methods usually involve a part of stochastic approach, like the Simulated Annealing [1, 6, 27], the Genetic or Evolutionary Algorithms [28, 29, 36], the Ant Colony Optimization [11] or the Particle Swarm Optimisation [4, 8].An important difficulty which appears in complex optimization problems is the existence of constraints which can be strict and inviolable or soft but with penalization which increase strongly with the degree of violation.

A possible acceleration of the convergence can be obtained by using tunnelling algorithms.

The muliobjective optimization is considered at the end of this paper with presentation of the OWA approach, of the Choquet integral and the Pareto optimality.

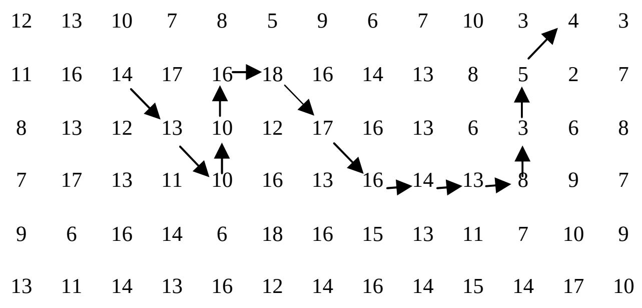
2. Tabu Search

Tabu search [12, 15, 16] is a local optimization method which enhances the performance of a local search method by using memory of the previous obtained solutions in order to permit to escape to a local optimum. It is an iterative local search procedure which enables to move from a solution to another solution in its neighbourhood until stopping criterion is satisfied. In practice the main Tabu search approach consists to determine, starting from a solution, the best solution in its immediate neighbourhood with interdiction to go to one of the N previous obtained solutions. Let us denote N(x) the list of the N solutions that have been visited in the recent past, at each step of iteration we eliminate the oldest solution of the list and we add the new on. With this method we avoid to have a cyclic evolution. It can appear that during some time we can have a degradation of the solution but it enables us to get out of a local optimum and to enlarge the search space.

Another type of Tabu search corresponds to another definition of the Tabu list: which can prohibit solutions that have certain attributes or which can prevent certain solutions that contain prohibited attributes.

An example of Tabu search corresponds to the research of the smallest value in the following Notice Board.

The initial solution corresponds to a fitness equal to 12 and the Tabu list comports 5 elements, figure 1.



12	13	10	7	8	5	9	6	7	10	3	4	3
11	16	14	17	16	18	16	14	13	8	5	2	7
8	13	12	13	10	12	17	16	13	6	3	6	8
7	17	13	11	10	16	13	16	14	13	8	9	7
9	6	16	14	6	18	16	15	13	11	7	10	9
13	11	14	13	16	12	14	16	14	15	14	17	10

Figure 1. The research of smallest value

3. Simulated Annealing

3.1 Principle

Simulated annealing [23, 32] is a generic probabilistic algorithm developed to solve global optimization problems for a function defined in a large search space. The simulated annealing has obtained excellent results in various complex problems known for their important combinatory properties. It is inspired of the physical thermic annealing. At each step of calculation of this algorithm the current solution is replaced by a nearly one chosen with a probability that depends of the variation of a fitness function (called the energy function by analogy with the physical process) via a parameter T (called the temperature) which gradually and regularly decrease during the process. In this approach the solution changes almost randomly for the large value of T and tends globally to obtain the minimum of the energy function as T tends to zero. The random evolution enables motions in which the energy can sometime increase which avoid falling and being trapped in a local minimum which can appear with usual downhill methods as the gradient method.

This algorithm can be presented as follows: let us denote *s*, *T* and *e* respectively the current state, temperature and energy, and s_n and e_n respectively the new state and energy.

The process is initialize with $s: = s_o$ and $e: = e_o$ which correspond to the initial state s_o of energy e_o at time $k = 0$.

While the stopping condition is not satisfied (time $k < k_m$, and energy $e>e_m$), pick some state in the neighbourhood and compute its energy.

3.2 Example of simulated annealing algorithm

Initialization:

```
$s = s_0, e = E(s), k = 0, T = T_0$
while $k < k_m$ and $e > e_m$
$s_{k+1} = \text{neighbour}\left(s_k\right)$
$e_{k+1} = E\left(s_{k+1}\right), \Delta e_k = e_{k+1} - e_k$
if random $\left[0,1\right] < \exp\left(-\Delta e_k / T_k\right)$
then $s_k := s_{k+1}, e_k := e_{k+1}, T_{k+1} := T_k, k = k+1$
return while
```

After stabilization, decrease T and return to s.

We always save the best solution that will be the final solution given by the simulated annealing algorithm.

4. Genetic Algorithms

4.1 Principle

Genetic algorithms [9, 13, 17, 18, 26] are iterative algorithms whose aim is to optimize a fitness function. These exploration algorithms are a particular class of evolutionary algorithms [2, 19, 30, 37] based on natural biological evolution of a population who evolve by selection, crossover between individuals and mutation.

4.2 Crossover

The crossover corresponds to an exchange of genes usually between two individuals of the population. For example if we have two parents, the two points crossover corresponds to the exchange of genes represented in Figure 2.

The two parents are selected according to their fitness via a probability defined by the roulette wheel and the crossover points can be decided with a stochastic approach or using special rules.

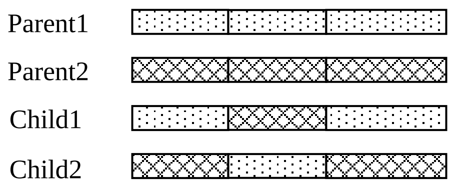


Figure 2. Two point's crossover

4.3 Example of mutation

Several positions are randomly chosen in the chromosome and the corresponding genes are randomly modified. The mutation enables to keep a sufficient diversity in the population and enables to acquire a chromosome gene value which was not already present in the population.

4.4 General algorithm

With each generation a new set of individuals (population) is created by using best parts of the precedent generation as well as innovating parts. The genetic algorithms are not purely random. They effectively exploit information obtained previously to speculate in the choice of new solutions to explore, with the hope to improve the performance. The main difficulty in the implementation of the genetic algorithms to solve an optimization problem is to determine a good coding of the problem called chromosome. Each chromosome represents an individual (a solution). An individual can have a binary representation using number 0 and 1, but any other alphanumerical encoding can be used. The algorithm is initialized by a population that can be determined by another approach or whose individuals are randomly generated. Starting from this initial population new generations are created from which the fitness of every individual is evaluated.

An example of implementation of genetic algorithm can be summarized as follows, figure 3:

1. Create an initial population
2. Evaluate the fitness of each individual of this population
3. While the terminating condition is not satisfied, repeat:
   - Select best ranking individuals to reproduce.

- Breed new generation through crossover and mutation to create offspring.
- Evaluate the fitness of the new individuals.
- Replace worst ranked individuals of the population by the new ones.

In practice, the algorithm needs to be adapted to the specificities of the studied problem and in particular crossover and mutation are to be defined in order to create viable individuals satisfying all the conditions needed for the specific problem.

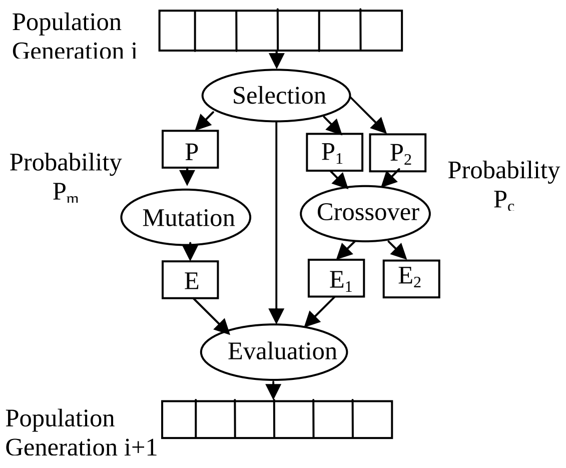


Figure 3. Genetic Algorithms

The more important is to choose the good chromosome for the encoding of the solution. The mutations which correspond to the random change of some characteristics of individual insure to maintain a sufficient diversity in the population and avoid converging prematurely towards local optimum rather than to the global optimum of the problem.

4.5 Example of coding for planning and scheduling optimization

O_{ij} : operation *i* of job *j*
k_{ij} : machine use to achieve operation O_{ij}
t_{ij} : starting time of the operation O_{ij}
For example for three jobs, two with three operations and one with two operations

O_{11}, k_{11}, t_{11}	O_{21}, k_{21}, t_{21}	O_{31}, k_{31}, t_{31}
O_{12}, k_{12}, t_{12}	O_{22}, k_{22}, t_{22}	O_{32}, k_{32}, t_{32}
O_{13}, k_{13}, t_{13}	O_{23}, k_{23}, t_{23}	

The various operations have to satisfy various constraints:

- precedence,
- resource disponibility,
- preemption possible or not,
- earliest starting time,
- due date,
- perisability,…

For optimizing various criteria

- makespan,

- maximum workload,
- maximum delay penalty,…

5. Ant Colony Optimization

5.1. Principle

Initially, the ant colony optimization [10, 25] was inspired by the ability and the organisation of real ant colony using external chemical pheromone trails acting as means of communication, figure 4.

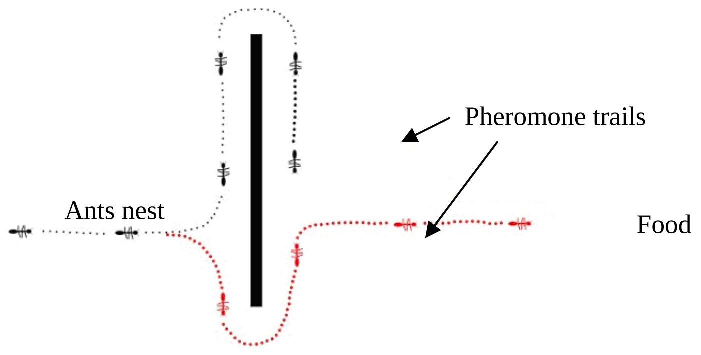


Figure 4. Pheromone trail for ants' communication

The main idea is that of a parallel search over several constructive computational solutions based on characteristics problem data and on a dynamic memory structure containing information on the quality of the previous obtained solutions. Generally the behaviour of an ant system optimization mechanism depends on many unsure parameters, incomplete knowledge of the real ant system attitude and the imprecise information for the identification of the relationship, the strategy of choice of the parameters and the global behaviour of the ant system metaheuristics. In practice, ants looking for food at the beginning wander randomly and having found food return to their colony while laying down pheromone trails. So other ants finding such a path are likely not to keep travelling at random but prefer generally to follow the trail, returning and reinforcing it. In fact pheromone trail slowly evaporate, reducing its attractive strength, so the more time it takes for an ant to achieve its trip to the food, the more time the pheromones have to evaporate. So the shortest path keep the highest density of pheromone trail as pheromone is laid on the path faster as it can evaporate. The evaporation is necessary in order to avoid the premature convergence to a locally optimal solution. Once an ant has found a short path from the colony to a food source, other ants prefer to follow that path, which involves a positive feedback such as finally all the ants will follow this single path. The ant colony optimization algorithms mimic this behaviour with simulated ants evolving on the graph representing the problem to solve.

5.2. Example of application of ACO to planning and scheduling

P_{ijk}^{f}	: Probability for the ant f to assign the operation i of job j (O_{ij}) to machine k (O_{ijk})
p_{ijk}	: Processing time of O_{ij} with the machine k
τ_{ijk}	: Pheromone trail related to O_{ijk}
D	: Set of none performed operations
α, β, ρ	: Parameters of the algorithm (positive)
L_{ijk}^{f}	: Minimum value of the criterion obtained by the ant f performing O_{ijk}
$L_{\min}$	: Global obtained minimum

- Example of ACO Algorithm

$$\tau_{ijk}(t+1) = \rho.\tau_{ijk}(t) + \sum_{f} \Delta\tau_{ijk}^{f}(t+1) \quad (1)$$

$$\Delta\tau_{ijk}^{f}(t) = \frac{L_{\min}}{L_{ijk}^{f}(t)} \quad (2)$$

$$P_{ijk}^{f} = \frac{(\tau_{ijk})^{\alpha}.(p_{ijk})^{-\beta}}{\sum_{j\in D}(\tau_{ijk})^{\alpha}.(p_{ijk})^{-\beta}} \qquad if \quad j \in D \quad (3)$$

$$P_{ijk}^{f} = 0 \qquad otherwise \quad (4)$$

α, β positive parameters of the algorithm.

5.3 Application

Let us consider a flexible job shop scheduling problem composed by three jobs *Jj* (j=1,2,3) and six machines M_k, k = 1;:: 6.

The objective is to optimize the completion time of scheduling, the makespan.

Table 1 depicts, for each job, the operation ordering and the processing time required by each machine.

Table.1. Example: benchmark 3 jobs - 6 machines: processing time

		M_1	M_2	M_3	M_4	M_5	M_6
J_1	$O_{1,1}$	10	7	6	13	5	1
	$O_{2,1}$	4	5	8	12	7	11
	$O_{3,1}$	9	5	6	12	6	17
	$O_{4,1}$	7	8	4	10	15	3
J_2	$O_{1,2}$	15	12	8	6	10	9
	$O_{2,2}$	9	5	7	13	4	7
	$O_{3,2}$	14	13	14	20	8	17
J_3	$O_{1,3}$	7	16	5	11	17	9
	$O_{2,3}$	9	16	8	11	6	3
	$O_{3,3}$	6	14	8	18	21	14

Applying the ant colony optimization meta-heuristic, the results simulation propose different scheduling with *Cmax* = 19 ut (unit of time), table 2 and table 3.

Table.2. Solution for benchmark 3 jobs - 6 machines.
$\alpha = 0.3 ; \beta = 0.7 ; \rho = 0.5$

S_1	O_1	O_2	O_3	O_4
J_1	M_6: [0,1]	M_1: [1,5]	M_5: [5,11]	M_6: [10,13]
J_2	M_4: [0,6]	M_2: [6,11]	M_5: [11,19]	***
J_3	M_3: [0,5]	M_6: [5,8]	M_1: [8,14]	***

The solution given in the table 2 has a makespan equal to 19 ut. The machine M_5 is the cause of this value of makespan. To solve this problem, the tabu search optimisation work is applied for this solution. This method finds the operation $O_{2,2}$ for job J_2 on M_2 that can be swapped with other machines which will reduce makespan to 18 ut. And this method finds that the operation $O_{3,1}$ for the job J_1 executed by M_2 and can be swapped with M_5 who will execute the operation $O_{2,2}$ for the job J_2. The obtained solution by the tabu search method presented in table 3, [25].

Table.3. Tabu search optimisation solution**.**

S_1	O_1	O_2	O_3	O_4
J_1	M_6: [0,1]	M_1: [1,5]	M_2: [5,10]	M_6: [10,13]
J_2	M_4: [0,6]	M_5: [6,10]	M_5: [11,18]	***
J_3	M_3: [0,5]	M_6: [5,8]	M_1: [8,14]	***

6. Particle Swarm Optimization

6.1 Principle

Particle swarm optimization [21, 22] is a population based stochastic optimization technique. It is founded on the notion of cooperation between agents (the particles) that can be seen as animals with limited intellectual capacities: small memory and small intelligence. The exchange of information between them permits nevertheless that globally they succeed to solve difficult problems as it appears with bees, fishes or birds. It appears that social sharing of information among individuals in competition offers an evolutionary advantage. In the particle swarm optimization algorithm, particles move in multidimensional space and are characterized by a position and a velocity. They have two essential reasoning capabilities: the memory of their own best position and the knowledge of their neighbourhood best position.

The standard version of the algorithm can be summarized as follows:

At the beginning the particles of the swarm have a random repartition in the search space and a random velocity [5].

Then at each time step:

- Each particle evaluates the quality of its position and memorizes the best position it has reached at this time and its quality.
- Each particle exchanges information with other particles in its neighbourhood in order to know the best performance of each of them.
- At each time step each particle choose the best performance it knows and modify its velocity according to the whole data it has, to define its moving as a compromise between three tendencies : to keep on its present velocity, to go back towards its own previous best position and to move towards the overall best position it knows, figure 5.

6.2 Particle swarm optimization algorithm

$x_i(t)$: position of the particle i at time t
$v_i(t)$: velocity of the particle i at time t
$x_{im}(t)$: best known position reached by the particle i at time t
$x_M(t)$: best known position reached by the swarm at time t
$x_i(t+1) = f(x_i(t), v_i(t), x_{im}(t), x_M(t))$
Very often this relation is linear

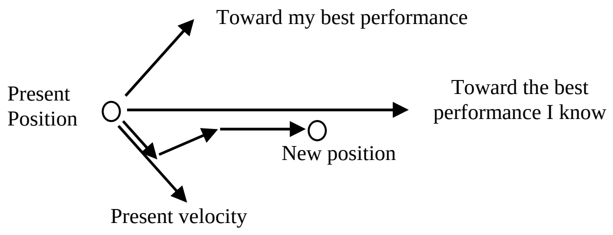

Toward my best performance
Present Position
Toward the best performance I know
New position
Present velocity

Figure 5. Particle swarm optimization principle

7. Tunneling Algorithms

7.1. Principle

Tunneling algorithms [24] enable to escape to local optima. The idea is the following: each time a local optimum is reached the algorithm bore a tunnel towards a new valley of the objective function f, figure 6.

At the origin tunneling approaches have been defined for problems with continuous variables and have been adapted to combinatorial problems later.

Two main strategies have been proposed: the stochastic tunneling and tunneling with penalty functions:

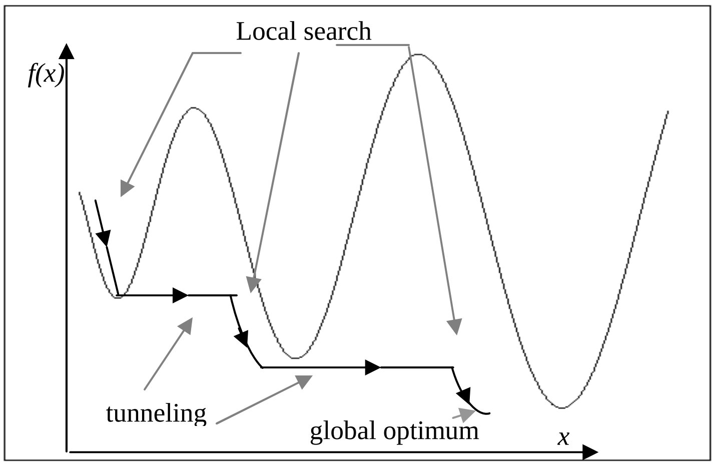


Figure 6. Tunneling algorithm

7.2 Stochastic tunneling

The stochastic tunneling [34] was initially defined to escape to local minima when implementing the simulated annealing algorithms at low temperature. The idea was to circumvent the slow dynamics of ill-shaped energy function by applying a non-linear transformation to the objective function.

7.3 Tunneling with penalty function

The tunneling with penalty functions modifies the value of local optima by adding penalty values in order to facilitate the algorithm to escape to these local optima.

7.4 Example of penalty function for Tunneling

If $f(x)$ is the fitness function we can choose the new fitness function $f_{Tun}(x)$:

$$f_{Tun}(x) = 1 - \exp(-\gamma(f(x) - f(x^*))) \quad (5)$$

where x^* corresponds to the best known solution and $\gamma > 0$.

8. Multiopjective Optimization

When we have several criteria in competition to optimize, various approaches can be considered:

8.1 Ordered Weighted Operator (OWA)

The operator of aggregation C_{OWA} has been introduced by Yager [35]. After normalisation, the various criteria $C_i(x)$ are aggregate in a single one with weight coefficients

$$C_{OWA}(x) = \sum_{i=1}^{n} w_i C_i(x) \tag{6}$$

With $w_i \in [0,1], \sum_{i=1}^{n} w_i = 1$ (7)

8.2 Choquet Integral [7]

It is an OWA type approach in which the weights w_i are calculated according to the interaction between various criteria.

In order to be self-contained as far as possible, necessary definitions are given in this section, adapted for multi-criteria decision making.

Let consider a finite interval set $N_c = \{1,...,n_c\}$, which can be thought as an index set of the given criteria.

- *Definition 1*

A fuzzy measure over $N_c = \{1,...,n_c\}$ is a set function $\mu : P(Nc) \rightarrow [0,1]$, such that:

1. $\mu(\phi) = 0,\ \mu(N_c) = 1$
2. $\mu(A) \leq \mu(B)$ *whenever* $A \subset B \subset N_c$

The meaning attributed to $\mu(A)$ is usually the importance or the power of the coalition A (e.g., for decision making)

- *Definition 2*

Let μ be a fuzzy measure over Nc and $a = (a_1,...,a_{n_c})$ the vector of criteria. The discrete Choquet integral C_μ with respect to μ is defined by:

$$C_\mu(a_1,...,a_{n_c}) = \sum_{i=1}^{n_c} (a_i - a_{i-1}) \mu(\{i,...,n_c\}) \tag{8}$$

with $a_0 = 0$ *and* $a_1 \leq ... \leq a_{n_c}$. (9)

- *Definition 3*

Let μ be a fuzzy measure over Nc. The shapely index I_i, for every $i \in N_c$, is defined by:

$$I_i = \sum_{k \in N_c - \{i\}} \frac{(n_c - |k| - 1)!|k|!}{n_c!} \left(\mu(k \cup \{i\}) - \mu(k)\right) \tag{10}$$

where $|k|$ indicates the cardinal of k and $0! = 1$.

- *Definition 4*

The average interaction index I_{ij} between two criteria i and j, with respect to a fuzzy measure μ, is defined by:

$$I_{ij} = \sum_{k \in N_c - \{i,j\}} \frac{(n_c - |k| - 2)!|k|!}{(n_c - 1)!} \left(\mu(k \cup \{i,j\}) - \mu(k \cup \{i\}) - \mu(k \cup \{j\}) + \mu(k)\right) \tag{11}$$

the interaction index, ranged in $[-1,1]$, is negative in the case of redundancy, and positive in the case of synergy.

- *Definition 5*

The Choquet integral formulation in terms of interaction representation is reduced to an easily interpretable form in the case of (at most) 2-additive measures, which is for any $a=\left(a_1,...,a_{n_c}\right)$, as following:

$$C_\mu\left(a\right)=\sum_{I_{ij}>0}\left(a_i\wedge a_j\right)|\,I_{ij}\,|+\sum_{I_{ij}<0}\left(a_i\vee a_j\right)|\,I_{ij}\,|+\sum_{i=1}^{n_c}a_i\left(I_i-\frac{1}{2}\sum_{i\neq j}|\,I_{ij}\,|\right) \tag{12}$$

with $\wedge$ and $\vee$ denote min and max respectively.

8.3. Pareto optimality approach

Pareto optimality [20, 33] is a measure of efficiency in multicriteria problems.

In this approach, a non dominated solution is such that there is no other solution that performs at best as well on every criterion and which is strictly better on at least one of criteria. For a Pareto optimal solution a criterion cannot be improved without damaging at least one of the other criteria.The set of Pareto-optimal solutions corresponds to the Pareto optimal curve also called front of Pareto, figure 7.

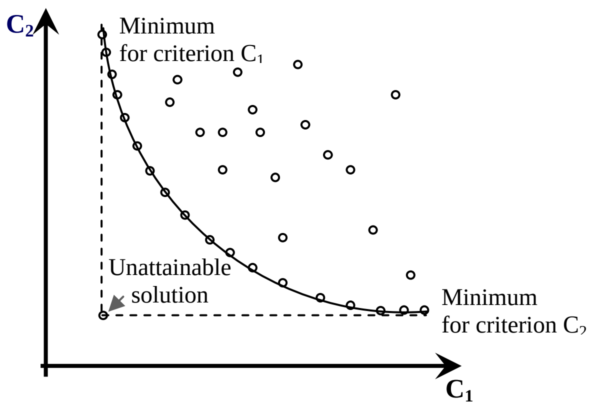


Figure 7. Pareto optimal solution for a problem with two criteria

If we have the possibility to determine lower bounds of the various criteria the Pareto optimality approach can be associate with the OWA approach. After normalisation of the criteria, we realize an aggregation of the various criteria with adaptive weights which enable a dynamic search in the direction of the lower bounds point.

For example if the optimisation is realised with a genetic algorithm, figure 8 represents the evolution of the population.

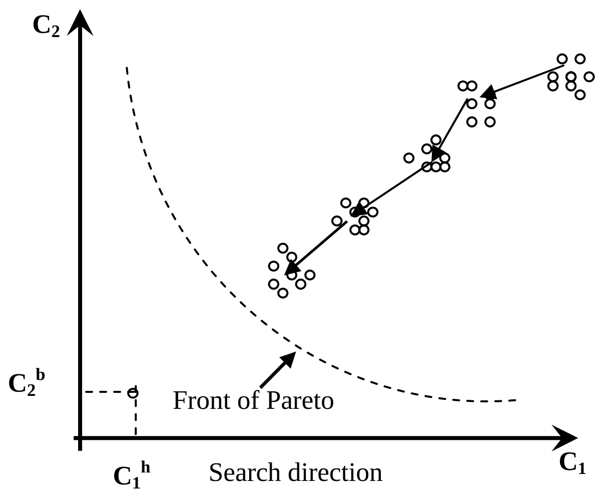


Figure 8. The population evolution

9. Conclusions

The various metaheuristics which have been presented here have been implemented in various applications such as the optimization manufacturing problems but in each case the formulation of the problem have to be adapting to the chosen algorithm.

Very often hybrid approaches are implemented using simultaneously several metaheuristics and usual local search like the hill climbing methods.

REFERENCES

1. AARTS, E.H.L., VAN LAARHOVEN P.J.M., **Simulated Annealing: Theory and Applications**, D. Reidel Publishing Company, 1987.
2. BÄCK, T., HAMMEL U., SCHWEFEL H.P., **Evolutionary Computation, Comments on the History and Current State**, IEEE Transactions on Evolutionary Computation, Vol. 1, 1997, pp. 3-17.
3. BLUM, C., ROLI A., **Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison**, ACM Computing Surveys, Vol. 35, No. 3, September 2003, pp. 268-308.
4. BONABEAU, E., DORIGO M., THERAULAZ G., **Swarm Intelligence: From Natural to Artificial Systems**, Oxford University Press, 1999.
5. BORNE, P., TANGOUR F., **Metaheuristics for the Optimization in Planning and Scheduling**, the Fourth Conference on Management and Control of Production and Logistics MCPL 2007, Sibiu, ROMANIA, September 27- 30, 2007.
6. CHAISEMARTIN, P., DREYFUS G., FONTET M., KOUKA E., LOUBIÈRES P., SIARRY P., **Placement and Channel Routing by Simulated Annealing: Some Recent Developments**, Computer Systems Science and engineering, Vol. 41, 1989.
7. CHOQUET, G., **Theory of Capacities,** Annales de l'Institut Fourier, No 5, 1993, pp. 131-295.
8. CLERC, M., KENNEDY J., **The Particle Swarm. Explosion, Stability, and Convergence in a Multidimensional Complex Space**, IEEE Transactions on Evolutionary Computation, Vol. 6, pp. 58-73, 2002.
9. DEJONG, K.A., SPEARS W.M., **Using Genetic Algorithms to Solve NP- Complete Problems**, George Mason University, Fairfax, VA, 1989.
10. DENEUBOURG, J.L., GOSS S., VERHAEGHE J.C., **Probabilistic Behaviour in Ants**, Journal of Theorical Biology, Vol. 105, 1983, pp. 259-271.
11. DORIGO, M., COLORNI A., MANIEZZO V. ,**The Ant System: Optimization By a Colony of Cooperating Agents**, IEEE Transactions on Systems, Man, and Cybernetics-Part B, Vol. 26, No 1, pp.1-13, 1996.
12. ENNIGROU, M., GHÉDIRA K., **Flexible Job-Shop scheduling with Multi-agent System and Tabu Search,** Journal Européen des Systèmes Automatisés, JESA, Vol. 38, No. 7-8, 2004.
13. GHÉDIRA, K., JLIFI B., **A Distributed Guided Genetic Algorithm for Max-CSPs**, Revue d'Intelligence Artificielle, 2002, pp. 1-16.
14. GLOVER, F., LAGUNA M., **Tabu Search**, Kluwer, Norwell, MA, 1997.
15. GLOVER F., **Tabu Search**, **Part I** - ORSA Journal on Computing 1, 1989, pp. 190-206.
16. GLOVER F., **Tabu Search, Part II** , ORSA Journal on Computing 2, 1990, pp. 4-32.
17. GOLDBERG, D.E., **Genetic Algorithms in Search**, Optimisation, and Machine Learning, Reading, Mass, Addison-Wesley, 1989.
18. HOLLAND, J.H., **Genetic Algorithms and the Optimal Allocations of Trials**, In SIAM Journal of computing, Vol. 2, 1973, pp. 88-105.
19. KACEM, I., HAMMADI S., BORNE P., **Fuzzy Evolutionary Approach for Multiobjective Combinatorial Optimization: Application to Scheduling Problems**, Studies in fuzziness and soft computing, Vol. 126, 2003, pp. 197-219.

20. KACEM, I., HAMMADI S., BORNE P., **Pareto-optimality Approach for Flexible Job-shop Scheduling Problem: Hybridization of Genetic Algorithms with Fuzzy Logic**, Mathematics and Computers in Simulation, Vol. 60, 2002, pp. 245-276.

21. KENNEDY, J, EBERHART R., SHI Y., **Swarm Intelligence**, Morgan Kaufmann Academic Press, 2001.

22. KENNEDY, J., EBERHART R., **Particle Swarm Optimization**, IEEE International Conference on Neural Networks, Piscataway, NJ, Proc., 1995, pp. 1942-1948.

23. KIRKPATRICK, S., GELATT C.D., VECCHI M.P., **Optimization by Simulated Annealing**, Science, Vol. 220, No. 4598, 1983, pp. 671-680.

24. LEVY, A.V., GOMEZ S., **The Tunneling Method Applied to Global Optimization**, In: Boggs PT, Byrd R.H., Schnabel RB (eds), Numerical Optimization, SIAM, 1985, pp. 213-244.

25. LIOUANE, N., SAAD I., HAMMADI S., BORNE P., **Ants System and Local Search Optimisation for Flexible Job-shop scheduling Production**, International Journal of Computers, Communication and Control, Vol. 11, No. 2, 2007, pp. 66-75.

26. MESGHOUNI, K., BORNE P., HAMMADI S., **Evolutionary Algorithms for Job-shop scheduling**, Applied Mathematics and Computer, Vol. 14, No. 1, 2004, pp. 91-103.

27. METROPOLIS, N., ROSENBLUTH A.W., ROSENBLUTH M.N., TELLER A.H., TELLER E., **Simulated Annealing**, J. Chem. Phys. 21, 1953.

28. PATNAIK, S., **Genetic Algorithms: A Survey**, IEEE computer society, Vol. 27, No. 6, pp.17-26, 1994.

29. RENDERS, J.M., **Algorithmes génétiques et réseaux de neurones**, Paris: Hermès, 1995.

30. RICHARDSON, J.T., PALMER M.R. LIEPINS G., HILLIARD M., **Some Guidelines for Genetic Algorithms with Penalty Functions**, Proceedings of the Third International Conference on Genetic Algorithms, pp. 191-197, 1989.

31. SCHWEFEL, H.P., **Numerical Optimization of Computer Models**, Chichester Wiley, 1981.

32. SIARRY, P., BERTHIAU G., DURBIN F., HAUSSY J., **Enhanced Simulated Annealing for Globally Minimizing Functions of Many Continuous Variables**, ACM Transactions on Mathematical Software, Vol. 23, No. 2, pp. 209-228, 1997.

33. TANGOUR, F., I. SAAD, **Multiobjective Optimization Scheduling Problems by Pareto-optimality in Agro-alimentary Workshop**, International Journal of Computers Communications & Control, IJCCC**,** Vol. 1, No. 3, 2006, pp. 71-83.

34. WENZEL, W., HAMACHER K., Stochastic **Tunneling Approach for Global Minimization of Complex Potential Energy Landscapes**, Phys. Rev. Lett., Vol. 82, No. 15, 1999, pp. 3003-3007.

35. YAGER, R.R., **On Weighted Medium Aggregation Operators in Multicriteria Decision Making**, IEEE Trans. On Systems, Man and Cybernetics, Vol. 18, 1988, pp. 183-190.

36. ZOMAYA, P.F., **Parallel Genetic Algorithms, Parallel & Distributed Computing**, Handbook, McGraw Hul, 1996.

37. ZRIBI, N., BORNE P., **Hybrid Genetic Algorithm for the Flexible Job-shop Problem under Maintenance Constraints**, Lecture Notes in Computer Science, Vol. 3612, 2005, pp. 259-268.