Improvement of Intelligent Optimization by an Experience Feedback Approach

Paul Pitiot[1], Thierry Coudert[1], Laurent Geneste[1], and Claude Baron[2]

[1] Laboratoire Génie de Production, Ecole Nationale d'Ingénieurs de Tarbes, 47, av. d'Azereix BP 1629 - 65016 Tarbes, France
[2] Laboratoire d'Etude des Systèmes Informatiques et Automatique, INSA de Toulouse, 135, av. de Rangueil - 31077 Toulouse, France
{paul.pitiot,thierry.coudert,laurent.geneste}@enit.fr, claude.baron@insa-toulouse.fr

Abstract. Intelligent optimization is a domain of evolutionary computation that emerges since a few years. All the methods within this discipline are based on mechanisms for maintaining a set of individuals and, separately, a space of knowledge linked to the individuals. The aim is to make the individuals evolve to reach better solutions generation after generation using the knowledge linked to them. The idea proposed in this paper consists in using previous experiences in order to build the knowledge referential and then accelerate the search process. A method which allows reusing knowledge gained from experience feedback is proposed. This approach has been applied to the problem of selection of project scenario in a multi-objective context. An evolutionary algorithm has been modified in order to allow the reuse of capitalized knowledge. This knowledge is gathered in an influence diagram allowing its reuse by the algorithm.

Keywords: Project management, evolutionary algorithm, knowledge management, experience feedback, influence diagrams.

1 Introduction

The management of industrial projects is a more and more complex activity. The constraints to take into account are: multi-domain projects, uncertain and dynamic environment, innovative systems and/or components, multi-criteria optimization, etc. In this context, optimal methods are not suitable because of the complex and large search space. Several studies (see [1] for example) highlight the fact that combinatorial optimization techniques are in general relatively blind methods (i.e. they are not a priori guided). It is usual to launch an optimal search algorithm considering that the search space is uniformly interesting. This hypothesis is, in practice, often proved to be inappropriate. However, some techniques based on meta-models propose to gain knowledge (learning) during the resolution processes. They build a model of objective function (response surfaces, Kriging, etc.). Another alternative is to propose, during the exploration process, some interesting hypotheses of configuration and to use them as guidelines [2].

Nevertheless, being able to reuse the knowledge capitalized during previous optimization processes can be an interesting way to improve future explorations of the search space. The capitalized knowledge can provide interesting information for initial conditions and during exploration of search space. It is however necessary to adapt the knowledge capitalized during previous explorations to the current one.

In the domain of project management, the problem of scenario selection is very difficult, particularly when projects concern the design of new products. This context provides an interesting framework for knowledge reuse. Indeed, the reuse of components or knowledge when designing new systems is an increasingly important and strategic issue for companies. So a lot of information is already available, but not used to accelerate the optimization process. Moreover, the reuse process must take into account variations of environment to improve information used.

So, in this paper a framework for integration of an experience feedback process during optimization processes is proposed. The study is adapted to scenario selection in project management. In the next section, a state of the art about knowledge utilization to guide search methods is presented as well as the objectives of the proposed method based on experience feedback process. In section 3, knowledge acquisition process and model used by experience feedback process are described. Section 4 presents different ways to use knowledge during resolution before concluding and presenting prospects.

2 Knowledge as Guidelines for Optimization Methods: State of Art

The goal of combinatorial optimization techniques is to find a good solution, if possible optimal, according to a set of criteria, in the state space of input parameters (domains of combinatorial parameters). The method searches in this space combinations of parameters leading to interesting areas with respect to evaluation criteria. When the studied problem is too complex, two kinds of search methods are usually used: the meta-modeling of the objective function or the meta-heuristic methods.

The meta-modeling methods, such as neural networks, consider that the objective function, even complex, is coherent. Those methods try to build a regression model used to guide exploration procedure. One major problem with this type of method is the lack of explanations about obtained solutions. Knowledge is learned by the program and then stored in a model (e.g. weights on arcs between neurons) but it remains inaccessible for the user. Their advantage is the capacity of generalization which ensures a good reuse of knowledge on new cases.

Most heuristics and meta-heuristics methods assume that exploration process (local or global) will make it possible to find a good solution in a reasonable time and this by formulating assumptions on the data structure [3]. Heuristics methods are rules improving search of solutions for certain types of problems or for a particular aspect of a complex problem. Thus, they carry out a partial knowledge about a part of the problem or its structure. The search procedure is then accelerated but it is not enough to find a complete solution.

Meta-heuristics are more general models which must be adapted according to the problem to solve. For most of these methods and especially for Evolutionary Algorithms (EA), the method does not consist in spending time to capitalize knowledge about the problem which is too complex or changing. It rather consists in testing (quickly) a great number of possible solutions and to make sure that exploration process converges towards increasingly interesting solutions. This kind of method indirectly reuses knowledge associated with the problem via the evaluation of generated solutions. The assumption is made that carried out knowledge is linked to the quality of generated solutions. The method tries to improve these solutions. But knowledge, used during exploration, is not preserved from one execution to the other. Moreover, it is impossible to reuse only a part of it.

Recently, new methods, called "intelligent optimization methods" [2][4] suggest a coupling between a Model of Knowledge (MoK) about the problem to be solved and a traditional search method. The MoK must guide search towards promising zones while a traditional search method provides a "virtually contextualized information" to the MoK. For the majority of methods listed above, the use of knowledge is achieved indirectly. It is represented by means of classes of operators [5], intervals [6], assumptions on the parameters values [7] or by attributes about good solutions [8]. Works carried out by Chebel-Morello in [9] or Huyet in [4] propose to model the knowledge directly using parameters classes. Each class of parameters is more or less favorable to the different objectives. The problem is that it is very difficult to directly handle this knowledge with the employed methods (Knowledge Discovery in Databases - e.g. decision trees or neural network).

Finally, other methods [10] [11] use different kinds of Bayesian network as MoK. The MoK is learned from a database containing selected individuals from previous generation and it is used to generate directly the new population of individuals by sampling. The step of induction on the probability model constitutes the hardest task to perform and this task had to be performed for each generation.

Moreover, no study proposes a reuse mechanism of knowledge obtained during previous resolution processes to better solve new problems. Such reuse allows building and improving a complex MoK of the problem before optimization process and only using it during search (no knowledge actualization).

Objectives of our study. The proposed framework suggests to use a hybrid method including a meta-heuristic for search and a MoK to provide heuristics adapted to the current case. The MoK should not provide all information precisely but has to give some orientations with respect to a given situation. Michalski in [2] shows that fixing some interesting solutions properties is enough so that search method generates very quickly some solutions close to the optimal one. The system has to ensure the following properties:

1) The search process has to be efficient and to provide optimization even with an incomplete MoK or a failing one. In this study an Evolutionary Algorithm for the search process and an Influence Diagram as MoK are used;
2) Reuse and continuous improvement of operational knowledge has to be performed in an interactive manner (achieved projects) within an experience feedback process;

3) Reuse of the knowledge resulting from the simulations produced by the genetic algorithm. This is possible by means of a Knowledge Discovery in Databases (KDD) process for example;
4) Capacities of knowledge generalization that allow adapting knowledge according to new current cases. This adaptation has also to be performed in interaction with the meta-heuristic.

Application problem. In the domain of project management, the problem of scenario selection is considered. The aim of this application is to solve simultaneously the problem of selecting design alternatives for a system and the project planning problem to achieve this system. The constraints to be taken into account during the project planning are modeled by a project graph proposed in [12] and shown on figure 1. This model allows considering simultaneously the planning constraints and the design constraints. The project graph includes the tasks to be carried out, the AND nodes (parallel tasks) and the OR nodes representing the possible decisions during the design process called "design alternatives". Tasks are represented by means of rectangles with a task number, AND nodes by means of vertical double-bars, OR nodes by means of circles. The gray rectangles inside the tasks represent the different possibilities to carry out a task, called "task options". The selection of a path in the graph represents a potential scenario for the project as show on figure 1.

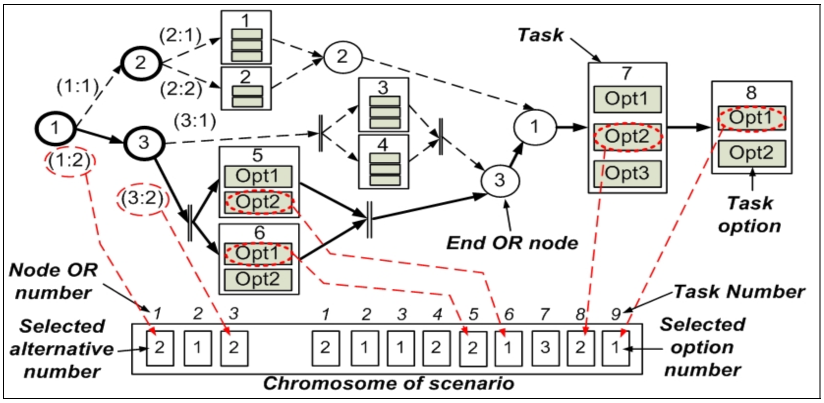

Fig. 1. Graphs of a project and scenario encoding; for example, this one concerns the realization of a pen with a scenario highlighted

Search method used in previous study. The search method is an evolutionary algorithm (EA) proposed by C. Baron in [12] for scenario selection. An EA is a meta-heuristic for stochastic optimization, used for global exploration. The SPEA method used in this study (Strength Pareto Evolutionary Algorithm) [13] is a traditional EA with classical steps (initialization, evaluation, selection) and genetics operators cross-over and mutation. It ensures the multi-objective evaluation of individuals according to the two following steps: i) the Taguchi approach is used in order to evaluate cost of a scenario for each criterion [14]; ii) multi-criteria evaluation is then achieved by means of Pareto front in order to compare and classify the scenarios (concept of dominance between solutions). The probability of selection for an individual is proportional to its fitness. The fitness depends on the position of the individual compared to the Pareto front (maximum fitness for the Pareto-optimal solutions). The fitness of an individual is calculated by formula (1) according to the strength (***S****ᵢ*) of individual which it dominates (Pareto-dominance). The strength of an individual is given by the formula (2) where ***n*** is the number of dominated solutions and ***Pop*** is the population size.

$$f_j = \frac{1}{1 + \sum_{i,\, i<j} s_i} \ . \tag{1}$$

$$s_i = \frac{n}{|Pop| + 1} \ . \tag{2}$$

After selection, cross-over operator is applied (selection of two "parents" and then crossing of their "genes"). Finally, the mutation operator is applied (selection of an individual and change of one or more genes). The criterion used for stopping this process is a strict limit of the number of generations.

EA requires an encoding of individuals. In this model, an individual represents one scenario for the project, i.e. an instantiation of the graph as shown on figure 1. The chromosome of an individual gathers on first part (the left one) the design alternatives choice. Each gene corresponds to a path choice in the project graph and it is represented in chromosome by a number corresponding to the selected alternative. Then on the second part of the chromosome (right one), each gene corresponds to selection of a task option (a number corresponding to selected option).

All possible choices are always represented whereas majority of them are inactive since they are inhibited by alternatives choices. For example, in figure 1, the first design choice (*OR node ① - choice of arc (2) noted (1:2)*) inhibits node ② and then, tasks 1 and 2. The second possible design choice for this scenario (*OR node ③ - choice of arc (2) noted (3:2)*) inhibits the tasks 3 and 4. Consequently, the tasks 1, 2, 3 and 4 are present in the chromosome but their genes are inhibited. This encoding ensures a constant viability of solutions but requires an additional control on genetics operators. A check has to be performed to avoid inefficient mutation or cross-over.

3 Knowledge Acquisition and Modeling

3.1 Knowledge and Experience Feedback

Knowledge management relies on expert's knowledge extraction and direct use of this knowledge through a modeling. Methods using this process (MKSM, Kads) encounter problems such as difficulty of data extraction, expert's availability or knowledge actualization. Experience feedback proposes acquisition and knowledge reuse through the experiences (spontaneous declaration of knowledge during their application). It rests mainly on two cycles of information management: capitalization and generalization. Capitalization is carried out by memorizing behavior of the expert.

Experiences are used as vectors to build knowledge reference frame. Each time that an event occurs, actors formalize their judgment. Indeed, to generalize a model of knowledge starting from lived experiments is easier than to clarify knowledge apart from its context.

In our study, knowledge corresponds to the identification of system and environment of system to be realized. It consists in probabilistic or determinist links between three state spaces: i) input parameters of the problem (i.e. genome where each gene corresponds to a design or planning decision); ii) the evaluation criteria and objectives (discrete values in our study); and iii) parameters characterizing the environment of the project. Indeed, context modeling is necessary to adapt knowledge to current situation. Knowledge about criteria and objectives is related to functionalities of the designed system (e.g. customer's requirement) or on requirements for the project management (e.g. minimize cost, delay, etc). Among knowledge about genome, two different kind of knowledge can be distinguished. The first one is a structural knowledge related to the problem to solve (e.g. the constraints of precedence and inhibition between decisions). Note that this part of knowledge referential is specific to the graph routing problem. The second kind of knowledge used is the set of preferences between genes according to criteria and objectives. Knowledge is extracted from: a) the project graph shape; b) information and analyses of operational experience feedback and finally; c) intermediate simulation results (individuals evaluated by EA).

3.2 The Model of Knowledge (MoK)

Paradigm used to model required knowledge is Influence Diagrams (ID) [15], a stepwise-solvable bayesian network. Firstly, they represent an interesting way for representation and use of knowledge because they were conceived for conceptual representation in decision support. So, they allow the representation of expertise and an interactive management of knowledge coming from experience feedback. Moreover, they allow an automated learning process on simulation results. This double source of knowledge (expertise and learning process) allows an interesting way for knowledge extraction [15]. Indeed, an expert can easily provide a structure of problem (or parts of it) but with uncertain parameters values. Formulating this expert knowledge by means of an ID, some rules, based on probabilities, allow calculating estimated data. These data can be compared with the data resulting from simulation or from previous carried out projects. Considering the process of KDD, statistical extraction of structures is a much more complex problem than the statistical extraction of parameters values. If we already have a structure (resulting from expertise), calculations for parameters acquisition are simplified. This cycle of information extraction improves and facilitates the construction of the model by using the two sources of knowledge (i.e. expertise and results of simulation).

An Influence Diagram (ID) is a regular acyclic and no-forgetting graph of probabilistic relations between decisions, objectives, decision criteria and environment (see left part of figure 2), based on a net. Three kinds of nodes are used: "utility nodes", "decision nodes" and "chance nodes". "Utility nodes" do not have "children nodes". To each of them, an exact value is associated for each combination of "parent nodes". They represent in our case the objectives of the project. "Decision nodes" represent the possible decisions, i.e. the genes of EA. Decision rules enable to associate to each possible configuration of “parent nodes” a single decision. Lastly, the “chance nodes” are used in order to represent context and decision criteria. A table of conditional probabilities is associated to each node. It contains all the probabilities depending on states of “parent nodes”. The example on left part of figure 2 is linked to the scenario shown in figure 1. Decisions d5 and d6 are related to the options of realization (respectively Task T5 and T6) for a pen built in two parts. An expert determines that for these two decisions the principal criterion is the mode of realization (internal or external). This criterion is conditioned by the external supply (E2) of the required resources (for example a subcontracting supplier). Based on its experience, the expert estimates that these criteria can be aggregated in “Mode of realization” (C4). Once this analysis is carried out, we have a structure which could be completed by an estimated distribution of probabilities provided by the expert.

Nevertheless, a better way consists in setting these probabilities by confronting the ID with the real data coming from previous similar scenarios (e.g. a similar decision chain). This ID constitutes an “experience” in our system. This diagram ensures a conceptual classification of the properties of the decisions. The interest of conceptual classification is that it enables to define and classify the objects according to their descriptions (concepts used) without any considerations about raw data. The search for new projects or similar ones in different contexts can be done at conceptual level. This largely facilitates exchange with user and knowledge reusing.

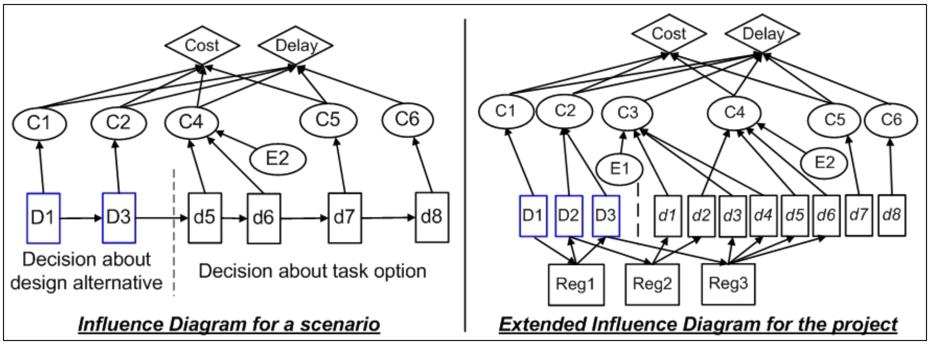
Cost
Delay
C1
C2
C4
C5
C6
E2
D1
D3
d5
d6
d7
d8
Decision about design alternative
Decision about task option
Influence Diagram for a scenario
Cost
Delay
C1
C2
C3
C4
C5
C6
E1
E2
D1
D2
D3
d1
d2
d3
d4
d5
d6
d7
d8
Reg1
Reg2
Reg3
Extended Influence Diagram for the project

Fig. 2. Influence Diagram for a scenario and for the project

The ID represented on left part of figure 2 results from a learning process, but for only one scenario. To obtain a global MoK able to guide the EA for all the possible scenarios, it is necessary to carry out a second cycle of abstraction by confronting the various ID. This global MoK is an extended influence diagram (on right part of figure 2) representing the set of possible decisions, criteria and sub-criteria linking decisions to objectives and environment. It is carried out by differential analysis of the contexts of each concept (objective, criterion or decision) used in the IDs. The context of a concept is defined as the set of concepts on which it is linked in the ID. For example, the context of a decision of realization associated with a task in a particular project (for example D1 on figure 2) is the set of the other decisions concerning the scenario, the criteria, functions and objectives related to this task. Mechanisms of context changing can be inferred each time that this task is used within a project.

These mechanisms activate or inhibit concepts associated with the task according to its context of use. The global ID is composed of three parts: in the higher part, a net of chance and utility nodes gathers the decision criteria and the objectives. It is the result of generalization of the selected decision criteria. It enables to represent all the combinations of relevant concepts. Then, the set of decisions to be evaluated are represented (D1,…, d8). It corresponds to the project's genome. Another set of decision nodes (Reg1 to Reg3) related to design alternatives is also represented. It corresponds to the structural aspect of knowledge resulting from project graph. This knowledge can be interpreted as activation constraints [16] and illustrates the fact that selection of an alternative activates corresponding tasks and inhibits tasks associated with other possible alternatives. For example, decision D3 activates d3 and d4 or d5 and d6.

4 Different Way to Use knowledge in Evolutionary Algorithm

Within our model of knowledge, some information links the search space (parameters to be selected), the objectives of project and the space of global context. So, in order to accelerate the exploration process within the EA, we use the capitalized knowledge (gathered in the MoK) adapting it to the current case. Considering that the MoK can be unsuitable or incomplete in certain cases and that it is not revised during search process, it is thus necessary to preserve independence of search method when predictions of MoK are not appropriated. For this reason, evaluation and selection steps of classic EA are preserved. Moreover, a mechanism is envisaged allowing to partially come back to traditional genetics operator when insufficient progress is observed.

The main problem for direct interaction between ID and EA is the step of inference in probability model especially because of multi-criteria fitness in ID consumes too much computing time. As a consequence, knowledge should be clusterised [2] [11] with respect to objectives and provided to EA during initialization. This way developed in next section allows general orientation to reach interesting zones of search space. The main difference between this study and the others using a MoK to guide search [10] [11] to guide search is that the MoK is more complex here (multi objectives, interdependencies between variables) but it is made before optimization process, as an off-line process.

Interaction based on knowledge clustering. This kind of interaction is based on two classifications: i) the classification of individuals with respect to objectives by EA and ii) the conceptual classification of scenarios by project's global ID. During initialization step, the global I.D. provides classes of favorable genes for each class of objectives. The classes of objectives are distributed uniformly on search space and their number is fixed by the user. A Class of genes gathers probabilities of selection for each alternative of each gene of the genome.

In a traditional EA, initial population is generated randomly. Here, the initial population is build according to the classes of genes, in order to start search procedure with a priori good orientations. The individuals are divided through the various classes of genes. Then for each individual, the probabilities of his class are used to fix the value of genes as shown on figure 3. In the class of gene, each table corresponds to the selection's probabilities for each state of a gene. For example, the values (1; 0) of the first table indicate that, to reach the matched class of objectives, it is necessary to choose with a probability of 100% the first alternative. This choice implies that genes 3, 6, 7, 8 and 9 (respectively linked to decision D3, d3, d4, d5 and d6 on figure 2) are inhibited according to structural knowledge (their probability on the class of genes is fixed to -1). When a gene is inhibited by previous genes already instantiated (genes encoding OR node's options choice), its value is fixed to 1.

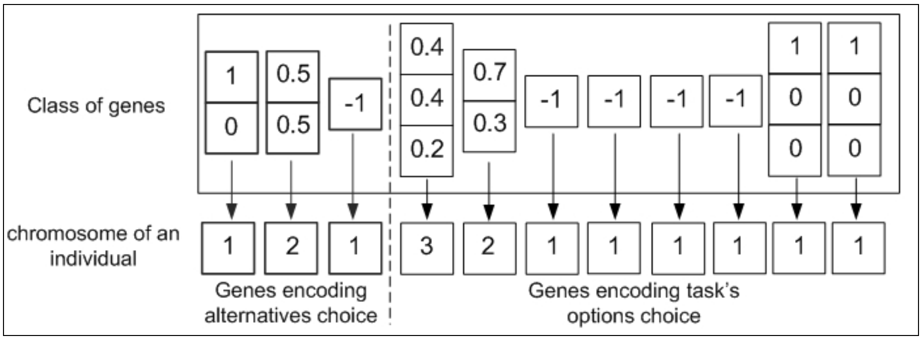
Class of genes
chromosome of an individual
1 0
0.5 0.5
-1
0.4 0.4 0.2
0.7 0.3
-1
-1
-1
-1
1 0 0
1 0 0
1 2 1 3 2 1 1 1 1 1 1
Genes encoding alternatives choice
Genes encoding task's options choice

Fig. 3. Utilization of gene's class for individual building during initialization

During EA process, classes of genes are matched to current cluster of Pareto-optimal individuals (see left part of figure 4). The centered solution of cluster (i.e. which minimizes distance with other solutions) will be used as reference point for the class of genes to which it is matched. That makes it possible affecting to each individual the class of genes to which the center is closest.

The mutation operator, shown on the right part of figure 4, selects an individual randomly among the population and secondly, the probabilities of its class are used to fix the value of genes similarly as during initialization except that a gene mutation occurs according to the mutation probability and mutations on inhibited genes are skipped. This method allows to preserve good properties of an individual, to avoid useless changes and thus to concentrate the changes on the remaining genes.

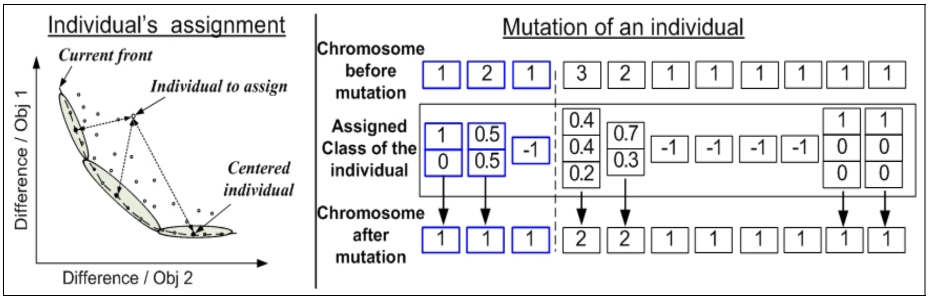

Fig. 4. Assignment of individual to classes and mutation operator

The crossover operator, illustrated on figure 5, enables exploration or intensification of search space. It corresponds then to an “inter-class” exchange by crossing individuals belonging to different classes or an “intra-class” exchange by crossing on the same class, according to the strategy of selection of parents. Once parent selection is done, probabilities of their classes are used to determine points of crossing. For each gene, if one of its parents has an inhibited gene or if the two parents have a probability of 1 for two different alternatives, the crossing does not take place for this gene. If only one of the two parents has a probability of 100% for an alternative, a unilateral crossing is done for this gene (from the parent with 100% towards the other as for first gene on figure 1). Lastly, if none of the preceding case is applicable, the crossing will be carried out in a traditional way according to the probability of crossing as for second gene on figure 1. This method makes it possible to preserve and, if possible, to exchange favorable genes of each individual.

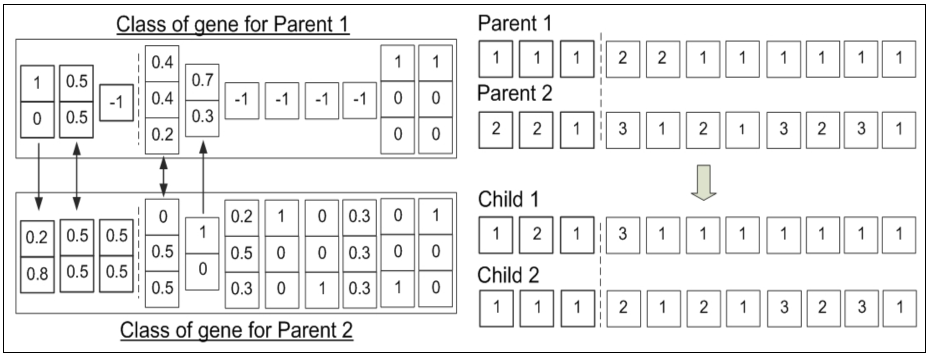

Fig. 5. Crossover operator

First results. We use a platform coded in C++ which allows testing our method. Table 1 presents preliminary results obtained for a graph with sixty tasks (three options per task) and fifteen OR node (three alternatives per OR node). The first row is the mean of best solution’s evaluation obtained after ten runs. The evaluation of an individual is calculated by adding the square of difference to each objective multiplies by a coefficient define by user (the optimal solution’s evaluation for this problem is 3051.28). The second row is the mean of generation’s number in which the best solution has been found and the last one is the time needed to achieved the ten runs on a AMD Sempron 3400+ with 512Mb of RAM.

For a first evaluation of this approach, the intelligent evolutionary algorithm (IEA) has been tested with a perfect learned MoK (class of genes learned with every best solutions generated separately) and without MoK (MoK with class equiprobable for every gene, equivalent to a traditional EA). Tests were realized for ten generations, with a population of fifty or ten individuals and with a genetic strategy (GS: P_{Cross}=0.7, P_{Mut}=0.3) or an evolutionary strategy (ES: P_{Cross}=0.3, P_{Mut}=0.7). The best strategy, here the ES one, is linked to problem instance. These first encouraging results show that the learned MoK guide very quickly the algorithm to interesting individuals as shown by runs done with ES strategy and fifty individuals where the

IEA find optimal solution at every execution in only five seconds. The learned MoK achieved best results (average of eight percents for all runs) to find the best solution and done as well for every class of objectives (the entire Pareto front is improved).

Table 1. Mean results after ten executions with and without learned MoK. The Eval. and Gen. columns represent respectively the best value obtained and the generation in which it added.

Gen/Pop/ Strat.	I.E.A. using learned MoK		I.E.A. without MoK		time
	Eval.	Gen.	Eval.	Gen.	
10/50/ GS	3074.27	1.66	3316.96	3.73	6s
10/50/ ES	3051.28 (optimal)	3.26	3274.85	4.9	5s
100/10/ ES	3051.28 (optimal)	6.4	3212.72	61.1	8s

5 Conclusion / Perspectives

An original approach based on coupling between a graphic model of knowledge built starting from experiments in an interactive and iterative way and a multi-objective evolutionary algorithm has been proposed in this paper. The first tests show promising results. We currently perform more experiments to test and validate the approach when MoK is incomplete or failing. Then some perspectives have to be carrying out. First of all, we investigate different way to use a direct interaction between MoK and EA without classification matching in order to use knowledge more specifically. Two possibilities are now investigated. The first one consists in compiling all information with an automaton as proposed by J. Amilastre in [17]. This model has very short response times but must be built before launching scenario's search process. The second possibility is to simplify the global ID. This operation can be carried out by means of an adaptive meta-model representing the relevance of a node as proposed M. Crampes in [18]. Secondly, another perspective concerns the update of the global ID starting from the past experiments (i.e. starting from the set of all the MoK corresponding to all previous instantiated scenarios). It has to be specified as well as the interaction between models and decision makers.

References

1. Talbi, D.: Application of Optimization Techniques to Parameter Set-up of Industrial Scheduling Software. Computers In Industry 55, n°2, 105–124 (2005)
2. Michalski, R.S., Wojtusiak, J., Kaufman, K.A.: Intelligent Optimization via Learnable Evolution Model. In: 18th IEEE Conf. on Tools with Artificial Intelligence, pp. 332–335 (2006)
3. Russel, S., Norvig, P.: Artificial Intelligence: A modern approach, 2nd edn. Prentice Hall/Pearson education international, London (2003)
4. Huyet, A.-L., Paris, J.-L.: Synergy between Evolutionary Optimization and Induction Graphs Learning for Simulated Manufacturing Systems. International Journal of Production Research 42(20), 4295–4313 (2004)
5. Sebag, M., Schoenauer, M.: A rule based similarity measure, vol. 837, pp. 119–130. Springer, Heidelberg (1994)

6. Cervone, K.: Experimental Validations of the Learnable Evolution Model. Congress on Evolutionary Computation, San Diego CA (2000)
7. Alami, J., El imrani, A.: A multipopulation cultural algorithm using fuzzy clustering. Applied Soft Computing 7(2), 506–519 (2007)
8. Chung, C.J.: Knowledge based approaches to self adaptation in cultural algorithms, PhD thesis, Wayne State University, Detroit, USA (1997)
9. Chebel-Morello, B., Lereno, E., Baptiste, P.: A New Algorithm to Select Learning Examples from Learning Data. In: Leung, K.-S., Chan, L., Meng, H. (eds.) IDEAL 2000. LNCS, vol. 1983, pp. 13–15. Springer, Heidelberg (2000)
10. Larrañaga, P., Lozano, J.A.: Estimation of Distribution Algorithms. A new Tool for Evolutionary Computation. Kluwer, Dordrecht (2001)
11. Miquélez, M., Bengoetxea, E., Larrañaga, P.: Evolutionary computation based on Bayesian classifiers. Int. Journal AMCS 14(3), 335–349 (2004)
12. Baron, C., Rochet, S., Esteve, D.: GESOS: a multi-objective genetic tool for project management considering technical and non-technical constraints, Art. Intel. Applications and Innovations (AIAI), IFIP World Computer Congress, Toulouse (2004)
13. Zitzler, E., Thiele, L.: Multi objective evolutionary algorithms: a comparative case study and the strength Pareto approach. IEEE Trans. on evolutionary computation 3, n°4, 257–271 (1999)
14. Watthayu, W., Peng, Y.: A Bayesian network based framework for multi-criteria decision making. In: MCDM 2004, Whistler, Canada (2004)
15. Becker, A., Naim, P.: Les Réseaux Bayésien: modèles graphique de connaissance, Eyrolles (1999)
16. Vareilles, E., Aldanondo, M.: Evaluation of a Solution in Interactive Aiding Design Process. In: INCOM 2006, Saint Etienne, France (2006)
17. Crampes, M.: Méta modèle adaptatif de la pertinence d'un modèle de connaissance. In: RFIA 2004, Toulouse (2004)
18. Amilastre, J., Fargier, H., Marquis, P.: Consistency restoration and explanations in dynamic CSPs – Application to configuration, vol. 135(1-2), pp. 199–234. Elsevier A.I, Amsterdam (2001)