A Priori Knowledge Integration in Evolutionary Optimization

Paul Pitiot, Thierry Coudert, Laurent Geneste, Claude Baron

A Priori Knowledge Integration in Evolutionary Optimization

Paul Pitiot[1,2], Thierry Coudert[1], Laurent Geneste[1], and Claude Baron[3]

[1] Laboratoire Génie de Production, Ecole Nationale d'Ingénieurs de Tarbes,
47, av. d'Azereix BP 1629 - 65016 TARBES, France
{paul.pitiot,thierry.coudert,laurent.geneste}@enit.fr
[2] Centre de Génie Industriel, Ecole des Mines d'Albi, Université de Toulouse
Campus Jarlard, 81013 Albi CT CEDEX 09
paul.pitiot@mines-albi.fr
[3] LATTIS, INSA de Toulouse, 135, av. de Rangueil - 31077 TOULOUSE, France
claude.baron@insa-toulouse.fr

Abstract. Several recent works have examined the effectiveness of using knowledge models to guide search algorithms in high dimensional spaces. It seems that it may be a promising way to tackle some difficult problem. The aim of such methods is to reach good solutions using simultaneously evolutionary search and knowledge guidance. The idea proposed in this paper is to use a bayesian network in order to store and apply the knowledge model and, as a consequence, to accelerate the search process. A traditional evolutionary algorithm is modified in order to allow the reuse of the capitalized knowledge. The approach has been applied to a problem of selection of project scenarios in a multi-objective context. A preliminary version of this method was presented at EA' 07 conference [1]. An experimentation platform has been developed to validate the approach and to study different modes of knowledge injection. The obtained experimental results are presented.

Keywords: Project management, product preliminary design, guided evolutionary algorithm, experience feedback, bayesian network.

1 Introduction

Many companies, in order to meet the requirements of their clients and to provide them with adequate products, implement two key processes:

- the "product design or configuration" process, which aims at defining precisely the architecture of the product and its components,
- the "project design or configuration" process which aims at specifying how the product will be realized (sequence of tasks, used resources...).

These two processes are often implemented sequentially: first the product is designed then the realization project is elaborated. For example, when a client wants to build a house, the architect designs at first a plan of the house, then the corresponding realization project is developed and launched. Since the project constraints (for example delays) are not explicitly taken into account in the product design, this can lead to additional iterations between “product design” and “project design” processes. A better integration (or coupling) of both processes is therefore a way to improve the global performance of companies. An in-depth study of several mechanisms that can facilitate integration has been launched in a project called ATLAS, funded by the French National Research Agency and involving academic laboratories, industrialists and the competitiveness cluster Aerospace Valley. The work presented in this paper takes place in the context of the ATLAS project.

In this paper, a simplified integration product / project model is first proposed. Indeed, in both environments (product and project), design processes are achieved according to a hierarchical decomposition (see Figure 1(a)):

- products are recursively decomposed into smaller sub-products (“AND” connectors), eg. product P_1 is made of P_{11} and P_{12} (yellow cloud on figure 1.a represent the fact that to make P_1, “P_{11} and P_{12}” are needed and this “global” task will be decomposed on the next analysis level illustrated underneath),
- accordingly, projects are recursively decomposed into sub-projects,
- alternatives (“XOR” connectors) can be defined in products (e.g. choice between components) and in projects (e.g. choice between sub-contractors to achieve a task).

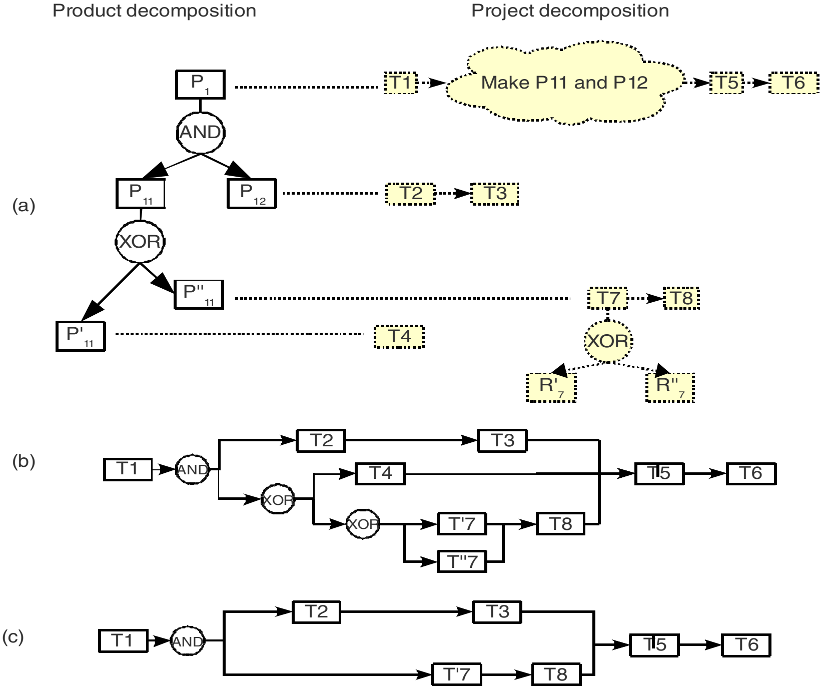

Fig. 1. Product / Project decomposition

In order to represent the links between both hierarchies, an integrated model is used. This model consists of a graph which nodes are: tasks of the project, AND nodes and XOR nodes. Figure 1(b) represents such a model of the example given in Figure 1(a). A "scenario", corresponds to a graph in which all the choices are made (i.e. with no more XOR nodes). An example of scenario, corresponding to the model in Figure 1(b), is illustrated in Figure 1(c). The problem addressed is to find, among all the possible scenarios, an optimal one with respect to multiple criteria (such as weight of the product, delay of the project, cost of both, etc.). In this paper, two objectives are considered: minimise the project delay (time needed for the execution of a scenario (due date of the final task) and minimise cost of the project (sum of the cost of every selected task in the scenario)).

Let us point out that this problem can be considered as an extended product configuration problem. The existing literature on the subject is dedicated to finding a feasible configuration according to constraints and knowledge on the domain. However, as mentioned in [2] it is very difficult to optimize the resulting configured product a problem of combinatorial explosion appears especially when the problem is loosely constrained. In this case, using an optimization approach can help to focus on good solutions. In [3] a search method, based on a classical multiobjective evolutionary algorithm, was proposed for the problem of scenario selection with promising results. In this paper, we propose to improve this method by taking into account the knowledge that can be capitalized from previous optimizations (learning from exp erience). The background of our work with regard to existing approaches that mix learning and search is given in section 2. Then, the proposed approach, based on an hybridation between bayesian networks (for learning) and evolutionary algorithms (for searching) is described in section 3. Finally, the obtained results are discussed in section 4.

2 Background

The method proposed in this paper is close to a new family of algorithms called "intelligent" or "guided" evolutionary optimization [4][5]. This kind of algorithms is based on the interaction between a search process and a knowledge extraction process achieved by a learning procedure. The goal is to merge advantages of each approach. The search process aims at improving a set of solutions by selection and combination operations. The learning process goal is to extract, to capitalize and to exploit knowledge contained into the solutions in order to guide the search process. The learning process has to give orientations with respect to a given context. Michalski in [4] shows that fixing some interesting solutions properties is enough for the search method to generate very quickly some solutions close to the optimal one.

As a possible search process, EA are well suited for the coupling with learning methods. Indeed, in a multi-criteria search context, they provide the learning algorithm with a set of individuals that "represent" the global space search. This kind of method indirectly reuses knowledge associated with the problem via the evaluation of the generated solutions. But this knowledge, used during search, is not preserved from one execution to another. In order to do so, it is necessary to complement the EA with a model adapted to knowledge capitalisation and reuse.

Among the different methods coupling optimisation and learning, Bayesian Optimization Algorithms (BOA) uses Bayesian Networks (BN) as a Model of Knowledge (MoK) [6]. In these methods, MoK is learned from a database containing selected individuals from the previous generation (according to their fitness). Then, from the MoK, a sampling procedure is used to generate directly the new population of individuals. The induction of the probability model, especially parameters interaction (i.e. definition of the network structure), constitutes the hardest task to perform [7]. Therefore, classical BOA learning process limits itself to the study of most influent parameters interaction.

The use of prior knowledge allows either to speed up algorithm convergence by introducing some high-quality or partial available solutions [8], or to improve the learning procedure using an available structural knowledge (prior probabilities of networks structure [7][8][9]). The model proposed in this paper (section 3) acquires prior knowledge about the whole structure of the network from an expert. Then, the learning achieved during optimization process concerns only probabilities updating. This method make it possible to use a MoK with every parameter and main interactions always represented (given by experts) and then to concentrate the learning effort to the probabilities estimation. The hypothesis used is that experts provide a structure close to optimal one, which is enough, after a quick probabilities learning, to guide efficiently the EA.

For the majority of guided evolutionary methods listed above, the use of knowledge is achieved indirectly. Knowledge is represented by means of classes of operators [10], intervals [4], and assumptions on the parameters values or by the attributes about good solutions [11]. A.L. Huyet in [5] proposes to model directly the knowledge using classes of parameters. The problem is that it is nearly impossible to directly handle this knowledge with the used formalisms (e.g. decision trees or neural networks). Furthermore, there is no model that dissociates objectives in order to have a representation of the influence of solutions on each of them. Objectives are generally aggregated and then, partial knowledge is impossible to reuse. In the different encountered approaches, the two processes (search and learning) have few interactions during execution, especially for the crossover operator. The model proposed in next section gives some answers to the issues listed above.

3 Proposed Framework and Algorithm

The proposed framework uses a hybrid method that makes interacting an EA for the multi-criteria search process and a Model of Knowledge (MoK) able to provide orientations adapted to the treated case. The Bayesian Network (BN) formalism is used for the knowledge base. Two sources of knowledge are used for supplying this base: on one hand, the case base that contains a selection of individuals (solutions) provided by the EA and, on the other hand, the expert knowledge base used in order to define the structure of the BN. The resulting BN provides probabilities that EA can use as orientations for guiding its search process. These orientations are taken into account directly by the evolutionary operators. Using the case base, a learning step enables the BN to be updated by means of an inference algorithm. BN inference algorithms are time consuming even if they are only used to compute probabilities. Therefore, the knowledge is clusterised [4] with respect to objectives, represented by discrete nodes in the BN. So, in the proposed approach, objectives are represented as discrete nodes (the values of an objective are represented as discrete states – e.g. Low, Medium, and High). A class of objectives is defined as the combination of different objective states. It corresponds to a region of the objective space. In a multi-criteria decision making process, the method has to provide decision makers with a set of solutions belonging to the Pareto front. A good quality of this set is obtained when all the classes of objectives corresponding to the Pareto front have at least one solution. So, the proposed method enables to guide the EA to reach, at each generation, an ideal Pareto front or, more exactly, interesting zones of search space represented by the different classes of objectives (see Figure 3).

MoK acquisition. The structure of the model of knowledge is built from expert knowledge. As illustrated in Figure 2, it contains 4 kinds of nodes: objectives, decision, concepts and environment nodes. The decision nodes correspond to the XOR connectors of the model. The objective nodes represent the set of objectives used for optimization. The concepts nodes are used by experts to express which characteristics of the domain are important and discriminatory on one or several objectives. Environment nodes enable to contextualize the knowledge contained into concept nodes. The whole structure is organised in an heterarchical oriented network from decisions to the objectives nodes, established by experts. Then probabilities of the BN are inferred from some representative cases using EM[1] algorithm.

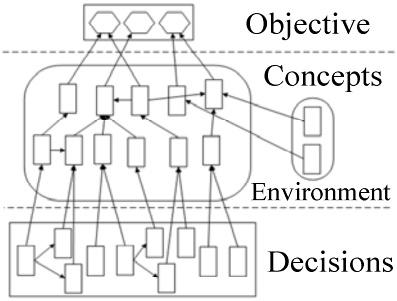

Fig. 2. Decision analysis and capitalization in global MoK

MoK actualization. Considering that, in certain cases, MoK can be unsuitable or incomplete, it is thus necessary to preserve the independence of the search method when the predictions of the MoK are not appropriate. For this reason, the evaluation and selection steps of a standard EA, developed in the next section, are preserved. Moreover, when insufficient progress is observed, two alternatives are implemented and tested: 1) a probability smoothing mechanism allowing to progressively come back to traditional genetic operators, 2) a MoK updating by an online parametric learning.

[1] The algorithm EM (Expectation - Maximization) is used for learning. EM algorithm is used considering an industrial implementation perspective because of abilities for dealing with missing or partial data.

Individual representation. In the model, first proposed by [3], an individual represents one scenario for the project (see Figure 1.c). The chromosome of an individual gathers on the first part the XOR nodes derived from Product decomposition (choice between components). Instantiations of the genes of this first part (selection of a state) lead to inhibition of some others gene in the chromosome. On the second part of the chromosome, genes represent the XOR nodes derived for Project decomposition (choices to achieve tasks). All choices are always represented whereas the majority of them are inactive since they are inhibited by choices realized on genes of the first part. This encoding ensures a constant viability of the solutions.

Selection, evaluation and archiving. The search algorithm is adapted from a SPEA method (Strength Pareto Evolutionary Algorithm) proposed in [12]. It is a traditional EA with classical steps: initialization, evaluation, selection, crossover and mutation operators. SPEA ensures the multi-objective evaluation of individuals according to two steps: i) the Taguchi approach is used in order to evaluate the cost of a scenario for each criterion; ii) then, the multi-criteria evaluation is achieved by means of Pareto front in order to compare and classify the scenarios. The probability of selection of an individual is proportional to its performance (fitness). This fitness depends on the position of the individual compared to the Pareto front.

New evolutionary algorithm. The modified EA is represented in figure 3 with the three new evolutionary operators. During the "loading step", the objective classes are built with respect to the BN. In order to reinforce the main characteristics of each class, probabilities superior to 0.95 are set to 1 and probabilities inferior to 0.05 are set to 0. When a gene is inhibited by a previous gene instantiation (probability of 1 for a particular state of a gene in the first part of the chromosome), the value -1 appears in the class of objective indicating its inhibition. This inhibition mechanism represent so called structural knowledge. It is represented by lists of inhibited gene for each genes instantiation, computed before optimisation process. This behaviour is called *Kstruct.*

The initial population is built according to the objective classes in order to start the search procedure with a priori good orientations (KO-initialisation on figure 3). The individuals are distributed through the various objective classes. Then, for each individual, the probabilities of its class are used to fix the value of genes as shown on figure 3. A gene can be inhibited when operators are used on the chromosome. Indeed evolutionary operators are applied progressively on the chromosome from the first part to the second. Therefore, it is possible to use the structural knowledge in order to add the value -1 to inhibited genes after each instantiation of a gene in first part. In other hand, when a gene is inhibited, either it is kept as it is, or it can be modified by a classical evolutionary process (mutation and crossover). This second possibility enables a genetic mixing. This behaviour is called *diploid knowledge preservation (mode Diplo).*

During the EA process, for each generation, all the individuals have to be associated to an objective class. So, the objective classes are matched to current cluster of Pareto-optimal individuals. The central solution of the cluster (i.e. which minimizes Euclidian distance with other solutions) is used as a reference point for the objective class to which it is matched. It makes it possible to assign to each individual the class of objective to which the centre is closest.

The *Knowledge Oriented mutation operator (KO-mutation)*, selects an individual randomly among the population and secondly, the probabilities of its class are used to fix the value of genes as during initialization except that a gene mutation occurs according to the probability of mutation. If the diploid knowledge preservation is not used, the mutation is performed randomly and uniformly on the inhibited genes.

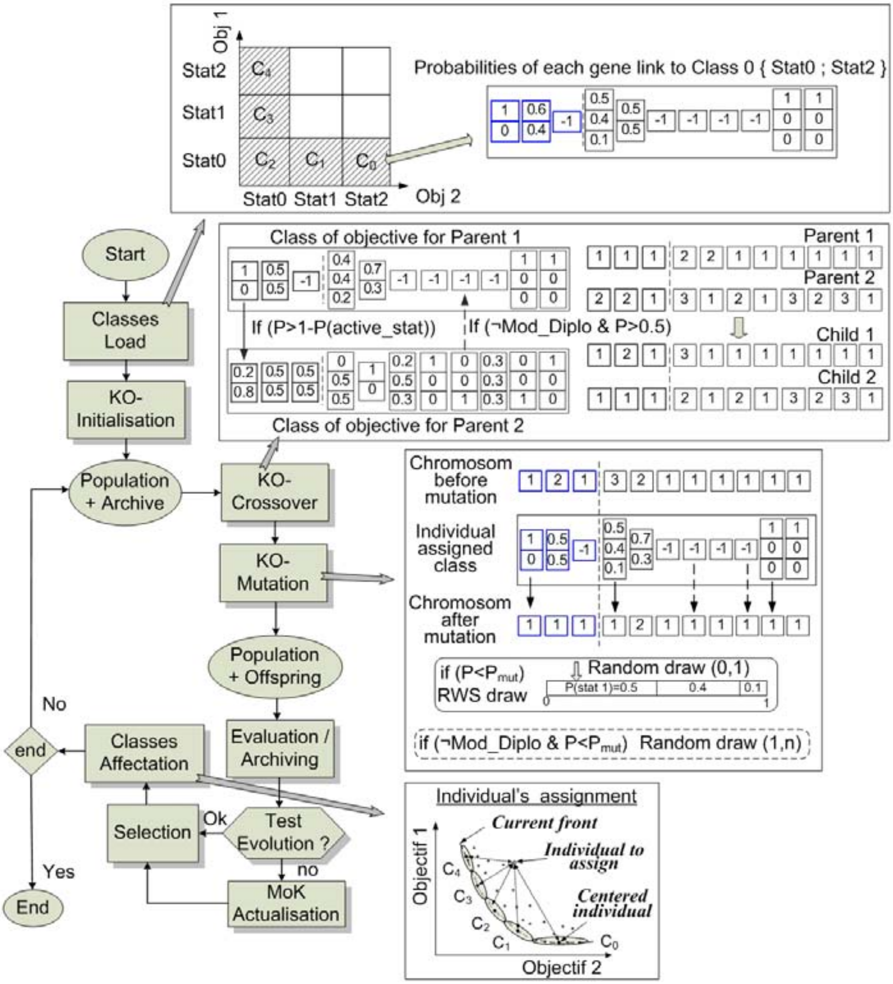

Fig. 3. Initialisation, mutation operator and crossover operator

The *Knowledge Oriented crossover operator (KO-crossover)*, enables exploration or intensification of the search space. It corresponds to an "inter-class" exchange by crossing individuals belonging to different classes or to an "intra-class" exchange by crossing individuals of the same class, according to the selection strategy of the parents. Once parent selection is done, probabilities of their classes are used to determine the points of crossover. The crossover is performed in a specific manner for each individual (unilateral crossover). For each gene, the probability of crossover is equal to 1 minus the probability given by the class of the active individual. This method makes it possible to preserve and, if possible to exchange, favourable genes of each individual. When the value linked to a gene in the corresponding objective class is -1 (inhibited gene), a unilateral crossover is done with a probability of 0.5 if the Diplo mode is inactive (uniform crossover of inhibited genes). If the Diplo mode is active, inhibited genes are preserved from the evolutionary process.

4 Experimentation and Validation

Main contribution of this study concerns a priori knowledge use by three forms: 1) a conceptual dependency structure between parameters expressed by a B.N., 2) probabilities of this model stem from analysis of previous plans and, 3) explicit structural knowledge (inhibitions between genes stem from graph shape). To evaluate the use of each type of knowledge, the behaviour of three algorithms is studied:

- Classical EA (without a priori knowledge) is ran with equiprobable objective classes (each state has an equivalent probability to be mute or crossed). The features in this case are the inhibition mechanism and crossover strategies.
- Evolutionary Algorithm Oriented by Knowledge using on line learning (noted $\mathrm{EAOK_X}$), the network structure is defined at the beginning of optimization while probabilities tables, initially uniforms, are learned every X generations,
- EAOK guided by an exact model (noted $\mathrm{EAOK_{init}}$), structure and probabilities are learned using a sample of optimal solutions previously generated with an exact approach for small instances or resulting from previous runs of $\mathrm{EAOK_X}$.

Experimentation has been planned following two steps. In the first step, algorithm is confronted to problems with limited size (different graph shape with 35 to 90 task nodes and 10 to 40 XOR nodes). This first step allows checking the general behaviour of the algorithm as well as tuning of multiple parameters (evolutionary parameters, crossover strategies, learning parameters, using of structural knowledge and diploid knowledge preservation). On a second phase, the behaviour of the proposed algorithm is studied on a large project (approximately hundred XOR nodes).

Figure 4 and tables 1, 2 and 3 introduce first tests results on different small projects (35 task nodes randomly generated, 12 XOR nodes for the figure 4 for example). The first curve of the figure 4 illustrates the average performance of the population of individuals obtained with modes EA, $\mathrm{EAOK_{init}}$, $\mathrm{EAOK_1}$ and $\mathrm{EAOK_5}$. Second curve illustrates the average performance of the individuals of the Pareto front. Each curve represents average values obtained after one hundred executions. The setting of evolutionary parameter is linked to the graph shape especially the number of XOR nodes. They are experimentally tuned with the classical EA mode for each graph then used with others modes ($\mathrm{EAOK_{init}}$ and $\mathrm{EAOK_X}$).

$\mathrm{EAOK_{init}}$ shows good performances. After initialisation, individuals of the population are 25% better than those obtained with EA. These results outcome from different combination of others parameters (crossover strategies, knowledge use, etc.) This explains the important standard deviation, but ratio between EA and $\mathrm{EAOK_{init}}$ is constant for equivalent setting. Initial gap between EA and $\mathrm{EAOK_{init}}$ corresponds to the direct impact of injection during initialization. This gap varies according to MoK quality and complexity of solved problem. On last generation, the gap between EA and EAOK$_{init}$ is about 16% with a Relative Standard Deviation (RSD) of 30% less for individuals of current population. Population generated by the guided EAOK is always improved in comparison with classical EA, because the MoK leads to a concentration of the population within performing areas. The final Pareto-optimal individuals mean fitness is improved 4.82% (RSD 30% less) on twentieth generation. EA performance meets EAOK$_{init}$ ones very progressively, according to problem complexity (number of parameters and complexity of injected knowledge) and according to evolutionary parameters setting.

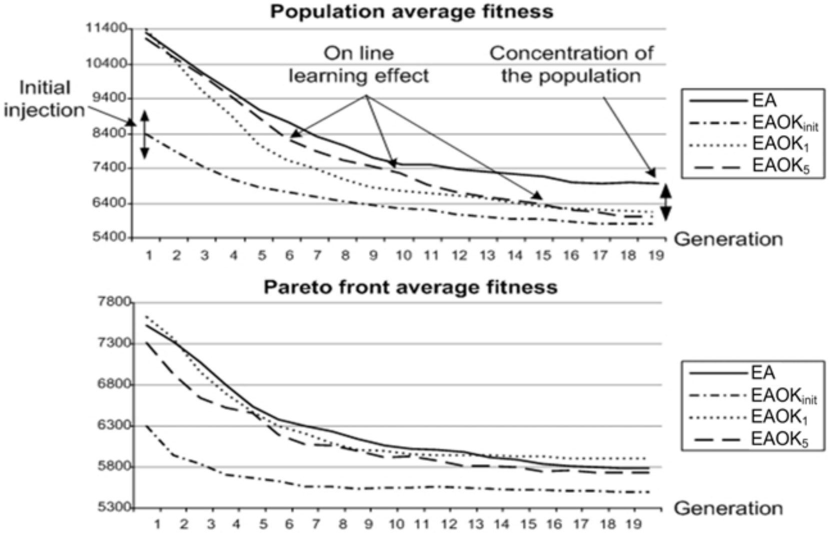

Fig. 4. Population and Pareto front average fitness. Test realized for twenty generations, with a population of thirty individuals, a maximum of nine individual in Pareto front, $P_{mut} = 0.5$, $P_{cross}=0.5$ and five classes of objectives.

Table 1. Values corresponding to curves showed on figure 4. The first line of table 1 gives results obtained by EA mode (value and relative standard deviation (RSD) for the hundred executions, while values of other lines are expressed as a percentage compared to the EA.

Mode	Mean Fitness for entire population				Mean Fitness for Pareto individuals			
	Generation 0		Generation 19		Generation 0		Generation 19	
	Value	RSD	Value	RSD	Value	RSD	Value	RSD
EA	11289	4.4	6961	16.5	7521	13.9	5781.4	7.5
EAOK$_{init}$	25.52 %	3.9	16.46 %	11.8	16.17 %	10.3	4.82 %	4.9
EAOK$_1$	-0.56 %	4.4	11.93 %	27	-1.37 %	13.7	-2.06 %	10.9
EAOK$_5$	1.28 %	4.4	13.65 %	30	0.83 %	14.5	0.90 %	10.6

Figure 4 presents first tests for on line learning algorithms. They are equivalent to EA at the beginning of optimisation process (uniform probabilities distribution). They deviate from EA after every learning phase. Learning effect is particularly visible in mode $EAOK_5$ with three zones where difference with EA is intensified (generations 5, 10 and 15). At beginning of process, mode $EAOK_1$ is more performing than $EAOK_5$, but the difference is progressively reduced and finally $EAOK_5$ gives better results. Indeed, it gives degrees of freedom to the search process in order to refine individuals between each learning phase. For the mean of final Pareto-optimal individual fitness, $EAOK_1$ has the worst performances. Indeed, when individuals selected for learning are not enough diversified, the guiding tends to limit search around the existing individuals. $EAOK_5$ take advantage of search and guiding combined effects. This performance has been improved by regulating learning parameters.

Concerning adjustment of learning algorithm, two important characteristics emerge: the quality of cases used for learning and the parameter settings of learning algorithm. After various tests, a very fast learning[2] has been chosen because it is sufficient to make emerging main properties of search space and thus obtain a global guidance. Quality of the available learning cases set seems to be the most important characteristic in order to obtain a correct model. When the number of cases per class of objectives is too restricted, the phenomenon of "over-learning" involves search stagnation around the already founded individuals, with a risk of stagnation in local minima. Thus, the progressive smoothing of MoK probabilities can be used and provides two functions: i) it makes possible to limit over-learning when cases provided to the learning are too similar; ii) it constitutes a mean for gradually giving degrees of freedom to the search process, i.e., for release the guiding by the MoK.

Crossover strategies are also preliminary evaluated. The exploratory strategy during the whole optimization process gives the better results, so it have been selected form the following tests. Finally, every combination[3] of Structural Knowledge (SK) and Diploid Knowledge Preservation (DKP) has been evaluated. The results are presented in table 2 and concern hundred executions of each mode on a project of fifty tasks nodes. Structural knowledge can be used to indirectly manage knowledge contained in the individuals. If it allows an initial improvement of the EA, it also involves a reduction of the genetic diversity by reducing exchanges between the individuals. On the other hand, the use of structural knowledge with a learned MoK allows using only individual specific information among knowledge contained in his corresponding class. The diploid knowledge preservation mode gives good results only when the individuals have already a good level of performance, by preserving the inactive combinations which can be re-used if the corresponding genes were reactivated. Conversely, the best strategy with reliable information ($EAOK_{init}$) is to use neither structural knowledge, nor diploid knowledge preservation. Guidance by the model is then complete, but this strategy has not to be maintained because stagnation risks increase (strict guiding towards existing individuals).

Our method has finally been tested on a problem with large size (350 tasks nodes and more than hundred XOR nodes in the project graph). The project graph is obtained by gathering five small projects previously used. An exact algorithm is not suitable in such large project. Individuals used for the construction of complete model ($EAOK_{init}$) are obtained by collecting individuals obtained during one execution of the

[2] Stop criterion of algorithm EM: 1% of log-likelihood minimal improvement.

[3] Note that in EA mode, DKP is completely linked to structural knowledge activation, while in other modes, genes could be inactivated by learned knowledge.

Table 2. Average fitness of individuals of the Pareto front at the beginning (generation 0 to 2), in progress (generation 10 to 12) and at the end of the optimisation, with various indicators allowing to evaluate more precisely the Pareto front quality: relative standard deviation of the average fitness of Pareto front individuals (PD), RSD of distance between two consecutive individuals (DI), overall length of the Pareto front (Lg) and number of individuals of the Pareto front (Nb). The last column presents the average fitness of the best final individual.

Mode	DKP / KS	Average fitness of Pareto front individual								PD	DI	Lg	Nb	Best
		0	1	2	10	11	12	18	19					
EAOK$_{init}$	1 / 0	1667	**1465**	1445	1359	1348	1344	1360	1369	0,08	0,20	34	**8,6**	**656**
	1 / 1	1706	1545	1465	1384	1390	1400	1399	1396	**0,07**	0,29	32	8	659
	0 / 1	**1665**	1527	1492	1356	1357	1355	1379	1383	0,08	0,24	33	8,3	657
	0 / 0	1716	1484	**1409**	**1326**	**1321**	**1336**	**1353**	**1355**	0,08	**0,20**	**34**	8,6	658
EA	- / 0	2322	2215	2032	1541	**1522**	1518	**1451**	**1439**	0,1	0,26	31	**6,9**	**665**
	1 / 1	**2309**	2148	**1925**	**1537**	1528	**1510**	1464	1464	0,11	**0,24**	30	6,9	674
	0 / 1	2377	**2083**	1984	1667	1627	1584	1461	1459	0,1	0,25	**31**	6,7	674
EAOK$_{10}$	1 / 0	2505	2255	2032	1570	1498	1483	1369	1363	0,07	0,26	32	7,4	669
	1 / 1	2324	2104	2008	**1468**	**1421**	**1419**	**1364**	**1357**	0,08	0,26	31	7,1	**661**
	0 / 1	**2270**	**2091**	**1889**	1562	1539	1475	1406	1399	0,1	**0,25**	31	**7,5**	669
	0 / 0	2380	2183	2095	1606	1545	1500	1398	1391	**0,07**	0,26	**32**	7,4	664

Table 3. The table below presents average values and associated RSD (for the twenty executions) for the performance of population individuals (Pop.), Pareto front individuals (Pareto) and best individual (best) at the end of optimization process, as well as qualitative indicators for Pareto front (PD, DI, Lg et Nb) and the execution time in second.

	Pop.		Pareto		PD	DI	Lg.	Nb.	best		time
	Val.	RSD	Val.	RSD					Val.	σ	
EA	11357	0.199	6557	0.24	0.09	0.18	11.2	3.95	5453	0.21	217
EAOK$_{10}$	7348	0.20	5688	0.14	0.07	0.16	9.4	4.1	4876	0.12	298
EAOK$_{init}$	5420	0.18	3601	0.17	0.11	0.21	9.5	3.6	2953	0.11	201

EAOK$_{10}$ (390 individuals). Table 3 presents the average of twenty executions of our algorithm (thirty generations of fifty individuals, P_{mut}= P_{cross}=0.5).

The EAOK$_{10}$ algorithm shows an interesting behaviour. The population is overall improved as well as individuals of the Pareto front. At last generation, the variation between EA and EAOK$_{10}$ respectively reaches 54% (population), 15% (Pareto front) and 11% (better individual) in favour of the EAOK$_{10}$. Moreover, these performances are more regular than with EA. The learning improves the results, especially the precision and reliability of optimisation. It also seems that the performances obtained strongly depend on the quality of research before the first learning. An interesting prospect is to use an adjustment of the EA supporting the diversity of individuals, in order to improve quality of individuals provided to the learning algorithm. However, in current version of the platform, the time of inference needed to update the probabilities classes remains important. The EAOK$_{10}$ requires indeed approximately 300 seconds to reach the thirtieth generation with two learning phase, so approximately 27% of additional time required compared to the EA.

5 Conclusion/Perspectives

Obtained results show the interest of different levels of knowledge reuse. When the knowledge contained in the model of knowledge is reliable, our method allows a significant improvement of performance. When the MoK is erroneous or incomplete, the tests realised on learning algorithm enabled us to study learning process abilities with suggested method. To validate our approach completely, it still remains to confront it with standard problems ("benchmarks"). However, tests carried out show the higher performances of our guided evolutionary algorithm compared to a traditional EA. Moreover, the advantages of our model relate a well guided and more efficient optimization than with classical EA, but also the possibility of knowledge capitalizing on the projects planned according to their context, as well as the possibility of providing to the expert the MoK used during optimization in addition to the optimized solutions. It is indeed useful, for the decision maker, to be able to consult a bayesian network, thanks to the tools offered by this formalism, and to directly visualize the influence of its future decisions on the objectives.

References

1. Pitiot, P., Coudert, T., Geneste, L., Baron, C.: Improvement of Intelligent Optimization by an experience feedback approach. In: Monmarché, N., Talbi, E.-G., Collet, P., Schoenauer, M., Lutton, E. (eds.) EA 2007. LNCS, vol. 4926, pp. 316–327. Springer, Heidelberg (2008)
2. Li, B., Chen, L., Huang, Z., Zhong, Y.: Product configuration optimization using a multiobjective GA. I.J. of Adv. Manufacturing Technology 30, 20–29 (2006)
3. Baron, C., Rochet, S., Esteve, D.: GESOS: a multi-objective genetic tool for project management considering technical and non-technical constraints. In: IFIP World Computer Congress on Art. Intel. Applications and Innovations, AIAI (2004)
4. Michalski, R.S., Wojtusiak, J., Kaufman, K.A.: Intelligent Optimization via Learnable Evolution Model. In: 18th Conf. on Tools with Artificial Intelligence, pp. 332–335 (2006)
5. Huyet, A.-L., Paris, J.-L.: Synergy between Evolutionary Optimization and Induction Graphs Learning for Simulated Manufacturing Systems. Inter. J. of Production Research 42(20), 4295–4313 (2004)
6. Pelikan, M., Sastry, K., Goldberg, D.E.: Sporadic model building for efficiency enhancement of the hBOA. Genetic Programming and Evolvable Machines 9, 53–84 (2008)
7. Baluja, S.: Using a priori knowledge to create probabilistic models for optimization. Inter. J. of approximate reasoning 31(3), 193–220 (2002)
8. Schwarz, J., Ocenasek, J.: A problem knowledge-based evolutionary algorithm KBOA for hypergraph bisectioning. In: 4th Joint Conf. on Knowledge-Based Software Engineering, pp. 51–58. IOS Press, Amsterdam (2000)
9. Hauschild, M.W., Pelikan, M., Sastry, K., Goldberg, D.E.: Using previous models to bias structural learning in the hierarchical BOA. In: Proceedings of the 10th annual conference on Genetic and evolutionary computation, pp. 415–422 (2008)
10. Sebag, M., Schoenauer, M.: A rule based similarity measure. In: Wess, S., Richter, M., Althoff, K.-D. (eds.) EWCBR 1993. LNCS, vol. 837, pp. 119–130. Springer, Heidelberg (1994)
11. Chung, C.J.: Knowledge based approaches to self adaptation in cultural algorithms. PhD thesis, Wayne State University, Detroit, USA (1997)
12. Zitzler, E., Thiele, L.: Multi objective EA: a comparative case study and the strength Pareto approach. IEEE Trans. on evolutionary computation 3(4), 257–271 (1999)