# Construction of a Cooperative Metaheuristic system based on Data Mining and Soft-Computing: Methodological issues

**J.M. Cadenas** **M.C. Garrido** **E. Muñoz**
Dpto. de Ingeniería de la Información y las Comunicaciones
Facultad de Informática. Universidad de Murcia. Spain
jcadenas@um.es carmengarrido@um.es enriquemuba@dif.um.es

## Abstract

Metaheuristics are effective strategies for solving optimization problems. However, when trying to solve an instance of this kind of problems it is hard to know which algorithm should be used. Hybrid systems provide flexible tools that can help to cope with this problem. Therefore a hybrid system based on the intelligent combination of different strategies will give more robustness and will allow to find higher quality solutions for different instance types.

**Keywords:** Metaheuristic, Cooperative System, Fuzzy Rules, Data Mining, Knowledge Extraction.

## 1 Introduction

Metaheuristics provide effective strategies for solving optimization problems. However, when we use them we can find algorithm selection problem, [13], which tries to decide which algorithm has to be used to solve an instance of a problem, trying to maximize a measure of performance. To cope with it we proposed to use a hybrid system that combines intelligently different strategies. Hybrid systems allow us to solve complex problems, very hard to solve using less tolerant approaches, but to obtain an "intelligent" combination of strategies that achieves good results for all type of instances and problems we need a tolerant approach, as the one provided by "Soft Computing". The problem is that this increase in tolerance may produce some precision loss, however it can be sacrificed in order to obtain a more robust system. The use of different strategies together with the methodologies provided by Soft Computing for building hybrid systems, will give us reasoning mechanisms and search methods which will allow us to combine domain knowledge with experimental data in order to obtain new computation tools to solve complex problems that are very difficult to solve with less tolerant approaches.

In this paper we show the different stages we have reached in the obtaining of this hybrid system from its definition to its final tuning.

## 2 Related Work

Several studies have shown that heuristics and metaheuristics are successful tools for providing reasonably good solutions (excellent in some cases) using a moderate number of resources. For that reason it will be interesting to use them in our hybrid system. There are mainly two fields that follow our approach, that is to obtain hybrid strategies which cooperate in a parallel way in order to solve a problem, and they are: *parallel metaheuristics* and *hybrid metaheuristics*.

Many efforts have been focused on these fields, and we can find different implementations. First appeared synchronous implementations, where information is shared regularly, such as [10]. More recently asynchronous implementations showed up, such as [8], and, according to the reports provided in [7], they ob-

L. Magdalena, M. Ojeda-Aciego, J.L. Verdegay (eds): Proceedings of IPMU'08, pp. 1246–1253
Torremolinos (Málaga), June 22–27, 2008

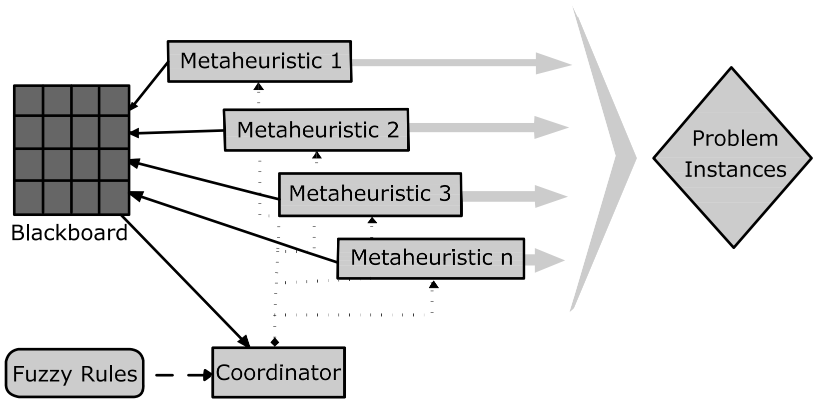


Figure 1: Multiagent Metaheuristic System

tain better results than synchronous. It has also been pointed out that these approaches obtain better results than independent methods, but previous studies, [8], show that if the access to shared information is not restricted they can experiment premature convergence problems. This seems to be owe to the stabilization of the shared information produced as a result of the intense exchange of the best solutions. Trying to cope with this problem in [12] is proposed a cooperative strategy that uses memory to control this effect. Here a coordinating agent, modeled by a set of rules defined by the user, monitors a set of solver agents, that implement the same metaheuristic, and sends orders to them about how they have to continue.

It has also been noticed that those strategies based on an unique metaheuristic do not cover all the possibilities, thus we find two interesting challenges:

- To find ways of controlling the information exchange.
- To combine different metaheuristics.

With our hybrid system we try to face both of them and propose a structure similar to the one in [12, 6] where a coordinator modeled by a set of rules gets information about the performance of the different metaheuristics and sends orders to them. The main differences are that it combines a set of different metaheuristics and that the rules are obtained as the result of a knowledge extraction process.

# 3 Model and Design of the System

In [1, 2] we proposed the seed of this system, which is based on a multiagent system that can be seen on fig. 1. In the system each metaheuristic is implemented by an agent that tries to solve the problem while cooperates with the rest of metaheuristics. To control this cooperation we propose the use of a coordinating agent which will monitor and modify the behaviour of the agents, having two fundamental tasks: to gather information on the performance of each of the metaheuristics and to send orders to modify their search behaviour.

To perform the communication among the different metaheuristics we use an adapted blackboard model. In this model each agent controls a part of the blackboard where periodically writes the best solution it has found. The coordinator then, consults the blackboard in order to monitor the behaviour of each metaheuristic, and decides which actions have to be taken to improve the performance.

The problem that arises immediately is how to describe the coordinator in such a way that it can modify the behaviour of the metaheuristics efficiently. The solution we propose is to give intelligence to the coordinator using a set of fuzzy rules, since they allow to represent data in a very similar way to human reasoning and allow to incorporate expert knowledge as ad hoc rules. Finally the intelligence will arise as the result of a knowledge extraction process from which we will obtain the set of fuzzy rules.

The knowledge extraction process, [1, 2], is supervised and divided in three phases. It starts with Data Preparation phase, where a database which contains useful information for data mining is obtained. Next, Data Mining phase is applied, and a model of the coordinator of the system is obtained using data mining techniques. And the process ends with the Model Evaluation phase, where the efficiency of the set of fuzzy rules that model the coordinator is tested.

## 4 A First Approach

In [4] we applied this process for the first time trying to obtain a system to solve Knapsack Problem. That way, we built two prototypes, the first coordinates three metaheuristics, a Genetic Algorithm, a Tabu Search and a Simulated Annealing, and the second adds Ant Colony to the previous three. With this aim we applied the knowledge discovery process trying to obtain fuzzy rules to model the following behaviours:

- When and how has to be changed the solution of a metaheuristic using the solution of another one having a better performance.
- What is the set of parameters that has to be used to initiate a metaheuristic, depending on the instance that has to be solved.
- When and how have to be changed the parameters of a metaheuristic which is performing worse than the rest.
- Which rule has to be selected if more than one has been activated.

Some examples of the rules that we obtained can be found on table 1. The first rule illustrates how the solution of a metaheuristic (in this case, Simulated Annealing) is changed for the solution of another with a better performance (in this case the Genetic Algorithm), here if its time to exchange solutions, the difficulty of the instance being solved is $difficult$, and the difference of the objective function between these metaheuristics is large enough, then the solution is exchanged. The second one shows an example of how the parameters are modified. Here the parameter cross probability of the genetic algorithm is changed if the difference between its objective function and the one obtained by the Tabu Search is *small* or *large*, it is time to change parameters and the difficulty of the instance being solved is *very easy*.

Table 1: Example rules

| |
|---|
| `IF [Time IS` *Restart* `Y Difficulty IS` $Difficult$ `AND p_gen_ann IS (`*large* `OR` *verylarge*`)] THEN Restart Simulated Annealing with the solution of the Genetic Algorithm.` |
| `IF [p_gen_tab IS (`*small* `OR` *large*`) AND Time IS` $CP$ `AND Difficulty IS` *veryeasy*`] THEN pcross IS` *high* |
| .......... |

These first fuzzy rules, in spite of being perfectly valid as they come from data, are too many and not abstract enough. For that reason we decided to create a more general set of rules.

## 5 Making the System More General

As we said before, the rules obtained during the Data Mining phase are not desirable, therefore a more general template of rules was created, [4], mixing the previous rules and the behaviour expected from the coordinator. These template has to be filled using data mining and it is composed of the following two rules:

- IF [($weight_1 * d_1$ OR ... OR $weight_n * d_n$) IS *enough*] THEN change the current solution of the worst metaheuristic.
- IF [($weight_1 * d_1$ OR ... OR $weight_n * d_n$) IS *high* AND (time IS $THigh$ OR $TVeryHigh$)] THEN *changeParameters* of Metaheuristic.

where:

- $d_n$ is the difference between the benefit obtained by the metaheuristic $n$ and the one that is being studied divided by the best.
- the weights are obtained during the knowledge extraction process.
- *Enough* is a fuzzy set with trapezoidal membership function defined as follows:

$$\mu(x, a, b, c, d) =$$
$$\begin{cases} 0 & x \leq a \ or \ x \geq d \\ (x-a)/(b-a) & x \in (a, b] \\ (d-x)/(d-c) & x \in [c, d) \\ 1 & x \in [b, c] \end{cases}$$

  where $a$, $b$, $c$, $d$ are 0.005, 0.01, 1, 1 respectively. This membership function, as the rest of membership functions described, was determined by trial and error.
- $High$ is a fuzzy set with trapezoidal membership function where $a$, $b$, $c$, $d$ are 0.05, 0.1, 1, 1, respectively.
- $THigh$ is a fuzzy set with trapezoidal membership function where $a$, $b$, $c$, $d$ are 0.4, 0.5, 0.7, 0.8, respectively.
- $TVeryHigh$ is a fuzzy set with trapezoidal membership function where $a$, $b$, $c$, $d$ are 0.7, 0.8, 1, 1, respectively.
- $ChangeParameters$ is a function that changes the parameters of a metaheuristic for a new set of parameters which showed good performance during the process of knowledge extraction.
- A rule is considered to be fired if its activation is bigger than an $\alpha - cut$, that can be configured.

The first rule tries to indicate how the position in the search space of the metaheuristic that is obtaining the worst result can be changed for a position nearer to another metaheuristic with a better behaviour. In order to change the solution of the worst metaheuristic, we can take into account the following situations:

- The metaheuristic that receives the solution is based on trajectories. The best solution obtained among the metaheuristics that have fired the rule is sent to it.

- The metaheuristic that receives the solution is based on populations. We can find different options:

  - ◇ The metaheuristic that sends the solution is trajectory based. It has to send a set of solutions consisting of solution solutions near its best solution.
  - ◇ The metaheuristic that sends the solution is population based. It has to send a set of solutions consisting of the best members of its population.
  - ◇ Different metaheuristics have to send solutions. They have to send a set of solutions where each metaheuristic choose its solutions as said before and they are combined paying attention to their weights.

On the other hand, the second rule shows how the parameters of the different metaheuristics have to be modified. That way if a metaheuristic is obtaining solutions with a benefit smaller than the rest, and it has been a long time since its parameters have been changed, then we can change its parameters for a new set using the rules obtained in the Data Mining phase.

## 6 Improving the System

### 6.1 Improving the Adaptation to Different Kinds of Instances

With this model of the system we performed some tests that can be seen on [3, 5]. But this system was generated using basic data mining and obtained static results, for that reason we propose the use of fuzzy decision trees to obtain the weights and the parameters of the different metaheuristics. In order to do that we apply the knowledge extraction process to obtain a set of fuzzy decision trees using FID 3.4 [9]. These trees will try to ascertain two

Table 2: Comparisons cooperation vs no-cooperation

| Type | Coo2Met | Coo3Met | NoCoo2Met | NoCoo3Met |
|---|---|---|---|---|
| unc span | 0.0051 | 0.0005 | 0.0092 | 0.0021 |
| wea span | 0.0045 | 0.0043 | 0.0082 | 0.0082 |
| str span | 0.0064 | 0.0062 | 0.0148 | 0.0148 |
| pceil | 0.0007 | 0.0006 | 0.0009 | 0.0008 |
| circle | 0.0407 | 0.0330 | 0.0978 | 0.0496 |
| **Items** | | | | |
| 500 | 0.0034 | 0.0031 | 0.0039 | 0.0160 |
| 1000 | 0.0102 | 0.0078 | 0.0312 | 0.0159 |
| 1500 | 0.0145 | 0.0116 | 0.0313 | 0.0158 |
| 2000 | 0.0177 | 0.0132 | 0.0312 | 0.0159 |

things: first a tree has to decide which algorithm will be better to solve the current instance and assign weights accordingly. Second a set of trees, one for each metaheuristic, will decide which set of parameters is the best for solving the current instance and will provide an order over the different sets. That way, the best set of parameters is used as the initial set and, if it is necessary, the parameters are changed using the ordered list.

### 6.2 Applying a Feedback Process

Once we have a model for the system it also is important to perform a process of system evaluation and system feedback. After applying this process we have noticed that fuzzy rules give us flexibility to adapt the system to the changing conditions of the problems, because of the fact that the system is partially controlled by fuzzy sets, and thus we can obtain, for each $\alpha - cut$, a different set of rules, which provide better results depending on instance type. So we propose to apply a process of data mining to obtain a fuzzy decision tree that will indicate us which $\alpha - cut$ is better for the instance being solved.

## 7 Results

In this section we present some examples of the results obtained by the different approaches commented on this paper. In order to compare the systems we decided that each one had to be executed during 120 seconds stopping each 6 seconds to execute the coordination. Each instance was solved 10 times, and the results show the averages. All tests were executed on an Intel core2 Quad 1.66Ghz with 2GB of Memory.

The systems were implemented synchronously, that is, every communication is carried out at a given moment, previously specified. At this moment each metaheuristic writes its current solution and the coordinator checks which actions have to be performed. In order to obtain the fuzzy engine used to model the coordinator we used the tool XFuzzy 3.0 [11]. With this tool we modeled the different rules and obtained a fuzzy engine that was finally slightly modified to obtain an appropriate engine to our purpose.

In order to model the systems, we applied the knowledge extraction process with the following parameters. In Data Preparation phase we solved 2000 instances using these algorithms with different parameter configurations. In Data Mining phase we used a fuzzy decision tree, FID 3.4 [9] to fill the template, obtaining different sets of parameters for each metaheuristic.

In table 2 we can see the results of two prototypes of the system for solving knapsack problem, the first prototype coordinates two metaheuristics, a Genetic Algorithm and a Tabu Search, and the second adds Simulated Annealing to the former two. In this table we compare the two prototypes with two systems which combine the same metaheuristics, but where no cooperation was applied. In the first part of the table the average error ratio is shown for each type of instance, and in the second for each instance size.

In table 3 we show the results obtained by

Table 3: Comparisons of nature inspired systems

| Type | Coord Syst | No Coop Syst |
|---|---|---|
| unc span | 0.0000 | 0.0000 |
| wea span | 0.7659 | 0.8729 |
| str span | 1.2819 | 1.2951 |
| pceil | 0.1353 | 0.1386 |
| circle | 3.1250 | 3.2639 |
| **Items** | | |
| 500 | 0.9292 | 0.9711 |
| 1000 | 1.0758 | 1.1095 |
| 1500 | 1.1095 | 1.1821 |
| 2000 | 1.1319 | 1.2138 |

Table 4: Comparisons changing $\alpha - cut$

| Type | Crisp results | $\alpha$=0.75 results | best results | $\alpha \rightarrow$ best result |
|---|---|---|---|---|
| unc span | 0.463 | 0.431 | 0.33 | 0.6 |
| wea span | 2.647 | 2.606 | 2.556 | 0.7 |
| str span | 0.913 | 0.899 | 0.899 | 0.75 |
| pceil | 0.114 | 0.116 | 0.115 | 0.9 |
| circle | 7.821 | 7.893 | 7.724 | 0.8 |
| **Items** | | | | |
| 500 | 2.238 | 2.144 | 2.102 | 0.5 |
| 1000 | 2.745 | 2.833 | 2.711 | 0.8 |
| 1500 | 3.205 | 3.25 | 3.22 | 0.9 |
| 2000 | 3.771 | 3.719 | 3.541 | 0.8 |

a prototype of the system based only on nature inspired metaheuristic for solving knapsack problem. This prototype used two metaheuristics, an Ant Colony and a Genetic Algorithm and was compared with a system with the same metaheuristics and without cooperation. Once again, in the first part of the table the average error ratio is shown for each type of instance, and in the second for each instance size.

Finally, in order to show the new lines we are exploring, in table 4 we show some results about the changes in the performance that can be obtained by changing $\alpha - cut$. In this table we show the results obtained by a crisp system, the $\alpha - cut$ we usually use and the $\alpha - cut$ that is obtaining the best results.

## 8 Summary

In this paper we have reviewed the development of a hybrid, centralized, cooperative system for solving optimization problems showing the changes we have done in order to improve it.

The system is based on a multiagent system, where each metaheuristic is an agent that has to solve the problem while cooperates with the rest. In order to accomplish this coordination we use a coordinating agent which controls and modifies the behaviour of the agents during their execution. To add intelligence to the coordinator we propose to apply a knowledge extraction process to obtain a set of fuzzy rules, that will allow the coordinator to:

- Change the position in the search space of a metaheuristic which is obtaining poor results for another position near to the position of a metaheuristic with a better performance.

- Change the parameters of a metaheuristic intelligently if it persists in having bad results.

It is important to outline that if we increment the number of metaheuristics cooperating the knowledge extraction process will take more time, but the performance of the system will not decrease, because each metaheuristic is independent from the others. Only the coordinator will be a bit slower, however, its execution time compared with the execution time of the metaheuristics is negligible.

Our first idea was to blindly apply the knowledge extraction process. However, after using this approach we observed that the rules obtained were too numerous and not abstract enough, therefore we proposed a more general model based on a template of fuzzy rules. This template is filled using data mining. In a first approach we used basic data mining and obtained static information. For that reason, in order to improve system adaptability, we propose to substitute the former model by a model based on fuzzy decision trees. To finish after developing the final model it is interesting to apply a feedback process to improve the performance of the model, for instance obtaining a tree to decide which $\alpha - cut$ is better suited for the instance being solved.

The system has only been applied to a very simple problem, knapsack problem, which is considered one of the "easiest" NP problems. For that reason we are going to apply this process to more complex problems.

It has also been noticed that the cost of the knowledge extraction process can be too large. To cope with this problem we propose two approaches:

- To use Active Learning to reduce the time expended in Data Preparation phase, as with this technique we can reduce the instances that need to be solved to generate the database.
- To use a system base on Online Learning. That way we will have an initial system with a "bad" performance that will be improving as it solves more instances.

## Acknowledgements

The authors thank the "Ministerio de Educación y Ciencia" of Spain and the "Fondo Europeo de Desarrollo Regional" (FEDER) for the support given to this work under the project TIN2005-08404-C04-02. And the "Fundación Séneca, Agencia de Ciencia y Tecnología de la Región de Murcia", which give support under the "Programa de Ayudas a Grupos de Excelencia de la Región de Murcia" (Plan Regional de Ciencia y Tecnología 2007/2010).

## References

[1] J.M. Cadenas, M.C. Garrido, L.D. Hernández, E. Muñoz, *Towards a definition of a Data Mining process based on Fuzzy Sets for Cooperative Metaheuristic systems.* International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, IPMU2006, 2828–2835, Pars, 2006.

[2] J.M. Cadenas, R.A. Daz-Valladares, M.C. Garrido, L.D. Hernndez, E. Serrano, *Modelado del Coordinador de un sistema Meta-Heurstico Cooperativo mediante SoftComputing.* Campus Multidisciplinar en Percepcin e Inteligencia (CMPI 2006), Albacete, España, 679–689, 2006.

[3] J.M. Cadenas , M.C. Garrido, E. Muñoz, *A Cooperative System of Metaheuristics.* 7th International Conference on Hybrid Intelligent Systems, HIS 2007, Kaiserslautern - Germany, 120–125, 2007.

[4] J.M. Cadenas, M.C. Garrido, V. Liern, E. Muoz, E. Serrano, *Un prototipo del coordinador de un sistema metaheurstico cooperativo para el problema de la Mochila.* Congreso Nacional sobre Metaheursticas, Algoritmos Evolutivos y Bioinspirados (MAEB 2007), Tenerife, Espaa, 811–818, 2007.

[5] J.M. Cadenas, M.C. Garrido, E. Muoz, *A Hybrid System of Nature Inspired Metaheuristics.* Workshop on Nature Inspired Cooperative Strategies for Optimization (NISCO 2007), Acireale, Italy, 1–10, 2007.

[6] J.M. Cadenas, M.C. Garrido, E. Muoz, C. Cruz-Corona, D. Pelta, J.L. Verdegay, *Different Approaches for Cooperation with Metaheuristics.* Encyclopedia of Artificial Intelligence (pendiente de publicacin), Idea Group Reference, 2007.

[7] T. G. Crainic, M. Toulouse, M., *Parallel Strategies for Metaheuristics*, Handbook of Metaheuristics, Kluwer Academic Publisher. 2003.

[8] T. G. Crainic, M. Gendreau, P. Hansen, N. Mladenovic, *Cooperative parallel variable neighborhood search for the p-median*, Journal of Heuristics 10: 293–314. 2004.

[9] C. Z. Janikow, *Fuzzy Decision Tree/Forest. FID3.4*, URL: http://www.cs.umsl.edu/∼janikow/fid.

[10] K. Lee, S. Lee, *Efficient parallelization of simulated annealing using multiple markov chains: an application to graph partitioning.* In Proceedings of the International Conference on Parallel Processing, 177–180, 1992.

[11] F.J. Moreno-Velo, I. Baturone, S. Sánchez-Solano, A. Barriga, *(2001) XFUZZY 3.0: A Development Environment for Fuzzy Systems*, International Conference in Fuzzy Logic and Technology, Leicester, 93–96, 2001.

[12] D. Pelta, C. Cruz, A. Sancho-Royo, J. L. Verdegay, *Using memory and fuzzy rules in a cooperative multi-thread strategy for optimization*, Information Sciences 176(13): 1849-1868, 2006.

[13] J. R. Rice, *The algorithm selection problem*, Advances in Computers 15:65–118, 1976.