A Cooperative System of Metaheuristics

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 mgarrido@dif.um.es enriquemuba@dif.um.es

Abstract

Hybrid systems give more flexible mechanisms for solving complex problems that can be very difficult to solve using less tolerant approaches. Therefore, a hybrid system will be the most suitable tool in order to cope with the algorithm-instance problem, which says that it is possible that an algorithm and its parameters that obtain good results for an instance of a problem, do not get the same results for another instance of the same problem. All this leads us to use different algorithms to solve combinatorial optimization problems within a single coordinated schema, that is a hybrid cooperative system of metaheuristics. In order to build this system we have proposed a methodology for the construction of a hybrid system, based on Data Mining and Soft Computing.

In order to test the usefulness of this methodology two hybrid systems based on a fuzzy model have been constructed to solve the knapsack problem. The first system coordinates two metaheuristics, a Genetic Algorithm and a Tabu Search. The second one adds a third metaheuristic, Simmulated Annealing, in order to check the robustness of the system and its capacity of obtaining higher quality solutions when a metaheuristic is added. Results obtained by this systems and a comparison with the ones obtained with individual metaheuristics are shown.

Keywords: Metaheuristic, Cooperative System, Fuzzy Rules, Data Mining.

1 Introduction

In optimization problems, generally, we search for the best solution (or one good enough) of a problem among a lot of alternatives. Optimization problems are common in our daily living. Every person constantly solves optimization problems such as finding the shortest way from home to work with traffic restrictions. Humans can find solutions to these problems efficiently because are small enough. However, these problems can become more complex, for example reducing the consumption of fuel of a fleet of plains trying to obtain more benefits. To tackle this kind of problems computational algorithms are required.

For that reason to solve these problems different strategies have been proposed, and one of the best are metaheuristics, which can be defined as [8]:

> *Intelligent strategies for designing and improving very general heuristic procedures with high performance.*

Metaheuristics provide effective strategies for finding approximate solutions to optimization problems. However, when we work with them it is easy to find the algorithm-instance problem or algorithm selection problem [10]. This problem tries to decide which algorithm has to be selected to solve a given instance trying to maximize a measure of performance. The most common approach is to measure the performance of a set of algorithms over a set of instances and use the one with the best performance. In any case, this problem is undecidable, but that does not deny the possibility of inducing a system that suggests an algorithm, or a set of algorithms, from theoretical and experimental evidence obtained from the application of a set of different algorithms to sets of instances with different characteristics.

An intelligent combination of different metaheuristics is expected to provide more efficient approaches and more flexibility when we try to solve these problems, obtaining robustness by using different metaheuristics which intercooperate, to give high quality solutions for different types of instance, [5]. But combining different metaheuristics intelligently requires sacrificing the precision in order to obtain more tolerance. And that constitutes the base of the methodology proposed by Soft Computing. The use of a set of metaheuristics along with the methodologies provided by Soft Computing for building hybrid systems will give us the reasoning mechanisms and the search methods which will allow us to combine the knowledge of the scope with the experimental data that we have in order to obtain new flexible computational tools. With these tools we can solve com-

plex problems that would be very difficult to solve with less tolerant approaches.

In this paper we show the construction of two prototypes of that system designed to solve the knapsack problem. These prototypes were designed following the methodological process proposed in [1, 2] for the design of a hybrid system that coordinates a set of metaheuristics. Thus, section 2 describes the structure and operation of the systems, section 3 shows the construction process, section 4 shows the results obtained by the prototypes constructed, and finally, section 5 presents our conclusions and future work.

2 A Cooperative Metaheuristic System

There exist many ways to carry out the cooperative execution of the different metaheurístics. The one that seems to be more appropriate to us is a multiagent system, in which each metaheuristic is an agent that has to solve the problem while coordinates itself with the rest of metaheuristics. In this schema, an approach to perform the cooperation is the use of a coordinating agent which will control and modify the behaviour of the agents, [3]. the coordinator will have two fundamental tasks: to gather information on the performance of each of the metaheuristics and to send orders to modify their search behaviour, [5, 3].

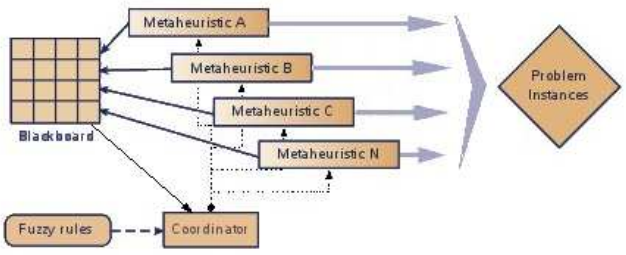

Figure 1. Multiagent Metaheuristic System

To perform the communication among the different metaheuristics we will use an adapted blackboard model. In this model each agent controls a part of the blackboard in which it can write the information about its performance, and periodically it writes the solution that it has found. The coordinator then, consults the blackboard in order to monitor the behaviour of each metaheuristic, and decides how it has to modify their behaviour.

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 and soften the problems showed before related with the behaviour of the metaheuristics when taken individually.

We propose to give intelligence to the coordinator using a set of fuzzy rules, since they allow to represent data in a way very similar to human reasoning, and that will do more comprehensible the system and besides will allow to incorporate easily expert knowledge into the system as ad hoc rules. Among the different types of fuzzy rules, we chose Mamdani [7] because they are easy to understand. The fuzzy rules will give intelligence to the coordinator and this will arise as a result of the methodological process proposed on [1, 2], which is based on a process of intelligent knowledge extraction (Fig. 2).

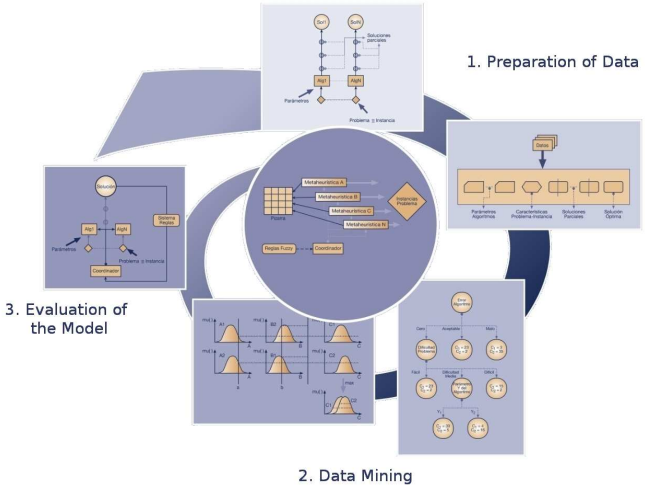

Figure 2. Knowledge Extraction Process

This process starts with the Data Preparation phase, where a database which contains useful information for data mining is obtained. Next, Data Mining phase is applied, in which the 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 .

3 System construction

3.1 The Knowledge Extraction Process

In this section we will briefly describe how has been applied each phase of the process in the construction of the two prototypes of the hybrid system which solve the knapsack problem. The first prototype (Coor2Met) coordinates two metaheuristics, namely, a Genetic Algorithm and a Tabu Search. The second prototype (Coor3Met) coordinates three metaheuristics, the previous two and a Simulated Annealing. The knowledge extraction process is the same in both cases, but for Coor3Met data about the new metaheuristic are added. For that reason we will explain the process only once. The details of this knowledge extration process are shown on [3].

As we said before, this process starts in the phase of Preparation of Data. The first thing we did in this phase was choosing the problem to solve, the knapsack problem. After that we selected the metaheuristics that were going to be used, we expected to use both metaheuristics based on trajectories and based on populations, because this is a very popular classification [4] and we wanted to study how the coordinator could manipulate each type. For that reason we selected a Tabu Search and a Simulated Annealing, as trajectory based metaheuristics, and a Genetic algorithm as a population based.

After that we solved a broad set of test instances (with an optimum solution known) of the knapsack problem, using the chosen metaheuristics and gathering any interesting information. With this information we created the database that was to be used in the data mining phase, but before that it was advisable to apply a preprocess to it in order to select those instances and attributes more relevant.

Next phase was Data Mining. In this phase we started choosing the data mining technique, finally selecting a fuzzy decision tree, FID 3.4, because it produces interpretable models and allows a suitable treatment of imperfect information.

After that, we applied FID to different subsets of the database trying to find rules of all kinds. As a result a wide set of trees was obtained, and from their interpretation we inferred a set of fuzzy rules which constituted the first model of the coordinator, examples of these rules can be found on table 1. These rules modeled 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.
- With which parameters has to be initiated 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 other.
- Which rule has to be selected if more than one has been activated.

However, these 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 which will be explained later.

The last phase of the process is the phase of System Evaluation, in this phase we test the efficiency of the model of the coordinator, and if it is performing efficiently with regard to computational cost and the solutions obtained. Moreover we proposed to test the robustness of the system when we add a new metaheuristic to it.

Table 1. Example rules

IF [Time IS $Restart$ Y Difficulty IS $Difficult$ AND p_gen_ann IS ($large$ O $verylarge$)] 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 $high$

IF [Time ES $Initial$] THEN pcross IS low

.....

3.2 Model of the Coordinator Obtained

As we said before, the rules obtained during the Data Mining phase seemed to be too many and not abstract enough to become the model of the coordinator. Therefore a more general set of rules was created obtained mixing the previous rules and the behaviour expected from the coordinator. That way we obtained two rules:

- IF [($weight_1 * d_1$ O ... O $weight_n * d_n$) IS *enough*] THEN change the current solution of the worst metaheuristic.
- IF Metaheuristic *isMuchWorseThanEveryone* Y *isMomentOfChangingParameters* THEN *changeParameters* of Metaheuristic.

The first rule tries to indicate how can be changed the position in the search space of the metaheuristic that is obtaining the worst result for a position nearer to another metaheuristic with a better behaviour.

With the experiments carried out we can observe how each metaheuristic behaves and taking this into account we can assign weights to each one of them. Thus, d_i is the difference between the benefit obtained by metaheuristic i and the obtained by the metaheuristic which has obtained the worst result. And the weights for each metaheuristic are 0.64 for the Genetic Algorithm and 0.36 for the Tabu Search in Coor2Met and 0.45 for the Genetic Algorithm, 0.3 for the Simulated Annealing and 0.25 for the Tabu Search in Coor3Met.

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. There are different options:
  - ◇ The metaheuristic that sends the solution is trajectory based. It has to send a set of solutions consisting of the replication of 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 has 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.

3.3 Prototypes

Finally we implemented the two prototypes (Coor2Met, Coor3Met) of the system as synchronous systems, that is, every communication is carried out at a given moment, previously specified. In 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 [9]. With this tool we modeled the different rules and obtained a fuzzy engine that was finally slightly modified to obtain an engine appropriate to our purpose.

4 Experimentation

In this section we will show the results obtained by the two prototypes, and also we will compare the two prototypes with the metaheuristics that form them, and one with each other.

4.1 Test Database

To carry out the tests we solved a database of instances composed of 80 instances in which changed both size (number of objects) and the way they were generated. Regarding size we can find four different sizes: 500, 1000, 1500 and 2000 objects. As for the way the instances are generated there exist five types:

- Spanner: These instances are constructed in such a way that all their items are multiple of a small set of items called key. That key con be generated using any distribution. In this case we consider three distributions:
  - Uncorrelated,
  - Weakly correlated,
  - Strongly correlated.
- Profit ceiling: In these instances all the benefits are multiple of a given parameter d.
- Circle: These instances are generated in such a way that the benefits are a function of the weights, having its graph an elliptic representation.

4.2 Methodology

In order to compare the different systems and metaheuristics we decided that each one of the hybrid systems has to be executed during 100 seconds. Similarly each individual metaheuristic was executed during 100 seconds using its best parameters.

4.3 Coor2Met Results

We will start showing the results obtained by Coor2Met, that coordinates a Genetic Algorithm and a Tabu Search. Table 2 shows the error ratio obtained in the resolution of the different types of instances both for Coor2Met and for the independent metaheuristics. 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 fig. 3 we show the graphs for these tables.

Table 2. Average error Coor2Met vs. individual metaheuristics

Type	Coor2Met	Gen. Alg.	Tabu S.
unc span	0,00506	0,08245	0,00930
wea span	0,00449	0,07497	0,00830
str span	0,00637	0,06492	0,01485
pceil	0,00070	0,05911	0,00095
circle	0,04068	0,09790	0,11748
Items			
500	0,00340	0,00406	0,02674
1000	0,01021	0,05689	0,03130
1500	0,01453	0,10405	0,03137
2000	0,01771	0,13848	0,03130

As we can see, the hybrid system always gets solutions at least equal to the best independent algorithm, and that is important because this was the minimum demanded from the system. But we can also observe how in most of the cases we obtain a markable improvement reducing error to a half. This improvement is especially important in circle instances, which are the most difficult to solve and in which we reduce the error from 10% to 4 %.

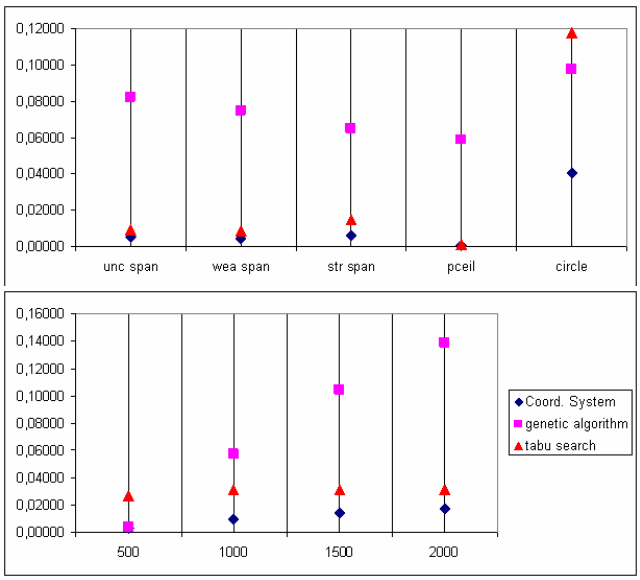

Figure 3. Coor2Met vs. individual metaheuristics

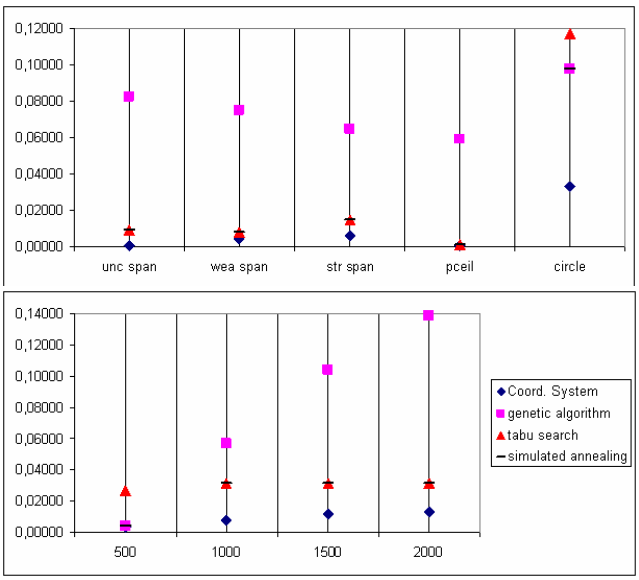

Figure 4. Coor2Met vs. individual metaheuristics

4.4 Coor3Met Results

Once we have seen Coor2Met results we show the results obtained by Coor3Met, that coordinates a Genetic Algorithm, a Tabu Search and a Simulated Annealing. In table 3 the error ratios of Coor3Met and the individual metaheuristics are shown. 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 fig. 4 we show the graphs for these tables.

Table 3. Average error Coor3Met vs. individual metaheuristics

Type	Coor3Met	Gen. Alg.	Tabu S.	Sim. Ann.
unc span	0,0005	0,0824	0,0092	0,0092
wea span	0,0043	0,0749	0,0082	0,0082
str span	0,0062	0,0648	0,0148	0,0148
pceil	0,0006	0,0590	0,0009	0,0009
circle	0,0330	0,0978	0,1174	0,0978
Items				
500	0,0031	0,0039	0,0266	0,0039
1000	0,0078	0,0568	0,0312	0,0312
1500	0,0116	0,1040	0,0313	0,0313
2000	0,0132	0,1384	0,0312	0,0312

As we can see, as Coor2Met this coordinator fulfills the requirement of obtaining at least the same results obtained by the individual metaheuristics. Furthermore it keeps obtaining a markable improvement in the results, so we can say that the system is robust when we add new metaheuristics since its results do not get worse, in fact they are improved, as we can see in fig. 5, that compares the two hybrid systems. In this figure we can see how as the instance size grows, and so the dificulty, the outperforming of Coor3Met over Coor2Met increases. On the other hand if we attend to instance type, differences are not so clear for every type, however, in circle instances, which are the most difficult to solve, and in uncorrelated spanner type, differences keep being noticeable.

5 Conclusions and Work to Develop

During the development of this paper we have shown the construction of Coor2Met and Coor3Met, two prototypes of a hybrid, centralized, cooperative system of metaheuristics for solving knapsack problem, as well as the results obtained of the application of them to a broad set of test instances.

The prototypes were built as a multiagent system, in which each metaheuristic is an agent that has to solve the problem while coordinates itself with the rest of metaheuristics. In order to accomplish this coordination we have used a coordinating agent which controls and modifies the behaviour of the agents during their execution.

This coordinator needs some kind of intelligence and to give this intelligence to it we have applied a knowledge extraction process. As a result of this process a broad set of rules was obtained, which finally was reduced to two high level rules, that use the rest when it is necessary. That way a reduction of the power of the system is avoided and it is easier to add new metaheuristics to the system. With these two rules the coordinator can:

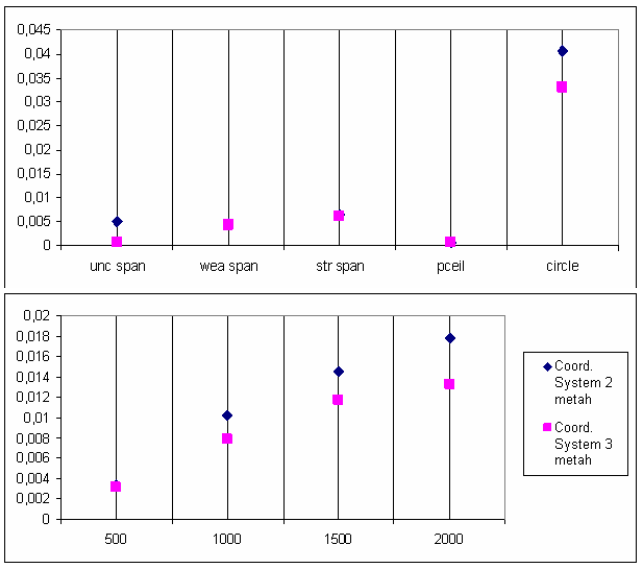

Figure 5. Coor2Met vs. Coor3Met

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

With the models of the coordinator we built the two prototypes, which were implemented as synchronous systems, that is, each metaheuristic has to write its parcial solutions at a given moment, previously specified. And in this moment the coordinator checks which actions have to be performed.

Finally these prototypes were used to solve a set of 80 instances which vary both in size and type, and compare them with the individual strategies and one with each other. After comparing them we could observe how in every case the hybrid systems obtained at least results as good as the obtained by the individual metaheuristics, obtaining in most cases better results, being the improvement more substantial as the difficulty of the instance is increased. In the comparison of the two prototypes we could observe how adding a new metaheuristic to the system does not damage the system, but improves its results, once again especially on instances of more difficulty.

Regarding future work we can outline that the systems constructed are very conservative, as those algorithms which had a better performance in experimentation always have priority over the rest, and only soft changes of parameters are performed. Therefore an immediate idea would be the construction of a more aggressive system. Similarly the system built is synchronous, and it would be interesting building an asynchronous version.

It will be also interesting performing researches about how can be exchanged information among the different metaheuristics and which information is more interesting. Finally we expect to apply this process to a more complex problem as the p-median.

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 Séneca”.

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, París, 2006.

[2] J.M. Cadenas, R.A. Díaz-Valladares, M.C. Garrido, L.D. Hernández, E. Serrano, *Modelado del Coordinador de un sistema Meta-Heurístico Cooperativo mediante SoftComputing*, Campus Multidisciplinar en Percepción e Inteligencia, CMPI2006, 679–689, Albacete, 2006.

[3] J.M. Cadenas, M.C. Garrido, V. Liern, E. Muñoz, E. Serrano, *Un prototipo del coordinador de un Sistema Metaheurístico Cooperativo para el Problema de la Mochila*, V congreso español sobre Metaheurísticas, Algoritmos Evolutivos y Bioinspirados, MAEB07, 811–818, Tenerife, 2007.

[4] D. Corne, M. Dorigo, and F. Glover, editors. *New Ideas in Optimization*. McGraw-Hill, 1999.

[5] C.A. Cruz, *Estratégias coordinadas paralelas basadas en soft-computing para la solución de problemas de optimización*, Tesis doctoral del Dpto. de Ciencias de la Computación e Inteligencia Artificial, Universidad de Granada, 2005.

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

[7] E. Mamdani, *Applications of fuzzy logic to approximate reasoning using linguistic synthesis*, IEEE transacions on computers 26(2):1182-1191, 1997.

[8] B. Melián, J.A. Moreno, J.M. Moreno, *Metaheurísticas: una visión global*, Revista Iberoamericana de Inteligencia Artificial 19:7–28, 2003.

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

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