Nature-Inspired Cooperative Strategies for Optimization

Juan R. González
University of Granada, Spain

Alejandro Sancho-Royo
University of Granada, Spain

David A. Pelta
University of Granada, Spain

Carlos Cruz
University of Granada, Spain

INTRODUCTION

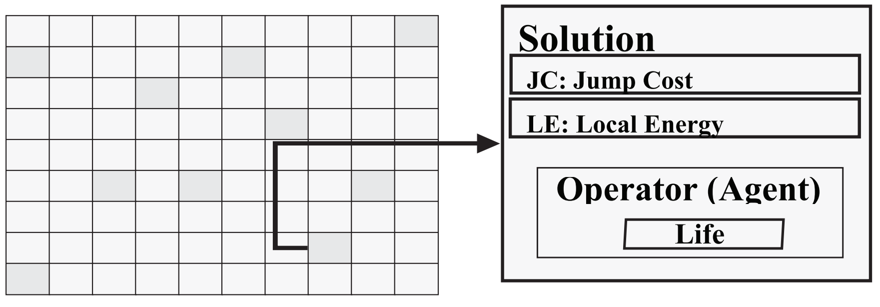
Biological entities (ranging from bacteria to humans) can engage in many and varied types of social interaction, from altruistic cooperation to open conflict. A paradigmatic case of social interaction is cooperative problem solving (CPS), where a group of autonomous entities work together to achieve a common goal. For example, we might find a group of people working together to move a heavy object, play a symphony, choose a business strategy, or write a joint paper. CPS has been studied by researchers from a variety of areas such as distributed A. I., soft computing, economics, philosophy, organization science, and the social and natural sciences among others.

In the context of optimization problems, this situation can be seen as follows: the goal is to find the "best" solution for the problem at hand, while the "entities" can be thought as optimization algorithms. The idea of having a portfolio of solving strategies and use all of them in a parallel and coordinated fashion to solve the problem is justified because of the following fact: no algorithm outperforms another one in all circumstances. Besides, experiences in the field of optimization show that problem instances can be grouped in classes and there exists an algorithm for each class that solves the problems of that class most efficiently.

In order to assess the benefits of a proposed cooperation strategy in the context of optimization, we should compare it against other ones over a set of test problems (more specifically a set of instances of each particular problem). This task is far from trivial, time consuming, and requires a great amount of programming (design, implementation, and test) effort. In this situation, it would be of great help to have tools that allow us, easily, to get a feeling about the potential of the strategy proposed before approaching its full implementation, comparison, and assessment.

The main aim of this contribution will be to show how to use a problem-independent framework to simulate

Figure 1. Scheme of the model showing the information stored in each object; gray cells have an operator inside



the main general features of any optimization problem; and to use this framework together with a simulation tool as a test bed to evaluate the behavior of cooperative strategies. In particular, we will focus on cooperation models among unrelated individuals.

The modelization of the (nature inspired) cooperation strategies may be done through IF-THEN rules. Due to the inherent uncertainty of the natural processes, it seems to be more adequate to use fuzzy rules to obtain a better representation of the situation. Furthermore, the cooperation mechanism can be seen as a decentralized control problem, and fuzzy rules have been successfully applied in control contexts. That's why we will use fuzzy rules in this proposal.

To achieve the objectives proposed, the entry is organized as follows: "The framework" section describes the problem- independent framework to model optimization problems. "The cooperation models" section focuses on the topic of cooperation among unrelated individuals and the corresponding representation using fuzzy rules. Finally, "The simulation" section shows the simulation's results obtained using the freely distributed NetLogo simulation tool. "Conclusion" section finalizes with the conclusions achieved.

THE FRAMEWORK

Every optimization problem has, at least, the following two elements: the set S of feasible solutions and the cost function $f(x)$ where $x \in S$. In general, the search for a solution $x^* \in S$ such that $f(x)^*$ is the maximum / minimum value is hard because of the existence of local optima, so a successful search strategy should provide mechanisms to avoid getting trapped into them.

In our model, we depart from a matrix or lattice of solutions with dimensions $N \times M$ where each cell (i, j) is occupied by a solution $x \in S$. The only information available for a solution is its current cost. It is important to note that the matrix does not capture any metric or topological feature of the solutions; it acts as a repository of solutions. Besides solutions, our model has a set P of agents. Each agent can be associated with a modification or move operator $o \in P$ and, essentially, is a function $S \to S$. We start with k agents that are also placed in the matrix and $k << N \times M$. A graphical representation of the model is shown in Figure 1.

When an agent "falls" onto the cell (x, y), it modifies the solution stored in such position. Truly, this modification is done over the cost of the solution as follows: each agent has certain amount of *life* available and solutions have (besides the cost) two additional values: *JC* (jump cost), the energy needed to positively change its cost, and *LE* (local energy) the current energy stored. The essential procedure says that if an agent takes a solution where $LE > JC$, then it will change positively its cost, and it will gain some *life*. If $LE < JC$, then, the agent will change negatively the cost of the solution but it will left part of its *life* in *LE*. The amounts of energy gained/lost from the solutions/agents and the cost variations are governed by fuzzy rules that will be described later.

Once the solution is changed, the agent moves to another position (randomly or based on some information from the neighbors). The agents' *life* is also used to manage its survival and reproduction chances. So we have in this model, two sets or populations that are jointly evolved: (1) solutions, which are optimized in terms of their costs, and (2) agents, which are evolved in a "Darwinian" sense (survival of the fittest).

In order to avoid any specific relation to a concrete problem, we don't use any mention to the objective function. To do that, we have substituted the calculation of the objective function f with another one $h(x)$ that represents the energy needed to escape from a local optimum of cost x. In this way, we can:

- Avoid thinking on a specific optimization problem
- Manipulate the hardness of the model through different definitions of the function h

Now, the question is how h looks like given that the distribution of the local optima is not known. As a first approximation we assume that it is harder to escape from local optima whose values are close to the global optimum, than to escape from those that are far (in value) from this reference point. In this work, we suppose that the set of local optima are finite from 1 to 100, and we sort it in such a way that the worst is the first and the global optimum is 100. Then, for each possible value $x \in \{1,...,100\}$, we calculate the amount of energy needed to escape from that point as:

$$h(x) = \left(\frac{x}{20}\right)^4 \left| \sin\left(\frac{80}{(\frac{3}{2}x)^{0.99} - 100}\right) \right| \quad (1)$$