INTERNATIONAL CONGRESS ON COMPUTATIONAL BIOENGINEERING
M. Doblaré , M. Cerrolaza and H. Rodrigues (Eds.)
©I3A, España, 2003

ARTIFICIAL LIFE TOURNAMENTS: ENCOURAGING CREATIVITY

Diego Milone, Daniel Beber, José Biurrun

Facultad de Ingeniería
Universidad Nacional de Entre Ríos, Argentina.

SUMMARY

An important challenge while teaching is to generate professionals with creativity, which can find spontaneously new solutions.

Notice the impact of modern computer games in teenagers, the time they spend in those games and the skills they develop for abstraction and generation of creative solutions. This innovative impulse must be improved in the university career by adding technical knowledge and working methodology to it, in order to form a successful professional.

This paper describes the pedagogical experience acquired during the implementation of a game-programming contest and the contest's technical basis.

1. INTRODUCTION

Computer simulation games may provide a great source of motivation when teaching programming. Besides, artificial life has a special attractive to bioengineering students.

Because of this, we implemented a programming tournament, in which a group of students designs a microorganism (MO) and then all the groups participate in a match that takes place in a common environment. Each MO must know how to move to get nutrients (and therefore increase his energy), fight against another species, or if its energy level is too low, move away and avoid confrontation. We also gave the students the environment source code (which they could not modify) and several MO examples (that they would improve). Periodically, we organized competitions using the original environment and the students' MO source codes.

The environment was designed according to Object Oriented Programming, and now we are going to describe its main features.

2. ENVIRONMENT

The *Microorganism* class defines the MO' s basic behavior. Starting from this abstract class, the students must develop their MOs by inheritance, with their own survival tactics. Each MO must provide a virtual method that returns the position that the MO wants to move to, and consequently making a choice about feeding, attacking or reproducing.

The *Agar* class models the "culture medium" that the MOs feed from. The MOs can ask the **agar** object about the information they may need, i.e., the nutrient distribution or the energy and species of any other MO.

The MO' s life takes place at the container class *Petri* (which models a Petri dish), where the rules of life are applied. This set of rules verifies the validity of the movements required by every MO. According to these rules, the MOs feed, fight (if they move to the same position and belong to different species) or procreate (if they move to the same position but they belong to the same species).

The *Colony* class contains a group of MOs that belong to the same species. It moves the MOs when **petri** object requires to do so. The MOs do not have access to this class.

3. TOURNAMENT RULES

A tournament begins and ends with only one source code per group, but it can be improved during the time between competitions. The source code of the MO that wins the day' s competition is published, so all the students have the possibility to enhance their MO with better strategies every time, making the tournament evolve. Finally, the winner of the tournament is the group whose MO won more battles.

4. CONCLUSION

This didactic tool generated great enthusiasm about programming among the students. It also made easier to teach topics such as Object Oriented Design, inheritance, aggregation, polymorphism, class templates, STL and OpenGL libraries, all of them included in some way in the simulation code.

We also noted important improvements along the tournament in many different aspects: abstraction, codification skills, group working, dedication and innovation.