

# Instituto Politécnico Nacional

ESC Superior de Cómputo

"EVOLUTIONARY COMPUTING" USING GOLLY FOR THE GAME OF LIFE

Jorge Luis Rosas Trigueros

Saldaña Aguilar Gabriela

01/11/16

## THEORETICAL FRAMEWORK

Evolutionary algorithms have been shown to be successful for a wide range of optimization problems. While these randomized search heuristics work well for many optimization problems in practice, a satisfying and rigorous mathematical understanding of their performance is an important challenge in the area of genetic and evolutionary computing. It has been proven for various combinatorial optimization problems that they can be solved by evolutionary algorithms in reasonable time using a suitable representation together with mutation operators adjusted to the given problem. The representations used in these papers are different from the general encodings working with binary strings as considered earlier in theoretical works on the runtime behavior of evolutionary algorithms. The chosen representations reflect some properties of partial solutions of the problem at hand that allow to obtain solutions that can be extended to optimal ones for the considered problem.

Dynamic programming is a well-known algorithmic technique that helps to tackle a wide range of problems. A general framework for dynamic programming has been considered by e. g. Woeginger and Kogan. The technique allows the design of efficient algorithms, that solve the problem at hand to optimality, by extending partial solutions to optimal ones.

**Framework for Evolutionary Algorithms** An evolutionary algorithm consists of different generic modules, which have to be made precise by the user to best fit to the problem. Experimental practice, but also some theoretical work, demonstrate that the right choice of representation, variation operators, and selection method is crucial for the success of such algorithms. We assume that the problem to be solved is given by a multi-objective function  $g$  that has to be optimized. We consider simple evolutionary algorithms that consist of the following components. The algorithm (see Algorithm 2) starts with an initial population  $P_0$ .

During the optimization the evolutionary algorithm uses a selection operator  $\text{sel}(\cdot)$  and a mutation operator  $\text{mut}(\cdot)$  to create new individuals. The  $d$ -dimensional fitness function together with a partial order  $\text{par}$  on  $\mathbb{R}^d$  induce a partial order  $\text{dom}$  on the phenotype space, which guides the search. After the termination of the EA, an output function  $\text{out}(\cdot)$  is utilized to map the individuals in the last population to search points from the original search space.

The first thing we need to ask ourselves is ¿What I'm going to evaluate with my genetic algorithm?

R= We need to evaluate how good is a solution for the given problem, it means that we will have a bunch of solutions and we want the better one. In short Chromosome=Possible solution.

## THE GAME OF LIFE

The Game of Life is not a typical computer game. It is a 'cellular automaton', and was invented by Cambridge mathematician John Conway.

This game became widely known when it was mentioned in an article published by Scientific American in 1970. It consists of a collection of cells which, based on a few mathematical rules, can live, die or multiply. Depending on the initial conditions, the cells form various patterns throughout the course of the game.

### The Rules

**For a space that is 'populated':**

Each cell with one or no neighbors dies, as if by solitude.

Each cell with four or more neighbors dies, as if by overpopulation.

Each cell with two or three neighbors survives.

**For a space that is 'empty' or 'unpopulated'**

Each cell with three neighbors becomes populated.

## GOLLY

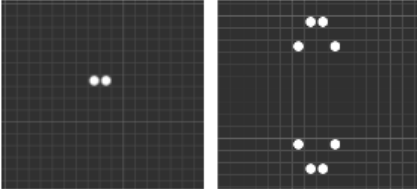
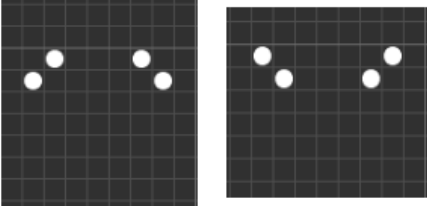
Golly is a tool for the simulation of cellular automata. It is free open-source software written by Andrew Trevorrow and Tomas Rokicki; it runs on Linux, FreeBSD, OpenBSD, Windows, OSX, iOS, and Android, and can be scripted using Lua or Python. It includes a hashlife algorithm that can simulate the behavior of very large sparse patterns in Conway's Game of Life such as Paul Rendell's Life Universal Turing Machine, and that is fast enough to simulate some patterns for  $2^{32}$  or more time units. It also includes a large library of predefined patterns in Life and other rules.

## EQUIPMENT AND MATERIAL.

SOFTWARE: The program is made in Python

## BODY

We were told to find a Glider and an Oscillator for the rules bellow (see e. g. [1]).

B2/S68	B3/S7	B4/S5
<div><div>Glider</div><div></div><div> </div></div>	NO HAY	NO HAY
<div><div>Oscillator</div><div></div></div>	NO HAY	NO HAY

F1:Golly

## CONCLUSION

I liked it the way we took the class in the lab like if it was free style so we could play with the golly program and have something more interactive for the class, also as we have seen this in the previous class we found it familiar.

## BIBLIOGRAPHY

Evolutionary Computing class