CUARTO CONGRESO ARGENTINO DE ENSEÑANZA DE LA INGENIERÍA (IV CAEDI)
Buenos Aires 1, 2 y 3 de Septiembre de 2004

Competencias de Vida Artificial para Fomentar la Creatividad

Daniel Beber[1], Diego Milone[2], José Biurrun[3], Aldo Sigura[4]
[1] Cátedra de Computación II, Facultad de Ingeniería Universidad Nacional de Entre Ríos, Ruta 11 km 10-Oro Verde-Entre Ríos. Tel. 0343-4975100/101, e-mail: dbeber@edeersa.com
[2] e-mail: d.milone@ieee.org, [3] e-mail: jose_biurrun@yahoo.com.ar, [4] e-mail: asigura@edeersa.com

RESUMEN

Un desafío en la enseñanza es la formación de profesionales con creatividad que pueden encontrar espontáneamente nuevas soluciones. Por otro lado, es importante observar el impacto en los adolescentes de los juegos modernos de computadora, el tiempo que le dedican y las habilidades que desarrollan utilizando la abstracción y generando soluciones creativas. Este impulso innovador se debe mejorar en la carrera universitaria agregando conocimiento técnico y trabajando metodológicamente para formar un profesional exitoso.
Los juegos de simulación por computadora pueden proporcionar una gran fuente de motivación en la enseñanza de la programación. Además, la vida artificial tiene un atractivo especial en los estudiantes de bioingeniería. Debido a esto, se implementó un torneo de programación para motivar a los estudiantes y animarlos a utilizar su creatividad innata. En este torneo los estudiantes tuvieron que diseñar e implementar una forma de vida artificial, aplicando sus conocimientos de programación en una diferente y quizás más atractiva manera.
Durante el dictado de la materia se realizaron dos versiones de una misma idea. En la primera versión, llamada "El Torneo", los estudiantes tuvieron que diseñar un microorganismo (MO) y participar en un certamen donde los MO competían un ambiente común. Esta participación fue opcional y abierta a todos los estudiantes de la Facultad mediante una publicación en Internet. Cada MO tenía que saber moverse para conseguir los alimentos (y por lo tanto aumentar su energía), luchar contra otra especie o, si su nivel de energía era demasiado bajo, separarse de los adversarios para evitar la confrontación. Para su entrenamiento, se dio a los estudiantes el código fuente del ambiente y varios ejemplos de MO a partir de los que ellos podían diseñar sus propias estrategias de supervivencia. Para este caso, no se analizaba el diseño ni la implementación, debido a que estaba obligado por la estructura propia de la codificación del ambiente. Periódicamente, se organizaron competencias de los MO realizado por los estudiantes.
La segunda versión, que llamamos "La Evolución", simula el desarrollo de una colonia de microorganismos que poseen sólo un cromosoma (evoMO). La codificación de esta implementación debió ser realizada completamente por los alumnos que cursaban la materia y fue requisito para obtener la aprobación, en la que se prestaba atención a todos los aspectos de la programación (buen estilo del diseño, eficiencia en el código, etc.). Los evoMO tienen que sobrevivir mejorando sus habilidades para conseguir el alimento que los "sembradores" (SMO) han esparcido por el territorio.
En esta experiencia didáctica se observó un importante incremento en la motivación de los alumnos para resolver los trabajos prácticos plateados, lo cual simplificó la enseñaza de conceptos avanzados de programación. Además se observaron importantes mejoras en las capacidades para la abstracción, creación de soluciones novedosas y trabajo en equipo.

***Palabras clave**: enseñanza de programación, C++, juegos motivadores, vida artificial.*

1. Introducción.

La poca experiencia que poseen los estudiantes de los primeros años de la carrera para realizar una planificación adecuada de los tiempos para el cursado de las asignaturas, sumado a una creciente falta de motivación para focalizar su esfuerzo, se ha convertido en un fuerte escollo para la enseñanza. En años anteriores hemos percibido un aumento notable en la tasa de deserción de alumnos. Por ejemplo, de un total de 113 inscriptos al cursado, solamente 11 lograron regularizar la materia y 6 quedaron libres por no superar las evaluaciones, pero 96 alumnos abandonaron el cursado sin aprovechar todas las instancias de evaluación. En un análisis preliminar de las causas que llevaron a esta situación, realizamos una separación en aquellas que se pueden considerar extra-cátedra e intra-cátedra. Dentro de las causas externas a la cátedra se pueden citar problemas de articulación en el plan de estudio, la deficiente preparación con que ingresan los alumnos a la Universidad, los problemas económicos que los obligan a trabajar paralelamente al estudio, etc. En el presente trabajo presentamos una alternativa didáctica para atacar la que creemos puede ser la principal causa intra-cátedra: la falta de motivación a la realización de los trabajos prácticos por parte del alumno.
A continuación se describirán los modelos utilizados en las experiencias didácticas que implementamos, distinguiendo a "*el torneo*", que se realizó fuera del cursado de la materia y la utilización del modelo adaptado para la evolución artificial como trabajo práctico dentro del cursado formal de la materia. En la siguiente sección se presentará el diseño orientado a objetos y algunas referencias a la implementación del programa en el que se basan los alumnos para crear sus propios microorganismos y participar del torneo. En la sección 3 se describe la codificación genética del comportamiento de los microorganismos y los algoritmos relacionados con "*la evolución*" simulada. En la sección 4 se realiza una discusión entorno a nuestra experiencia didáctica y finalmente, en la sección 5 se encuentran las conclusiones.

2. Vida artificial: "El torneo".

El modelo de "El torneo" incluyó las clases: Microorganismo, Agar, Petri y Colonia. En la Figura 1 se muestra las clases básicas y sus herederas.

2.1. La clase *Microorganismo*.

Esta clase define el comportamiento básico de los MO. Comenzando por esta clase, los estudiantes deben desarrollar sus MO por herencia, con sus propias estrategias de supervivencia. Cada MO debe proporcionar un método que devuelva la posición a la cual el MO desea moverse e implícitamente su decisión de buscar comida, atacar o reproducirse.

Pueden hacer uso de información relativa a la ubicación de sus enemigos o la ubicación del alimento sobre la superficie. Esta valiosa información será provista por la clase *Agar*, que se describe a continuación.

2.2. La clase *Agar*.

La clase *Agar* modela el "medio de cultivo" del cual se alimentan los MO. Los MO pueden solicitar al objeto **agar** cierta información que este puede brindar, por ejemplo, la distribución de nutriente o también la energía y especie de otro MO. Por ejemplo, si un MO desea obtener la información sobre una posición específica, las peticiones serían:

1. Cantidad de nutriente en la posición (x , y): agar.nutriente(x,y);
2. Energía del MO localizado en la posición (x , y): agar.energia(x,y);
3. Especie del MO localizado en (x , y): agar.ocupacion(x,y);

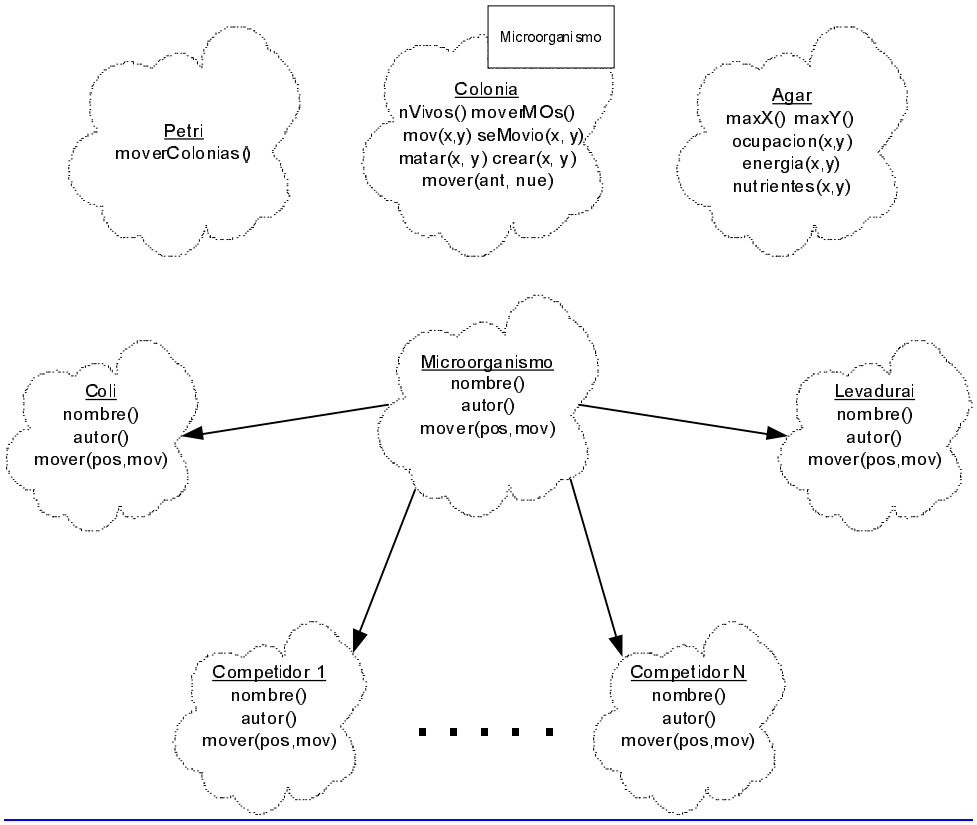

Figura 1: Diagrama de clase de "El Torneo"

2.3. La clase *Petri.*

La vida de los MO ocurre en la clase contenedora *Petri* (que es el modelo de un Cápsula de Petri), donde las reglas de la vida se aplican (véase la figura 2). Este sistema de reglas verifica la validez de los movimientos requeridos por cada MO. Según estas reglas, los MO se alimentan, luchan o procrean. Si un MO desea moverse a una posición específica, hay tres situaciones posibles:

- La posición es vacía y está dentro del Cápsula de Petri: **petri** mueve el MO.
- La posición está ocupada por un MO que pertenece a otra especie: los MO luchan, y sobrevive el de mayor energía. Si tienen la misma cantidad de energía, **petri** elige aleatoriamente quién permanece vivo.

- La posición está ocupada por un MO que pertenece a su especie: hay posibilidad de procreación. Es posible solamente si una de las ocho posiciones circundantes está libre, así pueden ubicar a su descendiente. Si procrean, ambos padres pierden el 10% de su energía, que constituye la energía del nuevo MO.

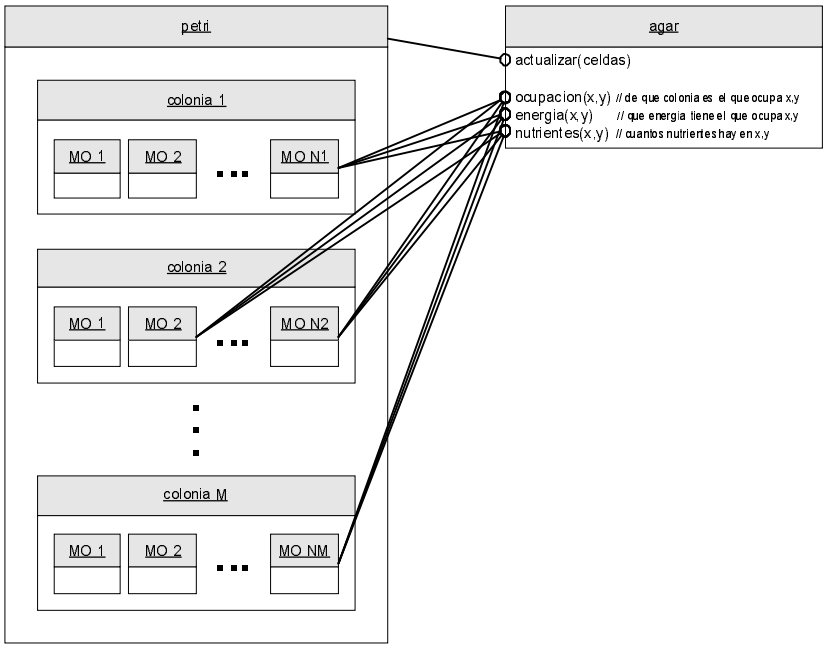

Figura 2: Diagrama de agregación de "El torneo"

Otras reglas:

- Cada MO pierde 5 unidades de energía por unidad del tiempo debido al envejecimiento y 10 por cada movimiento.
- En cada unidad de tiempo todos los MO incrementan su energía con el 1% del alimento existente en su posición actual.
- La distribución de nutriente inicial se selecciona aleatoriamente para cada torneo: a modo de ejemplo, una distribución bidimensional de la comida podría ser la siguiente:

$$n(x,y) = \sum_i N_i e^{-\left(\frac{x-x_i}{\Delta_i}\right)^2 - \left(\frac{y-y_i}{\Delta_i}\right)^2}$$

donde N_i determina el valor máximo de nutrientes en el centro (x_i, y_i) de cada distribución gaussiana y Δ_i, controla la dispersión de comida entorno a cada centro.

2.4. La clase *Colonia*.

La clase *Colonia* contiene un grupo de MO que pertenecen a una misma especie. Mueve los MO cuando el objeto **petri** lo requiera. Los MO no tienen acceso a esta clase, un MO decide moverse como respuesta a la petición del objeto de la **colonia**, devolviendo un movimiento relativo.

2.5. Las reglas de “El torneo”.

Podemos resumir las reglas de “El torneo” en los siguientes puntos:

- Los estudiantes pueden formar grupos para diseñar los MO.
- Las competencias comienzan y terminan con un solo código fuente por grupo, pero este puede ser mejorado entre competencias.
- Una **batalla** es ganada por la eliminación de todos los MO de su colonia oponente.
- Una **competencia** es ganada cuando se gana 5 batallas.
- El código fuentes del MO ganador es publicado, de esta forma se da la posibilidad a todos los estudiantes de mejorar sus MO conociendo la mejor estrategia del momento, haciendo de esta manera que el torneo vaya mejorando.
- El ganador del **torneo** es el grupo que gane más competencias.

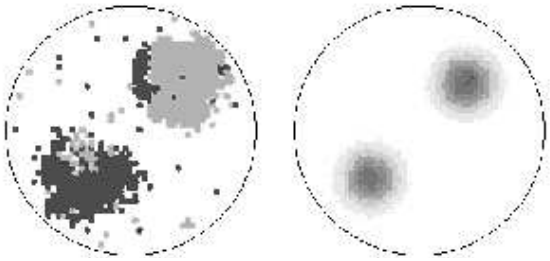

Figura 3: Ejemplo de salida de la simulación. La colonia se encuentra a la izquierda, y la distribución de nutrientes a la derecha

3. Vida artificial: “La evolución”.

A diferencia de la versión anterior, en este nuevo modelo los MO tienen la capacidad de evolucionar, es decir, de ir mejorando sus habilidades durante el desarrollo del programa. Esto se logra porque ahora tienen un cromosoma con la información genética que determina sus movimientos. En este modelo no hay una competencia de supervivencia entre especies, sino que hay una presión selectiva que empuja a los *evoMO* a que mejoren sus habilidades constantemente para desarrollar una mejor manera de alimentación. Esta presión es impuesta por la necesidad de conseguir el alimento que los organismos "sembradores" dejan detrás de su trayectoria [3].

3.1. La clase *evoMO* .

La clase *evoMO* debe proporcionar métodos para devolver la información genética del único cromosoma del microorganismo, su nivel de energía, su posición real y la posición a la que desea moverse. Si la posición de un *evoMO* es (x,y), las celdas a su alrededor pueden ser numeradas de la siguiente manera:

1	0	7
2	*x,y*	6
3	4	5

sinc(*i*) Laboratory for Signals and Computational Intelligence (http://fich.unl.edu.ar/sinc)
D. Beber, D. H. Milone, J. Biurrun & A. Sigura; "Competencias de vida artificial para fomentar la creatividad"
IV Congreso Argentino de Enseñanza de Ingeniería. 2004.

Un organismo "inteligente" tomaría la siguiente determinación: *si hay nutrientes en la celda nº 7, entonces me moveré a la celda nº 7*. La información referida a dónde moverse de acuerdo a la ubicación del nutriente es almacenada en el cromosoma de la siguiente manera:

Si hay nutrientes en la celda:	0	1	2	3	4	5	6	7
Moverse a la celda:	0	1	2	3	4	5	6	7

Sin embargo, la información del cromosoma (representada en este ejemplo por la segunda fila) es aleatoriamente generada, por lo tanto se tiene muchos patrones de cromosomas. Por ejemplo, suponer que se tiene el siguiente cromosoma:

Si hay nutrientes en la celda:	0	1	2	3	4	5	6	7
Moverse a la celda:	4	4	4	4	4	4	4	4

Un *evoMO* con el cromosoma anterior se moverá siempre para abajo, independientemente del lugar donde el sensor haya detectado comida. El éxito de la evolución consiste en desarrollar mejores cromosomas en cada generación a partir de la supervivencia y reproducción de los más aptos.
Cada determinada cantidad de tiempo, los *evoMO* sin energía se eliminan y se realiza una selección entre los que permanecen vivos para determinar quienes serán los padres del nuevo descendiente [4]. Cinco *evoMOs* se eligen aleatoriamente, y los dos con nivel de energía más alto se seleccionan para **reproducir** e intercambiar su información genética. Esto se realiza eligiendo un cruce al azar en los cromosomas de los padres. Por ejemplo, si los cromosomas de los padres eran:

0	0	0	0	0	0	0	0

4	4	4	4	4	4	4	4

y el punto de cruce es 3, el nuevo *evoMO* tendrá el siguiente cromosoma:

0	0	0	4	4	4	4	4

Las reproducciones terminan cuando todos los MO eliminados son reemplazados.
La **mutación** es una característica de especial importancia en la evolución de las especies. A partir de una probabilidad (generalmente muy baja) de alteración en la información genética, la mutación es simulada cambiando uno de los valores del cromosoma seleccionado. Por ejemplo, si tenemos el cromosoma anterior y los números generados al azar para seleccionar la posición y el valor de la mutación fueran 3 y 7, el cromosoma resultante será:

0	0	7	4	4	4	4	4

sinc(*i*) Laboratory for Signals and Computational Intelligence (http://fich.unl.edu.ar/sinc)
D. Beber, D. H. Milone, J. Biurrun & A. Sigura; "Competencias de vida artificial para fomentar la creatividad"
IV Congreso Argentino de Enseñanza de Ingeniería. 2004.

3.2. La clase *SMO.*

Este microorganismo es encargado de repartir el alimento cuando se está moviendo a través del ambiente. Para simular un movimiento más realista se guardará su dirección por cierta cantidad de iteraciones, eligiendo una nueva al azar para realizar el cambio.

3.3. Reglas de la evolución.

El modelo de esta evolución puede resumirse en los siguientes pasos [4]:

Inicialización de las colonias
mientras la evolución no se detenga
seleccionar los mejores MO
reproducir y **generar** la nueva población

El cromosoma inicial se define en la inicialización, asignando valores al azar entre 0 y 7. El comportamiento de los primeros *evoMO* es totalmente imprevisible, pero evolucionarán con cada nueva generación.
Los *evoMO* pierden su energía por envejecimiento, moviéndose y procreando. Luego de cierto número de iteraciones, hay una selección estocástica de los mejores especimenes que tendrán la responsabilidad de procrear. Los *evoMO* con nivel de energía más alto tienen mejores habilidades para sobrevivir (mayores probabilidades de ser seleccionados) motivo por el cual transferirán su información genética.
Para prevenir la extinción de la colonia, si no se alcanza la cantidad de iteraciones necesarias para una reproducción y el número de *evoMO* cae por debajo de un valor mínimo, la simulación se detiene y el proceso de la procreación se realiza nuevamente.

4. Discusión y conclusiones.

Estas herramientas didácticas generaron gran entusiasmo en la programación entre los estudiantes, motivo por el cual también se hizo más fácil enseñar temas avanzados de la programación orientada del objeto.
El esfuerzo y la práctica son también esenciales en el proceso de aprendizaje. Ambas áreas fueron cubiertas extensamente con esta experiencia. También observamos importantes mejoras durante las competencias en diversos aspectos: abstracción, habilidades de la codificación, trabajo del equipo, esmero e innovación. Pero por sobre todos estos puntos, observamos notables desarrollos con creatividad al diseñar las implementaciones, algo que no sucede en muchos casos con la enseñanza mediante los métodos tradicionales.
En cuanto a la programación es importante destacar que debido a que los MO tienen que interactuar con partes del programa hecho por terceros, el alumno tiene seguir estrictamente un conjunto de especificaciones muy precisas para que su propuesta se integre adecuadamente. En esta propuesta se impone al alumno el desafío de trabajar con el espíritu de investigación y creación sometida a la crítica permanente, pero por otro lado se requiere una gran cuota de responsabilidad y compromiso con los

resultados ante el cumplimiento de tareas requeridas por la cátedra o los encuentros del torneo.
En muchas ocasiones se tiene una errónea interpretación de la "optimización" de la tarea docente que conduce al desarrollo de prácticas muy puntuales, rutinarias, mecánicas y de directo traslado de los contenidos teóricos de cada tema. Todo ello puede llevar a una desmotivación del alumnado (además de la deficiente formación).
Para paliar dichos problemas, se hace necesaria la utilización de técnicas docentes que fomenten el interés de los alumnos utilizando los recursos disponibles. Este desafío exige de los alumnos asumir un papel de protagonistas activos, de creadores de soluciones aplicando los conocimientos vistos en la teoría a situaciones prácticas con problemas abiertos, donde se puede arribar a muchas soluciones correctas.
Es de destacar que en estos emprendimientos se ha realizado un uso masivo de grupos de discusión creados por la Cátedra específicamente. Esta herramienta ha complementado muy bien a los encuentros presenciales, proveyendo al alumno una vía alternativa para evacuar sus dudas, discutir soluciones y acceder a la documentación provista por la Cátedra.

5. Trabajos futuros.
Respecto a la evaluación de los beneficios obtenidos en la enseñanza, a priori se observa ventajas cualitativas y se está obteniendo los datos para el cálculo de un resultado cuantitativo.
En cuanto a los aspectos meramente técnicos en la programación, en futuros modelos se pueden considerar ambientes tridimensionales, la posibilidad de que más de dos colonias intentan sobrevivir en el mismo ecosistema, incorporar la evolución en el modelo para torneo y distribuciones de nutrientes que varíen en el tiempo.
Actualmente se está trabajando en una nueva versión de estos programas, en los cuales hay una red neuronal entre los sensores y los pies de los MO [6][7]. La información de los pesos sinápticos se almacena en los genes, modelados con números de punto flotante [5]. Este modelo es más complejo que los que están descritos en este trabajo, así que estimamos que no es conveniente para alumnos de segundo año como trabajo práctico durante el cursado, pero si aplicable al torneo con alumnos del ciclo superior de la carrera.
Otros aspectos a considerar en el futuro son los relacionados con la incorporación de herramientas que se desarrollan en otras unidades de la materia, tales como recursividad y cálculo numérico, proponiendo nuevas alternativas del juego. Actualmente, a partir de las experiencias docentes, se están adaptando las guías de trabajos prácticos para asegurar un incremento gradual en la complejidad del diseño e implementación de las soluciones.

6. Referencias.
[1] Rocha, L.F., "Dinámica de una Red de Autómatas Celulares" . 1991. UBA.
[2] Sitharama Iyengar, S., "Computer Modeling of Complex Biological Systems", CRC Press.
[3] Bäck T., Hammel U. and Schewfel H-F., "Evolutionary Computation: Comments on History and Current State," IEEE Trans. on Evolutionary Computation, vol. 1, no. 1, apr. 1997.

[4] Goldberg D. E., "Genetic Algorithms in Search, Optimization & Machine Learning", Addison Wesley, chap. 1, 3, 4, mar. 1997.
[5] Michalewicz Z., "Genetic Algorithms + Data Structures = Evolution Programs", Springer - Verlag, 1992.
[6] Maniezzo V., "Genetic Evolution of the Topology and Weight Distribution of Neural Networks," IEEE Trans. on Neural Networks, vol. 5, no. 1, jan. 1994.
[7] Montana D. J., "Neural Networks Weight Selection Using Genetic Algorithms," chap. 5 in Intelligent Hybrid Systems, J. Wiley & Sons, 1995.