Curso 09/10

Herramientas de la Inteligencia Artificial

1. Introducción

La práctica consiste en crear con ProGen una aplicación de Programación Genética (PG) que sea capaz de generar algoritmos de ordenación. La salida del programa será, por tanto, un árbol que sea capaz de ordenar de menor a mayor vectores de un máximo de 5 elementos. El número máximo de elementos se fija en 5 porque el esfuerzo computacional de que vamos a disponer en esta práctica es únicamente el que pueda darnos uno de los equipos del laboratorio en apenas unos minutos de ejecución. Este ámbito nos sujeta a limitaciones un poco pobres, pero sirve como muestra de lo que se podría conseguir invirtiendo las ingentes cantidades de tiempo y potencia de cómputo que necesitaría un sistema de PG de producción real.

Los algoritmos de ordenación son un dominio clásico de las Ciencias de la Computación en el que se lleva trabajando prácticamente desde que existe esta disciplina. Hoy en día existen algoritmos de ordenación que han demostrado ser extremadamente eficientes. Pese a que todos ellos tienen un caso peor que suele ser bastante malo, aunque generalmente poco probable, es muy difícil o prácticamente imposible que mediante PG y con unos recursos computacionales tan limitados como los nuestros, consigamos generar un algoritmo que mejore alguno de los ya existentes.

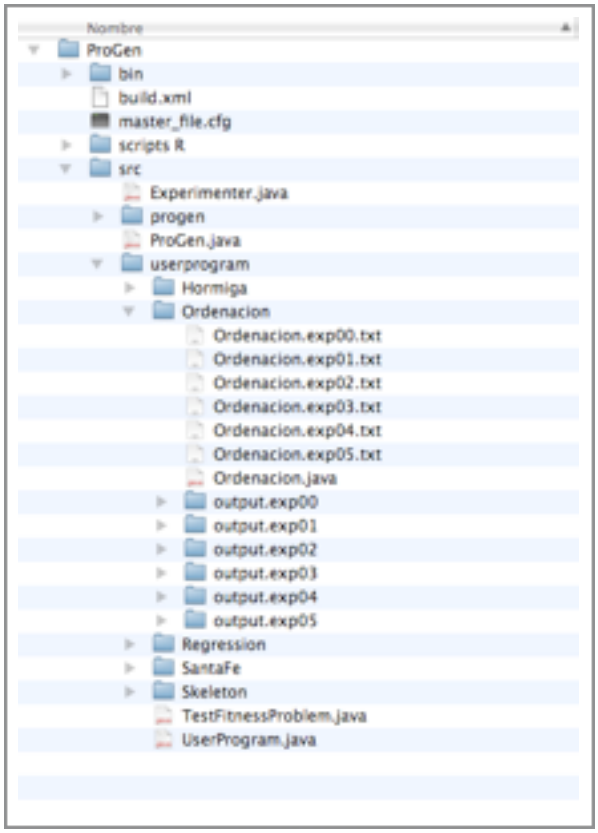
Aún así, resulta muy interesante ver cómo la PG puede ser usada para resolver problemas del mundo real. Además, si alguien consiguiese generar un algoritmo que mejore alguno de los ya existentes habría conseguido él solo uno de los objetivos de la Inteligencia Artificial: generar comportamientos artificiales que puedan competir con los creados por el ser humano.

2. El problema más a fondo

El objetivo primordial, como ya hemos dicho, es generar expresiones que ordenen vectores y cualquier aplicación que lo consiga será suficiente para superar la práctica. Sin embargo, es obvio que lo que nos interesa es obtener algoritmos de ordenación que podamos utilizar, así que la baja complejidad, la rapidez y la concisión del código generado serán muy valoradas. De este modo, una expresión correcta de 3000 nodos y profundidad 150 puntuará mucho menos que una expresión igual de correcta pero con 70 nodos y profundidad 10. Del mismo modo, la rapidez con la que la expresión sea capaz de ordenar vectores será igualmente valorada.

Cada alumno debe entregar su copia de ProGen, incluyendo el UserProgram que hayan programado. Dentro del UserProgram, se incluirá el .java con al función de fitness, y tantos archivos de configuración .txt como experimentos distintos se documenten en la memoria. También hay que incluir los respectivos outputs obtenidos en los experimentos. Por ejemplo, si decido llamar a mi problema Ordenacion, entregaré un ProGen conteniendo en userprogram un directorio llamado Ordenación, con un Ordenacion.java, varios Ordenacion.exp00.txt, Ordenacion.exp01.txt, etc. y sus correspondientes output.exp00, output.exp01, etc.

Un ejemplo de jerarquía de directorios correcta:



Todos los experimentos tienen que estar documentados en la memoria, incluyendo los parámetros que distinguen ese experimento de los demás, cuál es el razonamiento o la justificación de probar esos parámetros, cuáles han sido los resultados del experimento, qué conclusiones se obtienen y qué nuevos experimentos sugieren los resultados.

Las prácticas entregadas se valorarán según dos parámetros:

1. La calidad del diseño de la aplicación de programación genética, lo cual incluye: la elección del conjunto de funciones y terminales, la forma de implementar la función de fitness, la claridad y concisión del código y la medida en que el programa se ajuste a los estándares que propone ProGen.

2. La calidad de las soluciones que genere, lo cual incluye: la generalidad de los mismos (que funcionen para todos los vectores posibles -independientemente de su longitud- y no sólo para algunos) y su eficiencia (la sencillez de la solución y el tiempo de ejecución).

3. Documentación a aportar

Cada alumno debe entregar el código fuente de su aplicación integrado en ProGen junto con una memoria, cuya estructura podría ser la siguiente:

3. Introducción: una descripción breve del problema.
4. Funciones y terminales utilizados (justificando la elección a ser posible).
5. Función de fitness utilizada.
6. Experimentación realizada: Cuál o cuáles son las soluciones que el alumno considera más interesantes y por qué. Cómo se ha llegado a la solución aportada, cuáles han sido los resultados obtenidos en las distintas pruebas y en la de la versión final entregada, parámetros utilizados y su influencia, conclusiones, etc.
7. Puntos a destacar de la práctica entregada: Si la práctica tiene alguna cualidad importante que debamos valorar, debería aparecer aquí.
8. Conclusiones.

El alumno, por supuesto, debe sentirse libre de presentar una memoria que no se ajuste a este modelo.