TEST CON PREGUNTAS DE RESPUESTA BREVE
DURACIÓN: 1 HORA

(0.5 puntos) **Decir dos maneras de evitar la convergencia prematura en Programación Genética**

Hay que mantener la diversidad. Dos métodos vistos en clase son utilizar conjuntos de torneo pequeños y utilizar poblaciones grandes. También, el paralelismo con islas.

(1 punto) **Enumerar cuatro métodos para acelerar la Programación Genética**

Paralelismo de casos de fitness, de individuos, de población y evolución de código máquina.

Disminuir el tamaño de la población o utilizar un número pequeño de generaciones no es una buena respuesta. Utilizar un tamaño de población 0 y un número de generaciones 0 haría que la PG fuera muy rápida, pero obviamente no encontraría ninguna solución.

(0.5 puntos) **Normalmente, los individuos de la Programación Genética, codifican programas. Pero, cuando se usa "desarrollo de embriones", ¿qué codifican los individuos?. Decir un tipo de problemas donde se utiliza "desarrollo de embriones".**

Codifican programas que construyen otros programas (u otras estructuras, como circuitos, antenas, o circuitos cuánticos)-

(0.5 puntos) **¿Porqué le resulta complicado a la Programación Genética evolucionar programas con bucles o recursividad?.**

Porque la función de fitness puede tardar mucho en evaluar a los individuos que contengan bucles. Se puede poner un tiempo límite para cada individuo, pero esto puede resultar complicado, porque a-priori no se conoce cuanto va a tardar la solución.

(0.5 puntos) **Las fases de una computación cuántica son:**

1. **Se parte de un estado no superpuesto**
2. **Se evoluciona a un estado superpuesto**
3. **El estado se transforma a través de una serie de puertas**
4. **¿Qué se hace al final?**

Al final hay el estado superpuesto colapsa a un estado concreto, que es la solución del problema. Esto ocurre de manera probabilística.

(0.5 puntos) **Supongamos que queremos realizar las siguientes tareas. Para cada una de ellas, decir a qué tipo de tarea de minería de datos pertenece (clasificación, regresión, asociación o agrupación):**

a) Predecir si una persona es proclive a padecer cáncer basándose en su código genético
b) En una tienda de internet, determinar tipos de compradores con características similares

c) Determinar si existe una relación entre comprar libros de ciencia ficción y comprar libros de cosmología
d) Predecir la cantidad de electricidad que consumirá la ciudad de Madrid en la siguiente hora

A clasificación, B agrupación, C asociación, D regresión

(1 punto) **Una ejecución de Programación Genética se evalúa durante 25 generaciones y otra ejecución, durante 50 generaciones. En ambas ejecuciones se utiliza el mismo tamaño de población. ¿Esto implica que el tiempo para la segunda ejecución es aproximadamente el doble que para la primera?. Justificar la respuesta.**

En el enunciado se está haciendo la suposición implícita de que todos los individuos requieren el mismo tiempo para ser evaluados. Pero esto no es cierto. Por ejemplo, un individuo que tenga el doble de nodos que otro, podría tardar el doble en ser evaluado. Tampoco todas las funciones tardar el mismo tiempo en ser evaluados. En la práctica pudisteis comprobar que el tamaño de los individuos tiende a crecer a medida que aumentan las generaciones (lo que en clase llamamos "bloat"), así que en general se puede decir que cuanto más "viejo" es un individuo, más grande es y más tiempo lleva evaluarlo.

(0.5 puntos) **Queremos resolver un problema de clasificación con Programación Genética. Disponemos de 10 ejemplos. La "raw fitness" será el número de ejemplos bien clasificados. ¿Cuánto valdrían la "standard fitness" y la "adjusted fitness"?.**

Standard fitness: 10-x, Adjusted fitness: 1/(1+s)

(1 punto) **Describir brevemente como funciona el procedimiento de validación cruzada y porqué es preferible usarlo a dividir el conjunto de datos en una parte para entrenamiento y otra para test. Si utilizamos validación cruzada, ¿los clasificadores obtenidos serán mejores (mayor porcentaje de aciertos) que si utilizaramos el método de entrenamiento/test?**

La validación cruzada consiste en dividir el conjunto de datos en k partes y repetir un ciclo en el que se entrena con k-1 partes y se hace el test con la parte restante. Al final, se calcula la media de los k tests. Es preferible al método de entrenamiento/test porque con un sólo conjunto de test pueden aparecer sesgos por azar, mientras que con k conjuntos de test, esto es más dificil que ocurra.

IMPORTANTE: En principio, no es cierto que con el método de validación cruzada se obtengan mejores clasificadores, puesto que no es ese su objetivo. Lo que mejora es la fiabilidad de la estimación del porcentaje de aciertos esperado. Es decir, si la validación cruzada nos dice que el porcentaje de aciertos esperado es del 60%, podemos tener más confianza en ese valor, que si hubiéramos utilizado un único conjunto de test.

Sin embargo hay que decir que en muchas ocasiones se utiliza el procedimiento de validación cruzada para estimar el porcentaje de aciertos esperado, pero posteriormente se construye el clasificador final utilizando todos los datos. Esta es la manera de operar de Weka, y al utilizar todos los datos, es posible que el clasificador final tenga mayor calidad. Por otro lado, también hay que observar que utilizando todos los datos para construir el clasificador final, se pierde la independencia entre el cálculo del porcentaje de aciertos esperado y el clasificador final.

(1 punto) **Supongamos que tenemos un conjunto de datos de entrenamiento para hacer minería, con dos atributos X e Y discretos, mas la clase. Supongamos que en esos datos, la clase vale 1 si el atributo X toma el mismo valor que el Y, y 0 en caso contrario. Decir cual de los algoritmos de minería de datos vistos en clase sería más apropiado y porqué.**

Realmente, ninguno sería apropiado, porque ninguno permite comprobar si un atributo es igual a otro. Por ejemplo, las reglas permiten hacer comprobaciones del tipo IF X=3, pero no del tipo IF X=Y. Lo mismo ocurre con los árboles de decisión. El vecino mas cercano tampoco funcionaría, porque el ejemplo (0,0) tendría que estar cercano al ejemplo (1,1) y al (2,2), … y al (1000, 1000). En realidad, (0,1) y (1,0) están mas cerca de (0,0) que (1,1).

(2 puntos) **Utilizando Programación Genética en un problema de clasificación,queremos encontrar un individuo (un clasificador) que obtenga una alta precisión y además que sea rápido. De hecho, la función de fitness es *aciertos/tiempo* , donde "aciertos" es el porcentaje de aciertos alcanzado en un conjunto de entrenamiento de 100 ejemplos, y "tiempo" es el tiempo que le cuesta clasificar el ejemplo más complicado (es decir, en el que tarda más). Se quiere trabajar con una población de 100 individuos y se dispone de 10001 ordenadores trabajando en red. Se sabe que los individuos pueden tardar bastante en ser evaluados porque tienen bucles. ¿Qué esquema de paralelismo podríamos utilizar para poder abortar la evaluación de aquellos individuos poco prometedores, tan pronto como sea posible?. Desarrollar y explicar con cierto detalle**

Podemos tener un ordenador central que para cada ejemplo de entrenamiento y cada individuo, lo ejecute en un ordenador distinto. Harían falta 100*100 ordenadores, mas uno que se encargaría de enviar los individuos y recibir la fitness. De cada ordenador se podría recibir o bien un 0 o bien un 1, según el individuo correspondiente clasifique mal o bien el ejemplo. En el momento en que un ejemplo esté clasificado en el ordenador correspondiente, este envía un mensaje al ordenador central. Así, el ordenador central puede computar la fitness de ese individuo, obtenida hasta el momento.

Si un individuo es poco prometedor, clasificará pocos ejemplos, y a medida que pase el tiempo el valor *aciertos/tiempo* se irá haciendo más y más pequeño. Esta es la característica importante del problema que permite detener a los individuos malos. Se puede abortar su ejecución según varios criterios. Por ejemplo si la fitness conseguida hasta el momento por un individuo es menor del 10% de la del mejor individuo hasta el momento, se le puede abortar y asignarle la fitness conseguida hasta ese momento.

Podemos ir más lejos. Supongamos que (f,t) son parejas de (fitness, tiempo) de los mejores individuos de generaciones pasadas. Si sabemos que para obtener "f" necesitamos un tiempo "t", podremos abortar a aquellos individuos que, habiéndose ejecutado un tiempo "t", han obtenido una fitness mucho peor que "f".

En cualquier caso, nada garantiza con certidumbre completa, que los individuos abortados, si les diéramos más tiempo, no fueran a obtener una fitness alta. Por eso hay que evitar abortar a aquellos individuos que son solo ligeramente peores que el mejor. Se trata de tan solo una heurística. Por cierto, que todo esto es similar al método de corutinas de Maxwell que vimos en clase.