Programación evolutiva

Facultad de Informática

Curso 2011/2012

Práctica 2.

El objetivo de esta práctica es implementar, utilizando en lo posible el esquema de la práctica anterior, un algoritmo evolutivo para optimizar la asignación de alumnos a grupos en un sistema automático de formación de grupos. El algoritmo deberá tener en cuenta tanto la uniformidad de los grupos (medida como la suma total de las notas de sus integrantes en un examen previo) como las preferencias de los alumnos (expresada mediante una lista de compañeros con los que prefieren no volver a formar grupo). Así, un alumno se representa mediante:

- Un identificador (un número entero, no repetido en ningún otro alumno)
- Una nota numérica (no negativa)
- Una lista de 0 o más identificadores de alumnos con los que prefiere no formar grupo

Los cromosomas del problema se deben expresar mediante permutaciones. Así, si hay *N* alumnos, y se desean formar grupos de *M* personas, se puede interpretar cualquier ordenación de estos alumnos como "los grupos resultantes de tomarlos de *M* en *M*". Para lograr que *M* divida exactamente a *N*, se deben tantos insertar "alumnos vacíos" (con IDs negativos, una nota de 0, y sin incompatibilidades) como sea necesario.

Así, dados 5 alumnos con identificadores 10, 20, 37, 59 y 73, y M = 2:

- 10 20 37 59 73 -1 se interpretaría como las parejas {10, 20}, {37, 59}, {73, -1}
- 20 10 37 59 73 -1 equivale a las mismas parejas (ya que {10, 20} = {20, 10})
- 37 59 73 -1 10 20 se puede traducir como {37, 59}, {-1, 73}, {10,20}

En la representación interna del cromosoma, puede ser más sencillo operar con los índices de un array (ó `ArrayList`) en lugar de los identificadores originales. No obstante, los resultados siempre se deben mostrar usando los identificadores originales.

El objetivo es minimizar desequilibrios e incompatibilidades, usando

$$desequilibrio = \sum_{i=1}^{g} \left(\sum_{j=1}^{m} x_{i,j} - \bar{x} \right)^2$$

$$f.evaluacion = \alpha \cdot desequilibrio + (1-\alpha) \cdot incompatibilidades$$

Donde se entiende que g es el número de grupos, $x_{i,j}$ es la nota del alumno i del grupo j, y $\bar{x}$ es la nota media de todos los alumnos; y se cuenta 1 incompatibilidad por cada caso en el que un alumno se ve agrupado con otro con el que había solicitado *no* estar.

NOTA: El problema, y los datos de prueba (anonimizados, por supuesto) provienen de grupo *e-UCM* (del que forma parte el profesor Manuel Freire); si vemos soluciones que nos permiten mejorar las nuestras, estaremos interesados en usarlas, dando crédito a sus respectivos autores.

Interfaz

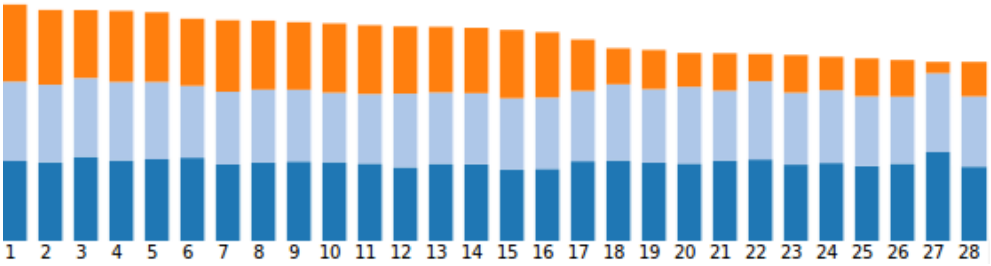
La interfaz gráfica será similar a la de la Practica 1, y soportará, además de los parámetros básicos (tamaño de población, número de generaciones, tasas de mutación, etcétera) los siguientes:

- Como opciones del problema
  - Fichero del que cargar los datos (ver siguiente sección)
  - Número de alumnos por grupo (*M*)
  - Tipo de f. de evaluación a usar. Aunque es obligatorio poder usar la función proporcionada, y poder modificar su valor de α, se deja como opcional proporcionar otras funciones que puedan dar mejores resultados.

- Como métodos de selección
  Ruleta – Torneo aleatorio y determinista – Ranking – (y un método propio)
- Como métodos de cruce
  PMX – OX – Una variante de OX – Ordinal – (y un método propio)
- Como métodos de mutación
  Inserción – Intercambio – Inversión – Heurística

En todos los casos, si una opción puede aceptar parámetros (por ejemplo, tamaño del torneo), será posible modificarlos desde la interfaz.

Es opcional, pero recomendable, incluir en la interfaz alguna forma de inspeccionar el resultado, para entender de un vistazo hasta qué punto el algoritmo está funcionando como se espera. Una posible representación de una agrupación es la siguiente (donde cada barra representa la "nota total" de un grupo, y la nota de cada componente del grupo se representa por un color; se podrían marcar las restricciones violadas con cualquier distinción visual; por ejemplo, números de grupo en rojo):



Ficheros de datos

Cada fichero de datos contiene, en su primera línea, dos números *N* y *R* (donde *N* es el número de alumnos totales, y *R* las restricciones de emparejamiento de esos alumnos). A continuación habrá *N* líneas con dos números cada una: el identificador del alumno, y su última nota. Finalmente, habrá otras *R* líneas, con dos números cada una; el primer número es el identificador de un alumno, y el segundo, el de otro alumno con el que el primero prefiere *no* trabajar.

NOTA: Puedes usar un `br = new BufferedReader(new FileReader(`*f*`))` para ir leyendo del `File` *f* de entrada, línea a línea (mediante sucesivos `br.readLine()`), y luego interpretar cada línea por separado. Para extraer números de una línea, primero puedes partirla con `split(" ")`, y luego puedes convertir los números a enteros con `Integer.valueOf(`*n*`)` ó a coma flotante con `Float.valueOf(`*n*`)`. No te olvides de cerrar `br` al acabar la lectura.

Entrega

Debes entregar

- El proyecto eclipse (igual que en la Práctica 1)
- Una memoria que no debe exceder de 20 hojas con
  - El nombre de los integrantes del grupo y el número de grupo
  - Los resultados más significativos
  - Las conclusiones obtenidas al estudiar y analizar los valores obtenidos con distintos parámetros y distintas ejecuciones
  - Explicaciones claras de los métodos propios

Hay que subir al campus virtual antes del 1 de mayo a las 23:00 el archivo comprimido con el código Java de la aplicación (proyecto con nombre **LXGXXP2**) y la memoria (no hay que imprimirla). En el campus virtual el ejercicio está identificado como **Practica 2**.

La corrección se podrá realizar en la sesión del martes 24 de abril o en la sesión del 8 de mayo.