

Proyecto de Matemática Discreta II-2016 (segunda parte)

Contents

1 Segunda Parte	1
2 Operadores	1
2.1 Population	1
2.2 Generaciones	2
2.3 Fitness	2
2.4 Selection	2
2.5 Mutation	2
2.6 Crossover	2
3 Otras Cosas	2
3.1 Mantener el mejor	2
3.2 Impresión	2

1 Segunda Parte

En esta segunda parte deben entregar un programa que use las funciones definidas en la primera parte para colorear un grafo usando algoritmos genéticos.

Deben entregar un solo archivo, `maingenABCD.c`, donde en vez de `ABC` deben estar las iniciales de los integrantes del grupo.

Pondré ese archivo en el `dirmain` del proyecto que entregaron y compilaré como antes, reemplazando el `main` anterior por este.

Al igual que la primera parte, el `main` deberá leer desde `standard input` un grafo con el formato especificado en la primera parte. Las impresiones deben ser todas por `standard output`.

El `main`, además de leer por `standard input`, debe tomar tres parámetros. (ver abajo).

El `maingenABCD.c` puede tener funciones auxiliares que implementen lo de abajo, o no, pero no puede tener ningún `include` que no sea a `stdlib.h`, `stdio.h` o `Cthulhu.h`. Y sólo debe usar las funciones definidas dentro de él, o las dadas por la API de la primera parte. En particular, recalco otra vez, debe funcionar con CUALQUIER `Cthulhu.h` y no sólo el suyo.

2 Operadores

2.1 Population

La población sobre la cual se empleará el algoritmo genético debe consistir de permutaciones de los números $\{0, \dots, n - 1\}$, donde n es el número de vértices del grafo.

Obviamente los individuos pueden ser simplemente las permutaciones en sí, pero si quieren definir una `IndividuoStructure` (nombre a elección de ustedes) con más campos porque les es conveniente, pueden hacerlo, siempre que uno de esos campos guarde una permutación o un puntero a una permutación.

También, si les es conveniente, pueden definir una `PopulationStructure` (nombre a elección de ustedes) que guarde además del conjunto de todas las permutaciones otros datos que consideren útiles. El tamaño de la población debe ser el primer parámetro de entrada del `main`.

2.2 Generaciones

El número de generaciones debe ser el segundo parámetro de entrada del main.

2.3 Fitness

La función de fitness será la siguiente: para cada individuo (permutación) deben usar la función `OrdenEspecifico()` de la primera parte para proveerle ese orden a la estructura `Nimhe`, y luego deben correr `Greedy`. (llamando a la función de la primera etapa) Si `Greedy` retorna k , la fitness inicial de ese individuo será $n + 1 - k$, donde n es el número de vértices del grafo. Luego de calcular todas estas fitness iniciales, deben encontrar la mayor fitness y multiplicar las fitness de los individuos con mayor fitness por alguna constante mayor o igual a 1. (esto para proveer alguna ventaja evolutiva). Esta constante debe ser el tercer parámetro de entrada del programa.

2.4 Selection

Deben usar selección por ruleta.

2.5 Mutation

Deben elegir al azar dos posiciones y permutar los elementos en esas posiciones. Si las dos posiciones son iguales, deben permutarla con la siguiente posición.

2.6 Crossover

Deben usar `Cyclic Crossover`, eligiendo el punto de inicio al azar.

3 Otras Cosas

3.1 Mantener el mejor

Hice algunos runs y en algunas ocasiones el coloreo empeoraba de una generación a la siguiente y se perdía para siempre. Para evitar esto necesitaba poner una presión evolutiva muy alta, lo cual provoca el riesgo de `early convergence`. Así que en vez de eso, lo que haremos es lo siguiente: luego de calcular todas las fitness y ajustarlas como se dice arriba, se compara la mejor fitness de la nueva generación con la mejor fitness de la anterior. Si empeoró, se toma el individuo de la nueva generación con peor fitness, se lo descarta, y en su lugar se usa uno de los individuos de la anterior generación con mejor fitness.

3.2 Impresión

No voy a especificar mucho el formato de salida esta vez, pero si deben imprimir, además del número de colores que obtienen al final, líneas en donde figure el número de la generación, el mejor número de colores obtenido en esa generación y el promedio de colores de esa generación.