TMH - Proyecto de prácticas

#miarfid

Jorge Jiménez García

Descripción del problema

El problema elegido es una variación de la k-coloración de grafos, en el que cada color tiene un coste asociado y el objetivo pasa a ser de minimización de coste, convirtiendo en minimización el problema en lugar de satisfacibilidad.

El problema se modela como la asignación de distintos tipos de cultivos entre pueblos colindantes de un país, de tal manera que el consumo de agua nacional se minimiza. Sin embargo, para evitar discusiones entre vecinos por competir con el mismo producto, se establece la norma de que dos pueblos vecinos no pueden tener el mismo tipo de cultivo.

Caracterización del problema

Como objetivo, se minimiza $Q(C_k)$ donde C_k es una k-coloración del grafo y Q es una función de coste. Como cada color tiene un coste asociado, se observa que $Q(C_k^{'}) < Q(C_{k+1}^{'}) \iff C_{k+1}^{'} < C_{k+1}^{a} \forall C_{k+1}^{a}$ donde C_k^a es una coloración arbitraria de G con k-colores y $C_k^{'}$ es una coloración de coste mínimo de G
Por tanto, cualquier k-coloración óptima del grafo será mejor que cualquier $(k+1)$-coloración óptima. Esto permite cambiar el problema de multiobjetivo con K y $Q(C_k)$ como objetivos de minimización a solo minimización de $Q(C_k)$.

Con un grafo definido como $G = \{E, V\}$, el individuo se codifica como una assignación $V \to C$ donde C es un set de colores. Como los vertices siempre son estáticos, el genotipo se define como la lista de colores asociados a cada vértice, es decir, $\{c_1, c_2, \ldots, c_{|V|}\}$. La talla del genotipo es siempre $|V|$

Los costes del problema se definen como un mapa $c \to \mathbb{R}\ \forall c \in C$.

El grafo elegido para modelar el país es el grafo de Levi-Desargues, un grafo con 20 vértices y 30 aristas que representa relaciones de perspectiva aplicadas a visión 3D.

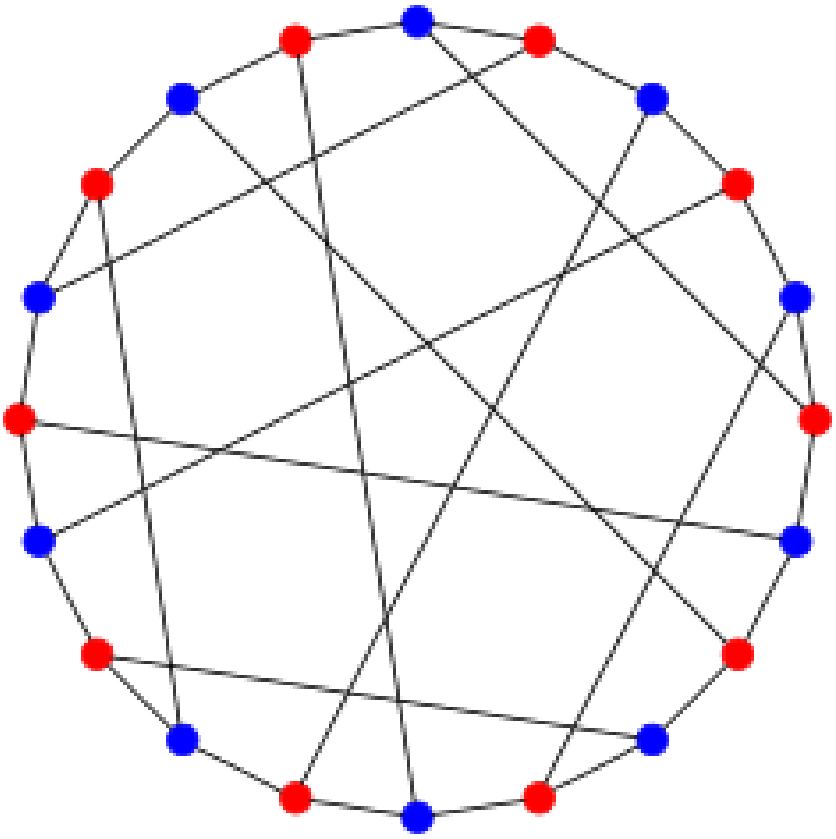


Fig. 1: 2-coloración del Grafo de Desargues

Se sabe que este grafo tiene número cromático 2, es decir, que es 2-colorable y por tanto, un grafo bipartito. Además, se sabe que con particiones del mismo número de vertices.

Por tanto, al ser G bipartito con particiones de talla igual, se sabe que para un mapa de costes ordenado $Q = \{q_1, q_2, \ldots, q_{|V|}\}$, con $q_1 \leq q_2 \leq \ldots \leq q_{|V|}$, el coste mínimo será $C_{opt} = 10 * q_1 + 10 * q_2$.

Con un mapa de costes aleatorio definido como:

```
{
        0: 11.7,
        1: 12.2,
        2: 14,
        3: 14.1,
        4: 14.7,
        5: 15.0,
        6: 21.1,
        7: 23.1,
        8: 43.1,
        9: 43.2,
        10: 100,
        11: 100,
        12: 100,
        ...
}
```

Podemos ver que la solución óptima será $C_{opt_{desargues}} = 10 * 11.7 + 10 * 12.2 = 239$.

Calculo empírico dice que existen $\sum_{i=0}^{|V|} ||\mathbb{C}_i|| = 331124390582408341566388 16 = 3.311e24$, o *treinta y tres septillones ciento doce sextillones cuatrocientos treinta y nueve quintillones cincuenta y ocho cuadrillones doscientos cuarenta trillones ochocientos treinta y cuatro billones ciento cincuenta y seis millones seiscientos treinta y ocho mil ciento ochenta y seis* coloraciones **válidas** del grafo, sin considerar las que rompen la regla de adyacencia de colores. Por tanto, se considera suficientemente complejo.

Solución via Algoritmos Genéticos

Propiedades del problema

Gracias a la propiedad anterior que define $Q(C_k^{'}) < Q(C_{k+1}^{'})$, se incentiva minimizar K lo mas rápido posible para moverse en un espacio de soluciones más cercano al óptimo.

Población inicial

La población inicial se controla con un parámetro que define su tamaño, y consiste en n individuos con coloración aleatoria para sus V vértices.

Operadores y su efecto

Los operadores con mayor relevancia para Algoritmos Genéticos en este problema son el cruce y la mutación.

El cruce se define como un cruce n-simétrico del genotipo de dos individuos. Como el genotipo se define como una lista de colores asociados a cada vértice, esta operación es similar a la permutación de color entre vértices. Este operador controla la reducción de k.

La mutación, por otro lado, se define como, con probabilidad para cada grafo parametrizada, $Color_a \rightarrow Color_b \; \forall V \in Color_a$, es decir, es una transformación de un color a otro para todos los vértices que pertenecen a esa clase de color. Este operador es responsable principal de la reducción de coste $Q(C_k)$, ya que para una K-coloración, se espera encontrar una combinación donde los colores utilizados son los de menor coste..

Adicionalmente, se definen operaciones de purga, tanto por fitness, que elimina un porcentaje de los individuos con peor coste, como de purga con probabilidad aleatoria parametrizada.

Selección de cruce

Dado que la fitness no guarda relación directa con la proximidad a una mejor solución, ya que el cruce es de combinación de genotipos, y además el valor del coste será idéntico entre muchos individuos cuando la población se estabiliza, se desestiman la selección via torneo y proporcional.
Se escoge emparejamiento para cruces aleatorio entre la población.

Exploración de hiperparámetros

La exploración se realiza para los siguientes parámetros:

- `symetrical_crossing_offspring`: El punto de cruce simétrico entre genotipos de padres
- `graph_mutate_chance`: La probabilidad de mutar la coloración de un grafo
- `introduce_n_random_individuals`: El número de grafos aleatorios que añadir a la población en cada iteración, para mantener diversidad genética
- `least_fitting_prune_enabled`: Si se utiliza la purga por menor fitness o no
- `least_fitting_prune_ratio`: El porcentaje de la población a eliminar por purga de menor fitness, si esta activa
- `randomness_survival_prune_enabled`: Si se utiliza la purga aleatoria o no
- `randomness_survival_chance`: La probabilidad de supervivencia de cada individuo en purga aleatoria, si esta activa

Se realiza la exploración de la siguiente manera, con un rango de valores para cada parámetro (o un conjunto de opciones para los parámetros categóricos), se estudian todas las combinaciones posibles de parámetros. Esta exploración se realiza con un número determinado ejecuciones del algoritmo, para minimizar aleatoriedad, y con un límite de tiempo para la exploración.

Los rangos de valores elegidos son los siguientes:

```
{
 symetrical_crossing_offspring_l = [10],
 graph_mutate_chance_l = np.arange(0.3, 0.9, 0.2),
 introduce_n_random_individuals_l = range(1_000, 2_001, 500),
 least_fitting_prune_enabled_l = [True, False],
 least_fitting_prune_ratio_l = np.arange(0.2, 0.6, 0.2),
 randomness_survival_pruning_enabled_l = [True, False],
 randomness_survival_chance_l = np.arange(0.2, 0.5, 0.2),
}
```

Los 3 mejores resultados, para 2 ejecucciones de validación y un límite de exploración de 4 minutos por exploración, son los siguientes:

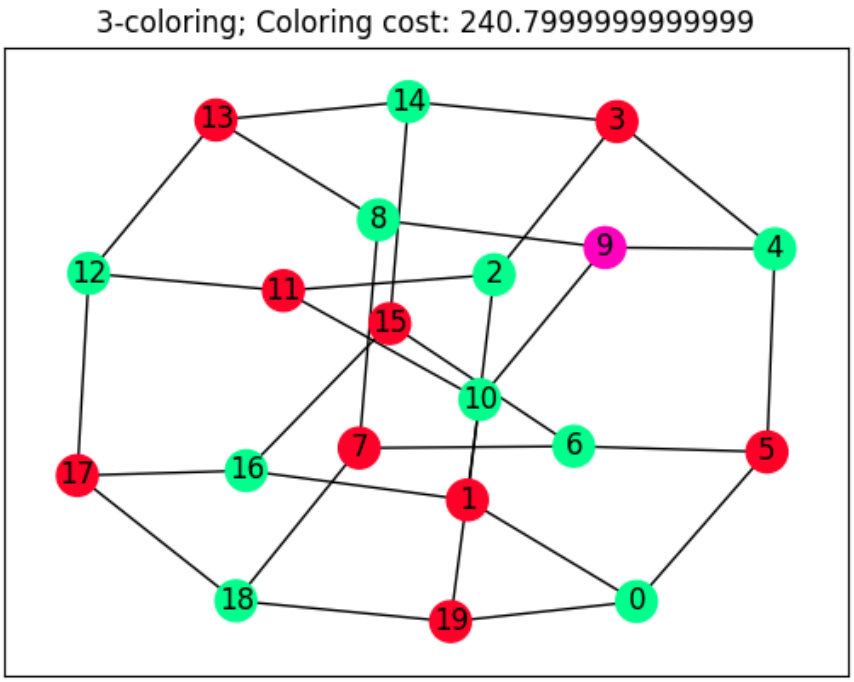
Parámetro	**Top 1**	**Top 2**	**Top 3**
`symetrical_crossing_offspring`	10	10	10
`graph_mutate_chance`	0.9	0.3	0.7
`introduce_n_random_individuals`	2000	2000	2000
`least_fitting_prune_enabled`	No	No	No
`least_fitting_prune_ratio`	-	-	-
`randomness_survival_prune_enabled`	Sí	Sí	Sí
`randomness_survival_chance`	0.2	0.2	0.2
Media del coste	**251.3**	252.4	252.45
Desviación estandar	1.6	0.3	2.65

Es notable que, para las combinaciones óptimas, sus combinaciones de parámetros hacen que el tamaño de la población se mantenga relativamente constante, variando en un factor de entre 1.011 y 0.987. Esto hace que el tiempo se mantenga relativamente constante por iteración (sobre 0.4s) y se puedan realizar más en el límite de tiempo que otras combinaciones de parámetros cuyas poblaciones quizás se disparan.

Resultados

Con estos parámetros en mano, se realiza una exploración final con un límite de tiempo de 5h, que obtiene los siguiente resultados:

```
==========================================================================================================================================

It52668:        Current Best=240.800; Population best cost=240.800
                   K=3      ;            best    K=3
------------------------------------------------------------------------------------------------------------------------------------------
                Solution summary:
                        Population best: 3-colored graph;Proper coloring (No violations); With cost 240.800
                        Current best   : 3-colored graph;Proper coloring (No violations); With cost 240.800
                No improvement
------------------------------------------------------------------------------------------------------------------------------------------
                Progress summary:
                        Deadline  =0:00:00.031491 s
                        Patience  =99963715 iterations
                        Iterations=52668 / 100000000 (0.053%)
                        Optimum estimation: ~ 239.000; distance to estimation: 1.7999999999998977
------------------------------------------------------------------------------------------------------------------------------------------
                Population summary:
                        Current Population size: 14476; (56.6 MB)
                        New Population size    : 14513; (56.7 MB)
                                Population Increase ratio: 1.003

Iteration took 0:00:00.341117 seconds; Total time elapsed: 5:00:00.287047
==========================================================================================================================================



******************************************************************************************************************************************
Stopped: Reason: Hit time deadline
Best found solution: 240.7999999999999
Best: 3-colored graph;Proper coloring (No violations); With cost 240.800
Best coloring: {0: 2, 1: 1, 2: 2, 3: 1, 4: 2, 5: 1, 6: 2, 7: 1, 8: 2, 9: 3, 10: 2, 11: 1, 12: 2, 13: 1, 14: 2, 15: 1, 16: 2, 17: 1, 18: 2, 19: 1}
Took 5:00:00.287156 s
******************************************************************************************************************************************
```



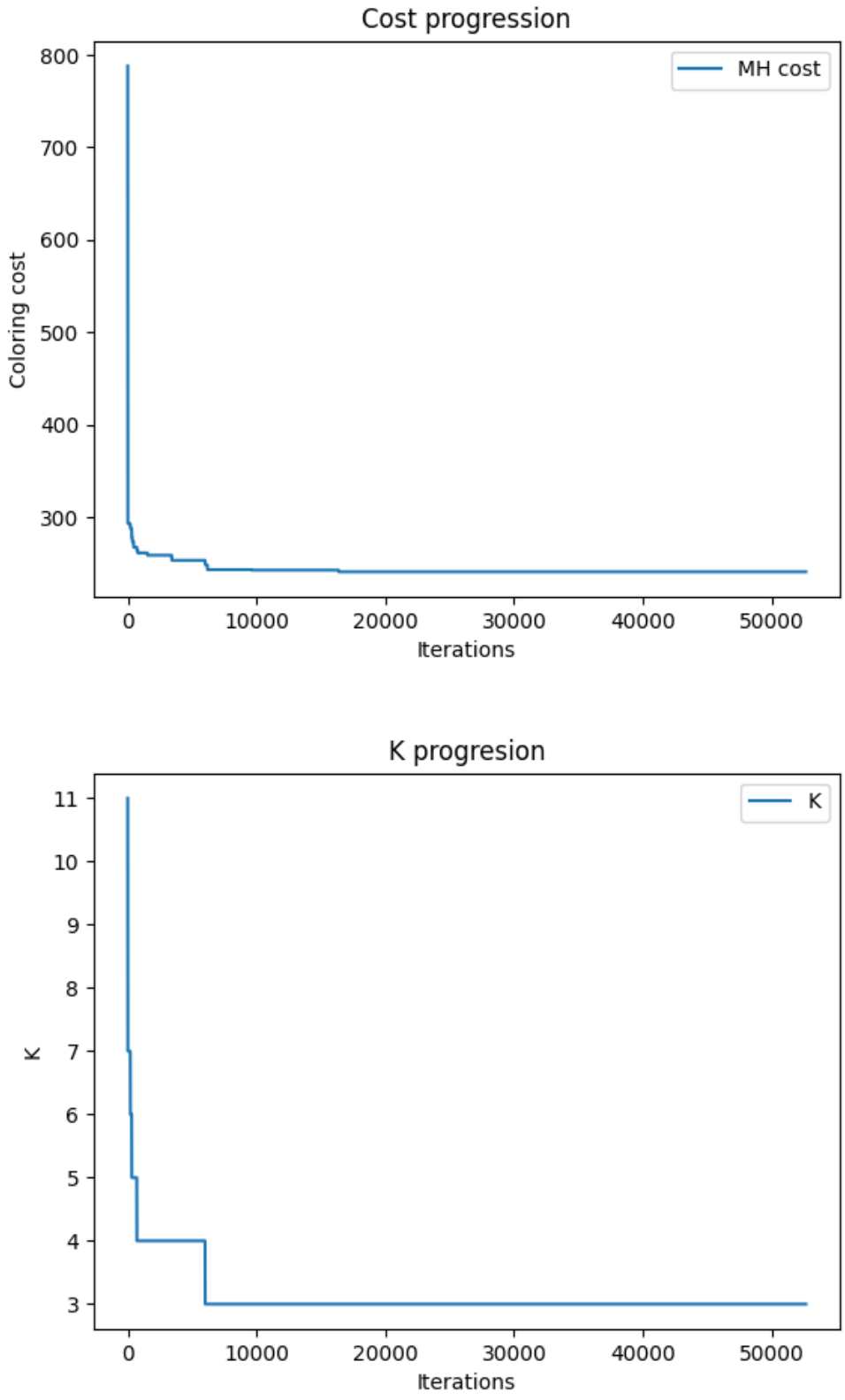


Fig. 2: resultados del algoritmo Genético

Solución via Enfriamiento Simulado

Individuo inicial

Como población inicial se utiliza una coloración trivial del grafo, es decir, una $|V|$-coloración, donde cada vértice tiene un color único asignado.

Operadores de enfriamiento simulado

Enfriamiento Simulado funciona principalmente gracias a las funciones de vecindad, por tanto, se exploran varias definiciones y sistemas de alternar entre ellas.

Se definen 3 operadores de vecindad, `fix_violation_or_decay`, `whole_color_permute` y `whole_color_change`.

- `fix_violation_or_decay`: Si existen violaciones de color, el vecino se define como un grafo idéntico salvo con uno de los nodos que provocan la violación de color mutado a un color distinto. Si no lo hay, un vértice aleatorio muta de color.
  - El color al que cambia es siempre $c \in C_k$ donde C_k es la coloración actual del grafo, es decir, que nunca se añaden colores adicionales ya que el objetivo es minimizar k.
- `whole_color_permute`: Se aplica la transformación $color_a \iff color_b$, donde todos los vertices $\in color_a$ cambian a la clase $color_b$ y viceversa.
  - Este operador tiene el efecto de reducción de coste, donde se intentan obtener vecinos con misma k y coste inferior.
    - Es de notar que en el caso particular del grafo de Desargues, que es bipartito (2-colorable), este operador no obtendrá menor coste para una k-coloración optima, ya que como solo existen 2 2-coloraciones, son equivalentes en términos de coste
- `whole_color_change`: Se aplica la transformación $color_v \to color_b \forall v \in color_a$, es decir, que todos los vertices de la clase de color $color_a$ cambian a otro color distinto.
  - Este operador acelera la búsqueda de vecinos de menor coloración

En cuanto a la reducción de temperatura, se implementan 3 operadores, parametrizados con un parámetro k.

- `linear`: La temperatura decrece linearmente en `k` unidades cada iteración
- `factor`: La temperatura se multiplica por `k` ($\texttt{k} \in (0, 1)$) cada iteración
- `decay`: La temperatura se reduce a $t/(1 + t * k)$ cada iteración

Heuristica en la aplicación de operadores de vecindad

La estrategia de vecindad es la que sigue: Si existen violaciones en la coloración, se aplica siempre `fix_violation_or_decay` para volver a un espacio de soluciones más cercano a la factibilidad.
Si la coloración es válida según la regla de adyacencia, los operadores se aplican con probabilidad uniforme.

Temperatura y ajuste de hiperparámetros

Este problema en particular presenta un desafio a la hora de realizar el ajuste de temperatura. Como la mayoria de vecinos son permutaciones con el mismo número de colores para el grafo, aunque asignados a distintos vértices, en general, ΔC es 0, y por tanto la probabilidad de acceptación es generalmente $e^{0/T} = e^0 = 1$.

Esto dificulta el ajuste de T, y de la exploración grid search realizada, T parece no tener impacto, ya que como hemos visto la acceptación es generalmente 1. Esto se exacerba junto a la velocidad del algoritmo, que presenta

otro problema que se verá en resultados, donde la solución óptima se alcanza muy rápidamente. Por tanto, para este problema, el ajuste de hiperparámetros es irrelevante, aunque se realiza de todos modos.

Los rangos de parámetros a explorar son los siguientes:

```
{
 temperature_grid = range(1, 11, 1),
 decay_and_param = [
        (['linear'], np.arange(0.001, 0.005, 0.001)),
        (['decay'], np.arange(0.9, 0.99, 0.01)),
        (['factor'], [0.8, 0.9]),
 ]
}
```

Por completitud, se incluyen los resultados de grid search, con los mismos parámetros de busqueda como en algoritmos Genéticos; 2-folds, límite de exploración de 4 minutos:

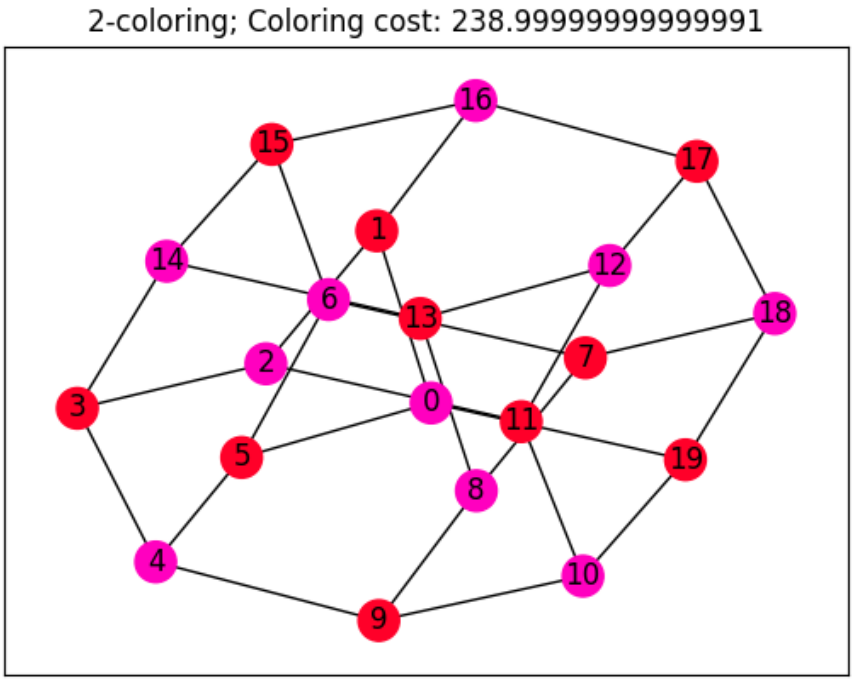
Parámetro	**Top 1**	**Top 2**	**Top 3**	**Top 4**	**Top 5**
`temperature`	1	1	1	1	1
`temperature_reduction_method`	`linear`	`linear`	`linear`	`linear`	`decay`
`temperature_k`	0.001	0.002	0.003	0.004	0.9
Media del coste	**239**	**239**	**239**	**239**	**239**
Desviación estandar	0	0	0	0	0

De las 150 combinaciones de parámetros exploradas, todas las combinaciones alcanzan la solución óptima, salvo 6 de ellas.

Resultados

Con estos parámetros en mano, se realiza una exploración final con un límite de tiempo de 5h, que obtiene la solución óptima en 1 segundo:

```
================================================================================================
It5724: Current best=249.500; Predecessor cost=238.500; Successor cost=238.500
------------------------------------------------------------------------------------------------
        Solution summary:
                Best       : 3-colored graph;Proper coloring (No violations); With cost 249.500
                G          : 2-colored graph;IMProper coloring (4 violations); With cost 238.500
                G successor: 2-colored graph;Proper coloring (No violations); With cost 239.000

        Cost delta = 0.0;
------------------------------------------------------------------------------------------------
        Update summary:
                Improvement over best
                Current temperature: 0.00017640573318632885; Reduction method: decay;
------------------------------------------------------------------------------------------------
        Progress summary:
                        Deadline  =0:03:58.358186 s
                        Iterations=5724 / inf (0.000%)
                        Optimum estimation: ~ 239.000; distance to estimation: -8.526512829121202e-14
================================================================================================



************************************************************************************************************************************
Stopped: Reason: Solution within 0 of estimated optimum
Best found solution: 238.99999999999991
Best: 2-colored graph;Proper coloring (No violations); With cost 239.000
Best coloring: {0: 1, 1: 0, 2: 1, 3: 0, 4: 1, 5: 0, 6: 1, 7: 0, 8: 1, 9: 0, 10: 1, 11: 0, 12: 1, 13: 0, 14: 1, 15: 0, 16: 1, 17: 0, 18: 1, 19: 0}
Took 0:00:01.641849 s
************************************************************************************************************************************
```



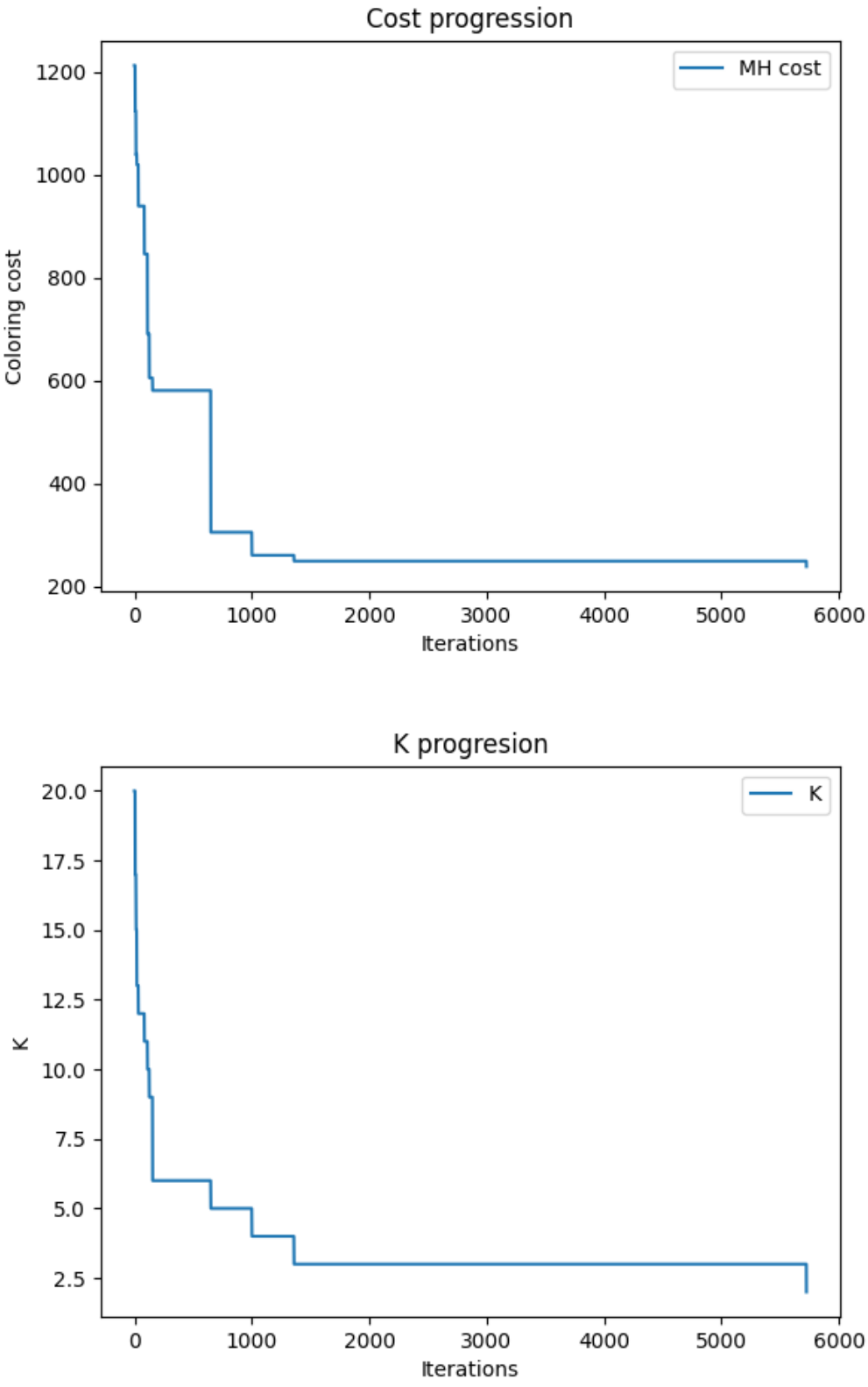


Fig. 3: Resultados de Enfriamiento Simulado

Enfriamiento simulado obtiene la solución óptima, y lo hace en solo 1.64s, completamente aniquilando el resultado de Algoritmo Genético.

Extra: Enfriamiento Simulado para grafos *mucho* más grandes

La familia de grafos de Dorogovtsev-Goltsev-Mendes (DGM) es una familia de grafos recursivamente definidos que son cordales (donde todo ciclo de más de 4 vértices tiene un camino intermedio que conecta dos vértices que no son adyacentes en el ciclo), de ápice (donde eliminar un solo vértice convierte el grafo en planar) y sobre todo, redes de mundo pequeño, donde se puede alcanzar cualquier vértice arbitrario desde cualquier otro en un número relativamente pequeño de saltos (relativo a $|V|$). Se obtuvieron en 2012, como aproximación para redes aleatorias libres de escala (redes donde el grado de cada vértice sigue una cierta ley de potencias o jerarquía), una estructura común en redes sociales, de comercio, o entre neuronas.

Dado que el algoritmo de Enfriamiento Simulado obtiene tan buen resultado, se opta por comprobar su rendimiento frente a un problema de talla *mucho* mayor, DGM(4). Se escoge DGM(4) en particular por ser el grafo mas grande de la familia que Mathematica tiene precomputado y por tanto, podemos calcular el resultado óptimo numericamente.

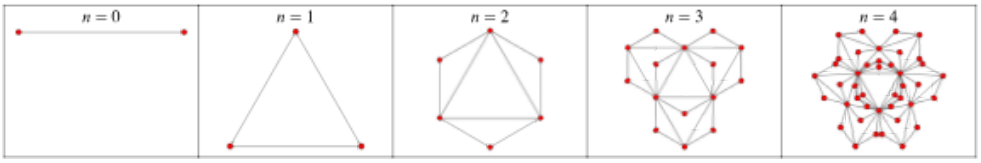


Fig. 4: La familia de grafos DGM(n), hasta n=4. De Wolfram Mathworld

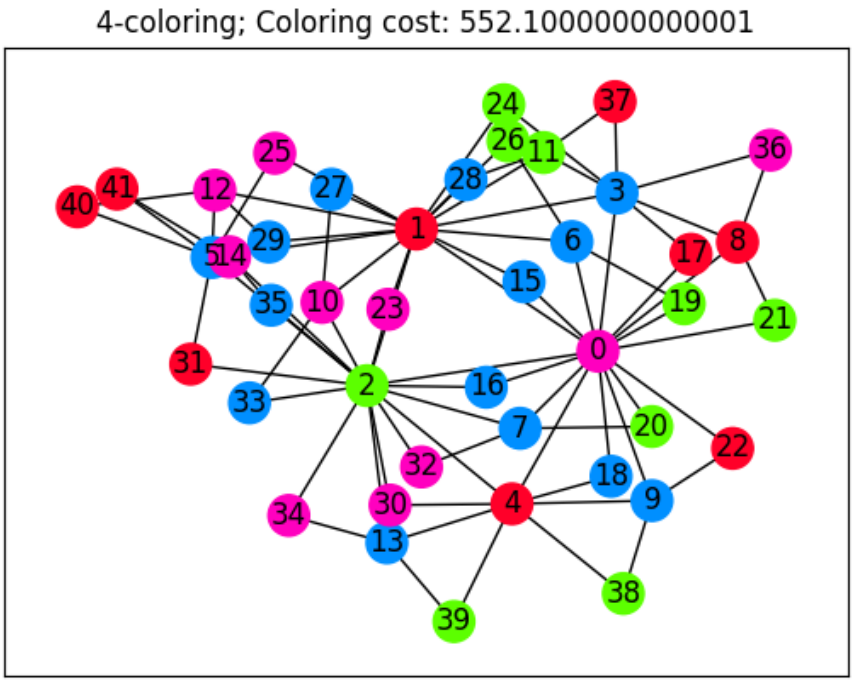
El grafo tiene una talla de $3/2 * (3^{n-1} + 1) = 42$ Vértices.
El número de coloraciones válidas de DGM(4) es
$\sum_{i=0}^{|V|} ||\mathbb{C}_i|| =$
$562663065223563432850816474225838515231324123200000000000000000000000000000000000000000 =$
$5.62663065223563432850816474225838515231324123 2 \times 10^{85}$, es decir, *cincuenta y seis septenvigintillones doscientos sesenta y seis sexvigintillones trescientos seis quinvigintillones quinientos veintidós catrovigintillones trescientos cincuenta y seis trevigintillones trescientos cuarenta y tres duovigintillones doscientos ochenta y cinco unvigintillones ochenta y un vigintillones seiscientos cuarenta y siete novemdecillones cuatrocientos veintidós octodecillones quinientos ochenta y tres septendecillones ochocientos cincuenta y tres sexdecillones quinientos veintitrés quindecillones ciento treinta y dos catordecillones cuatrocientos doce tredecillones trescientos veinte duodecillones*

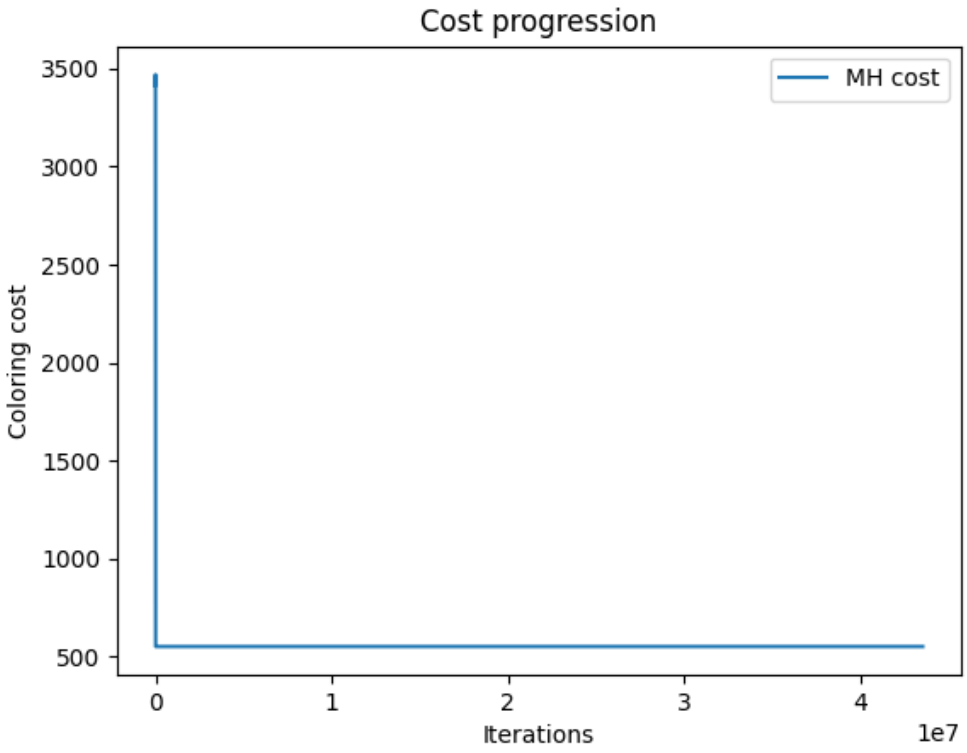
El número cromático de DGM(4), $\chi(DGM(4))$ es 3, por tanto la menor coloración es una 3-coloración. El coste óptimo es 530,6.

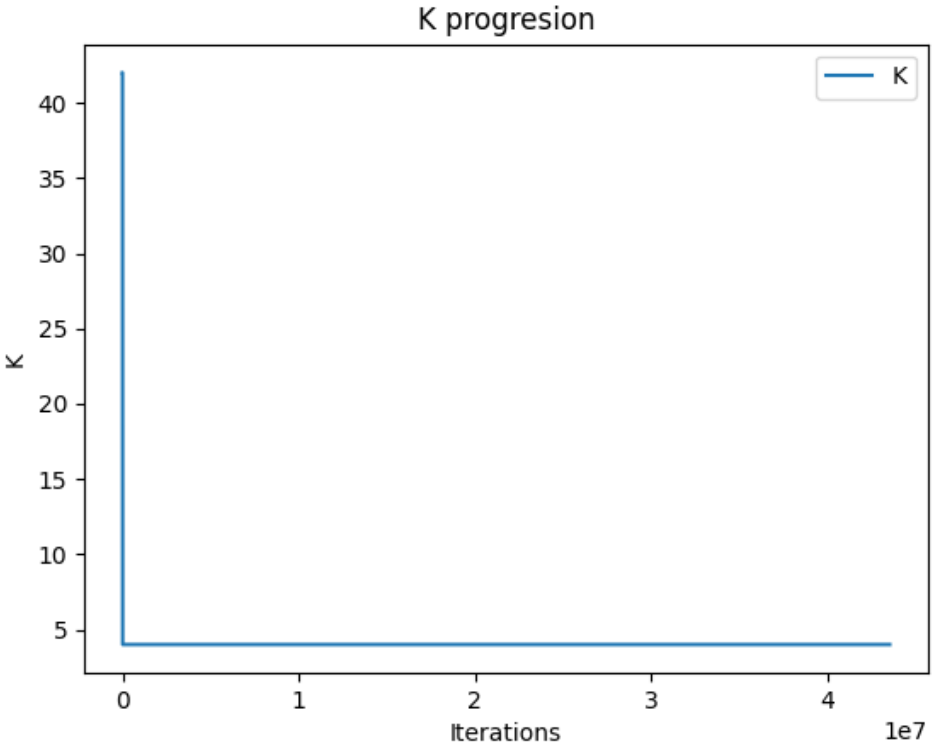
Resultados

Para DGM(4), se realiza una exploración de 5h, con los mismos parámetros que en enfriamiento simulado para el grafo de Desargues, reducción via decay, T=1, k=0.99:

```
================================================================================================
It43566793: Current best=552.100; Predecessor cost=540.100; Successor cost=540.100
------------------------------------------------------------------------------------------------
        Solution summary:
                Best       : 4-colored graph;Proper coloring (No violations); With cost 552.100
                G          : 3-colored graph;IMProper coloring (25 violations); With cost 540.100
                G successor: 3-colored graph;IMProper coloring (25 violations); With cost 542.400

        Cost delta = 0.0;
------------------------------------------------------------------------------------------------
        Update summary:
                No improvement over current; Cost delta = 0.0; Accept chance = 0.9999999904229663; Accepted;
                Current temperature: 2.3185111731694404e-08; Reduction method: decay;
------------------------------------------------------------------------------------------------
        Progress summary:
                        Deadline  =-1 day, 23:59:59.999796 s
                        Iterations=43566793 / inf (0.000%)
                        Optimum estimation: ~ 530.600; distance to estimation: 21.500000000000114
================================================================================================



************************************************************************************************************************
********
Stopped: Reason: Hit time deadline
Best found solution: 552.1000000000001
Best: 4-colored graph;Proper coloring (No violations); With cost 552.100
Best coloring: {0: 3, 1: 0, 2: 1, 3: 2, 4: 0, 5: 2, 6: 2, 7: 2, 8: 0, 9: 2, 10: 3, 11: 1, 12: 3, 13: 2, 14: 3, 15: 2, 16:
2, 17: 0, 18: 2, 19: 1, 20: 1, 21: 1, 22: 0, 23: 3, 24: 1, 25: 3, 26: 1, 27: 2, 28: 2, 29: 2, 30: 3, 31: 0, 32: 3, 33: 2,
34: 3, 35: 2, 36: 3, 37: 0, 38: 1, 39: 1, 40: 0, 41: 0}
Took 5:00:00.000298 s
************************************************************************************************************************
********
```







Vemos que Enfriamiento simulado obtiene una solución optimizada de 552.1 muy rápido pero que no es capaz de mejorarla. La distancia al coste óptimo es de 21.5