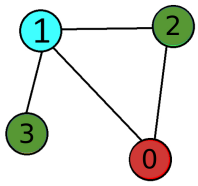
Lab10: Codificar el 3-coloreado de grafos en instancias de SAT

Vamos a ver como los SAT-solvers modernos se pueden usar para resolver problemas difíciles.

El problema y su codificación a SAT

El problema de colorear un grafo con tres colores consiste en decidir si los nodos del grafo pueden colorearse empleando esos tres colores de manera que dos nodos adyacentes no tengan el mismo color. Se sabe que este problema es NP-completo. La codificación del problema de la 3-coloración para el grafo siguiente como una instancia del problema SAT se explica a continuación.



- Supongamos que disponemos de los colores $\{r, g, b\}$ (rojo, verde y azul). El conjunto de variables proposicionales es:

  $$\{x_{0,r}\ ,\ x_{0,v}\ ,\ x_{0,a}\ ,\ x_{1,r}\ ,\ x_{1,v}\ ,\ x_{1,a}\ ,\ x_{2,r}\ ,\ x_{2,v}\ ,\ x_{2,a}\ ,\ x_{3,r}\ ,\ x_{3,v}\ ,\ x_{3,a}\}$$

  Una variable $x_{i,j}$ se evalúa a True si, y sólo si, el nodo i tiene asignado el color j.

- La instancia SAT está formada por las siguientes cláusulas:

  Un conjunto de cláusulas que expresan que cada nodo tiene asignado un único color. Para ello, primero se añaden unas cláusulas expresando que cada nodo tiene asignado al menos un color:

  $$(x_{0,r} \vee x_{0,g} \vee x_{0,b}) \wedge (x_{1,r} \vee x_{1,g} \vee x_{1,b}) \wedge$$

  $$(x_{2,r} \vee x_{2,g} \vee x_{2,b}) \wedge (x_{3,r} \vee x_{3,g} \vee x_{3,b})$$

  y, luego, se añaden las cláusulas expresando que cada nodo tiene asignado a lo sumo un color. Es importante tener en cuenta que $\neg(x_{0,r} \wedge x_{0,g})$ es equivalante a escribir $(\neg x_{0,r} \vee \neg x_{0,g})$.

  $$(\neg x_{0,r} \vee \neg x_{0,g}) \wedge (\neg x_{0,r} \vee \neg x_{0,b}) \wedge (\neg x_{0,g} \vee \neg x_{0,b}) \wedge$$
  $$(\neg x_{1,r} \vee \neg x_{1,g}) \wedge (\neg x_{1,r} \vee \neg x_{1,b}) \wedge (\neg x_{1,g} \vee \neg x_{1,b}) \wedge$$
  $$(\neg x_{2,r} \vee \neg x_{2,g}) \wedge (\neg x_{2,r} \vee \neg x_{2,b}) \wedge (\neg x_{2,g} \vee \neg x_{2,b}) \wedge$$
  $$(\neg x_{3,r} \vee \neg x_{3,g}) \wedge (\neg x_{3,r} \vee \neg x_{3,b}) \wedge (\neg x_{3,g} \vee \neg x_{3,b})$$

Finalmente, se añade otro conjunto de cláusulas expresando que nodos adyacentes tienen asignados colores diferentes.

$$(\neg x_{0,r} \vee \neg x_{1,r}) \wedge (\neg x_{0,g} \vee \neg x_{1,g}) \wedge (\neg x_{0,b} \vee \neg x_{1,b}) \wedge$$
$$(\neg x_{0,r} \vee \neg x_{2,r}) \wedge (\neg x_{0,g} \vee \neg x_{2,g}) \wedge (\neg x_{0,b} \vee \neg x_{2,b}) \wedge$$
$$(\neg x_{1,r} \vee \neg x_{2,r}) \wedge (\neg x_{1,g} \vee \neg x_{2,g}) \wedge (\neg x_{1,b} \vee \neg x_{2,b}) \wedge$$
$$(\neg x_{1,r} \vee \neg x_{3,r}) \wedge (\neg x_{1,g} \vee \neg x_{3,g}) \wedge (\neg x_{1,b} \vee \neg x_{3,b})$$

Observa que, a partir de una interpretación que satisface la fórmula, se puede generar una coloración válida: si la variable $x_{i,j}$ tiene asignado el valor True, se asigna el color j al nodo i.

Tarea a realizar

1. Debes implementar la función `reduce_3colorable_to_SAT(graph)` que recibe como entrada un grafo con el formato habitual. Esta función debe generar la CNF, φ, que se obtiene cuando se reduce el problema de la 3-coloración según la explicación del apartado anterior. Como las variables de φ son números positivos, puedes considerar que el color 'r' tiene código 0, 'g' tiene código 1 y 'b' tiene código 2.

   Además dispones de una función biyectiva `var2positive(i,j)`, que asocia un número positivo a la variable $x_{i,j}$. En concreto `var2positive(i,j)` $= 3 * i + j + 1$. Por otro lado, la función `positive2var(m)` hace la operación contraria: dado un positivo m, devuelve los dos valores (i, j) de la variable asociada a m.

   Cuando hayas generado φ, hay que llamar al solver de Pysat (en este caso al solver Mergesat3) y obtener una asignación cuando φ es satisfactible. Esto lo hace la función `visualizeGXGraph` del fichero `colour_tools`, que ya está implementada. Esta función además se encarga de dibujar la solución del problema a partir de la asignación obtenida. Cuando φ es satisfactible puedes ver el coloreado del grafo en la ventana `Plots`.

2. Usa tu SAT-solver para ver cómo resuelve este problema. Para ello tendrás que cambiar la función `visualizeGXGraph` del fichero `colour_tools` donde llamarás a tu SAT-solver.