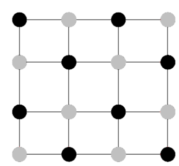
Algorithmique (AL5)
TD2, Exercice 5 (Correction)

Question 1. Déterminer si la maison est un graphe biparti.

La maison a un triangle (cycle impair) donc elle n'est pas un graphe biparti (voir la Question 3).

Question 2. Même question pour la grille $G_{4,4}$.

Une bicoloration de la grille $G_{4,4}$ se trouve ci-dessous :



Question 3. Montrer qu'un graphe est biparti $\Leftrightarrow$ il n'a pas de cycle de longueur impaire. La longueur d'un cycle est le nombre d'arêtes qui le composent.

$\Rightarrow$) Si un graphe est biparti alors il n'a pas de cycle de longueur impaire.

Soit $v1, v2, \ldots, v_k$ un cycle arbitraire de G, soit $v_1 \in V_1$. Vu que toute arête connecte un sommet de V_1 à un sommet de V_2, $v_2 \in V_2, v_3 \in V_1$, etc. Alors, $v_k \notin V_1$ car $(v_1, v_k) \in E$, donc la longueur du cycle est paire.

$\Leftarrow$) Si un graphe n'a pas de cycle de longueur impaire alors il est biparti.

Lemma 3.1. *Soit $Dist$ le tableau de distances d'un parcours en largeur du graphe G depuis un sommet v. Si $(x, y) \in E$ alors $|Dist(x) - Dist(y)| \leq 1$.*

Proof. Soit $(x, y) \in E$ tel que $|Dist(x) - Dist(y)| > 1$, et soit $Dist(x) < Dist(y)$. Alors, x devient *gris* avant y. Or y est voisin de x, donc quand on explore x, $Couleur(y) = blanc$ et donc le parcours visite y et met $Dist(y) = Dist(x) - 1$. Contradiction. □

On fait un parcours en largeur du G par un sommet arbitraire $v \in V$. On fait une répartition des sommets comme suite : pour un sommet x, si $Dist(x) \bmod 2 = 0$ alors $x \in V_1$, si non $x \in V_2$.

Prouvons que la répartition est correcte. Supposons que ce n'est pas le cas, il y a x, y dans la même partie (disons V_1) et $(x, y) \in E$. Par Lemma 3.1 on a $|Dist(x) - Dist(y)| \leq 1$, et vu que $x, y \in V_1$ $Dist(x) = Dist(y)$. On trouve a l'ancêtre commun de x, y dans l'arbre du parcours ayant $Dist(a)$ maximale. Alors il existe un chemin $a \to x$ et un chemin $a \to y$, qui consistent seulement d'arêtes de l'arbre. On a $Dist(x) - Dist(a) = Dist(y) - Dist(a) := k$. Alors, la longueur du cycle $a \to x, (x, y), y \to a$ est $k + 1 + k$ qui est impaire. Contradiction.

Question 4. Proposer un algorithme qui prend en entrée un graphe, qui teste si le graphe est biparti, et qui renvoie un cycle impair sinon. On pourra se servir du parcours en largeur (BFS) vu en cours.

On exécute la fonction Bip(G,s) sur un sommet $s \in V$ choisi au hasard. Cette fonction fait grosso-modo un parcours largeur depuis s, en utilisant aussi un tableau `Partie` qui prend les valeurs 0 ou 1 (0 indique V_1 et 1 indique V_2). Tant que les voisins d'un sommet x en train d'être traité sont *blanc*, la fonction met les voisins dans la partie à la quelle x n'appartient pas. Si on trouve un voisin y de x qui n'est pas *blanc* et qui appartient à la même partie où on a mis x, on a trouvé un cycle impair : on exécute la procédure AfficherCycle(x,y,Π) pour l'afficher et on renvoie false. Si Bip(G,s) termine bel et bien, le graphe est biparti, on renvoie true.

```
Fonction Bip(G,s):
  pour chaque x ∈ V\{s} faire
    Couleur[x] := blanc; Π[x] := nil; Dist[x] := inf; Partie[x] := nil;
  Couleur[s] := gris; Π[s] := nil; Dist[s] := 0; Partie[s] = 0;
  F := File vide;        // File = structure FIFO
  Ajouter(F,s);
  tant que F != ∅ faire
    x := ExtraireTête(F);
    pour chaque (x,y) ∈ E faire
      si Couleur[y] = blanc alors
        Couleur[y] := gris;
        Dist[y] := Dist[x] + 1;
        Π[y] := x;
        Partie[y] := 1 - Partie[x];
        Ajouter(F,y);
      else
        si Partie[x] = Partie[y]
          AfficherCycle(x,y,Π);
          return false;
    Couleur[x] := noir;
  return true;

Procédure AfficherCycle(x,y,Π) :
  F := File vide;        // File = Structure FIFO
  P := Pile vide;        // Pile = Structure LIFO
  a1 = x; a2 = y;
  tant que a1 != a2 faire
    Ajouter(F,a1); a1 := Π(a1);
    Ajouter(P,a2); a2 := Π(a2);
  Affiche(a1);
  tant que P != ∅ faire
    Affiche(ExtraireTête(P));
  tant que F != ∅ faire
    Affiche(ExtraireTête(F));
  Affiche(a1);
```

Question 5. Évaluer la complexité de votre algorithme.

L'algorithme prend temps $O(|V| + |E|)$: AfficherCycle(x,y,Π) prend temps $O(|V|)$ alors que le reste de Bip(G,s) n'est qu'un BFS.