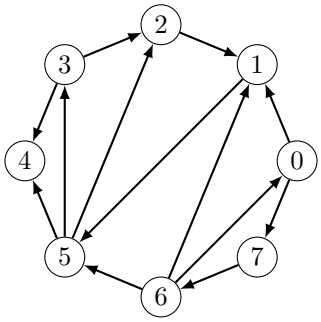
1 Parcours de graphe

Exercice 1

Déterminer l'ordre de parcours des sommets lors de l'exécution d'un DFS et d'un BFS en partant du sommet 0 dans le graphe suivant. On supposera choisir d'explorer le sommet de plus petit numéro en premier en cas de choix multiples.



Corrigé

On obtient $(0, 1, 5, 2, 3, 4, 7, 6)$ pour le DFS et $(0, 1, 7, 5, 6, 2, 3, 4)$ pour le BFS.

Exercice 2

Soit $G = (S, A)$ un graphe implémenté par tableau de listes d'adjacence.

1. Montrer que la complexité temporelle d'un DFS ou BFS est $\mathcal{O}(|S| + |A|)$.
2. On suppose G non orienté. On note $C = (S_0, A_0)$ la composante connexe d'un sommet $s_0 \in S$. Parmi les propositions suivantes, déterminer en justifiant la complexité temporelle d'un parcours (en profondeur ou largeur) depuis s_0 :
   - $\Theta(|S| + |A|)$;
   - $\Theta(|S_0| + |A|)$;
   - $\Theta(|S| + |A_0|)$;
   - $\Theta(|S_0| + |A_0|)$.

Corrigé

1. La création du tableau de booléens est en $\mathcal{O}(|S|)$. Lors du parcours, la boucle `for` à l'intérieur du `while` parcourt au plus une fois et une seule chaque liste d'adjacence. Les autres opérations sont en temps constant. La complexité totale est donc en $\mathcal{O}\left(|S| + \sum_{s \in S} \deg(s)\right) = \mathcal{O}(|S| + |A|)$.
2. La complexité est en $\Theta(|S| + |A_0|)$. La création du tableau de booléen a toujours lieu. Seules les listes d'adjacence des sommets de S_0 sont explorées, d'où le résultat.

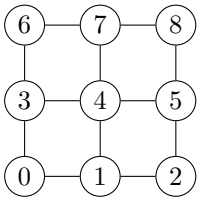
Exercice 3

On reprend l'algorithme général de parcours (WFS pour *WhateverFirstSearch*), en le modifiant légèrement :

```
Fonction WFS1(G, s_0)
    Mettre s_0 dans le sac.
    Tant que le sac n'est pas vide Faire
        Sortir un élément s du sac.
        Si s n'est pas marqué Alors
            Marquer s.
            Pour tout t voisins de s Faire
                Mettre t dans le sac.
```

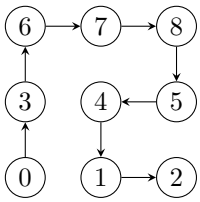
```
Fonction WFS2(G, s_0)
    Mettre s_0 dans le sac.
    Marquer s_0.
    Tant que le sac n'est pas vide Faire
        Sortir un élément s du sac.
        Pour tout t voisins de s Faire
            Si t n'est pas marqué Alors
                Mettre t dans le sac.
                Marquer t.
```

1. On suppose que le sac est une file. Déterminer en justifiant si les données suivantes sont modifiées entre les deux parcours :
   (a) l'ordre de parcours ;
   (b) la complexité temporelle ;
   (c) la complexité spatiale.
2. On suppose que le sac est une pile. En appliquant WFS1 et WFS2 au graphe suivant, déterminer l'ordre de parcours et l'arborescence du parcours. On supposera que les listes d'adjacence sont triées par ordre croissant.

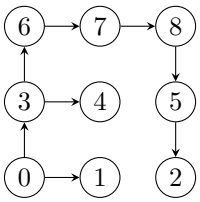


Corrigé

1. (a) l'ordre de parcours ne change pas car les sommets sont ajoutés à la première fois à la file dans le même ordre (c'est le principe de la file) ;
   (b) si la file est correctement implémentée (opérations en $\mathcal{O}(1)$), la complexité temporelle ne change pas, le raisonnement de l'exercice précédent reste le même ;
   (c) la complexité spatiale peut cependant être potentiellement plus élevée pour `WFS1` que pour `WFS2` : la file de `WFS2` ne peut pas contenir de doublon d'après le principe du marquage. Sa taille est donc $\mathcal{O}(|S|)$. Cependant, la file de `WFS1` peut contenir plusieurs fois le même sommet, sa taille peut être de l'ordre du nombre d'arêtes.
2. On obtient les arborescences et ordres suivants :



`WFS1` : $(0, 3, 6, 7, 8, 5, 4, 1, 2)$



`WFS2` : $(0, 3, 6, 7, 8, 5, 2, 4, 1)$

Le `WFS1` peut être considéré comme un parcours en profondeur (l'arbre le plus profond), mais pas le `WFS2`.

Exercice 4

Soit $G = (S, A)$ un graphe orienté. On suppose que pour tout $(s, t) \in S^2$, exactement une arête parmi (s, t) et (t, s) est dans A.

1. Montrer qu'il existe un chemin hamiltonien dans G, c'est-à-dire un chemin qui passe une fois et une seule par chaque sommet.
2. Déterminer un algorithme pour trouver un tel chemin.
3. Montrer que ce chemin est unique si et seulement si G ne possède pas de cycle de taille 3.

Corrigé

1. On montre ce résultat par récurrence sur $n = |S|$:
   - pour $n = 1$ ou $n = 2$, le résultat est vrai d'après les hypothèses ;
   - supposons le résultat établi pour tout graphe d'ordre $n \geqslant 2$. Soit $G = (S, A)$ un graphe d'ordre $n + 1$ ($S = \{0, \ldots, n\}$) vérifiant les hypothèses. Par hypothèse de récurrence, il existe un chemin $(s_0, s_1, \ldots, s_{n-1})$ qui passe une fois et une seule par chaque sommet de $\{0, \ldots, n-1\}$. Montrons qu'on peut « insérer » le sommet n dans ce chemin en distinguant les cas :
     * si $(n, s_0) \in A$, alors $(n, s_0, \ldots, s_{n-1})$ est un chemin hamiltonien dans G ;
     * sinon, $(s_0, n) \in A$; si $(n, s_1) \in A$, alors $(s_0, n, s_1, \ldots, s_{n-1})$ est un chemin hamiltonien dans G ;
     * ...
     * sinon, $(s_{n-1}, n) \in A$, alors $(s_0, \ldots, s_{n-1}, n)$ est un chemin hamiltonien dans G ;

     Il suffit donc de trouver le plus petit i (potentiellement n s'il n'y en a pas) tel que $(n, s_i) \in A$. Dès lors, $(s_0, \ldots, s_{i-1}, n, s_i, \ldots, s_{n-1})$ sera un chemin hamiltonien.

   On conclut par récurrence.
2. Ici, il serait plus efficace de travaille avec une matrice d'adjacence, car c'est l'existence des arêtes qui nous intéresse plus que le voisinage. On peut suivre le principe décrit plus haut.

   Entrée : Graphe $G = (S, A)$ vérifiant les hypothèses donné sous forme de matrice d'adjacence
   Début algorithme
   - $n \leftarrow |S|$
   - *chem* $\leftarrow [0]$
   - **Pour** $s = 1$ à $n - 1$ **Faire**
     - Insérer s dans *chem* avant la première valeur t telle que $(s, t) \in A$
   - **Renvoyer** *chem*

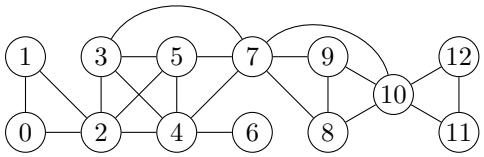
3. Sens direct : par contraposée, supposons que G possède un cycle de taille 3 (u, v, w). Dans l'algorithme précédent, on remarque que l'ordre relatif des sommets ne change pas au cours de la boucle `Pour`. Ainsi, on pourrait commencer par (u, v, w) ou (v, w, u) puis « insérer » les autres sommets et obtenir deux chemins hamiltoniens distincts.

   Sens réciproque : par contraposée, supposons qu'il existe deux chemins hamiltoniens distincts $c_1 = (s_0, \ldots, s_{n-1})$ et $c_2 = (t_0, \ldots, t_{n-1})$. Soit i l'indice minimal tel que $s_i \neq t_i$. Remarquons tout d'abord que $i < n-2$. En effet, si $i = n-1$, alors les deux chemins ont $n-1$ sommets en commun mais pas le dernier, ce qui est absurde. Sinon, $i = n-2$, et alors (s_i, t_i) et (t_i, s_i) sont dans A, ce qui est absurde. Dès lors :
   - si $(s_{i+2}, s_i) \in A$, alors (s_i, s_{i+1}, s_{i+2}) est un 3-cycle ;
   - sinon, $(s_i, s_{i+2}) \in A$ et alors si $(s_{i+3}, s_i) \in A$, alors (s_i, s_{i+2}, s_{i+3}) est un 3-cycle ;
   - en poursuivant le raisonnement, on arrive à la conclusion que soit il existe un 3-cycle, soit $(s_i, s_j) \in A$ pour tout $i < j$;
   - en faisant le même raisonnement dans c_2, on arrive à la même conclusion. Sachant qu'il existe $i < j$ et $i < k$ tels que $s_i = t_j$ et $t_i = s_j$, on trouve nécessairement un 3-cycle.

Exercice 5

Soit $G = (S, A)$ un graphe non orienté connexe. On dit que $s \in S$ est un **sommet coupant** si $G[S \setminus \{s\}]$ n'est pas connexe.

1. Déterminer les sommets coupants du graphe suivant :



2. Décrire un algorithme linéaire qui détermine, étant donné G et s, si s est un sommet coupant ou non.
3. En déduire un algorithme naïf qui détermine tous les sommets coupants d'un graphe G donné et déterminer sa complexité.
4. Soit T l'arborescence d'un DFS de G depuis un sommet s quelconque.
   (a) Montrer que s est un sommet coupant de G si et seulement si s est de degré $\geqslant 2$ dans T.
   (b) Montrer qu'un sommet $t \in S$ est un sommet coupant de G si et seulement si t possède un fils u tel qu'aucun descendant (dans T) de u n'est voisin (dans G) d'un ancêtre strict (dans T) de t.
   (c) En déduire un algorithme linéaire qui détermine tous les sommets coupants d'un graphe G donné.

Corrigé

1. Les sommets coupants sont $\{2, 4, 7, 10\}$.
2. On construit une copie de G sans s et on vérifie si le graphe obtenu est connexe. Ces deux opérations sont bien en temps linéaire.
3. On fait le test précédent pour chaque sommet du graphe. La complexité totale est en $\mathcal{O}(|S|(|S| + |A|)) = \mathcal{O}(|S||A|)$ puisque le graphe est connexe.
4. Pour montrer (a) et (b), on peut commencer par montrer un lemme :

Lemme

Si s est un nœud de T ayant pour fils u et v, alors il n'existe aucune arête entre un descendant de u et un descendant de v.

Preuve

Ce résultat découle du principe du parcours en profondeur. Sans perte de généralité, supposons que $pre(u) < pre(v)$. Cela implique également $post(u) < pre(v)$ (par définition des parcours préfixes et postfixes). Soit x un descendant de u et y un descendant de v. S'il existe une arête $\{x, y\}$, alors le sommet y aurait été empilé pendant l'appel à `DFS(x)`. On aurait alors $post(y) < post(x) < post(u) < post(y)$, ce qui est absurde.

   (a) Par le lemme, si s a au moins deux fils, il n'existe pas d'arêtes entre des sommets de leurs descendants. Tout chemin entre de tels descendants dans deux fils distincts doit donc forcément passer par s qui est donc coupant. Réciproquement, si s n'a qu'un seul fils (ou aucun, ce qui est un cas trivial), alors sa suppression ne brise pas la connexité (il existe toujours des chemins entre deux sommets quelconques, donnés par $T \setminus \{s\}$).
   (b) Sens direct : par contraposée, supposons que tous les fils de t ont un descendant voisin d'un ancêtre strict de t. Soit $(u, v) \in (S \setminus \{t\})^2$. Montrons qu'il existe toujours un chemin de u à v ne passant pas par t. Distinguons :

- si ni u, ni v n'est un descendant de t ou si u et v sont descendants du même fils de t, alors T donne déjà un chemin qui ne passe pas par t ;
- si u est un descendant de t mais pas v (ou l'inverse), alors soit w le fils de t qui est un ancêtre de u et soit $\{s_1, s_2\}$ l'arête entre un descendant de w et un ancêtre strict de t. Alors $u \rightsquigarrow s_1 \to s_2 \rightsquigarrow v$ est un chemin dans G ne passant pas par t (les $\rightsquigarrow$ désigne les chemins dans T, qui sont uniques) ;
- si u et v sont descendants de t mais pas du même fils de t, soit w et x les fils de t dont u et v sont descendants, et soient $\{s_1, s_2\}$ l'arête entre un descendant de w et un ancêtre strict de t et $\{s_3, s_4\}$ l'arête entre un descendant de x et un ancêtre strict de t. Alors $u \rightsquigarrow s_1 \to s_2 \rightsquigarrow s_4 \to s_3 \rightsquigarrow v$ est un chemin dans G ne passant pas par t.

On en déduit que t n'est pas un sommet coupant.

Sens réciproque : supposons que t possède un fils u tel qu'aucun descendant de u n'est voisin d'un ancêtre strict de t. Soit alors x un descendant de u et y un sommet qui n'est pas un descendant de u. Supposons $y \neq t$ et montrons qu'il n'existe pas d'arête $\{x, y\}$.

- si y est un ancêtre strict de t, alors par hypothèse $\{x, y\}$ ne peut pas exister ;
- si y n'est pas un ancêtre strict de t, alors il existe un ancêtre commun z à x et y, qui possède deux fils distincts dont x et y sont descendants. D'après le lemme, $\{x, y\}$ ne peut pas exister.

Si on suppose qu'il existe un chemin $(s_0, \ldots, s_k)$ de x à y, alors il existe $i \in [\![0, k]\!]$ minimal tel que s_i n'est pas un descendant de u. On vient de montrer que $s_i = t$, donc tous les chemins de x à y passent par t qui est donc coupant.

(c) Lors de l'exécution du parcours, on peut garder en mémoire l'existence de ces arêtes arrière (une arête vers un sommet qui a un plus petit numéro préfixe). En gardant en mémoire l'ancêtre le plus haut vers lequel on a pointé, on peut mettre le résultat à jour pour un nœud en fonction des ancêtres pointés par les descendants de ses fils. Un sommet coupant sera un sommet s dont aucun fils n'a d'arête arrière vers un ancêtre précédant s dans le parcours. On peut traiter à part le cas de la racine. Les opérations supplémentaires ne changent pas la complexité du parcours.

2 Arbres couvrants

Exercice 6

Soit $G = (S, A)$ un graphe non orienté connexe et $s \in S$. Montrer que si l'arborescence d'un DFS de G depuis s est la même que l'arborescence d'un BFS de G depuis s, alors G est un arbre.

Corrigé

On montre la contraposée. Supposons que G possède un cycle. Considérons un cycle de taille minimale $(s_1, \ldots, s_k)$, avec $k \geqslant 3$. Comme le cycle est de taille minimale, il n'existe pas d'arête entre deux sommets non consécutifs du cycle. Sans perte de généralité, supposons que s_1 soit le premier sommet exploré par le BFS. Alors nécessairement s_2 et s_k seront deux fils de s_1 dans l'arborescence du BFS. Or, sachant qu'il existe un chemin de s_2 à s_k passant par des sommets non explorés, l'un sera nécessairement un descendant de l'autre dans l'arborescence du DFS, ce qui conclut.

Exercice 7

Montrer qu'un arbre avec au moins deux sommets possède au moins deux sommets de degré 1.

Corrigé

On peut montrer le résultat par récurrence sur n le nombre de sommets :

- c'est vrai pour $n = 2$ (il n'y qu'un seul arbre à deux sommets) ;
- supposons le résultat vrai pour $n \geqslant 2$ fixé. Soit $G = (S, A)$ un arbre à $n + 1$ sommets. On a déjà prouvé que G étant sans cycle, il possède au moins un sommet s de degré 0 ou 1. Le cas 0 est impossible car le graphe est connexe. En supprimant ce sommet de G, par hypothèse de récurrence, il existe u et v deux sommets de degré 1. L'un de ces deux sommets n'était pas adjacent à s dans G, donc il était de degré 1 dans G.

On conclut par récurrence.

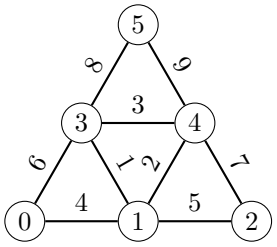
Exercice 8

Soit $G = (S, A, f)$ un graphe non orienté pondéré connexe tel que $|A| \geqslant 3$. On suppose f injective (tous les poids d'arêtes sont distincts). Soit T l'unique arbre couvrant minimal de G.

1. Montrer que T contient les deux arêtes de poids minimal de A.
2. Montrer par un contre-exemple que T ne contient pas nécessairement les trois arêtes de poids minimal de A.
3. Montrer que pour pour chaque cycle de G, T ne contient pas l'arête de poids maximal du cycle.
4. T contient-il l'arête de poids minimal de chaque cycle ?

Corrigé

1. Supposons que T ne contienne pas l'arête de poids minimal. En rajoutant cette arête à T, on crée un cycle. On peut donc supprimer une autre arête pour obtenir un nouvel arbre couvrant qui est de poids strictement inférieur. Donc T contient l'arête de poids minimal. Supposons que T ne contienne pas la deuxième arête de poids minimal. En rajoutant cette arête à T, on crée un cycle de taille au moins 3. On peut donc supprimer une autre arête, qui n'est pas celle de poids minimal, pour obtenir un nouvel arbre couvrant qui est de poids strictement inférieur. Donc T contient la deuxième arête de poids minimal.
2. Un 3-cycle convient.
3. Soit a une arête de poids maximal d'un cycle de G. Supposons que T contienne a. En supprimant a, on obtient deux composantes connexes. On peut alors rajouter une des arêtes du cycle contenant a pour obtenir un nouvel arbre couvrant de poids inférieur, ce qui est absurde.
4. Non, par exemple dans le graphe suivant :



Les trois arêtes centrales sont chacune l'arête de poids minimal d'un cycle du graphe, mais forment un cycle donc ne peuvent pas être simultanément présentes dans un arbre couvrant.

Exercice 9

Soit $G = (S, A, f)$ un graphe non orienté pondéré tel que f est injective et T son unique arbre couvrant minimal. On note T_2 le deuxième arbre couvrant de poids minimal après T.

1. Montrer que T_2 et T ont $|S| - 2$ arêtes en commun.

2. En déduire un algorithme pour calculer T_2.

Corrigé

1. Notons $T = (S, B)$ et $T_2 = (S, B_2)$. Supposons que $|B \Delta B_2| > 2$. Notons a l'arête de poids minimal de $B \Delta B_2$. Si on suppose que $a \in B_2$, alors en rajoutant a à T, on crée un cycle. Il existe une arête a' de $B \setminus B_2$ dans ce cycle. Si on la supprime, on obtient un arbre couvrant de poids strictement inférieur à $f(T)$, ce qui est absurde. On en déduit $a \in B \setminus B_2$. En rajoutant a à T_2, on crée un cycle. Il existe une arête $a' \in B_2 \setminus B$ dans ce cycle. Si on la supprime, on obtient un arbre couvrant de poids strictement inférieur à $f(T_2)$, mais différent de T, ce qui est absurde. On en déduit que $|B \Delta B_2| = 2$ (ne peut pas être zéro par unicité de T).
2. Il s'agit de trouver l'arête à supprimer de T et celle à rajouter. On peut commencer par calculer T avec l'algorithme de Kruskal puis, pour chaque arête $a \in B$, on supprime a de G temporairement et on relance l'algorithme de Kruskal. On garde l'arbre de poids minimal parmi ceux obtenus. La complexité serait en $\mathcal{O}(|S||E|\alpha(|S|))$.

3 Graphes bipartis et couplages

Exercice 10

Montrer qu'un arbre est biparti.

Corrigé

Soit $G = (S, A)$ un arbre et $s_0 \in S$ un sommet quelconque. On pose $X = \{s \in S \mid d(s_0, s) \text{ est pair}\}$ et $Y = \{s \in S \mid d(s_0, s) \text{ est impair}\}$. Montrons qu'il n'existe pas d'arête entre deux sommets de X ou deux sommets de Y.

Par l'absurde, supposons $(s, t) \in X^2$ tel que $\{s, t\} \in A$. Par définition de la distance, on a $d(s_0, s) = d(s_0, t)$ (sinon la différence entre les distances doit être d'au moins 2, ce qui est impossible car il existe une arête entre les deux sommets). De plus, l'arête $\{s, t\}$ n'est pas sur un plus court chemin de s_0 à s (ni de s_0 à t). On en déduit l'existence d'un cycle dans le graphe, ce qui est contradictoire. On raisonne de même pour Y.

Exercice 11

Décrire un algorithme qui prend en argument un graphe G non orienté et détermine si G est biparti ou non (on ne demande pas de le programmer, mais une description en français ou pseudo-code). Déterminer la complexité de cet algorithme.

Corrigé

On suppose le graphe connexe, sinon on applique l'algorithme suivant dans chaque composante connexe (en utilisant le même tableau de numéros). On attribue un entier 1 ou 2 à chaque sommet selon le principe suivant :

```
Entrée : Graphe S = (S, A) non orienté connexe, sommet s_0 ∈ S.
Début algorithme
    Poser num(s_0) = 1.
    Mettre (s_0, 1) dans un sac.
    Tant que le sac n'est pas vide Faire
        Sortir un couple (s, c) du sac.
        Pour tout t voisins de s Faire
            Si num(t) = c Alors
                Renvoyer Faux
            Sinon si t n'a pas de numéro Alors
                Poser num(t) = 3 − c
                Mettre (t, 3 − c) dans le sac.
    Renvoyer Vrai
```

Exercice 12

Montrer qu'un graphe G est biparti si et seulement si G ne contient aucun cycle de longueur impaire.

Corrigé

Sens direct : par contraposée, supposons que G possède un cycle de longueur impaire $(s_1, \dots, s_{2k+1})$. Alors pour toute partition $X \sqcup Y$ de S, il existe nécessairement deux sommets consécutifs qui sont dans le même ensemble, donc G n'est pas biparti.

Sens réciproque : c'est le même raisonnement que pour montrer qu'un arbre est biparti : on pose $X = \{s \in S \mid d(s_0, s) \text{ est pair}\}$ et $Y = \{s \in S \mid d(s_0, s) \text{ est impair}\}$ et on montre qu'il n'y a pas d'arête entre deux sommets d'une même partie (sinon il existerait un cycle de longueur impaire).

Exercice 13

Un échiquier $n \times n$ contient des pièces d'échec sur certaines cases. On veut savoir s'il est possible de recouvrir toutes les cases inoccupées en plaçant des dominos 2×1 de telle sorte que chaque domino recouvre exactement deux cases inoccupées.

Décrire un algorithme qui répond au problème.

Corrigé

On crée un graphe dont les sommets sont les cases inoccupées et dont les arêtes existent entre deux cases voisines (pas en diagonale). Le graphe obtenu est biparti (cases blanches/cases noires). On cherche s'il existe un couplage parfait dans ce graphe. On peut utiliser l'algorithme du cours.

Exercice 14

Soit $G = (S, A)$ un graphe orienté. On appelle **couverture par cycle** de G un ensemble de cycles de G dont les sommets forment une partition de S (chaque sommet $s \in S$ appartient à exactement un cycle de la couverture).

1. Soit $A' \subseteq A$. Montrer l'équivalence entre les deux propositions suivantes :
   (a) chaque sommet de (S, A') est de degré entrant et sortant égal à 1 ;
   (b) (S, A') est une couverture par cycle de G.
2. En utilisant le problème du couplage maximum, décrire un algorithme qui détermine s'il existe

une couverture par cycle de G et en renvoie une le cas échéant.

Corrigé

1. – (a)⇒(b) : on peut construire explicitement les cycles ; en partant d'un sommet s_0, il existe un unique voisin s_1, qui possède un unique voisin s_2... Ce processus termine sur un cycle en un nombre fini d'étape. Il n'existe aucune arête entre un sommet du cycle et un autre (ou l'inverse), ni entre deux sommets non consécutifs du cycle. On peut répéter jusqu'à obtenir tous les cycles.
   – (b)⇒(a) : dans un cycle, chaque sommet a un degré entrant et sortant égal à 1. C'est le cas pour tous les cycles car ils sont disjoints.
2. Si on suppose $S = \{0, \dots, n-1\}$, alors on crée un graphe biparti $G' = (S', A')$ de la manière suivante :
   – $S' = X \cup Y$ où $X = \{s_0, \dots, s_{n-1}\}$ et $Y = \{t_0, \dots, t_{n-1}\}$;
   – $\{s_i, t_j\} \in A'$ si et seulement si $(i, j) \in A$.

   Alors il existe une couverture par cycle de G si et seulement s'il existe un couplage parfait dans G' : dans G', une arête $\{s_i, t_j\}$ représente un degré sortant de 1 pour i et un degré entrant de 1 pour j.

Exercice 15

Soit $G = (X \cup Y, A)$ un graphe biparti. Montrer le théorème de Hall : G admet un couplage parfait si et seulement si pour tout $U \subseteq X$, $|U| \leqslant |N(U)|$, où $N(U)$ désigne l'ensemble des voisins des sommets de U.

Corrigé

Sens direct : Soit C un couplage parfait de G et $U \subseteq X$. Soit V l'ensemble des voisins d'un sommet de U par une arête de C. Le couplage étant parfait, $|V| = |U|$. Mais comme $V \subseteq N(U)$, on a bien $|N(U)| \geqslant |V| = |U|$.

Sens réciproque : par contraposée, supposons que G ne possède pas de couplage parfait et soit C un couplage maximum de G. Soit $s \in X$ un sommet libre pour C. Notons Z l'ensemble des sommets t de S tels qu'il existe un chemin alternant pour C de s à t. Notons $U = X \cap Z$ et $V = Y \cap Z$. Remarquons qu'un chemin alternant de longueur maximale termine nécessairement dans U (sinon le chemin serait augmentant et C ne serait pas maximum). On en déduit que chaque sommet de V est apparié à un sommet de U (autre que s qui est le seul non apparié), et donc $|V| = |U| - 1$. Par ailleurs, $N(U) \subseteq V$: en effet, un voisin d'un sommet de U par une arête hors de C est dans V car il prolonge un chemin alternant, et un voisin par une arête de C est dans V par l'association précédente. On en déduit $|N(U)| \leqslant |V| < |U|$, ce qui conclut.

Exercice 16

Soit $G = (X \cup Y, A)$ un graphe biparti tel que $|X| = |Y| = k$. Montrer que si chaque sommet est de degré au moins $\frac{k}{2}$, alors G possède un couplage parfait.

Corrigé

Supposons que G ne possède pas de couplage parfait. D'après le théorème de Hall, il existe $U \subseteq X$ tel que $|N(U)| < |U|$. Comme $U \neq \emptyset$, $|N(U)| \geqslant \frac{k}{2}$. Par ailleurs, soit $V = Y \setminus N(U)$. Alors clairement $N(V) \subseteq X \setminus U$. Comme $V \neq \emptyset$, on en déduit que $|X \setminus U| \geqslant |N(V)| \geqslant \frac{k}{2}$. Finalement, $|X| = |U| + |X \setminus U| > k$ ce qui est absurde.