

Parcours Eulériens

Michel Habib

habib@irif.fr

<http://www.irif.fr/~habib>

Cachan, décembre 2017

D'un joli théorème a un bel algorithme

Théorème Euler, 1735

Si G est un graphe connexe. G admet un sentier fermé ssi tous les degrés sont pairs.

L'algorithme très simple de Fleury, cité en 1891 par Lucas,
un parcours en profondeur sur les arêtes

Algorithme 1: DFS-cycle-eulerien**Entrées :** G : graphe non vide, fini, sans boucle, connexe . x_0 : un sommet de G .**Résultat :** (r, w) : où r est un booléen indiquant si G est eulérien ou non et w est ou bien une liste de sommets représentant un cycle eulérien dans le cas où G est eulérien ou bien un sommet de degré impair.**Données :** S : une pile d'arêtes.**début**

```

1  si  $V(G) = \{x_0\}$  alors
2  |    $w \leftarrow x_0$ 
3  |   retourner  $w$ 
4  |
5  |    $w \leftarrow \emptyset$ 
6  |    $x \leftarrow x_0$ ;  $x_1 \leftarrow x_0$ 
7  |   soit  $xy$  une arête dans  $E(G)$  incidente à  $x$ 
8  |    $S \leftarrow$  empiler  $\perp$  sur  $\emptyset$ 
9  |   tant que  $S \neq \emptyset$  faire
10 |   |    $E(G) \leftarrow E(G) - xy$ 
11 |   |   si il existe une arête  $yz$  dans  $E(G)$  incidente à  $y$  alors
12 |   |   |    $S \leftarrow$  empiler  $xy$  sur  $S$ 
13 |   |   |    $x \leftarrow y$ ;  $y \leftarrow z$ 
14 |   |   sinon
15 |   |   |   si  $y \neq x_1$  alors
16 |   |   |   |   /*Graphe non eulérien!*/
17 |   |   |   |   retourner (False,  $y$ )
18 |   |   |   sinon
19 |   |   |   |    $w \leftarrow w \cdot x_1$ 
20 |   |   |   |    $x_1 \leftarrow x$ 
21 |   |   |   |    $xy \leftarrow$  dépiler  $S$ 
22 |   |   /*Graphe eulérien!*/
23 |   |    $w \leftarrow w \cdot x_0$ 
24 |   |   retourner (True,  $w$ )

```

Proposition

L'algorithme décide si un graphe fini, sans boucle et connexe G est eulérien en temps $O(|E(G)|)$. Le cas échéant il construit un sentier fermé (ou encore un cycle eulérien) de G , sinon il retourne un sommet de degré impair.

Montrons d'abord que l'algorithme s'arrête en au plus $2 \times |E(G)|$ itérations. Lorsqu'on exécute la boucle il y a trois possibilités disjointes :

- ▶ on sort en erreur en 5
- ▶ on empile une arête en 3
- ▶ on dépile une arête en 8

Le nombre d'itérations est donc au plus égale à la somme du nombre d'empilements et du nombre de dépilements. On note que toute arête qui est empilée a été précédemment enlevée du graphe (instruction 2) ce qui implique qu'on empilera au plus $|E(G)|$ arêtes. Le test d'arrêt de la boucle, pile vide, implique qu'on exécutera l'instruction (instruction 1) de dépilement un nombre de fois au plus égal au nombre d'empilements d'où le résultat attendu. Montrons maintenant que le résultat de l'algorithme est correct. Pour cela montrons tout d'abord les invariants suivants.

Invariant 1

En toute instruction de l'algorithme $x \neq y$ (les contenus des variables x et y sont différents).

Le nombre d'itérations est donc au plus égale à la somme du nombre d'empilements et du nombre de dépilements. On note que toute arête qui est empilée a été précédemment enlevée du graphe (instruction 2) ce qui implique qu'on empilera au plus $|E(G)|$ arêtes. Le test d'arrêt de la boucle, pile vide, implique qu'on exécutera l'instruction (instruction 1) de dépilement un nombre de fois au plus égal au nombre d'empilements d'où le résultat attendu. Montrons maintenant que le résultat de l'algorithme est correct. Pour cela montrons tout d'abord les invariants suivants.

Invariant 1

En toute instruction de l'algorithme $x \neq y$ (les contenus des variables x et y sont différents).

proof

Cet invariant est trivialement vérifié du fait que le graphe est sans boucle.

Invariant 2

Pour tout sommet s notons $\delta_0(s)$ le degré initial du sommet et $\delta(s)$ son degré courant. Alors après l'instruction 2 les deux conditions suivantes sont vérifiées :

- ▶ ou bien $y = x_1$ ou bien $\delta(y)$ et $\delta_0(y)$ ont des parités différentes
- ▶ ou bien $x = x_1$ ou bien $\delta(x)$ et $\delta_0(x)$ ont mêmes parités

proof

À la première itération l'invariant 2 est vérifié car nous avons de façon certaine $\delta(y) = \delta_0(y) - 1$ et $y \neq x_1$ et $x = x_1$. Maintenant supposons vrai l'invariant à l'itération i et situons nous à l'itération $(i + 1)$ juste avant d'exécuter l'instruction **??**. Si l'arête xy avait déjà été enlevée alors exécuter l'instruction de modifiera pas l'invariant. Si l'arête xy existe encore dans le graphe il est utile de considérer tous les cas possibles.

Le cas $y = x = x_1$ ne peut se produire par l'invariant 1.

Si $x = x_1$ alors on doit vérifier que $\delta(y)$ et $\delta_0(y)$ ont même parités.

Si y n'a jamais été vu c'est trivialement vrai, sinon en raison de l'échange effectué à l'instruction 4, pour chaque arête enlevée d'extrémité y il y eu une arête d'origine y enlevée d'où l'égalité des degrés.

Enfin si $y = x_1$ on doit vérifier que $\delta(x)$ et $\delta_0(x)$ auront même parité après exécution. Si la dernière instruction exécutée est l'instruction 4 alors $\delta(x)$ possède l'ancienne valeur de $\delta(y)$ qui, par hypothèse d'induction, avait une parité différente de celle de $\delta_0(y)$ puisque y était différent de x_1 . Ainsi l'exécution de l'instruction rétablira l'invariant. Si la dernière instruction exécutée est l'instruction 8 alors l'arête xy qui était dans la pile avait déjà été enlevée et exécuter l'instruction ne modifiera pas l'invariant.

La conséquence de l'invariant 2 est que si l'algorithme se termine par l'instruction 5 alors $\delta(y)$ et $\delta_0(y)$ auront des parités différentes et comme $\delta(y) = 0$, on en déduit que $\delta_0(y)$ est impair et donc le graphe G n'est pas eulérien.

Invariant 3

À chaque instruction si la liste de sommets w n'est pas vide elle contient un sentier d'extrémité x_0 .

proof

Si l'instruction 6 n'est jamais exécutée w reste vide mais cela signifie qu'on sort de l'algorithme par l'instruction 5 car le graphe G n'est pas eulérien comme vu précédemment.

Sinon montrons l'invariant par induction sur le nombre de fois qu'est exécutée l'instruction 6. A la première exécution on écrit le sommet initial x_0 ainsi la liste w contient un sentier dégénéré d'extrémité x_0 et l'invariant est donc bien vérifié. Supposons l'invariant vérifié après la i ème exécution de l'instruction 6 et notons le sentier w par $y_1 \cdot y_2 \dots y_i$ où $y_1 = x_0$. Si on exécute une $(i + 1)$ ème fois l'instruction c'est que $y = x_1$ et donc on ajoute x_1 à la liste w or le dernier élément de la liste y_i , a été ajouté juste avant l'affectation de x_1 en 7 donc $x_1 y_i$ est une arête de G précédemment dépilée et donc $y_1 \cdot y_2 \dots y_i \cdot x_1$ forme un sentier d'extrémité x_0 .

Pour finir la preuve on note que si l'algorithme ne se termine pas en 5 alors on passe par l'instruction 9 qui rajoute le sommet initial x_0 à w qui par l'invariant 3 est un sentier d'extrémité x_0 ainsi on obtient un cycle eulérien.