

COLLÈGE  
DE FRANCE  
— 1530 —

# Algorithmes pour graphes dynamiques

Claire Mathieu



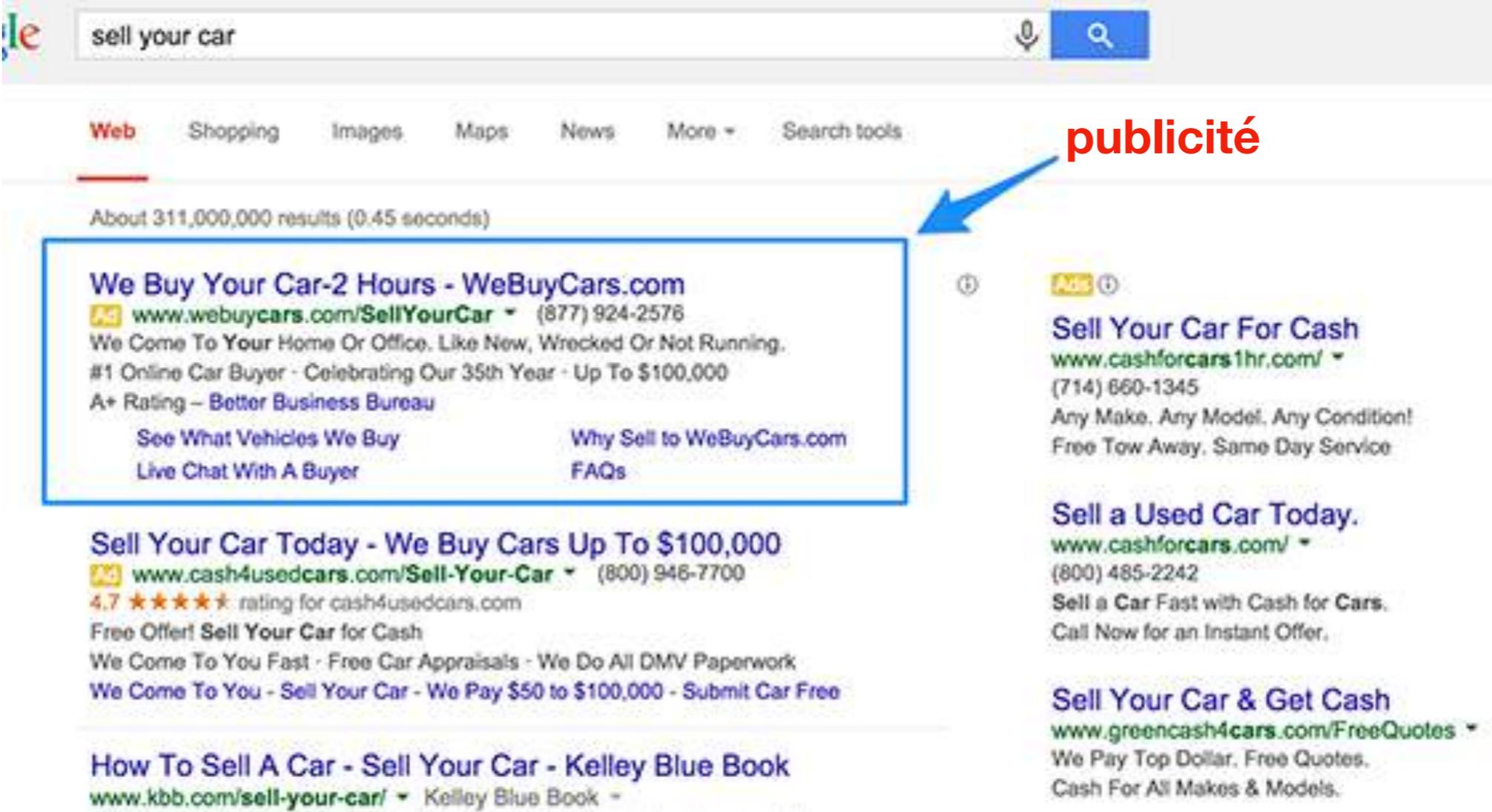
# Des graphes qui évoluent au fil du temps



**Construire un couplage biparti en-ligne**  
**Clôture transitive de graphe sans cycle**  
**Connexité de graphe**

**Construire un couplage biparti en-ligne**

# Comment les moteurs de recherche sont financés



The image shows a Google search results page for the query "sell your car". The search bar at the top contains the text "sell your car" and a search button. Below the search bar, there are navigation links for "Web", "Shopping", "Images", "Maps", "News", "More", and "Search tools". The search results show "About 311,000,000 results (0.45 seconds)".

The first result is an advertisement for "We Buy Your Car-2 Hours - WeBuyCars.com". This ad is highlighted with a blue box and a blue arrow pointing to it from the word "publicité" (advertisement) written in red. The ad text includes: "www.webuycars.com/SellYourCar", "(877) 924-2578", "We Come To Your Home Or Office. Like New, Wrecked Or Not Running.", "#1 Online Car Buyer - Celebrating Our 35th Year - Up To \$100,000", "A+ Rating - Better Business Bureau", "See What Vehicles We Buy", "Why Sell to WeBuyCars.com", "Live Chat With A Buyer", and "FAQs".

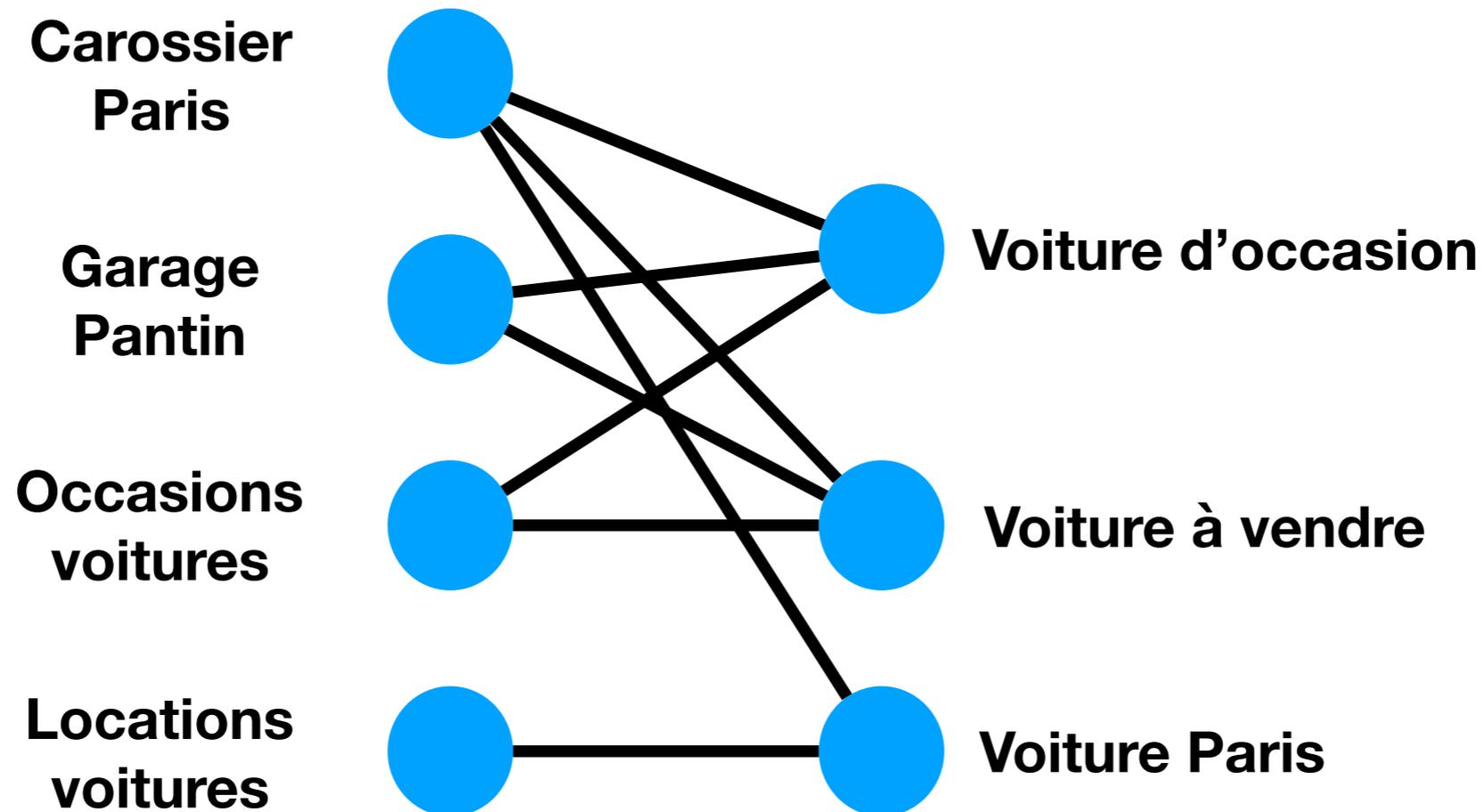
Other advertisements visible on the page include:

- "Sell Your Car For Cash" from www.cashforcars1hr.com/ with phone number (714) 660-1345. Text: "Any Make. Any Model. Any Condition! Free Tow Away. Same Day Service".
- "Sell a Used Car Today." from www.cashforcars.com/ with phone number (800) 485-2242. Text: "Sell a Car Fast with Cash for Cars. Call Now for an Instant Offer."
- "Sell Your Car & Get Cash" from www.greencash4cars.com/FreeQuotes. Text: "We Pay Top Dollar. Free Quotes. Cash For All Makes & Models."

Organic search results include "Sell Your Car Today - We Buy Cars Up To \$100,000" from www.cash4usedcars.com/Sell-Your-Car with phone number (800) 946-7700 and a 4.7 star rating, and "How To Sell A Car - Sell Your Car - Kelley Blue Book" from www.kbb.com/sell-your-car/.

À quel publicitaire allouer cet emplacement ?

# Connaissances a priori du moteur de recherche



# Requêtes au fil du temps

- Voiture d'occasion
- Voiture à vendre
- Voiture d'occasion
- Voiture Paris
- Voiture Paris
- Voiture d'occasion
- ...

Lorsqu'une requête arrive  
à quel publicitaire allouer l'emplacement ?

The image shows a Google search results page for the query "sell your car". The search bar at the top contains the text "sell your car" and a search icon. Below the search bar, there are navigation links for "Web", "Shopping", "Images", "Maps", "News", "More", and "Search tools". The search results are displayed below, with a blue box highlighting the first organic result: "We Buy Your Car-2 Hours - WeBuyCars.com". A blue arrow points from the right side of the page towards this highlighted result. To the right of the organic results, there are several sponsored ads, including "Sell Your Car For Cash" and "Sell a Used Car Today".

sell your car

Web Shopping Images Maps News More Search tools

About 311,000,000 results (0.45 seconds)

**We Buy Your Car-2 Hours - WeBuyCars.com**  
www.webuycars.com/SellYourCar (877) 924-2576  
We Come To Your Home Or Office. Like New, Wrecked Or Not Running.  
#1 Online Car Buyer - Celebrating Our 35th Year - Up To \$100,000  
A+ Rating - Better Business Bureau  
See What Vehicles We Buy Why Sell to WeBuyCars.com  
Live Chat With A Buyer FAQs

**Sell Your Car Today - We Buy Cars Up To \$100,000**  
www.cash4usedcars.com/Sell-Your-Car (800) 946-7700  
4.7 ★★★★★ rating for cash4usedcars.com  
Free Offer! Sell Your Car for Cash  
We Come To You Fast - Free Car Appraisals - We Do All DMV Paperwork  
We Come To You - Sell Your Car - We Pay \$50 to \$100,000 - Submit Car Free

**How To Sell A Car - Sell Your Car - Kelley Blue Book**  
www.kbb.com/sell-your-car Kelley Blue Book

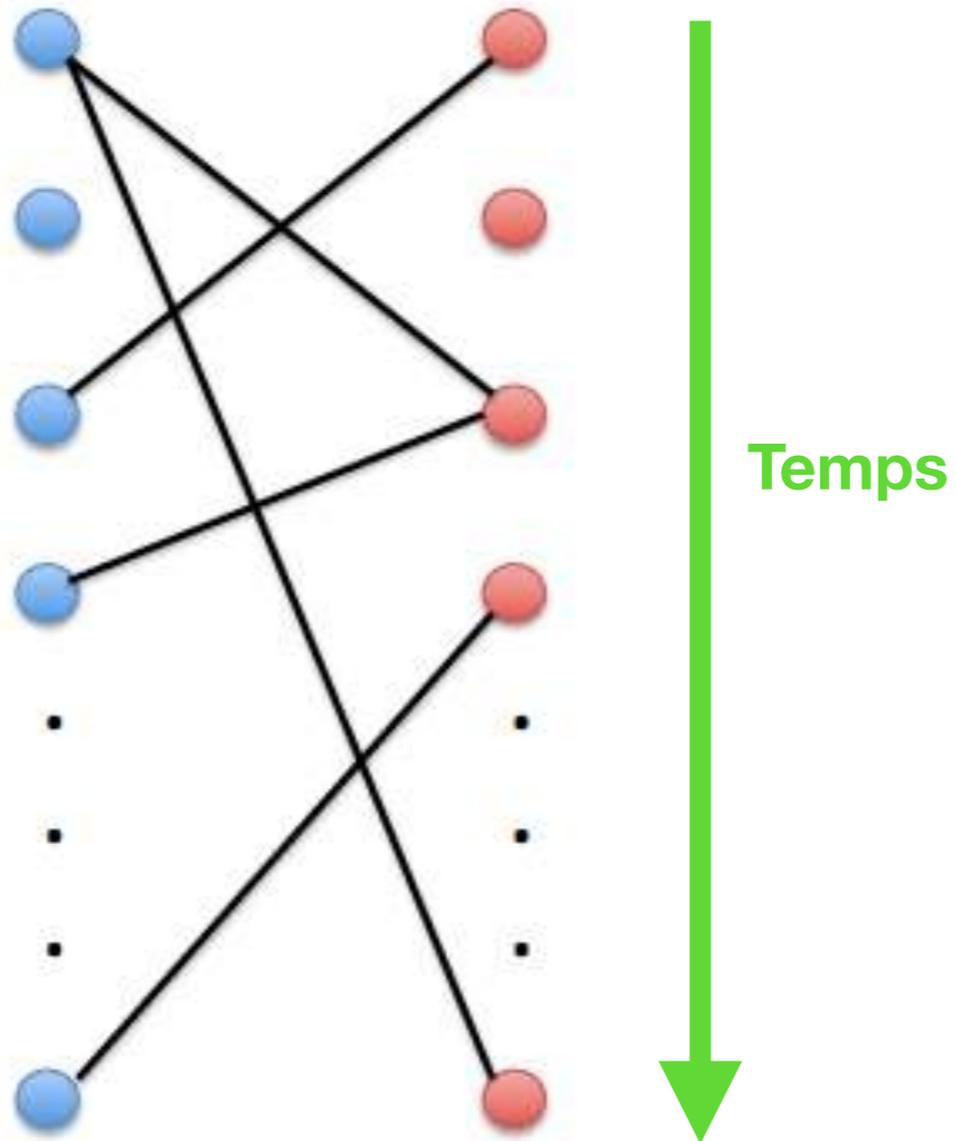
**ADS**  
**Sell Your Car For Cash**  
www.cashforcars1hr.com/ (714) 660-1345  
Any Make. Any Model. Any Condition!  
Free Tow Away. Same Day Service

**Sell a Used Car Today.**  
www.cashforcars.com/ (800) 485-2242  
Sell a Car Fast with Cash for Cars.  
Call Now for an Instant Offer.

**Sell Your Car & Get Cash**  
www.greencash4cars.com/FreeQuotes  
We Pay Top Dollar. Free Quotes.  
Cash For All Makes & Models.

**Publicitaires**

**Requêtes**



**On connaît les publicitaires : sommets de gauche**

**À chaque instant une requête arrive : sommet de droite**

**On voit toutes les arêtes : publicitaires intéressés par la requête**

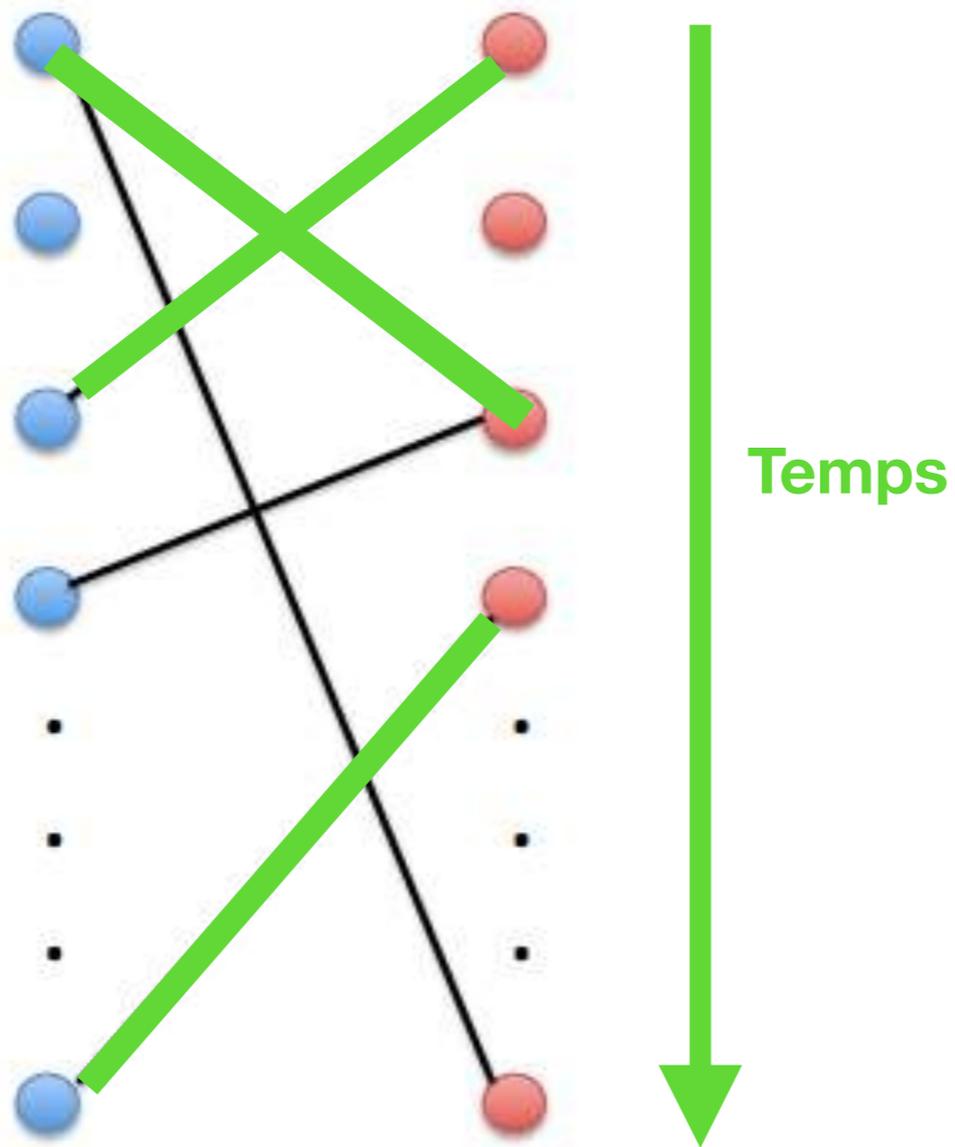
**On décide quel publicitaire faire apparaître**

**dans la réponse à la requête**

**Le graphe est révélé progressivement**

**Publicitaires**

**Requêtes**



**On connaît les publicitaires : sommets de gauche**

**À chaque instant une requête arrive : sommet de droite**

**On voit toutes les arêtes : publicitaires intéressés par la requête**

**On décide quel publicitaire faire apparaître**

**dans la réponse à la requête**

**Le graphe est révélé progressivement**

# Couplage de taille maximum

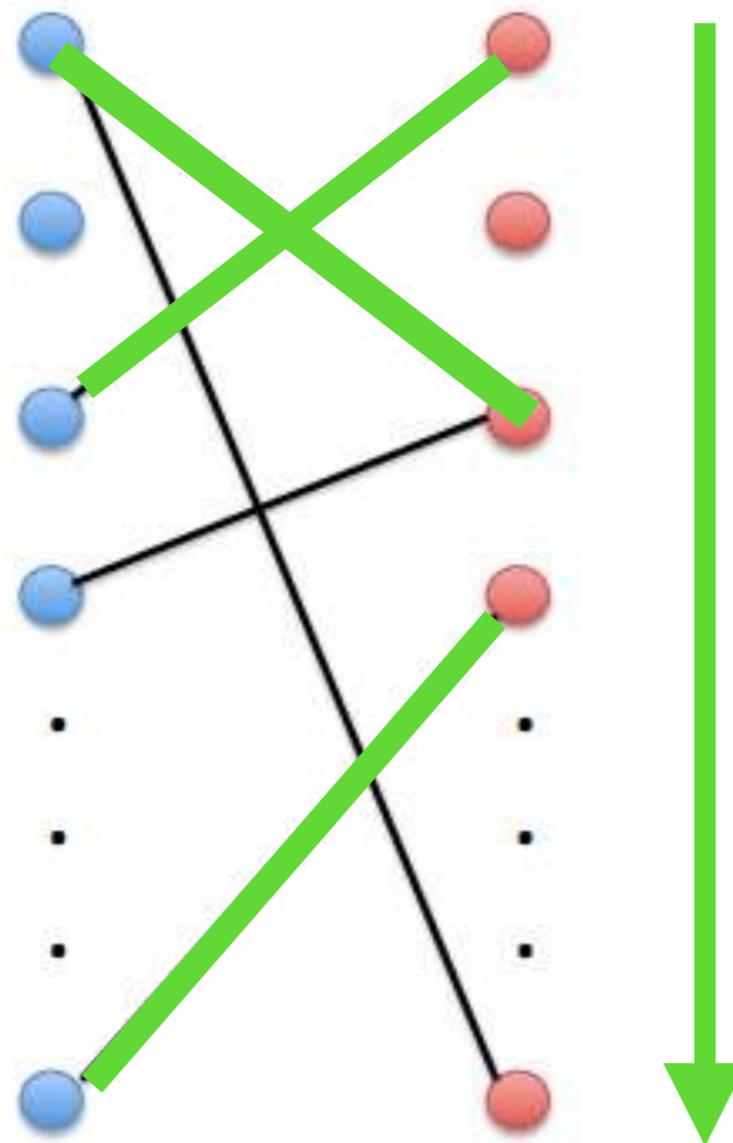
Le publicitaire a payé pour apparaître **une fois**, pas plus —  
Chaque sommet de gauche est choisi au plus une fois.

Il y a **un seul** bon emplacement —

Chaque sommet de droite est choisi au plus une fois  
=  
Couplage

Gauche

Droite



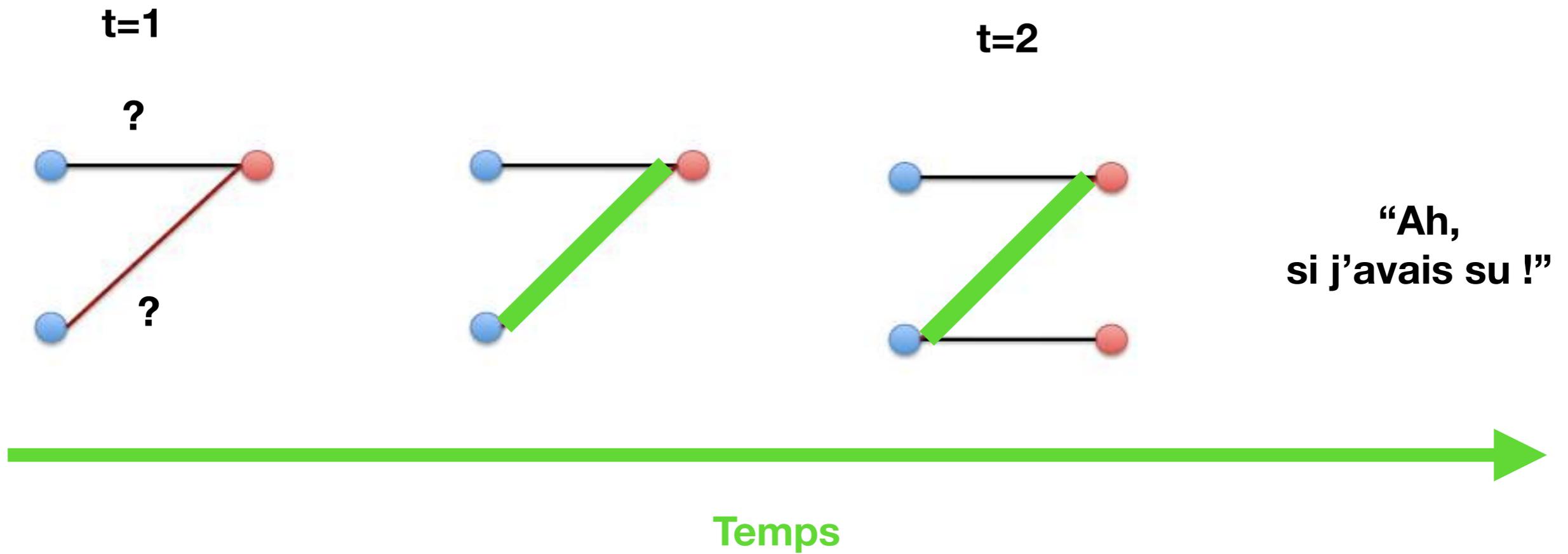
Maximiser le nombre de publicitaires satisfaits

Maximiser le nombre d'arêtes du couplage

On connaît les sommets de gauche  
À chaque instant un sommet de droite arrive  
On voit toutes les arêtes à ce sommet  
On décide quelle arête choisir

**Le graphe est révélé progressivement**

# Analyse : Impossible d'être parfait

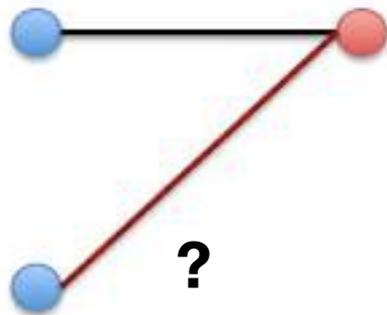


# Algorithme randomisé mal conçu

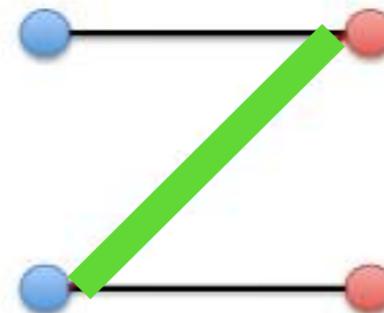
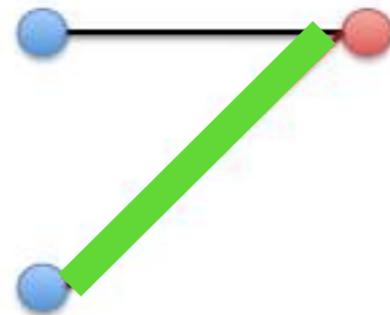
Arrivée du sommet  $u$  :  
Parmi les voisins  $v$  de  $u$   
qui sont encore libres  
en choisir un **au hasard**

$t=1$

?

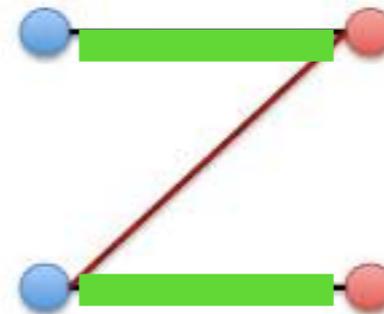
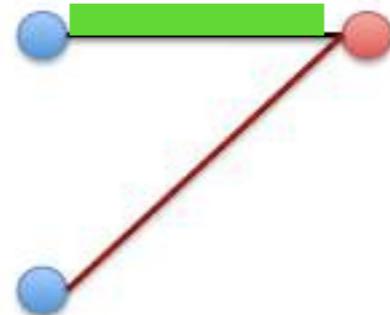
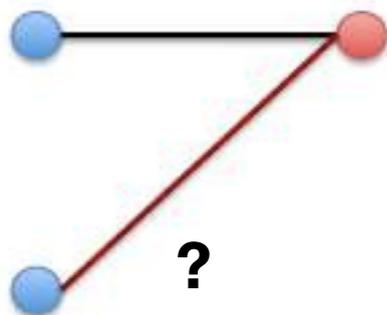


$t=2$



“Ah,  
si j’avais su !”

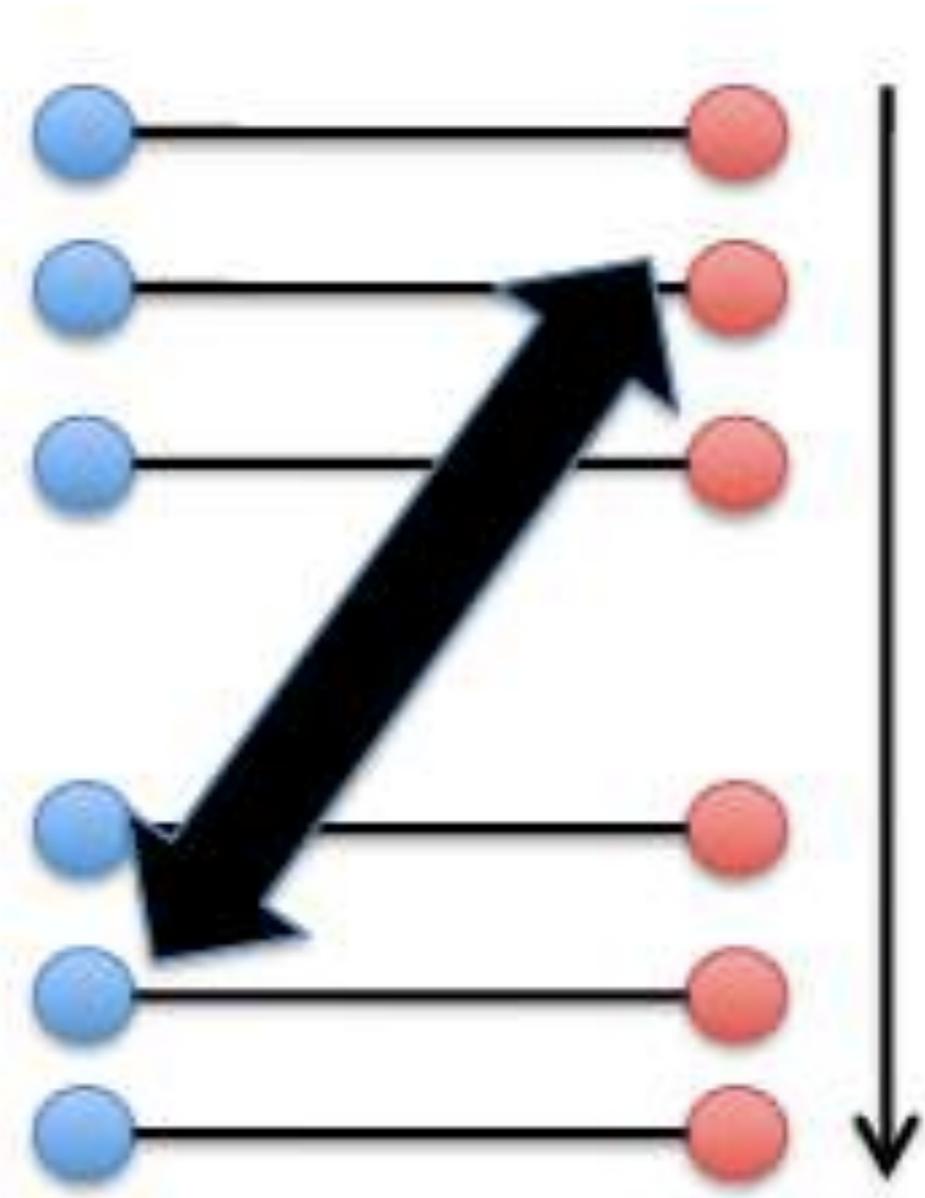
?



“Bien joué”

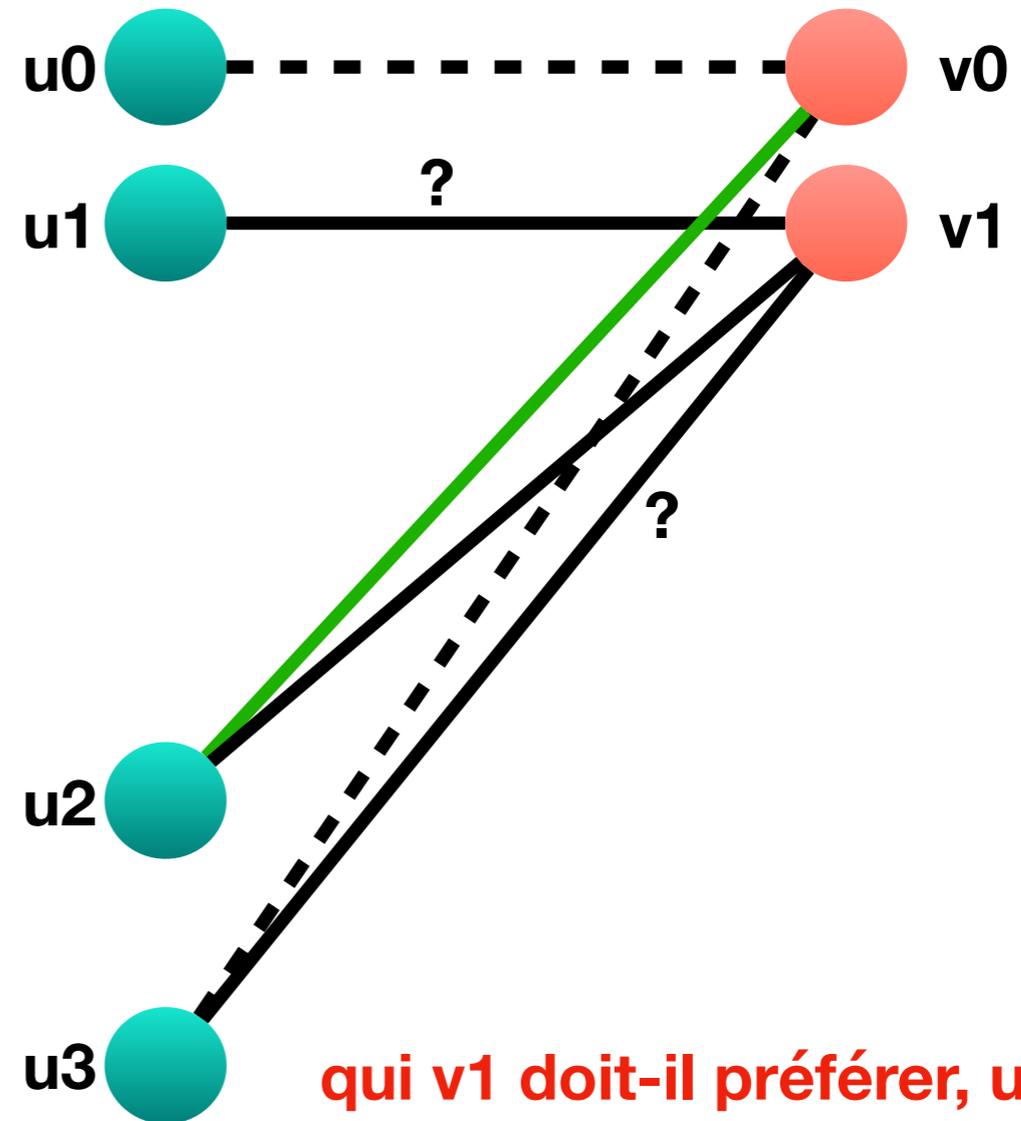
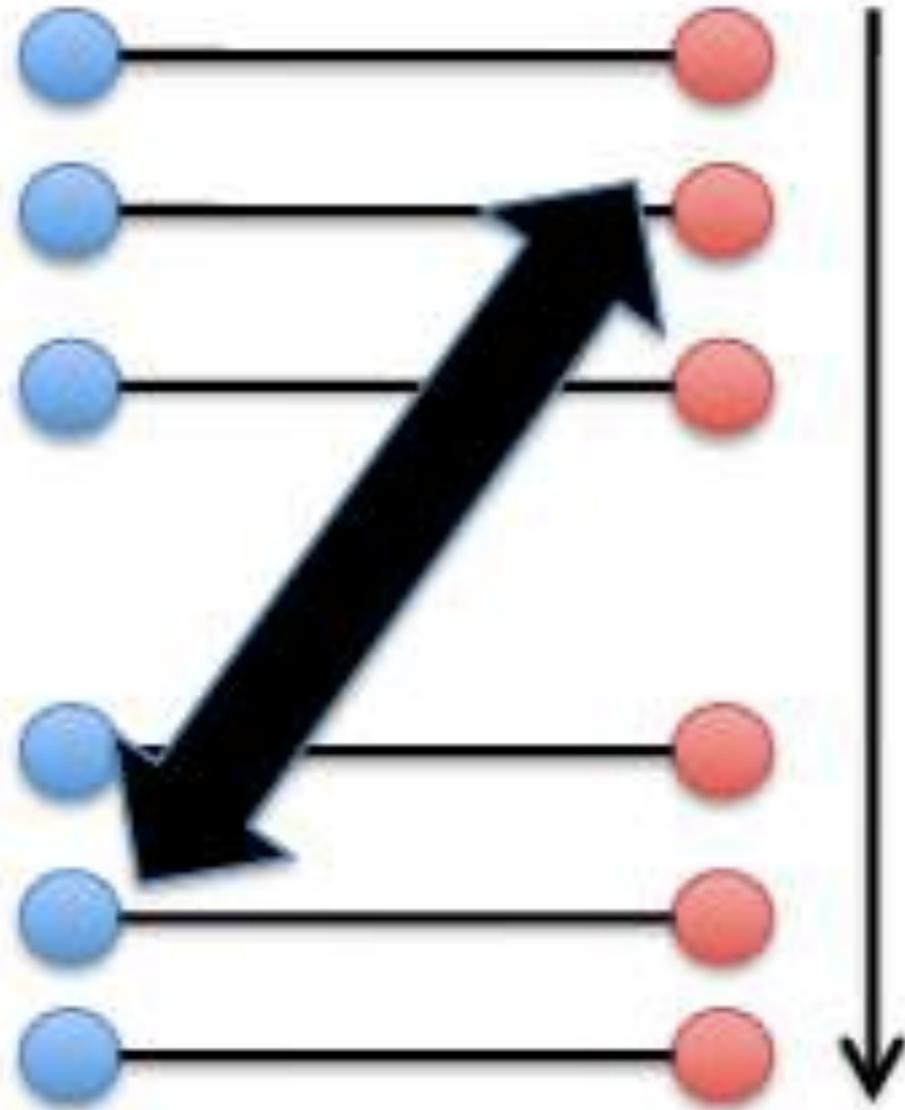
# Algorithme randomisé mal conçu

Arrivée du sommet  $u$  :  
Parmi les voisins  $v$  de  $u$   
qui sont encore libres  
en choisir un **au hasard**



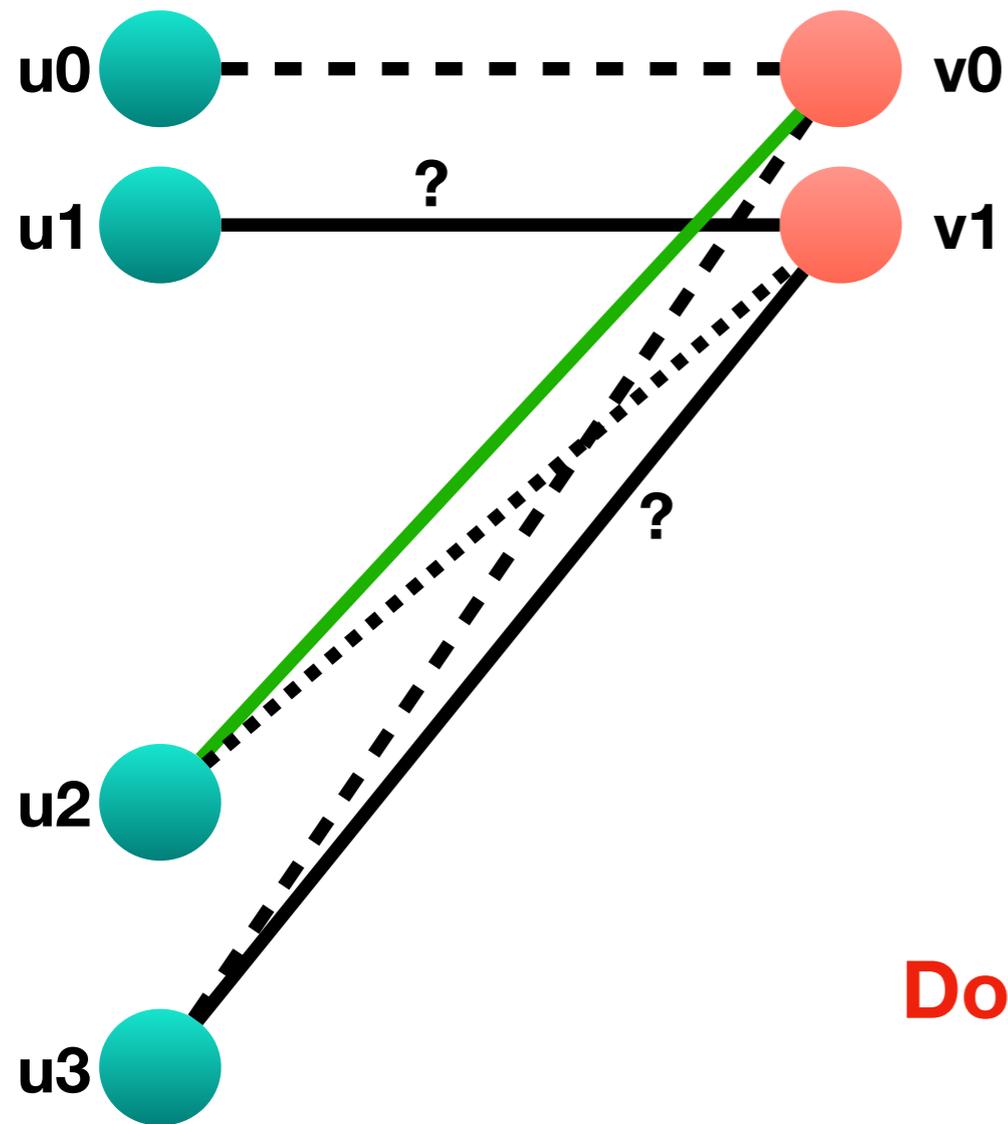
Seuls 50% des sommets sont appariés

# Comment corriger l'algorithme ?



**qui v1 doit-il préférer, u1 ou u3 ?**  
**Un raisonnement possible :**  
u3 a déjà eu une chance avec v0  
u1 n'a pas eu l'occasion  
u3 aura peut-être une autre chance  
donc : donner sa chance à u1

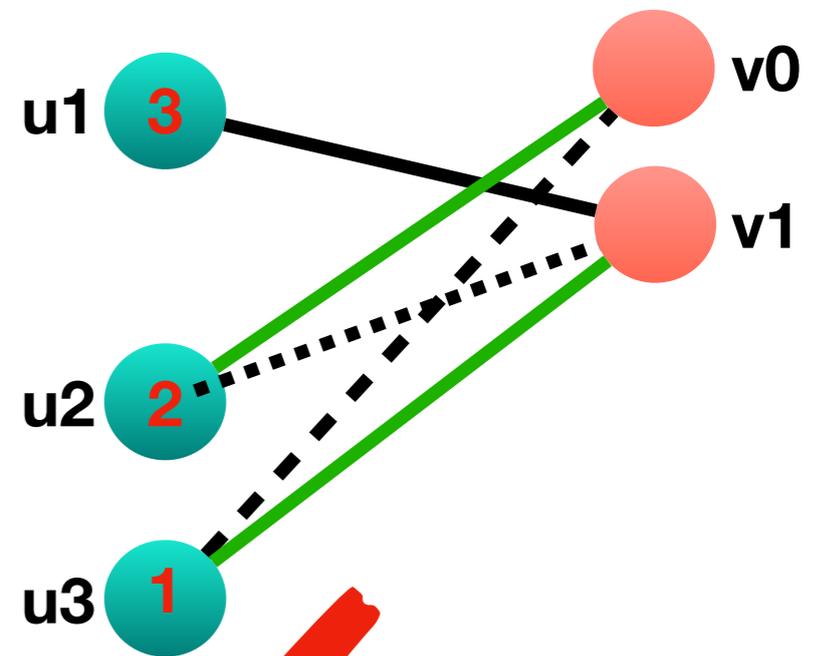
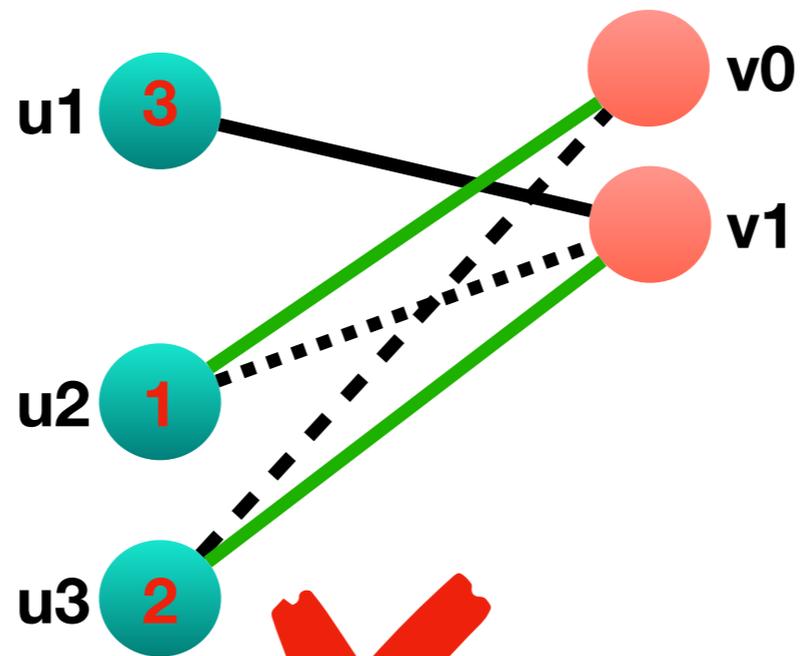
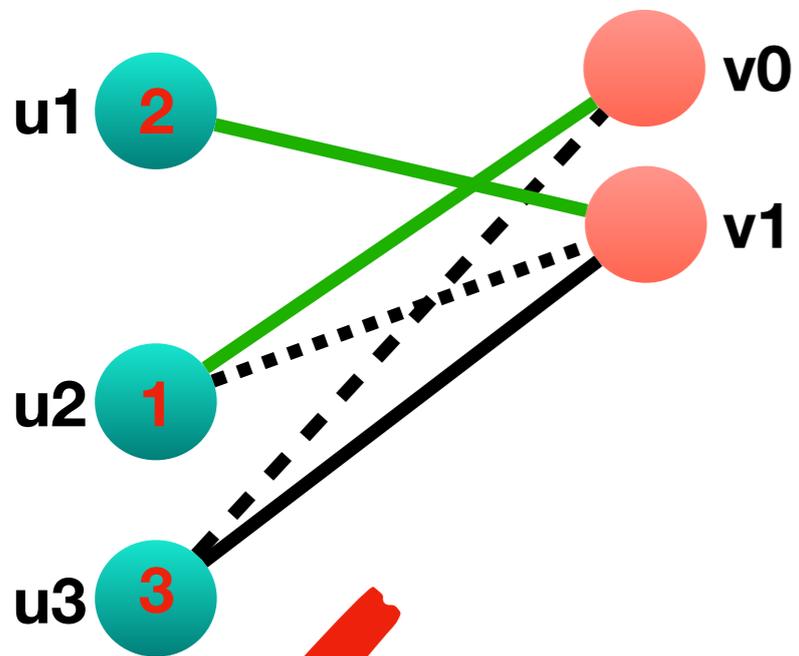
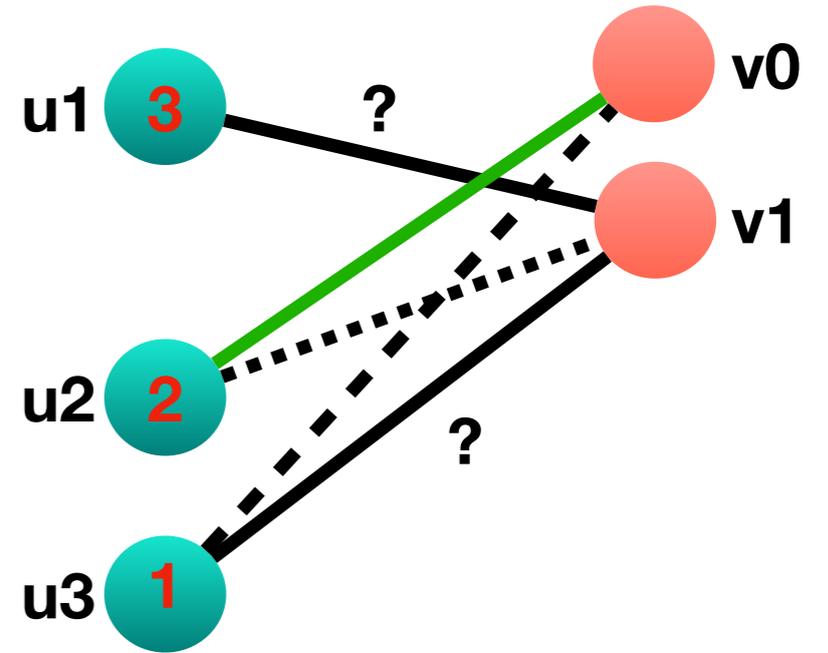
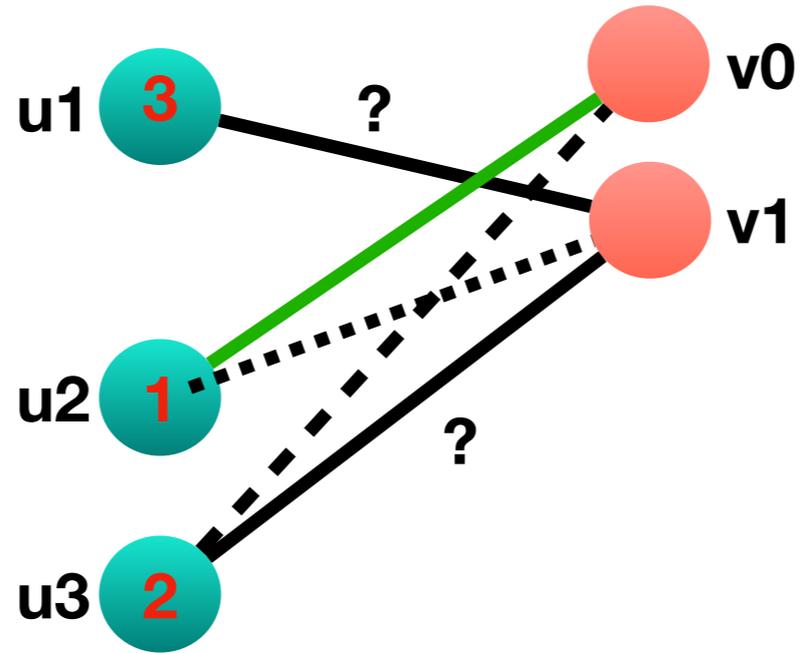
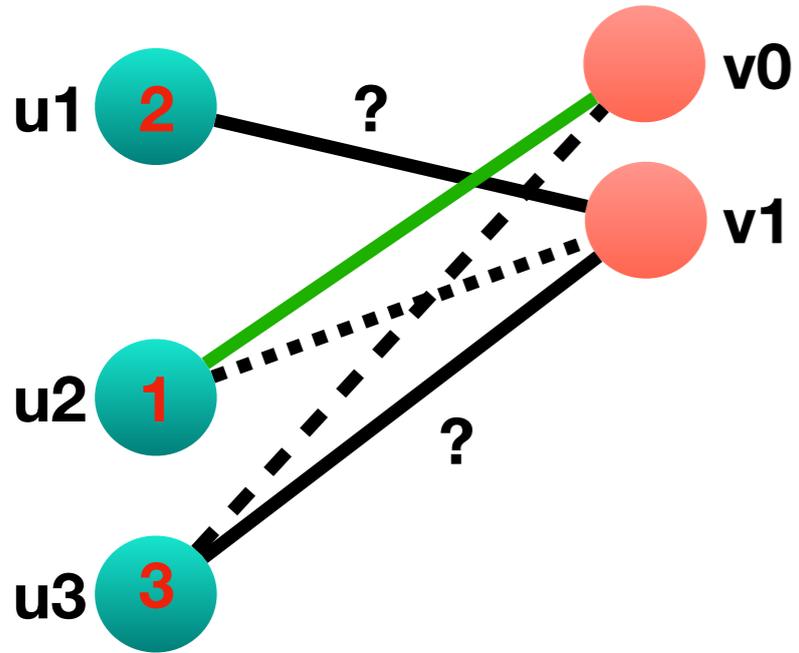
# Comment favoriser u1 par rapport à u3 par une règle simple ?



**un raisonnement possible :**  
u3 a déjà eu une chance avec v0  
u1 n'a pas eu l'occasion  
u3 aura peut-être une autre chance  
donc : donner sa chance à u1

**Donner à chaque  $u_i$  une priorité aléatoire**

# Donner à chaque $u_i$ une priorité aléatoire



$\text{Pr}(\text{bon choix})=2/3$

# Algorithme randomisé bien conçu

Choisir une **priorité aléatoire** aux sommets de gauche

Arrivée du sommet  $u$  :

Parmi les voisins  $v$  de  $u$

qui sont encore libres

choisir **le premier dans l'ordre de priorité**

**Théorème :**

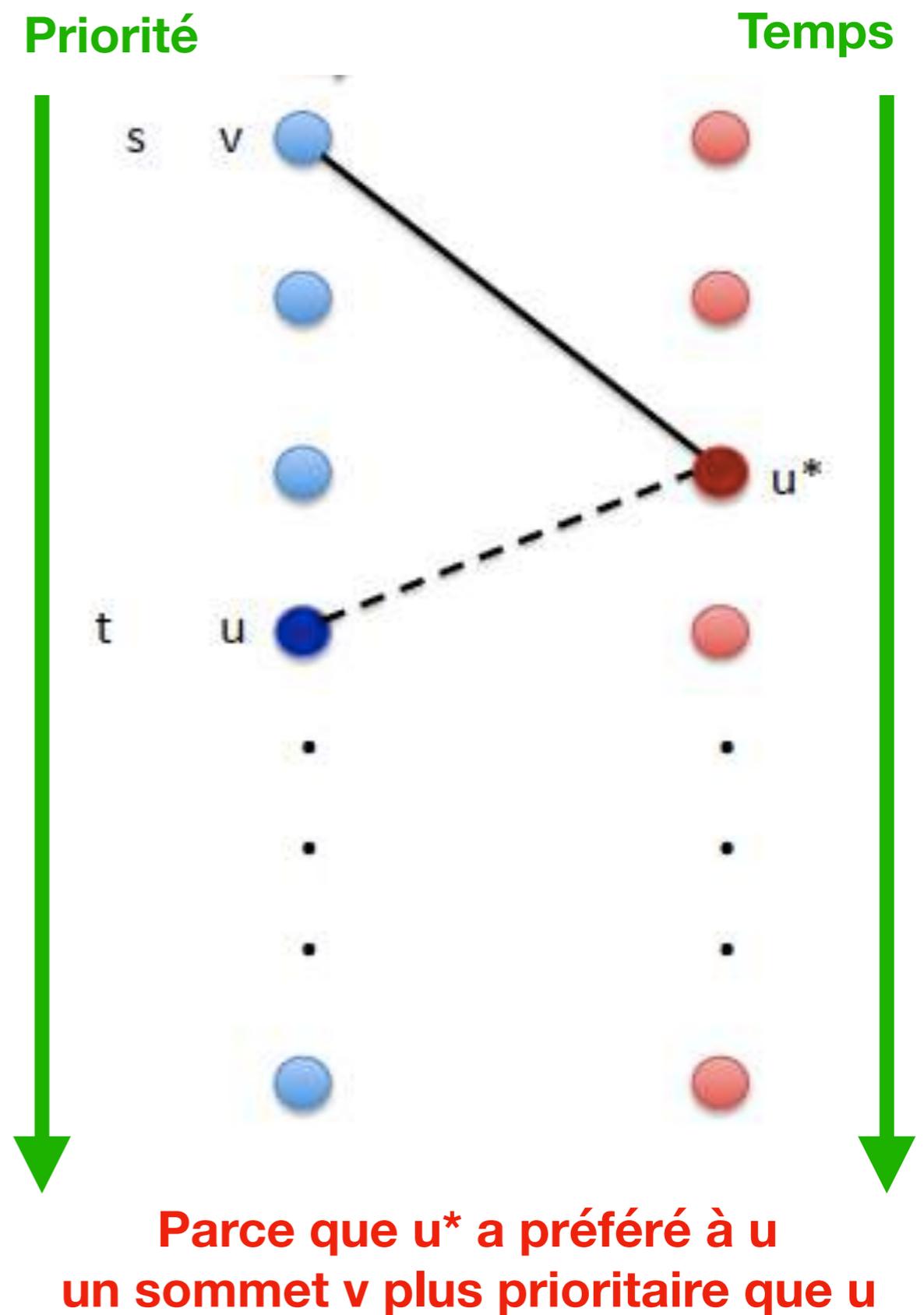
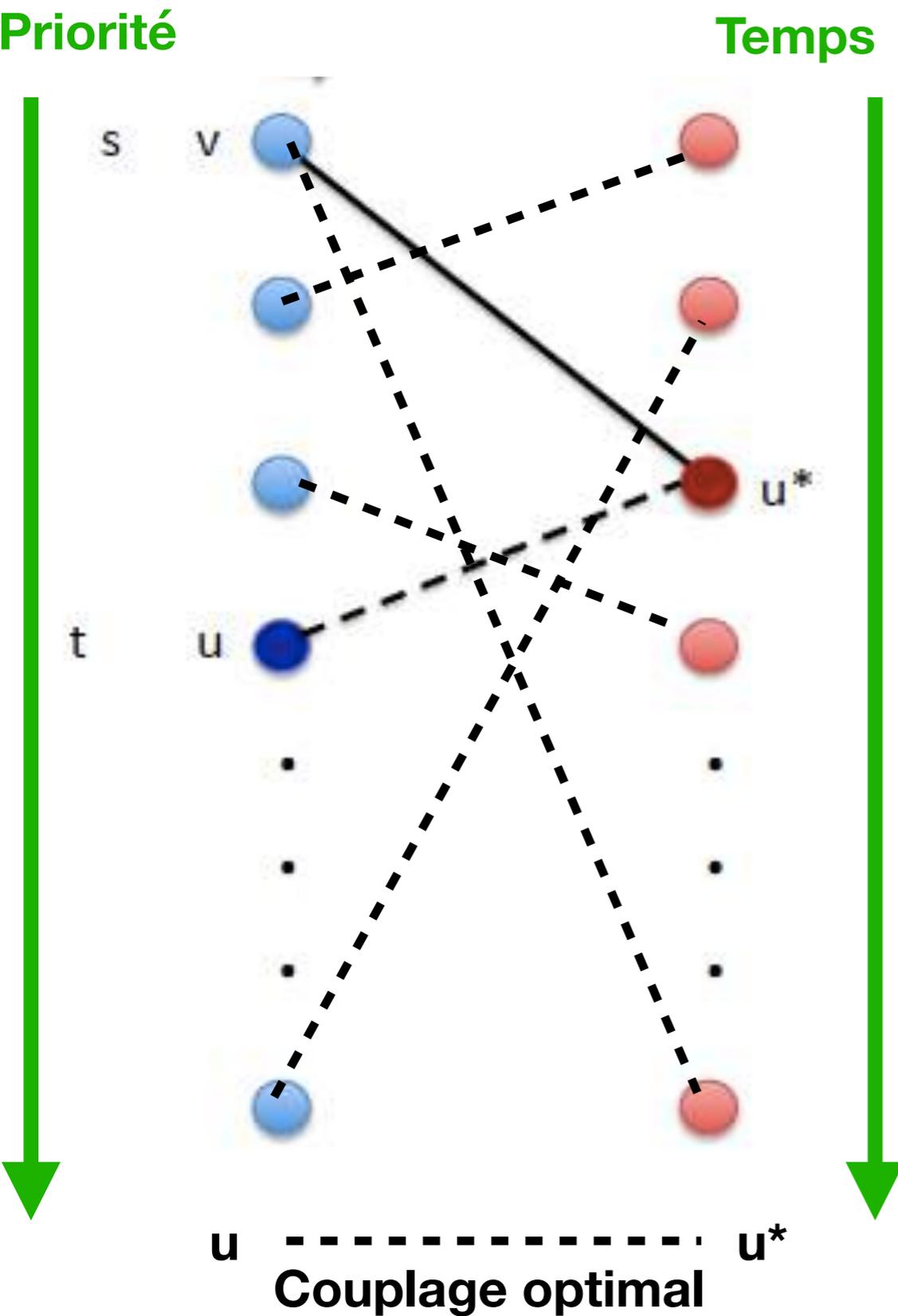
Quel que soit le graphe biparti

Quel que soit l'ordre d'arrivée des sommets de droite

En moyenne, 63% des sommets sont appariés.

**C'est optimal...**

# Analyse : pourquoi u se retrouve-t-il tout seul ? (1/2)



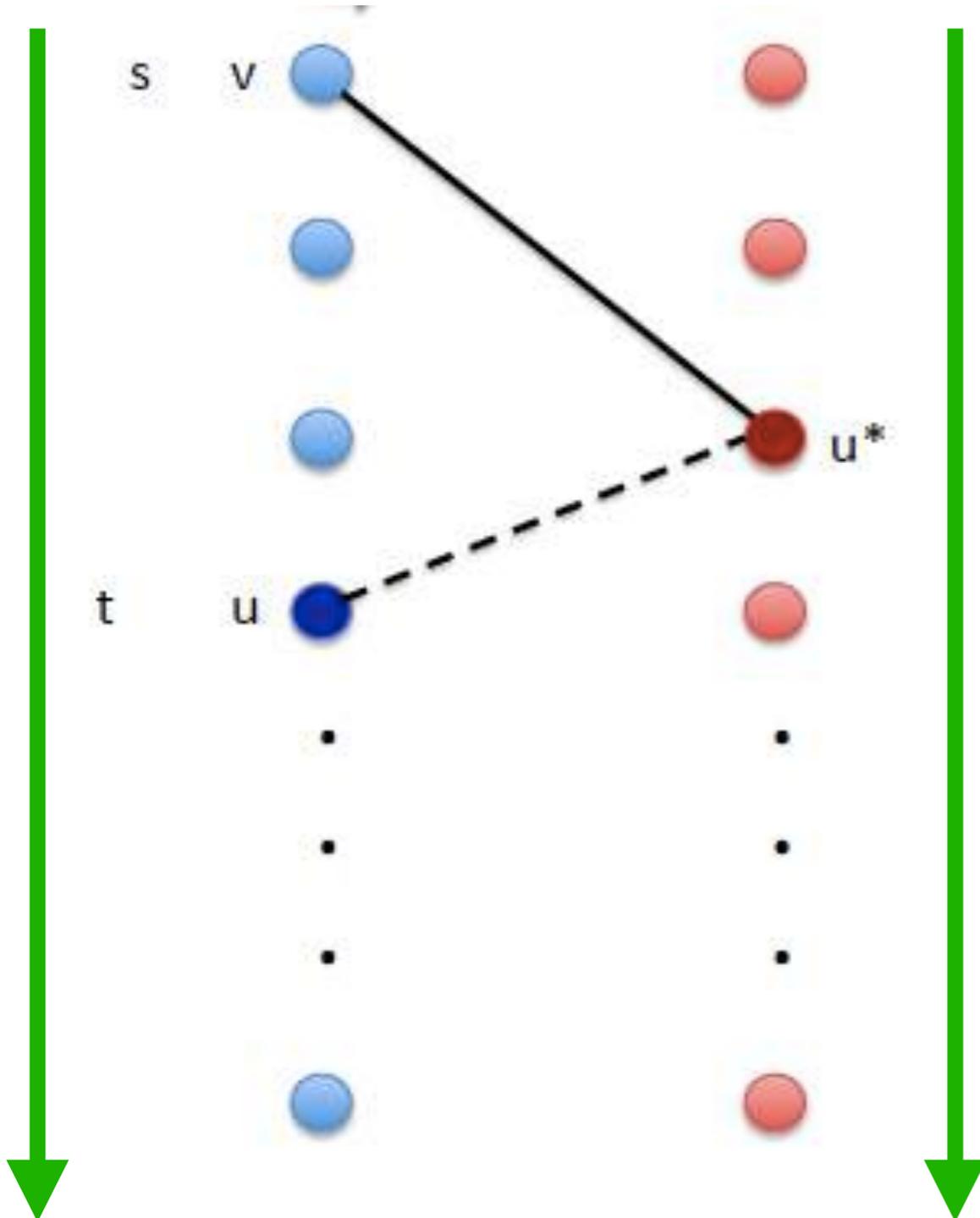
# Analyse (2/2)

Pourquoi u se retrouve-t-il tout seul ?

Un peu d'analyse d'algorithme...

Priorité

Temps



Parce que  $u^*$  a préféré à  $u$   
un sommet  $v$  plus prioritaire que  $u$

$$\Pr(\text{sommet de priorite } t \text{ tout seul}) \leq \frac{1}{n} \sum_{s < t} \Pr(\text{sommet de priorite } s \text{ pas tout seul})$$

$$1 - x_t \leq \frac{1}{n} \sum_{s < t} x_s$$

Un peu de calcul...

$$x_1 = 1$$

$$x_2 \geq 1 - \frac{1}{n} x_1$$

$$x_t \geq 1 - \frac{1}{n} (x_1 + x_2 + \dots + x_{t-1})$$

$$\text{Nb sommets apparies : } \sum_{s=1}^n x_s \geq n \left(1 - \frac{1}{e}\right) = .63n$$



# Le seul problème, c'est que la preuve est fausse !

Puzzle : trouver l'erreur !

Choisir une **priorité aléatoire** aux sommets de gauche  
Arrivée du sommet  $u$  :  
Parmi les voisins  $v$  de  $u$   
qui sont encore libres  
choisir **le premier dans l'ordre de priorité**

**Théorème :**  
Quel que soit le graphe biparti  
Quel que soit l'ordre d'arrivée des sommets de droite  
En moyenne, 63% des sommets sont appariés.

## Morale...

- **Les probabilités permettent de formaliser l'intuition**
- **La conception est simple (mais pas trop)**
- **L'analyse est délicate**

# Clôture transitive de graphe sans cycle

**Connaissant : les vols directs**  
**En déduire : les itinéraires avec correspondances**

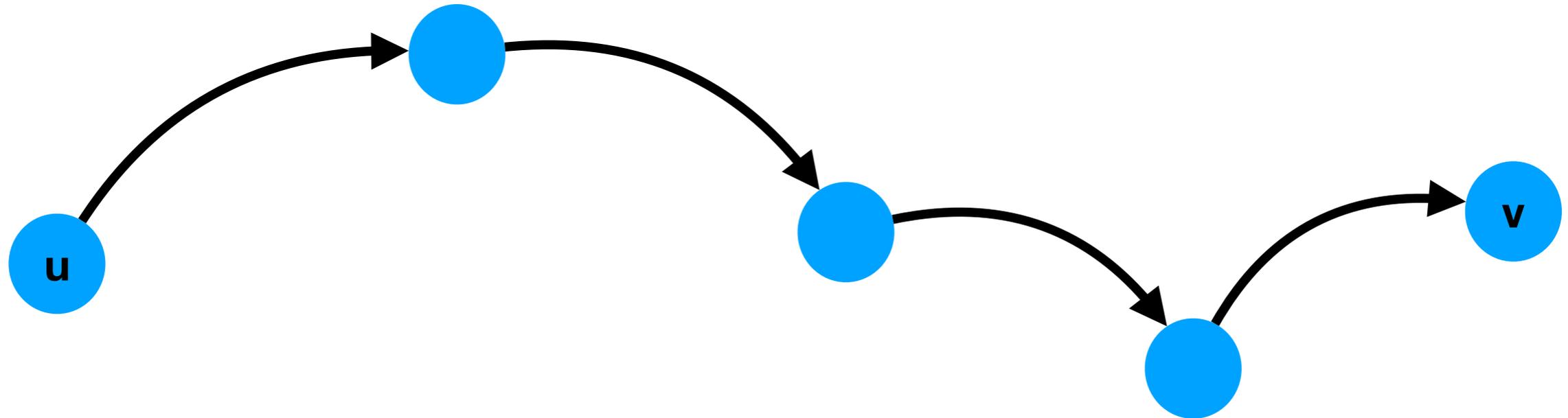
**S'il y a un vol Montréal-Paris la nuit  
un vol Paris-Nice le matin  
et un vol Nice-Venise le midi...**

**... alors on peut aller de Montréal à Venise en une journée**



# Graphe orienté

## Peut-on atteindre $v$ à partir de $u$ ?



Calcul par exploration de graphe

Que faire si le graphe évolue de manière dynamique ?

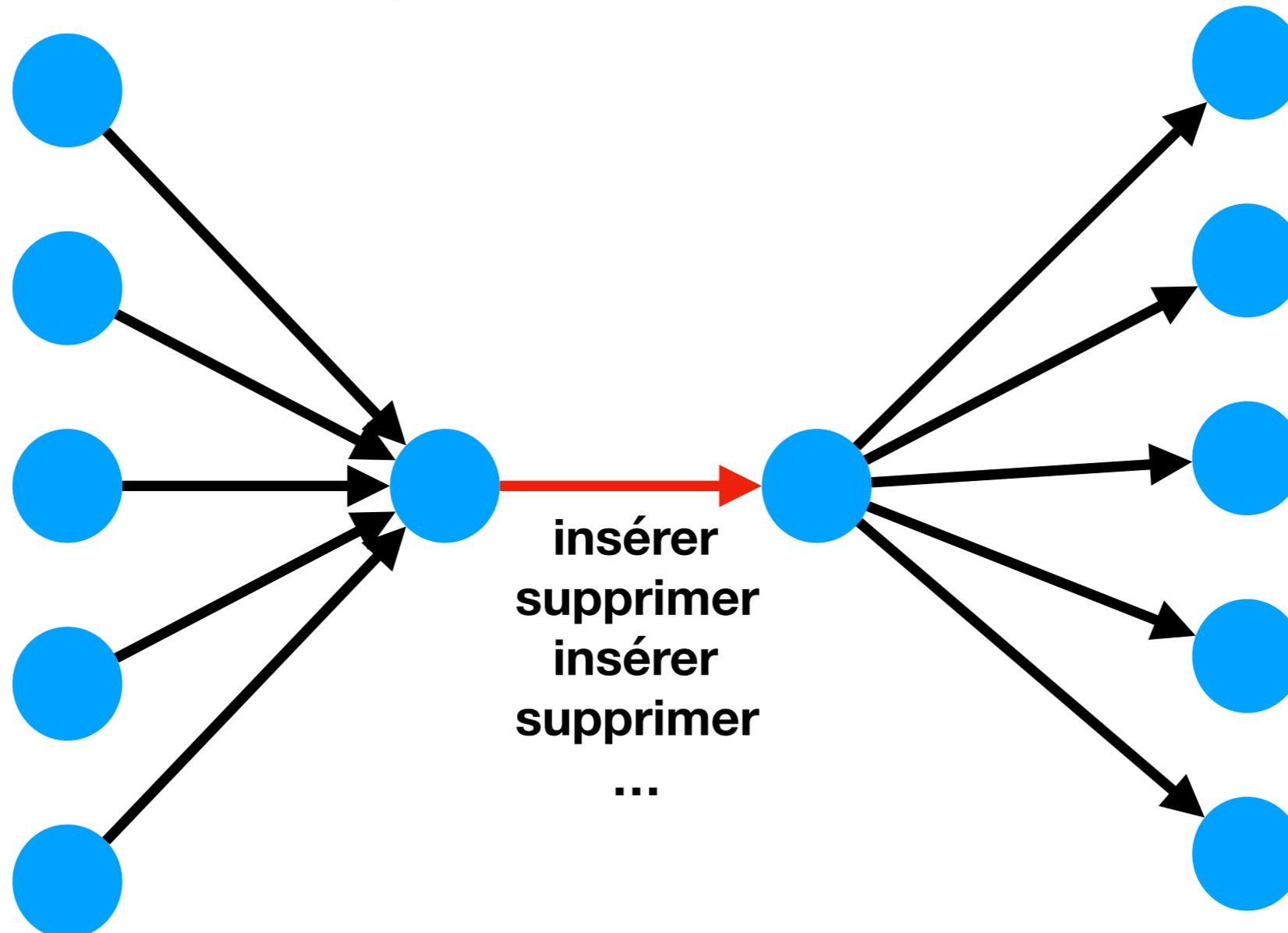
**Opérations de modification du graphe :**

insertions d'arcs  
suppressions d'arcs  
graphe à l'instant  $t$

**Demandes d'information sur le graphe :**

$v_i$  est-il atteignable à partir de  $v_j$  ?

$C[u,v]=$   
1 si v atteignable à partir de u  
0 sinon



Recalculer la clôture transitive après chaque modification :  
trop coûteux  
(multiplications de matrices)

## Idée : redondance

$C[u,v]=$

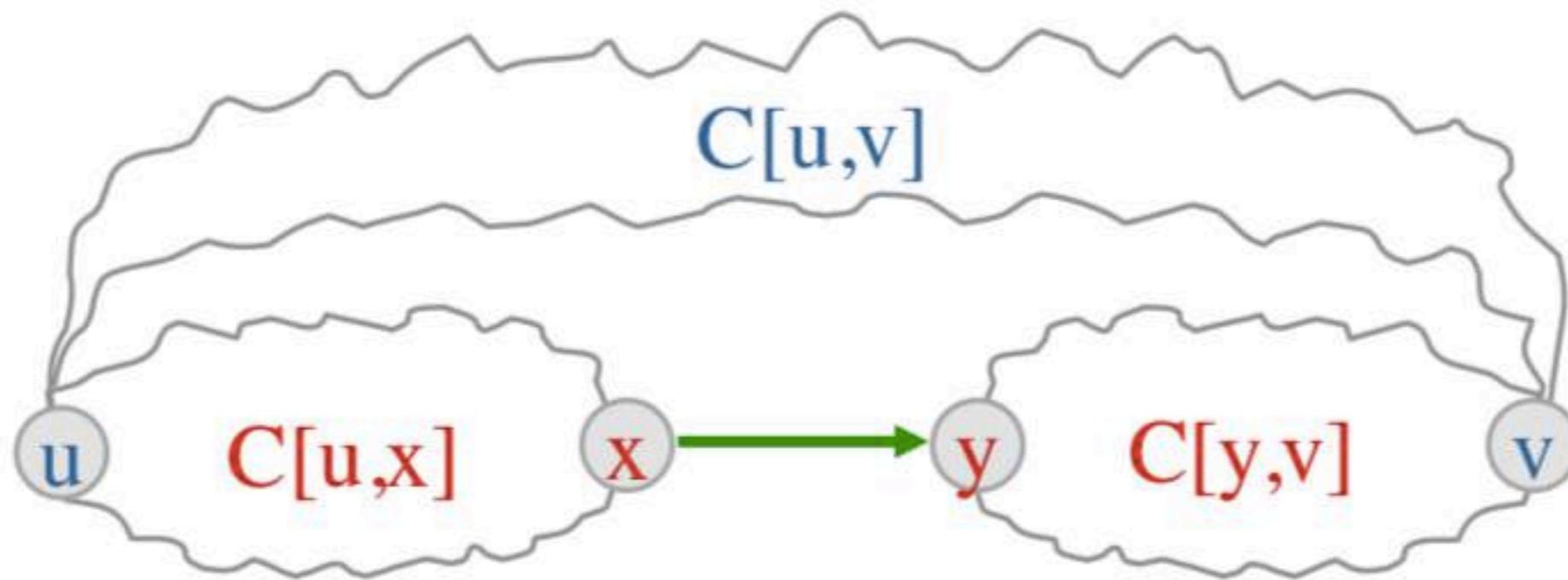
1 si  $v$  atteignable à partir de  $u$ , 0 sinon

Pour répondre à une requête :  $v$  atteignable à partir de  $u$   
si et seulement si  $C[u,v]=1$



$C[u,v]=$  **nombre** de chemins de  $u$  à  $v$

Pour répondre à une requête :  $v$  atteignable à partir de  $u$   
si et seulement si  $C[u,v]>0$

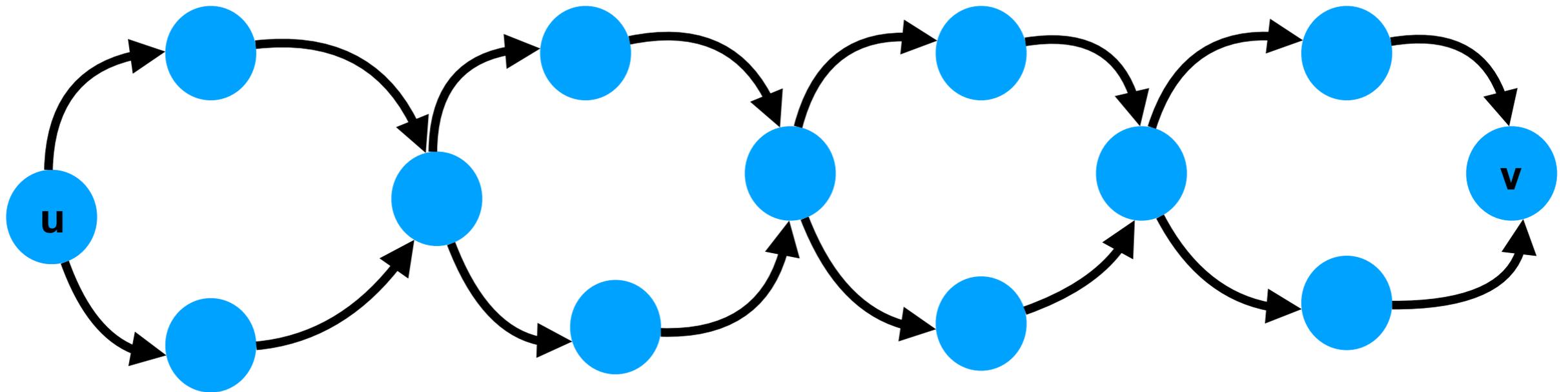


$$\forall u,v: \quad C[u,v] \leftarrow C[u,v] + C[u,x] \cdot C[y,v] \quad O(n^2)$$

$$\forall u,v: \quad C[u,v] \leftarrow C[u,v] - C[u,x] \cdot C[y,v] \quad O(n^2)$$

**Le seul problème, c'est que  
les nombres sont trop grands !**

$C[u,v]$  = **nombre** de chemins de  $u$  à  $v$

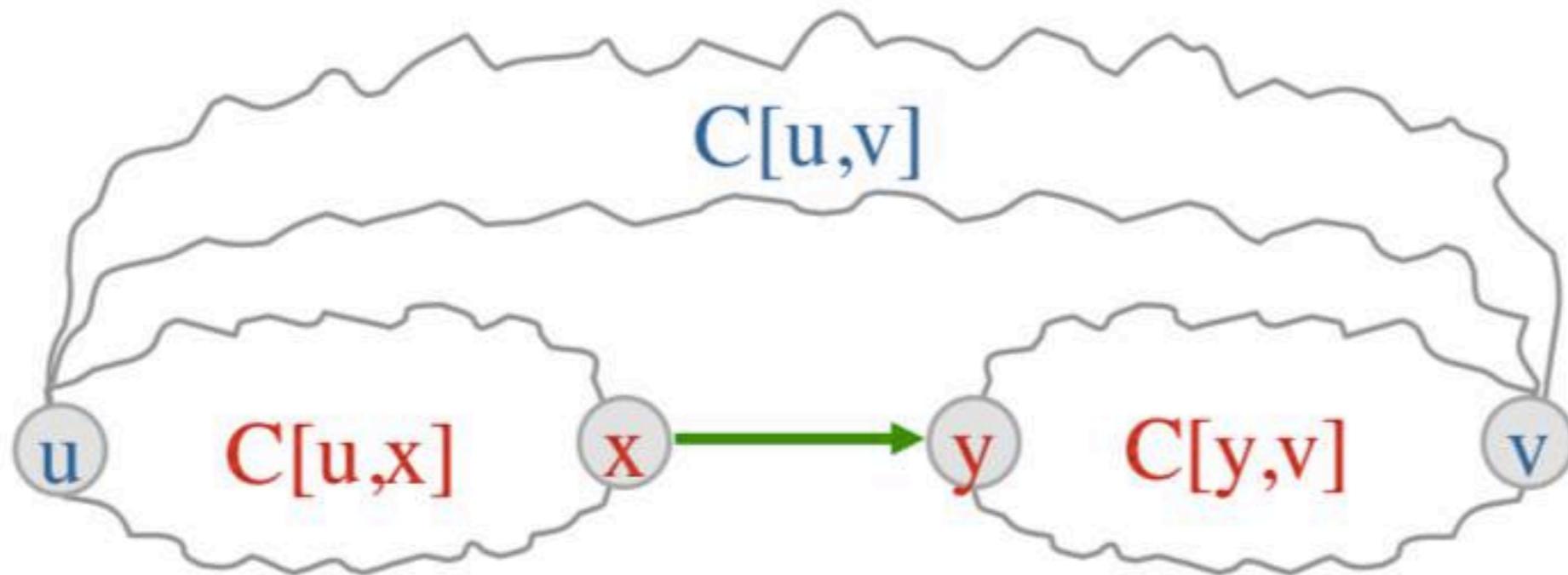


$$C[u, v] \sim 2^{n/3}$$

## Solution : calcul modulo p

p nombre premier aléatoire,  
de l'ordre de  $n^{10}$

si le nombre de modifications est inférieur à  $n^8$   
alors on ne tombera pas sur  $C[u,v]=0$  par hasard



$$\forall u,v: C[u,v] \leftarrow C[u,v] + C[u,x] \cdot C[y,v] \pmod p \quad O(n^2)$$

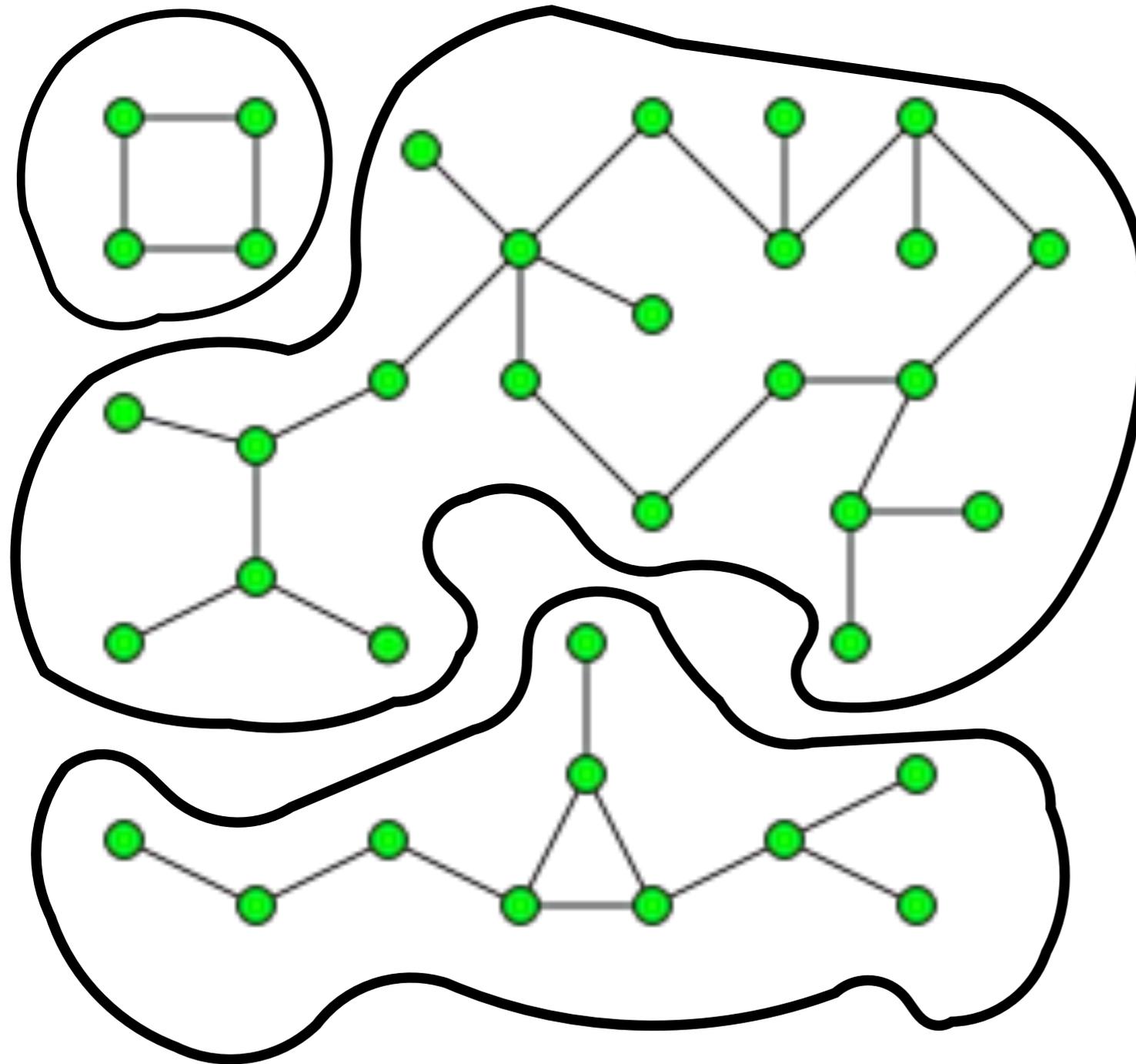
$$\forall u,v: C[u,v] \leftarrow C[u,v] - C[u,x] \cdot C[y,v] \pmod p \quad O(n^2)$$

## Morale...

- **Redondance de l'information**
- **Pièges des grands nombres**
- **Calculs modulo nombres premiers aléatoires**

# Connexité de graphe

# Connexité de graphe

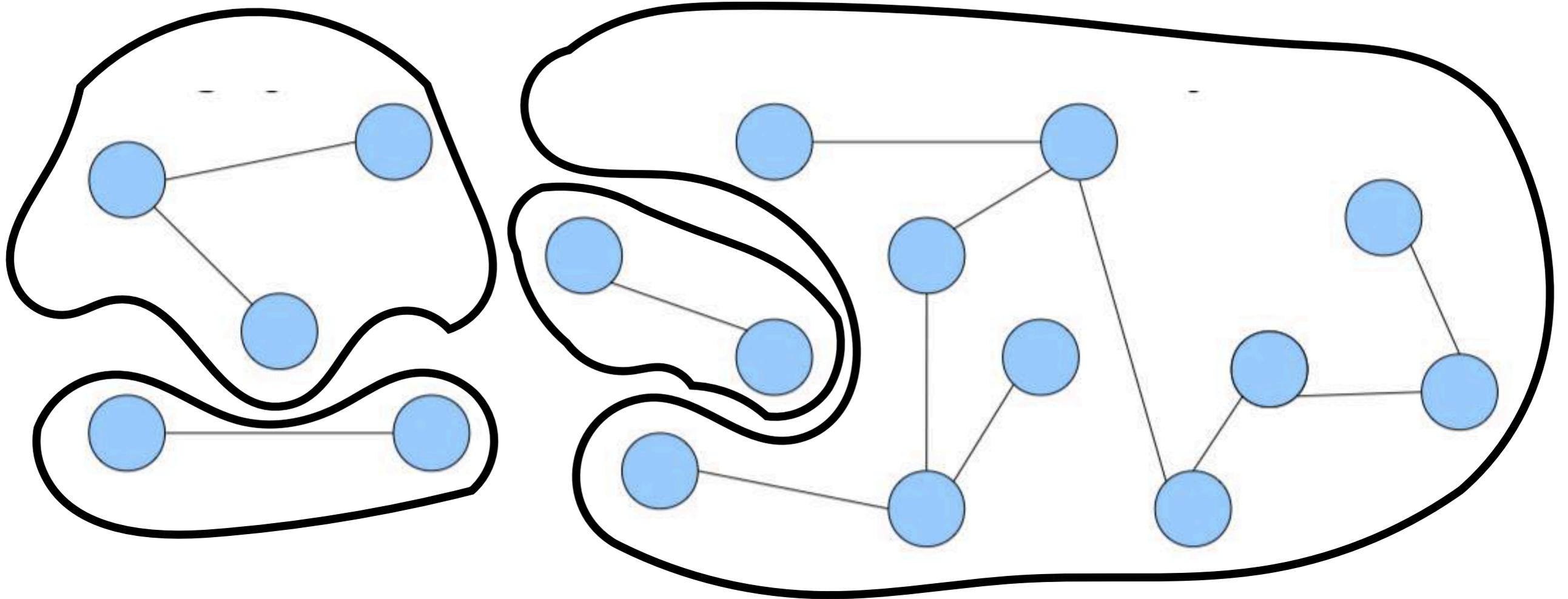


**Calcul de distances**  
**Algorithme de Dijkstra**  
**Distance finie ou infinie**

**Répondre à question**  
**“u et v sont-ils**  
**dans la même**  
**composante ?”**

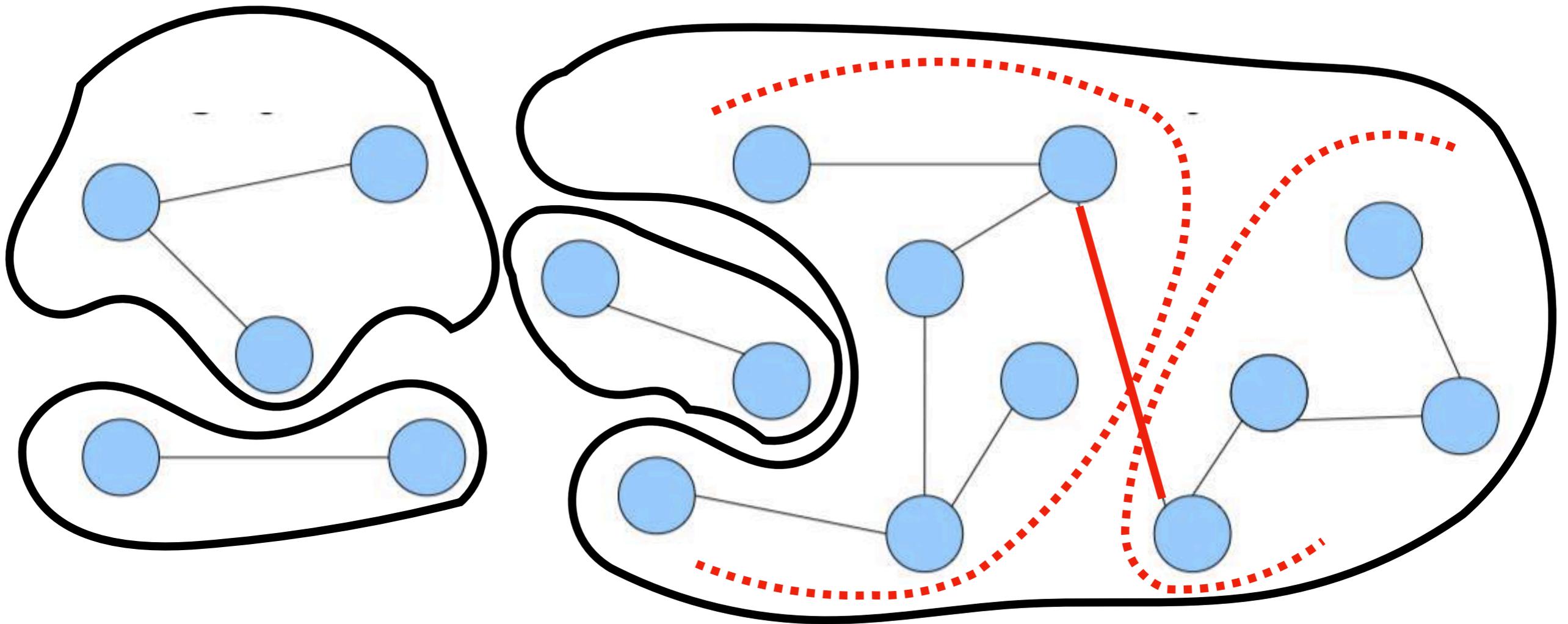
**Quand le graphe change**  
**peut-on mettre à jour**  
**sans tout recalculer ?**

# Forêt : graphe sans cycle



**Composantes connexes dans une forêt  
avec insertion et suppression d'arêtes**

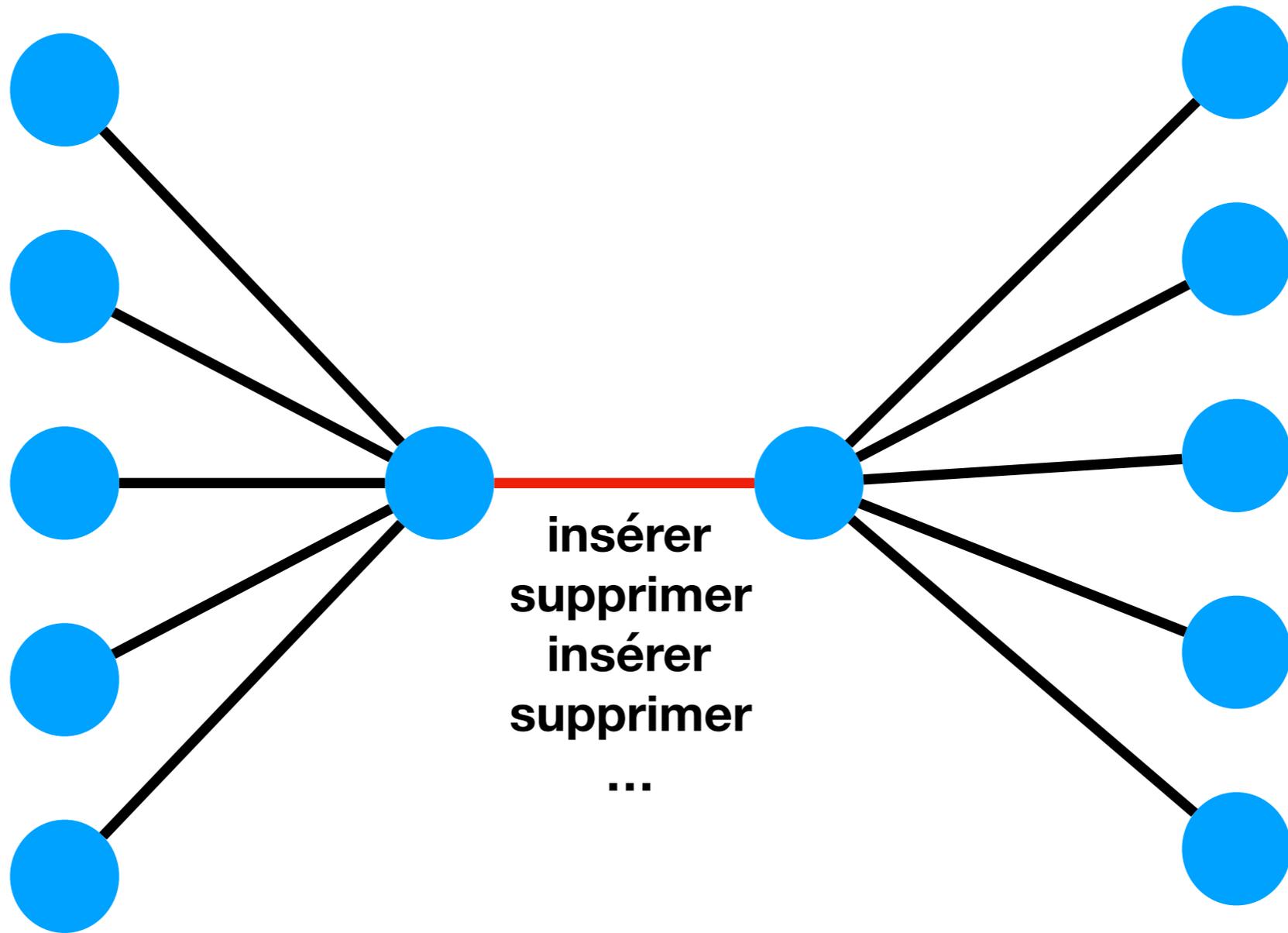
**Trois composantes connexes**  
**Si on ajoute une arête**  
**Fusionner les deux ensembles**  
**Si on supprime une arête**  
**Partitionner l'ensemble**



Temps de réponse à question sur  $u, v$ : **constant** ✓

Temps de mise à jour :  
**taille de la composante** ✗

Peut-on mieux faire ?



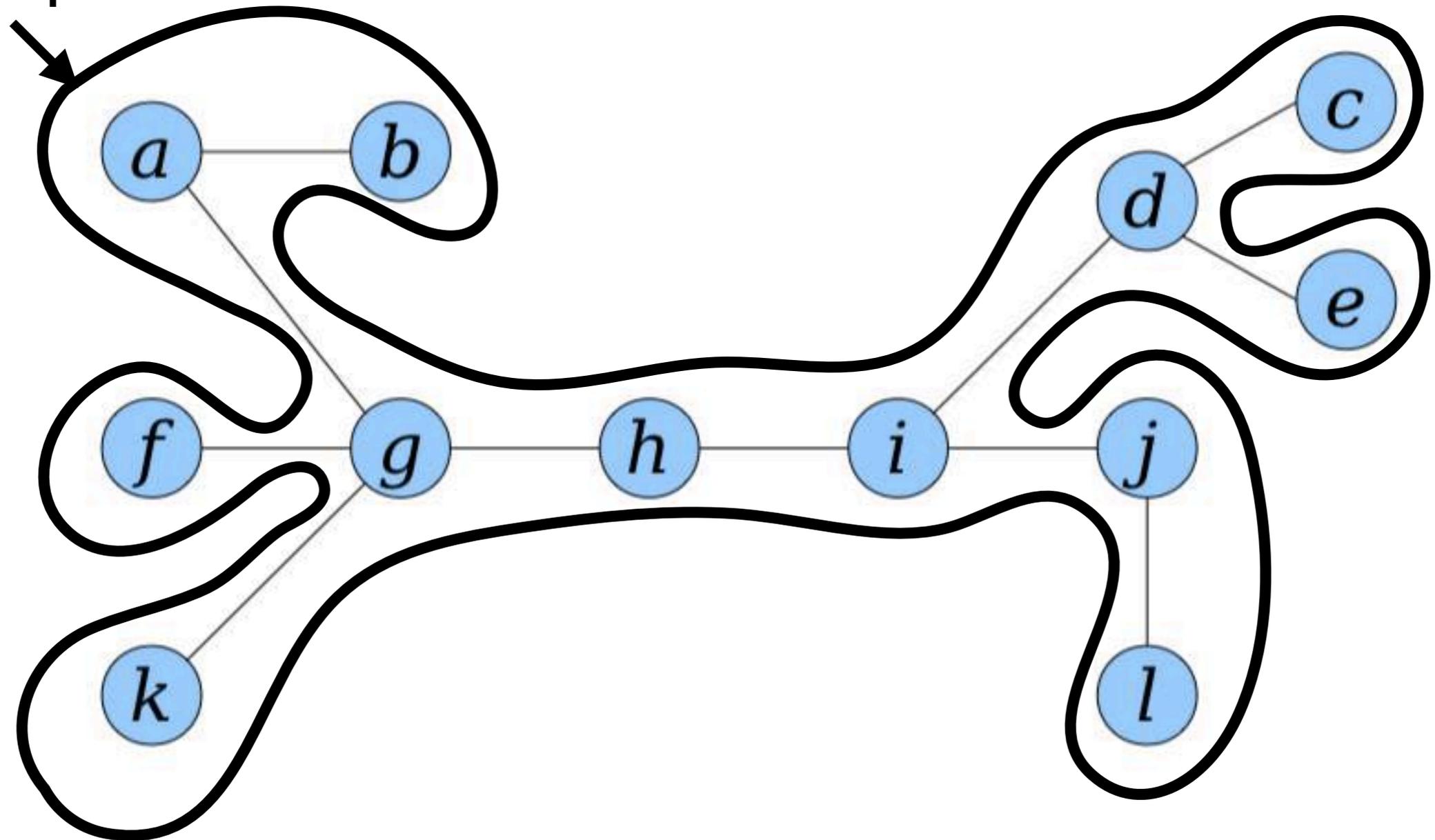
**On dirait qu'il faut payer un temps linéaire  
rien que pour écrire les changements  
des composantes connexes...**

# Idée pour mieux faire : changer de représentation

Introduire redondance pour partition plus facile

On "fait le tour" de chaque arbre  
on marque les sommets qu'on voit

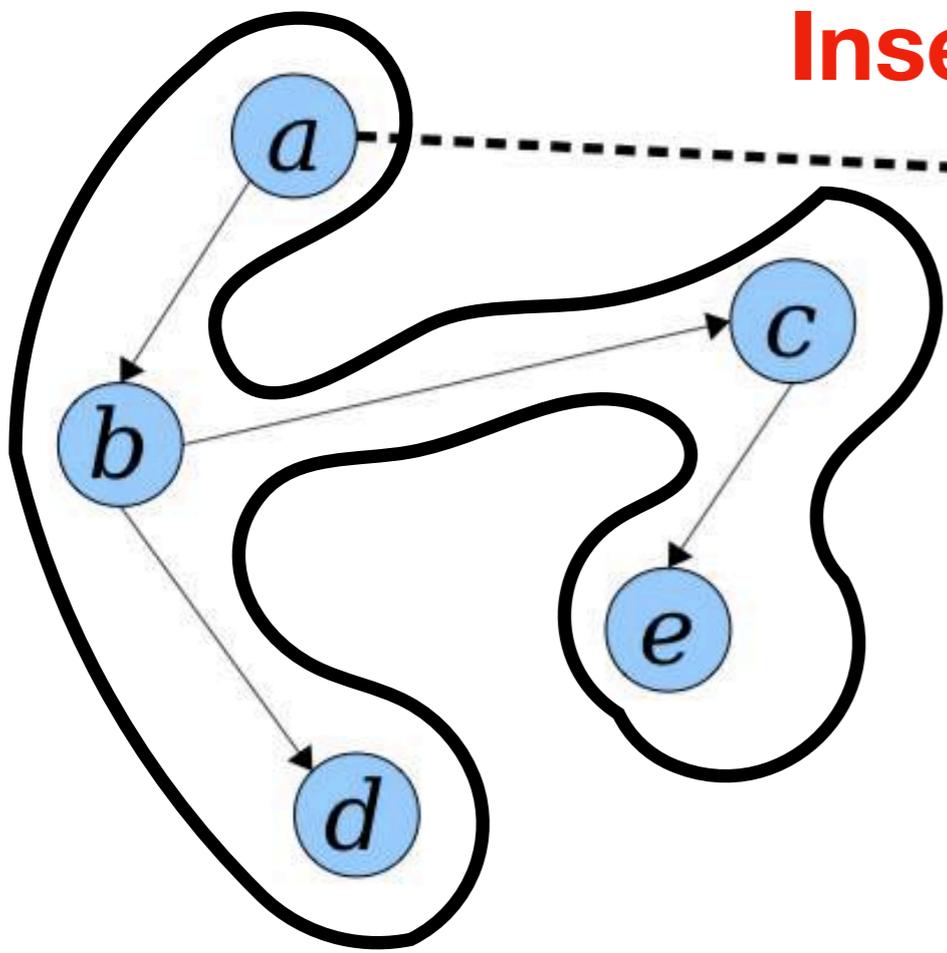
point de départ



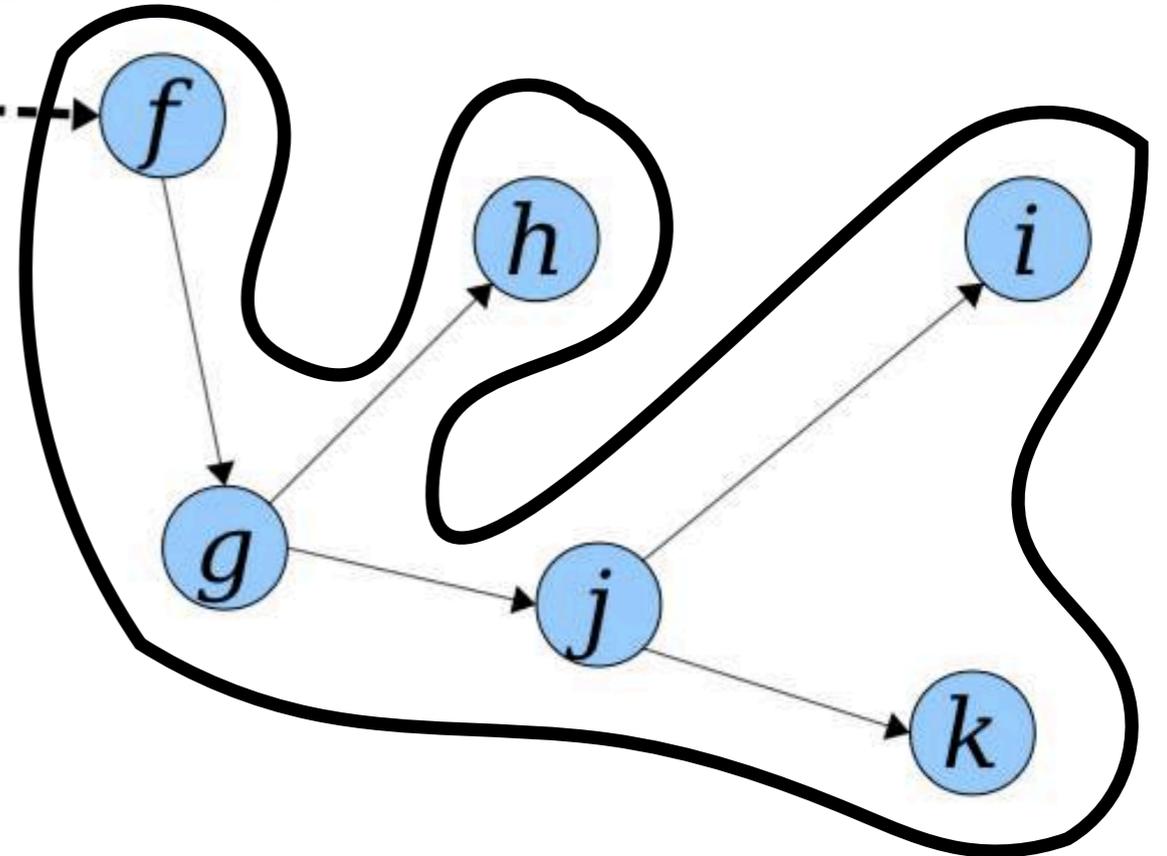
***a b a g h i d c d e d i j l j i h g f g k g a***

Un même sommet peut être vu plusieurs fois

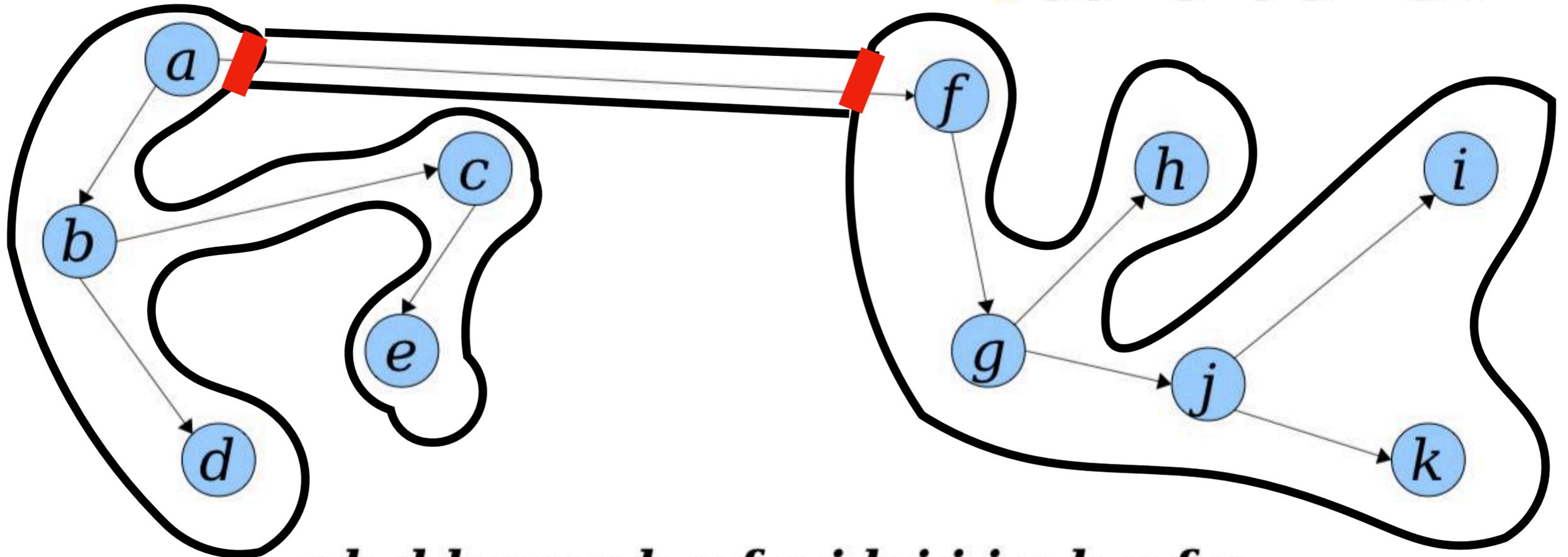
# Insertion d'une arête



*a b d b c e c b a*

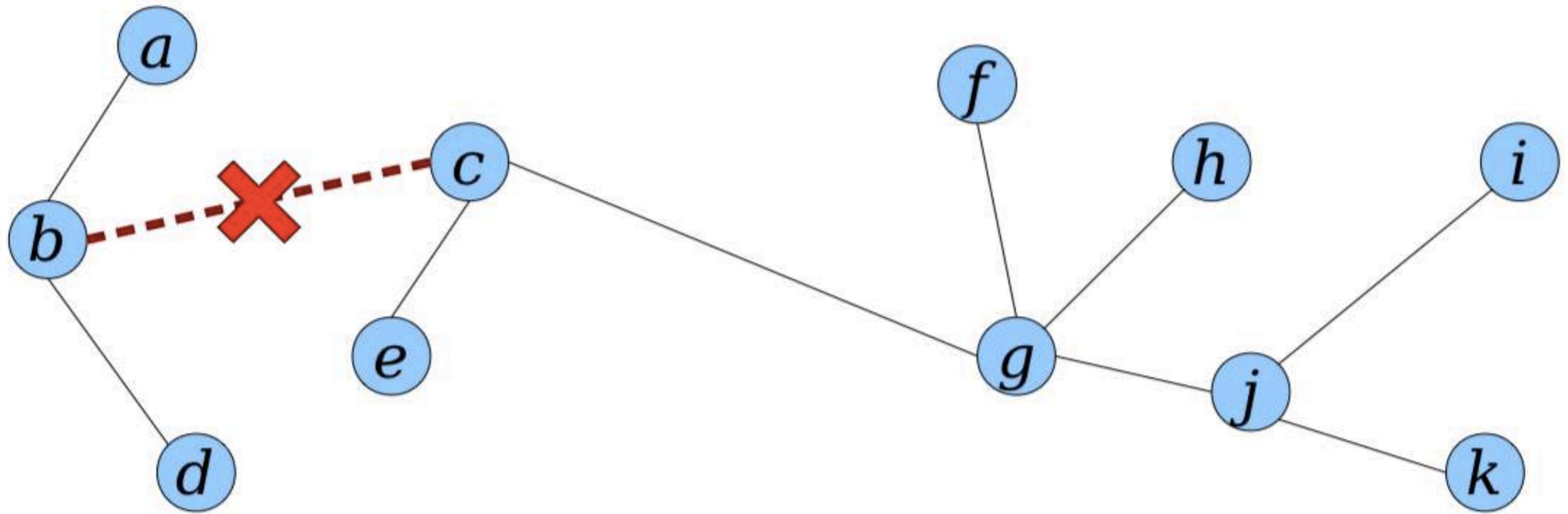


*f g j k j i j g h g f*

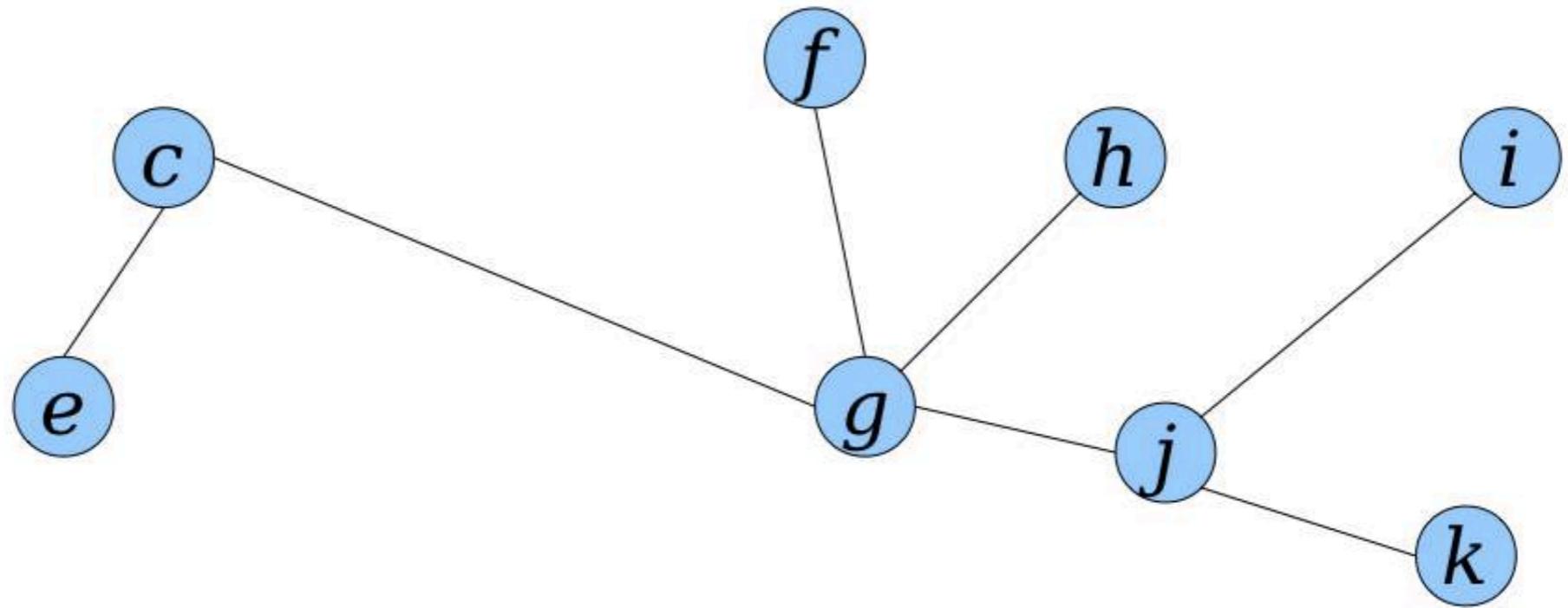
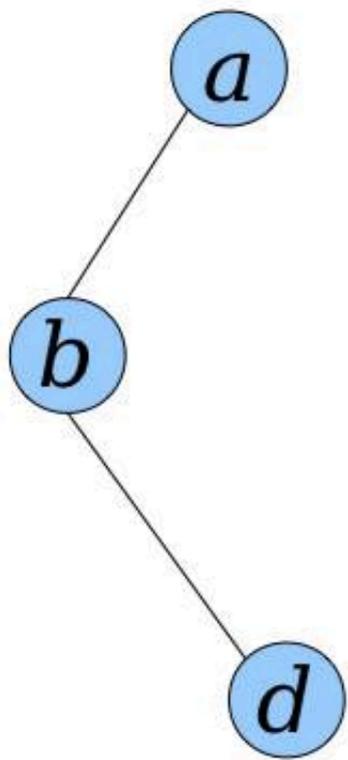


*a b d b c e c b a f g j k j i j g h g f a* Temps constant !

# Suppression d'une arête



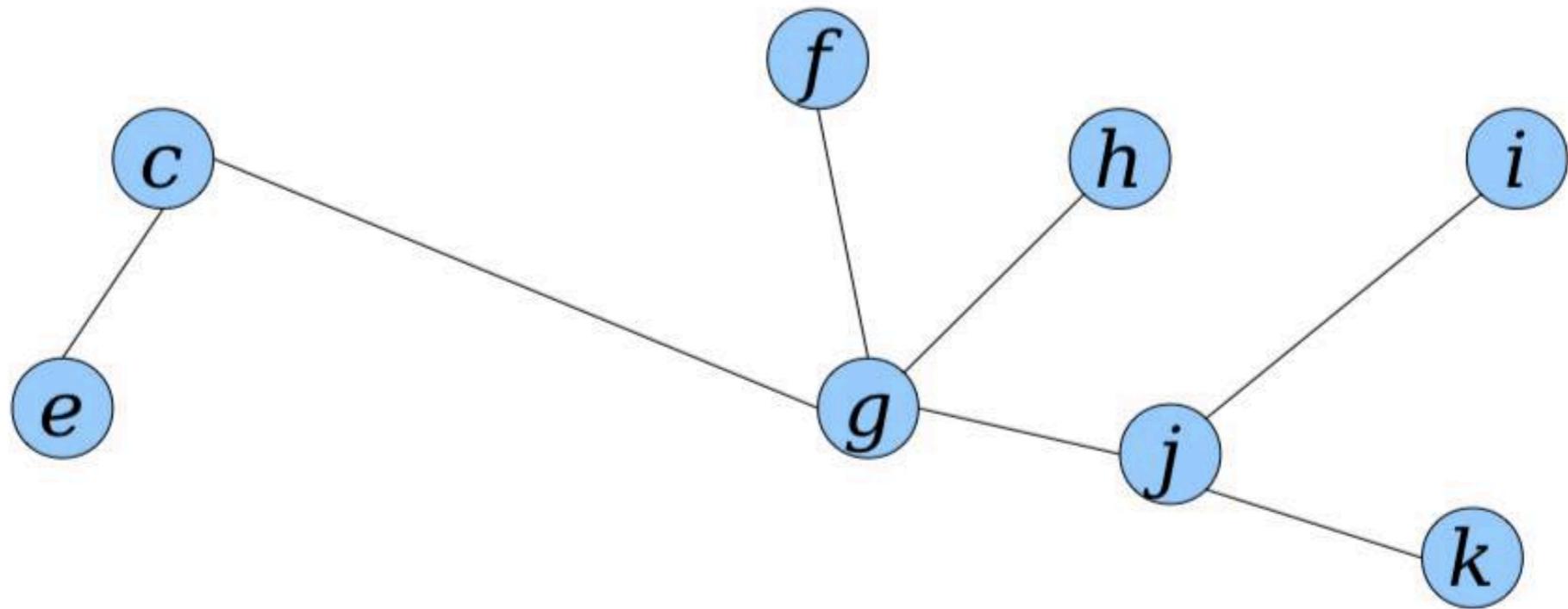
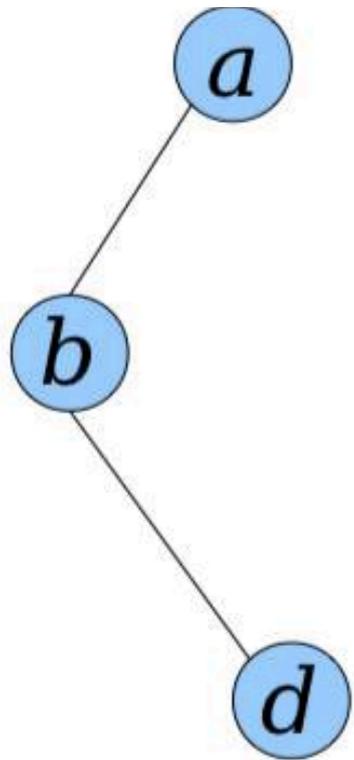
*c e c b a b d b c g j k j i j g h g f g c*



*c e c*

*b a b d b*

*c g j k j i j g h g f g c*

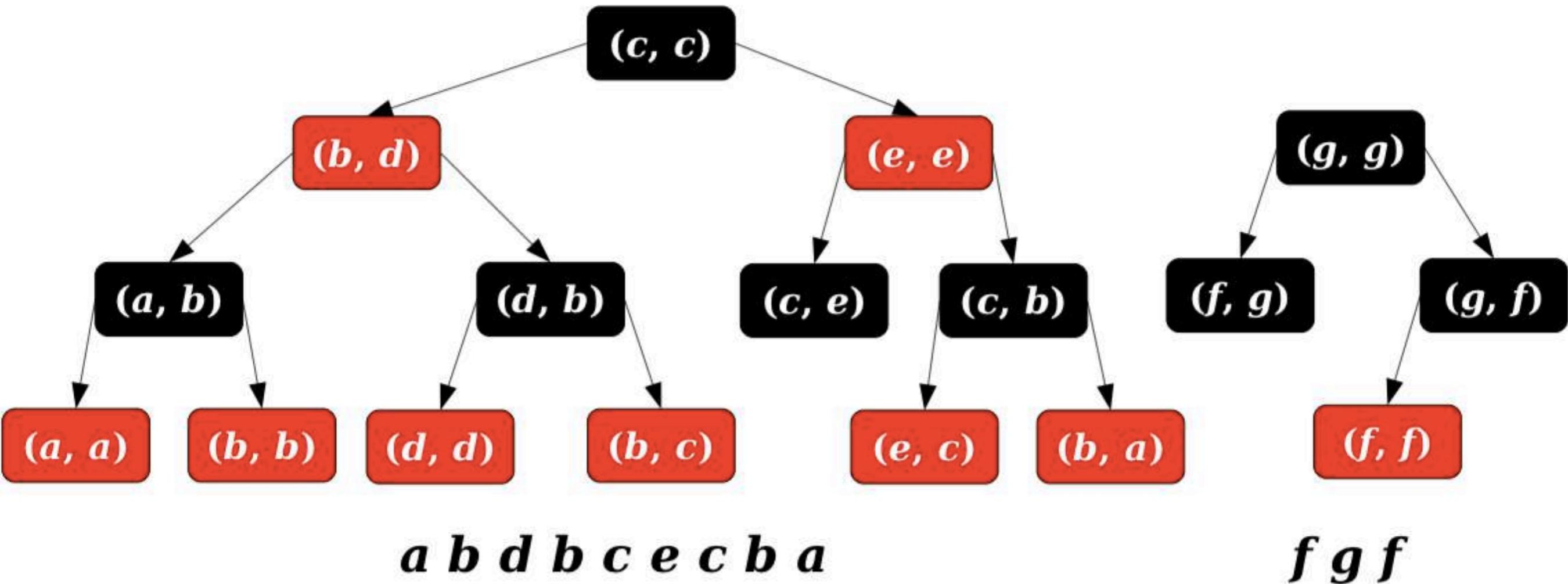


*b a b d b*

*c e c g j k j i j g h g f g c*

**Suppression en temps constant !**

**Comment savoir si deux sommets  $u$  et  $v$  sont dans la même composante connexe ?**



**Si le tour de l'arbre de  $u$  et le tour de l'arbre de  $v$  sont le même**

- organiser le tour dans un arbre équilibré pour répondre en temps logarithmique**

# Morale...

- **Redondance de l'information**
- **Simplicité**

**Conclusion**

# **Pour concevoir des algorithmes pour des graphes qui évoluent au cours du temps**

- **Redondance de l'information**
- **Simplicité de la conception**
- **Ordre aléatoire, calculs modulo nombres aléatoires**
- **Analyses délicates**