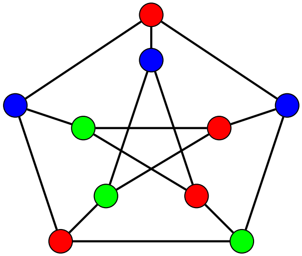
Coloriage de sommets

Coloriage de sommets

Colorer un réseaux signifie attribuer **une couleur** à chacun de ses nœuds de manière que deux nœuds voisins soient de couleur différente.

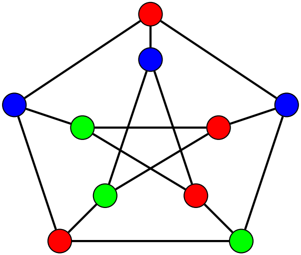


Une coloration du graphe de Petersen avec 3 couleurs

Coloriage de sommets

Définition

Soit un réseau $G = (V, E)$, soit c_v la couleur de chaque noeud $v \in V$ tel que : $\forall (v, w) \in E : c_v \neq c_w$



Une coloration du graphe de Petersen avec 3 couleurs

Applications

- La coloration de sommet a de nombreuse applications pratiques.
- Par exemple dans le domaine de réseaux sans fil où la coloration est la base de protocoles tel que MAC TDMA.
- En général, la coloration de sommets est utilisé comme un moyen pour casser des symétries, un des thèmes principaux dans le calcul distribué.

Modèle

- Chaque noeud à un identifiant unique
- Chaque identifiant peut-être codé en $\log n$ bits, où n est le nombre de noeuds.
- Sans perte de généralité, on suppose que les identifiants sont noté de 1 à n.

Nombre chromatique

Définition

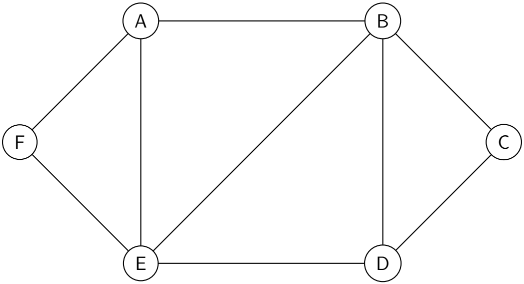
Soit un graphe non orienté $G = (V, E)$, le nombre chromatique $x(G)$ est le nombre minimum de couleur qu'il faut pour résoudre le problème de la coloration de sommets.

Algorithme glouton centralisé

Algorithm 1: Algorithme glouton

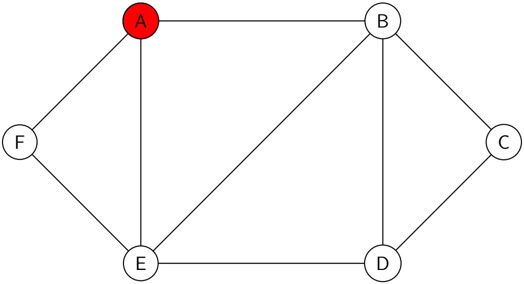
while *il existe un noeud non-coloré v* **do**
  Colorié v avec la couleur minimale qui n'entre pas en conflit avec ces voisins

Exemple



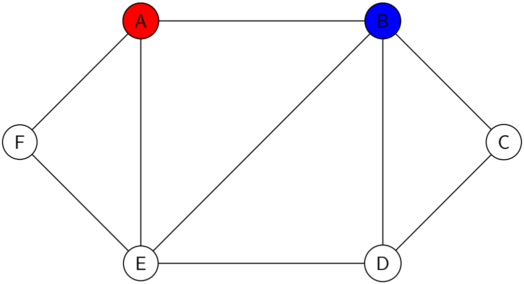
Couleurs : 1 2 3 4

Exemple



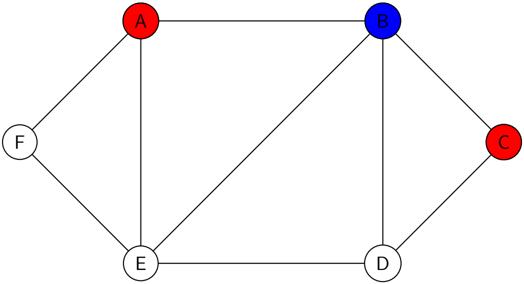
Couleurs : 1 2 3 4

Exemple



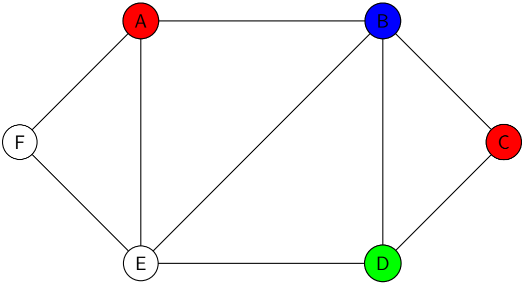
Couleurs : 1 2 3 4

Exemple



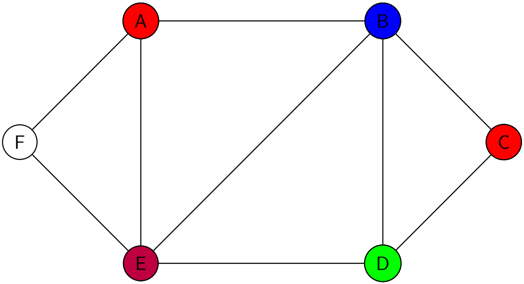
Couleurs : 1 2 3 4

Exemple



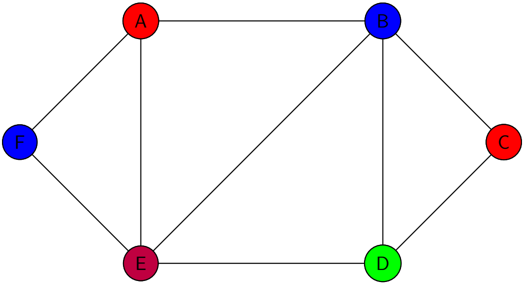
Couleurs : 1 2 3 4

Exemple



Couleurs : 1 2 3 4

Exemple

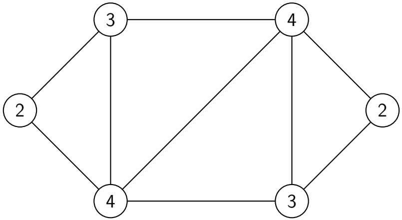


Couleurs : 1 2 3 4

Degré d'un noeud

Définition

Le nombre de voisins d'un noeud v, est noté par $d(v)$, et appelé degré de v.

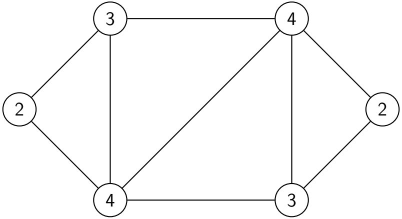


Degrés des noeuds

Degré du graphe

Définition

Le degré du graphe est le degré maximum de tout les noeuds du graphe, on le note $\Delta(G)$ ou simplement Δ.



Graphe de degré 4

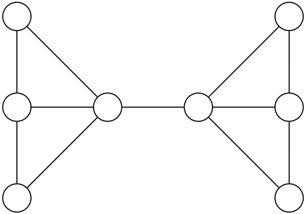
Théorème

L'algorithme glouton centralisé est correct et termine en n étapes. Il utilise $\Delta + 1$ couleurs.

Algorithme glouton distribué

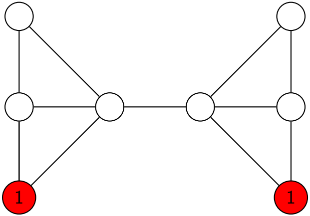
- **Initialisation de v :**
  - $Etat_v := Reveille$
  - $c_v := \emptyset$
  - $\forall u \in N(v) : c_u := \emptyset$
- **Procédure ChoixCouleur :**
  - $couleur := \{1, \ldots, \Delta\}$
  - $\forall u \in N(v) : couleur := couleur - \{c_u\}$
  - $c_v := \min\{couleur\}$
  - $\forall u \in N(v)$: Envoie $< Color, c_v >$
- **Réveil spontané :**
  - **Initialisation**
  - **ChoixCouleur**
- **Lors de la réception de $< Color, c >$ envoyé par u :**
  - Si $Etat_v \neq Reveille$ faire **Initialisation**
  - $c_u := c$
  - **ChoixCouleur**

Approche distribuée : vision locale



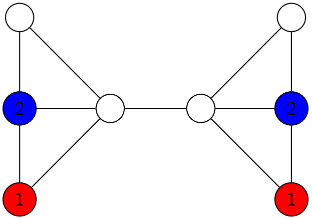
Couleurs : 1 2 3 4

Approche distribuée : vision locale



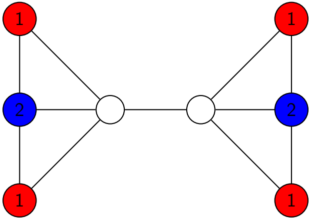
Couleurs : 1 2 3 4

Approche distribuée : vision locale



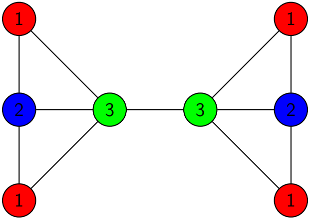
Couleurs : 1 2 3 4

Approche distribuée : vision locale



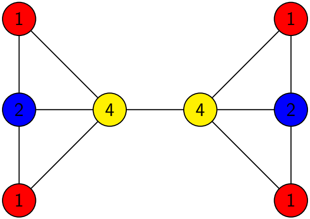
Couleurs : 1 2 3 4

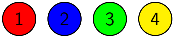
Approche distribuée : vision locale



Couleurs : 1 2 3 4

Approche distribuée : vision locale



Couleurs : 

Question :

Comment éviter ce problème ?

Question :

Comment éviter ce problème ?

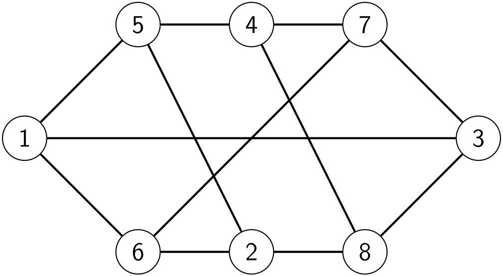
Réponse :

Utiliser les identifiants.

Algorithme glouton distribué

- **Initialisation de v :**
  - $Etat_v := Reveille$
  - $c_v := \emptyset$
  - $\forall u \in N(v) : c_u := \emptyset$ et Envoie $< Color, \emptyset >$ à u
- **Procédure ChoixCouleur :**
  - Si $\forall u \in N(v) \mid id(u) > id(v) \wedge c_u \neq \emptyset$
    - $couleur := \{1, \ldots, \Delta\}$
    - $\forall u \in N(v) : couleur := couleur - \{c_u\}$
    - $c_v := \min\{couleur\}$
    - $\forall u \in N(v)$: Envoie $< Color, c_v >$
- **Réveil spontané :**
  - **Initialisation**
  - **ChoixCouleur**
- **Lors de la réception de $< Color, c >$ envoyé par u :**
  - Si $Etat_v \neq Reveille$ faire **Initialisation**
  - $C_u := c$
  - **ChoixCouleur**

Exemple : Le noeud 1 se réveille.



Couleurs :

Coloration d'Arbres

Arbres

Question :

Combien de couleur faut-il pour colorier un arbre ?

Arbres

Question :

Combien de couleur faut-il pour colorier un arbre ?

Réponse :

- Il faut 2 couleurs.
- Comment faire ?

Lemme coloration dans un arbre

Lemme 1

Le nombre chromatique d'un arbre est inférieur ou égal à 2.

Démonstration.

Si la distance du noeud à la racine est paire, la couleur est 1, sinon (impair) la couleur est 0. Les noeuds à une distance pair on uniquement des voisins à une distance impair et vice versa. Si on suppose que chaque noeud connait son parent (la racine n'a pas de parent) et ses enfants dans l'arbre, cette preuve donne l'algorithme suivant.

□

Arbres et Arborescence

Remarques :

- Cet algorithme fonctionne si une racine est désignée.
- Dans le cas contraire il faut d'abord exécuter un algorithme d'élection.

Algorithme lent de coloration d'arbre

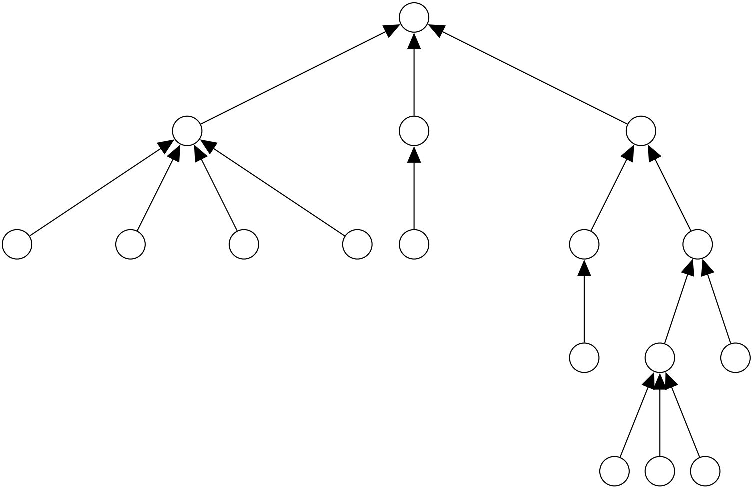
Initialisation de la racine :

- $c_r := 0$
- Envoyer $< Couleur, 0 >$ à tous les enfants.

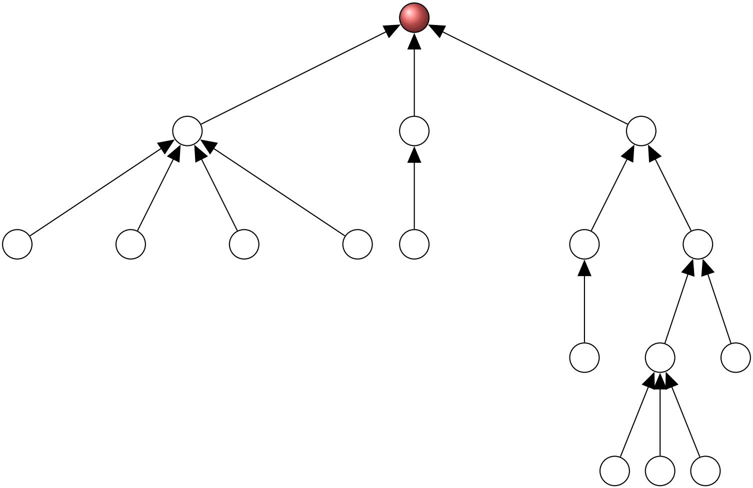
Lors de la réception de $< Couleur, c >$ de parent de v :

- $c_v := |c - 1|$
- Envoyer $< Couleur, c_v >$ à tous les enfants.

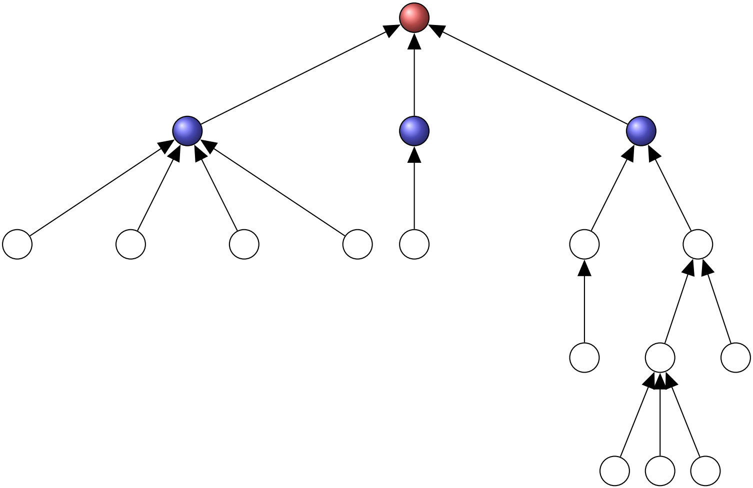
Exemple



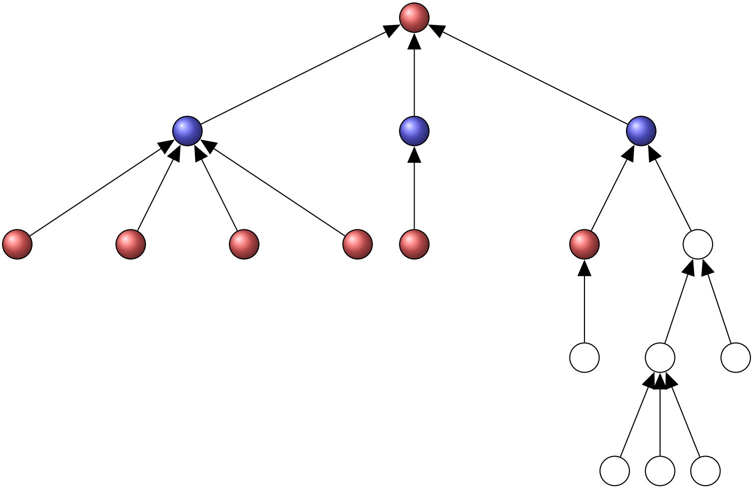
Exemple



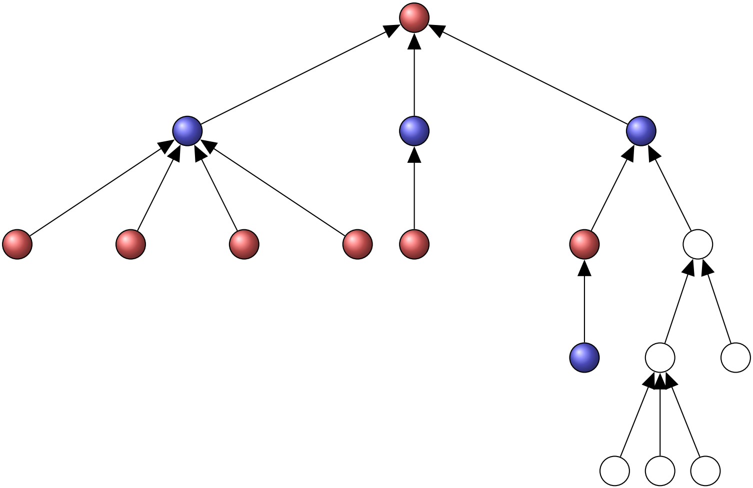
Exemple



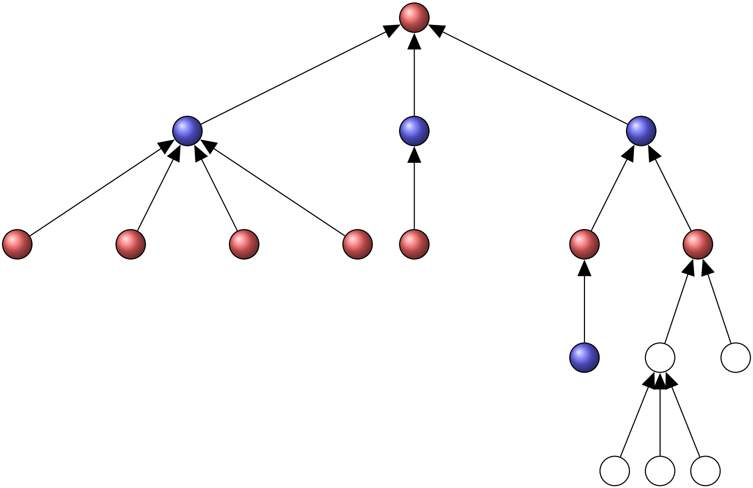
Exemple



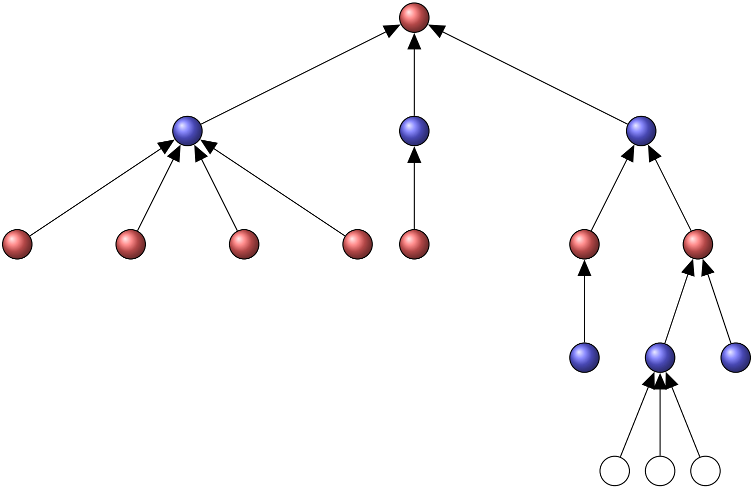
Exemple



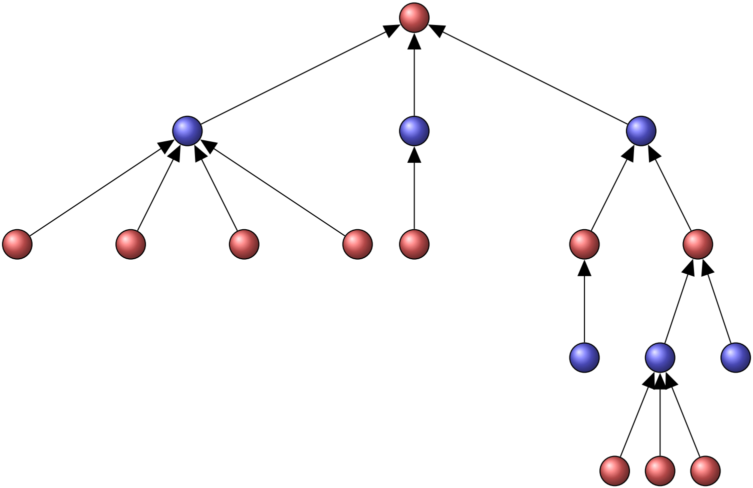
Exemple



Exemple



Exemple



Algorithme lent de coloration d'arbre

Complexities

- Taille des messages : $O(1)$ bits
- Nombre de messages :
  - $O(\text{hauteur de l'arbre})$ bits
  - $O(\log n)$bits.
- Temps :
  - $O(\text{hauteur de l'arbre})$ étapes
  - $O(\log n)$ étapes.

Coloration rapide d'arbres

log étoile

Definition

- $\forall x \leq 2 : \log^* x := 1$
- $\forall x > 2 : \log^* x := 1 + \log^*(\log x)$

log étoile

Remarques

- log-étoile est une fonction qui croie de façon **incroyablement** lentement.
- log-étoile du nombre d'atomes observables dans l'univers (estimé à 10^{80}) est de 5.

Algorithme rapide

Modèle

Synchrone

Algorithme 6-couleurs en $\log^* n$ étapes

Réveil spontané du nœud v :

- $c_v := id_v$ (en binaire)
- Si v est la racine $c_v := 0$
- Envoyer $< Couleur, c_v >$ à tous les enfants.

Lors de la réception de $< Couleur, c >$ de parent de v :

Tant que $c_v > 6$

- $k := \min\{i : c_v[i] \neq c[i]\}$
- $c_v := k(\text{en binaire}).c_v[k]$
- Envoyer $< Couleur, c_v >$ à tous les enfants.

Comparaison binaire

Indice	9	8	7	6	5	4	3	2	1	0
Grand − parent	0	0	1	0	1	1	0	0	0	0
parent	1	0	1	0	**0**	1	0	0	0	0
nœud	0	**1**	1	0	0	1	0	0	0	0

- $c_{parent} := 5.0 := 101.0 := 1010$

Comparaison binaire

Indice	9	8	7	6	5	4	3	2	1	0
$Grand - parent$	0	0	1	0	1	1	0	0	0	0
parent	1	0	1	0	**0**	1	0	0	0	0
nœud	0	**1**	1	0	0	1	0	0	0	0

- $c_{parent} := 5.0 := 101.0 := 1010$
- Remarque $10 \notin \{0, \ldots, 5\}$ on recommence pour parent

Comparaison binaire

Indice	9	8	7	6	5	4	3	2	1	0
Grand − parent	0	0	1	0	1	1	0	0	0	0
parent	1	0	1	0	**0**	1	0	0	0	0
nœud	0	**1**	1	0	0	1	0	0	0	0

- $c_{parent} := 5.0 := 101.0 := 1010$
- Remarque $10 \notin \{0, \ldots, 5\}$ on recommence pour parent
- $c_{nœud} := 8.1 := 1000.1 := 10001$

Comparaison binaire

Indice	9	8	7	6	5	4	3	2	1	0
Grand − parent	0	0	1	0	1	1	0	0	0	0
parent	1	0	1	0	**0**	1	0	0	0	0
nœud	0	**1**	1	0	0	1	0	0	0	0

- $c_{parent} := 5.0 := 101.0 := 1010$
- Remarque $10 \notin \{0, \ldots, 5\}$ on recommence pour parent
- $c_{n\oe ud} := 8.1 := 1000.1 := 10001$
- Remarque $17 \notin \{0, \ldots, 5\}$ on recommence pour nœud

Comparaison binaire

Indice	4	3	2	1	0
Grand − parent	1	0	0	1	0
parent	0	**1**	0	1	0
nœud	1	0	0	0	**1**

- $c_{parent} := 3.1 := 11.1 := 111$

Comparaison binaire

Indice	4	3	2	1	0
$Grand - parent$	1	0	0	1	0
$parent$	0	**1**	0	1	0
$nœud$	1	0	0	0	**1**

- $c_{parent} := 3.1 := 11.1 := 111$
- Remarque $7 \notin \{0, \ldots, 5\}$ on recommence pour parent

Comparaison binaire

$$\begin{array}{r|ccccc} Indice & 4 & 3 & 2 & 1 & 0 \\ Grand-parent & 1 & 0 & 0 & 1 & 0 \\ parent & 0 & \mathbf{1} & 0 & 1 & 0 \\ nœud & 1 & 0 & 0 & 0 & \mathbf{1} \end{array}$$

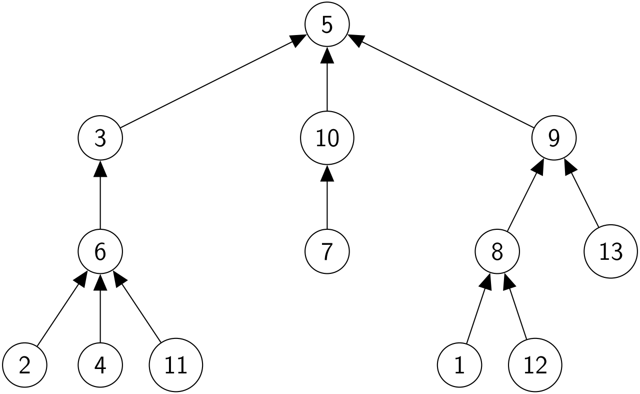
- $c_{parent} := 3.1 := 11.1 := 111$
- Remarque $7 \notin \{0, \ldots, 5\}$ on recommence pour parent
- $c_{nœud} := 0.1 := 01$

Comparaison binaire

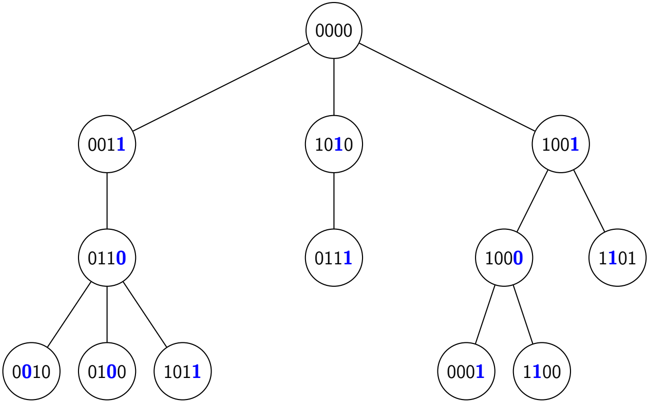
Indice	4	3	2	1	0
Grand − parent	1	0	0	1	0
parent	0	**1**	0	1	0
nœud	1	0	0	0	**1**

- $c_{parent} := 3.1 := 11.1 := 111$
- Remarque $7 \notin \{0, \ldots, 5\}$ on recommence pour parent
- $c_{nœud} := 0.1 := 01$
- Remarque $1 \in \{0, \ldots, 5\}$ le calcul s'arête pour nœud.

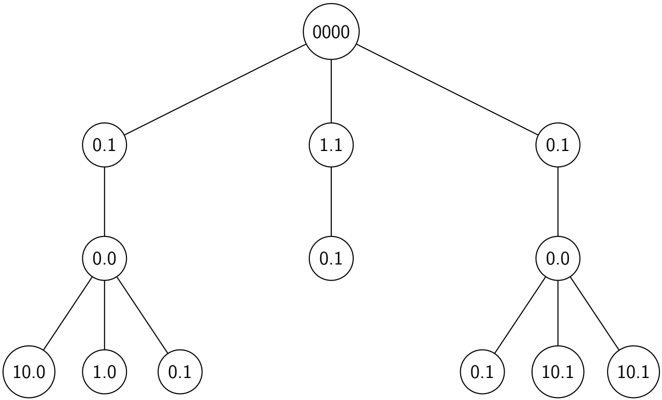
Exemple



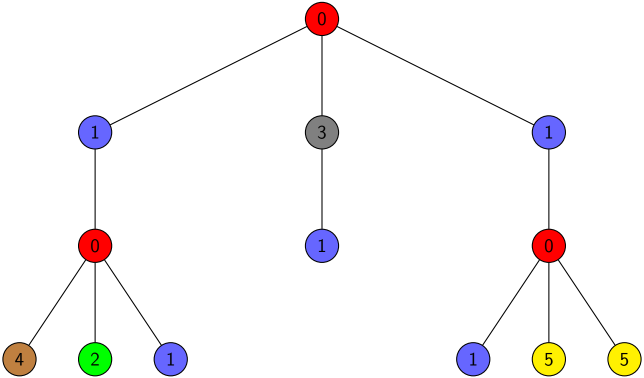
Exemple


0000
0011
1010
1001
0110
0111
1000
1101
0010
0100
1011
0001
1100

Exemple



Exemple

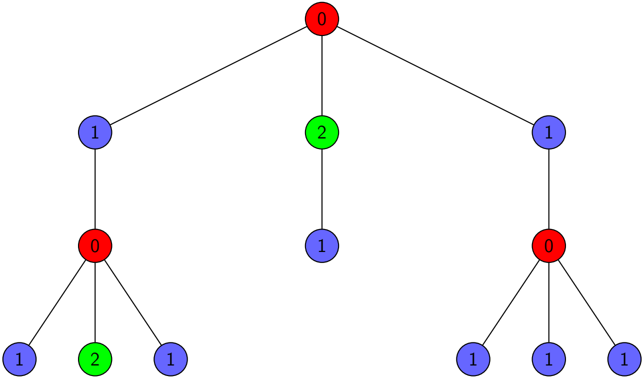


Algorithme 6-to-3 en $\log^* n$ étapes

Pour chaque nœud v :

- Si $c_v \in \{3, 4, 5\}$
- $c_v :=$ choisir la plus petite couleur dans $\{0, 1, 2\}$ qui n'est pas utilisée par les voisins.

Exemple



Coloration rapide d'anneau

Coloration rapide d'anneau

Modèle

- Synchrone
- les noeuds de l'anneau connaisse n le nombre de nœud
- Les nœuds de l'anneau ont une notion commune de droite et de gauche

Algorithme 3-couleurs en $\log^* n$ étapes

Réveil spontané du nœud v

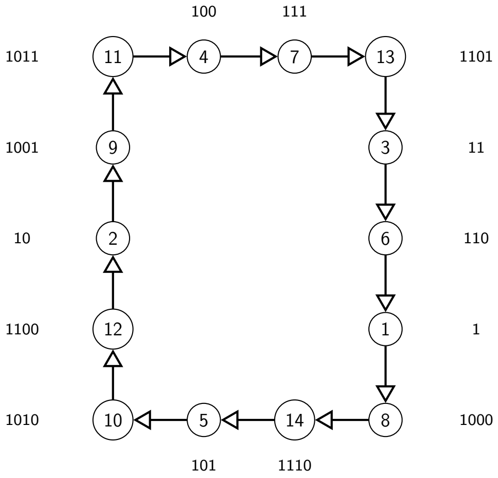
- $c_v := id_v$ (en binaire)
- Envoyer $< Couleur, c_v >$ au voisin de droite.

Lors de la réception de $< Couleur, c >$ du voisin de gauche :

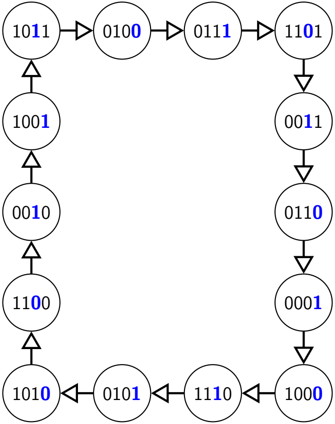
Tant que $c_v > 3$

- $k := \min\{i : c_v[i] \neq c[i]\}$
- $c_v := k(\text{en binaire}).c_v[k]$
- Envoyer $< Couleur, c_v >$ au voisin de droite.
- Si possible réduction

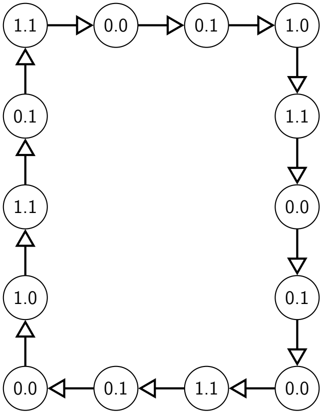
Exemple


100
111
1011
11
4
7
13
1101
1001
9
3
11
10
2
6
110
1100
12
1
1
1010
10
5
14
8
1000
101
1110

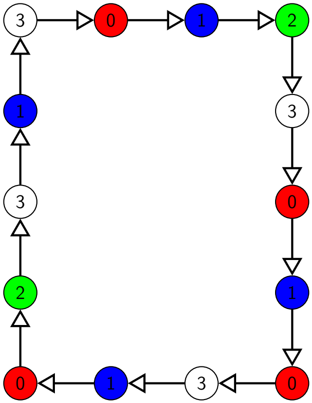
Exemple


1011
0100
0111
1101
1001
0011
0010
0110
1100
0001
1010
0101
1110
1000

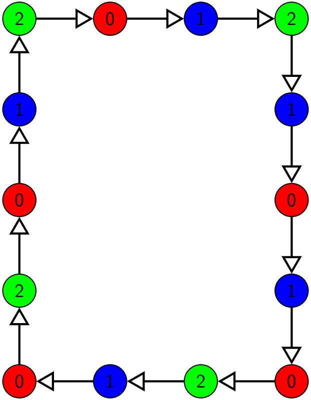
Exemple



Exemple



Exemple : réduction



Références

- Distributed Computing : A Locality-Sensitive Approach. David Peleg. Society for Industrial and Applied Mathematics (SIAM), 2000, ISBN 0-89871-464-8
- http ://www.dcg.ethz.ch/lectures/podc/