Le couloir des prisonniers

Par Enigmath, le 9 janvier 2026

Énoncé

Dans une prison, 100 cellules sont alignées les unes à côté des autres le long d'un couloir. Chaque cellule est totalement isolée des autres : il n'y a ni fenêtres ni moyens de communication directe entre les prisonniers. Le gardien lance alors un défi aux prisonniers.
Chaque jour, le gardien rend visite à chaque prisonnier deux fois : une fois le matin et une fois le soir. Lors de la visite du matin, chaque prisonnier choisit un numéro (entier positif) et l'annonce au gardien. Lors de la visite du soir, le gardien annonce au prisonnier au plus deux numéros, correspondant aux numéros choisis le matin même par les prisonniers occupant les cellules immédiatement adjacentes à la sienne. Le gardien ne précise pas quel numéro provient de la cellule de gauche ou de celle de droite. Les cellules situées aux deux extrémités du couloir, n'ayant qu'un seul voisin, ne reçoivent qu'un seul numéro.
À tout moment, deux prisonniers occupant des cellules adjacentes ne doivent jamais choisir le même numéro (sinon, tous les prisonniers sont exécutés).
Avant d'être séparés et de commencer le défi, les prisonniers se réunissent afin d'élaborer collectivement un « mode d'emploi » décrivant la stratégie qu'ils suivront. Une fois ce plan établi, ils sont endormis et perdent tout souvenir de l'élaboration. À leur réveil, le matin, chacun se retrouve dans une cellule, avec pour seule indication le mode d'emploi qu'ils ont rédigé collectivement, et le numéro de sa propre cellule (les numéros des cellules ont été permutés arbitrairement).
Les prisonniers sont libérés par le gardien le premier jour où chacun d'eux a choisi un numéro appartenant à l'ensemble $\{1, 2, 3\}$.

Questions :

1. 🌶[1.3] Trouver un mode d'emploi permettant d'atteindre cet objectif en 97 jours.
2. 🌶[2.0] Trouver un mode d'emploi permettant d'atteindre cet objectif en 7 jours.
3. 🌶[2.1] Trouver un mode d'emploi permettant d'atteindre cet objectif en 5 jours.

Solution

1. La prison peut être modélisée par un graphe en forme de *chemin*. L'objectif est d'obtenir un *coloriage propre* avec trois couleurs $\{1, 2, 3\}$, c'est-à-dire que deux nœuds voisins aient toujours des couleurs différentes. On note $N(v)$ l'ensemble des voisins de v, $c(v)$ la couleur actuelle de v et $M_v = \{c(u) \mid u \in N(v)\}$. Le mode d'emploi peut être décrit par l'algorithme suivant :

$$\text{Si } c(v) > \max M_v, \quad c(v) \leftarrow \min\big(\{1, 2, 3\} \setminus M_v\big).$$

Lemme. *Étant donné un coloriage propre avec x couleurs, une étape de l'algorithme produit un coloriage propre avec $\max(3, x - 1)$ couleurs.*

Démonstration. On note $c'(v)$ la nouvelle couleur du nœud v. Deux cas se présentent :

— $c(v) < \max M_v = c(u)$ et v est *passif* et ne change pas sa couleur :

$$c'(v) = c(v) \leq c(u) - 1 \leq x - 1.$$

— $c(v) > \max M_v$ et v est *actif* :

$$c'(v) = \min\big(\{1, 2, 3\} \setminus M_v\big) \leq 3.$$

Pour montrer que c' reste un coloriage propre, considérons un arête $\{u, v\}$:
— Si u et v sont passifs, $c'(u) = c(u) \neq c(v) = c'(v)$.
— Sinon, supposons que u est actif : on a $c(u) > c(v)$ et v est passif. Donc, $c'(v) = c(v) \in M_u$, alors que $c'(u) \notin M_u$, donc $c'(v) \neq c'(u)$. □

Ainsi, en partant de 100 couleurs (les prisonniers choisissant initialement le numéro de leur propre cellule), l'algorithme réduit progressivement le nombre de couleurs : après 1 jour, il reste 99 couleurs ; après 2 jours, 98 couleurs ; ... ; et après 97 jours, il ne reste plus que 3 couleurs.

2. Montrons comment réduire le nombre de couleurs de $2^x + 1$ à $2x + 1$ en une seule journée. On pourra alors passer de $100 < 2^8 + 1$ couleurs à $2 \times 8 + 1 = 16 + 1 = 2^4 + 1$ couleurs, puis à $2 \times 4 + 1 = 8 + 1 = 2^3 + 1$ couleurs, et enfin à $2 \times 3 + 1 = 6 + 1 = 7$ couleurs, le tout en 3 jours. En appliquant ensuite la stratégie de la question précédente pendant les 4 jours restants, on obtient finalement 3 couleurs après 7 jours.

   L'algorithme permettant de passer de $2^x + 1$ à $2x + 1$ couleurs est le suivant. Avant de décrire l'étape $2^x + 1 \to 2x + 1$ qui va être répétée jour après jour, nous introduisons la notion de *successeur*.

   Le premier matin, les prisonniers choisissent le numéro de leur cellule ; on appelle *successeurs* d'un sommet v les voisins dont le numéro de cellule est (strictement) supérieur au sien. Il est important de noter que les successeurs sont déterminés lors du premier soir, et qu'ils restent fixés pendant toute la durée de l'exécution de l'algorithme $2^x + 1 \to 2x + 1$. Un sommet peut avoir 0, 1 ou 2 successeurs.

   Nous réservons une couleur spéciale, dédiée uniquement aux sommets ayant exactement deux successeurs. Ces sommets choisissent cette couleur dès le deuxième jour et la conservent jusqu'à la fin de l'exécution de l'algorithme $2^x + 1 \to 2x + 1$. Cette règle est valide car deux sommets adjacents ne peuvent pas avoir simultanément deux successeurs. En mettant ainsi de côté la couleur réservée aux sommets ayant deux successeurs, il reste à montrer, pour les autres nœuds, que l'on peut passer de 2^x à $2x$ couleurs en une seule étape.

   Nous pouvons ainsi nous restreindre à l'étude des sommets ayant 0 ou 1 successeur. Chaque sommet u connaît alors deux valeurs :
   — $c_0(u)$, la couleur actuelle du sommet u ;
   — $c_1(u)$, la couleur actuelle de son successeur.
   Si un sommet ne possède aucun successeur, il procède comme s'il en avait un de couleur différente de $c_0(u)$.

   On peut interpréter $c_0(u)$ et $c_1(u)$ comme des chaînes binaires de longueur x, représentant des entiers compris entre 0 et $2^x - 1$. On sait que la couleur actuelle du sommet u est différente de celle de son successeur, c'est-à-dire

$$c_0(u) \neq c_1(u).$$

   Il existe donc au moins une position de bit où ces deux chaînes binaires diffèrent.

   On définit alors :
   — $i(u) \in \{0, 1, \ldots, x - 1\}$ comme l'indice du premier bit où $c_0(u)$ et $c_1(u)$ diffèrent ;
   — $b(u) \in \{0, 1\}$ comme la valeur du bit d'indice $i(u)$ dans $c_0(u)$.
   Enfin, le sommet u choisit

$$c(u) = 2\, i(u) + b(u) \in \{0, 1, \ldots, 2x - 1\}$$

   comme nouvelle couleur.

   Soient u et v deux sommets voisins. Si l'un des deux possède deux successeurs, on sait déjà que leurs couleurs finales seront distinctes (c'est précisément le rôle de la couleur spéciale).

Sinon, l'un des deux est le successeur de l'autre ; supposons sans perte de généralité que v soit le successeur de u. Par définition, on a $c_1(u) = c_0(v)$. Il reste à montrer que

$$c(u) \neq c(v).$$

Deux cas sont à distinguer :

(*a*) $i(u) = i(v) = i$. Dans ce cas, $b(u)$ est la valeur du bit d'indice i dans $c_0(u)$, tandis que $b(v)$ est la valeur du bit d'indice i dans $c_1(u) = c_0(v) \neq c_0(u)$. Par la définition de $i(u)$, ces deux bits sont différents. On en déduit que $b(u) \neq b(v)$, et donc que $c(u) \neq c(v)$.

(*b*) $i(u) \neq i(v)$. Dans ce cas, quels que soient les choix de $b(u) \in \{0, 1\}$ et $b(v) \in \{0, 1\}$, on a nécessairement $c(u) \neq c(v)$.

3. La structure générale du nouvel algorithme reprend celle du précédent. En particulier, nous utilisons la même définition de successeur. Cette fois cependant, au lieu d'effectuer une réduction de $2^x + 1$ vers $2x + 1$, nous allons réaliser une réduction de

$$h(x) + 1 \text{ vers } 2x + 1,$$

avec $h(x) \triangleq \binom{2x}{x}$. La technique de choix de la nouvelle couleur est donc différente : nous n'interprétons plus les couleurs comme des chaînes de bits, mais comme des ensembles. Pour cela, on définit $H(x)$ comme l'ensemble de tous les sous-ensembles

$$X \subseteq \{1, 2, \ldots, 2x\} \quad \text{tels que} \quad |X| = x.$$

Il y a exactement $h(x)$ tels sous-ensembles. On peut donc fixer une bijection

$$L : \{1, 2, \ldots, h(x)\} \longrightarrow H(x).$$

Comme on a $c_0(v) \neq c_1(v)$,

$$L(c_0(v)) \neq L(c_1(v)).$$

Comme $L(c_0(v))$ et $L(c_1(v))$ sont deux sous-ensembles de taille x, il s'ensuit que

$$L(c_1(v)) \setminus L(c_0(v)) \neq \varnothing.$$

On choisit alors la nouvelle couleur $c(v)$ d'un sommet $v \in V$ comme suit :

$$c(v) = \min L(c_1(v)) \setminus L(c_0(v)).$$

Supposons que v soit le successeur de u. On a $c_1(u) = c_0(v)$. On pose $A = L(c_1(u)) = L(c_0(v))$, $B = L(c_0(u))$, $C = L(c_1(v))$. on a donc que $A \setminus B$ et $C \setminus A$ sont disjoints. En particulier, $c(u) = \min A \setminus B \neq \min C \setminus A = c(v)$.

On pourra alors passer de $100 < h(10) + 1$ couleurs à $2 \times 10 + 1 = 20 + 1 = h(3) + 1$ couleurs, puis à $2 \times 3 + 1 = 6 + 1 = h(2) + 1$ couleurs, et enfin à $2 \times 2 + 1 = 4 + 1 = 5$ couleurs, le tout en 3 jours. En appliquant ensuite la stratégie de la première question pendant les 2 jours restants, on obtient finalement 3 couleurs après 5 jours.

Notes et références

Cette énigme, tout comme *The Cyclic Prisoners*, s'inscrit dans le cadre de l'algorithmique distribuée, où des agents identiques tentent de se coordonner malgré une symétrie parfaite. Elle est issue de [1]. Plus précisément, le modèle sous-jacent est celui des *identifiants uniques* (par exemple des adresses IP) dans un réseau d'ordinateurs interconnectés, que l'on restreint ici à une topologie de graphe très simple : un chemin non orienté.

En algorithmique distribuée, on considère un ensemble de nœuds (ou ordinateurs) reliés par des canaux de communication. Chaque nœud exécute le même algorithme déterministe, échange des messages uniquement avec ses voisins immédiats, et doit décider localement de son état final. Un problème fondamental, souvent utilisé comme exemple introductif, est celui du *coloriage*, où chaque nœud doit choisir une couleur de sorte que deux voisins n'aient jamais la même couleur.

Dans le modèle des identifiants uniques, la symétrie entre les nœuds est levée grâce aux identifiants. Chaque nœud est initialement colorié par son identifiant. Le défi algorithmique consiste alors à réduire progressivement le nombre de couleurs, tout en garantissant un coloriage valide à chaque étape. L'énigme présentée ici peut être vue comme une reformulation narrative de ce problème classique, où les prisonniers jouent le rôle de nœuds exécutant un algorithme distribué sur un chemin.
L'algorithme rapide de coloriage en 3 couleurs utilisé dans la deuxième question a été initialement proposé par Cole et Vishkin [2] et affiné par Goldberg et al. [3]. Dans la littérature, il est couramment désigné sous le nom d'*algorithme de Cole–Vishkin*.

Sources

[1] Juho Hirvonen and Jukka Suomela. Distributed algorithms 2020. *Finland : Aalto University, Dec*, 11:221, 2021.

[2] Richard Cole and Uzi Vishkin. Deterministic coin tossing with applications to optimal parallel list ranking. *Information and Control*, 70(1):32–53, 1986.

[3] Andrew Goldberg, Serge Plotkin, and Gregory Shannon. Parallel symmetry-breaking in sparse graphs. In *Proceedings of the nineteenth annual ACM symposium on Theory of computing*, pages 315–324, 1987.