Jeux d'inondation dans les graphes

Aurélie Lagoutte

Jeux d'inondation dans les graphes

Aurélie LAGOUTTE

12 août 2010

LIP, ENS Lyon
IXXI: Institut des Systèmes Complexes
5, rue du Vercors
69007 LYON

Table des matières

1 Introduction

1.1 Contexte

Le sujet de ce rapport est inspiré des jeux Flood-It et Mad Virus (jouables respectivement aux liens [1] et [2]). Ces deux jeux sont des casses-têtes qui fonctionnent selon le même mécanisme. Dans les deux cas, on joue avec un plateau dont les cases sont colorées : grille carrée de taille $n \times n$ pour Flood-It, grille hexagonale pour Mad Virus. Deux cases du plateau sont dites adjacentes si elles partagent un côté commun. Une zone du plateau est définie comme un ensemble connexe maximal de cases de la même couleur. Une étape du jeu consiste alors a "inonder" avec une nouvelle couleur la zone qui contient la case en haut à gauche pour Flood-It (une case arbitraire pour Mad Virus), c'est-à-dire changer toutes les cases de cette zone avec cette nouvelle couleur. Cela entraine la connexion de cette zone a toues les zones voisines ayant cette couleur. Le but du jeu est de colorer tout le plateau avec une unique couleur et en un nombre minimum d'étapes. Ce problème d'inondation est donc un problème d'optimisation combinatoire.

Dans cette version du jeu, la zone à inonder est fixée et le joueur a juste à choisir une nouvelle couleur à chaque étape. La figure 1 présente un plateau avec 5 couleurs et les effets d'un choix de couleurs successives. Il existe une autre version du jeu où, à chaque étape, le joueur est libre de choisir la zone qu'il veut inonder et bien sûr la nouvelle couleur de cette zone. Cette autre version est appelée Free-Flood-It dans le cas de la grille carrée.

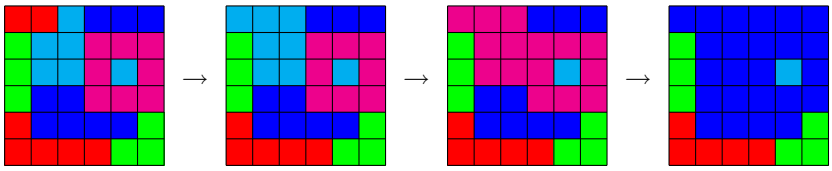


FIG. 1 – Séquence d'inondations bleu clair-rose-bleu foncé pour le jeu Flood-It.

Un article récent publié dans la conférence "Fun with Algorithms" [5, 6] montre que le calcul du nombre mininum d'inondations dans le jeu Flood-It est NP-dur dès qu'il y a 3 couleurs (avec 2 couleurs, la résolution est évidente car toute séquence d'inondations est une alternance des deux couleurs). Cet article est entièrement consacré au cas de la grille carrée et présente plusieurs problèmes ouverts, en particulier concernant la variante Free-Flood-It.

Le mécanisme de jeu de Flood-It et Mad Virus peut être généralisé à n'importe quel graphe coloré (même définition d'une zone, même définition d'une étape de jeu). Les graphes correspondants à Flood-It et Mad Virus (les sommets étant les cases et l'adjacence celle décrite au début) sont très particuliers. L'objectif était de mieux comprendre la dynamique du jeu en l'étudiant sur les graphes en général, notamment pour chercher quels aspects de la topologie jouent sur la difficulté de la résolution du casse-tête. Cette approche nous a permis de mieux saisir l'essence du problème d'inondation et de résoudre un des problèmes ouverts proposés dans [5].

Dans une première partie introductive (Partie 1), donnant les définitions et les notations nécéssaires à la compréhension du rapport, nous montrons qu'il est toujours possible de se ramener en temps lineaire à des instances réduites du problème où le graphe réduit est plus petit et la coloration est "propre" (les sommets adjacents ont des couleurs distinctes). Vient alors l'étude du problème d'inondation dans deux cas particuliers qui permettent de localiser plus précisement la difficulté : on montre que le problème sur les arbres est NP-dur dès 3 couleurs (Partie 3), et que le problème sur les cycles se résout en temps polynomial pour un nombre arbitraire de couleurs (Partie 4). Dans la partie 3, il est aussi montré que la version Free-Flood-It (où on choisit la zone à inonder et sa nouvelle couleur) est NP-dure dans le cas des arbres dès 3 couleurs, donc dans le cas d'un graphe quelconque. Dans [5], ce résultat de complexité avait été démontré dans le cas particulier de la grille carré avec 3 couleurs, mais pour 2 couleurs les auteurs indiquaient que la complexité était ouverte. Nous sommes parvenus à montrer que pour deux couleurs et un graphe quelconque, Free-Flood-It est polynomial (Partie 5) : le nombre minimum d'étapes d'inondations pour n'avoir plus qu'une couleur sur le graphe est égal au rayon du graphe initial réduit, qui se calcule en temps polynomial. En corollaire, ce résultat s'applique au cas particulier de la grille carrée de [5], résolvant ainsi leur problème ouvert.

Dans une dernière partie, nous nous concentrons sur le cas de la grille carrée étudiée dans [5] en donnant une caractérisation des graphes réduits des grilles carrées $n \times n$ colorées par un nombre arbitraire de couleurs (Partie 6). Cette caractérisation montre que la topologie de ces instances est très particulière. L'objectif était ici de trouver des propriétés amenant des optimisations dans les algorithmes résolvant le casse-tête pour ces instances.

Une conclusion termine ce rapport en discutant des extensions possibles de ce travail.

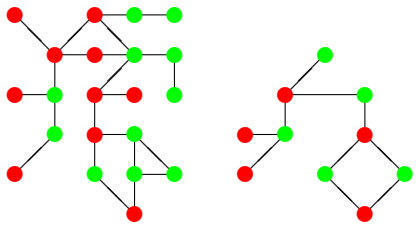
1.2 Définitions et notations

Soit $G = (V, E)$ un graphe non orienté. On se servira du vocabulaire suivant :

- Une *c-coloration* φ de G est une application surjective de V dans un ensemble de couleurs C, où $c = |C|$. La surjectivité de cette application implique que toutes les couleurs sont utilisées.
- Une *c-coloration propre* φ de G est une c-coloration de G vérifiant la propriété suivante : $\forall s_1, s_2 \in V \quad s_1 s_2 \in E \Rightarrow \varphi(s_1) \neq \varphi(s_2)$. Cela signifie que deux sommets adjacents dans G n'ont pas la même couleur.
- G est *c-coloriable* si on peut le munir d'une c-coloration propre.

On suppose que G est muni d'une c-coloration φ.

- Une *zone* de G est un ensemble connexe maximal de sommets de G ayant la même couleur.
- Le *graphe réduit* de G, noté G^{Red} est le graphe (V^{Red}, E^{Red}) où V^{Red} est l'ensemble



(a) G (b) G^{Red}

FIG. 2 – Un graphe muni d'une 2-coloration et son graphe réduit.

des zones de G et E^{Red} est défini de la manière suivante :

$$\forall\ z_1,\ z_2\ \in\ V^{Red},\qquad (z_1z_2 \in\ E^{Red}\ \Leftrightarrow\ (\exists s_1 \in z_1, s_2 \in z_2,\quad s_1s_2 \in E))$$

Deux zones sont donc adjacentes si et seulement s'il existe dans G une arête reliant un élément de chaque zone. La coloration φ^{Red} induite par φ sur G^{Red} est telle que pour une zone $z \in V^{Red}$, $\varphi^{Red}\ (z) = \varphi(s), \forall s \in z$. Si φ est une c-coloration, alors φ^{Red} est une c-coloration propre de G^{Red} (voir Fig 2).

On munit G d'une c-coloration φ. Soit s un sommet de G.

- On appelle *inondation depuis une source* s l'opération qui consiste à choisir une couleur c_0 puis à modifier φ en φ' de telle sorte que si s' est dans la même zone que s, alors $\varphi'(s') = c_0$, et sinon $\varphi'(s') = \varphi(s')$. Ainsi, tous les sommets de la même zone induisent la même inondation. Remarquons que l'opération d'inondation entraîne généralement la modification des zones.
- On dit que G muni de φ est *entièrement inondé* si φ est une 1-coloration.
- On note $N(s) = \{s' \in V | ss' \in E\}$ l'ensemble des voisins de s.
- On appelle c-FLOOD-IT le problème qui prend en entrée un graphe G muni d'une c-coloration φ, et un sommet s_0 qui sera la source de chaque inondation, et qui retourne le nombre minimal d'inondations depuis s_0 nécessaires pour inonder G entièrement.
- On appelle c-FREE-FLOOD-IT la variante de c-FLOOD-IT qui prend en entrée un graphe G muni d'une c-coloration φ, et qui retourne le nombre minimal d'inondations nécessaires pour inonder G entièrement, lorsque le joueur peut choisir à chaque coup la source de l'inondation.
- On appelle *c-plateau* une grille carrée de taille $n \times n$ muni d'une c-coloration des cases. Cela correspond à une configuration initiale du plateau du jeu Flood-It [1].

Remarque : Les problèmes c-FLOOD-IT et c-FREE-FLOOD-IT sont une généralisation à un graphe quelconque de la dynamique des jeux Flood-It [1] et Mad Virus [2], où

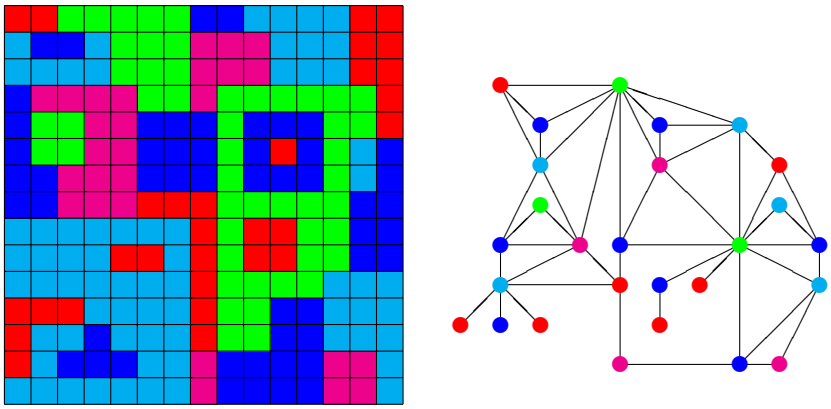


FIG. 3 – Un 5-plateau et son graphe réduit.

l'on identifie les cases du plateau aux sommets du graphe, et où deux sommets sont adjacents si et seulement si les cases correspondantes ont un côté commun. Grâce à cette identification, on pourra parler de zone et de graphe réduit d'un c-plateau (voir Fig 3).

Lors des réductions de la partie 3, on utilisera les problèmes NP-complets nommés *SCS (shortest common supersequence)* et *SCS sans doublon*, dont les définitions sont indiquées ci-dessous :

- Soit s_1 une chaîne de caractère sur un alphabet Σ. Une chaîne de caractère s est dite *superséquence de* s_1 si l'on peut obtenir s_1 à partir de s en supprimant certaines de ses lettres Par exemple, *abcabac* est une superséquence de *acba*.
- *Le problème SCS :* une instance du problème SCS sur un alphabet Σ est la donnée de k chaînes de caractères $s_1, \ldots, s_k$. Il s'agit de trouver la longueur de la plus courte superséquence commune à $s_1, \ldots, s_k$. La version décisionnelle comporte une entrée supplémentaire, à savoir un entier l. Il s'agit alors de savoir s'il existe une superséquence commune à $s_1, \ldots, s_k$ de taille inférieure ou égale à l.
- *Le problème SCS sans doublon :* une instance du problème SCS sans doublon sur un alphabet Σ est la donnée de k chaînes de caractères $s_1, \ldots, s_k$ vérifiant les deux propriétés suivantes : d'une part, pour tout $i \leq k$, il n'existe pas deux lettres consécutives identiques dans s_i. D'autre part, il existe une lettre a de Σ telle qu'aucun s_i, $i \leq k$, ne commence par la lettre a. La version décisionnelle comporte, comme pour le problème SCS, une entrée supplémentaire, à savoir un entier l, et fonctionne de la même manière.

2 Algorithme de calcul du graphe réduit

L'algorithme de l'annexe A calcule le graphe réduit d'un graphe G contenant n sommets et m arêtes, muni d'une c-coloration. Il s'applique en particulier au calcul du graphe

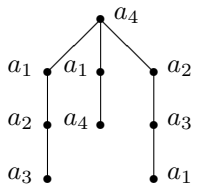


FIG. 4 – Arbre utilisé pour l'instance de SCS sans doublon suivante : $s_1 = a_1a_2a_3$, $s_2 = a_1a_4$ et $s_3 = a_2a_3a_1$.

réduit d'un c-plateau. Il s'agit d'un parcours de graphe qui donne la priorité aux voisins de même couleur. Ainsi, tous les sommets correspondant à la même zone de couleur sont parcourus avant de visiter d'autres sommets. La complexité de cet algorithme est linéaire en le nombre de sommets et d'arêtes, soit $\mathcal{O}(n+m)$.

3 Étude du problème sur des arbres, $c \geq 3$

On montre dans cette partie que le problème d'inondation d'un arbre, pour $c \geq 3$, est NP-difficile, et que la version décisionnelle est NP-complète.

Proposition 1 (cf annexe B et [3, 5])**.** *Le problème SCS sans doublon sur un alphabet Σ de taille $|\Sigma| \geq 3$ est NP-dur, et la version décisionnelle est NP-complète.*

Proposition 2. *Le problème c-FLOOD-IT et c-FREE-FLOOD-IT sur un arbre avec $c \geq 3$ est NP-dur, et la version décisionnelle est NP-complète.*

Démonstration. En s'inspirant de la preuve dans [5], on effectue une réduction à partir du problème SCS sans doublon : on dispose de k chaînes de caractères sans doublon $s_1, \ldots, s_k$ de longueur au plus w sur un alphabet $\Sigma = \{a_1, \ldots, a_r\}$ où $r \geq 3$. On sait qu'il existe une lettre a_{i_0} de Σ qui est différente de la première lettre de chaque s_i.

Σ représente l'ensemble de couleurs que l'on va utiliser. Dans un premier temps, traitons le cas où l'on ne peut inonder qu'à partir de la racine. On crée l'arbre A dont la racine est de couleur a_{i_0}, et qui contient k fils. Le sous-arbre engendré par le i-ème fils est une chaîne de sommets de longueur $|s_i|$, dont le j-ème sommet porte la couleur correspondant à la j-ème lettre de s_i (cf Fig 4).

On note une séquence d'inondations sous la forme de la concaténation successive des couleurs utilisées. On voit aisément qu'une séquence d'inondation correspond exactement à une superséquence pour $s_1, \ldots, s_k$.

On en déduit que le problème c-FLOOD-IT sur un arbre, en inondant depuis de la racine, est NP-difficile pour $c \geq 3$. La version décisionnelle où on veut savoir si on peut tout inonder en moins de l étapes est NP-complète car le problème est dans NP : tout graphe connexe à n sommets peut être entièrement inondé en n étapes, donc seule les

instances avec $l \leq n$ sont intéressantes et ensuite générer une séquence de couleurs de longueur l et tester si elle inonde tout le graphe se fait en temps polynomial.

Prouvons que ce résultat est vrai dans la variant c-Free-Flood-It où l'on peut choisir à chaque coup le sommet depuis lequel on inonde. Pour cela, on construit un nouvel arbre A' à partir de l'arbre A. Soit l la longueur de la plus courte superséquence. Remarquons tout d'abord que $l \leq kw$ (on crée une superséquence en concaténant les k mots de longueur $\leq w$). On obtient A' en copiant $(w+1)$ fois chaque sous-arbre engendré par un fils de la racine. Supposons que l'on dispose d'une séquence de mouvements de longueur l inondant A. En effectuant les mêmes opérations, on peut inonder A' en l coups. Montrons que l'on ne peut pas faire mieux en choisissant d'inonder à partir d'un sommet qui n'est pas la racine. Si l'on choisit un sommet différent de la racine, cela signifie qu'on inonde depuis une branche. Cette inondation ne peut pas affecter une autre branche ; supposons qu'elle permette par miracle d'inonder la branche entière. Il faudrait ainsi au moins $k(w+1)$ coups tels que celui-ci pour inonder toutes les branches, ce qui est supérieur à l.

Ainsi, le problème c-Free-Flood-It dans un arbre, avec $c \geq 3$, est NP-difficile. La version décisionnelle est NP-complète car ici aussi tout graphe peut être entièrement inondé en n étapes et générer une séquence de couples (couleur,sommet) de longueur $l \leq n$ puis tester si elle inonde tout le graphe se fait en temps polynomial. □

4 Étude du problème sur les cycles, c quelconque

Cette partie propose un algorithme polynomial pour résoudre le problème c-Flood-It sur un cycle, sans restriction sur c.

Définitions et notations Soit G un cycle à n sommets, contenant un sommet distingué s_0 et muni d'une c-coloration propre φ. Les définitions suivantes sont illustrées à la figure 5.

- On munit G de deux numérotations : une *numérotation directe*, qui attribue le numéro 0 au sommet s_0, puis qui numérote les sommets en parcourant le cycle dans un sens arbitraire ; une *numérotation indirecte*, qui attribue le numéro 0 au sommet s_0, puis qui numérote les sommets en parcourant le cycle dans le sens inverse au sens précédent. Remarquons que si un sommet porte le numéro a dans la numérotation directe, alors il porte le numéro $n-a$ dans la numérotation indirecte.
- Le couple (a, b) désigne la paire de sommets constituée du sommet numéroté a dans la numérotation directe, et du sommet numéroté b dans la numérotation indirecte. Un couple (a, b) est dit *valide* si $a + b \leq n - 1$ et $\varphi(a) = \varphi(b)$.
- Le couple valide (c, d) est dit *emboîté* dans le couple valide (a, b), et on adopte la notation $(a, b) < (c, d)$, si $a < c$ et $b < d$. Cela définit une relation d'ordre.

Proposition 3. *Le problème c-Flood-It sur un cycle, $c \in \mathbb{N}^*$, se résout en temps $\mathcal{O}(n^2 \log n)$, où n est le nombre de sommets du cycle.*

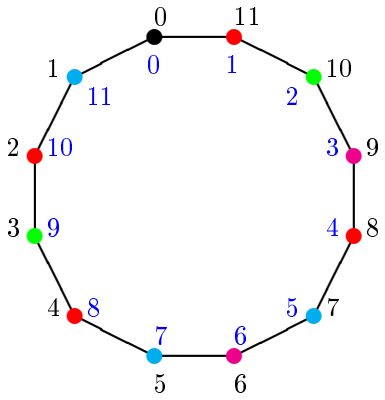


FIG. 5 – Un exemple : Ce cycle réduit est muni d'une numérotation directe (en noir) et d'une numérotation indirecte (en bleu). Les couples valides (3,2) et (2,4) ne sont pas comparables ; Le couple (4,4) est emboîté dans le couple (3,2). Une chaîne maximale de couples emboîtés est (2,1)-(3,2)-(4,4)-(5,5).

Démonstration. Soit G un cycle à n sommets contenant un sommet distingué et muni d'une c-coloration φ. Raisonnons sur G^{Red} , son graphe réduit à n_{Red} sommets contenant également un sommet distingué s_0, muni de φ^{Red} . Notons que l'obtention du graphe réduit se fait en temps linéaire en n, et que ce graphe reste un cycle. Chaque inondation depuis s_0 agrandit la zone contenant s_0 d'un ou deux sommets. Le nombre d'inondations minimal est donc obtenu en maximisant le nombre d'inondations qui agrandissent la zone de deux sommets. Pour cela, nous devons rechercher le nombre maximal n_2 de couples emboîtés. Ainsi, le nombre minimal d'inondations à effectuer pour inonder tout le graphe est $n_{Red} - 1 - n_2$. On obtient les nouvelles numérotations en temps $\mathcal{O}(n_{Red})$, puis l'algorithme de l'annexe C calcule en $\mathcal{O}({n_{Red}}^2)$ les m couples valides, m pouvant atteindre l'ordre de grandeur de $(n_{Red})^2$. Le nombre n_2 est ensuite donné par la hauteur de la plus grande chaîne croissante pour l'ordre d'emboitement défini sur les couples, qui se calcule en $\mathcal{O}(m \log m)$ [8]. Finalement, on obtient n_2 en temps $\mathcal{O}((n_{Red})^2 \log n_{Red})$. Il suffit ensuite de remarquer que n_{Red} est de l'ordre de n pour conclure que le nombre minimal d'inondations nécessaires pour inonder tout le graphe est obtenu en temps $\mathcal{O}(n^2 \log n)$. □

5 Étude du problème 2-FREE-FLOOD-IT

L'étude de la complexité de 2-FREE-FLOOD-IT est un problème ouvert proposé dans [5]. Cette partie montre que, pour un graphe quelconque G, le nombre d'inondations nécessaires pour 2-FREE-FLOOD-IT est le rayon du graphe réduit G^{Red} , qui se calcule en temps polynomial.

Définitions Soit $G = (V, E)$ un graphe. On rappelle la définition des notions suivantes :
- Pour tout sommet $x \in V$, l'*excentricité en* x : $r(x) = \max\{d(x, y) \mid y \in V\}$
- Le *rayon du graphe* : $R(G) = \min\{r(x) \mid x \in V\}$
- Le *centre du graphe* : $C(G) = \{x \in V \mid r(x) = R(G)\}$

Proposition 4. *Soit $G = (V, E)$ un graphe, et $s_0 \in V$. Alors, pour le problème 2-*Flood-It *(*i.e. *on inonde à chaque coup depuis s_0), le nombre minimal de coups pour inonder G est l'excentricité $r_{G^{Red}}(s_0)$ de s_0 dans G^{Red} .*

Démonstration. On munit G d'une 2-coloration φ et G^{Red} de la 2-coloration propre induite φ^{Red} . On appelle 1 la couleur du sommet s_0, et 2 l'autre couleur. Si l'on ne fait que des inondations utiles, c'est-à-dire des inondations qui agrandissent d'au moins un sommet la zone contenant s_0, alors la séquence d'inondations est imposée : il s'agit de la séquence $2121 \dots 21$ (ou $2121 \dots 12$ selon la parité de la longueur), de longueur $r_{G^{Red}}(s_0)$. En effet, au i-ème coup on inonde les sommets situés à distance i de la zone contenant s_0 dans G^{Red} , ce qui mène au résultat. □

Notations On aura besoin d'effectuer, sur des graphes muni d'une 2-coloration propre, une inondation en un point s, puis de recalculer immédiatement le nouveau graphe réduit. L'opération qui correspond à ceci est appelée *contraction* du graphe G en s et consiste en la modification G en $G/s = (V/s, E/s)$ où

$$V/s = V \setminus N(s)$$

et

$$E/s = (E \setminus \{ss' \in E | s' \in V\}) \cup \{s's \mid \exists s_1 \in N(s) \quad s's_1 \in E\}$$

Pour tout chemin γ dans G, on note γ/x le chemin correspondant dans G/x. On remarque que tout chemin dans G/x peut s'écrire γ/x, où γ est un chemin dans G. Ainsi, pour raisonner sur tous les chemins dans G/x (notamment pour minorer une distance entre deux points dans G/x), il suffit de considérer tous les chemins γ de G et d'étudier ce qu'il se passe pour γ/x.

Lemme 1 (prouvé en annexe D)**.** *Soit $c \in C(G)$, $y \in V$ tels que $d(c, y) = R(G)$ et $\gamma = ca_1 \dots a_{r-1}y$ un chemin minimal de c à y (ie $r = R(G)$). Alors*

(i). Il existe $z \neq y$ tel que pour tout chemin minimal μ de c à z, $\gamma \cap \mu = \{c\}$ et $R(G) - 1 \leq d(c, z) \leq R(G)$.

(ii). Plus précisément : ou bien il existe $z \in V$ vérifiant les conditions du (i) ainsi que $d(c, z) = R(G)$, ou bien pour tout z vérifiant les conditions du (i), on a $d(c, z) = R(G) - 1$ et il existe un z_0 parmi ces sommets tel que pour tout μ chemin (non minimal) allant de c à z_0, de longueur $d(c, z_0) + 1$, on a $\gamma \cap \mu = \{c\}$.

Lemme 2 (prouvé en annexe E)**.** *Soient a, b, x trois sommets de G.*

(i). On suppose qu'il n'existe pas de chemin minimal allant de a à b en passant par x. Alors

$$d_{G/x}(a,b) \geq d_G(a,b) - 1$$

et l'égalité est réalisée si et seulement s'il existe un chemin de longueur $d_G(a,b)+1$ allant de a à b en passant par x dans G.

(ii). On ne fait aucune hypothèse sur a, b et x. Alors

$$d_{G/x}(a,b) \geq d_G(a,b) - 2$$

Théorème 1. *Soit $G = (V,E)$ un graphe et $x \in V$. Alors $R(G) - 1 \leq R(G/x) \leq R(G)$*

Démonstration. De manière évidente, pour tous $y, z \in V$, on a $d_{G/x}(y,z) \leq d_G(y,z)$ donc $R(G/x) \leq R(G)$. Montrons que $R(G) - 1 \leq R(G/x)$.

Au moins une des propriétés suivantes est vérifiée :

1. $x \in C(G) \cap C(G/x)$
2. il existe $c \in C(G) \cap C(G/x)$, $c \neq x$, il existe $y \in V$ et γ un chemin minimal de c à y tel que $|\gamma| = R(G)$ et $x \in \gamma$.
3. $C(G/x) \cap C(G) \neq \emptyset$ et pour tout $c \in C(G) \cap C(G/x)$ et pour tout $y \in V$ tel que $d_G(c,y) = R(G)$, il n'existe pas de chemin minimal allant de c à y en passant par x dans G.
4. $C(G/x) \cap C(G) = \emptyset$

Traitons chacun des cas.

1. Comme $x \in C(G)$, alors il existe $y \in V$ et γ un chemin minimal de x à y tel que $|\gamma| = R(G)$. On note $\gamma = xa_1 \dots a_{r-1}y$, où $r = R(G)$. Comme γ est minimal, alors d'une part $\gamma/x = xa_2 \dots a_{r-1}y$ et d'autre part $d_{G/x}(x,y) = |\gamma/x|$. On en déduit que $r_{G/x}(x) \geq d_{G/x}(x,y) = R(G) - 1$ donc, comme $x \in C(G/x)$, on obtient $R(G/x) = r_{G/x}(x) \geq R(G) - 1$.
2. Selon le lemme 1, on a l'un des cas suivants :
   - Il existe $z \in V$ tel que pour tout chemin minimal μ de c à z, $\gamma \cap \mu = \{c\}$ et $d_G(c,z) = R(G)$. Alors x n'appartient à aucun chemin minimal allant de c à z, donc d'après le lemme 2.(i), $d_{G/x}(c,z) \geq d_G(c,z) - 1 = R(G) - 1$. Comme $c \in C(G/x)$, alors $R(G/x) = r_{G/x}(c) \geq d_{G/x}(c,z) \geq R(G) - 1$.
   - Il existe $z \in V$ tel que pour tout chemin minimal μ de c à z, $\gamma \cap \mu = \{c\}$ et $d_G(c,z) = R(G) - 1$. De plus, pour tout chemin (non minimal) μ' allant de c à z et vérifiant $|\mu'| = d_G(c,z) + 1 = R(G)$, on a $\gamma \cap \mu' = \{c\}$. D'après le lemme 2.(i), $d_{G/x}(c,z) \geq R(G) - 1$.
     Donc $R(G/x) = r_{G/x}(c) \geq d_{G/x}(c,z) \geq R(G) - 1$.
3. Soit $c \in C(G) \cap C(G/x)$ et $y \in V$ tel que $d_G(c,y) = R(G)$ (l'existence d'un tel y est assurée par la définition de $R(G)$). Alors, d'après le lemme 2.(i), on a l'inégalité $d_{G/x}(c,y) \geq d_G(c,y) - 1 = R(G) - 1$ donc $R(G) = r_{G/x}(c) \geq d_{G/x}(c,y) \geq R(G) - 1$.
4. Soit $c \in C(G/x)$. Alors $r_G(c) \geq R(G) + 1$. Or selon le lemme 2.(ii)

$$
\begin{array}{lr}
\forall y \in V, & d_G(c,y) - 2 \leq d_{G/x}(c,y) \\
\text{donc} & \max\{d_G(c,y)|y \in V\} - 2 \leq \max\{d_{G/x}(c,y)|y \in V/x\} \\
\text{donc} & R(G) + 1 - 2 \leq R(G/x) \\
\text{donc} & R(G) - 1 \leq R(G/x)
\end{array}
$$

Finalement, dans tous les cas $R(G/x) \geq R(G) - 1$. □

Lemme 3. *Si $c \in C(G)$, alors $R(G/c) = R(G) - 1$.*

Démonstration. Soit $y \in V$. Alors $d_{G/c}(c,y) \leq d_G(c,y) - 1$. En effet, si $\gamma = c\, a_1 \ldots a_{r-1}\, y$ est un chemin minimal allant de c à y, alors $\gamma/c = c\ a_2 \ \ldots \ a_{r-1}\ y$, par conséquent $d_{G/c}(c,y) \leq |\gamma/c| \leq d_G(c,y) - 1$. On a ainsi la relation suivante, par passage au maximum : $\max\{d_{G/c}(c,y)|y \in V\} \leq \max\{d_G(c,y)|y \in V\} - 1$, ce qui équivaut à $r_{G/c}(c) \leq r_G(c) - 1 = R(G) - 1$. On en déduit que $R(G/c) \leq R(G) - 1$. Or, selon le théorème 1, $R(G/c) \geq R(G) - 1$. Finalement, $R(G/c) = R(G) - 1$. □

Théorème 2. *Soit $G = (V, E)$ un graphe 2-coloriable. Alors, pour le problème 2-*Free-Flood-It *(*i.e. *on choisit à chaque coup le sommet depuis lequel on inonde), le nombre minimal de coups pour inonder G est le rayon $R(G)$.*

Démonstration. On voit grâce au lemme 3 que $R(G)$ est un majorant du nombre de coups en considérant la séquence d'inondation obtenue en choisissant un centre de G, et en inondant à chaque coup depuis ce sommet. Montrons par récurrence sur n la propriété suivante :

$\mathcal{H}(n)$ =" si G est un graphe 2-coloriable et $R(G) = n$, alors il faut au moins n coups pour inonder G."

- $\mathcal{H}(0)$: Si G est un graphe, il faut au moins 0 coups pour inonder G.
- $\mathcal{H}(n) \Rightarrow \mathcal{H}(n+1)$: soit G un graphe 2-coloriable de rayon $n+1$. Soit $s = a_1 \ldots a_r$ une séquence d'inondation de G, où a_i représente le sommet choisi pour inonder G à la i e étape. Alors, selon le théorème 1, $R(G) - 1 \leq R(G/a_1) \leq R(G)$. Soit k le premier entier tel que l'inondation par a_k fait baisser le rayon d'une unité (k existe car à la fin de la séquence d'inondation, on a un graphe de rayon 0 : un seul sommet). Alors $|s| = |a_1 \ldots a_k| + |a_{k+1} \ldots a_r| \geq 1 + |a_{k+1} \ldots a_r|$. Or $R(G/a_1/\ldots/a_k) = R(G) - 1$ donc par hypothèse de récurrence, il faut au moins $R(G) - 1$ coups pour inonder $G/a_1/\ldots/a_k$. Donc $|a_{k+1} \ldots a_r| \geq R(G) - 1$. Finalement, $|s| \geq R(G)$: il faut au moins $n + 1$ coups pour inonder G.

□

Corollaire 1. *Soit G un graphe à n sommets et m arêtes, muni d'une 2-coloration φ. Alors le nombre minimal de coups pour inonder G, sachant que l'on peut choisir la source de chaque inondation, est $R(G^{Red})$. Le calcul du graphe réduit se fait en temps linéaire selon l'algorithme 1, et le rayon d'un graphe se calcule aisément en $\mathcal{O}(nm)$ (n parcours en largeur). Par conséquent, le problème 2-*Free-Flood-It *se résoud en temps polynomial en n, le nombre de sommets du graphe.*

6 Caractérisation des graphes réduits

La partie 5 nous montre l'utilité de la notion de graphe réduit. Cette partie consiste en l'établissement d'une caractérisation de ces graphes réduits : il s'agit des graphes obtenus à partir de graphes planaires admettant une carte dont toutes les faces ont 3 ou 4 côtés, en "accrochant" un nombre quelconque d'arbres à des sommets (cf Fig 3). Notons que dans les lemmes 4 et 5, on ne manipule que des graphes sans feuilles. Le traitement de ces arbres intervient plus tard, lors de la caractérisation formelle des graphes réduits dans le théorème 3.

Définitions et notations

- On rappelle qu'un *c-plateau* est une grille carrée de taille $n \times n$ muni d'une c-coloration.
- La case (i, j) d'un plateau désigne la case située à i-ème ligne en partant du haut, et la j-ème colonne en partant de la gauche.
- Un graphe G est dit *planaire* s'il existe une représentation de G dans le plan telle qu'aucune arête n'intersecte une autre. Une telle représentation est appelée *carte* de G.
- Étant donnée une carte d'un graphe planaire, on appelle *face* une partie du plan délimitée par des arêtes, et ne contenant aucune arête. On dit d'une face qu'elle a n *côtés* si elle est délimitée par n arêtes.
- Étant donnée une carte d'une graphe planaire G, le *graphe dual* de G est un graphe planaire dont les sommets correspondent aux faces de G, et deux sommets sont adjacents si et seulement si les faces correspondantes possèdent une arête commune.
- On désignera par *grille* le graphe dual d'un 0-plateau : les arêtes sont les côtés des cases, et les sommets sont les coins des cases.
- Un sommet de la grille est dit *en contact* avec une case du plateau si ce sommet correspond à un coin de la case.

Soit G un graphe et P un 0-plateau.

- Dans notre contexte, on définit une *association* comme une application ψ qui, à chaque case de P, associe un sommet de G ou $\perp$.

Munissons P d'une association ψ.

- On dit qu'une case (i, j) du plateau est *associée* à un sommet s de G si $\psi(i, j) = s$.
- On dit qu'une case (i, j) est *non attribuée* si $\psi(i, j) = \perp$.
- On appelle *zone vide* un ensemble maximal connexe de cases non attribuées.

Lemme 4. *Soit G un graphe planaire, non réduit à un sommet, admettant une carte dont toutes les faces, sauf éventuellement la face externe, ont exactement 4 côtés. Alors G est 2-coloriable.*

Démonstration. Soit s_0 un sommet de G. On colorie s_0 de la couleur 0. On effectue ensuite un parcours en largeur pour colorer de couleur d mod 2 les sommets qui sont à distance d de s_0. Montrons que la 2-coloration est correcte.

On montre par récurrence la propriété suivante :

$\mathcal{H}(n)$="si G_n est le graphe obtenu à partir de G en ne gardant que les sommets x vérifiant $d(s_0, x) \leq n$, alors G_n est muni d'une 2-coloration propre."

- $\mathcal{H}(1)$ est vraie : G_1 est le graphe constitué uniquement du sommet s_0, colorié de la couleur 0, et de ses voisins, colorié de la couleur 1. Il n'existe aucune arête entre les voisins de s_0, car cela formerait une face à 3 côtés. G_1 est donc bien muni d'une 2-coloration propre.
- $\mathcal{H}(n) \Rightarrow \mathcal{H}(n+1)$: Raisonnons par l'absurde et supposons que G_{n+1} n'est pas muni d'une 2-coloration propre. Alors il existe deux sommets s_1 et s_2 adjacents et de même couleur. Par hypothèse de récurrence, G_n est muni d'une 2-coloration propre donc s_1 et s_2 appartiennent à $G_{n+1} \setminus G_n$, c'est-à-dire $d(s_0, s_1) = d(s_0, s_2) = n + 1$. Il existe donc y_1 (resp. y_2) tel que $s_1y_1 \in E$ et $d(s_0, y_1) = n$ (resp. $s_2y_2 \in E$ et $d(s_0, y_2) = n$). S'il existe plusieurs sommets vérifiant les mêmes propriétés que y_1 (resp. y_2), on choisit celui qui minimise l'angle $\widehat{s_2s_1y_1}$ (resp. $\widehat{s_1s_2y_2}$). Ainsi, les arêtes y_1s_1, s_1s_2, s_2y_2 délimitent la même face. Comme les faces ont exactement 4 côtés, alors $y_1y_2 \in E$. C'est absurde car y_1 et y_2 sont de la même couleur et appartiennent à G_n, et G_n est muni d'une 2-coloration propre. Finalement, G_{n+1} est muni d'une 2-coloration propre.

□

Lemme 5 (prouvé en annexe F)**.** *Soit G un graphe planaire sans noeud de degré inférieur ou égal à 1. On suppose qu'il existe une carte telle que toutes les faces de G, sauf éventuellement la face externe, admettent 3 ou 4 côtés. Alors G est le graphe réduit d'un 2-plateau, si toutes les faces admettent 4 côtés, et d'un c-plateau sinon, avec $c = 3$ ou $c = 4$.*

Lemme 6. *Soit P un c-plateau, et G le graphe réduit de P. Alors G est planaire et il existe une carte de G telle que le nombre maximal de côtés d'une face est inférieur ou égal au nombre de côtés des cases de P, c'est-à-dire 4. Cette borne est atteinte.*

Démonstration. On appelle *frontière* une courbe empruntant les arêtes de la grille duale de P, et délimitant deux zones. On appelle F le graphe dont les sommets sont les intersections de frontières, et dont les arêtes sont les portions de frontière ne contenant pas de sommet. On obtient ainsi une carte planaire de F. On remarque alors que G est le dual de F. Ainsi, selon [10], G est planaire et on obtient aisément une carte de G à partir de la carte de F. Elle vérifie la propriété suivante : le nombre de côtés d'une face dans la carte de G est égal au degré du sommet correspondant dans la carte de F. Comme le degré des sommets dans F est limité par le degré des sommets dans la grille duale de P, et que ce degré est égal au nombre de côtés des cases de P, alors le nombre maximal de côtés d'une face dans la carte de G est limité par le nombre de côtés des cases dans P. Cette borne est atteinte de la manière décrite à la figure 6 : le sommet s est de degré 4, donc la face correspondante dans G aura 4 côtés.

□

(a) Plateau (b) Graphe réduit

FIG. 6 – Création d'une face à 4 côtés.

THÉORÈME 3. *Soit G un graphe et G' le graphe obtenu après élagage total de G (c'est-à-dire après suppression récursive des sommets de degré 1). Alors*
(i). G est le graphe réduit d'un c-plateau si et seulement s'il existe une carte planaire de G' dont toutes les faces, sauf éventuellement la face externe, admettent au plus 4 côtés.
(ii).G est le graphe réduit d'un 2-plateau si et seulement s'il existe une carte planaire de G' dont toutes les faces, sauf éventuellement la face externe, admettent exactement 4 côtés.

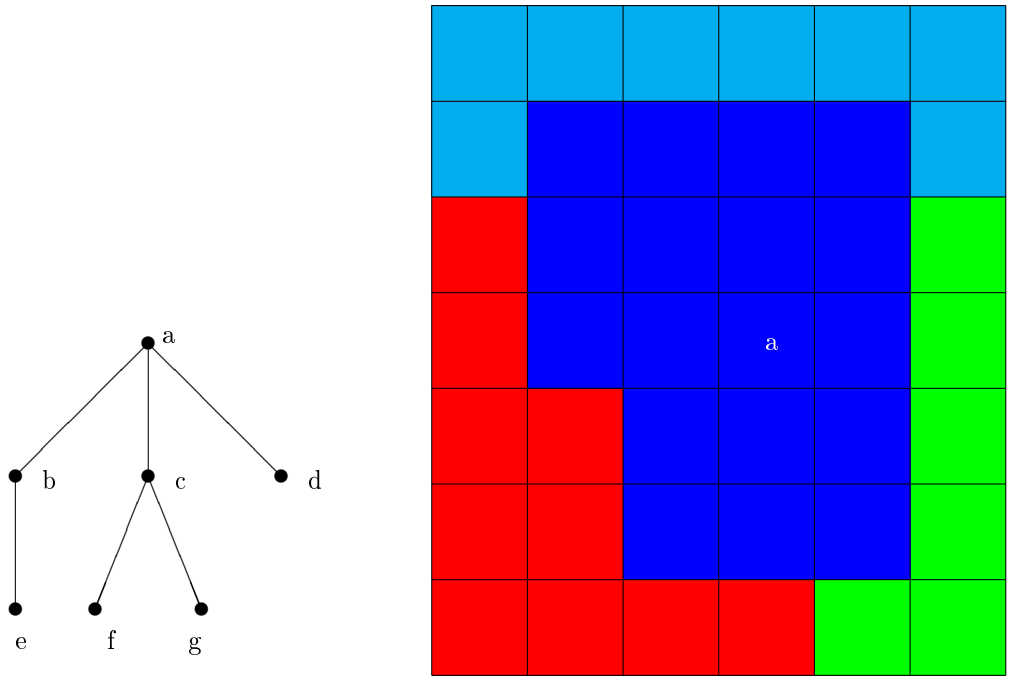
Si G est le graphe réduit d'un c-plateau, alors on obtient une carte particulière de G à partir de celle de G' en ajoutant des arbres de manière adéquate à un certain nombre de sommets.

Démonstration. Sens direct : Construisons la carte de G' en partant du c-plateau P correspondant à G. On numérote chaque zone, dans un ordre arbitraire. Transformons cette configuration et construisons simultanément les arbres qui seront nécessaires pour l'obtention de la carte particulière de G : pour chaque zone, on crée un arbre constitué uniquement d'une racine, portant le numéro de la zone. Ensuite, on répète l'opération suivante autant de fois que possible : pour chaque zone A de couleur c_1, adjacente à une seule autre zone B de couleur c_2, on accroche le sous-arbre enraciné en A comme fils de B, puis on colorie sur le plateau toutes les cases de A de la couleur c_2, ce qui fusionne les zones A et B.

Á la fin de ce processus, chaque zone admet au moins deux zones adjacentes distinctes. Le lemme 6 nous donne l'existence et la construction par dualité d'une carte du graphe G', qui est en fait le graphe réduit de ce c-plateau modifié. Ce lemme nous assure également que G' a uniquement des faces à 3 ou 4 côtés. Si P est un 2-plateau, alors le plateau obtenu après modification est également un 2-plateau. Cela signifie que G', en tant que graphe réduit d'un 2-plateau, est 2-coloriable. Dans ce cas, G' n'a pas de cylces impairs, en particulier pas de faces à 3 côtés. Cela nous donne les conditions que le nombre de côtés des faces de G'.

On obtient ensuite une carte particulière de G en raccrochant les arbres obtenus en début de construction aux sommets correspondants.

Sens réciproque : Le lemme 5 donne un c-plateau correspondant à G' (avec les conditions sur c en fonction des condition sur les faces de G'). Il s'agit ensuite de créer les zones correspondantes aux sommets élagués de G. On procède par opérations successives, dans le sens inverse d'élagage. Cela permet d'utiliser uniquement l'opération d'ajout d'une feuille, que l'on définit ainsi (voir Fig 8) : on choisit une case *intérieure* à la zone père,



(a) Arbre à ajouter (b)

FIG. 7 – Description de la zone a et de l'arbre que l'on souhaite accrocher.

c'est-à-dire une case dont les huit cases l'entourant sont associées à la zone père. Si aucune case de la zone père n'est intérieure, on choisit une case quelconque de la zone père et on subdivise la ligne et la colonne contenant la case en trois selon chaque dimension (ainsi, la colonne devient 3 colonnes et la ligne devient 3 lignes). On est ensuite assurés de pouvoir choisir une case intérieure de la zone père. On associe cette case au sommet fils. On colorie le fils d'une couleur différente de celle de son père (ce qui est toujours possible car on dispose toujours d'au moins deux couleurs). Cette opération correspond exactement à l'ajout d'une feuille à un sommet, donc on peut construire le c-plateau correspondant à G en la répétant successivement.

□

Remarque : Les plongements de graphes dans des plateaux décrits dans les preuves du lemme 5 et du théorème 3 ne sont pas optimaux. Cependant, la taille de la grille obtenue est polynomiale en $(n+m)$, où n est le nombre de sommets de G et m le nombre d'arêtes. Les plongements effectués dans ces preuves ont été choisis pour des raisons de simplicité de rédaction et de compréhension.

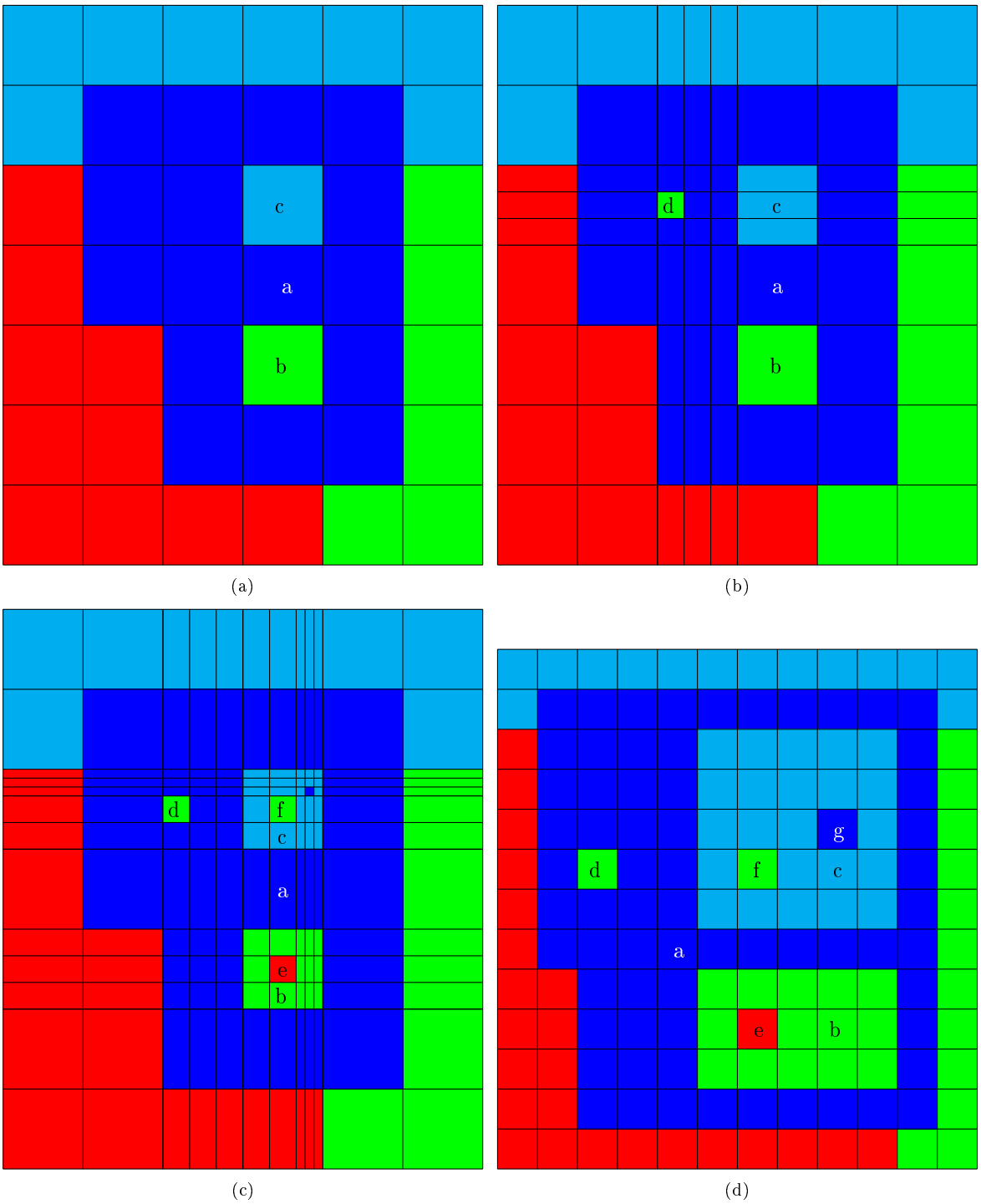


FIG. 8 – Ajout de l'arbre de la figure 7 au sommet a.

7 Conclusion

D'autres pistes de recherche sont encore a explorer, notamment le probleme c-Free-Flood-It sur les cycles, puisque seule la version c-Free-Flood-It a été résolue sur les cycles pour l'instant. En supposant que le graphe de départ et/ou sa coloration sont aléatoires, une approche probabiliste des graphes réduits serait également intéréssante, permettant de connaître le nombre moyen de sommets, le rayon moyen de ces graphes, etc. Aussi, les résultats que j'ai obtenus ont soulevé de nouveaux problèmes. En effet, la caractérisation des graphes réduits laisse espérer un algorithme plus efficace de calcul du rayon dans le cas de ces graphes particuliers. De plus, la taille du plateau issu du plongement d'un graphe réduit peut être optimisée. La plus grande question ouverte restante est bien sûr de préciser la frontière pour laquelle le problème d'inondation devient polynomial. Il est très souvent difficile (arbres des 3 couleurs) mais il existe une classe intéressante où la résolution est polynomiale (cycles). Peut-on étendre cette classe ? Y a-t-il des paramètres qui, s'ils sont contraints, assurent des résolutions polynomiales ?

Références

[1] Jeu flood-it. http ://floodit.appspot.com/.

[2] Jeu mad virus. http ://www.bubblebox.com/play/puzzle/539.htm.

[3] A. Darte. On the complexity of loop fusion. Technical report, Unité mixte de recherche CNRS-INRIA-ENS de Lyon, 1998.

[4] K-J. Räihä et E. Ukkonen. The shortest common supersequence over binary alphabet is np-complete. *Theoretical Computer Science*, 187-198, 1981.

[5] D. Arthur et R. Clifford et M. Jalsenius et A. Montanaro et B. Sach. The complexity of flood filling games. Arxiv, Jan 2010.

[6] D. Arthur et R. Clifford et M. Jalsenius et A. Montanaro et B. Sach. The complexity of flood filling games. In *Proceedings of the Fifth International conference on Fun with Algorithms*, Juin 2010. LNCS 6099, Springer.

[7] K. Appel et W. Haken. Every planar map is four coloriable. *Bulletin of the American Mathematical Society*, 82, 1976.

[8] D.E. Knuth. *The Art of Computer Programming*, volume 3. Addison-Wesley, 1973. Section 5.1.4.

[9] M. Kaufmann U. Fössmeier, G. Kant. 2-visibility drawings of planar graphs. Technical report, Universität Tübigen, Germany et Utrecht University, Netherlands, 1997.

[10] Douglas B. West. *Introduction To Graph Theory.* Prentice Hall, seconde edition, 2001.

Annexes

A Algorithme de calcul du graphe réduit

Algorithme 1: `Calcul du graphe réduit`

Entrées : $G = (V, E)$ graphe muni d'une c-coloration φ
Sorties : Graphe réduit représenté par le tableau des listes d'adjacences `A` et le nombre de sommets `n_red`, muni d'une coloration `Couleur`

début
- $n \leftarrow |V|$;
- $i_z \leftarrow 0$; `/* l'indice de la zone courante */`
- `A` $\leftarrow$ tableau de taille n contenant les listes d'adjacences, initialisé à [ ];
- `Couleur` $\leftarrow$ tableau de taille n contenant la coloration du graphe réduit;
- `I` $\leftarrow$ tableau de taille n contenant pour chaque sommet l'indice de la zone du graphe réduit correspondante;
- **pour chaque** $s \in V$ **faire**
  - `s.visite` $\leftarrow 0$;
- **pour chaque** $s \in V$ *tel que* `s.visite=0` **faire**
  - $i_z \leftarrow i_z + 1$;
  - `Z` $\leftarrow [s]$; `/* Sommets de la zone courante à visiter */`
  - `Couleur`$[i_z]$`=s.couleur`;
  - **tant que** `Z` $\neq [\,]$ **faire**
    - `s` $\leftarrow$ head(`Z`) ;
    - `Z` $\leftarrow$ tail(`Z`) ;
    - `s.visite` $\leftarrow 1$;
    - `I[s]` $\leftarrow i_z$;
    - **pour chaque** s' *voisin de* s **faire**
      - **si** *s'.couleur=s.couleur et s'.visite=0* **alors**
        - `Z` $\leftarrow s' ::$ `Z`;
      - **sinon si** *s'.couleur* $\neq$ *s.couleur et s'.visite=1* **alors**
        - `A`$[i_z]$ $\leftarrow$ `I[s']` $::$ `A`$[i_z]$;
        - `A[I[s']]` $\leftarrow i_z ::$ `A[I[s']]`;
- `n_red` $\leftarrow i_z$;

B Preuve de la propriété 1

Démonstration. Ce résultat a été prouvé dans [3] pour $|\Sigma| = 3$, ce qui implique le résultat pour $|\Sigma| \geq 3$. Cependant la preuve est longue, c'est pourquoi on propose ci-dessous une preuve plus simple dans le cas $|\Sigma| \geq 4$.

Réduisons le problème SCS classique avec un alphabet de taille $|\Sigma| \geq 2$ au problème SCS sans doublon avec $|\Sigma| \geq 4$. Soit une instance de SCS classique, à savoir k chaînes de caractères $s_1, \ldots, s_k$ sur un alphabet $\Sigma = \{a_1, \ldots, a_r\}$ où $r \geq 2$. On définit alors

$\Sigma' = \{b_1, \ldots, b_r\}$ un nouvel alphabet à r lettres. Pour tout $i \in \{1, \ldots, r\}$, soit ${s_i}'$ la chaîne obtenue à partir de s_i en ajoutant le caractère b_j après chaque a_j. Par exemple, $a_4a_3a_6$ devient $a_4b_4a_3b_3a_6b_6$.

Ainsi, le nouvel alphabet est $\Sigma \cup \Sigma'$, de taille $2r \geq 4$. Notons également qu'aucun ${s_i}'$ ne comporte deux lettres consécutives identiques, et que la lettre b_1 n'est jamais utilisée en tant que première lettre d'un mot (c'est également le cas pour toutes les autres lettres b_j, $2 \leq j \leq r$). La donnée des chaînes de caractères ${s_i}'$, $1 \leq i \leq k$ sur l'alphabet $\Sigma \cup \Sigma'$ forme donc une instance du problème SCS sans doublon sur un alphabet de taille supérieure ou égale à 4.

Soit l la taille de la plus petite superséquence pour $s_1, \ldots, s_k$ et s' la plus petite superséquence pour ${s_1}', \ldots, {s_r}'$. Montrons que la taille de s' est exactement $2l$.

Commençons par montrer que $|s'| \leq 2l$. Soit s une superséquence minimale de $s_1, \ldots, s_k$, ce qui implique $|s| = l$. Soit s'' la chaîne obtenue à partir de s en ajoutant le caractère b_j après chaque a_j. Alors s'' est une superséquence de ${s_1}', \ldots, {s_k}'$ et $|s''| = 2l$.

Montrons ensuite que $|s'| \geq 2l$. Soit π_Σ (resp. $\pi_{\Sigma'}$) la projection de $\Sigma \cup \Sigma'$ sur Σ (resp. Σ'). Alors $\pi_\Sigma(s')$ est une superséquence de $\pi_\Sigma({s_1}') = s_1, \ldots, \pi_\Sigma({s_k}') = s_k$ donc $|\pi_\Sigma(s')| \geq l$. De même, $\pi_{\Sigma'}(s')$ est une superséquence de $\pi_{\Sigma'}({s_1}'), \ldots, \pi_{\Sigma'}(s_k)$ et $|s'| = |\pi_\Sigma(s')| + |\pi_{\Sigma'}(s')|$. Par symétrie, $|\pi_{\Sigma'}(s')| \geq l$, donc $|s'| \geq 2l$.

Finalement, comme le problème SCS est NP-dur pour un alphabet d'au moins deux lettres [4], alors le problème SCS sans doublon est NP-dur pour un alphabet d'au moins 4 lettres.

De plus, on vérifie en temps polynomial qu'une chaîne s est une superséquence donc le problème décisonnel est dans NP, donc NP-complet.

□

C Algorithme de calcul des couples valides

Algorithme 2: `Calcul des couples valides`

Entrées : $G = (V, E)$ cycle muni d'une c-coloration propre φ, et dont les sommets sont numérotés selon une numérotation directe
Sorties : `liste_couples` la liste des couples valides
début
    `liste_couples` ← [ ];
    $n \leftarrow |V|$;
    pour *i allant de 1 à* $n-1$ **faire**
        pour *j allant de* $n-i-1$ *à* 1 **faire**
            si $\varphi(i) = \varphi(j)$ **alors**
                `liste_couples` ← (i, j) :: `liste_couples`;

D Preuve du lemme 1

Démonstration. (i). Commençons par montrer qu'il existe $z \in V$ différent de y tel que $R(G)-1 \leq d(c,z) \leq R(G)$. Raisonnons par l'absurde et supposons que pour tout $z \in V$ différent de y, on a $d(c,z) \leq R(G)-2$. Alors $d(a_1,y) = R(G)-1$. De plus, pour tout $z \in V$ différent de y, $d(a_1,z) \leq d(a_1,c)+d(c,z) \leq 1+R(G)-2 = R(G)-1$. Ainsi, $r(a_1) = R(G)-1$. C'est absurde car $R(G) = \min\{r(x)|x \in V\}$.

Montrons alors la propriété (i). Raisonnons à nouveau par l'absurde. Nous venons de montrer l'existence de $z \in V$ vérifiant $R(G)-1 \leq d(c,z) \leq R(G)$. On note $z_1,\ldots,z_n$ les points différents de y vérifiant cette inégalité. On suppose que pour tout $i \in \{1,\ldots,n\}$ il existe $\mu_i = cb_{i,1}\ldots b_{i,r-\varepsilon_i}z_i$ (avec $\varepsilon_i \in \{1;2\}$) un chemin minimal de c à z_i tels que $\gamma \cap \mu_i \neq \{c\}$. Cela signifie que pour tout $i \in \{1,\ldots,n\}$ il existe $k_i,\ j_i \in \{1,\ldots,r\}$ tels que $a_{k_i} = b_{j_i}$. Comme les chemins γ et μ sont minimaux, alors $k_i = j_i$. On a donc

$$\begin{aligned} \forall i \in \{1,\ldots,n\} \qquad d(a_1,z_i) &\leq |a_1\ldots a_{k_i}b_{i,k_i+1}\ldots b_{i,r-\varepsilon_i}z_i| \\ &\leq r-\varepsilon_i \\ &\leq R(G)-1 \end{aligned}$$

De plus, pour tout sommet $x \in V \setminus \{z_1,\ldots,z_n,y\}$

$$\begin{aligned} d(a_1,x) &\leq d(a_1,c)+d(c,x) \\ &\leq 1+R(G)-2 \\ &\leq R(G)-1 \end{aligned}$$

Enfin, $d(a_1,y) = |a_1\ldots a_{r-1}y| = r-1 = R(G)-1$.

Finalement, $r(a_1) = R(G)-1$, ce qui est absurde.

(ii). Soit $\{z_1,\ldots,z_n\}$ l'ensemble des sommets vérifiant les conditions du (i). On suppose que pour tout i, $d(c,z_i) = R(G)-1$. Montrons qu'il existe $i \in \{1,\ldots n\}$, tel que tout chemin μ_i (non minimal) allant de c à z_i de longueur $d(c,z_i)+1$ vérifie $\mu_i \cap \gamma = \{c\}$. On raisonne par l'absurde et on suppose que pour tout i, il existe un chemin μ_i allant de c à z_i, de longueur $d(c,z_i)+1$ tel que $\mu_i \cap \gamma \neq \{c\}$. On note $\mu_i = cb_{i,1}\ldots b_{i,r-1}z_i$. Alors pour tout i, il existe k_i tel que $a_{k_i} = b_{i,k_i}$ ou $a_{k_i} = b_{i,k_i+1}$. Alors

- $d(a_1,y) = R(G)-1$
- pour tout i, on a $d(a_1,z_i) = |a_1\ldots a_{k_i}b_{i,k_i+\varepsilon_i}\ldots b_{i,r-1}z_i|$ où $\varepsilon_i \in \{1;2\}$ donc $d(a_1,z_i) \leq R(G)-1$.
- pour tout x différent des z_i vérifiant l'inégalité $R(G)-1 \leq d(c,x) \leq R(G)$, il existe un chemin minimal $\delta = cd_1\ldots d_{r-\varepsilon}x$ (avec $\varepsilon \in \{1,2\}$) allant de c à x tel que $\delta \cap \gamma \neq \{c\}$. Alors il existe $k_i,\ j_i$ tels que $a_{k_i} = b_{j_i}$. Comme les chemins γ et δ sont minimaux, alors $k_i = j_i$. On a donc $d(a_1,x) \leq |a_1\ldots a_{k_i}b_{i,k_i+1}\ldots b_{i,r-\varepsilon}z_i|$ donc $d(a_1,x) \leq r-\varepsilon \leq R(G)-1$.
- pour tout x tel que $d(c,x) \leq R(G)-2$ (ceci couvre tous les autres cas), on a : $d(a_1,x) \leq d(a_1,c)+d(c,x) \leq R(G)-1$.

Finalement $r(a_1) = R(G)-1 < R(G)$. C'est absurde.

□

E Preuve du lemme 2

Démonstration. (i). On raisonne par l'aburde et on suppose qu'il existe un chemin minimal λ allant de a à b dans G/x, de longueur inférieure ou égale à $d_G(a,b)-2$. Soit μ un chemin dans G tel que $\mu/x = \lambda$. Alors μ va de a à b, ce qui assure que $|\mu| \geq d_G(a,b) \geq |\lambda|+2$. Une telle réduction de longueur signifie que λ passe par x dans G/x. Comme λ est minimal, on peut le décomposer sous la forme $\lambda = \lambda_1 x \lambda_2$ où λ_1 et λ_2 sont minimaux et ne passent pas par x dans G/x. Ainsi, ils ont été inchangés par la fusion de x avec ses voisins, et ils sont également minimaux dans G. On a ainsi : $\mu = \lambda_1 x_1 \dots x_n \lambda_2$ où $\{x_1, \dots, x_n\} \subseteq N(x) \cup \{x\}$ et $n = |\mu| - |\lambda| + 1$. On peut donc créer dans G un chemin $\mu' = \lambda_1 x_1 x x_n \lambda_2$ de longueur $|\lambda|+2 \leq d_G(a,b)$ passant par x. Or μ' va de a à b dans G donc $|\mu'| \geq d_G(a,b)$. Finalement, $|\mu'| = d_G(a,b)$ donc μ' est un chemin minimal allant de a à b dans G, passant par x : c'est absurde.
Cas d'égalité :
- Sens direct : Il existe un chemin minimal λ allant de a à b dans G/x, de longueur $d_G(a,b)-1$. Une construction de μ' de la même manière que dans le cas précédent nous donne un chemin de longueur $d_G(a,b)+1$ allant de a à b en passant par x dans G.
- Sens réciproque :Soit μ' un chemin de longueur $d_G(a,b)+1$ allant de a à b en passant par x dans G. Alors on peut décomposer μ sous la forme $\mu' = \mu_1 x_1 x x_2 \mu_2$, où x_1 et x_2 sont des voisins de x. Alors, $|\mu'/x| \leq |\mu'| - 2 = d_G(a,b) - 1$.

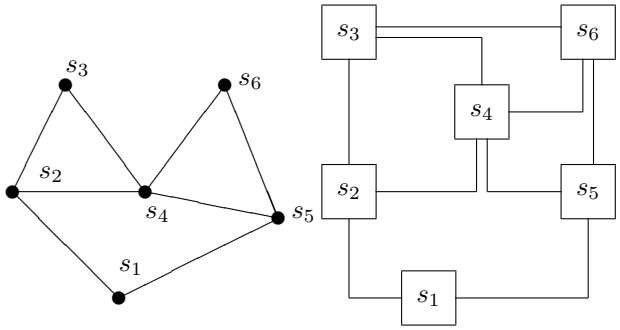
(ii). Soit λ un chemin minimal allant de a à b dans G/x. De la même manière que précédemment, on construit un chemin μ' allant de a à b dans G tel que $|\mu'| = |\lambda|+2$. Donc $d_G(a,b) \leq d_{G/x}(a,b)+2$, c'est-à-dire $d_G(a,b) - 2 \leq d_{G/x}(a,b)$.

□

F Preuve du lemme 5

Démonstration. Construisons un plongement de la carte particulière de G dans un c - plateau, c vérifiant les conditions définies dans l'énoncé du lemme. L'algorithme de construction se décompose en six étapes.

- **Étape 1 : Construction du dessin orthogonal**
  Selon l'algorithme donné dans [9], on construit un dessin orthogonal de G qui conserve la carte. Cela consiste en une représentation planaire de G, où les sommets sont représentés par des rectangles, et les arêtes par une succession de traits horizontaux et verticaux (cf Fig 9). Dans cette construction, chaque arête a au plus un changement de direction. Tous ces objets sont alignés sur une grille, dont le nombre de colonnes et le nombre de lignes sont linéaires en le nombre de sommets. On considère P le plus petit 0-plateau correspondant à cette grille et contenant tous les éléments construits. Dans les étapes suivantes, P va subir des subdivisions, mais on continuera d'appeler P le plateau sur lequel on travaille.



(a) Graphe planaire (b) Dessin orthogonal

FIG. 9 – Dessin orthonal d'un graphe.

- **Étape 2 : Plongement des arêtes sur les cases du plateau**
  Grâce à la tranformation précédente, les sommets sont représentés sur des cases du plateau et les arêtes du graphe sur les arêtes de la grille. Représentons maintenant les arêtes du graphe sur les cases du plateau. On subdivise la grille en deux selon les deux dimensions (une case devient 4 cases). On ajoute également une ligne en haut et une colonne à gauche. Soit une portion d'arête de longueur 1. On étend cette arête de la manière suivante :

  (i). *Arête horizontale* Si l'arête est horizontale, on la représente sur la case située au-dessus. Formellement, si la portion considérée se situe entre les case (i, j) et $(i + 1, j)$, alors on représente l'arête sur la case (i, j).

  (ii). *Arête verticale* Si l'arête est verticale, on la représente sur la case située à gauche. Formellement, si la portion considérée se situe entre les cases (i, j) et $(i, j + 1)$, alors on représente l'arête sur la case (i, j).

  (iii). *Coins* Selon la construction des arêtes décrite dans [9], il n'y a aucun coin d'arête de la forme décrite dans la figure 10.(d), c'est-à-dire une portion d'arête entre les cases $(i+1, j)$ et $(i+1, j+1)$, ainsi qu'une portion d'arête entre les cases $(i, j+1)$ et $(i+1, j+1)$. Ce cas est le seul qui déconnecte une arête par la transformation précédente (cf Fig 10), donc il n'y a pas de problème de déconnexion.

  Ainsi, une arête est un chemin sur les cases du plateau et non plus sur les arêtes de la grille. La notion d'adjacence est définie par un côté commun. Notons que la subdivision de la grille assure que deux arêtes ne peuvent pas devenir adjacentes par la transformation ci-dessus.

- **Étape 3 : Attribution des cases correspondant aux arêtes et aux sommets**
  Soit ψ l'association qui, à toute case de P, associe $\perp$. Modifions ψ. À chaque case comprise dans un rectangle qui représente un sommet s, on associe ce sommet s. Attribuons les cases occupées par les arêtes du graphe. Soient a et b deux sommets reliés par une arête e. Si l est le nombre de cases empruntées par e, on associe les $\lfloor l/2 \rfloor$ premières cases de e à a, et les autres à b (voir Fig 11).

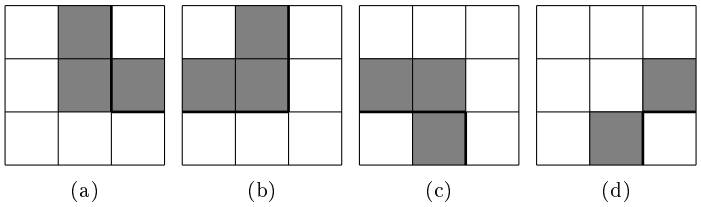


FIG. 10 – Gestion des changements de direction des arêtes. Les traits épais représentent les arêtes avant l'étape 2, et les cases grisées correspondent à ce que l'on obtient après la transformation. Le cas représenté à la figure (d) pose problème pour notre transformation, mais n'arrive jamais.

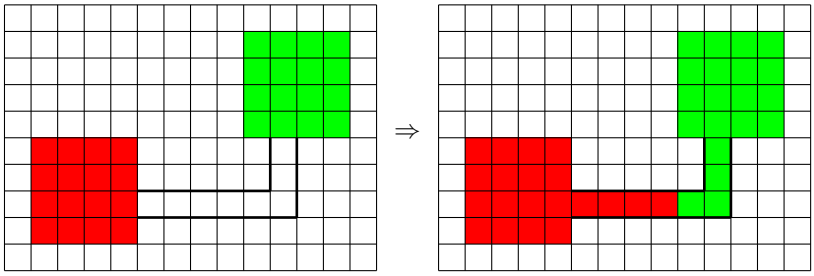


FIG. 11 – Attribution des cases d'une arête.

- **Étape 4 : Attribution des cases situées dans une zone intérieure vide**
  Une zone vide est dite *intérieure* si aucune case de cette zone n'est adjacente au bord du plateau. Un sommet s de G est dit *voisin d'une zone vide* A s'il existe une case associée à s adjacente à une case de la zone vide A. Par construction du dessin orthogonal, et des opérations précédentes, une zone vide correspond à une face interne de G dans la carte. Ainsi, le nombre de voisins d'une zone vide est égal au nombre de côtés de la face correspondante dans la carte. Soit une zone vide A, associons chacune de ses cases à un voisin :

  (i). Si la zone A a 3 voisins :

  On choisit arbitrairement l'un des voisins de A, que l'on appelle *voisin principal*. On associe toutes les cases de cette zone à ce sommet. Le plongement obtenu correspond encore au graphe G, car le voisin principal était déjà adjacent aux deux autres voisins de la zone.

  (ii). Si la zone A a 4 voisins :

  Soient a, b, c, d les voisins de la zone vide, énumérés dans le sens trigonométrique. Soit s_{ab} l'unique sommet de la grille en contact à la fois avec une case de A, une case associée à a et une case associée à b. Si ce sommet est un coin de la zone vide (c'est-à-dire qu'il est en contact avec une seule case de A), on choisit plutôt l'un des deux sommets de la grille adjacents au sommet précédent, qui sont également adjacents à la zone vide (cf le sommet s_{ab} Fig 12.(a)). On définit de même s_{cd},

s_{da}, et s_{bc}. On subdivise la grille en deux selon chaque dimension. On considère les sommets et arêtes de la grille strictement à l'intérieur de la zone vide, plus les quatre sommets cités précedemment. La connexité de cette portion de grille étant assurée par la subdivision, il existe un chemin γ parcourant les arêtes de la grille reliant s_{ab} à s_{cd}, divisant A en deux sous-zones A_1 et A_2. On subdivise à nouveau la grille en deux selon chaque dimension. Cela nous assure tout d'abord que l'on peut trouver deux arêtes e_1 et e_2 consécutives dans γ, de telle sorte que e_1 et e_2 sont toutes les deux soit horizontales, soit verticales. Supposons qu'elles sont toutes les deux horizontales (l'autre cas se traite de même, en échangeant les directions horizontale et verticale). Soit s_m le sommet de la grille commun à e_1 et e_2, de coordonnées (i_m, j_m). La subdivision nous assure de plus que les intérieurs stricts des zones A_1 et A_2 sont connexes. A_1 et A_2 contiennent chacune l'un des deux sommets s_{da} et s_{bc}, et chacune l'un des deux sommets $(i_m - 1, j_m)$ et $(i_m + 1, j_m)$. On appelle s^- (resp. s^+) le sommet de $\{s_{da}, s_{bc}\}$ contenu dans la même sous-zone que $(i_m - 1, j_m)$ (resp. $(i_m + 1, j_m)$). Ainsi, il existe δ_- et δ_+, deux chemins sur les arêtes de la grille, reliant respectivement $(i_m - 1, j_m)$ à s^- et $(i_m + 1, j_m)$ à s^+. On ajoute à δ_- (resp. δ_+) l'arête reliant $(i_m - 1, j_m)$ (resp. $(i_m + 1, j_m)$) à (i_m, j_m).
Cette opération a divisé notre zone vide initiale A en quatre sous-zones vides délimitées par $\gamma, \delta_-, \delta_+$ et le contour de A. Définissons pour chaque sous-zone vide un *voisin principal* et éventuellement des *voisins secondaires*, parmi les sommets de G.
Cas 1 : Une sous-zone n'admet qu'un seul voisin : il s'agit de son voisin principal. Cette sous-zone n'a pas de voisin secondaire.
Cas 2 : Une sous-zone admet deux voisins : par construction des sous-zones, un voisin admet au moins cinq cases adjacentes à la sous-zone (le voisin principal), et l'autre voisin admet exactement quatre cases adjacentes à la sous-zone (le voisin secondaire). Ces deux voisins sont obligatoirement adjacents dans G.
Cas 3 : Une sous-zone admet trois voisins : par construction des sous-zones, un voisin admet au moins cinq cases adjacentes à la sous-zone (le voisin principal), et les deux autres voisins admettent exactement quatre cases adjacentes à la sous-zone (les deux voisins secondaires). Les deux voisins secondaires sont obligatoirement adjacents au voisin principal dans G.
Ainsi, on associe chaque case d'une sous-zone à son voisin principal. La construction de γ, δ_- et δ_+, en particulier la formation d'un "coin", assure que deux sommets non adjacents dans G n'admettent pas de cases associées adjacentes (voir Fig. 12).
Pour éviter d'avoir un nombre exponentiel de subdivisions, on traite toutes les zones vides en parallèle. Ainsi, on trace tous les chemins γ avant d'effectuer la seconde subdivision. Cette étape de l'algorithme ne requiert donc que deux subdivisions.

- **Étape 5 : Ajustement des dimensions du plateau et attribution des cases restantes**
  Il s'agit maintenant d'ajuster les dimensions du plateau pour obtenir le même nombre de lignes et de colonnes. Soient *nb_lignes* et *nb_colonnes* le nombre respectif de lignes et de colonnes de P à cette étape de la construction. On pose

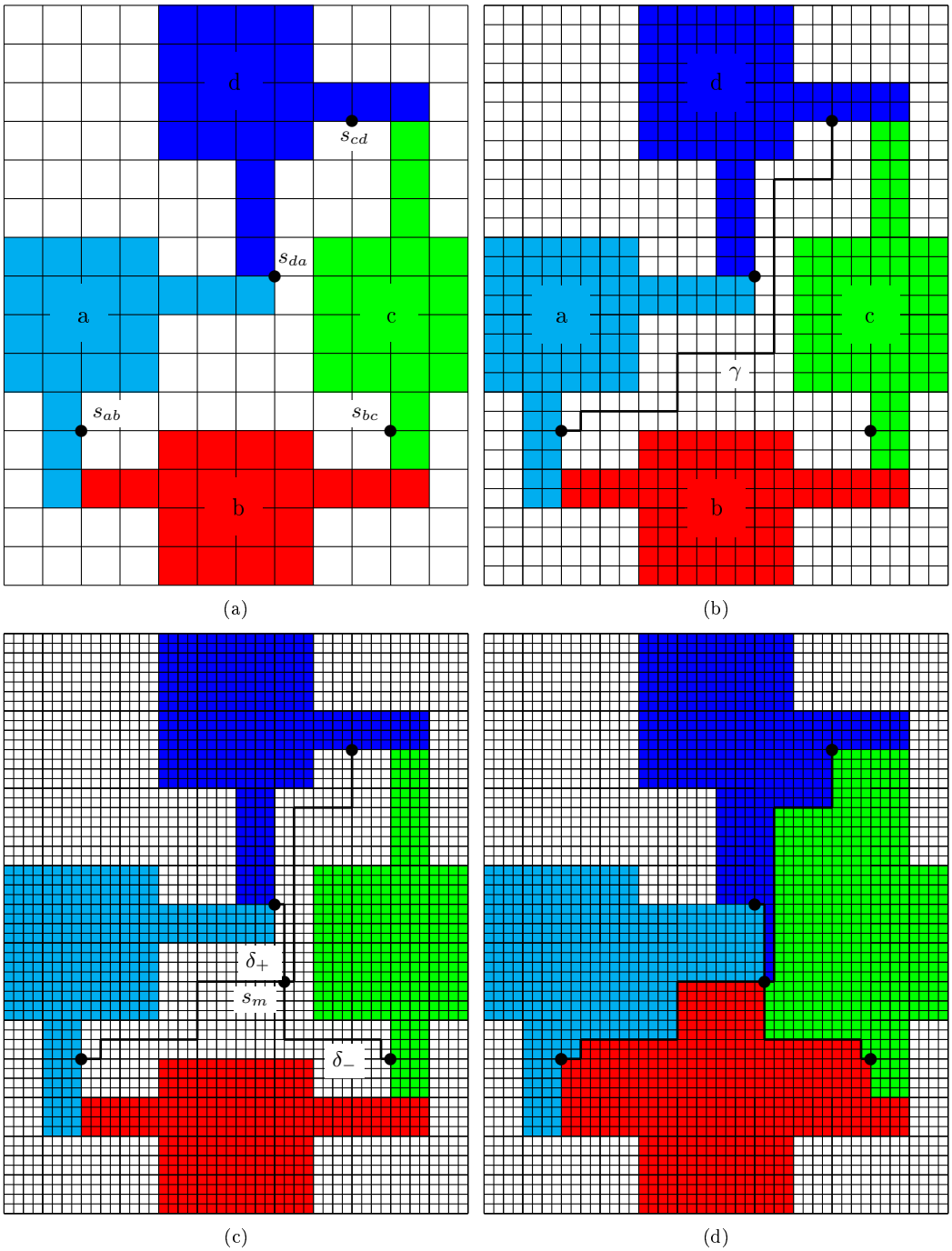
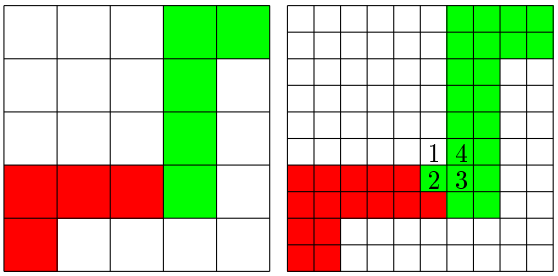


FIG. 12 – Différentes étapes pour l'attribution des cases dans les zones vides.



(a) Configuration initiale (b) Configuration après transformation

FIG. 13 – Transformation dans le cas où la délimitation entre deux sommets est dans un coin.

$l = \max(nb\_lignes, nb\_colonnes)$. On ajoute, si besoin est, des colonnes à droite et des lignes en bas pour que P contienne l lignes et l colonnes. S'il existe au moins un sommet de la grille en contact avec trois cases associées à trois sommets différents, et avec une case non attribuée, alors on subdivise la grille en deux selon chaque dimension et pour chaque sommet s de la grille vérifiant cette propriété, on effectue l'opération suivante, sachant que le sommet s vérifie toujours la même propriété après la subdivision : on appelle la case non attribuée la case 1, puis on numérote les quatre cases adjacentes à ce sommet dans le sens trigonométrique. On change l'attribution de la case 2 pour l'associer au même sommet que la case 3 (voir Fig 13). La subdivision est utile pour nous assurer que le sommet précédemment associé à la case 2 ne perd aucun voisin par cette transformation.
Ainsi, on est assuré qu'il n'existe plus de sommet vérifiant les mêmes propriétés que s.
On définit une notion de distance sur les cases du plateau : si x et y sont deux cases de coordonnées respectives (i_x, j_x) et (i_y, j_y), alors $d(x, y) = \max(|i_x - i_y|, |j_x - j_y|)$.
Le plateau est maintenant constitué d'une portion B contenant les cases attribuées, et de zones vides *extérieures*, à savoir des zones vides délimitées par les bords du plateau et les contours de B. Soit R le nombre maximal de voisin qu'une zone vide admet. On subdivise la grille en $2R$ selon chaque dimension. On appelle *macrocase* un ensemble de cases correspondant à exactement une case du plateau avant subdivision. Comblons une zone vide A ayant r voisins, $r \leq R$ numérotés $s_1, \ldots, s_r$ dans le sens trigonométrique : pour chaque voisin s_i, $2 \leq i \leq r-1$, pris dans l'ordre de numérotation, on effectue l'opération suivante : on choisit une case non attribuée adjacente à une case associée à s_i. On trace alors depuis cette case un chemin parcourant les cases à distance 1 des cases attribuées dans le sens anti-trigonométrique, jusqu'à rencontrer une case située au bord du plateau. En dernier lieu, on associe toutes les cases non attribuées de A, au sommet s_r. (voir Fig 14)
La subdivision nous assure de pouvoir faire passer tous ces chemins dans une couche d'épaisseur 1 de macrocases bordant les macrocases attribuées. En effet, cette couche

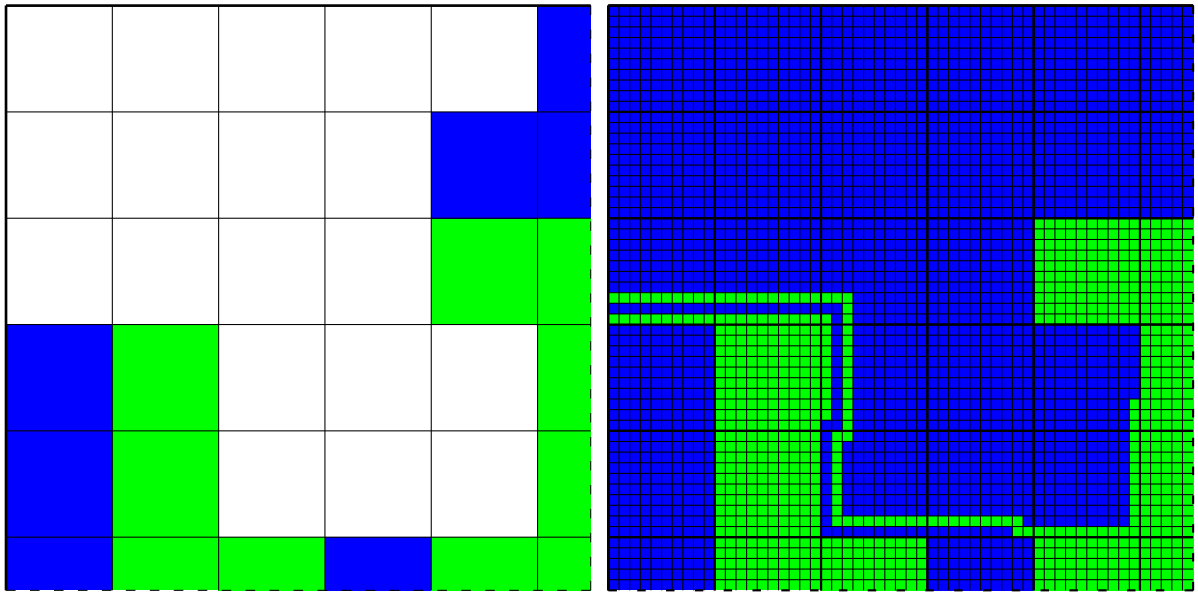


FIG. 14 – Attribution des cases d'une zone vide extérieure.

de macrocases existe obligatoirement par définition de la zone. De plus, la figure 15 illustre les différentes configurations de macrocases consécutives constituant ce chemin, et la manière dont on peut faire passer un nombre suffisant de chemins sur les cases dans le cas correspondant.

- **Étape 6 : Coloration**
  Colorions le graphe. Si toutes les faces de G (sauf éventuellement la face externe) admettent 4 côtés, alors on peut munir G d'une 2-coloration propre, selon le lemme 4. Sinon, on munit G d'une c-coloration propre, avec c minimal donc $c \leq 4$ car G est planaire (cf [7]). Soit φ la coloration propre ainsi obtenue. On définit une coloration

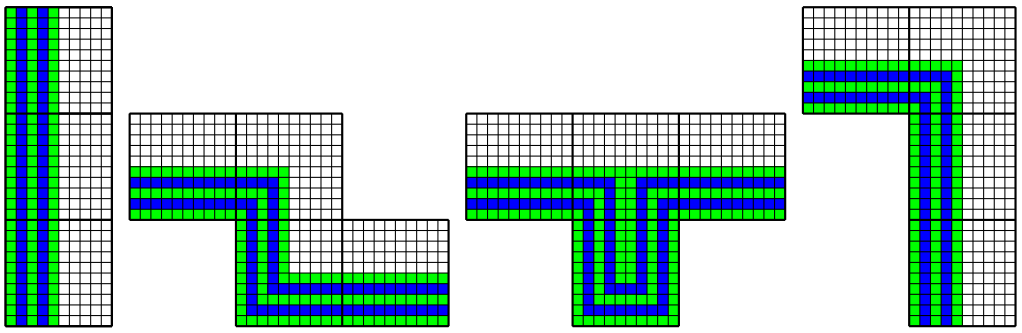


FIG. 15 – Toutes les configurations locales de macrocases, à symétries près ; on vérifie que l'on peut faire passer R chemins sur les cases dans une macrocase subdivisée en $2R$. Ici, $R = 5$.

φ' sur P, utilisant exactement le même nombre de couleurs que φ, de la manière suivante : pour chaque case (i, j), de P, on pose $\varphi'(i, j) = \varphi(\psi(i, j))$, c'est-à-dire qu'une case prend la même couleur que le sommet auquel elle est associée.

En conclusion, cette construction nous donne un plongement du graphe G dans un c-plateau, c vérifiant les propriétés requises selon le nombre de côtés des faces dans la carte de G.

□

Remarque : Si le graphe G est déjà muni d'une c-coloration propre φ on utilise le même algorithme que dans la preuve du lemme précédent, à l'exception de la dernière étape : on garde la coloration φ sur G au lieu de chercher une coloration propre avec un nombre de couleurs minimal.