

(Méta)-noyaux constructifs et linéaires dans les graphes peu denses

Valentin Garnero

▶ **To cite this version:**

Valentin Garnero. (Méta)-noyaux constructifs et linéaires dans les graphes peu denses. Complexité [cs.CC]. Univ. Montpellier, 2016. Français. NNT : . tel-01376809

HAL Id: tel-01376809
https://inria.hal.science/tel-01376809

Submitted on 5 Oct 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

(Méta)-noyaux constructifs

et linéaires dans les graphes peu denses

V. Garnero

19 juin 2016

Table des matières

1 Introduction **5**
- 1.1 Prologue . . . 6
- 1.2 Quelques notions de graphes . . . 14
  - 1.2.1 Éléments de graphes . . . 14
  - 1.2.2 Classes de graphes peu denses . . . 20
  - 1.2.3 Largeur arborescente . . . 26
- 1.3 Quelques bases d'algorithmie . . . 29
  - 1.3.1 Problèmes paramétrés . . . 29
  - 1.3.2 Complexité classique . . . 33
  - 1.3.3 Extraction de noyaux . . . 36
  - 1.3.4 Complexité paramétrée . . . 39
- 1.4 Contexte scientifique . . . 42
  - 1.4.1 Résultats antérieurs . . . 42
  - 1.4.2 Travaux de thèse . . . 44

2 Noyaux *ad hoc* **47**
- 2.1 Méthode de décomposition en regions . . . 49
- 2.2 Noyau pour Domination Rouge-Bleu . . . 63
  - 2.2.1 Réduction de Domination Rouge-Bleu . . . 64
  - 2.2.2 Analyse de la taille du noyau . . . 74
- 2.3 Noyau pour Domination Totale . . . 88
  - 2.3.1 Réduction de Domination Totale . . . 90
  - 2.3.2 Analyse de la taille du noyau . . . 103
- 2.4 Discussion . . . 111

3 Méta-noyaux explicites **115**
- 3.1 Règle de remplacement de protrusions . . . 118
  - 3.1.1 Définitions préliminaires . . . 119
  - 3.1.2 Encodeur de programmation dynamique . . . 124
  - 3.1.3 Relation de $\mathcal{E}$-équivalence . . . 127
  - 3.1.4 Remplacement explicite et générique . . . 135

3.2 Outils pour les applications 140
3.2.1 Théorème de décomposition en protrusions 140
3.2.2 Choix de la fonction d'optimisation 144
3.3 Application à r-Domination 147
3.3.1 Décomposition en protrusions d'une instance 148
3.3.2 Encodeur pour r-Domination 148
3.3.3 Noyau pour r-Domination 153
3.4 Application à r-Indépendance 156
3.4.1 Décomposition en protrusions d'une instance 157
3.4.2 Encodeur pour r-Indépendance 157
3.4.3 Noyau pour r-Indépendance 162
3.5 Application à $\mathcal{F}$-Paquetage 164
3.5.1 Décomposition en protrusions d'une instance 165
3.5.2 Encodeur pour $\mathcal{F}$-Paquetage 168
3.5.3 Noyau pour $\mathcal{F}$-Paquetage 175
3.6 Discussion 176

Chapitre 1

Introduction



FIGURE 1.1 – Nèfle [Wikipédia]. La nèfle est le fruit du néflier (*Eriobotrya japonica*). La nèfle est un fruit charnu de forme ovoïde, de 3 à 7 cm de long et de couleur orangé à maturité. La peau est légèrement duvetée, un peu épaisse. La chair est juteuse et légèrement acidulée. La nèfle contient des noyaux (des pépins du point de vu botanique), au nombre de 4 à 5, parfois moins du fait de l'avortement de certains ovules.

Considérons un fruit, épluchons-le et jetons la peau, elle ne nous sert pas ; retirons-en la chair, nous pouvons la jeter aussi, elle ne nous est pas utile ; car tout ce dont nous avons besoin c'est du noyau, en plantant celui-ci nous obtiendrons un arbre lequel produira de nouveaux fruits, entiers. Cela est possible parce que le noyau à lui seul contient toute l'information génétique pour produire un arbre et des fruits. Le procédé algorithmique appelé extraction de noyau est basé sur la même idée, il s'agit d'un calcul préliminaire visant à réduire les données d'un problème à leur plus simple expression tout en conservant suffisamment d'information pour reconstituer une solution.

Pour mieux comprendre la notion algorithmique de noyau, il convient d'abord de faire quelques rappels sur l'algorithmie.

1.1 Prologue

Un algorithme est une séquence d'instructions qui a pour but de répondre à un problème donné. L'algorithmie est la science qui s'intéresse à la conception des algorithmes et à l'étude de leur complexité, c'est-à-dire au temps qui leur est nécessaire pour répondre au problème. Dans cette section introductive nous tenterons de donner une intuition de ces trois notions : algorithme, problème, et complexité. Nous n'aborderons les définitions formelles que dans les sections 1.2 et 1.3.

Algorithme

Un algorithme est, comme nous l'avons dit, une séquence d'instructions qui vise à résoudre un problème particulier. En d'autres termes, c'est une recette qui spécifie quels calculs effectuer pour répondre à un problème qui est une inconnue pour le concepteur de l'algorithme et qui sera ensuite instanciée par l'utilisateur. Il peut être mis en parallèle avec la notion mathématique de fonction, la variable d'une fonction correspondant à l'entrée d'un algorithme. Cependant alors que la fonction est une sorte de boite noire qui étant donnée une certaine valeur de la variable renvoie un résultat, l'algorithme ce doit d'indiquer comment, pour une certaine instanciation de l'entrée, il est possible de calculer la réponse. Dans une certaine mesure, un informaticien pourrait considérer une fonction (calculable) comme la classe d'équivalence de tous les algorithmes qui, pour une même entrée, renvoient la même réponse ; possiblement par des calculs différents (et donc plus ou moins efficaces). Par exemple un polynôme peut s'écrire indifféremment sous sa forme développée $P(X) = X^2 + 2X + 1$ ou sous sa forme factorisée $P(X) = (X+1)^2$. Il s'agira toujours de la même fonction, pourtant du point de vue du calcul, il s'agit de deux séquences d'opérations bien distinctes.

Le plus célèbre des algorithmes est l'anthyphérèse d'Euclide, qui calcule le plus grand commun diviseur de deux entiers. Cependant, l'algorithmie ne devient une science à part entière qu'au XX^{e} siècle : le 10^{e} des 23 problèmes de Hilbert était de trouver une méthode pour déterminer l'existence de solutions à une équation diophantienne. Une telle méthode (c'est-à-dire, un tel algorithme) n'existe pas ; pour en obtenir la preuve, il a fallu définir rigoureusement ce qu'est une méthode. En effet, il est toujours plus facile d'exhiber un exemple d'une chose, que de démontrer l'inexistence de cette chose ; pour dire qu'une chose n'existe pas, il faut d'abord définir quelle forme elle peut prendre. En bref, pour répondre au problème de Hilbert, il a fallu défricher une nouvelle branche des mathématiques. Il existe donc des modèles de calcul, tels que la machine à computer de Turing ou le λ-calcul de Church,

qui formalisent la notion d'algorithme. Cependant, le sujet de notre thèse ne nécessite pas de rentrer dans un tel niveau de technicité. Nous nous satisferons très largement de la définition informelle : un algorithme est une sequence d'instructions dépendant d'une entrée.

Le corps d'un algorithme est composé d'instructions élémentaires (comme l'addition si l'entrée est un nombre, ou l'ajout de sommet si c'est un graphe) et d'instructions particulières qui permettent d'automatiser le calcul : l'instruction conditionnelle (**si** *une condition est vérifiée* **alors** faire l'instruction), la boucle conditionnelle (**tant que** *une condition n'est pas vérifiée* **faire** l'instruction à répéter), la boucle indexée (**pour chaque** *élément d'un ensemble* **faire** l'instruction à répéter), l'appel de fonction (pour l'appel récursif, par exemple). Ces instructions correspondent aux expressions utilisées dans les langages de programmation. Par hypothèse, les instructions élémentaires peuvent être exécutées en une unité de temps ; elles sont ensuite répétées un certain nombre de fois grâce aux boucles. Se pose alors la question du temps d'exécution d'un algorithme.

Problème

Il est difficile de décrire ce qu'est un problème sans avoir recours à une multitude d'exemples, dans l'espoir d'illustrer toutes les catégories de ce que nous souhaiterions appeler un problème. Une définition formelle passerait certainement par un usage abondant de logique, par la théorie des langages, ou par un modèle de calcul, c'est-à-dire une formalisation d'algorithme ; ni l'une ni l'autre de ces approches ne nous semblent satisfaisantes pour donner une intuition, et une fois encore, nous n'en avons pas besoin dans cette thèse. Disons, plus simplement, qu'un problème est constitué d'une question et d'une instance. La question est, ou du moins se base sur, une propriété mathématique. L'instance décrit la structure, la forme des objets mathématiques qui composent la donnée du problème ; c'est ce qui correspond à l'entrée de l'algorithme. Par exemple un problème pourrait être « Étant donnés deux entiers x, y, quel entier z vérifie $x^2 + y^2 = z^2$? », avec comme question « Quel entier z vérifie $x^2 + y^2 = z^2$? » et comme instance la paire d'entiers x, y.

Tout comme l'algorithme, le problème vise à une certaine généralité en cela que l'instance est une inconnue du problème et que nous souhaitons résoudre le problème sur toutes les instanciations valides (autrement dit, sans connaissance *a priori* de l'instance). Le problème englobe en son sein un ensemble de questions similaires. Par exemple, la question « Les entiers 3 et 5 sont ils premiers entre eux ? » (sans instance) n'est pas à proprement parler un problème ; par contre la question « Étant donné un entier x, 3 et x

sont ils premiers entre eux ? » (avec pour instance l'entier x) est un problème ; plus général encore « Étant donnés deux entiers x, y, x et y sont ils premiers entre eux ? » (avec pour instance la paire x, y).

Peut-être la notion de problème se comprend-elle mieux en se penchant sur la résolution de celui-ci. Un problème définit un univers de solutions potentielles (c'est-à-dire, un ensemble d'objets mathématiques ayant une même structure) ; le résoudre consiste à trouver, dans cet univers, une solution effective (parfois appelée un certificat) qui dépend de l'instance.

Afin de mieux appréhender le concept de problème, nous signalons quelques catégories identifiables par la forme de leur question (et qui sont particulièrement étudiées). Nous les illustrons avec la question des diviseurs communs de deux entiers x, y (l'instance).

— Les problèmes de décision sont des problèmes pour lesquels la réponse attendue est OUI ou NON. Par exemple « Existe-t-il un entier z tel que z divise x et y ? ».

— Les problèmes d'optimisation sont des problèmes pour lesquels la réponse attendue est une valeur correspondant à la mesure d'une solution optimum (minimum ou maximum). Par exemple « Quel est le plus grand entier z tel que z divise x et y ? ».

— Les problèmes de dénombrement sont des problèmes pour lesquels la réponse attendue est un entier correspondant au nombre de solutions. Par exemple « Combien y a-t-il d'entiers z tel que z divise x et y ? ».

Bien évidemment, cette liste n'est pas exhaustive, nous pourrions par exemple considérer les versions computationnelles des trois problèmes précédents, dans lesquelles il faut exhiber le ou les entiers solutions.

Complexité

La complexité d'un algorithme (et par extension du problème qu'il résout) est le nombre d'instructions élémentaires que celui-ci devra exécuter pour effectuer le calcul ; autrement dit, le temps nécessaire avant de renvoyer le résultat (il existe aussi une complexité en espace mémoire, que nous n'utiliserons pas dans cette thèse). Bien sûr, le temps d'exécution dépend de l'entrée : plus l'entrée sera de grande taille, plus l'algorithme prendra du temps (ne serait-ce que pour lire l'entrée). La complexité d'un algorithme sera donc une fonction, dépendant de la taille de l'entrée. Puisque ce sont des fonctions, les complexités sont comparées asymptotiquement (c'est-à-dire pour une taille d'entrée assez grande). De plus, il n'est pas raisonnable de vouloir

décrire exactement la complexité d'un algorithme, le plus souvent, seule une domination asymptotique du pire cas est donnée.

La complexité permet ainsi de classer les problèmes avec une certaine hiérarchie. Un problème est d'autant plus difficile que sa fonction de complexité est asymptotiquement dominante. À une famille de fonctions correspond une classe de problèmes. Par exemple, en ce qui nous concerne, les problèmes polynomiaux (c'est-à-dire, admettant un algorithme dont la complexité est un polynôme) forment une classe de problèmes que nous considérons comme faciles ; cette classe est notée P. Bien sûr le degré du polynôme peut être grand (ce qui implique un algorithme relativement lent), mais asymptotiquement, l'algorithme polynomial sera plus efficace qu'une énumération exhaustive de toutes les possibilités (en général de complexité exponentielle). Tous les problèmes n'admettent pas un algorithme polynomial : il est prouvé que certains problèmes nécessitent un nombre exponentiel d'étapes pour être résolus. Il s'agit pour nous des problèmes trop difficiles. Nous étudions donc des problèmes de difficulté intermédiaire, qui sont les problèmes dont une solution peut être vérifiée en temps polynomial ; la classe de ces problèmes est notée NP. Il est assez facile de voir qu'un problème qui peut être résolu en temps polynomial peut être vérifié en temps polynomial. Par exemple, il est plus facile de répondre à la question « L'entier z divise-t-il x ? » avec comme instance z, x, que de répondre à « Existe-t-il un entier z tel que z divise x ? » avec comme instance x. Les problèmes vérifiables en temps polynomiale sont donc potentiellement plus difficiles que les problèmes résolubles. Reste la question de savoir s'ils sont strictement plus dur. Il s'agit du problème $\mathsf{P} \stackrel{?}{=} \mathsf{NP}$, connu comme une des principales questions ouvertes de la théorie de la complexité. Néanmoins, la conjecture $\mathsf{P} \neq \mathsf{NP}$ (qui semble plus raisonnable) sert de fondement à la plupart des travaux en algorithmie. En d'autres termes, s'il est prouvé qu'un problème est plus dur que tous les problèmes vérifiables en temps polynomial, alors il y a peu d'espoir qu'il admette un algorithme polynomial.

Partant de ce postulat, de nombreuses branches de l'algorithmie ont développé des méthodes pour attaquer les problèmes difficiles (mais dans NP) en temps raisonnable. Il a d'abord été question d'heuristiques, qui sont des algorithmes efficaces en pratique, sans garantie théorique. En termes informels, une heuristique renvoie une solution exacte ou presque, en un temps polynomial ou presque, dans tous les cas ou presque. Chacune de ces relaxations a ouvert la voie à une branche théorique (avec toutes les intersections possibles).

— Les algorithmes approximés, qui s'attaquent aux problèmes d'optimisation, cherchent à renvoyer (en temps polynomial) un résultat approché

avec une garantie sur le rapport entre le résultat et la solution optimale.

— Les algorithmes probabilistes cherchent à renvoyer (en temps polynomial) une solution presque toujours correcte.

— Les algorithmes faiblement exponentiels cherchent à renvoyer une solution (exacte) en un temps exponentiel plus court que l'énumération exhaustive.

— Les algorithmes paramétrés cherchent à renvoyer une solution (exacte) en un temps polynomial en fonction de la taille de l'instance et exponentiel en fonction d'un paramètre.

(Méta)-noyau

Considérons plus en détail l'algorithmie paramétrée. Cette branche part du postulat que la taille de l'instance n'est pas toujours le bon indicateur pour mesurer la complexité du problème. Elle considère donc une seconde mesure de l'instance, appelée paramètre, qui est la cause de l'explosion combinatoire (c'est-à-dire de la complexité exponentielle). La complexité d'un problème est analysée en fonction de la taille et du paramètre.

Si la taille de l'instance n'est pas cause de complexité, autrement dit, si un accroissement de l'instance n'implique pas une croissance de la difficulté, c'est bien que l'instance contient de l'information superflue (non nécessaire à la résolution du problème). Le principe de l'extraction de noyaux consiste à retirer cette information inutile. Cette idée existait déjà dans le monde des heuristiques, il s'agissait alors de faire un pré-calcul facile, afin de simplifier les données du problème, avant de s'attaquer à la partie difficile. L'extraction de noyaux est très exactement un pré-calcul, mais avec deux garanties théoriques : celle que l'ensemble des solutions reste identique, et celle que l'instance a été suffisamment élaguée, compressée.

De plus, comme l'annonce son titre, cette thèse s'intéresse à des méta-algorithmes d'extraction de noyau. Nous l'avons vu, un problème généralise une question en laissant une partie, l'instance, inconnue. Pour résoudre un problème il faut utiliser un algorithme qui généralise le calcul, y laissant une variable, l'entrée, indéterminée. L'objectif de cela est de proposer une méthode qui permet d'automatiser une famille de calculs similaires. De même que, dans un problème, l'instance est une inconnue, une variable ; de même nous pourrions, pour plus de généralité encore, laisser indéterminée la question. La déscription du problème ferait alors partie de l'entrée de l'algorithme. Bien sûr nous ne pourrions prétendre faire de la question une inconnue totale. Si nous n'imposons pas certaines contraintes sur le problème de l'entrée,

notre algorithme devient une sorte de machine universelle de Turing. Et nous nous heurterions à la théorie de la calculabilité qui dit impossible l'analyse de complexité sur une telle machine. Mais nous pourrions vouloir considérer en un seul bloc tous les problèmes dont la question peut être exprimée sous une certaine forme. Un méta-problème est une famille de problèmes ayant une description commune. Un méta-algorithme est un algorithme qui, en une fois, décrit la méthode pour résoudre toute une famille de problème.

Un méta-noyau est donc un algorithme qui, étant donné un problème et une instance, est capable de réduire l'instance, de sorte que instance initiale et instance réduite soient équivalentes par rapport au problème donné. Une autre approche consiste à voir un méta-noyau comme un algorithme qui, étant donnée la description d'un problème, renvoie une extraction de noyau adapté à ce problème.

Dans notre thèse nous nous intéresserons uniquement à certains (méta)-problèmes de décision dont l'instance contient un graphe.

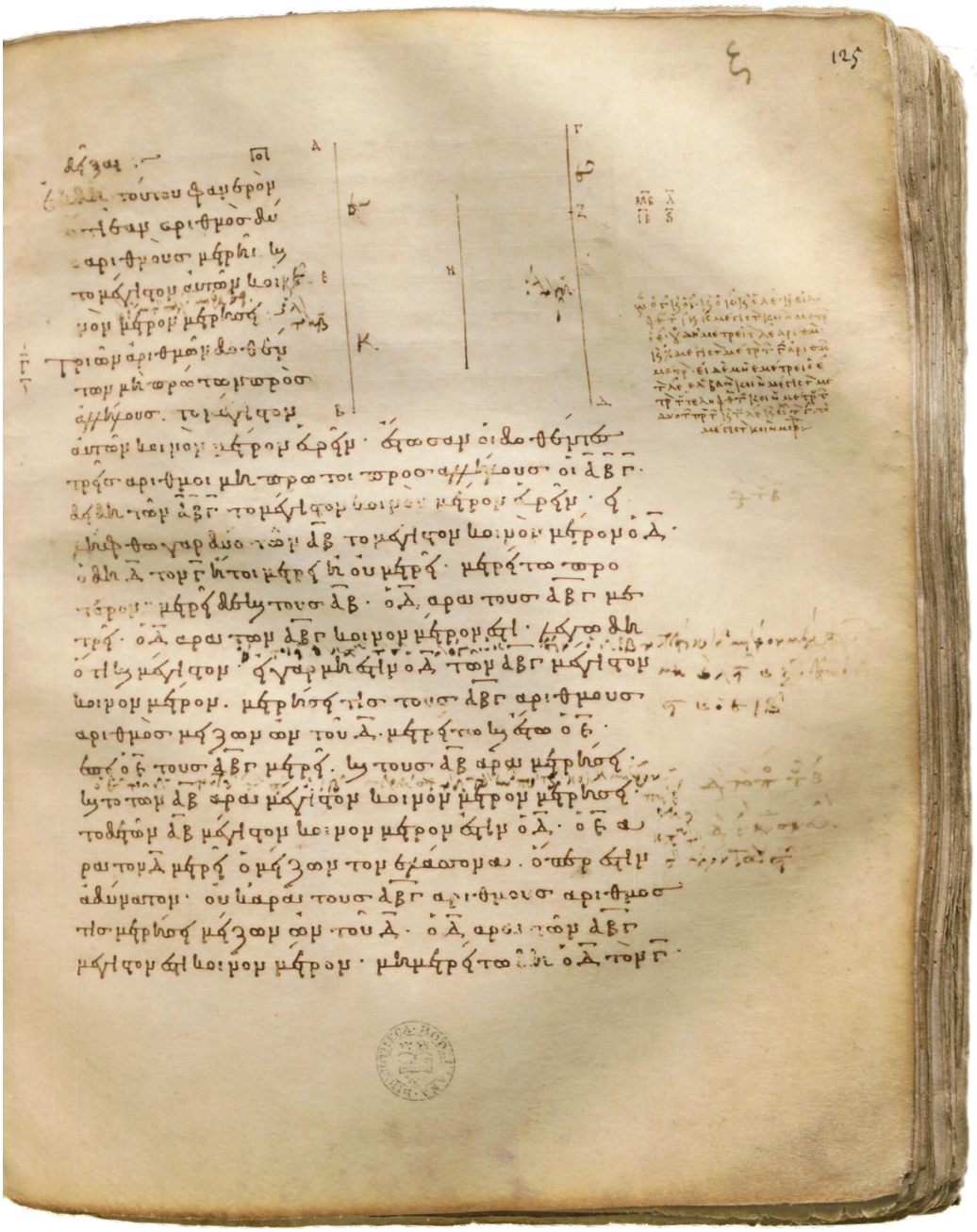


FIGURE 1.2 – Utilisation de l'anthyphérèse dans les *Εὐκλείδου Στοιξείων* (les Eléments d'Euclide) [22]. Il s'agit du calcule du plus grand dénominateur commun de deux et trois nombres (livre VII propositions 2 et 3). Fac-similé du manuscrit de Stephanos le clerc daté de 888 (selon l'édition de Théon d'Alexandrie), conservée aux *Bodleian Libraries, University of Oxford.*

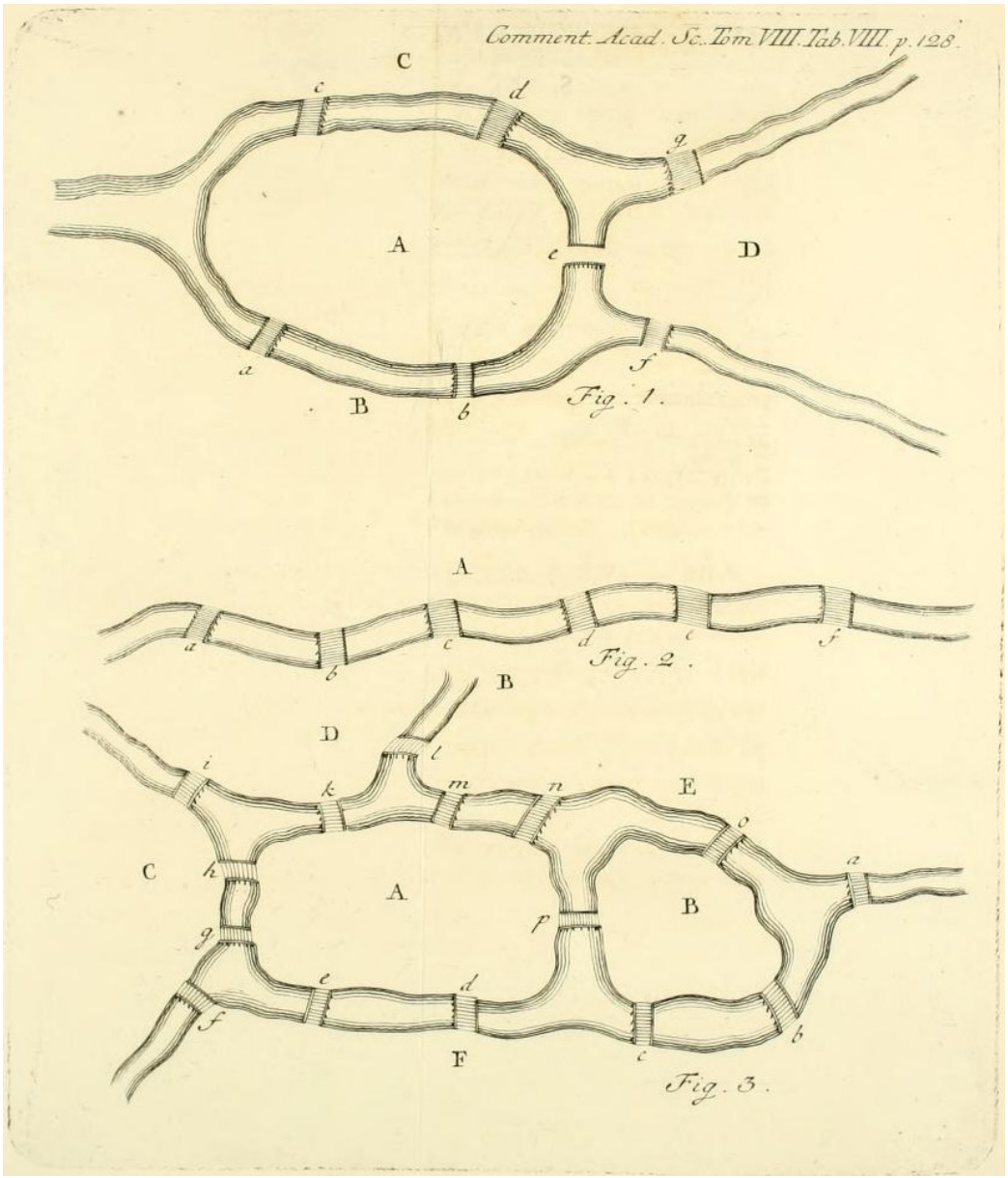


FIGURE 1.3 – Schéma du problème des sept ponts de Königsberg (*Fig.1*) dans le *Solutio problematis ad geometriam situs pertinentis* d'Euler [23]. Il s'agit d'un multigraphe dont les sommets sont A, B, C, D et les arêtes a, b, c, d, e, f, g. Les autres figures sont des cas imaginaires (*Fig.2* et *Fig.3*). Fac-similé du tome 8 des *Commentarii Academiae scientiarum imperialis Petropolitanae* publié en 1726 et conservé à l'*American Museum of Natural History Library*.

1.2 Quelques notions de graphes

Dans cette section nous donnons les définitions de graphe et des notions élémentaires qui y sont liées. Ces notions permettent de décrire, de manipuler et de modifier les graphes, soit localement, soit de façon globale. Nous définissons ensuite quelques classes de graphes auxquelles nous restreignons nos problèmes (réputés difficiles sur les graphes en général). Nous décrivons enfin la largeur arborescente, un paramètre structurel particulièrement utilisé en algorithmie paramétrée et dont nous avons besoin (dans le chapitre 3).

1.2.1 Éléments de graphes

Un graphe est un objet mathématique dont l'aspect discret et combinatoire en fait un outil très largement utilisé en informatique. Il sert généralement à modéliser un réseau : réseau routier, réseau informatique, réseau social... mais aussi des arbres phylogénétiques, des interactions moléculaires, des structures atomiques... et plus généralement tout ensemble dont les éléments peuvent interagir deux à deux. Outre leurs nombreuses applications, l'étude des graphes pour eux même, en tant qu'objet abstrait forme à elle seul un domaine mathématique [49].

Le premier problème de graphe est attribué à Euler ; il s'agit du problème des sept ponts de Königsberg. Sachant que la ville de Königsberg est traversé par le fleuve Pregel, Euler pose la question suivante « Est-t-il possible de faire une promenade en passant une et une seul fois par chaque pont ? » (étant entendu qu'on ne peut traverser le Pregel qu'en passant sur les ponts). Euler démontre qu'une tel promenade est impossible et propose un argument permettant de savoir si une telle promenade est possible pour n'importe quelle configuration de fleuve et de ponts, autrement dit, pour n'importe quel graphe.

Intuitivement, un graphe est le dessin d'un ensemble de points, appelés sommets, reliés deux à deux par des lignes, appelées arêtes. Plus formellement, un graphe peut être vu comme un ensemble d'éléments (les sommets) muni d'une relation binaire (représentée par les arêtes). Dans cette thèse, nous considérons essentiellement des graphes simples. Les graphes simples sont sans boucle, c'est-à-dire qu'un sommet n'est pas en relation avec lui même ; sans arête multiple (dans le cas contraire nous utiliserons le terme de multigraphe), c'est-à-dire qu'il y a, au plus, une arête entre deux sommets ; et non orientés, c'est-à-dire que la relation est symétrique (dans le cas contraire il est question de graphe orienté).

Définition 1 : graphe

Un *graphe* G est un couple $(V(G), E(G))$ où $V(G)$ est un ensemble quelconque dont les éléments sont appelés les *sommets* de G, et $E(G)$ est un ensemble de paires de sommets dont les éléments sont appelés les *arêtes* de G.

⌟

Par abus de notation nous notons $|G| = |V(G)|$ la taille du graphe G.

Définition 2 : extrémité, incidence, adjacence

Les *extrémités* d'une arêtes $\{v; w\}$ sont les sommets v et w.
Une arête e est *incidente* à un sommet v si v est une extrémité de e.
Deux sommets v, w sont *adjacents* si $\{v; w\}$ est une arête de E.

⌟

Vérifier l'adjacence de deux sommets ou l'incidence d'une arête et d'un sommet est généralement considérer comme une opération élémentaire d'un algorithme.

Définition 3 : graphes particuliers

Le graphe *complet* à n sommets K_n est un graphe tel que :

— $V(K_n) = \{v_1, \dots v_n\}$

— *et* $E(K_n) = \{\{v_i; v_j\} : i, j \in [1, n]\}\}$.

Le graphe *biparti complet* à $n + m$ sommets $K_{n,m}$ est un graphe tel que :

— $V(K_{n,m}) = \{v_1, \dots v_n\} \cup \{w_1, \dots w_m\}$

— *et* $E(K_{n,m}) = \{\{v_i; w_j\} : i \in [1, n], j \in [1, m]\}$.

Le graphe *étoile* à $n + 1$ sommets E_{n+1} est un graphe tel que :

— $V(E_{n+1}) = \{v, w_1, \dots w_n\}$

— *et* $E(E_{n+1}) = \{\{v; w_j\} : j \in [1, n]\}$.

Le graphe *grille* à $n \times n$ sommets Γ_n est un graphe tel que :

— $V(\Gamma_n) = \{v_{1,1}, \dots v_{n,n}\}$

— *et* $E(\Gamma_n) = \{\{v_{i,j}; v_{i+1,j}\}\{v_{i,j}; v_{i,j+1}\} : i, j \in [1, n-1]\}$
$\cup \{\{v_{i,n}; v_{i+1,n}\}\{v_{n,j}; v_{n,j+1}\} : i, j \in [1, n-1]\}$.

Le graphe *pseudo-grille* à $n \times n$ sommets Γ'_n est un graphe tel que :

— $V(\Gamma'_n) = \{v_{1,1}, \dots v_{n,n}\}$

— *et* $E(\Gamma'_n) = E(\Gamma_n)$
$\cup \{\{v_{i,j}; v_{i+1,j+1}\} : i, j \in [1, n-1]\}$
$\cup \{\{v_{1,n}; v_{i,1}\}, \{v_{1,n}; v_{i,n}\} : i \in [2, n]\}$
$\cup \{\{v_{1,n}; v_{1,j}\}, \{v_{1,n}; v_{n,j}\} : j \in [1, n-1]\}$.

⌟

Remarquons que la définition formelle correspond bien à l'intuition que nous avons donné : puisque une paire est un ensemble de deux éléments distincts non ordonnés, les arêtes ne bouclent pas et ne sont pas orientées. Par la suite, nous ne considérerons jamais de graphes avec boucle ni de graphes orientés. Nous aurons par contre recours aux multigraphes dans le chapitre 2 ; étant donné que nous utilisons les multigraphes en tant qu'outils et non en objets d'étude, nous reportons leur définition à ce chapitre 2.

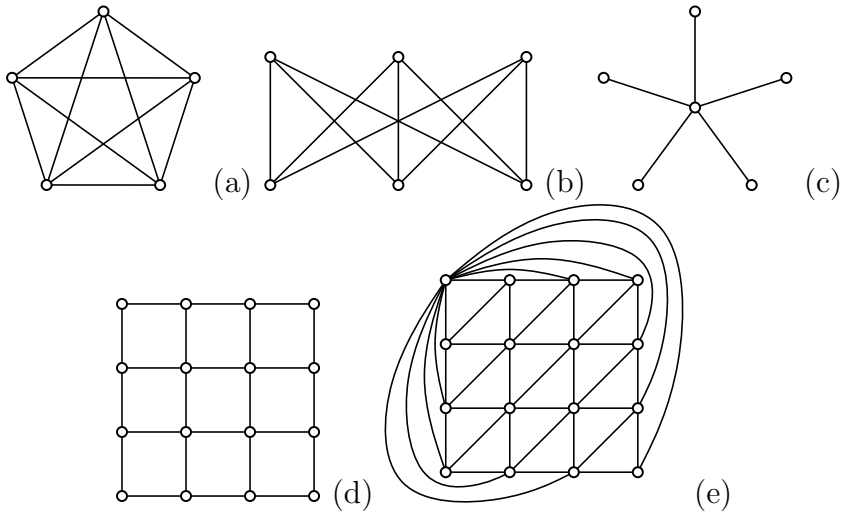


FIGURE 1.4 – Exemple de différents graphes (cf. définition 3). (a) Le graphe complet à 5 sommets. (b) Le graphe biparti complet à $3+3$ sommets. (c) L'étoile à $5+1$ sommets. (d) La grille à 4×4 sommets. (e) La pseudo-grille à 4×4 sommets.

Puisque nous étudions des problèmes de graphe, nous allons devoir élaborer des algorithmes qui manipulent les graphes, soit en les parcourants, soit en les transformant. Dans notre cas, nous allons surtout transformer les instances de nos problèmes. Nous voulons donc ajouter ou supprimer des sommets, des arêtes, ou contracter des arêtes, c'est-à-dire, identifier ses deux extrémités.

Définition 4 : suppressions, contraction

La *suppression d'un sommet* v dans un graphe G, noté $G-v$ est l'opération qui transforme $V(G)$ en $V(G) \setminus v$ et $E(G)$ en $E(G) \setminus \{\{v;u\} : u \in V(G)\}$.

La *suppression d'une arête* e dans un graphe G, noté $G \setminus e$, est l'opération qui transforme $E(G)$ en $E(G) \setminus e$.

La *contraction d'une arête* $\{v;w\}$ dans un graphe G, noté $G/\{v;w\}$ est l'opération qui transforme $V(G)$ en $V(G) \setminus \{v,w\} \cup \{y\}$ et $E(G)$ en $E(G) \setminus$

$(\{\{v;u\} : u \in V(G)\} \cup \{\{w;u\} : u \in V(G)\}) \cup (\{\{y;u\} : \{v;u\} \in E(G)\} \cup (\{\{y;u\} : \{w;u\} \in E(G)\})$.

⌟

Chacune des ces opérations étant commutative, elles peuvent naturellement être généralisées à un ensemble de sommets ou d'arête ; nous notons alors $G - V$, $G \setminus E$, G/E, respectivement. Par abus de langage nous dirons qu'une arête est contractée sur une de ses extrémités lorsque nous voulons réutiliser le nom de l'extrémité pour le nouveau sommet. Ces opérations, ainsi que l'adjonction d'un sommet ou d'une arête, peuvent être considérées comme les instructions élémentaires, s'exécutant en temps constant, qu'un algorithme peut effectuer sur un graphe.

Relation d'ordre sur les graphes

Un bonne manière de comprendre la structure d'un graphe est de déterminer s'il contient (pour une bonne définition de contenir) la structure d'un autre graphe, plus simple et plus compréhensible.

La notion de sous-graphe est la manière la plus naturelle d'exprimer la notion d'inclusion entre deux graphes. En effet, un sous-graphe H d'un graphe G est tel que $V(H) \subseteq V(G)$ et $E(H) \subseteq E(G)$. Un sous-graphe induit d'un graphe G est un sous-graphe qui respecte les adjacences de G.

Définition 5 : sous-graphe (induit)
Étant donné G un graphe, un sous-graphe H de G est un graphe obtenu par suppression de sommets et d'arêtes de G.
Étant donné $V \subseteq V(G)$, le *sous-graphe induit* $G[V]$ par V est le graphe obtenu par suppression des sommets de $V(G) \setminus V$, $G[V] = (V, E \cap V^2)$.

⌟

Par abus de notation nous notons $\partial H = \partial V(H)$ le bord d'un sous-graphe H de G, c'est-à-dire l'ensemble des sommets de H dont le voisinage dans G n'est pas inclus dans H.

Remarquons qu'un sous-graphe induit est en particulier un sous-graphe. Remarquons aussi qu'un sous-graphe est obtenu à partir d'un sous-graphe induit par suppression d'arêtes.

Un mineur d'un graphe G est un graphe obtenu par contraction d'arête d'un sous-graphe de G. Intuitivement, cela traduit le fait que la structure recherchée peut avoir été dilatée dans G. Cette notion a été introduite par Robertson et Seymour. Un mineur topologique est obtenu en contractant uniquement des arêtes subdivisées. Intuitivement, un mineur topologique à la même topologie (le même dessin) que le graphe qui le contient.

Définition 6 : mineur (topologique)

Étant donné G un graphe, un *mineur* H de G est un graphe obtenu par suppression de sommets et d'arêtes de G et par contraction d'arêtes de G.

Étant donné G un graphe, un *mineur topologique* H de G est un graphe obtenu par suppression de sommets et d'arêtes de G et par contraction d'arête de G dont une extrémité est de degré deux.

⌟

Remarquons qu'un mineur topologique est en particulier un mineur. Remarquons aussi qu'un mineur est obtenu à partir d'un sous-graphe en contractant des composantes connexes et qu'un mineur topologique est obtenu en contractant des chemins induits.

Les notions de sous-graphes induits, sous-graphes, mineurs topologiques, et mineurs impliquent des relations d'ordre partiel sur les graphes. De plus, l'ordre partiel dérivé de la notion mineurs est bien fondé. La démonstration par Robertson et Seymour [53] de ce théorème (conjecturé par Wagner) constitue un résultat majeur en théorie des graphes.

Voisinage

Le voisinage d'un sommet est l'ensemble des sommets qui lui sont proches. Nous en faisons abondamment usage dans le chapitre 2. Le degré est le nombre de ses voisins. Ce sont deux façons de décrire localement un graphe.

Définition 7 : voisinage

Le *voisinage ouvert* $N_G(v)$ d'un sommet v est l'ensemble des sommets adjacents à v, $N_G(v) = \{u \mid \{u; v\} \in E(G)\}$.

Le *voisinage fermé* $N_G[v]$ d'un sommet v est l'ensemble $N_G(v) \cup \{v\}$.

Les *voisinages* $N_G[S]$ et $N_G(S)$ d'un ensemble de sommets $S \subseteq V(G)$ sont respectivement les ensembles $N_G[S] = \bigcup_{v \in S} N_G[v]$ et $N_G(S) = N_G[S] \setminus S$.

Nous notons $N_G^r[v] = N_G[N_G^{r-1}[v]]$ et $N_G^r(v) = N_G(N_G^{r-1}(v))$ respectivement, avec $N_G^1[v] = N_G[v]$ et $N_G^1(v) = N_G(v)$.

⌟

Le graphe G étant généralement claire d'après le contexte, nous notons $N_G(v) = N(v)$. Dans le chapitre 2, nous utilisons surtout la notion de voisinage d'un ensemble dans le cas d'une paire v, w ; dans ce cas, pour la simplicité, nous notons $N[\{v, w\}] = N[v, w]$ et $N(v, w) = N[v, w] \setminus \{v, w\}$. La notation étant explicite, nous ne précisons pas textuellement si le voisinage est ouvert ou fermé.

Définition 8 : degré

Étant donné un graphe G, le *degré* $\delta_G(v)$ d'un sommet v est le nombre d'arêtes incidentes à v.

⌟

Remarquons que pour un graphe G, $\delta_G(v) = |N(v)|$. Le graphe G étant généralement clairement fixé, nous omettons l'indice : $\delta_G(v) = \delta(v)$.

Chemins

Un chemin entre deux sommets est la partie du graphe qu'il faut parcourir pour relier un sommet à l'autre. Les chemins permettent de définir la notion de distance et de connexité dans une graphe.

Définition 9 : chemin, parcours, marche

Une *vw-marche* est une suite de sommets et d'arêtes $(v = u_0, e_1, u_1, e_2, u_2, \ldots u_{r-1}, e_r, u_r = w)$ telle que $v, w, u_i \in V(G), e_i \in E(G)$ et $e_i = \{u_{i-1}; u_i\}$ pour $i \in [1, r]$.

Un *vw-parcours* est une vw-marche dont toutes les arêtes sont distinctes deux à deux.

Un *vw-chemin* est une vw-marche dont tous les sommets (et donc toutes les arêtes) sont distincts deux à deux.

Une marche *fermée* est une vw-marche telle que $v = w$.

Un *cycle* est une marche fermée dont tous les sommets (sauf ses extrémités) sont distincts deux à deux.

La *longueur* d'une marche est le nombre d'arêtes (avec multiplicité) qu'elle contient.

⌟

Formulons quelques remarques à propos de cette définition. Un vw-chemin est nécessairement un vw-parcours qui est nécessairement une vw-marche. De façon équivalente une marche peut être définie comme un suite d'arêtes $(e_1, e_2, \ldots e_r)$ telle que $e_i = \{u_{i-1}; u_i\}$ et $e_{i+1} = \{u_i; u_{i+1}\}$ (autrement dit, deux arêtes successives ont une extrémité commune). De même, une marche peut être définie comme une suite de sommets $(v = u_0, u_1, u_2, \ldots u_{r-1}, u_r = w)$ telle que $\{u_{i-1}; u_i\} \in E(G)$ (autrement dit, deux sommets successifs sont adjacents). Nous dirons qu'un sommet ou qu'une arête appartient à une marche si le sommet ou l'arête est un éléments de la suite. Enfin, les marches, parcours et chemins sont parfois appelés respectivement chemins, chemins simples, chemins élémentaires.

Dans les chapitres suivants nous considérons le plus souvent des chemins définis comme une suite de sommets. Nous n'utiliserons la notion de marche que dans cette section (cf. définition 17) et dans la section 2.1.

Définition 10 : distance
Étant donné un graphe G, la *distance* $d_G(v, w)$ entre deux sommets v, w est la longueur d'un plus court vw-chemin.
⌟

Remarquons que, de façon équivalente, $d_G(v, w)$ est le plus petit entier r tel que $w \in N^r[v]$. Lorsque que le graphe G est clairement défini par le contexte, pour la simplicité, nous noterons $d_G(v, w) = d(v, w)$.

Définition 11 : graphe connexe
Un graphe G est *connexe* si pour toute paire de sommets v, w, il existe un vw-chemin.
Une *composante connexe* d'un graphe G est un sous-graphe de G, maximal et connexe.
⌟

Définition 12 : séparateur
Un *séparateur* dans un graphe G est un ensemble de sommets tel que $G - S$ a au moins deux composantes connexes.
⌟

1.2.2 Classes de graphes peu denses

Comme nous l'avons déjà mentionné, les graphes ont un grand pouvoir d'expression, et par conséquent une structure très complexe. Pour cette raison, il est souvent difficile de résoudre un problème sur un graphe quelconque, mais si le problème est restreint à une classe de graphes (c'est-à-dire un ensemble (infini) de graphes définis par une caractéristique commune) il est parfois possible de le résoudre efficacement. Dans cette thèse nous nous intéressons, en particulier, à la classe des graphes planaires et à ses généralisations : les classes des graphes de genre borné, les classes des graphes sans mineur, et les classes des graphes sans mineur topologique. Ce sont des classes de graphes peu denses au sens ou le rapport entre nombre d'arêtes et nombre de sommets est faible (c'est-à-dire sous-quadratique).

Graphes de genre borné

Intuitivement, un graphe planaire est un graphe qui peut être dessiné dans le plan sans que ses arêtes ne se croisent. Plus généralement un graphe est de genre borné s'il peut être dessiné sur une surface de genre borné.

Dans le chapitre 2 nous ne considérons que des graphes planaires. Dans le chapitre 3 nous formulons nos résultats pour les graphes sans mineur, ce qui est plus générique. Par conséquent nous n'aurons pas à utiliser de graphes

de genre borné et nous n'en donnons la définition que pour information.

Pour donner la définition formelle de ces graphes, il faut utiliser un plongement, c'est-à-dire une application qui pour chaque élément du graphe indique où il doit être dessiné.

Définition 13 : plongement

Étant donné un graphe G et une surface $\mathcal{S}$, un *plongement* π de G est une application $\pi : G \longrightarrow \mathcal{S}$ dont l'image de chaque sommet est un point, et dont l'image de chaque arête est une courbe simple, tels que :

- pour toute paire de sommets distincts $v, w \in V$,
  les images $\pi(v)$ et $\pi(w)$ sont distinctes ;
- *et* pour toute arête $\{v; w\} \in E$,
  les images $\pi(v)$ et $\pi(w)$ sont les points extrémaux de $\pi(\{v; w\})$;
- *et* pour toute arête $\{v; w\} \in E$ et tout sommet u distinct de v, w,
  l'image $\pi(\{v; w\})$ n'intersecte pas l'image $\pi(u)$;
- *et* pour toute paire d'arêtes $e_1, e_2 \in E$,
  l'intersection $\pi(e_1) \cap \pi(e_2)$ ne contient que des points extrémaux.

⌟

Remarquons que, après une suppression ou une contraction, il y a une façon naturelle de plonger le nouveau graphe à partir d'un plongement du graphe initial.

Définition 14 : graphe planaire

Un graphe G est *planaire* s'il existe un plongement de G dans le plan.
Un graphe *plan* est une graphe muni d'un plongement fixé.
Un graphe planaire est *maximal* si l'ajout d'une arête le rend non planaire.

⌟

Définition 15 : graphe apex

Un graphe *apex* est un graphe avec un sommet v tel que $G - v$ est planaire.

⌟

Il s'avère que de nombreux problèmes considérés comme difficiles peuvent être résolus efficacement dans les graphes planaires. Il est alors naturel de se demander ce qu'il advient lorsqu'un graphe est plongé dans un autre type de surfaces. En fait, pour pouvoir plonger un graphe, la caractéristique d'une surface qui permet de distinguer les graphes est le genre. Le genre est une notion définie pour les surfaces closes (compact, connexe et sans bord ; au

sens topologique). Intuitivement, le genre d'une surface est le nombre de trous qu'il faut faire à la sphère pour obtenir cette surface. Plus précisément, c'est le nombre de courbe nécessaire pour déconnecter la surface.

Définition 16 : genre borné

Le *genre* g d'un graphe est le plus petit entier tel que G puisse être plongé sur une surface de genre g.

⌟

Remarquons que le plan n'est pas une surface close. Cependant nous pouvons facilement constater que les graphes planaires sont exactement les graphes plongeable sur la sphère. Il en découle que les graphes planaires forment la classe des graphes de genre 0. Le tore, par contre, est de genre 1, un graphe qui peut être plongé dans le tore n'est pas nécessairement planaire. Intuitivement, les graphes de genre g sont plus denses que les graphes de genre $g-1$ au sens où les trous permettent de dessiner plus d'arêtes.

Observons aussi que la classe des graphes de genre g est close par les opérations de suppression de sommet, suppression d'arête, et contraction d'arête, au sens où l'application de l'une de ces opérations sur un graphe n'augmente pas son genre. En d'autres termes, si G est de genre g et H est un mineur de G alors H est de genre au plus g.

Une fois le graphe plongé, les zones délimitées par le dessin du graphe sont fixées. Il est naturel de vouloir les considérer comme des éléments constitutifs du graphe. Cette idée est formalisée par la notion de face.

Définition 17 : face

Étant donné G un graphe plan, une *face* de G est un ouvert connexe f du plan tel que $V(f) = \emptyset$ et ∂f est composé d'images de marches fermées.

Un sommet ou une arête est *incident* à une face s'il appartient à une des marches. Le *degré* $\delta(f)$ de la face est la somme des longueurs des marches.

Nous notons $F(G)$ l'ensemble des faces du graphe plan G.

⌟

Les faces peuvent également être définies comme les composantes connexes du plan privé de l'image du graphe. Remarquons que $\delta(f) \neq |V(\partial f)|$ car certains sommets et arêtes peuvent apparaître plusieurs fois dans la marche qui délimite la face. Remarquons aussi que, si le bord de la face contient plusieurs marches, alors le graphe a au moins une composante connexe par marche ; intuitivement cela veut dire que la face contient des trous.

Sauf certains cas pathologiques (comme les faces trouées), il est possible de considérer un graphe comme une polygonation de la surface sur laquelle il

est plongé. La terminologie des graphes (sommet, arête, et face) correspond alors à celle des polygones. La notion de plongement cellulaire permet d'éviter les cas pathologiques.

Définition 18 : plongement cellulaire
Un plongement est *cellulaire* si toute face est homéomorphe à un disque. ⌟

Le plongement cellulaire d'un graphe est parfois appelé une carte.

Formulons quelques remarques à propos de cette définition. D'abord, il n'est pas possible de plonger cellulairement un graphe dans le plan, car le plan n'est pas une surface close (il y a une face extérieure). Pour plonger cellulairement un graphe planaire il convient de le plongé sur la sphère. Ensuite, (contrairement un plongement) un graphe cellulairement plongeable sur un surface de genre g n'est pas nécessairement plongeable cellulairement sur une surface de genre $g+1$; par exemple K_3 (de genre 0) n'est pas plongeable cellulairement sur le tore (de genre 1), alors que K_4 (de genre 0) l'est. Enfin, les graphes de genre g ne sont pas tous plongeables cellulairement sur un surface de genre g. Dans le cas particulier des graphes planaires, un graphe planaire est plongeable cellulairement sur la sphère si est seulement s'il est connexe.

Pour cette raison, et parce-que nous ne plongerons que des graphes planaire, nous n'aurons pas besoin d'utiliser explicitement la notion de plongement cellulaire.

La formule d'Euler montre que les graphes de genre borné sont peu denses. En particulier, dans les graphes planaires, elle a pour corollaire que le nombre d'arêtes est linéaire en le nombre de sommets. La formule d'Euler s'applique également aux multigraphes ; nous montrerons (cf. lemme 3) que les multigraphes planaires avec une condition supplémentaire, vérifient aussi le corollaire.

Lemme 1 : Formule d'Euler
Soit G un graphe plongé cellulairement sur une surface de genre g. Les nombres de sommets, d'arêtes et de faces de G vérifie l'égalité $|V(G)| - |E(G)| + |F(G)| = 2 - 2g$.

Soit G un graphe planaire. Si $|V(G)| \geqslant 3$ alors $|E(G)| \leqslant 3|V(G)| - 6$. ⌞ ⌟

Le fait de fixer le plongement d'un graphe peut simplifier énormément la manipulation de celui-ci. D'une part, le plongement permet de faire un découpage géométrique du graphe ; c'est le principe de la décomposition en régions (cf. chapitre 2). D'autre part, étant donné un sommet v, le plongement fixe un ordre cyclique sur les voisins de v. Autrement dit, un même

graphe peut avoir plusieurs représentations (plusieurs plongements), et nous en fixons une, limitant ainsi les possibilités combinatoires.

Lorsque nous disons que nous pouvons faire des considérations géométriques sur le graphe, cela signifie que nous pouvons manipuler un graphe à partir de zones du plan. Nous posons quelques notations à cet effet.

Définition 19 : sous-ensemble du plan

Étant donné G un graphe plan, π son plongement, et R un sous-ensemble du plan (ouvert ou fermé), nous notons :

— $V(R) = \{v \mid \pi(v) \in R\}$ l'ensemble des sommets dans R,

— $|V(R)|$ la *taille* de R,

— ∂R le bord de R.

⌟

Remarquons que, par abus de langage, nous utilisons le même terme et la même notation pour deux objets différents : le bord d'un sous-graphe ∂G et le bord d'un sous-ensemble du plan ∂R. Cet abus est justifié par le fait que, dans cette thèse, nous ne considérons que des sous-ensembles R du plan tels que le bord du sous-graphe induit par les sommets dans R a son image dans le bord de R, autrement dit, les images des sommets dans $\partial G[V(R)]$ sont dans ∂R.

Un autre intérêt du plongement, avons-nous dit, est qu'il permet d'ordonner, partiellement, les sommets. Intuitivement, autour de v, les sommets sont ordonnés par la position de leur image. Cela se formalise avec le concept d'ordre cyclique.

Définition 20 : ordre cyclique

Un *ordre cyclique* est une relation ternaire cyclique, asymétrique, transitive et totale.

Étant donné G un graphe plan et π son plongement, l'*ordre autour* de $v \in V$ est un ordre cyclique $[u_1 < u_2 < u_3]_v$ sur $N(v)$ tel que dans toute boule B_ε de centre $\pi(v)$ et de rayon $\varepsilon > 0$ suffisamment petit, les images des arêtes $\{v; u_1\}, \{v; u_2\}, \{v; u_3\}$ sont dans l'ordre trigonométrique.

Un sommet u est *minimum* autour de v à partir de w s'il n'existe pas de sommet $\tilde{u} \in N(v)$ tel que $[w < \tilde{u} < u]_v$.

⌟

Graphes sans mineur (ou mineur topologique)

Le plus souvent, les graphes sans mineur sont présentés comme une généralisation des graphes de genre borné. En effet, nous avons observé que

la classe des graphes de genre g est close par mineur. Or les travaux de Robertson et Seymour [53] ont démontré que les classes de graphes closes par mineur peuvent être caractérisées par un ensemble fini de mineurs exclus.

Du point de vue algorithmique, la plus part des problèmes pouvant être résolus efficacement sur les graphes planaires peuvent également l'être sur les graphes de genre borné et sur les graphes sans mineur.

Définition 21 : graphe sans mineur

Un graphe G est *sans mineur* H si H n'est pas un mineur de G.

⌟

Définition 22 : classe excluant un mineur

La classe des graphes $\mathcal{G}$ *excluant un mineur* H est l'ensemble de tous les graphes sans mineur H.

⌟

Dans la suite, assez souvent, nous n'avons pas besoin d'expliciter le mineur H qui est exclu, car la propriété ou le théorème que nous considérons est vrai quelque soit le mineur (bien que ses constantes dépendent de H). Dans ce cas, nous parlons de graphe sans mineur et de classe excluant un mineur, en sous-entendant un mineur H quelconque, mais fixé.

Nous avons mentionné que, d'après les travaux de Robertson et Seymour, les classes de graphes genre borné peuvent être caractérisés par un ensemble fini de mineur exclus. Par exemple, d'après la caractérisation de Kuratowski et Wagner [46, 54, 49], la classe des graphes planaires est la classe des graphes excluant K_5 et $K_{3,3}$ (le graphe complet à 5 sommets et le graphe biparti complet à $3 + 3$ sommets). C'est la seule classe de graphe de genre borné dont les mineurs exclut sont explicitement connus.

Théorème 1 : Kuratowski et Wagner [46, 54]

Soit G un graphe. G est planaire si et seulement si G est sans mineur K_5 ni mineur $K_{3,3}$.

⌞ ⌟

Il est possible de généraliser encore ses classes, en gardant la propriété informelle de faible densité d'arête. Il s'agit des classes de graphes excluant un mineur topologique. Rappelons que, si H est un mineur topologique de G alors c'est un mineur de G. Si une classe de graphes exclut un mineur H alors, en particulier elle exclut H comme mineur topologique. Les graphes sans mineur topologique sont donc bien une généralisation des graphes sans mineur.

Nous utilisons les classes de graphe excluant des mineurs essentiellement dans le chapitre 3. Dans les trois applications que nous proposons nous restreignons le problème aux graphes sans mineur. Mais notre théorème 4 peut être énoncé pour les graphes sans mineur topologique.

1.2.3 Largeur arborescente

Une autre classe de graphes sur laquelle la plupart des problèmes deviennent simples est la classe des arbres, c'est-à-dire des graphes ne contenant pas de cycle (comme sous-graphe). Dans le but de pouvoir résoudre ces mêmes problèmes sur un plus grand nombre de graphes, il est naturel de chercher à généraliser les arbres. La largeur arborescente [4, 41, 50] est un paramètre qui mesure à quel point la structure d'un graphe est proche de celle d'un arbre.

Du point de vue algorithmique, les méthodes de résolution qui peuvent être appliquées sur les arbres, peuvent s'adapter aux graphes de largeur arborescente bornée ; en particulier la programmation dynamique est une technique particulièrement efficace sur les décompositions arborescentes.

Les arbres sont les graphes les plus simples qui soient. Ils ont été très largement étudiés et leurs applications sont nombreuses. Nous n'en donnons ici qu'une rapide description.

Définition 23

Un arbre T est un graphe connexe tel que $|E(T)| = |V(T)|-1$. Les *noeuds* est les sommets d'un arbre, et une *feuille* est un nœud de degré 1.

Un arbre est *enraciné* s'il contient un sommet distingué r appelé *racine*. Un nœud n' est le *fils* d'un nœud n si n est le successeur de n' sur un rn'-chemin.

⌟

De façon équivalente, un arbre est un graphe connexe sans cycle. Les arbres forment une famille de graphe avec de nombreuses propriétés. Remarquons simplement que entre deux nœuds d'un arbre il existe un unique chemin ; et que dans un arbre enraciné, un nœud n définit naturellement un sous-arbre enraciné en n.

La décomposition arborescente permet d'organiser le graphe comme un arbre. Intuitivement, une décomposition arborescente de G est un arbre dont chaque nœud est un séparateur de G et dans lequel, passer d'un nœud à un nœud adjacent correspond à décaler le séparateur dans G. (cf. figure 1.5).

Définition 24 : décomposition arborescente

Une *décomposition arborescente* d'un graphe G est un couple $(T, \mathcal{X} =$

$\{X_n : n \in V(T)\}$) où T est un arbre dont chaque nœud n est étiqueté par un *sac* $X_n \subseteq V(G)$ tel que :

- pour tout sommet $v \in V(G)$, il existe un sac X tel que $v \in X$;
- *et* pour toute arête $\{v; w\} \in E(G)$, il existe un sac X tel que $v, w \in X$;
- *et* pour tout sommet $v \in V(G)$, l'ensemble $\{n \in V(T) : v \in X_n\}$ induit un sous-arbre de T.

Une décomposition arborescente $(T, \mathcal{X})$ est *bonne* si T est un arbre binaire enraciné tel que :

- si $n \in V(T)$ à un seul fils n', alors $|X_n| - 1 \leqslant |X_{n'}| \leqslant |X_n| + 1$;
- si $n \in V(T)$ à deux fils n_1, n_2, alors $X_n = X_{n_1} = X_{n_2}$.

⌟

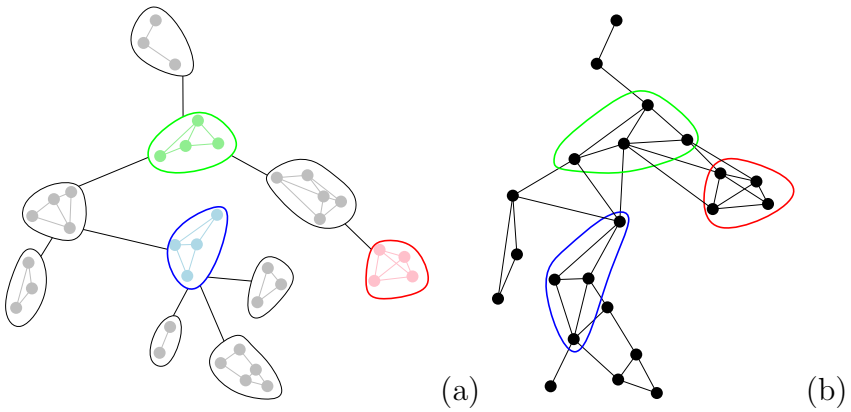


FIGURE 1.5 – Exemple d'une décomposition arborescente de largeur 4 (cf. définition 24). (a) L'arbre et ses sacs. (b) Le graphe et trois de ses séparateurs. Cette décomposition n'est pas optimale car le graphe est de largeur arborescente 3. Chaque sac de la décomposition arborescente correspond à un séparateur du graphe.

Il est important de noter que chaque sac X_n (sauf si n est une feuille) est un séparateur du graphe G. En effet, cela implique que le sous-arbre de T enraciné en n correspond à un sous-graphe de G relativement indépendant du reste de G. De cette façon il est possible résoudre de nombreux problèmes par programmation dynamique, en partant des feuilles de la décomposition arborescente et en remontant jusqu'à la racine.

Définition 25 : largeur arborescente

La *largeur* d'une décomposition arborescente $(T, \mathcal{X})$ est $\max\{|X| : X \in \mathcal{X}\} - 1$.

La *largeur arborescente* $\mathrm{tw}(G)$ d'un graphe G est la largeur minimale des décompositions arborescentes de G.

⌟

Pour les problèmes que nous considérons (dans le chapitre 3), nous ne nous restreignons pas aux graphes de largeur arborescente bornée. Nos instances sont donc de largeur arborescente arbitraire. Cependant, pour pouvoir faire la programmation dynamique, nous avons besoin de nous ramener à des graphes de largeur arborescente bornée, en supprimant un ensemble de sommets (si possible petit). Un tel ensemble est appelé un modulateur d'arborescence.

Définition 26 : modulateur d'arborescence

Un *t-modulateur d'arborescence* X d'un graphe G est un ensemble de sommet tel que $\mathrm{tw}(G - X) \leqslant t$.

⌟

1.3 Quelques bases d'algorithmie

Dans cette section nous donnons quelques éléments d'algorithmie et de complexité. Nous discutons d'abord du concept de problème et nous donnons une définition formelle de la catégorie de problèmes paramétrés que nous étudierons. Nous continuons par quelques rappels de complexité classique. Puis, nous présentons la notion de noyau, l'objet d'étude de cette thèse. Nous replaçons enfin les noyaux dans le contexte des algorithmes paramétrés et nous argumentons sur le bien fondé de cette approche.

1.3.1 Problèmes paramétrés

Comme nous l'avons fait remarquer dans le prologue, donner une définition (compréhensible et formelle) de problème dans tout sa généralité n'est pas chose aisée. Nous nous contenterons donc de donner une définition qui couvre l'ensemble des problèmes que nous voulons étudier dans cette thèse, assavoir des problèmes d'optimisation dans les graphes sous leur forme décisionnelle et paramétrés par la taille de la solution recherchée.

Pour donner une définition formelle de ces problèmes, nous avons recours à la théorie des langages, dont il nous faut faire quelques rappels. Un alphabet Γ est un ensemble de symboles (ou lettres) arbitraire. Ces symboles peuvent être concaténés pour former des mots. L'ensemble des mots sur Γ est noté Γ^*. Un langage Π est un sous-ensemble de Γ^* qui est reconnaissable par une machine de Turing. Intuitivement, les symboles servent de briques de base pour décrire un objet mathématique (dans notre cas, un graphe). Les symboles sont ensuite assemblés pour former des mots qui, s'ils sont syntaxiquement corrects, décrivent un tel objet (un graphe). Enfin, un langage identifie un ensemble d'objets vérifiant une propriété (l'ensemble des instances positives du problème).

Pour mieux comprendre la catégorie de problèmes qui nous intéresse, commençons par décrire les catégories proches mais plus élémentaires. Tout d'abord, nous ne considérons que des problèmes de graphes, c'est-à-dire des problèmes dont l'instance est un graphe. Ensuite, les problèmes qui nous intéressent sont à l'intersection de trois catégories de problèmes : les problèmes de décision, les problèmes d'optimisation, et les problèmes paramétrés.

Les problèmes de décision sont des problèmes dont la réponse peut être OUI ou NON. En général, ils sont formalisés par un langage Π sur Γ, où Γ est un alphabet (décrivant les graphes). L'ensemble Γ^* est l'ensemble des ins-

tances possibles et le langage Π est tout simplement l'ensemble des instances positives. Bien souvent, le langage Π est défini à partir d'un autre langage L^Π par $\Pi = \{G \mid \exists S, (G,S) \in L^\Pi\}$, où L^Π est un langage qui vérifie que le certificat S a bien la propriété voulue dans G.

Les problèmes d'optimisation (de maximisation ou de minimisation) sont des problèmes dont la réponse est le coût optimal d'une solution. En général, ils sont formalisés par un langage L^Π et une mesure m^Π, où L^Π reconnaît les couples (G,S) lorsque S est une solution dans G, et où m^Π donne la mesure (parfois appelé le coût) d'une solution. La réponse au problème est donnée par la fonction d'optimisation $f^\Pi(G) = \text{opt}\{m^\Pi(S) \mid (G,S) \in L^\Pi\}$ (où opt est la fonction min ou max selon le cas).

Les problèmes paramétrés sont des problème de décisions dont chaque instance G est associée à un entier k. En général, ils sont formalisés par un couple Π, κ, où $\Pi \subseteq \Gamma^*$ est le langage d'un problème de décision et $\kappa : \Gamma^* \to \mathbb{N}$ est une paramétrisation, c'est-à-dire une fonction calculable (souvent, la littérature impose calculable en temps polynomial) qui donne le paramètre de chaque instance. Le paramètre ne modifie pas le problème en lui même, mais il influence l'approche pour concevoir un algorithme avec une bonne complexité paramétrée.

La théorie de la complexité ne s'intéresse qu'aux problèmes de décision. Pour pouvoir étudier un problème d'optimisation du point de vue de la théorie de la complexité, il faut donc le transformer en problème de décision. Intuitivement, lorsque le problème d'optimisation (minimisation) demande « Quelle est la mesure minimum d'un solution vérifiant telle propriété ? », la question de la version décision est « Existe-t-il une solution de mesure au plus k » (et inversement pour les problèmes de maximisation). Puisqu'il s'agit d'un problème de décision nous le formalisons par un langage $\Pi \subseteq \Gamma^* \times \mathbb{N}$ dont la forme est conditionnée par le problème d'optimisation L^Π,m^Π. Comme le problème est défini à partir d'un problème d'optimisation, nous pouvons définir une fonction d'optimisation associée.

Définition 27 : problème

Un *problème* Π est un langage de la forme $\Pi = \{(G,k) \mid \exists S, m^\Pi(S) \leqslant k, (G,S) \in L^\Pi\}$ où :

- L^Π est un langage calculable qui reconnaît des couples de la forme $(G,S) \in \Gamma^* \times \Sigma^*$, où G est un graphe décrit sur l'alphabet Γ et S est un certificat décrit sur un alphabet Σ ;
- m^Π est une mesure calculable de $\Sigma^* \to \mathbb{N}$.

⌟

Définition 28 : instance

Une *instance* de Π est un couple $(G, k) \in \Gamma^* \times \mathbb{N}$. Une instance (G, k) est *positive* si $(G, k) \in \Pi$, et *négative* si $(G, k) \notin \Pi$. La *taille de* (G, k) est $|(G, k)| = |G| + \log k$.

⌟

Définition 29 : fonction d'optimisation

La *fonction d'optimisation* de Π est la fonction $f^\Pi : \Gamma^* \to \mathbb{N}$ telle que :

$$f^\Pi(G) = \text{opt}\{m^\Pi(S) \mid (G, S) \in L^\Pi\}.$$

⌟

Formulons plusieurs remarques à propos de cette définition de problèmes.

D'abord, un problème d'optimisation et sa forme décisionnelle sont bien équivalents au sens où, d'une part, avoir une réponse au problème optimisation donne une réponse pour le problème de décision, et d'autre part, il suffit de répéter plusieurs fois l'algorithme du problème de décision pour obtenir une réponse au problème d'optimisation. Remarquons aussi que ces problèmes sont monotones au sens où, si (G, k) est positive alors (G, k') est positive pour tout $k' \geqslant k$ s'il s'agit d'un problème de minimisation (respectivement, $k' \leqslant k$ pour les problèmes de maximisation).

Ensuite, dans le chapitre 2, nous étudions des problèmes particuliers, et nous n'aurons pas besoin de faire référence à cette définition formelle. Dans le chapitre 3 nous proposons des méta-noyaux, c'est-à-dire que nous donnons une technique d'extraction de noyaux suffisamment générique pour être appliquée à un grand nombre de problèmes ; dans les énoncés du chapitre 3, lorsque nous parlons de problème, nous faisons expressément référence à cette définition.

Puis, pour les problèmes de cette forme les instances sont de la forme (G, k) et donc, la paramétrisation la plus naturelle est $\kappa : (G, k) \mapsto k$. C'est la seule paramétrisation que nous rencontrerons après cette introduction et pour la légèreté notationnelle, nous ne ferons plus référence à la paramétrisation. De plus, nous confondrons souvent l'instance du problème avec le graphe G qu'elle contient (en omettant k).

Enfin, dans les problèmes que nous considérons, une solution S est un ensemble de sommets (non pondérés) du graphe (sauf dans la section 3.5, où une solution est un ensemble de sous-graphes), et donc, nous utilisons toujours la cardinalité des ensembles comme mesure $m^\Pi : S \mapsto |S|$.

Nos résultats ne sont valables que dans certaines classes de graphes (les classes excluant des mineurs). Pour cette raison, nous considérerons des restrictions d'un problème à une classe de graphes. Il s'agit simplement d'une

limitation du domaine de définition des instances. Un autre point de vue consiste à rendre négatives toutes les instances dont le graphe est hors de la classe.

Définition 30 : restriction à $\mathcal{G}$

Étant donné un problème Π et une classe de graphe $\mathcal{G}$, le problème $\Pi_{\mathcal{G}}$ *restreint* à $\mathcal{G}$ est l'ensemble $\Pi_{\mathcal{G}} = \{(G, k) \mid (G, k) \in \Pi, G \in \mathcal{G}\}$.

⌟

Nous l'avons déjà mentionné, nous traitons dans cette thèse de méta-algorithmes, c'est-à-dire des algorithmes qui permettent de résoudre un ensemble de problèmes. Un point de vue un peu plus formel consiste à voir un méta-algorithme comme un algorithme dont l'entrée est composée de la description d'un problème et d'une instance de ce problème. Bien sûr, la théorie de la calculabilité interdit que le problème de l'entrée soit quelconque. Il nous faut donc nous restreindre à des familles de problèmes raisonnables.

Une solution pour contraindre la famille de problèmes étudiée consiste à utiliser la logique. Rappelons que la logique d'ordre zéro considère les formules construites à partir de variables propositionnelles et des connecteurs logiques (conjonction, disjonction, implication, et négation) ; la logique du premier ordre substitue aux variables propositionnelles les variables symboliques et les prédicats et ajoute les quantificateurs ; la logique du second ordre ajoute en sus les variables relationnelles. Intuitivement, les variables symboliques représentent les objets mathématiques manipulés (dans notre cas, il s'agit des sommets et arêtes d'un graphe), et les prédicats représentent les propriétés et les relations sur ces objets. Les variables relationnelles représentent des prédicats. Autrement dit, la logique du second ordre permet de considérer les prédicats (du premier ordre) comme des variables, et donc un prédicat peut être quantifié ou donné comme argument d'un prédicat (du second ordre). La logique la plus souvent utilisée pour énoncer des méta-théorèmes est une logique intermédiaire entre le premier et le second ordre : il s'agit de la logique monadique du second ordre. Cette logique monadique permet seulement l'usage de variables relationnelles unaires lesquelles représentent des propriétés ou des ensembles d'objets. En d'autres termes, la logique monadique du second ordre permet de quantifier et d'énoncer des propriétés sur les ensembles (par exemple, les ensembles de sommets), mais pas sur les relations d'arité deux ou plus.

Remarquons que dans le chapitre 3 ; bien que nous présentions un méta-algorithme, nous n'aurons pas recours à la logique. En effet, nous contraignons la forme des problèmes étudiés en posant la condition que ces problèmes puissent être résolus par programmation dynamique. D'une certaine façon

cette contrainte est plus intuitive, mais plus difficile à formaliser, qu'une contrainte logique sur le langage du problème.

1.3.2 Complexité classique

L'algorithmie élabore des algorithmes pour résoudre des problèmes. Elle analyse ensuite leur complexité. La complexité d'un algorithme donne une indication sur la difficulté du problème qu'il résout. La théorie de la complexité s'intéresse à l'appartenance d'un problème (un problème de décision) aux diverses classes de complexité. Pour montrer qu'un problème est facile, c'est-à-dire qu'il appartient à une certaine classe de complexité, il suffit d'exhiber un algorithme dont la complexité est plus faible que celle de la classe. À l'inverse montrer qu'un problème est difficile, c'est-à-dire qu'il n'appartient pas à une classe, est chose moins aisé puisqu'il faut montrer qu'il n'existe pas d'algorithme résolvant ce problème avec la bonne complexité. C'est pour démontrer de tels résultats négatifs que la théorique de la complexité a développé ses outils propres. En fait, ces outils permettent de montrer qu'un problème est au moins aussi dur que tout les autres problèmes d'une classe.

La classe P est la classe des problèmes résolubles en temps polynomial. Elle est considérée comme la classe des problèmes faciles (pour la plupart des branches de la complexité). En effet, les polynômes forment une famille de fonctions qui croit relativement lentement ; et en général, il n'est pas raisonnable d'espérer obtenir des algorithmes sous-linaires puisqu'il faut compter au moins le temps de lire l'entrée. De plus cette classe est stable au sens où la combinaison de plusieurs algorithmes polynomiaux reste polynomiale. Tous les problèmes n'admettent pas nécessairement un algorithme qui les résout en temps polynomiale.

La classe Exp est la classe des problèmes résolubles en temps exponentiel. Elle est la classe contenant trivialement tous les problèmes que nous considérerons dans cette thèse. Pour une petite entrée, un algorithme polynomial peut être plus lent qu'un algorithme exponentiel, mais il existe toujours une taille d'entrée au-delà de laquelle l'algorithme polynomial est plus efficace.

Définition 31 : classes P et Exp

Un algorithme est *polynomial* si sa complexité est de la forme $O(|G|^d)$. La classe P est l'ensemble des problèmes résolus par un algorithme polynomial.

Un algorithme est *exponentiel* si sa complexité est de la forme $O(2^{|G|^d})$. La classe Exp est l'ensemble des problèmes résolus par un algorithme exponentiel.

⌟

L'inclusion de P dans Exp est claire, puisque tout polynôme est asymptotiquement dominé par n'importe quelle exponentielle. De plus il est prouvé que cette inclusion est stricte. Nous avons donc $\mathsf{P} \subset \mathsf{Exp}$.

La classe NP est la classe des problèmes vérifiables en temps polynomial. C'est une classe intermédiaire entre P et Exp. De façon équivalente, c'est aussi la classe des problèmes admettant un algorithme non déterministe polynomial.

Définition 32 : classe NP
La classe NP est l'ensemble des problèmes pouvant être vérifiés par un algorithme polynomial.
⌟

Encore une fois l'inclusion de P dans NP est facilement démontrable. Savoir si cette inclusion est stricte ou non constitue le problème ouvert $\mathsf{P} \stackrel{?}{=} \mathsf{NP}$. De plus, l'inclusion de NP dans Exp est également facile. Nous avons donc $\mathsf{P} \subseteq \mathsf{NP} \subseteq \mathsf{Exp}$.

Pour montrer qu'un problème est difficile par rapport à une classe il faut montrer qu'il est au moins aussi difficile que tous les problèmes de cette classe. Cette notion de difficulté d'un problème par rapport à un autre est formalisée par la notion de réduction. Une réduction est une transformation d'un problème Ξ en un autre problème Π. Intuitivement, une réduction est une méthode qui permet de simuler Ξ dans Π ; obtenir un solution pour Π permet donc a avoir un solution pour Ξ.

Définition 33 : réduction
Une *réduction* d'un problème Ξ vers un problème Π est une fonction $\mathcal{R} : \Xi \to \Pi$, calculable en temps polynomial et telle que, pour toute instance (G, k) de Ξ, $(G, k) \in \Xi$ si et seulement si $\mathcal{R}(G, k) \in \Pi$.
⌟

Remarquons que la définition formelle correspond bien à l'intuition. Le problème Ξ se réduit à Π, si Π est plus difficile que Ξ, au sens où il suffit de résoudre Π pour avoir la réponse a Ξ. En effet, tout algorithme de complexité polynomiale (ou plus) pour Π implique un algorithme de même complexité pour Ξ ; il suffit pour cela de composer l'algorithme avec la réduction. Puisque les instances sont équivalentes (au sens où trouver une solution pour l'une implique une solution pour l'autre, et inversement) l'algorithme pour Π donne la bonne réponse pour Ξ. Cela signifie que Π est plus expressif, qu'il est possible de coder Ξ dans Π, et que la transformation se fait sans perte d'information.

Remarquons que Π pourrait être plus dur et non équivalent à Ξ car il n'y a pas nécessairement bijection entre les instances des deux problèmes (certaines instances de Π n'ont pas d'instance correspondante dans Ξ). Autrement dit, un réduction vers Π ne donne pas d'information sur la classe de Π. Or l'information qu'un problème est plus difficile qu'une classe n'est pas toujours pertinente. Par exemple, nous savons que $\mathsf{P} \neq \mathsf{Exp}$ donc trivialement certains problème de Exp sont P-difficiles. Il convient donc de distinguer les problèmes plus difficiles qu'une classe et les problèmes difficiles au sein du classe. Ces dernier sont appelés problèmes complets.

Définition 34 : problème difficile, complet

Soit X une classe de complexité.

Un problème Π est X-*difficile* s'il existe une réduction depuis n'importe quel problème $\Xi \in \mathsf{X}$ vers Π.

Un problème Π est X-*complet* si Π est X-difficile et Π est dans X.

⌟

Entre autre, une réduction permet de montrer la NP-difficulté d'un problème en réduisant à lui un problème NP-complet déjà connu. Par exemple, le théorème de Cook [13] montre que le problème 3-Sat (problème de satisfiabilité d'une formule logique sous forme conjonctive avec trois littéraux par clause) est NP-complet. Ce qui signifie que, parmi les problèmes dans NP, 3-Sat est un problème plus difficile que tous les autres problèmes de NP, au sens où, s'il est possible de trouver une solution à 3-Sat en temps polynomial, alors il est possible de trouver une solution à tout problème de NP dans le même temps. L'ensemble des problèmes équivalents (en terme de difficulté) à 3-Sat forme donc la classe des problèmes NP-complets. Pour montrer qu'un problème est NP-difficile, il suffit de trouver une réduction depuis 3-Sat.

La théorie de la complexité à mis en évidence plusieurs indices qui laissent penser que la hiérarchie des classes ne peut pas s'effondrer ; c'est-à-dire qu'il est peu probable que deux classes de complexité (parmi celles connues actuellement) n'en forment en réalité qu'une seule. Il ne s'agit que d'un pressentiment, il n'y a pas de preuve connue à ce jour.

En particulier l'hypothèse que $\mathsf{P} \neq \mathsf{NP}$ est communément admise. Cela signifie que si problème est NP-difficile alors il y a peu d'espoir qu'il admette un algorithme polynomial.

Dans le même ordre d'idée, une autre conjecture fréquemment admise est l'hypothèse du temps exponentiel qui énonce que 3-Sat n'admet pas d'algorithme de complexité $2^{o(n)}$ (où n est le nombre de variables). Cela signifie que le problème 3-Sat ne peut pas être résolu en temps sous-exponentiel (par

exemple en $2^{\sqrt{n}}$). Donc, l'hypothèse du temps exponentiel implique $\mathsf{P} \neq \mathsf{NP}$; et par conséquent, un théorème démontré sous l'hypothèse du temps exponentiel n'est pas nécessairement vrai si $\mathsf{P} \neq \mathsf{NP}$ est démontré.

1.3.3 Extraction de noyaux

Définissons les noyaux proprement dit. Intuitivement, extraire un noyau consiste à transformer une instance d'un problème en une autre instance du même problème, mais de taille plus petite, et sur laquelle nous pourrons calculer (plus rapidement) une solution. Cette transformation est une sorte de pré-calcul, mais nous voulons des garanties théoriques.

La première des garanties que nous voulons assurer est que la transformation ne changera pas la réponse au problème. Pour cela nous réutilisons la notion de réduction polynomiale. La seconde garantie est que l'instance résultant de la transformation soit suffisamment petite pour que que la solution puisse être trouvée rapidement.

Une extraction de noyau est donc une réduction particulière depuis un problème vers lui même qui garantit que le résultat n'est fonction que du paramètre (cf. figure 1.6).

Définition 35 : extraction de noyaux

Une *extraction de noyaux* d'un problème Π est une réduction $\mathcal{K} : \Pi \to \Pi$ telle que pour toute instance (G, k) :

— $(G, k) \in \Pi$ si et seulement si $\mathcal{K}(G, k) \in \Pi$;

— *et* $|\mathcal{K}(G, k)| \leqslant g(k)$ avec g une fonction calculable.

Le *noyau* de (G, k) est l'instance réduite $\mathcal{K}(G, k)$.

⌟

Rappelons nous que nous voulons pouvoir résoudre le problème Π dans le noyau. Pour faire cela efficacement il importe donc que le noyau soit petit. La taille du noyau est définie par la fonction g, qui peut être considérée comme le taux de compression du problème. Tous les problèmes n'admettent pas des noyaux de même taille ; l'amélioration de la taille du noyau est donc un point essentiel.

Définition 36 : noyau linéaire (polynomial)

Un noyau est *linéaire* si la fonction g est linéaire.

Un noyau est *polynomial* si la fonction g est polynomiale.

⌟

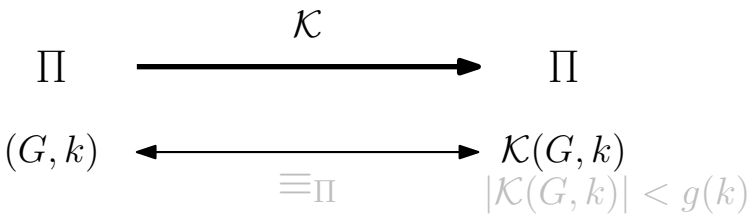


FIGURE 1.6 – définition schématique d'un noyau (cf. définition 35).

Formulons quelques remarques à propos de ces définitions. Si $\mathcal{K}(G,k) = (G',k')$, la condition $|\mathcal{K}(G,k)| \leqslant g(k)$ implique que la taille de G' et k' sont toutes deux bornées par une fonction de k. Borner la taille de G' était notre objectif, mais il est tout aussi important de borner k' sans quoi nous pourrions utiliser des arguments de théorie de l'information pour encoder l'instance initiale dans k'. Très souvent, la littérature impose en sus que $k' \leqslant k$.

Signalons que la définition de l'extraction de noyaux est relativement contrainte. Il existe plusieurs relaxations de cette définition. Entre autres, notons le binoyau dans la définition duquel la réduction ne se fait pas vers le problème lui même. Un binoyau donne la possibilité de faire la réduction vers un problème qui, intuitivement, similaire au problème initiale ; il permet aussi de relâcher les contraintes sur la classe des graphes instances.

Par la suite, il est possible que nous confondions les notions de noyau et d'extraction de noyaux.

En pratique, une extraction de noyau n'est pas présentée comme un algorithme, mais comme un ensemble de règles de réduction qui seront répétées itérativement. Une règle de réduction est un algorithme (souvent court) qui s'applique si une certaine configuration locale est trouvée dans le graphe. Par exemple, une règle peut supprimer des sommets qui n'apportent pas d'information pertinente pour la résolution du problème (en cela qu'ils n'ajoutent pas de contrainte, ne créent pas de choix, ne modifient pas l'ensemble des possibilités, pour construire les solutions). En général, la preuve de construction d'un noyau se fait en deux étapes : premièrement, la présentation des règles de réduction, avec les preuves de validité et de complexité ; deuxièmement, la démonstration d'une borne en fonction de k sur le graphe réduit. Observons que dans le chapitre 3, nos preuves ne suivent pas ce schéma ; en fait, nous prouvons seulement la validité de notre règle, car la borne nous est fournie par des résultats d'autres chercheurs.

La preuve de validité d'une règle consiste à montrer que l'instance initiale (G,k) et l'instance réduite (G',k') sont équivalentes, c'est-à-dire qu'elles

sont simultanément positives ou similairement négative. Pour ce faire, nous considérons une solution S de G' à partir de laquelle nous construisons une solution S' de G' (ce qui prouve que $(G, k) \in \Pi$ implique $(G', k') \in \Pi$), puis nous procédons inversement (ce qui prouve la réciproque). Le plus souvent, une règle effectue une modification locale du graphe ; il nous suffit donc de modifier la solution S et que vérifier localement que S' satisfait toutes les contraintes. Dans la plupart des cas, pour faire ces démonstrations, nous admettons une hypothèse sur la forme de la solution de sorte que celle-ci corresponde mieux à l'idée de la règle ; toutes les solutions ne vérifient pas forcement cette hypothèse, nous pouvons tout de même la formuler car, nous montrons que si G admet un solution, alors il en admet une vérifiant l'hypothèse. La preuve de complexité est plus facile : il suffit de montrer que chaque règle s'exécute en temps polynomial, en général, il est obvie que l'itération des règles est aussi polynomiale.

La démonstration de la borne est la partie la plus difficile, pour laquelle il n'y a pas vraiment de méthode générale. Pour trouver la borne, il faut utiliser les propriétés imposées par les règles sur le graphe réduit. Il est aussi possible de supposer que le graphe est une instance positive (respectivement, négative), ce qui nous donne plus d'informations sur la structure de celui-ci. Il est correct de faire cette supposition puisque si l'existence d'une solution permet d'exhiber une borne alors (par contraposée) tout graphe trop grand est nécessairement une instance négative (cela ne signifie pas que tous les graphes de taille inférieure à la borne sont des instances positives).

Entre les deux étapes de la preuve nous manipulons un graphe sur lequel les règles ont été appliquées, mais qui ne peut pas encore être appelé noyau, puisque sa taille n'a pas encore été bornée. Selon la manière de formuler les règles, il y a deux façons de formuler la condition d'arrêt de l'itération de règles : ou bien lorsque les règles ne peuvent plus s'appliquer (une condition n'est plus remplie par le graphe) ou bien lorsque les règles ne modifient plus le graphe. Pour nos noyaux (dans le chapitre 2, en particulier pour la règle 6) nous considérons cette seconde définition.

Définition 37 : graphe reduit

Un graphe est *réduit* selon une règle si l'application de cette règle ne modifie pas le graphe, à isomorphisme près.

⌟

L'intérêt principal des noyaux est qu'ils fournissent un algorithme à paramètre fixé, c'est-à-dire un algorithme efficace en terme de complexité paramétrée. En effet, après avoir fait l'extraction de noyau, nous pouvons résoudre le problème dans le noyau avec n'importe quel algorithme, y compris

l'énumération exhaustive, car sa taille est fonction du paramètre uniquement (par exemple, si l'énumération exhaustive se fait en $2^{|G|}$, nous obtenons un algorithme en $p(|G|) + 2^{g(k)}$. Cependant la méthode de l'extraction de noyau peut être adaptée à d'autres domaines, comme l'approximation [56].

Remarquons aussi que, presque toujours, les noyaux et les algorithmes paramétrés servent à attaquer des problèmes NP-difficiles ; pourtant il serait parfaitement envisageable d'avoir recours au noyaux pour améliorer en pratique le temps de résolution d'un problème polynomial dont le degré est élevé.

1.3.4 Complexité paramétrée

Comme il en a été discuté dans notre prologue, l'algorithmie paramétrée utilise le paramètre comme une seconde façon de mesurer l'instance afin de faire une analyse plus fine de la complexité des problèmes paramétrés. C'est une approche utilisée pour essayer de résoudre efficacement les problèmes réputés difficiles (au sens NP-difficiles) [19, 24, 17].

Dans ce paradigme, un algorithme est considéré comme efficace si sa complexité est composée d'une fonction calculable quelconque qui dépend du paramétré et d'un polynôme qui dépend de la taille de l'instance. Ces algorithmes sont dits à paramètre fixé.

Définition 38 : classe FPT

Un algorithme *à paramètre fixé* est un algorithme dont la complexité est de la forme $f(k) \cdot p(|G|)$ où f est un fonction calculable et p est un polynôme.

La classe FPT est l'ensemble des problèmes admettant un algorithme à paramètre fixé.

⌟

De façon équivalente les algorithmes à paramètre fixé peuvent être décrit comme ayant une complexité de la forme $f(k) + p(|G|)$.

Nous tenons à souligner que, bien que nous considérions qu'une seule paramétrisation $\kappa : (G, k) \mapsto k$, de nombreuses autres paramétrisations existent. Par exemple, il est possible de paramétrer un problème par le nombre d'arêtes du graphe, par la taille d'un sous ensemble de sommets particuliers dans l'instance, ou par un paramètre structurel (comme la largeur arborescente). Le choix de ce paramètre peut être une étape cruciale pour celui qui cherche à résoudre un problème efficacement ; certaines paramétrisations n'apportent pas d'information pertinente. Par exemple, un

problème paramétré par $\kappa : G \mapsto |G|$ est dans FPT s'il est calculable. Similairement, un problème paramétré par $\kappa : G \mapsto 1$ est dans FPT si et seulement s'il est dans P.

Une des motivations principales de cette approche est l'espoir que le paramètre k soit en pratique petit devant la taille de l'instance, ce qui améliorerait le temps de calcul. C'est le cas en algorithmie du texte (où l'alphabet est petit devant le texte) ou dans la fouille de données (où les requêtes sont petites devant la base de données).

Cependant, dans de nombreux cas, aucun indice ne permet de supposer que le paramètre sera petit en pratique. Dans d'autres circonstances, c'est la fonction f qui rend impraticable l'algorithme obtenu. D'un point de vue théorique, l'analyse paramétrée reste néanmoins intéressante. Elle permet de décomposer la complexité en une composante exponentielle (ou pire) et une composante polynomiale. Elle permet également d'identifier quelle est la difficulté du problème, l'origine de l'explosion combinatoire. Bref, elle permet un raffinement de la granularité dans l'analyse de complexité.

Remarquons que si nous considérons un problème paramétré comme une famille de sous-problèmes (dont l'énoncé dépend du paramètre), alors montrer que le problème paramétré est dans FPT implique que chaque sous-problème est dans P. Autrement dit, si le paramètre est fixé, nous obtenons un algorithme polynomial. Notons que la réciproque n'est pas vraie, car pour qu'un problème soit dans FPT, il faut que le degré du polynôme soit indépendant du paramètre. Un problème résolut au mieux en temps $|G|^{f(k)}$ n'est pas dans FPT. L'ensemble des problèmes ayant cette complexité forme la classe XP (qui est le pendant paramétré à la classe Exp).

Comme nous l'avons dit, une extraction de noyau fournit un algorithme à paramètre fixé (il suffit de résoudre le problème dans le noyau). Il est important de noter que la réciproque est vraie. Cela signifie que les problèmes qui admettent un noyau sont exactement ceux qui admettent un algorithme à paramètre fixé. Nous pouvons donc caractériser la classe FPT à partir des noyaux.

Théorème : [24]
Un problème Π est dans FPT si et seulement si il est calculable et admet un noyau.
⌞ ⌟

Remarquons qu'une borne sur la taille du noyau fournit une borne sur la complexité paramétrée du problème, mais que l'inverse est faux. En cela, dans une certaine mesure, obtenir un noyau est plus intéressant qu'un algorithme à paramètre fixé.

De même que, en complexité classique, il est possible de prouver que certains problèmes ne peuvent pas être résolus en temps polynomial (ou du moins, ne peuvent pas l'être sous l'hypothèse $\mathsf{P} \neq \mathsf{NP}$), de même certains problèmes paramétrés ne sont pas dans FPT. En complexité classique, c'est la classe NP qui sert d'indicateur de difficulté. En complexité paramétrée, la classe ParaNP des problèmes admettant une algorithme non déterministe à paramètre fixé ne saurait jouer le même rôle ; en effet, ParaNP n'est pas comparable avec XP. Cependant, dans leur intersection, il existe une hiérarchie de classes $\mathsf{W}[i]$ (avec $i \in \mathbb{N}$) dont l'union forme la classe appelé W[P] (cf. figure 1.7). C'est la classe W[1] qui nous sert d'indicateur de difficulté, autrement dit, un problème W[1]-complet n'est pas dans FPT (sauf contradiction de l'hypothèse du temps exponentiel, une conjecture communément admise).

De la même façon, il est aussi possible d'obtenir des résultats négatifs sur les noyaux. D'après la discussion précédente, très clairement tout problème W[1]-complet n'admet pas de noyaux. Cependant, il est possible d'être plus précis : des réductions particulières appelées OU-composition et ET-composition [7, 9] permettent de démontrer qu'un problème n'admet pas de noyau polynomial (sauf contradiction de la conjecture $\mathsf{coNP} \not\subseteq \mathsf{NP/poly}$).

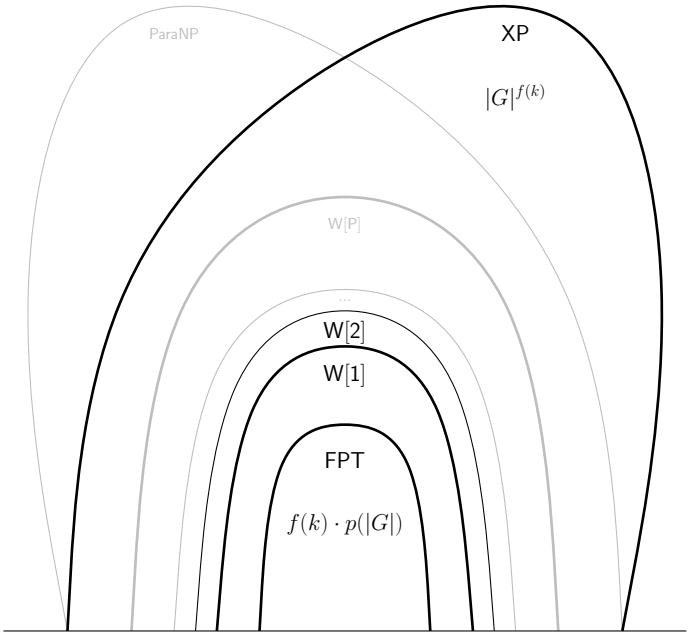


FIGURE 1.7 – Hiérarchie des classes paramétrées (cf. définition 38).

1.4 Contexte scientifique

1.4.1 Résultats antérieurs

Dans notre avant-propos, nous avons souligné que l'algorithme et le problème se caractérisent par leur prétention à généraliser, respectivement, le calcul et la question. Somme toute, il ne s'agit que d'une motivation traditionnelle de la science : énoncer des règles génériques pour décrire un ensemble de phénomènes aussi large que possible. C'est donc dans l'ordre des choses que de chercher à généraliser plus encore et de vouloir considérer en un seul bloc tout un ensemble de problème. Pour des raisons de calculabilité, il faut contraindre la forme sous laquelle peut être exprimée la question de cette famille de problèmes. Il est néanmoins possible d'énoncer des résultats de ce genre. Pour résoudre un méta-problème, il faut concevoir un unique méta-algorithme qui soit en mesure de donner une réponse à chacun des problèmes de la famille.

À force de généralisation, la littérature compte ne nombreux résultats qui peuvent être considérés comme sous cette angle. Par exemple, le théorème de Robertson et Seymour [53] peut être interprété comme un méta-résultat algorithmique puisqu'il garanti que la reconnaissance d'un classe close par mineur se fait en temps polynomial. Remarquons que ce résultat est non constructif puisque l'algorithme de reconnaissance doit utiliser la liste (finie) des mineurs exclus, laquelle n'est pas connue explicitement.

Dans le domaine des algorithmes à paramètre fixé, le théorème de Courcelle [14] est un résultat fondamental, qui démontre l'existence d'un méta-algorithme. Plus précisément, le théorème de Courcelle énonce que toute propriété (de graphe) exprimable en logique monadique du second ordre peut être décidée en temps $f(\mathrm{tw}(G)) \cdot O(|G|)$. En terme de complexité paramétrée cela signifie que tout problème paramétré par la largeur arborescente ($\kappa = \mathrm{tw}$) et exprimable en logique monadique du second ordre est FPT. Bien sûr, la généralité d'un tel méta-algorithme est contrebalancée par l'efficacité de celui-ci : la fonction f est une tour exponentielle dont la hauteur dépend du nombre de quantificateurs dans la formule logique. Il a été montré que nous ne pouvons pas espérer une meilleure complexité pour un résultat aussi générique. Plusieurs autres méta-théorèmes ont fait suite à celui de Courcelles [38, 45]. Entre autre il existe des équivalents du théorème de Courcelles pour d'autres largeurs structurelles. Les problèmes exprimables en logique du premier ordre admettent des algorithmes linéaires et des schémas d'approximation polynomiaux efficaces dans certaines classes de graphes peu denses. Le plus souvent ses résultats peuvent être réinterprétés comme des

algorithmes paramétrés.

Dans le domaine des noyaux, la question des méta-algorithmes d'extraction de noyaux s'est également posée ; cette fois-ci pour les problèmes paramétrés par la taille de la solution ($\kappa : (G, k) \mapsto k$).

Le premier pas vers les méta-noyaux est dû à Guo et Niedermeier [40] qui ont essayé de généraliser la méthode qu'Alber, Fellows et Niedermeier [3] ont mise au point pour le problème Domination dans les graphes planaires. En effet, et nous en rediscuterons dans le chapitre 2, la construction du noyau pour Domination se base essentiellement sur le fait que les graphes sont planaires et sur la propriété que tous les sommets sont à distance bornée d'une solution. Il est donc raisonnable de penser que cette méthode peut être appliquée à tout problème vérifiant cette propriété de distance. La méthode d'Alber, Fellows et Niedermeier [3] se base sur un objet appelé région qui permet de décomposer le graphe en zones relativement indépendantes les unes des autres. D'abord, la règle de réduction consiste, intuitivement, à remplacer une région par une région équivalente (au sens où elle a le même comportement vis-à-vis du problème) ; la validité de cette règle est démontrée grâce à la propriété de distance. Ensuite, la borne du noyau se démontre en décomposant le graphe en régions ; grâce à la planarité, la décomposition contient un petit nombre de régions. Malheureusement, comme nous en discuterons plus en détail dans la section 2.1, les preuves de ces deux résultats sont, en partie, erronées, et il n'y a pas de versions corrigées publiées à ce jour.

L'intérêt porté à une tentative de correction a été amoindri par la publication du résultat de Bodlaender, Fomin, Lokshtanov, Penninkx, Saurabh et Thilikos [8], qui généralise encore la méthode et offre un cadre plus générique, quoique non constructif. Ce résultat comporte deux théorèmes principaux qui énoncent l'existence d'un noyau pour des problèmes restreints aux graphes de genre borné. Le premier théorème garanti l'existence d'un noyau polynomial pour les problèmes exprimables en logique monadique du second ordre qui vérifient la propriété de recouvrabilité. Le second théorème garanti l'existence d'un noyau linéaire pour les problèmes qui vérifient les propriétés de quasi-recouvrabilité et d'indexation entière finie (cf. chapitre 3 ; par la suite, c'est surtout ce second théorème qui nous intéresse). Dans l'idée, l'évolution principale consiste à remplacer la notion de région par celle de protrusion. La réduction consiste à remplacer une protrusion par une protrusion équivalente ; l'index entier fini signifie précisément que le nombre de classes d'équivalence est fini, et donc que les tailles des plus petits représentants de chaque classe sont bornées par une constante. La décomposition en protrusions (avec un nombre linéaire de protrusions) est construite grâce à la propriété de recou-

vrabilité et au genre borné. La décomposition, avec un petit de nombre de protrusions remplacées par des petits représentants, donne la borne (existentielle) sur la taille du noyau. La faiblesse principale de ce résultat est qu'il n'exhibe ni borne explicite sur la taille du noyau, ni domination asymptotique de la complexité, ni l'algorithme d'extraction ; en un mot, c'est un résultat non constructif. Le défaut de constructivité se situe lors du remplacement de protrusion ; en effet, l'index entier fini permet d'assurer l'existence d'une borne sur les représentants sans la donner explicitement (ce qui permettrait de construire les représentants et de calculer la taille du noyau).

Ce résultat fondamental à été amélioré plusieurs fois. Une première fois par Fomin, Lokshtanov, Saurabh et Thilikos [27] qui l'étendent aux graphes sans mineur (en utilisant les propriétés bidimensionnalité et de séparabilité). Une deuxième fois par Kim, Langer, Paul, Reidl, Rossmanith, Sau et Sikdar [43] qui l'étendent aux graphes sans mineur topologique (en se ramenant à une largeur arborescente bornée). Ces deux extensions se sont uniquement intéressées à élargir les classes de graphes sur lesquelles il est possible de construire la décomposition en protrusions. Mais ils ont conservé tel quel le mécanisme de remplacement de protrusions. En cela ils éludent la question de la constructivité et demeurent, comme leur prédécesseur des résultats purement existentiels.

1.4.2 Travaux de thèse

Les résultats présentés dans cette thèse se placent dans la lignée de ceux que nous venons de mentionner, avec pour objectif de rester aussi constructifs que possible.

Dans le chapitre 2, nous présentons deux noyaux qui succèdent directement à l'article d'Alber, Fellows et Niedermeier [3]. Il s'agit des premières démonstrations de noyaux (linéaires) dans les graphes planaires pour les problèmes Domination Rouge-Bleu et Domination Totale qui sont réputés durs (W[2]-durs en général et NP-complets dans les planaires). Ces résultats sont constructifs : nous explicitons les algorithmes d'extraction ainsi que les bornes sur les tailles. De plus les bornes sont petites en comparaison à celles qui pourraient être obtenues par un méta-théorème.

Comme nous l'avons dit, la méthode d'Alber, Fellows et Niedermeier consiste à décomposer le graphe en régions pour borner, premièrement le nombre de régions, deuxièmement le nombre de sommets hors de la décomposition, troisièmement la taille de chaque région. Cependant la définition de décomposition en régions proposée par Alber, Fellows et Niedermeier ne permet

pas d'établir la borne sur le nombre de régions (cf. figure 2.4). Il s'en suit, nécessairement, que les preuves permettant d'établir la borne sont incomplètes. Nous proposons donc un définition alternative de décomposition en régions (cf. définition 42) qui utilise la notion de confluence (cf. définition 39). De là, nous pouvons établir des preuves correctes pour compter les régions (cf. lemme 3 et lemme 4). De cette façon, nous corrigeons le résultat d'Alber, Fellows et Niedermeier. Nos définitions et lemmes permettent de fournir une preuve rigoureuse du noyau pour le problème DOMINATION ; ainsi que de toutes les extractions de noyaux qui s'appuient sur la même méthode (sans en corriger les preuves).

De plus, nous proposons un nouvelle définition de région (cf. définition 40), et introduisons la notion de dominant partiel (cf. définitions 51 et 55). Nous faisons ainsi un première pas vers une généralisation de la méthode (sans pour autant fournir un cadre générique). Nous avons bon espoir que cette nouvelle formulation puisse servir de base solide pour compléter les travaux de Guo et Niedermeier.

La correction de la méthode de décomposition en régions, ainsi qu'une formulation correcte de la généralisation du cadre étaient deux taches importantes pour la communauté et laissées inachevées. Par le chapitre 2 de notre thèse, nous achevons la première et ouvrons une piste pour la seconde.

Dans le chapitre 3, nous proposons un cadre générique qui s'attache à rendre constructif les méta-théorèmes sur les noyaux [8, 27, 43].

Rappelons nous que le cadre des méta-noyaux est une généralisation de la méthode des régions, et par conséquent, nous retrouvons le même schéma : le graphe est décomposé en protrusions de sorte que le nombre de protrusions, le nombre de sommets hors des protrusions, et le nombre de sommets dans chaque protrusion soient bornés. Une différence avec la méthode des régions réside dans le fait que, ici, pour appliquer la règle de réduction il est nécessaire de connaître une protrusion de la décomposition. Les deux articles [27, 43] qui généralisent le cadre original [8] s'intéressent à la construction d'une décomposition. Pour notre part, nous nous intéressons à rendre constructif le remplacement des protrusions (c'est-à-dire, la règle de réduction). Pour cela, nous nous inspirons de la programmation dynamique sur décomposition arborescente : nous définissons une machinerie qui permet de simuler la programmation dynamique de façon relativement intuitive ; nous utilisons ensuite l'information calculée par le programme dynamique pour identifier les protrusions équivalentes. Plus précisent, la programmation dynamique nous permet de calculer les classes d'équivalence et d'exhiber une borne sur la taille des représentants.

Signalons que les méta-théorèmes sur les noyaux ne sont pas construc-

tifs en plusieurs points : certaines étapes de l'algorithme d'extraction ne sont pas constructives, la complexité de celui-ci n'est pas explicite, et la borne sur la taille des noyaux n'est pas explicite non plus. En ce qui nous concerne, nous exhibons toujours l'algorithme d'extraction, et sa complexité peut être explicitée ; mais selon le résultat que nous utilisons pour construire la décomposition en protrusions nous ne pouvons pas toujours donner une borne explicite sur la taille du noyau (cf. section 3.5).

Nous appliquons notre cadre à trois problèmes assez généraux : r-Domination, r-Indépendance, et $\mathcal{F}$-Paquetage. Il s'agit des premiers noyaux constructifs dans les graphes peu denses (bien que l'existence du noyau soit déjà connu pour certains [8, 27, 43]) pour ces problèmes réputés durs ($\mathsf{W}[1]$-durs en général et NP-complets dans les planaires). Nous nous servons des outils déjà existants pour la construction de décomposition en protrusions.

Les résultats obtenus au cours de notre préparation de doctorat ont donné lieu à la parution de plusieurs articles dont certains ne sont pas présentés dans cette thèse.

[35] V. Garnero, I. Sau, et D. M. Thilikos.
A linear kernel for planar red-blue dominating set.
Proceedings of the 12th Cologne Twente Workshop on Graphs and Combinatorial Optimization (CTW), p. 117–120, 2013.
Soumis pour publication en journal.

[34] V. Garnero, et I. Sau.
A Linear Kernel for Planar Total Dominating Set.
Soumis pour publication en journal.

[33] V. Garnero, I. Sau, C. Paul, et D. M. Thilikos.
Explicit linear kernels via dynamic programming.
Proceedings of the 31st International Symposium on Theoretical Aspects of Computer Science (STACS), vol. 25 of LIPIcs, p. 312–324, 2014.
SIAM Journal on Discrete Mathematics, vol. 29(4), p. 1864–1894, 2015.

[32] V. Garnero, I. Sau, C. Paul, et D. M. Thilikos.
Explicit Linear Kernels for Packing Problems.
Soumis en conférence.

[36] V. Garnero, et M. Weller.
Parameterized Certificate Dispersal and its Variants.
Theorical Computer Science, vol. 622, p. 66–78, 2016.

[10] A. Braga de Queiroz, V. Garnero, et P. Ochem.
On interval representations of graphs.
Discrete Applied Mathematics, vol. 202, p. 30–36, 2016.

Chapitre 2

Noyaux *ad hoc*

Ainsi que nous l'avons mentionné, le noyau linéaire pour Domination (paramétré par la taille de la solution) dans les graphes planaires [3] est un résultat majeur dans le domaine de l'extraction de noyaux. Ce résultat, d'Alber, Fellows et Niedermeier, construit un noyau borné par 335 fois le paramètre ; la constant multiplicative a par la suite été améliorée à 67 [12]. Alber, Fellows et Niedermeier procèdent en deux étapes : premièrement, ils fournissent une réduction polynomiale du problème ; deuxièmement, ils prouvent que si l'instance réduite admet une solution alors elle est de taille linéaire en le paramètre. La première étape consiste à montrer que le voisinage d'un sommet ou d'une paire de sommets peut être remplacé par un petit ensemble de sommets sans changer le comportement du graphe par rapport au problème (c'est-à-dire, sans changer la taille d'un dominant minimum). La deuxième étape consiste à montrer que, étant donné un dominant du graphe réduit, il existe une décomposition en régions (cf. définition 42) telle que, en fonction de la taille du dominant :

— elle contienne un nombre linéaire de régions,

— elle couvre tous les sommets sauf un nombre linéaire,

— chacune de ses régions contient un nombre constant de sommets.

Enfin, étant donné un graphe, son noyau est obtenu de la façon suivante : soit le graphe réduit est trop grand par rapport à la taille de solution cherchée et alors le graphe initial n'admet pas de solution (dans ce cas, son noyau est une instance négative arbitraire), soit le graphe réduit est assez petit et alors c'est un noyau.

La méthode de décomposition en régions a permis de construire des noyaux pour de nombreuses variantes de Domination telles que : Domination par Arête [40, 55], Domination Efficace [40], Domination Connexe [39, 47], et Domination Double [5] ; enfin, Guo et Nieder-

meier [40] ont cherché à généraliser la méthode pour la famille de problèmes vérifiant une certaine propriété de distance. Nous l'avons vu, d'autres généralisations ont été faites par la suite [8, 27, 43], pour des familles de problèmes plus grandes, et sur des classes de graphes plus larges. Cependant, ces résultats étant extrêmement généraux, ils ne peuvent avoir la précision d'un noyau conçu pour un problème spécifique. Ils assurent l'existence de noyaux linéaires, mais avec une constante multiplicative grande , voire non explicite. Pour cette raison, nous avons jugé intéressant de reprendre la méthode de décomposition en régions (plutôt que d'utiliser les théorèmes généraux) afin d'élaborer nos deux noyaux.

De plus, l'article d'Alber, Fellows et Niedermeier comporte certaines imprécisions, en particulier dans la définition de décomposition en régions. Sans pour autant remettre en question les résultats obtenus par cette méthode, ces imprécisions rendent les preuves invalides. Ces lacunes étaient connues, mais n'ont jamais été comblées ; nous nous sommes proposés d'y remédier ici.

Dans ce chapitre, nous présentons donc une version corrigée de la méthode. Nous l'appliquons à deux problèmes : Domination Rouge-Bleu et Domination Totale. Ces problèmes sont connues pour être W[2]-complets [19] en générale ; NP-complets [2, 24, 58, 34] et FPT [19, 2, 29] dans les graphes planaires. Bien qu'il soit possible de montrer l'existence d'un noyau (pour Domination Totale du moins [34]) via les théorèmes de méta-noyaux [8], ces deux problèmes n'avaient pas de noyaux explicites et *ad hoc* connus jusqu'alors.

Dans la prochaine section, nous redéfinissons les outils d'Alber, Fellows et Niedermeier (section 2.1). Dans les deux sections suivantes, nous appliquons la méthode à deux autres variantes de Domination : les problèmes de Domination Rouge-Bleu (section 2.2) et de Domination Totale (section 2.3).

2.1 Méthode de décomposition en regions

La méthode de décomposition en régions se base sur le fait que la planarité impose une faible densité d'arêtes (ce qui permet de borner la taille du graphe réduit). Elle consiste à montrer que, étant donné un dominant du graphe, dans un plongement fixé, le voisinage de toute paire de sommets dominants peut être découpé en régions de sorte que les régions ne se croisent pas. L'inégalité d'Euler permet de borner le nombre de régions. Le nombre de sommets à l'extérieur et à l'intérieur des régions peut, ensuite, être borné grâce aux contraintes imposées par les règles de réduction.

Dans cette section, nous reprenons les travaux d'Alber, Fellows et Niedermeier, et y apportons plusieurs modifications. Nous définissons une notion de région un peu plus générale de façon à pouvoir l'appliquer à nos deux problèmes : Domination Rouge-Bleu et Domination Totale. Nous redéfinissons ensuite la notion de décomposition en régions de façon non ambiguë, en particulier, nous ajoutons la notion de régions non sécantes qui formalise l'idée de croisement de régions. Nous prouvons aussi que le multigraphe sous-jacent (cf. définition 46, multigraphe dont chaque arête correspond à une région d'une décomposition) à la décomposition construite par l'algorithme d'Alber, Fellows et Niedermeier est planaire et donc que la décomposition contient peu de régions.

Cependant, nous ne proposons pas de cadre général (à la façon de Guo et Niedermeier [40]) : toutes les preuves utilisant des propriétés spécifiques à un dominant seront traitées dans les sections suivantes (sections 2.2 et 2.3).

Nous formalisons d'abord l'idée de non croisement de chemins par la notion de chemins confluents. Intuitivement, deux chemins dans un graphe plan sont confluents s'ils peuvent se mélanger, mais pas se croiser, autrement dit, si le plan peut être coupé en deux parties contenant chacune un des chemins. Nous utilisons cette notions dans les définitions de régions et de décomposition en région. Pour les régions, nous voulons que ses bords ne se croisent pas (autrement dit que la régions ne soit pas vrillée). Pour la décomposition, ne voulons que ses régions ne se croisent pas.

Définition 39 : chemins confluents

Étant donné un graphe plan G, deux chemins p_1 et p_2 sont confluents si :

- ils sont sommets-disjoints,
- *ou* ils sont arêtes-disjoints et tout sommet commun u vérifie $[v_1 < w_1 < v_2 < w_2]_u$ où v_i, w_i sont les voisins de u dans p_i,
- *ou* ils sont confluents après contraction des arêtes communes.

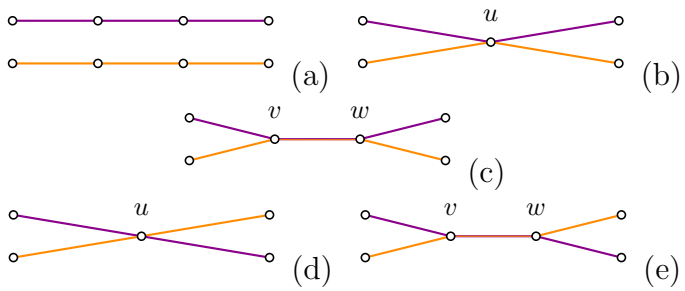


FIGURE 2.1 – Exemple et contre exemple de chemins confluents (cf. définition 39). (a) Les chemins sont confluents car ils sont disjoints. (b) Les chemins sont confluents car les sommets sont bien ordonnés autour de u. (c) Les chemins sont confluents car en contractant $\{v; w\}$ nous retrouvons (b). (d) Les chemins ne sont pas confluents car l'ordre autour de u n'est pas respecté. (e) Les chemins ne sont pas confluents car en contractant $\{v; w\}$ nous retrouvons (d).

Remarquons qu'un chemin est toujours confluent avec lui même. Nous soulignons que la terminologie peut prêter à confusion : par confluent, nous indiquons que les chemins forment un même faisceau, un même flux, qu'ils vont dans la même direction ; et nous ne faisons pas référence à une confluence au sens géographique (après laquelle il ne peut y avoir de séparation). Nous suivons la dénomination de Giannopoulou, Kamiński et Thilikos [37].

Dans le définition suivante, nous formalisons la notion de région. Intuitivement, il s'agit d'un sous-ensemble du plan qui induit un sous-graphe de petit diamètre, et tel que le bord de la région et le bord du sous-graphe correspondent. Par la suite, nous verrons que les sous-graphes induits par les régions peuvent être simulés par une petite famille de graphes de petites tailles ; cette famille est trouvée en considérant les façons dont une région peut intersecter une solution.

Définition 40 : région

Soit G un graphe plan, soient $v, w \in V(G)$ deux sommets distincts, et soit $r \in \mathbb{N}$. Une *vw-r-région* R est un fermé du plan tel que :

— ∂R est une paire de vw-chemins confluents de longueur au plus r,

— *et* tout sommet dans R appartient à $N^{\lfloor \frac{r-1}{2} \rfloor}[v, w] \cup (N^{\lceil \frac{r-1}{2} \rceil}[v] \cap N^{\lceil \frac{r-1}{2} \rceil}[w])$,

— *et* le complémentaire (dans le plan) de R est connexe.

Les *pôles* de R sont les sommets v et w. Les *chemins délimitants* de R sont les deux chemins vw-chemins confluents dont les images forment ∂R.

Étant donnés R_1, R_2 deux régions, nous utilisons la notation $R_1 \uplus R_2$ pour préciser que l'union des deux fermés $R_1 \cup R_2$ est aussi une région.

⌟

Formulons quelques remarques à propos de cette définition.

L'ensemble des sommets pouvant être dans la région est choisi en fonction de r pour pouvoir contenir entièrement les chemins délimitant (mais pas plus). Dans une région, comme sur un chemin délimitant, si r est pair alors il y a des sommets à égale distance des pôles (à distance $\lfloor \frac{r}{2} \rfloor$) ; si r est impair les sommets sont nécessairement plus proches de l'un ou de l'autre des pôles (à distance au plus $\lfloor \frac{r}{2} \rfloor$).

Le sous-graphe $G[V(R)]$ induit par les sommets d'une région est de diamètre au plus r, ce qui implique une largeur arborescente bornée [20] ; nous verrons dans le chapitre 3 que les régions peuvent être généralisées par la notion de protrusion. L'opération $\cdot \uplus \cdot$ n'est possible que lorsque les deux régions partagent les mêmes pôles et ont un chemin délimitant en commun ; le bord de la nouvelle région est formée des deux autres chemins.

Dans les deux utilisations suivante la longueur r est fixée (respectivement $r = 4$ dans la section 2.2 et $r = 3$ dans la section 2.3). Pour simplifier la notations, nous parlerons le plus souvent de vw-régions lorsque r est clairement défini par le contexte ou que son information n'est pas pertinente ; parfois nous parlerons de r-régions, lorsque l'information des pôles n'est pas utile.

Analysons maintenant les différentes contraintes que nous imposons dans cette définition. Tout d'abord, nous supposons que les pôles sont deux sommets distincts, cela nous évitera d'avoir à considérer des boucles dans le multigraphe sous-jacent (cf. définition 46). Puis, nous supposons que le bord est formé de deux chemins (et non des marches ou des parcours), ainsi, nous interdisons aux chemins délimitants de contenir des cycles (cf. figure 2.2(f)), cette contrainte est nécessaire car la confluence n'est définie que sur les chemins. Ensuite, nous supposons que le complémentaire d'une région est connexe, (cf. figure 2.2(g)), cette contrainte nous sera utile dans la preuve de la proposition 1. Enfin, nous imposons aux deux chemins délimitants de ne pas se croiser, cette contrainte nous sera nécessaire dans la preuve de la proposition 2. De plus, grâce à cette contrainte, les deux chemins sont définis de manière unique (cf. figure 2.2(e)).

À titre comparatif, nous donnons maintenant la définition d'Alber, Fellows et Niedermeier. Nous soulignons ensuite la première imprécision dans l'article de ceux-ci.

Définition : région (inexacte) [3]

Soit G un graphe plan et soit $v, w \in V(G)$ deux sommets. Une *vw-région* R est un fermé du plan tel que :

— ∂R est une paire de vw-parcours de longueur au plus 3,

— *et* tout sommet dans R appartient à $N[v, w]$.

⌟

Cette définition a été conçu pour DOMINATION, mais elle ne peut pas être utilisée telle-quelle pour DOMINATION ROUGE-BLEU. En effet, un sommet dominant (classique) est toujours à distance au plus 3 d'un autre sommet dominant, tandis que un sommet dominant rouge bleu est à distance au plus 4 d'un autre (sauf dans le cas d'une composante connexe dominé par un seul sommet ; une telle composante ne peut pas exister dans un graphe réduit, cf. règles 2, 3 et 4). Il nous faut donc proposer une définition dans laquelle la longueur des chemins délimitants de la région est un paramètre qui dépendant du problème. Remarquons que dans les deux cas qui nous intéressent ($r = 4$ dans les graphes bipartis et $r = 3$) les parcours sont nécessairement des chemins. Nous estimons cependant que les contraintes de confluence et de connexité sont manquantes.

Nous étendons ici l'idée de non croisement pour deux régions. Nous définissons la notion de régions non sécantes. Intuitivement, deux régions sont non sécantes si leurs intérieurs ne s'intersectent pas et si leurs bords sont confluents ; autrement dit, elles peuvent partager (en partie) un bord, mais qu'elles peuvent être séparées en coupant le plan en deux parties. Cette notion nous permet de montrer qu'une décomposition en région se comporte comme un graphe planaire (en particulier elle vérifie l'inégalité d'Euler).

Définition 41 : régions non sécantes

Deux régions R_1, R_2 sont *non sécantes* si :

— $(R_1 \setminus \partial R_1) \cap R_2 = (R_2 \setminus \partial R_2) \cap R_1 = \emptyset$,

— *et* les chemins délimitants de R_1 et ceux de R_2 sont confluents.

Nous dirons que deux régions R_1, R_2 sont *sécantes en u* si u est commun à deux chemins non confluents respectivement dans R_1 et R_2 tel que :

— les voisins de u ne respectent pas l'ordre imposé par la définition 39,

— *ou* u est l'extrémité d'une arête e telle que les deux régions sont sécantes en v_e après contraction de e en v_e.

⌟

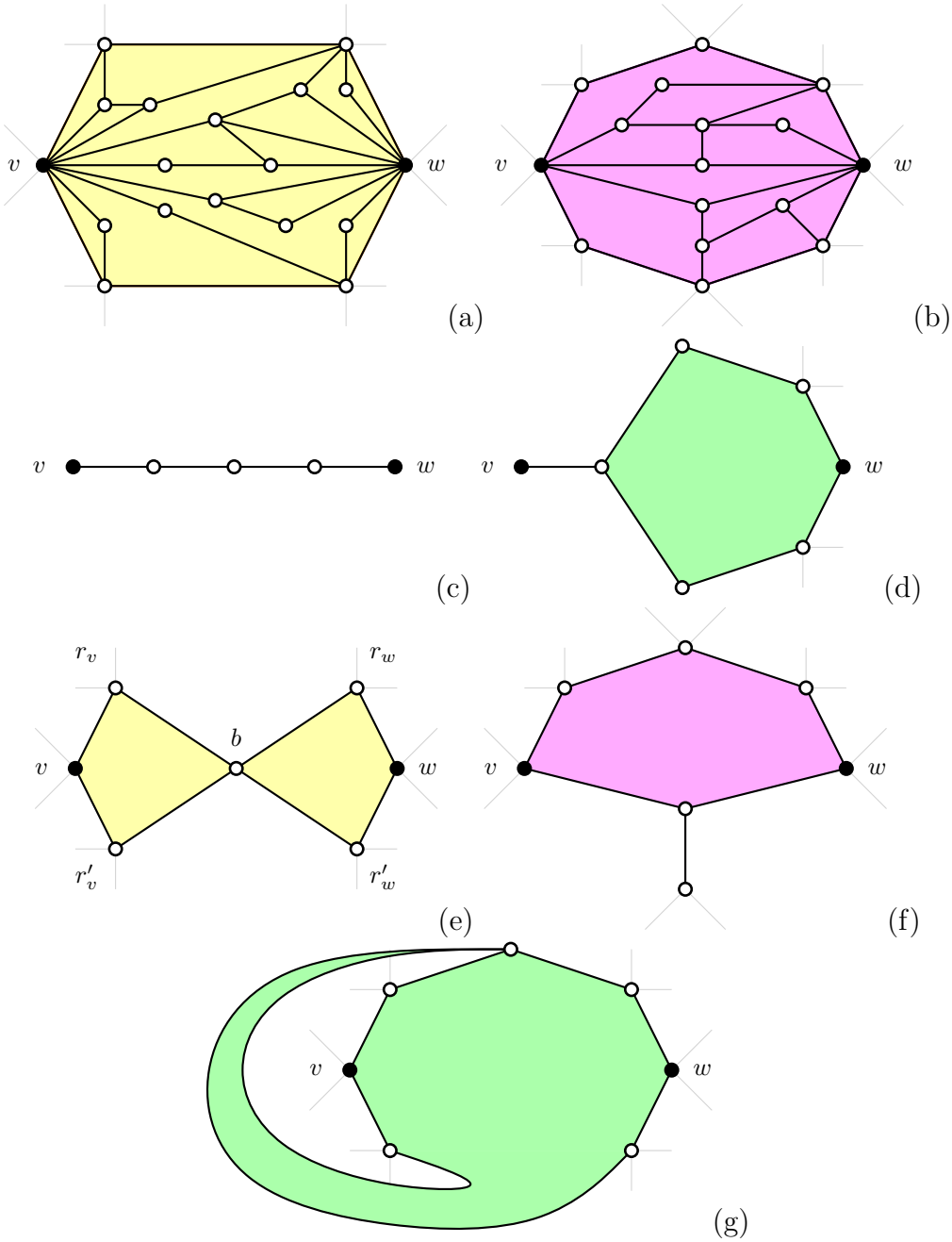


FIGURE 2.2 – Exemple et contre exemple de vw-régions (cf. définition 40). (a,b) Des régions pour les valeurs $r = 3$ et $r = 4$ respectivement. (c,d) Des 4-régions dégénérés : les chemins délimitants ont des sommets et des arêtes communs. (e) Une 4-région, sans la condition de confluence, son bord peut être défini de deux façons : ou bien par les chemins (v, r_v, b, r_w, w) et (v, r'_v, b, r'_w, w) ; ou bien par les chemins (v, r_v, b, r'_w, w) et (v, r'_v, b, r_w, w). (f) Un fermé qui n'est pas une région, un des chemins délimitants est une marche. (g) Un fermé qui n'est pas une région, son complémentaire est non connexe.

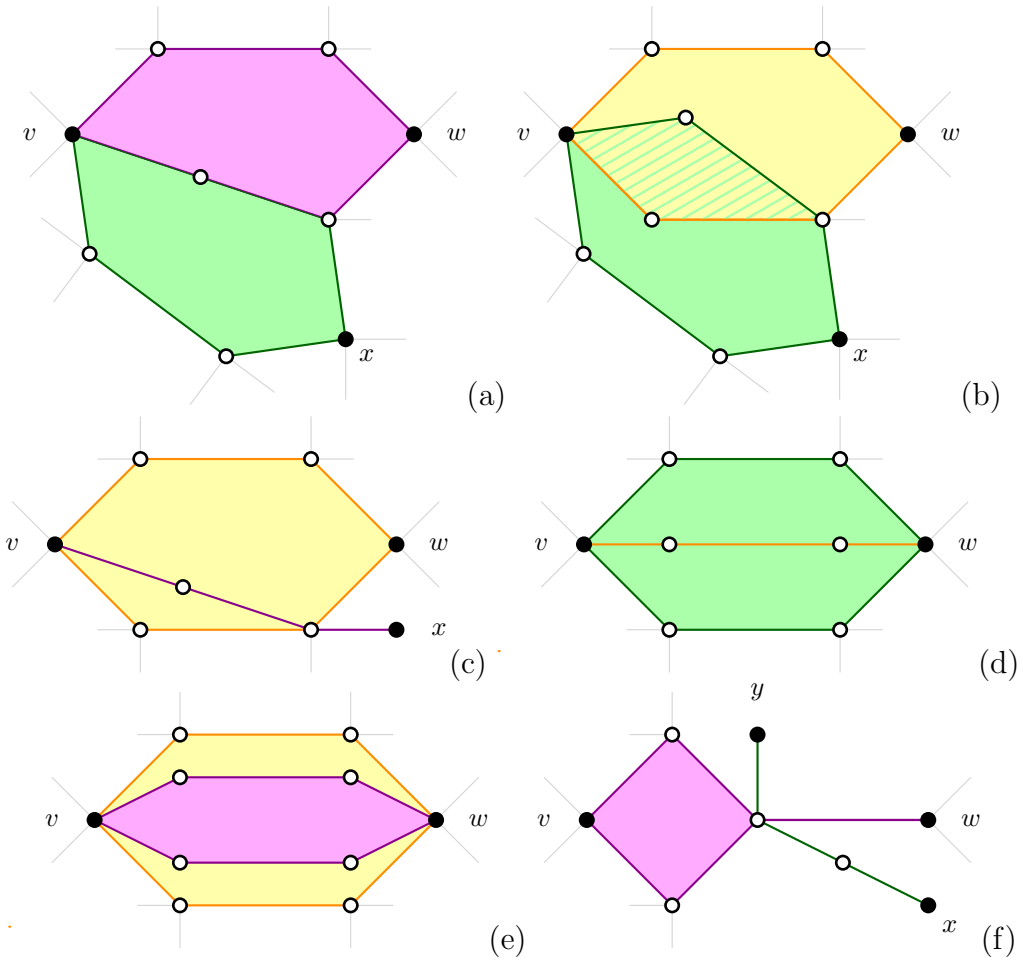


FIGURE 2.3 – Exemple et contre exemple de régions sécantes (cf. définition 41). (a) Les régions sont non sécantes. (b,c) Les régions sont sécantes car elles s'intersectent et les bords ne sont pas confluents. (d,e) Les régions sont sécantes car elles s'intersectent. (f) Les régions sont sécantes car les bords ne sont pas confluents.

Formulons quelques remarques à propos de cette définition. La première condition, celle de non intersection des intérieurs, ne sert qu'à interdire une région incluse dans une autre. Les chemins délimitants sont confluents deux à deux si et seulement si tout chemin inclus dans ∂R_1 est confluent avec tout chemin inclus dans ∂R_2. Par la suite, nous utiliserons le fait que deux régions sont sécantes en un sommet uniquement dans le cas où une des régions est un chemin.

Nous pouvons maintenant donner la définition d'une décomposition en régions. Intuitivement, il s'agit d'un découpage d'un graphe plan en régions dont les pôles font partie d'un dominant et qui ne se croisent pas. Par la suite nous montrerons que (sous certaines conditions) le nombre de régions dans la décomposition est linéaire en le nombre de pôles.

Définition 42 : décomposition en régions

Étant donné G un graphe plan et $D \subseteq V(G)$, une *D-décomposition en régions* de G est un ensemble $\Re$ de régions dont les pôles sont dans D tel que :

— toute vw-région $R \in \Re$, vérifie $D \cap V(R) = \{v, w\}$,

— ^{et} toutes les régions sont deux à deux non sécantes.

Par abus de notation, nous notons $V(\Re) = \bigcup_{R \in \Re} V(R)$. ⌟

Dans toute la suite de ce chapitre 2 une décomposition sera une décomposition en régions. Remarquons qu'une décomposition en régions ne couvre pas nécessairement tous les sommets du graphe $V(\Re) \subseteq V(G)$ (il y aura égalité dans la section 2.2, mais pas dans la section 2.3).

À titre comparatif, nous donnons maintenant la définition de décomposition en régions d'Alber, Fellows et Niedermeier, puis nous expliquons en quoi elle ne convient pas. Nous soulignons ainsi la deuxième imprécision dans l'article d'Alber, Fellows et Niedermeier.

Définition : décomposition (inexacte) [3]

Étant donné G un graphe plan et $D \subseteq V(G)$, une *D-décomposition en régions* de G est un ensemble $\Re$ de régions dont les pôles sont dans D tel que :

— toute vw-région $R \in \Re$, vérifie $D \cap V(R) = \{v, w\}$,

— ^{et} toute paire $R_1, R_2 \in \Re$, vérifie $R_1 \cap R_2 \subseteq \partial R_1 \cap \partial R_2$.

⌟

Cette définition ne permet pas de démontrer que le nombre de régions sera linéaire en $|D|$. L'erreur vient du fait que le second point de la définition cherche à interdire les croisements de région (au sens où le multigraphe sous-jacent (cf. définition 46) sera planaire), mais il n'interdit que le chevauchement (au sens où l'intersection des intérieurs est vide). En effet, dans la définition de régions, les deux chemins délimitants ne sont pas nécessairement disjoints, en particulier, le cas d'une région dégénérée en un chemin (cf. figure 2.2(c)), et donc d'intérieur vide, n'est pas interdit (nous utilisons largement ces régions dégénérées par la suite). Il est donc possible que deux régions (par exemple, des chemins) se croisent sans qu'elles ne se chevauchent (cf. figure 2.3(f)). De sorte que nous pouvons construire des familles d'exemples dans lesquelles le nombre de régions ne peut pas être bornée linéairement en fonction de la taille d'un dominant (cf. figure 2.4). La contrainte de confluence nous permet d'interdire les croisements de régions tout en permettant que les bords soient en partie confondus.

Nous considérons, en particulier, des décompositions maximales au sens où elles contiennent d'abord un nombre de sommets maximal, ensuite (sous cette contrainte) un nombre de régions minimal. Intuitivement, une décomposition est maximale s'il est impossible d'y ajouter de sommet, ni par ajout de région, ni par élargissement d'une région ; et s'il est impossible de fusionner de deux régions. Par la suite, supposer que nos régions sont maximales nous permettra de borner le nombre de sommets hors de la décomposition.

Définition 43 : décomposition maximale
Une D-décomposition en régions $\Re$ est *maximale* si :

— pour toute région $R \notin \Re$ telle que $V(\Re) \subset V(\Re \cup \{R\})$, $\Re \cup \{R\}$ n'est pas une D-décomposition,

— *et* pour toutes régions $R \notin \Re$ et $R' \in \Re$ telles que $V(R') \subset V(R)$, $\Re \cup \{R\} \setminus \{R'\}$ n'est pas une D-décomposition,

— *et* pour toutes régions $R_1, R_2 \in \Re$ telles que $R_1 \uplus R_2$ est une région, $\Re \cup \{R_1 \uplus R_2\} \setminus \{R_1, R_2\}$ n'est pas une D-décomposition.

⌟

Remarquons qu'une décomposition maximale est optimale en terme d'inclusion mais qu'elle n'est pas optimum au sens de la cardinalité.

Notre définition de décomposition maximale contient plus de conditions que celle d'Alber, Fellows et Niedermeier, cependant, il s'agit exactement des contraintes imposées par leur algorithme et qui sont utilisées au cours des preuves.

Lemme 2
Soit G un graphe plan et $D \subseteq V(G)$. Il existe une D-décomposition maximale $\Re$.

Démonstration. Nous considérons l'algorithme suivant.
Algorithme : Décomposition maximale.
pour chaque *sommet* $u \notin V(\Re)$ **faire**
ajouter à $\Re$ une vw-région R telle que
- $V(R) \cap D = \{v, w\}$,
- $\overset{et}{—}$ R est non sécante avec toute région dans $\Re$,
- $\overset{et}{—}$ $V(R)$ est maximale par inclusion,
- $\overset{et}{—}$ $u \in V(R)$;

si elle existe.

À chaque étape $\Re$ est une D-décomposition, en effet si une région R est ajoutée, alors $\Re \cup \{R\}$ est une décomposition (d'après les deux premiers items). De plus, à chaque étape pour toute région $R \in \Re$ il n'existe pas de région R' telle que $V(R') \subset V(R)$ et $\Re \cup \{R\} \setminus \{R'\}$ soit une décomposition ni de région R'' telle que $\Re \cup \{R_1 \uplus R_2\} \setminus \{R_1, R_2\}$ soit une décomposition (d'après le troisième item) ; lorsque l'algorithme s'arrête $\Re$ est maximale car tous les sommets ont été considérés (et d'après le quatrième item) donc la décomposition est maximale. □

Les définitions étant posées, nous allons maintenant décrire les outils qui nous permettrons de montrer qu'une D-décomposition maximale a un nombre de régions linéaire en $|D|$. Comme nous l'avons annoncé, pour démontrer cette borne il est nécessaire que l'ensemble D vérifie certaines propriétés, qui sont celles d'un ensemble dominant ; par conséquent, la preuve proprement dite sera faite dans les section 2.2 et section 2.3.

L'idée, pour obtenir la borne linéaire, consiste à voir la décomposition comme un multigraphe dont chaque arête correspond à une région. Nous montrons ensuite que la maximalité impose une certaine structure sur le multigraphe, laquelle permet d'appliquer l'inégalité d'Euler.

Définition 44 : multigraphe
Un *multigraphe* M est un couple $(V(M), E(M))$ où $V(M)$ est un ensemble de *sommets* de G et E est un multiensemble d'*arêtes* de G.

Pour une grande part, les multigraphes se comportent comme les graphes. La plupart des objets que nous avons définis sur les graphes (voisinage, degré,

chemin, distance, plongement ...) sont transposables à la notion de multigraphe. Néanmoins sur certains points les multigraphes diffèrent. Remarquons par exemple que, étant donné un sommet, le nombre ses voisins de n'est pas nécessairement égal à son degré. Remarquons aussi qu'un chemin ne peut pas être défini par la suite de ses sommets ; puisque une arête entre deux sommets n'est pas unique, il convient de spécifier laquelle est utilisée par le chemin.

Définition 45 : multigraphe mince

Un multigraphe planaire G est *mince* si il existe un plongement de G tel que, pour toutes arêtes e_1, e_2 avec des extrémités identiques, dans chacun des deux sous-ensembles du plan délimités par les images de e_1, e_2 se situe l'image d'un sommet.

⌟

Un multigraphe mince vérifie l'inégalité d'Euler. Nous soulignons ici la troisième imprécision dans l'article d'Alber, Fellows et Niedermeier, lequel fournit une preuve erronée de ce résultat. En particulier, leur lemme énonce que le multigraphe doit être mince, mais la preuve n'utilise pas cette propriété. Par soucis de complétude nous en fournissons une preuve correcte.

Rappelons que, pour un graphe planaire G, la formule d'Euler implique l'inégalité : $|E(G)| \leqslant 3|V(G)|-6$. Nous montrons que cela peut être généralisé aux multigraphes minces. De plus, afin d'améliorer nos bornes (cf. proposition 6), nous feront usage de l'inégalité d'Euler dans le cas particulier des multigraphes planaires bipartis.

Lemme 3

Soit M est un multigraphe planaire. Si M mince et $|V(M)| \geqslant 3$ alors $|E(M)| \leqslant 3|V(M)| - 6$.

En particulier, si de plus M est biparti alors $|E(M)| \leqslant 2|V(M)| - 4$.

⌞ ⌟

Démonstration. Nous rappelons qu'un multigraphe planaire (cellulairement plongé sur la sphère) vérifie la formule d'Euler : $|V(M)|-|E(M)|+|F(M)| = 2$ avec $F(M)$ l'ensemble des faces de M. Nous ajoutons des arêtes à M de façon à le rendre connexe. Nous considérons (avec deux approches) la somme des degrés des faces.

D'une part, chaque arête est comptée deux fois : ou bien une fois dans deux faces (l'arête fait partie d'un cycle), ou bien deux fois dans une face (l'arête fait partie d'un arbre) ; donc $\sum_{f \in F(M)} \delta(f) = 2|E(M)|$.

D'autre part (pour les multigraphes planaires quelconques), chaque face est de degré au moins 3. En effet, considérons une face f de degré 2, comme le multigraphe est mince, ∂f contient au moins deux marches, une de longueur

2 l'autre de longueur 0 (un sommet), ce qui implique deux composantes connexes, contradiction. Donc $3|F(M)| \leqslant \sum_{f \in F(M)} \delta(f)$.

Nous obtenons $3|F(M)| \leqslant 2|E(M)|$ et donc, par la formule d'Euler, $|E(M)| \leqslant 3|V(M)| - 6$.

Pour les multigraphes planaires bipartis, remarquons que, de plus, chaque face est de degré au moins 4. Nous obtenons $4|F(M)| \leqslant 2|E(M)|$ et donc, par la formule d'Euler, $|E(M)| \leqslant 2|V(M)| - 4$.

□

Étant donné un graphe G et une D-décomposition $\Re$, nous construisons un multigraphe $G_\Re$ dont chaque arête correspond à une région de $\Re$. Comme les régions sont non sécantes, nous pouvons montrer que $G_\Re$ est planaire. Nous soulignons ici la quatrième imprécision dans l'article d'Alber, Fellows et Niedermeier, lequel ne définit pas formellement le multigraphe $G_\Re$ et ne prouve pas sa planarité. Une telle preuve n'était pas possible avec la définition originelle de décomposition en région ; en effet, nous pouvons exhiber des exemples de décomposition (selon la définition d'Alber, Fellows et Niedermeier), dont le multigraphe sous-jacent n'est pas planaire.(cf. figure 2.4).

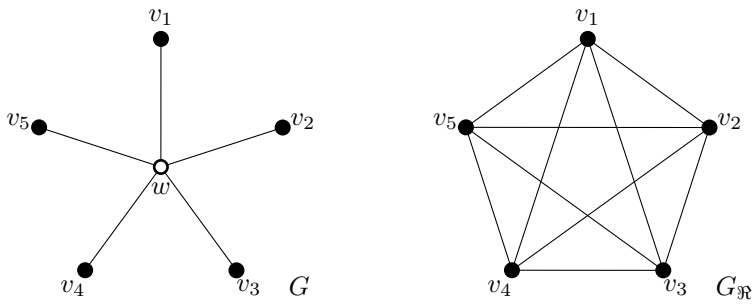


FIGURE 2.4 – Dans le graphe étoile G, si nous choisissons comme dominant $D = \{v_i \mid i \in [1,5]\}$ alors d'après la définition erronée nous pouvons construire la décomposition $\Re = \{\pi(v_i, w) \cup \pi(v_j, w) \mid i \neq j\}$. Le multigraphe $G_\Re$ sous-jacent à $\Re$ (cf. définition 46) est un K_5 donc non planaire. Remarquons que l'exemple ne correspond ni à un graphe réduit, ni à un dominant minimum.

Définition 46 : multigraphe sous-jacent

Soit G un graphe plan, $D \subseteq V(G)$ et $\Re$ une D-décomposition. Le multigraphe $G_\Re$ *sous-jacent à* $\Re$ est un multigraphe dont l'ensemble de sommets $V(G_\Re) = D$ et avec une arête $\{v; w\} \in E(G_\Re)$ pour chaque vw-région de $\Re$.

⌟

Lemme 4

Soit G un graphe plan, $D \subseteq V(G)$ et $\Re$ une D-décomposition. Le multigraphe $G_\Re$ sous-jacent à $\Re$ est planaire.

Démonstration. Soit π le plongement de G dans le plan. Nous considérons l'application $\pi_\Re$ de $G_\Re$ dans le plan telle que :

- pour tout sommet $v \in D$, $\pi_\Re(v) = \pi(v)$;
- pour toute arête $e \in E(G_\Re)$, $\pi_\Re(e) = \bigcup_{f \in p_e} \pi(f)$,
  où p_e est un chemin délimitant de la région correspondant à e
  (nous appellerons p_e le *support de* e, cf. proposition 1).

Remarquons que les images des sommets sont disjointes deux à deux, que l'image d'une arête $\{v; w\}$ à pour points extrémaux les images de v et w, et qu'elle ne contient pas d'image de sommet de $D \setminus \{v, w\}$; car π est un plongement et car une vw-région ne contient pas de sommet de $D \setminus \{v, w\}$. L'algorithme ci-dessous modifie $\pi_\Re$ de sorte que les images des arêtes soient disjointes deux à deux.

Pour un ensemble d'arêtes $E \subseteq E(G_\Re)$ nous notons $\pi_\Re(E) = \bigcap_{e \in E} \pi_\Re(e)$. Observons que, étant donnés $E, E' \subseteq E(G_\Re)$, si $E \subseteq E'$ alors $\pi_\Re(E) \supseteq \pi_\Re(E')$. Nous modifions $\pi_\Re$ par l'algorithme suivant.

Algorithme : Replongement.

tant que *il existe* $E \subseteq E(G_\Re)$ *tel que* $\pi_\Re(E) \not\subseteq \bigcup_{v \in D} \pi_\Re(v)$ **faire**
  soit E maximal par inclusion, telle que $\pi_\Re(E) \not\subseteq \bigcup_{v \in D} \pi_\Re(v)$;
  soit $C(E)$ un sous-ensemble du plan contenant pour chaque point $x \in \pi_\Re(E) \setminus \bigcup_{v \in D} \pi_\Re(v)$, une boule de centre x et de rayon $\varepsilon > 0$ avec ε suffisamment petit pour que $C(E)$ n'intersecte que des images d'arêtes dans E de façon connexe;
  pour chaque *composante connexe* $C'(E)$ *de* $C(E)$,
  et chaque *arête* $e \in E$ **faire**
    soit p, q les points extrémaux de $\pi_\Re(e) \cap C'(E)$;
    soit $C_e \subset C'(E)$ une courbe de points extrémaux p, q telle que $(C_e \setminus \{p, q\}) \cap \bigcup_{e' \in E} \pi_\Re(e') = \emptyset$;
    modifier $\pi_\Re(e) := \pi_\Re(e) \setminus C'(E) \cup C_e$.

Formulons plusieurs remarques sur cet algorithme. D'abord, lorsque l'algorithme s'arrête, pour tout $E \subseteq E(G_\Re)$, $\pi_\Re(E)$ est vide ou restreint à un ou deux points ; ces points sont nécessairement les images des extrémités communes à toutes les arêtes dans E. Ensuite, le fermé du plan $C(E)$ existe car les ensembles d'arêtes sont traitées dans l'ordre décroissant ; si $C(E)$ intersecte l'image d'une arête $e \notin E$ (sauf points extrémaux) alors $\pi_\Re(e) \cap \pi_\Re(E)$ n'est pas restreinte aux points extrémaux des images d'arêtes, contradiction avec la maximalité de E. Enfin, la courbe C_e existe car les chemins sont confluents ; si $\pi_\Re(e)$ est un point, alors $\pi_\Re(e)$ est l'image d'un sommet $u \in V(G)$ commun à tous les chemins p_e pour $e \in E$, les extrémités d'un chemin p_e sont

consécutives sur le bord de $C(E)$ (par définition de la confluence et de l'ordre autour de u) ; si $\pi_{\Re}(e)$ est une courbe, alors $\pi_{\Re}(e)$ est l'image d'arêtes de G communes à tous p_e pour $e \in E$, en contractant ses arêtes nous pouvons nous ramener à l'argumentation précédente (par définition de la confluence).

Lorsque l'algorithme ne peut plus être appliqué alors les images des arêtes sont disjointes (sauf les points extrémaux), donc l'application $\pi_{\Re}$ est un plongement de $G_{\Re}$ dans le plan.

□

Dans les sections suivantes (sections 2.2 et 2.3), nous montrerons que si D est un dominant et $\Re$ est maximal alors $G_{\Re}$ est mince en plus d'être planaire. Nous pourrons alors appliquer les lemmes 3 et 4 pour borner linéairement $|\Re|$.

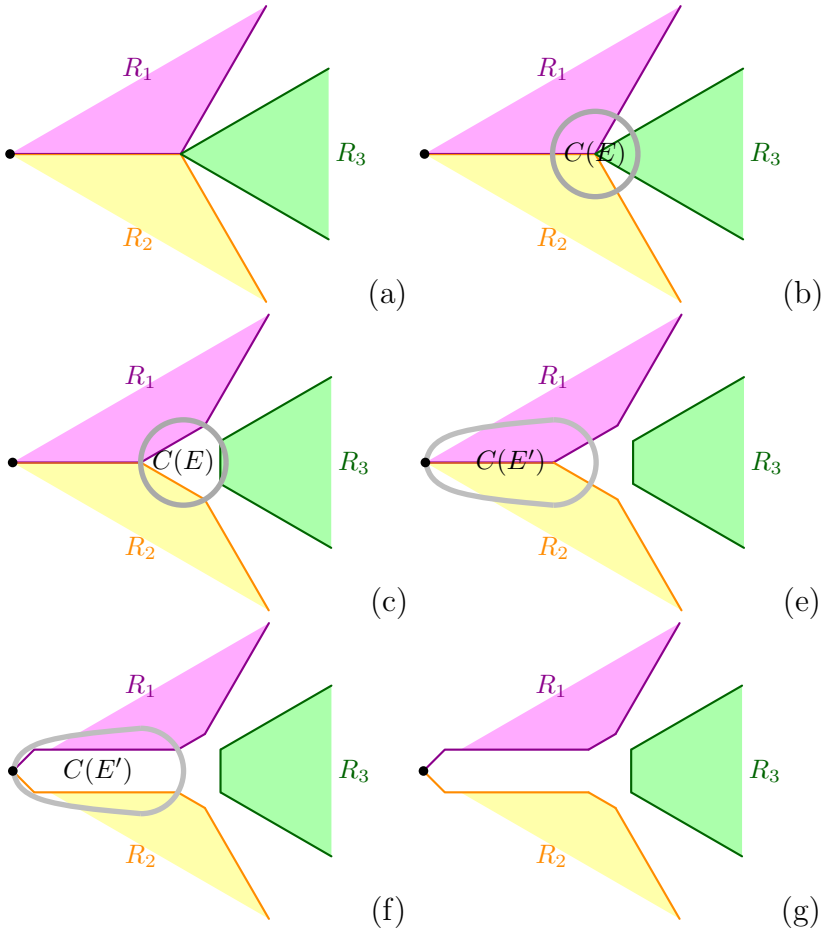


FIGURE 2.5 – Illustration de la procédure de replongement dans le lemme 4. (a) Les arêtes associées aux trois régions R_1, R_2, R_3 sont plongées sur le bord de la région. (b) La procédure considère d'abord E l'ensemble des trois arêtes dont l'intersection est un point. (c) Le plongement est modifié localement dans $C(E)$ autour de ce point. (d) La procédure considère ensuite E' l'ensemble de deux arêtes dont l'intersection est un segment. (e) Le plongement est modifié localement dans $C(E')$ autour de ce segment. (f) Le plongement vérifie toutes les conditions de la définition 13.

2.2 Noyau pour Domination Rouge-Bleu

Dans cette section, nous construisons un noyau linéaire pour le problème Domination Rouge-Bleu dans les graphes planaires. Nous utilisons pour cela la méthode de décomposition en régions, avec des 4-régions. Étant donné un graphe dont les sommets sont partagés en deux ensembles : les sommets rouges et les sommets bleus, le problème Domination Rouge-Bleu consiste à trouver un sous-ensemble de sommets bleus qui domine tous les sommets rouges.

Pour ce problème, nous considérons donc des graphes un peu particuliers puisque leurs sommets sont annotés par une couleur, ou bien bleu, ou bien rouge. Nous les appelons graphes bicolorés.

Définition 47
Un graphe *bicoloré* G est un graphe dont l'ensemble de ses sommets est partitionné en deux $V(G) = V_B \cup V_R$. Un sommet *bleu* est un sommet de V_B et un sommet *rouge* est un sommet de V_R.
⌟

Définition 48 : dominant rouge-bleu
Un *dominant rouge-bleu* est un ensemble $D \subseteq V_B$ de sommets bleus tel que $N(D) \supseteq V_R$.
⌟

Domination Rouge-Bleu :
instance : un graphe $G = (V(G) = V_B \cup V_R, E(G))$ et un entier k ;
paramètre: l'entier k ;
question : existe-t-il un dominant rouge-bleu de taille au plus k ?

Du point de vue de la complexité classique, Domination Rouge-Bleu est NP-complet, y compris dans les graphes planaires [2]. Du point de vue paramétré il est W[2]-complet en général [19], mais devient FPT [19] lorsqu'il est restreint aux graphes planaires (avec la taille de la solution comme paramètre). Remarquons que, de part la nature même du problème, nous pouvons supposer que les instances de Domination Rouge-Bleu sont bipartis (cf. règle 1). Autrement dit, Domination Rouge-Bleu est trivialement équivalent à Domination Rouge-Bleu sur les bipartis.

Signalons que ce problème a plusieurs applications, tant pratiques (dans le contexte des réseaux de chemin de fer [57]) que théoriques (pour prouver la non existence de noyaux polynomiaux [18]).

En ce qui concerne les noyaux, le problème Domination Rouge-Bleu n'entre pas directement dans le cadre standard des méta-résultats [8, 27, 40, 43]. Les énoncés de ces méta-théorèmes ne permettent pas de considérer des problèmes dont les instances sont des graphes annotés. Bien qu'il soit envisageable de suivre le même schéma de preuve pour montrer l'existence d'un noyau linéaire, les constantes qui en découleraient seraient larges voir non explicites. Nous fournissons un noyau *ad hoc* pour ce problème restreint aux graphes planaires. Notre noyau est constructif et sa constante est explicite.

Théorème

Le problème Domination Rouge-Bleu, restreint aux graphes planaires et paramétré par k la taille de la solution, admet un noyau linéaire de taille $43k$.

Nous soulignons le fait que notre noyau linéaire a une petite constante en comparaison à Domination (335 [3], puis 67 [12]), et que les règles de réduction pour y parvenir sont relativement simples. Le problème Domination Rouge-Bleu est intuitivement plus simple que les autres variantes de Domination en cela que nous pouvons supposer que le graphe est biparti en plus d'être planaire, et que nous avons une indication *a priori* sur quel sommet est dominant, quel sommet est dominé. Pour cette raison nous pensons qu'il est plus pédagogique d'illustrer d'abord la méthode de décomposition en régions sur ce problème.

Comme nous l'avons annoncé dans la section 2.1, la description du noyau se fait en deux étapes : dans la sous-section 2.2.1, nous donnons l'algorithme d'extraction de noyau, décrit par plusieurs règles de réduction appliquées itérativement ; dans la sous-section 2.2.2, nous utilisons la méthode de décomposition en régions pour borner la taille de l'instance réduite.

2.2.1 Réduction de Domination Rouge-Bleu

Dans cette sous-section, nous présentons les règles de réduction pour Domination Rouge-Bleu. Les règles de réduction décrivent des modifications locales de l'instance : si le voisinage d'un sommet, ou d'une paire de sommets, est dans une certaine configuration, alors il peut être remplacé par un ensemble plus petit (voir complètement supprimé), sans changer le comportement du graphe par rapport à Domination Rouge-Bleu. Certaines configurations seront donc impossibles dans un graphe réduit, ce qui nous permettra de borner la taille de celui-ci dans la sous-section 2.2.2.

Règles élémentaires

Nous décrivons d'abord quelques règles élémentaires, qui simplifient considérablement une instance. Elles imposent des contraintes claires et faciles à manipuler sur le graphe réduit. En particulier, d'après la règle 1, nous pouvons désormais nous restreindre au cas des graphes bipartis (nous ne ferons plus référence à la règle 1 par la suite). D'après les règles 2 et 3 nous pouvons toujours supposer que deux sommets ont des voisinages incomparables par la relation d'inclusion. Ces deux règles ont déjà été mentionnées par Weihe [57].

Règle 1
Soit $e \in E(G)$. Si e a ses extrémités de la même couleur :

— supprimer e.

Règle 2
Soit $b, b' \in V_B$. Si $N(b) \subseteq N(b')$:

— supprimer b.

Règle 3
Soit $r, r' \in V_R$. Si $N(r) \supseteq N(r')$:

— supprimer r.

Lemme 5
Soit G un graphe bicoloré. Si G' est le graphe obtenu par application de la règle 1, 2, ou 3, alors G admet un dominant rouge-bleu de taille k si et seulement si G' en admet un.

Démonstration. Si une arête e avec ses extrémités de la même couleur est supprimée alors, pour tout sommet $b \in V_B$ ou $r \in V_R$, l'ensemble $N(b) \cap V_R$ ou respectivement $N(r) \cap V_B$ n'est pas modifié. Les solutions de G sont donc les mêmes que celles de G'.

Si, pour deux sommets bleus b, b', $N(b) \subseteq N(b')$ alors tout sommet rouge dominé par b peut être dominé par b'. Donc toute solution de G contenant b peut être transformée en une solution de G et de G' contenant b'.

Si, pour deux sommets rouges r, r', $N(r) \supseteq N(r')$ alors tout sommet bleu dominant r' domine aussi r. Les solutions de G sont donc les mêmes que celles de G'

□

La règle suivante est particulière en cela qu'elle fait décroître le paramètre. Contrairement aux autres règles qui suppriment des sommets inutiles pour dominer, celle-ci identifie et supprime un sommet nécessairement dominant.

Règle 4

Soit $b \in V_B$. Si il existe $r \in N(b)$ tel que $\delta(r) = 1$.

— supprimer $N[b]$;

— diminuer le paramètre k de 1.

Si un sommet bleu b a un voisin rouge de degré 1, alors b est dans tout dominant rouge-bleu du graphe. Puisque b est nécessairement dominant, nous pouvons supposer qu'il a déjà été sélectionné et que son voisinage est déjà dominé ; il reste à trouver un dominant rouge-bleu de taille $k-1$ pour le reste du graphe.

Lemme 6

Soit G un graphe bicoloré, soit $b \in V_B$. Si $(G', k-1)$ est l'instance obtenue par application de la règle 4 sur b, alors G admet un dominant rouge-bleu de taille k si et seulement si G' en admet un de taille $k-1$.

Démonstration. Soit D un dominant rouge-bleu de G de taille k. Puisque la règle a été appliquée, il existe $r \in N(b)$ tel que $\delta(r) = 1$. Puisque r est dominé, $b \in D$. Or $V(G') = V(G) \setminus N(b)$. Donc $D \setminus \{b\}$ est un dominant rouge-bleu de G' de taille $k-1$.

Inversement, soit D' un dominant rouge-bleu de G' de taille k'. Puisque $V(G) = V(G') \cup N(b)$, $D' \cup \{b\}$ est un dominant rouge-bleu de G de taille $k'+1$. □

Règle pour un sommet seul

Nous décrivons maintenant une règle qui peut être obtenue par composition des règles 2 et 3. Cette règle n'est donc pas indispensable pour obtenir le noyau. Nous faisons le choix de la mentionner néanmoins. D'une part, nous conservons ainsi une structure similaire à l'article d'Alber, Fellows et Niedermeier [3] sur Domination, et à la section 2.3 sur Domination Totale. D'autre part, cette règle donne un premier aperçu de l'intuition de la règle 6 (qui suit), et des règles 8 et 9 (utilisées pour le noyau de Domination Totale).

Cette règle s'applique à un sommet bleu. De la même manière que la règle 4, elle se base sur une condition (plus large que dans la règle 4) qui lui permet de supposer que le sommet considéré est nécessairement dans tout dominant rouge-bleu. Si tel est le cas nous simplifions le voisinage.

Définition 49 : voisinage privé d'un sommet

Le voisinage *privé* d'un sommet bleu v est l'ensemble :

$$P(v) = \{r \in N(v) \mid N^2(r) \subseteq N(v)\}.$$

⌟

Remarquons qu'un sommet $r \in P(v)$ peut avoir un voisin $b \neq v$, mais ce voisin n'est pas meilleur que v pour la domination (au sens où, s'il existe une solution contenant b alors il existe une solution contenant v) car $N(b) \subseteq N(v)$. Autrement dit, si nous supposons le graphe réduit par la règle 2, nous pouvons alors dire que le voisinage d'un sommet v est séparé en deux : $P(v)$ l'ensemble des voisins adjacents à v uniquement et $N(v) \setminus P(v)$ l'ensemble des voisins adjacents au reste du graphe. Il s'en suit que si $P(v) \neq \emptyset$ nous pouvons supposer que $v \in D$.

Règle 5

Soit $v \in V_B$ un sommet bleu. Si $P(v) \geqslant 1$:

— supprimer $P(v)$,

— ajouter un sommet rouge v' et l'arête $\{v; v'\}$.

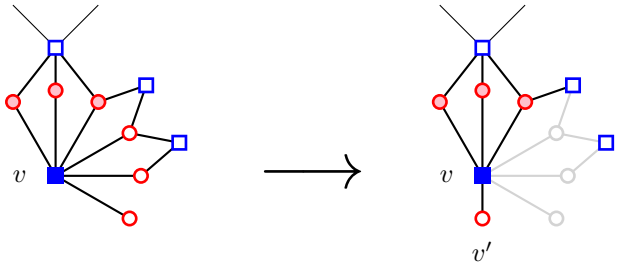


FIGURE 2.6 – Illustration de l'application de la règle 5 sur un sommet v. Remarquons que la règle 2 supprimera deux sommets bleus et que la règle 4 supprimera $N[v]$.

Le fait que $P(v)$ soit non vide implique que v est nécessairement un sommet dominant car les sommets de $P(v)$ n'ont pas d'autre voisin bleu (ou du moins pas de meilleur voisin). Le gadget v' simule $P(v)$: il force le choix de v dans la solution (cf. figure 2.6).

Lemme 7

Soit G un graphe bicoloré, soit $v \in V_B$. Si G' est le graphe obtenu par application de la règle 5 sur v, alors G admet un dominant rouge-bleu de taille k si et seulement si G' en admet un.

⌞ ⌟

Démonstration. Nous montrons que cette règle est une composition des règles 2 et 3. Soit $b \in N(P(v))$ distinct de v. Par définition $N(b) \subseteq N(v)$, donc b

est supprimé par la règle 2. Il s'en suit que $P(v)$ est un ensemble de sommets de degré 1, donc la règle 3 supprime tous les sommets de $P(v)$ sauf un. Ce qui prouve la correction.

□

Par la suite, dans l'analyse de la taille du noyau, nous utilisons souvent le fait qu'un graphe est réduit par les règles 4 et 5, ce qui implique que tout sommet bleu a un voisinage privé vide.

Règle pour une paire de sommets

Nous décrivons maintenant une règle s'appliquant à une paire de sommets bleus et son voisinage : en fonction de la manière dont les deux sommets peuvent ou non dominer le voisinage privé, leur voisinage peut être simplifié (c'est-à-dire, remplacé par un gadget qui préserve les différentes solutions). Intuitivement, nous voudrions identifier les cas où un des deux sommets, voire les deux, sont nécessairement dominants. Cependant, afin d'être plus précis dans la réduction (et d'obtenir une meilleure borne) nous considérons, en plus, toutes les façons de dominer le voisinage privé avec un seul sommet.

Définition 50 : voisinage privé d'une paire

Le voisinage *privé* d'une paire de sommets bleus v, w est l'ensemble :

$$P(v, w) = \{r \in N(v, w) \mid N^2(r) \subseteq N(v, w)\}.$$

Pour simplifier les notations, nous notons :

$$\bar{P}(v, w) = N(v, w) \setminus P(v, w).$$

⌟

Nous pouvons aisément observer que, au pire, il faut 2 sommets pour dominer $P(v, w)$ et que tant qu'à utiliser 2 sommets, v, w sont un meilleur choix (au sens où v, w dominent plus de sommets rouges que n'importe quelle autre paire dominant $P(v, w)$). Donc, si ni v ni w n'est dominant, c'est que $P(v, w)$ est dominé par un unique sommet. Afin de conserver un telle façon de dominer, nous considérons l'ensemble des sommets pouvant dominer $P(v, w)$ seul. Ensuite la règle remplacera $P(v, w)$ par un gadget qui permet les mêmes choix pour un ensemble dominant.

Définition 51 : sommets dominants $P(v, w)$

Soit $v, w \in V_B$. Nous considérons les sommets $d \in V_B$ tel que $P(v, w) \subseteq N(d)$; nous notons :

$$\mathcal{D} = \{d \in V_B \setminus \{v, w\} \mid P(v, w) \subseteq N(d)\}.$$

⌟

Règle 6

Soit v, w deux sommets bleus. Si $|P(v,w)| \geqslant 1$:

1. si $P(v,w) \not\subseteq N(v)$ et $P(v,w) \not\subseteq N(w)$:
   - supprimer $P(v,w)$,
   - ajouter deux sommets rouges v', w' et les arêtes $\{v;v'\}, \{w;w'\}$,
   - pour chaque $d \in \mathcal{D}$ ajouter les arêtes $\{d;v'\}, \{d;w'\}$;
2. si $P(v,w) \subseteq N(v)$ et $P(v,w) \subseteq N(w)$:
   - supprimer $P(v,w)$,
   - ajouter un sommet rouge y et les arêtes $\{v;y\}$ et $\{w;y\}$;
   - pour chaque $d \in \mathcal{D}$ ajouter l'arête $\{d;y\}$;
3. si $P(v,w) \subseteq N(v)$ et $P(v,w) \not\subseteq N(w)$:
   - supprimer $P(v,w)$,
   - ajouter un sommet rouge v' et l'arête $\{v;v'\}$;
   - pour chaque $d \in \mathcal{D}$ ajouter l'arête $\{d;v'\}$;
4. si $P(v,w) \not\subseteq N(v)$ et $P(v,w) \subseteq N(w)$:
   - faire symétriquement à l'item 3.

Cette règle se base sur le fait que pour dominer $P(v,w)$ il faudra utiliser au moins un sommet (car $P(v,w)$ est non vide), et au plus deux (car v et w dominent $P(v,w)$). L'item 1 considère le cas où ni v ni w ne peuvent dominer $P(v,w)$ seul. L'item 2 considère le cas où, au choix, v ou w peut dominer $P(v,w)$ seul. L'item 3 considère le cas où v peut dominer $P(v,w)$ (mais w ne peut pas). L'item 4 considère le cas où w peut dominer $P(v,w)$ (mais v ne peut pas). Dans chaque cas, il est possible de dominer $P(v,w)$ avec, ou bien v, w ou les deux selon le cas, ou bien un sommet de $\mathcal{D}$; mais il n'est pas possible de prédire le meilleur choix. Les gadgets ajoutés (respectivement v', w', y) permettent de simuler ce choix (cf. figure 2.7).

Formulons plusieurs remarques à propos de cette règle. D'abord, la règle suppose que $|P(v,w)| \geqslant 1$, cependant dans l'item 1, elle ajoute deux sommets. Nous montrons dans la preuve du lemme 8 que dans ce cas, la règle n'augmente pas la taille du graphe. De même il nous faudra montrer que l'ajout d'arêtes conserve la planarité. Ensuite, si $\mathcal{D}$ est vide alors selon le cas v, w ou les deux, sont nécessairement dominants. Dans les items 1, 3, 4, la règle ajoute des sommets de degré 1 qui forcent le choix. De tels sommets et leurs voisins seront supprimés par la règle 4 (cf. figure 2.8).

Enfin, nous signalons un formulation alternative de cette règle. Au lieu de la condition $|P(v,w)| \geqslant 1$, nous pourrions utiliser la condition $\mathcal{D} = \emptyset$

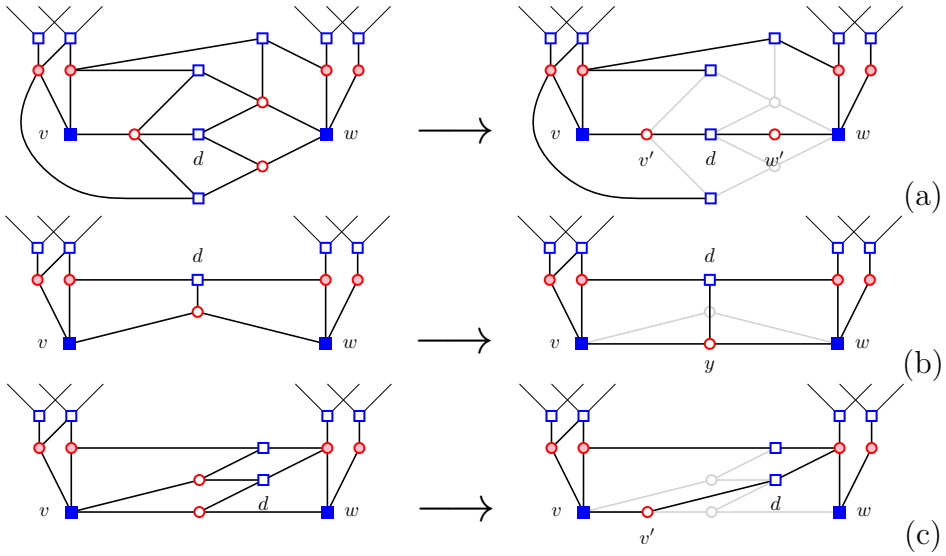


FIGURE 2.7 – Illustration de l'application de la règle 6 sur une paire v, w, avec $\mathcal{D} = \{d\}$. Dans tous les cas $P(v, w)$ peut être dominé, ou bien par d seul, ou bien par v, w, ou les deux selon le cas. Le gadget ajouté simule ces possibilités (a) Le cas 1. (b) Le cas 2. (c) Le cas 3. Remarquons que le plus souvent il reste des sommets bleus qui pourront être supprimés par la règle 2 ou la règle 7.

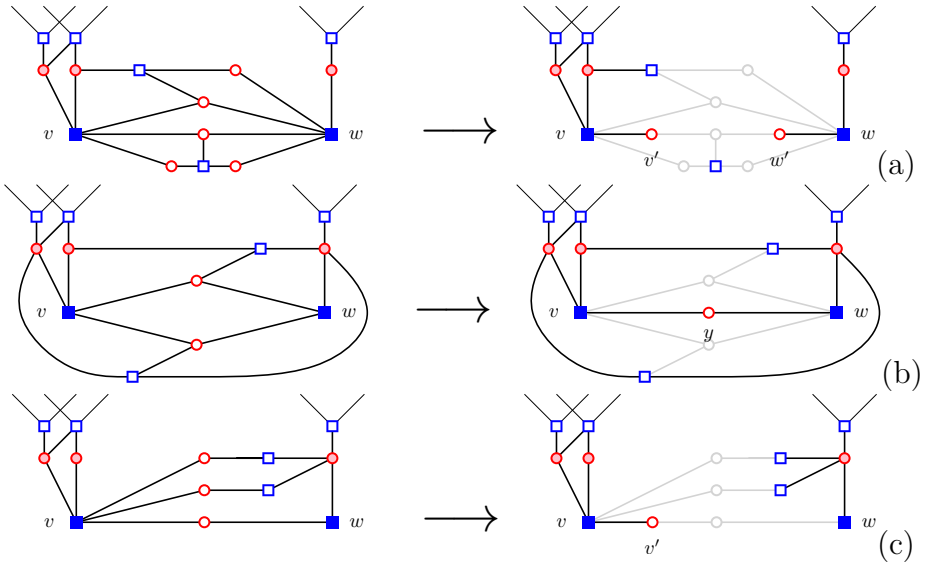


FIGURE 2.8 – Illustration de l'application du la règle 6 sur une paire v, w, avec $\mathcal{D} = \emptyset$. Selon les cas $P(v, w)$ est dominé par v, w, ou les deux. (a) L'item 1, le gadget ajouté est formé de deux sommets de degré 1. (b) L'item 2, le gadget est un sommet de degré 2. (c) L'item 3, le gadget est un sommet de degré 1. De même il reste des sommets bleus.

(qui est plus forte). Ainsi nous n'aurions pas à considérer les sommets $d \in \mathcal{D}$ dans chaque item, ce qui serait une formulation plus simple. Cependant, avec un telle formulation, l'application de la règle serait soumise un une condition plus contrainte, et par conséquent ne pourrait pas être appliqué dans des configurations où, avec la formulation pour laquelle nous avons opté, nous pouvons supprimer des sommets. En cela notre règle est plus précise. Nous obtenons aussi une meilleure borne et une preuve plus simple pour la proposition 3.

Lemme 8

Soit G un graphe bicoloré, soit $v, w \in V_B$. Si G' est le graphe obtenu par application de la règle 6 sur v, w, alors G admet un dominant rouge-bleu de taille k si et seulement si G' en admet un.

Démonstration. Nous montrons d'abord que la règle n'augmente pas la taille de l'instance et qu'elle conserve la planarité. Dans l'item 1 la règle ajoute deux sommets, mais d'après la condition de ce cas, il existe dans G deux sommets rouges privés $r \in N(v) \setminus N(w)$ et $r' \in N(w) \setminus N(v)$. Donc $|P(v, w)| \geqslant 2$. Dans tous les items, si la règle ajoute un sommet rouge, alors il existe dans G un sommet $r \in P(v, w)$ dont le voisinage contient $\mathcal{D}$ plus v ou w, selon le cas ; c'est-à-dire tel que $N(r)$ contient le voisinage du gadget. Donc la planarité est préservée.

Nous montrons qu'un dominant rouge-bleu de G est aussi un dominant rouge-bleu de G' et inversement, ce qui prouve que les instances sont équivalentes.

Soit D un dominant rouge-bleu de G. Nous distinguons deux cas selon le nombre de sommets utilisés pour dominer $P(v, w)$.

Si $|D \cap N(P(v, w))| \geqslant 2$ (c'est-à-dire, si deux sommets sont utilisés pour dominer $P(v, w)$) alors nous pouvons supposer que $v, w \in D$. En effet, si tel n'est pas le cas, il existe deux sommets dominants $d, d' \in N(P(v, w))$. Puisque $N(d, d') \subseteq N(v, w)$, nous pouvons remplacer d, d' par v, w dans D. Donc supposons $v, w \in D$. Dans ce cas, tout sommet rouge ajouté par la règle est dominé par v ou par w. Par conséquent D est un dominant de G'.

Si $|D \cap N(P(v, w))| < 2$ alors nous distinguons les quatre cas de la règle.

1. Dans ce cas, il existe un sommet $d \in \mathcal{D}$ tel que $d \in D$. Or $v', w' \in N(d)$.
2. Dans ce cas, il existe $d \in \mathcal{D} \cup \{v, w\}$ tel que $d \in D$. Or $y \in N(d)$.
3. Dans ce cas, il existe $d \in \mathcal{D} \cup \{v\}$ tel que $d \in D$. Or $v' \in N(d)$.
4. Symétriquement au cas 3.

Dans tous les cas D est un dominant de G'.

Inversement, soit D' une dominant rouge-bleu de G'. Nous distinguons les quatre cas de la règle.

1. Dans ce cas, pour dominer v', w', il existe $d \in \mathcal{D}$ dans D', ou $v, w \in D'$.
2. Dans ce cas, pour dominer y, il existe $d \in \mathcal{D} \cup \{v, w\}$ dans D'.
3. Dans ce cas, pour dominer v', il existe $d \in \mathcal{D} \cup \{v\}$ dans D'.
4. Symétriquement au cas 3.

Or, dans tous les cas $P(v, w) \subseteq N(d)$. Par conséquent D' est un dominant de G.

□

Nous décrivons finalement une dernière règle qui, comme la règle 2, supprime des sommets bleus qui ne seront pas utilisés. Comme nous l'avons remarqué en décrivant la règle 6, les sommets importants pour dominer $N(v, w)$ sont les sommets v, w et les sommets de $\mathcal{D}$. Cette règle supprime les autres sommets bleus.

Règle 7

Soit $v, w \in V_B$.

— Supprimer $N(P(v, w)) \setminus (\mathcal{D} \cup \{v, w\})$.

Pour dominer $P(v, w)$, il faut un sommet de $\mathcal{D}$, ou au moins deux sommets bleus ; or deux sommets peuvent toujours être remplacés par v, w (car $N(b, b') \subseteq P(v, w) \supseteq N(v, w)$ pour tous $b, b' \in N(P(v, w))$). Dans le cas où $P(v, w) = \emptyset$ la règle ne modifie pas le graphe.

Signalons que cette règle n'est pas nécessaire pour démontrer le noyau linéaire et qu'elle n'améliore pas la borne dans l'analyse du pire cas. Nous la formulons car elle nous permet de simplifier la preuve de la proposition 3.

Lemme 9

Soit G un graphe bicoloré. Si G' est le graphe obtenu par application de la règle 7, alors G admet un dominant rouge-bleu de taille k si et seulement si G' en admet un.

Démonstration. Soit D un dominant rouge-bleu de G. Si la règle supprime un sommet $b \in D$ alors $|D \cap N(P(v, w))| \geqslant 2$ (car b ne domine pas $P(v, w)$) ; $(D \setminus N(P(v, w))) \cup \{v, w\}$ est un dominant rouge-bleu de G'. Sinon, D est un dominant rouge-bleu de G'.

Inversement, soit D' un dominant rouge-bleu de G'. De façon evidente D' est un dominant rouge-bleu de G.

□

Complexité de la réduction

Nous prouvons ici que la réduction d'un graphe peut être faite en temps polynomial. Nous récapitulons l'algorithme de réduction dans l'algorithme 1.

Algorithme 1 : Réduction pour DOMINATION ROUGE-BLEU

pour chaque *arête* $e \in E(G)$ **faire**
  appliquer la règle 1;
tant que G *a été modifié à l'étape précédente* **faire**
  pour chaque *paire* $v, w \in V(G)$ **faire**
    appliquer la règle 2;
    appliquer la règle 3;
  pour chaque *sommet* $v \in V_B$ **faire**
    appliquer la règle 4 sur v;
  pour chaque *paire* $v, w \in V_B$ **faire**
    appliquer la règle 6.

Lemme 10
Soit G un graphe planaire. G peut être réduit par les règles 1, 2, 3, 4, 5, et 6, 7 en temps $O(|G|^4)$.

Démonstration. Nous posons $n = |G|$.

La règle 1 s'applique en temps $O(1)$ sur une arête e. Un graphe planaire est donc réduite par la règle 1 en temps $O(n)$. Remarquons que l'application des autres règles laisse le graphe réduit.

Les règles 2 et 3 s'appliquent en temps $O(n)$ sur une paire v, w. En effet, la comparaison des voisinages se fait en $O(n)$. Et la suppression du sommet se fait en $O(1)$.

La règle 4 s'applique en temps $O(\delta(v))$, sur un sommet v.

La règle 5 s'applique en temps $O(n)$ sur un sommet v. En effet, la construction du voisinage privé $P(v)$ se fait en $O(n)$: marquer toutes les arêtes incidentes aux sommets rouges de $N(v)$ en $\sum_{r \in N(v)} \delta(r) = O(n)$ et vérifier s'il existe une arête non marquée incidente à un sommet bleu de $N^2(v)$ en $\sum_{b \in N^2(v)} \delta(b) = O(n)$. Et les suppressions et ajouts de sommets se font en $O(\delta(v))$.

La règle 6 s'applique en temps $O(n)$ sur une paire v, w. De même que pour la règle 5 : la construction de $P(v, w)$ se fait en $O(n)$. Le calcul de $\mathcal{D}$ se fait en $\sum_{b \in N^2(v,w)} \delta(b) = O(n)$. Et la suppression des voisinages se fait en $O(\delta(v) + \delta(w))$ et l'ajout des gadgets en $O(n)$.

La règle 7 s'applique en temps $O(n)$, sur une paire v, w. De même que pour la règle 6 : le calcul de $\mathcal{D}$ se fait en $O(n)$.

Dans le pire cas, il faut tester si les règles 2 et 3 s'appliquent à chaque paire v, w, en temps $O(n^2 \cdot n)$. Ensuite, tester si la règle 4 s'applique à chaque sommet v en temps $O(n \cdot n)$. Enfin, tester si la règle 6 s'applique à chacune des n^2 paires de sommets, en temps $O(n^2 \cdot n)$. Toujours dans le pire cas, une seule règle a pu être appliquée une seule fois et n'a supprimé qu'un seul sommet. Les tests sont donc répéter au plus n fois.

Le graphe est donc réduit en temps $O(n^4)$.

□

2.2.2 Analyse de la taille du noyau

Dans cette sous-section nous montrons qu'un graphe plan réduit par nos règles a un nombre de sommets linéaire en la taille d'un dominant rouge-bleu (nous ne considérerons donc que des graphes bicolorés, réduits, et plans).

Étant donné un dominant rouge-bleu D, nous montrons l'existence d'une D-décomposition en régions $\Re$ maximale telle que :

— $\Re$ a au plus $3|D|$ régions,

— $\Re$ couvre tous les sommets,

— chaque région $R \in \Re$ contient au plus 14 sommets (plus ses pôles).

Chacune de ces caractéristiques est respectivement prouvée dans les propositions 1, 2, et 3 ci-dessous. Nous rappelons que nous considérons des 4-régions. Lorsque nous parlons de voisinages incomparables (au sens où il n'y a pas d'inclusion entre les deux ensembles), nous faisons implicitement référence à la règle 2 ou 3.

Nous mentionnons tout d'abord le fait que tout sommet est sur un chemin de longueur 4 entre deux sommets dominants. Intuitivement, un sommet dominant a toujours un voisin rouge ; ce sommet rouge étant non privé a un voisin bleu, ce sommet bleu étant incomparable avec le premier à un autre voisin rouge ; ce nouveau sommet rouge est dominé par un dernier sommet. Nous utilisons ce fait dans les preuves des trois propositions. Cependant, pour chacune des propositions, nous devons spécifier la démonstration. Pour cette raison, nous ne donnons pas de preuve formelle ici.

Taille de la décomposition

La proposition suivante borne le nombre de régions d'une décomposition maximale. Nous montrons que le graphe sous-jacent à la décomposition est

mince, par contradiction avec la maximalité de la décomposition. Nous utilisons ensuite les résultats de la section 2.1 pour en déduire la borne. Remarquons que dans l'énoncé de la proposition nous faisons l'hypothèse que le graphe est réduit, mais dans la preuve nous n'utilisons que la réduction par la règle 2.

Proposition 1

Soit G un graphe bicoloré plan réduit. Soit D un dominant rouge-bleu de G de taille au moins 3. Il existe une D-décomposition maximale $\Re$ de G telle que $|\Re| \leqslant 3|D| - 6$.

Démonstration. Soit $\Re$ la décomposition maximale exhibée par le lemme 2. D'après le lemme 4 $G_\Re$ est planaire ; nous considérons le plongement $\pi_\Re$ fourni par la preuve du lemme 4 à partir du plongement π de G. Nous prouvons que le multigraphe $G_\Re$ sous-jacent à $\Re$ est mince. C'est-à-dire que, pour chaque paire d'arêtes e_1, e_2 d'extrémités identiques v, w, il existe un sommet de D dans chacun des deux ouverts délimités par $\pi_\Re(e_1)$ et $\pi_\Re(e_2)$. Nous appliquerons ensuite le lemme 3 pour obtenir la borne sur le nombre de régions dans $\Re$.

Soient $e_1, e_2 \in E(G_\Re)$ d'extrémités identiques v, w, correspondant respectivement aux régions $R_1, R_2 \in \Re$ et dont les supports sont respectivement les chemins p_1, p_2 (c'est-à-dire, les chemins ayant servis à construire $\pi_\Re(e_1)$ et $\pi_\Re(e_2)$, cf. lemme 4). Soit $O_\Re$ un des deux ouverts délimités par e_1 et e_2. Soit O l'ouvert délimité par les plongements de p_1 et de p_2 correspondant à $O_\Re$ (au sens où, la même orientation est choisie pour parcourir e_1, e_2 d'une part et p_1, p_2 d'autre part), et privé des fermés R_1 et R_2. Notons que O est un sous-ensemble du complémentaire (dans le plan) de $R_1 \cup R_2$ (cf. figure 2.9).

Remarquons que la différence symétrique entre $O_\Re$ et O est composé de régions (R_1, R_2, ou les deux) plus des fermés du plan utilisés pour le replongement de e_1 et e_2 (les ensembles $C(E)$ tel que $e_1 \in E$ ou $e_2 \in E$, cf. lemme 4). Par conséquence, pour prouver qu'un sommet de D est dans $O_\Re$ il nous suffit de le prouver dans O.

Remarquons que O n'est pas vide, sinon R_1 et R_2 ont un chemin délimitant en commun, ce qui contredit la maximalité de $\Re$ puisque R_1 et R_2 pourraient être remplacés par $R_1 \uplus R_2$. Par contradiction, supposons maintenant que O ne contient pas de sommets de D. Nous distinguons trois cas.

— Si O intersecte une région $R_3 \in \Re$, alors R_3 est nécessairement une vw-région, sinon R_3 serait sécante avec R_1 ou R_2. Dans ce cas nous pouvons récursivement utiliser la même argumentation sur les ouverts délimités par R_1, R_3 ou R_2, R_3. Pour chaque $i \in \{1, 2, 3\}$, considérons un voisin $u_i \in V(R_i) \cap N(v)$ de v dans la région R_i. Nous avons $[u_1 < u_3 < u_2]_v$.

Puisque le degré de v est fini, le nombre de régions considérées est fini et la récursion termine.

— Sinon, si O ne contient pas de sommet bleu, alors les sommets rouges (s'il y en a) sont nécessairement dominés par v ou w. Le fermé $R_1 \cup O \cup \partial(O)$ est une région dont le bord est un chemin délimitant de R_1 et un chemin délimitant de R_2. Contradiction avec la maximalité de $\mathfrak{R}$.

— Sinon, O contient un sommet bleu $b \notin D$. Nous montrons que b appartient à un chemin (v, r, b, r', w), avec $r, r' \in V_R$. En effet, $N(b)$ est incomparable avec $N(w)$ donc il existe un voisin $r \in N(b)$ dominé par v ; de même $N(b)$ est incomparable avec $N(v)$ donc il existe un voisin $r \in N(b)$ dominé par w. Remarquons que r et r' sont dans O ou sur $\partial(O)$, donc le chemin (v, r, b, r', w) est une région non sécante avec R_1 et R_2. Contradiction avec la maximalité de $\mathfrak{R}$.

Par conséquent, le multigraphe $G_{\mathfrak{R}}$ est mince. Par hypothèse $|D| \geqslant 3$, donc, d'après le lemme 3, $G_{\mathfrak{R}}$ a au plus $3|D| - 6$ arêtes et puisque $G_{\mathfrak{R}}$ a une arête pour chaque région de $\mathfrak{R}$, la décomposition $\mathfrak{R}$ a au plus $3|D| - 6$ régions. □

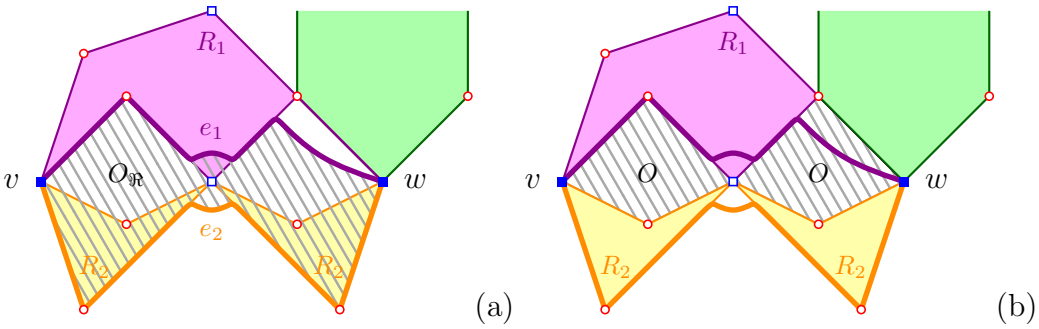


FIGURE 2.9 – Illustration des sous-ensembles ouverts du plan définis dans la preuve de la proposition 1. Remarquons que les deux arêtes e_1, e_2 de $G_{\mathfrak{R}}$ associées aux régions R_1, R_2 ont été replongées deux fois : à cause du sommet commun à R_1 et R_2 et à cause de l'arête commune à R_1 et une troisième région. (a) L'ouvert $O_{\mathfrak{R}}$ délimité par e_1 et e_2. (b) L'ouvert O délimité par deux chemins délimitants (respectivement de R_1 et R_2).

Extérieur de la décomposition

La proposition suivante énonce qu'une décomposition maximale couvre tous les sommets du graphe. Nous montrons que si un sommet n'est pas couvert par la décomposition alors celui-ci est sur un chemin entre deux

sommets dominants qui peut être considéré comme une région dégénérée, ce qui contredirait la maximalité. Remarquons que dans l'énoncé de la proposition nous faisons l'hypothèse que le graphe est réduit, mais dans la preuve nous n'utilisons pas la réduction par les règles 6 et 7.

Proposition 2

Soit G un graphe bicoloré plan réduit. Soit $D \neq \emptyset$ un dominant rouge-bleu de G. Soit $\Re$ une D-décomposition de G. Si $\Re$ est maximale alors $V_B \cup V_R = V(\Re)$.

Démonstration. Nous prouvons séparément la propriété pour les sommets rouges, pour les sommets dominants et pour les autres sommets bleus. Les démonstrations se font par l'absurde : si un sommet n'est pas couvert alors il appartient à une région qui peut être ajoutée à $\Re$.

Considérons d'abord les sommets rouges. Puisque D domine V_R, nous pouvons considérer que $V_R = \bigcup_{v \in D} N(v)$; il nous suffit de prouver que $N(v) \subseteq V(\Re)$ pour tout $v \in D$. Soit $v \in D$ et $r \in N(v)$. Par l'absurde supposons que $r \notin V(\Re)$. Nous montrons qu'il existe un chemin $p = (v, r, \dots w)$ avec $w \in D$ de longueur au plus 4 qui ne croise pas de région de $\Re$ (un tel chemin, considéré comme une région dégénérée, peut être ajouté à $\Re$, contredisant la maximalité).

Puisque G est réduit par les règles 4 et 5, $r \notin P(v)$, il existe $b \in N(r)$ distinct de v. Supposons que $b \in D$. Par hypothèse $r \notin V(\Re)$ (ce qui implique que v et b sont sur des bords de régions), donc le chemin $p = (v, r, b)$ ne croise pas de région de $\Re$. Dans ce cas, p est une région qui peut être ajoutée à $\Re$; contradiction avec la maximalité de $\Re$.

Supposons donc $b \notin D$. Nous considérons $\mathfrak{P}_b$ l'ensemble des chemins délimitants de régions dans $\Re$, qui contiennent b (notons que b n'est pas une extrémité).

Supposons d'abord $\mathfrak{P}_b \neq \emptyset$. Soit $(v_{\min}, r_{\min}, b, r_{\max}, w_{\max}) \in \mathfrak{P}_b$ tel que :

- $r_{\min}$ soit minimum autour de b à partir de r (par hypothèse $r \neq r_{\min}$),
- *et* $v_{\min}$ soit minimum autour de $r_{\min}$ à partir de b,
- *et* $r_{\max}$ soit maximum autour de b à partir de r ($r_{\min}$ étant fixé),
- *et* $w_{\max}$ soit maximum autour de $r_{\max}$ à partir de b.

Par définition de région, une des extrémités $v_{\min}$ ou $w_{\max}$ est distincte de v. Sans perte de généralité supposons $v \neq w_{\max}$. Par construction, le chemin $p = (v, r, b, r_{\max}, w_{\max})$ ne croise aucune région de $\Re$ (cf. figure 2.10). Dans ce cas, p est une région qui peut être ajoutée à $\Re$; contradiction avec la maximalité de $\Re$.

Supposons ensuite $\mathfrak{P}_b = \emptyset$. Puisque les voisinages de v et b sont incomparables, il existe $r' \in N(b) \setminus N(v)$ qui est dominé par un sommet distinct de v. En considérant $\mathfrak{P}_{r'}$ l'ensemble des chemins délimitants de régions dans $\Re$ qui contiennent r' et par des arguments similaires, nous trouvons un sommet $w \in D$ tel que $p = (v, r, b, r', w)$ ne croise aucune région de $\Re$. Dans ce cas, p est une région qui peut être ajoutée à $\Re$; contradiction avec la maximalité de $\Re$. Donc $V_R \subseteq V(\Re)$.

Considérons ensuite les sommets dominants. Soit $v \in D$. Par l'absurde ; supposons que $v \notin V(\Re)$. Puisque G est réduit par les règles 4 et 5, il existe un sommet rouge $r \in N(v)$. Nous avons montré que $r \in V(\Re)$, donc ce sommet est sur le bord d'une région de $\Re$. Une fois encore, en considérant $\mathfrak{P}_r$ et par des arguments similaires à ceux ci-dessus, nous pouvons trouver $b, r', w \in V(G)$ tels que $p = (v, r, b, r', w)$ peut être ajouté à $\Re$, contradiction avec la maximalité de $\Re$. Donc $D \subseteq V(\Re)$.

Considérons enfin les autres sommets bleus. Soit $b \in V_B \setminus D$. Par l'absurde ; supposons que $b \notin V(\Re)$. Puisque G est réduit par la règle 2 il existe deux sommets rouges $r, r' \in N(b)$ dominés par deux sommets distincts. De plus nous avons montré que $r, r' \in V(\Re)$, donc ces deux sommets sont sur le bord de régions de $\Re$. En considérant $\mathfrak{P}_r$ et $\mathfrak{P}_{r'}$ et par des arguments similaires à ceux ci-dessus, nous pouvons trouver $v, w \in D$ tels que $p = (v, r, b, r', w)$ peut être ajouté à $\Re$, contradiction avec la maximalité de $\Re$. Donc $V_B \subseteq V(\Re)$. □

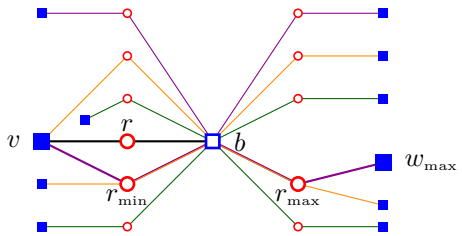


FIGURE 2.10 – Illustration de la construction du chemin $(v, r, b, r_{\max}, w_{\max})$ dans la proposition 2 (lorsque $b \in V(\Re) \setminus D$). Remarquons que la région $(v, r_{\min}, b, r_{\max}, w_{\max})$ partage deux arêtes $\{r_{\min}; b\}$, $\{b; r_{\max}\}$ avec une autre région et que $v = v_{\min}$. Pour facilité la lecture de la figure nous n'avons représenté que des régions dégénérées en chemins.

Taille d'une région

Comme annoncé au début de la section, la dernière étape avant de formuler le théorème consiste à montrer que les régions ont taille constante. Afin

d'obtenir la borne la plus petite possible nous devons utiliser simultanément la topologie de la région, les distances aux pôles de la région, l'incomparabilité des voisinages, et la structure imposée par la règle 6 ; d'où une preuve très technique.

Dans le but de simplifier la lecture de la preuve, nous démontrons d'abord qu'une région a un nombre constant de sommets, sans chercher la meilleure constante. Une telle proposition suffit à montrer l'existence d'un noyau linéaire pour DOMINATION ROUGE-BLEU. Cette première preuve se base sur deux arguments classiques : la caractérisation de Kuratowski et Wagner [46, 54] des graphes planaires, qui énonce que les graphes planaires sont exactement les graphes excluant K_5 et $K_{3,3}$ comme mineur ; et le théorème de Sperner [48], qui étant donné un ensemble de n éléments, énonce que la plus grande famille de sous-ensembles incomparables a taille $\binom{n}{n/2}$.

Proposition

Soit G un graphe bicoloré plan réduit. Soit D un dominant rouge-bleu de G et $v, w \in D$. Toute vw-région R est de taille constante.

Démonstration. Soit R une vw-région. Par définition, il y a au plus 4 sommets rouges et 4 sommets bleus sur ∂R. S'ils existent, nous appelons b et b' les deux sommets bleus distincts de v et w. Nous bornons séparément le nombre de voisins privés de v, w, de voisins non privés, et de sommets bleus dans R.

Considérons les sommets privés. Puisque G est réduit par la règle 6, $|P(v, w)| \leqslant 2$.

Considérons les sommets non privés. Remarquons que pour tout sommet $\bar{r} \in \bar{P}(v, w)$ strictement dans R, $N(\bar{r})$ contient au moins 3 sommets bleus :

— v ou w, car $\bar{r}$ est dans R et par définition $V(R) \cap V_R \subseteq N(v, w)$;

— *et* b ou b', car $\bar{r}$ est non privé et par définition $N^2(\bar{r}) \not\subseteq V(R)$;

— *et* un autre sommet, car $N(\bar{r})$ est incomparable avec $N(r)$ pour r sur ∂R.

Sans perte de généralité considérons les sommets dont le voisinage contient v et b. Par l'absurde, supposons qu'il y ait au moins trois tels sommets r_1, r_2, r_3. Remarquons qu'il faut au moins trois sommets bleus distincts (et distincts de v, b) pour que les voisinages de r_1, r_2, r_3 soient incomparables (car $\binom{3}{1} = 3$). Soit b_1, b_2, b_3, de tels sommets, respectivement voisins de r_1, r_2, r_3. Observons que, pour $i \in [1, 3]$, les voisinages de v et de b_i sont incomparables, donc il existe un sommet $\tilde{r}_i \in N(b_i) \cap N(w)$ (possiblement les $\tilde{r}_i$ sont confondus). Nous contractons les arêtes $\{b_i; \tilde{r}_i\}, \{\tilde{r}_i; w\}$ sur w. Après contraction, le graphe induit par $\{v, w, b, r_1, r_2, r_3\}$ forme un $K_{3,3}$; contradiction avec la planarité ([46, 54]). Il en est de même pour $\{v, b'\}$, $\{w, b'\}$, et $\{w, b'\}$, donc $|\bar{P}(v, w) \cap V(R)| \leqslant 4(2 + 1)$.

Considérons les sommets bleus. D'après les considérations précédentes, il y a au plus 14 sommets rouges dans R. Rappelons que les sommets bleus ont des voisinages incomparables entre eux, d'après le théorème de Sperner [48], $|V_B \cap V(R)| \leqslant \binom{14}{7} + 4$.

□

La proposition suivante borne le nombre de sommets contenus dans une région (par une petite constante). Nous montrons que soit la règle 6 a été appliquée sur les pôles de la région, et par construction il y a peu de sommets ; soit le voisinage privé des pôles est vide, et par planarité et incomparabilité des voisinages il y a peu de sommets.

Proposition 3

Soit G un graphe bicoloré plan réduit. Soit D un dominant rouge-bleu de G et $v, w \in D$. Toute vw-région R contient au plus 14 sommets distincts de ces pôles.

Démonstration. Soit R une vw-région. Il apparaîtra clairement au cours de la preuve que le pire cas est obtenu lorsque ∂R est composé de deux chemins disjoints de longueur 4, c'est-à-dire lorsqu'il contient le maximum de sommets, que nous appelons par la suite $v, r_v, b, r_w,\ w, r'_w, b', r'_v$ (intuitivement, un bord plus petit réduirait le nombre de combinaisons possibles pour le voisinage d'un sommet interne, or les voisinages sont incomparables).

Rappelons que, puisque G est réduit par les règles 2 et 3, les voisinages sont incomparables deux à deux. Nous devons toujours considérer que les sommets sur ∂R ont un voisinage plus large ou incomparable avec tout sommet strictement dans R, car nous ne connaissons pas leur voisinage en dehors de la région.

Nous bornons séparément le nombre de voisins privés de v, w, de voisins non privés, et de sommets bleus dans R.

D'abord, puisque G est réduit par la règle 6, $|P(v, w)| \leqslant 2$.

Ensuite, nous bornons le nombre de sommets rouges non privés strictement dans R. Soit $\bar{r} \in \bar{P}(v, w)$ strictement dans R. Sans perte de généralité, $N(\bar{r})$ contient v (car $\bar{r} \in N(v, w)$), b (car $\bar{r} \in \bar{P}(v, w)$) et au moins un autre sommet bleu (car $N(\bar{r}) \not\subseteq N(r_v)$). Dans la zone délimitée par les chemins (v, r_v, b) et $(v, \bar{r}, b)$ il n'y aucun sommet :

- aucun sommet bleu, car son voisinage serait inclus dans $N(v)$;
- aucun sommet non privé, car son voisinage serait inclus dans $N(\bar{r})$.
- aucun sommet privé, car le graphe est réduit par la règle 4 ;

Plus intuitivement, $\bar{r}$ isole r_v du reste de la région. Il y a donc au plus un sommet non privé adjacent à v, b. Par symétrie le même raisonnement

s'applique aux sommets non privés adjacents respectivement à v, b', à w, b, et à w, b'.

Donc, en comptant les sommets internes et ceux du bord, $|\bar{P}(v,w) \cap V(R)| \leqslant 4 \cdot 2 = 8$.

Il apparaîtra clairement au cours de la preuve que le pire cas est obtenu lorsque $|\bar{P}(v,w) \cap V(R)| = 8$, c'est-à-dire lorsqu'il y a un maximum de sommets non privés ; nous appelons par la suite $\bar{r}_v, \bar{r}_w, \bar{r}'_w, \bar{r}'_v$ les quatre sommets strictement dans R (intuitivement, $\bar{r}_v$ remplace r_v). Soit R' le fermé du plan délimité par les chemins $(v, \bar{r}_v, b, \bar{r}_w, w)$ et $(w, \bar{r}'_w, b', \bar{r}'_v, v)$ (cf. figure 2.11). Notons que $R' \notin \Re$ peut être vu comme une vw-région. Observons aussi que nous venons de montrer que $V(R \setminus \partial R) \subseteq V(R')$.

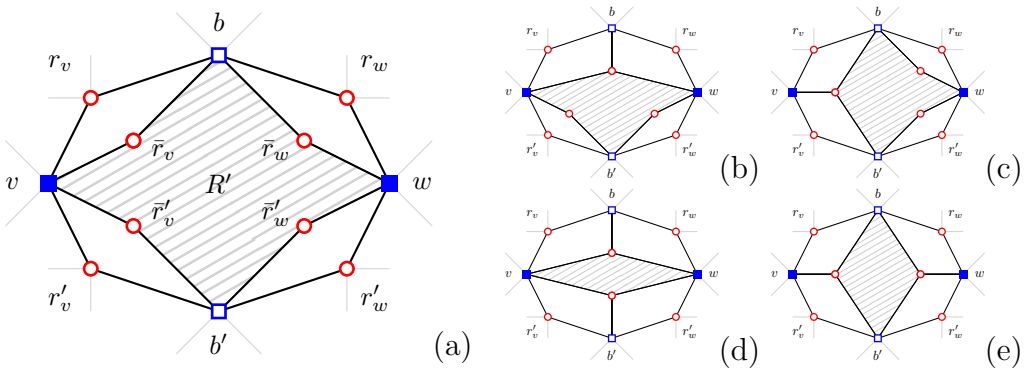


FIGURE 2.11 – Illustration des configurations possibles pour les sommets non privés pour la proposition 3. La zone R' contient tous les autres sommets de R, les 4 autres zones sont vides. (a) Le pire cas. (b-e) Toutes les autres configurations maximales, à symétrie prés. Les configurations restantes sont obtenues en supprimant un sommet de $\bar{P}(v,w)$. Nous pouvons remarquer que l'argument pour borner le nombre de sommets bleus peut être répété dans chaque cas. Intuitivement, un sommet strictement dans R remplace un sommet du bord.

Enfin, il reste à borner le nombre de sommets bleu strictement dans R'. Puisque le graphe est réduit selon la règle 7, il suffit de borner $\mathcal{D}$. Pour cela, nous distinguons cinq cas qui correspondent au cas où la règle 6 n'est pas appliquée, plus les quatre cas de celle-ci.

0. Supposons que $P(v,w) = \emptyset$ (la règle 6 ne s'applique pas). Rappelons que dans ce cas $\mathcal{D} = V_B$. Nous comptons les sommets bleus strictement dans R'

   En tenant compte de la topologie des régions et de l'incomparabilité des voisinages, il n'y a qu'un petit nombre de configurations possibles qui

peuvent être énumérées (cf. figure 2.12). Il apparaît que, dans le pire cas, il y a deux sommets strictement dans R'. Plus intuitivement, un sommet bleu $\bar{b}$ est nécessairement adjacent à $\bar{r}_v$ et $\bar{r}'_w$ (ou, symétriquement, adjacent à $\bar{r}'_v$ et $\bar{r}_w$) ; le chemin $(\bar{r}_v, \bar{b}, \bar{r}'_w)$ sépare R' en deux zones ce qui limite le nombre de combinaisons. Remarquons que, ici, nous utilisons la topologie des régions et non la planarité. En effet il serait possible d'ajouter d'autres sommets s'ils étaient dessinés à l'extérieur de la région.

Donc, en comptant les sommets internes et ceux du bord, $|V_B \cap V(R)| \leqslant 2 + 4 = 6$.

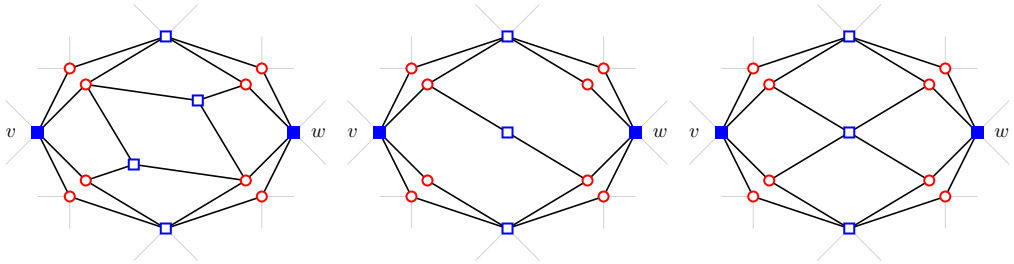


FIGURE 2.12 – Illustration de toutes les configurations possibles, à symétrie près,dans la proposition 3, lorsque la règle 6 ne s'applique pas.

1. Supposons $P(v, w) \nsubseteq N(v)$ et $P(v, w) \nsubseteq N(w)$ (la règle 6 item 1 s'applique). Dans ce cas $P(v, w) = \{v', w'\}$. Nous montrons que $|\mathcal{D}| \leqslant 2$ en exhibant un $K_{3,3}$.

   Par l'absurde, supposons qu'il existe $d_1, d_2, d_3 \in \mathcal{D}$. Par définition, d_1, d_2, d_3 sont adjacents à v' et w'. De plus, d_1, d_2, d_3 sont chacun adjacent à un sommet dans $\bar{P}(v, w) \cap V(R')$ différent, pour que leurs voisinages soient incomparables entre eux. En contractant les arêtes de $\partial R'$ en un sommet c et en supprimant les arêtes $\{v; v'\}$ et $\{w; w'\}$ nous obtenons que $G[\{d_1, d_2, d_3, v', w', c\}]$ est un $K_{3,3}$ mineur de $G[V(R)]$, contradiction avec la planarité ([46, 54] ; cf. figure 2.13). Plus intuitivement, d_2 est isolé par les chemins (v', d_1, w') et (v', d_3, w').

   Donc, en comptant les sommets internes et ceux du bord, $|V_B \cap V(R)| \leqslant 2 + 4 = 6$.

2. Supposons $P(v, w) \subseteq N(v)$ et $P(v, w) \subseteq N(w)$ (la règle 6 item 2 s'applique). Dans ce cas $P(v, w) = \{y\}$. Nous montrons que $|\mathcal{D}| \leqslant 2$ en exhibant encore un $K_{3,3}$.

   Par l'absurde, supposons qu'il existe $d_1, d_2, d_3 \in \mathcal{D}$. Par définition, d_1, d_2, d_3 sont adjacents à y. De plus, d_1, d_2, d_3 sont chacun adjacent

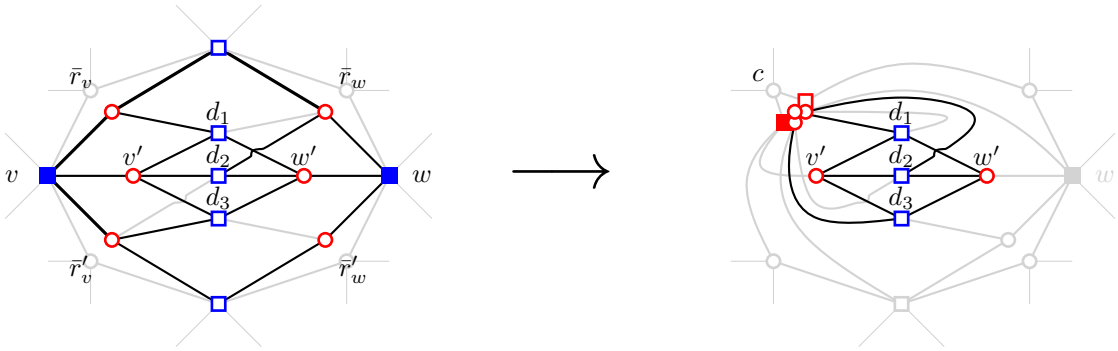


FIGURE 2.13 – Exibition du mineur $K_{3,3}$ dans la proposition 3, lorsque la règle 6 item 1 s'applique.

à un sommet dans $\bar{P}(v,w) \setminus N(w)$, par incomparabilité avec $N(w)$; et un dans $\bar{P}(v,w)\setminus N(v)$, par incomparabilité avec $N(v)$ (remarquons que ces trois paires doivent aussi être incomparables). En contractant, d'un coté $\{v;\bar{r}_v\}$ et $\{v;\bar{r}'_v\}$ sur v, et de l'autre coté $\{w;\bar{r}_w\}$ et $\{w;\bar{r}'_w\}$ sur w, nous obtenons un $K_{3,3}$ mineur de $G[V(R)]$, contradiction avec la planarité ([46, 54] ; cf. figure 2.14). Plus intuitivement, le chemin (v,y,w) sépare R' en deux zones ce qui limite le nombre de combinaison.

Donc, en comptant les sommets internes et ceux du bord, $|V_B \cap V(R)| \leqslant 2+4=6$.

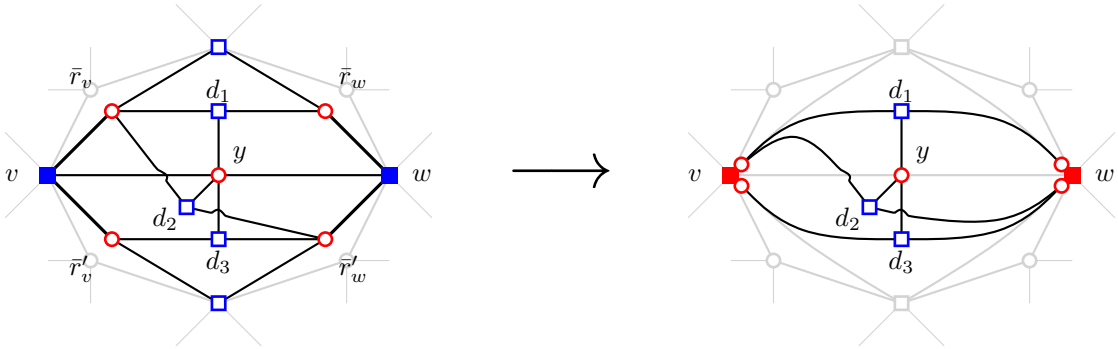


FIGURE 2.14 – Exibition du mineur $K_{3,3}$ dans la proposition 3, lorsque la règle 6 item 2 s'applique.

3. Supposons $P(v,w) \subseteq N(v)$ et $P(v,w) \nsubseteq N(w)$ (la règle 6 item 3 s'applique). Dans ce cas $P(v,w) = \{v'\}$. Nous montrons que $|\mathcal{D}| \leqslant 3$ en exhibant des $K_{3,3}$.

Par l'absurde, supposons qu'il existe $d_1, d_2, d_3, d_4 \in \mathcal{D}$. Par définition, d_1, d_2, d_3, d_4 sont adjacents à v'. De plus, d_1, d_2, d_3, d_4 sont chacun adjacent à un sommet dans $\bar{P}(v,w) \setminus N(v)$, par incomparabilité avec $N(v)$.

Observons d'abord qu'au plus deux sommets parmi les quatre sont adjacents à un même sommet de $\bar{P}(v,w) \setminus N(v)$. En effet, par l'absurde, supposons que d_1, d_2, d_3 soient adjacents à $\bar{r}_w$ (ou, symétriquement, adjacents à $\bar{r}'_w$). Puisque d_1, d_2, d_3 ont des voisinages incomparables entre eux, d_1, d_2, d_3 sont chacun adjacent à un sommet dans $\bar{P}(v,w) \setminus \{\bar{r}_w\}$. En contractant, $\{v; \bar{r}_v\}$ et $(v, \bar{r}'_v, b', \bar{r}'_w)$ sur v, nous obtenons un $K_{3,3}$ mineur de $G[V(R)]$, contradiction avec la planarité ([46, 54] ; cf. figure 2.15). Intuitivement, d_2 est une fois encore isolé par les chemins $(v', d_1, \bar{r}_w)$ et $(v', d_3, \bar{r}_w)$.

Sans perte de généralité supposons donc que d_1, d_2 sont adjacents à $\bar{r}_w$, et que d_3, d_4 est adjacent à $\bar{r}'_w$. Puisque $N(d_1)$ et $N(d_2)$ sont incomparables, d_1, d_2 sont chacun adjacent à un somment de $\bar{P}(v,w) \setminus N(w)$. De façon symétrique, d_3, d_4 sont chacun adjacent à un sommet de $\bar{P}(v,w) \setminus N(w)$. En contractant d'un coté $\{v; \bar{r}_v\}$ et $\{v; \bar{r}'_v\}$ sur v et de l'autre coté $\{w; \bar{r}_w\}$ et $\{w; \bar{r}'_w\}$ sur w, nous obtenons un $K_{3,3}$ mineur de $G[V(R)]$, contradiction avec la planarité ([46, 54] ; cf. figure 2.15). Plus intuitivement, d_2 est isolé par les chemins $(v', d_1, \bar{r}_w, w)$ et $(v', d_4, \bar{r}'_w, w)$, il est donc adjacent à r'_w ; ensuite d_3 est isolé par les chemins $(v', d_2, \bar{r}'_w)$ et $(v', d_4, \bar{r}'_w)$.

Donc, en comptant les sommets internes et ceux du bord, $|V_B \cap V(R)| \leqslant 3 + 4 = 7$.

4. Supposons $P(v,w) \not\subseteq N(v)$ et $P(v,w) \subseteq N(w)$ (la règle 6 item 4 s'applique). Symétriquement à l'item 3.

En sommant sur les ensembles $P(v,w), \bar{P}(v,w), V_B$, la région R contient au plus 16 sommets ($2 + 8 + 6 = 16$ dans l'item 1, ou $1 + 8 + 7 = 16$ dans les items 3 et 4). C'est-à-dire, au plus 14 sommets distincts de v et w (cf. figure 2.16).

□

Remarquons que l'argumentation de l'item 3 peut être utilisée pour les autres items (car le gadget de cet item est celui qui impose le moins de contraintes), mais cela augmenterait la borne. Remarquons aussi que, étant donnés v, w, une seul vw-région peut atteindre la borne car, lorsque nous comptons les sommets de $P(v,w)$ et $\mathcal{D}$, nous n'avons pas besoins de nous restreindre à l'intersection avec $V(R)$.

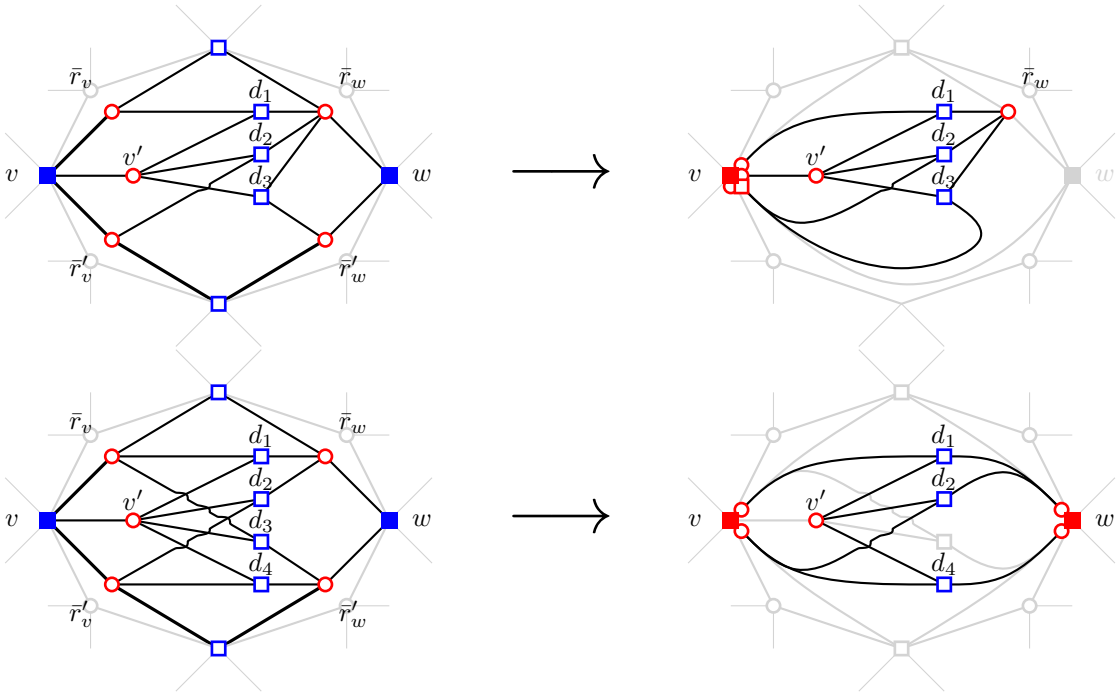


FIGURE 2.15 – Exibition du mineur $K_{3,3}$, dans la proposition 3, lorsque la règle 6 item 3 s'applique.

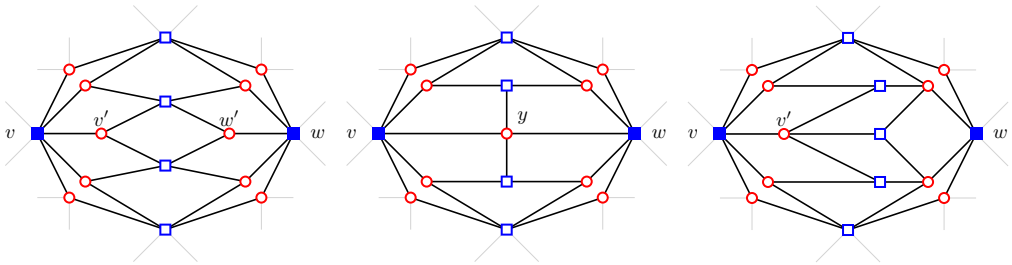


FIGURE 2.16 – Illustration des pires configurations possibles, dans pour la proposition 3, pour chacun des cas de la règle 6. Le pire cas correspond à l'item 1.

Noyau linéaire dans les graphes planaires

Nous pouvons maintenant utiliser les trois propositions pour prouver notre résultat.

Théorème 2

Le problème Domination Rouge-Bleu, restreint aux graphes planaires et paramétré par k la taille de la solution, admet un noyau linéaire de taille au plus $43k$.

Démonstration. Nous montrons qu'il existe un algorithme polynomial qui pour chaque instance planaire (G, k), ou bien répond négativement, ou bien renvoie une instance équivalente (G', k') avec $k' \leqslant k$ et $|G'| \leqslant 43k'$.

Soit G le graphe bicoloré plan de l'instance et G' le graphe réduit par les règles 1, 2, 3, 4, 5, 6, et 7 à partir de G. D'après les lemmes 5, 7, et 8, G admet un dominant rouge-bleu de taille au plus k si et seulement si G' en admet un de taille au plus k' (où $k' \leqslant k$, dépend de combien de fois la règle 4 a été appliquée). D'après le lemme 10, la réduction s'exécute en temps polynomial. Donc la transformation est bien une réduction polynomiale.

Supposons que G' (et donc G) ait une solution. Soit D' un dominant rouge-bleu de G' de taille $|D'| \leqslant k'$. Remarquons que si $|D'| < 3$ alors G' a au plus 1 sommet :

- si $|D'| = 0$ alors ou bien G' est le graphe vide, ou bien G' a un unique sommet bleu (G' n'a pas de sommet rouge et est réduit par la règle 2) ;
- $|D'| \neq 1$, car G' est réduit par la règle 4 ;
- $|D'| \neq 2$, car G' est réduit par la règle 6.

Nous pouvons donc supposer que $|D'| \geqslant 3$ (ce qui nous permet d'appliquer la proposition 1). D'après les propositions 1, 2,et 3, G' a taille au plus $14(3k' - 6) + k' \leqslant 43k'$. Par contraposée, si G' a plus de $43k'$ sommets alors c'est une instance négative.

Remarquons que, de façon équivalente, l'algorithme peut répondre négativement ou retourner une instance trivialement négative, par exemple $(G'' = (\emptyset \cup \{r\}, \emptyset), k = 0)$. En conséquence de quoi, l'algorithme est bien une extraction de noyau qui, étant donné une instance (G, k) renvoie un noyau (G', k') de taille au plus $43k'$. □

Notons que notre résultat implique un binoyau pour le problème Couverture de Faces (qui consiste à trouver un ensemble de sommets incidents à toutes les faces d'un graphe plan) puisque Couverture de Faces se réduit à Domination Rouge-Bleu en considérant le graphe radial. La

réduction est la suivante : étant donné un graphe G instance de Couverture de Faces, nous construisons le graphe G' instance de Domination Rouge-Bleu tel que $V(G')$ est l'union de $V(G)$, coloré en bleu, avec $F(G)$, coloré en rouge ; et $E(G')$ est l'ensemble des paires $\{v; f\}$ tel que le sommet v est incident à la face f dans G.

2.3 Noyau pour Domination Totale

Dans cette section nous construisons un noyau linéaire pour le problème Domination Totale dans les graphes planaires. Nous nous basons encore une fois sur la méthode de décomposition en région, avec des 3-régions. Étant donné un graphe, le problème Domination Totale consiste à trouver un ensemble de sommets tel que tout sommet du graphe admet un voisin dans l'ensemble solution.

Définition 52 : dominant total
Un dominant total est un ensemble de sommets $D \subseteq V(G)$ tel que $N(D) = V(G)$.

⌟

Domination Totale :
instance : un graphe $G = (V(G), E(G))$ et un entier k ;
paramètre: l'entier k ;
question : existe-t-il un dominant total de taille au plus k ?

Cette domination est totale au sens où un sommet ne peut pas se dominer lui même, il doit avoir un voisin dans la solution (d'où l'usage du voisinage ouvert $N(D)$). En conséquence de quoi, un dominant total induit un graphe sans sommet isolé. Remarquons qu'un dominant total est en particulier un dominant, et donc, pour un graphe donné, la taille d'un dominant minimum sera toujours inférieure (ou égale) à celle d'un dominant total. Intuitivement, il faudrait ajouter des sommets à un dominant pour le rendre total ; ce n'est cependant pas la meilleure manière d'obtenir un dominant total, en effet, les éléments de la solution peuvent être organisés de manière à économiser des sommets, il s'en suit qu'un dominant minimum n'est pas nécessairement un sous-ensemble d'un dominant total.

Le problème Domination Totale est connu pour être NP-complet [34, 24, 58] (y compris dans les graphes planaires). Paramétré par la taille de la solution, il est W[2]-complet [19], mais FPT [2, 29] dans les graphes planaires.

Le problème Domination Totale rentre parfaitement dans les cadres des méta-théorèmes [8, 34], ce qui implique l'existence d'un noyau linéaire dans les graphes planaires et plus généralement dans les classes de graphes peu denses. En particulier il rentre dans le cadre des méta-noyaux par programmation dynamique [33] décrit dans le chapitre 3, ce qui implique une extraction de noyau constructive. Néanmoins, nous avons jugé intéressant de construire un noyau *ad hoc* pour ce problème ; en effet, le cadre présenté dans le chapitre 3 permet de construire un noyau, mais avec une constante

non optimale. Afin d'obtenir une constante explicite et raisonnable, nous construisons un noyau conçu et ajusté spécifiquement pour le problème.

Théorème

Le problème Domination Totale, restreint aux graphes planaires et paramétré par k la taille de la solution, admet un noyau linéaire de taille $508 \cdot k$.

La qualification de petite pour notre constante est certes discutable, en particulier comparée à la constante fournie pour Domination Rouge-Bleu. Cependant, il convient de rappeler, d'une part, qu'une instance de Domination Rouge-Bleu peut être supposée biparti ; d'autre part que Domination Totale a une contrainte additionnelle par rapport à Domination, qui, intuitivement, rend le problème plus difficile. Nous soulignons que notre constante est du même ordre que celle fournie par Alber, Fellows et Niedermeier pour Domination, bien que celle-ci ait été améliorée par la suite (335 [3], puis 67 [12]), et que celle pour Domination Connexe [39, 47].

Quoique basées sur le même principe que la réduction pour Domination Rouge-Bleu, les règles présentées dans cette section sont notablement plus compliquées. Nous soulignons deux différences majeures. D'une part, lorsqu'un sommet est identifié comme dominant, il ne peut pas être supprimé du graphe avec son voisinage. En effet, le voisinage en question peut contenir un ou plusieurs sommets dominants qui n'ont pas encore été identifiés. De plus, il nous faut encore nous assurer que le sommet considéré est lui-même dominé. Il s'en suit que le paramètre n'est pas modifié par les règles. D'autre part, nous utilisons la notion de planarité lors de l'application de la règle 10 (et pas seulement pour l'analyse de la taille comme c'était le cas dans la section 2.2). Par conséquent nous devons supposer que notre graphe est plongé avant de faire la réduction. Remarquons de plus que les règles de réduction décrites ici semblent se prêter volontiers à une généralisation pour de nombreuses autres variantes de Domination. En effet, pour déterminer comment réduire les graphes à l'intérieur d'une région, la seconde règle simule toutes les manières (locales) de dominer totalement la région considérée. En d'autres termes, pour appliquer la seconde règle, il faut connaître tous les dominants totaux partiels de la région (nous avions déjà mentionné cette idée dans la section 2.2, ici l'usage est explicite). Il semble naturel de pouvoir élargir la notion de dominant partiel à une notion de solution partielle, qui dépendrait du problème.

De même que dans la section précédente, la description du noyau se fait en deux étapes : dans la sous-section 2.3.1 nous décrivons les règles de réduction

qui constituent l'extraction de noyau ; dans la sous-section 2.3.2 nous bornons la taille de l'instance réduite en nous basant sur la méthode de décomposition en régions.

2.3.1 Réduction de Domination Totale

Dans cette sous-section nous présentons les règles de réduction pour Domination Totale. Encore une fois, ces règles sont destinées à être itérées jusqu'à obtenir un graphe réduit (stable par application des règles). L'itération des règles constitue l'extraction de noyau. Comme précédemment (dans la section 2.2) l'algorithme s'articule principalement autour de deux règles : l'une s'applique sur un seul sommet, l'autre sur une paire de sommets. Ces règles suppriment les sommets non significatifs du voisinage des sommets considérés. Une troisième règle complète la réduction effectuée par la seconde ; rappelons qu'à cause de cette règle nous devons supposer que notre graphe est plan. Par la suite, la première règle nous permettra de montrer qu'il reste peu de sommets en dehors des régions, la seconde qu'il y a peu de sommets dans les régions.

Règle pour un sommet seul

Nous décrivons d'abord la règle qui s'applique à un seul sommet : si le sommet considéré est identifié comme étant nécessairement dominant, alors une partie de son voisinage peut être remplacée par un gadget qui force le sommet à être dans la solution.

Nous définissons ici une notion similaire à celle de voisinage privé (cf. définition 49)

Définition 53 : partition du voisinage d'un sommet
Le voisinage $N(v)$ d'un sommet v est partitionné en trois sous-ensembles :

$$N_1(v) = \{u \in N(v) \mid N(u) \setminus (N(v) \cup \{v\}) \neq \emptyset\},$$
$$N_2(v) = \{u \in N(v) \setminus N_1(v) \mid N(u) \cap N_1(v) \neq \emptyset\},$$
$$N_3(v) = N(v) \setminus (N_1(v) \cup N_2(v)).$$

Nous notons $N_{i,j}(v) = N_i(v) \cup N_j(v)$ pour $i, j \in [1, 3]$. ⌟

Ces trois ensembles jouent un rôle similaire à celui des voisinages privés et non privés. Les sommets de $N_1(v)$ (non privés) voient le reste du graphe, ils peuvent faire partie de l'ensemble solution (c'est pour cette raison que $N(v) \cup v$ ne peut pas être supprimé). Les sommets de $N_2(v)$ servent de

zone tampon, ils peuvent être dominés par des sommets de $N_1(v)$ ou par v, mais ne sont pas dominants. Les sommets de $N_3(v)$ (privés) ne peuvent être dominés que par v, mais pas par des sommets de $N_1(v)$ (pour cette raison, s'ils existent, nous pouvons supposer que v est dans la solution).

Règle 8

Soit $v \in V(G)$ un sommet. Si $N_3(v) \geqslant 1$:

— supprimer $N_{2,3}(v)$,

— ajouter un sommet v' et une arête $\{v; v'\}$.

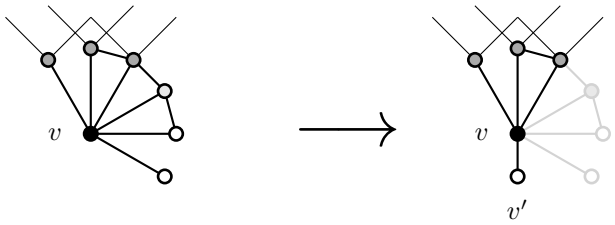


FIGURE 2.17 – Illustration de l'application du la règle 8 sur un sommet v. La règle supprime une partie de $N(v)$ et ajoute un sommet v'.

Remarquons que si certains sommets de $N_{2,3}(v)$ étaient sélectionnés dans le dominant (à la place de v), ce choix ne serait jamais le meilleur au sens où v dominerait toujours plus de sommets ($N(v) \supseteq N(N_{2,3}(v))$). De plus, tout sommet dominant dans $N_{2,3}(v)$ doit aussi être dominé, par un sommet qui dominerait aussi v. Par conséquent, s'il existe un dominant, nous pouvons supposer qu'il contient v.

Le gadget ajouté v' simule l'ensemble $N_3(v)$. D'une part, il force à choisir v dans la solution. D'autre part, si v était dominé par un sommet de $N_3(v)$, il peut être dominé par v' (rappelons que le dominant est total). Comme nous l'avons déjà dit, le sommet v n'est pas supprimé et donc le paramètre reste inchangé (cf. figure 2.17).

Lemme 11

Soit G un graphe, soit $v \in V(G)$. Si G' est le graphe obtenu par application de la règle 8 sur v, alors G admet un dominant total de taille k si et seulement si G' en admet un.

⌞ ⌟

Démonstration. À partir d'un dominant total de G, nous construisons un dominant total de G' de taille inférieure ou égale ; et inversement.

Soit D un dominant total de G de taille k. Nous pouvons supposer que $v \in D$. En effet, si tel n'est pas le cas, par hypothèse $N_3(v) \neq \emptyset$, donc il existe un dominant $u \in N_{2,3}(v)$ et un autre $w \in N(u)$. Puisque $N(u) \subseteq N(v)$ nous pouvons remplacer u par v dans D, et v est dominé par w. Donc, supposons $v \in D$. Nous construisons D' à partir de D tel que : si la règle supprime des sommets dominants alors $D' = D \setminus N_{2,3}(v) \cup \{v'\}$; sinon $D' = D$. Si la règle a supprimé un sommet u de D, alors $N(u) \setminus \{v\}$ est dominé dans G' par v. De plus, v' est dominé par v ; et v est dominé par $v' \in D'$. Sinon, v' est dominé dans G' par v ; et v par un sommet de $N_1(v)$. Par conséquent $|D'| \leqslant |D|$.

Inversement, Soit D' un dominant total de G' de taille k, nous savons que $v \in D'$ car v' doit être dominé. Nous construisons D à partir de D' tel que : si $v' \in D'$ alors $D = D' \setminus \{v'\} \cup \{u\}$, avec $u \in N_3(v)$ quelconque ; sinon $D = D'$. La règle n'a supprimé que des voisins de v qui sont dominés par v dans G. Si $v' \in D'$ alors v est dominé par u. Sinon, v est dominé dans G par un sommet de $N_1(v)$. Par conséquent $|D| = |D'|$. □

Remarquons que si $N_1(v) \neq \emptyset$, le choix de v' dans le dominant n'est pas le meilleur puisque n'importe quel sommet de $N_1(v)$ dominerait plus de sommets. Nous avons jugé plus simple pour la preuve d'utiliser v' pour dominer v ; cela nous permet de plus de souligner le fait que v doit être (totalement) dominé.

Règle pour une paire de sommets

Nous décrivons maintenant la règle qui s'applique à une paire de sommets : de la même façon, si un des deux sommets considérés est nécessairement dominant, alors une partie du voisinage des deux sommets est remplacée par un gadget qui force le sommet à être dans la solution. Dans certains cas, ces deux sommets ne suffisent pas à dominer totalement, il nous faut donc utiliser un gadget plus élaboré. Pour trouver ce gadget nous considérons toutes les façons de dominer le voisinage avec un petit nombre de sommets ; ainsi, nous sommes sûrs que la réduction ne perd pas d'information.

Définition 54 : partition du voisinage d'une paire

Le voisinage $N(v, w)$ d'une paire de sommets v, w est partitionné en trois sous-ensembles :

$$N_1(v, w) = \{u \in N(v, w) \mid N(u) \setminus (N(v, w) \cup \{v, w\}) \neq \emptyset\},$$
$$N_2(v, w) = \{u \in N(v, w) \setminus N_1(v, w) \mid N(u) \cap N_1(v, w) \neq \emptyset\},$$
$$N_3(v, w) = N(v, w) \setminus (N_1(v, w) \cup N_2(v, w)).$$

Nous notons $N_{i,j}(v, w) = N_i(v, w) \cup N_j(v, w)$ pour $i, j \in [1, 3]$. ⌟

Encore une fois, $N_1(v,w)$ joue le rôle des sommets non privés, connectés avec le reste du graphe ; $N_2(v,w)$ est la zone tampon ; et $N_3(v,w)$ correspond aux sommets privés, qui doivent être dominés. Intuitivement, nous voudrions encore réduire lorsque $N_3(v,w) \neq \emptyset$. Mais ici les sommets de $N_3(v,w)$ ne sont pas nécessairement dominés par v et w ; le choix d'un, deux, ou trois sommets dans $N_{2,3}(v,w)$ peut être meilleur que v, w.

Nous pouvons aisément observer que, au pire, dominer $N_3(v,w)$ utilisera 4 sommets : v, w et deux autres sommets pour dominer v et w respectivement. Lors de la réduction nous voulons conserver les possibilités de dominer avec 1,2 ou 3 sommets (qui seraient de meilleurs choix). Pour cela nous allons considérer tous les ensembles qui dominent $N_3(v,w)$ de taille au plus 3. Ensuite la réduction supprimera autant de sommets que possible tout en faisant en sorte que les dominants de $N_3(v,w)$ existent toujours et restent un meilleur choix que v, w.

Nous formalisons l'idée d'un ensemble qui domine $N_3(v,w)$ par la notion de dominant partiel. Puis nous définissons quels seront les dominants partiels que nous voulons considérer.

Définition 55 : dominant total partiel

Soit $S \subseteq V(G)$. Un dominant total *partiel* de S est un ensemble de sommets $D_S \subseteq N[S]$ tel que $N(D_S) \supseteq S$.

Soit $v, w \in V(G)$. Nous considérons certains dominants totaux partiels de $N_3(v,w)$ de taille au plus 3 ; nous notons :

$$\begin{aligned}
\mathcal{D} &= \{D_{N_3} \subseteq N_{2,3}(v,w) \mid N_3(v,w) \subseteq N(D_{N_3}),\ |D_{N_3}| \leqslant 3\},\\
\mathcal{D}_v &= \{D_{N_3} \subseteq N_{2,3}(v,w) \cup \{v\} \mid N_3(v,w) \subseteq N(D_{N_3}),\ |D_{N_3}| \leqslant 3,\ v \in D_{N_3}\},\\
\mathcal{D}_w &= \{D_{N_3} \subseteq N_{2,3}(v,w) \cup \{w\} \mid N_3(v,w) \subseteq N(D_{N_3}),\ |D_{N_3}| \leqslant 3,\ w \in D_{N_3}\}.
\end{aligned}$$

Nous notons $\bigcup \mathcal{D}_v$ (respectivement, $\bigcup \mathcal{D}_w$) l'ensemble $\bigcup \mathcal{D}_v = \bigcup_{D_{N_3} \in \mathcal{D}_v} D_{N_3}$.

⌟

Formulons plusieurs remarques à propos de ces définitions. Si D_S est un dominant total partiel de S, alors les sommets de $D_S \cap S$ doivent être dominés par un autre sommet, mais pas ceux dans $D_S \setminus S$. Autrement dit, les sommets dominants dans S ne doivent pas être isolés. L'ensemble $\mathcal{D}_v$ (respectivement, $\mathcal{D}_w$) contient les ensembles de sommets pouvant dominer totalement $N_3(v,w)$ en utilisant v (respectivement, w) et au plus deux autres sommets. L'ensemble $\mathcal{D}$ contient les dominants totaux partiels de $N_3(v,w)$ sans v ni w.

La règle 9 réduit le voisinage d'une paire v, w selon que v ou w peut être utilisé pour dominer $N_3(v, w)$ (en conservant les dominants totaux partiels intéressants). Intuitivement, si $\mathcal{D}_v$ (respectivement $\mathcal{D}_w$) n'est pas vide, cela signifie que v peut être utilisé pour dominer $N_3(v, w)$, sinon utiliser w est toujours un meilleur choix et nous simplifions $N(w)$.

Règle 9

Soit v, w deux sommets. Si $\mathcal{D} = \emptyset$:

1. si $\mathcal{D}_v = \emptyset$ et $\mathcal{D}_w = \emptyset$:
   - supprimer $N_{2,3}(v, w)$,
   - ajouter les sommets v', w' et les arêtes $\{v; v'\}, \{w; w'\}$,
   - s'il existe un voisin commun à v et w dans $N_3(v, w)$, ajouter un sommet y et les arêtes $\{v; y\}, \{w; y\}$;
2. si $\mathcal{D}_v \neq \emptyset$ et $\mathcal{D}_w \neq \emptyset$:
   - supprimer $(N_{2,3}(v, w) \cap N(v) \cap N(w)) \setminus (\bigcup \mathcal{D}_v \cup \bigcup \mathcal{D}_w)$
   - pour tous sommets $d, d' \in (\bigcup \mathcal{D}_v \cup \bigcup \mathcal{D}_w) \cap (N(v) \cap N(w))$ distincts, si $N(d) \setminus (N(v) \cap N(w)) \subseteq N(d') \setminus (N(v) \cap N(w))$, supprimer d,
   - si un sommet est supprimé, ajouter un sommet y et les arêtes $\{v; y\}, \{w; y\}$;
3. si $\mathcal{D}_v \neq \emptyset$ et $\mathcal{D}_w = \emptyset$:
   - supprimer $(N_{2,3}(v, w) \cap N(v)) \setminus \bigcup \mathcal{D}_v$,
   - pour tous sommets $d, d' \in \bigcup \mathcal{D}_v \cap N(v)$ distincts, si $N(d) \setminus N(v) \subseteq N(d') \setminus N(v)$, supprimer d,
   - si un sommet est supprimé, ajouter le sommet v' et l'arête $\{v; v'\}$;
4. si $\mathcal{D}_v = \emptyset$ et $\mathcal{D}_w \neq \emptyset$:
   - faire symétriquement à l'item 3.

Dans l'item 1, nous pouvons supposer que v et w sont dominants ; nous pouvons alors supprimer $N_{2,3}(v, w)$ qui n'est pas utile pour dominer et qui est dominé par v et w. Dans l'item 2, nous ne pouvons supposer ni que v ni que w est dans la solution, mais nous savons qu'au moins un des deux l'est ; dans ce cas, nous simplifions $N(v) \cap N(w)$ qui peut être dominé par v ou w indifféremment. Dans l'item 3 (respectivement, l'item 4) nous pouvons supposer que v est dans la solution ; nous simplifions alors $N(v)$ qui est dominé par v.

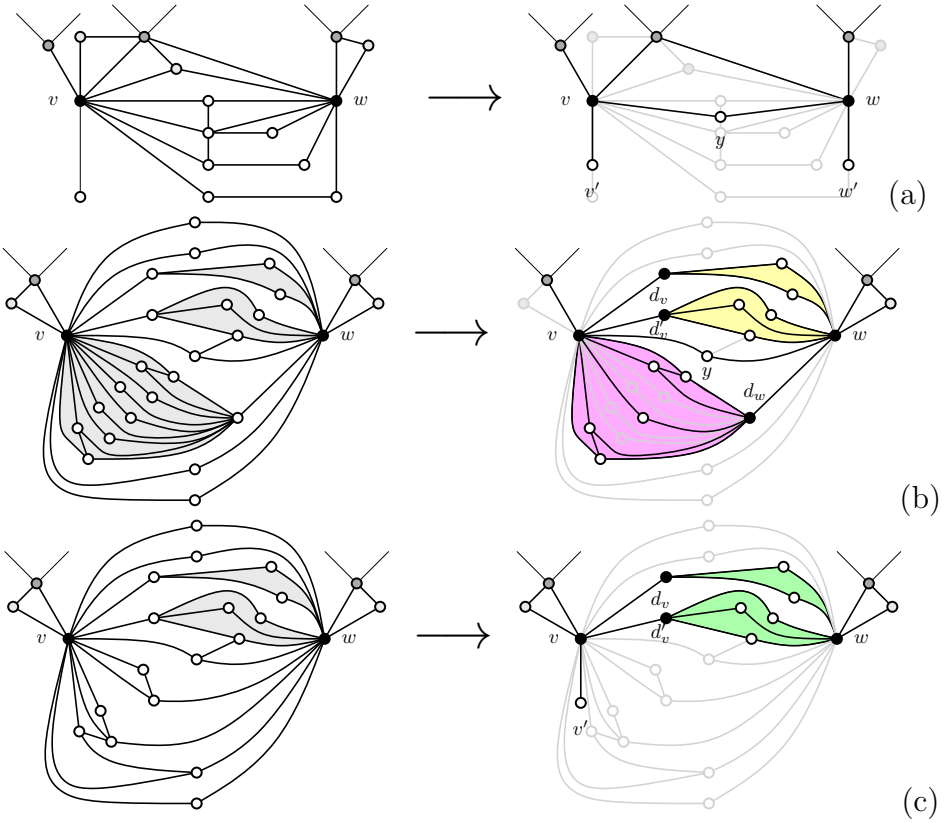


FIGURE 2.18 – Illustration de l'application du la règle 9 sur deux sommets v, w puis de la règle 10 si nécessaire. (a) Dans l'item 1 une partie de $N(v, w)$ est supprimée du graphe, et les sommets v', w', y sont ajoutés. (b) Dans l'item 2 la règle 9 supprime $N(v) \cap N(w)$ et le sommet y est ajouté, puis la règle 10 réduit une vd_w-région simple. (c) Dans le cas 3 une partie de $N(v)$ est supprimée, et le sommet v' est ajouté. Nous avons représenté les régions simples qui doivent être considérées par la règle 10 pour que nous puissions ensuite établir la borne sur la taille du noyau.

Dans tous les cas, lorsque nous simplifions un voisinage, nous faisons en sorte de conserver les sommets qui font partie d'un dominant total partiel. Le gadget v' (respectivement w') nous permet de forcer le choix de v (lorsque $\mathcal{D}_w = \emptyset$). Le gadget y nous permet de laisser le choix de v ou de w mais force au moins un des deux ; de plus y assure que la distance $d(v, w)$ n'a pas été augmentée.

Dans les cas où nous ne simplifions pas le voisinage $N(w)$ (respectivement $N(w)$) (lorsque $\mathcal{D}_v \neq \emptyset$), nous laissons de nombreux sommets qui ne peuvent pas être dominant et qui ne nous servent qu'a forcer le choix d'un dominant total partiel. Pour cette raison nous avons besoin de la règle 10. Intuitivement, nous voulons que la règle 10 soit appliquée sur w (respectivement, sur v) et les sommets d'un dominant partiel de $\mathcal{D}_v$ afin de réduire le voisinage $N(w)$ qui n'a pas été modifié par cette règle (cf. figure 2.18).

Avant de démontrer la correction de la règle 9, nous montrons quelques faits que nous utiliserons dans la démonstration. Intuitivement, ces faits soulignent les propriétés intéressantes d'un graphe sur lequel la règle 9 est appliquée sur v, w ; en particulier, ils montrent que nous pouvons toujours supposer que certains sommets dans les ensembles solutions. Dans ces faits, nous noterons G un graphe dans lequel la règle 9 est appliquée sur les sommets v, w et G' le graphe obtenu par cette réduction.

Le premier fait énonce que si la règle 9 peut être appliquée, alors nous pouvons supposer que v ou w appartient au dominant total de G que nous considérons.

Fait 1

Si $\mathcal{D} = \emptyset$ alors G admet une solution si et seulement s'il admet une solution contenant au moins un des deux sommets v, w.

Démonstration. Par hypothèse $\mathcal{D} = \emptyset$, donc tout dominant total de G doit contenir v ou w ou au moins 4 sommets de $N_{2,3}(v, w)$. Dans le dernier cas, les quatre sommets peuvent être remplacés par v, w et deux voisins de v et w respectivement. Nous pouvons donc supposer que v ou w appartient à la solution. □

Le second fait énonce que si nous ne pouvons pas utiliser w pour dominer dans G, alors nous pouvons supposer que v appartient au dominant total de G que nous considérons.

Fait 2

Si $\mathcal{D}_w = \emptyset$ (respectivement $\mathcal{D}_v = \emptyset$) et $\mathcal{D} = \emptyset$, alors G admet une solution

si et seulement s'il admet une solution contenant v (respectivement w). De plus $(N(v)\setminus N(w))\cap N_3(v,w) \neq \emptyset$ (respectivement $(N(w)\setminus N(v))\cap N_3(v,w) \neq \emptyset$).

Démonstration. Par hypothèse $\mathcal{D}_w = \emptyset$, donc aucun ensemble de la forme $\{w\}, \{w,u\}, \{w,u,u'\}$ avec $u,u' \in N_{2,3}(v,w)$ ne peut dominer $N_3(v,w)$. En particulier, v a un voisin dans G qui n'est pas voisin de w. Puisque $\mathcal{D} = \emptyset$, tout dominant total de G doit contenir v, ou au moins 4 sommets. Dans le deuxième cas (comme dans le fait 1), les quatre sommets peuvent être remplacés par v, w et deux voisins de v et w respectivement. Nous pouvons donc supposer que v appartient à la solution. □

Le troisième fait énonce (inversement) que si nous ne pouvions pas utiliser w dans G, alors nous pouvons supposer que v appartient au dominant total de G' que nous considérons.

Fait 3
Si $\mathcal{D}_w = \emptyset$ (respectivement $\mathcal{D}_v = \emptyset$) et $\mathcal{D} = \emptyset$, alors G' admet une solution si et seulement s'il admet une solution contenant v (respectivement w).

Démonstration. Si la règle 9 n'a pas modifié le graphe alors d'après le fait 2 $G' = G$ admet une solution contenant v. Sinon, la règle 9 a ajouté $v' \in V(G')$ avec $N(v') = \{v\}$, donc tout dominant total de G' doit contenir v pour dominé v'. □

Le dernier fait énonce que si un sommet de $N_{2,3}(v,w)$ est utilisé pour dominer, alors nous pouvons supposer que ce sommet fait partie d'un dominant total partiel. Autrement dit, les sommets ne faisant pas partie d'un dominant total partiel ne sont jamais un meilleur choix pour dominer.

Fait 4
Si $\mathcal{D}_v \neq \emptyset$ (respectivement $\mathcal{D}_w \neq \emptyset$) et $\mathcal{D} = \emptyset$, alors G admet une solution si et seulement s'il admet une solution ne contenant aucun sommet de $N_{2,3}(v,w) \setminus (\bigcup \mathcal{D}_v \cup \bigcup \mathcal{D}_w)$.

Démonstration. Considérons $u \in N_{2,3}(v,w) \setminus (\bigcup \mathcal{D}_v \cup \bigcup \mathcal{D}_w)$. Puisque $\mathcal{D} = \emptyset$, tout dominant total de G contenant u doit contenir au moins 4 sommets de $N[v,w]$. Dans ce cas, les quatre sommets peuvent être remplacés, ou bien par un ensemble $\mathcal{D}_v \cup \mathcal{D}_w$, ou bien par quatre sommets de $\{v,w\} \cup N_1(v,w)$. □

Nous pouvons maintenant prouver la correction de la règle 9.

Lemme 12

Soit G un graphe, soit $v, w \in V(G)$. Si G' est le graphe obtenu par application de la règle 9 sur v, w, alors G admet un dominant total de taille k si et seulement si G' en admet un.

Démonstration. Nous commençons par considérer D un dominant total de G afin de construire D' un dominant total de G' (plus petit ou de même taille). Nous prouvons indépendamment chacun des cas de la règle. Soit D dominant total de G.

1. Dans le cas ou $\mathcal{D}_v = \emptyset$ et $\mathcal{D}_w = \emptyset$ (cas de l'item 1).

   Nous construisons D' à partir de $D \cap V(G')$; selon que, respectivement, la règle supprime un sommet dans $D \cap N(v) \setminus N(w)$, dans $D \cap N(w) \setminus N(v)$, ou dans $D \cap N(v) \cap N(w)$, nous ajoutons les sommets v', w', y.

   D'après le fait 2 nous pouvons supposer que $v, w \in D$. D'une part, v', w', y sont dominés dans G' par v ou par w. D'autre part, si la règle a supprimé un sommet $u \in D$, alors, $u \in N_{2,3}(v, w)$ et par définition, $N(u) \subseteq N(v, w)$, et donc $N(u) \setminus \{v, w\}$ est dominé dans G' par v ou w ; les sommets v, w sont quant à eux dominés, ou bien par v', w', y (s'il ont été ajoutés), ou bien par $N_1(v, w)$ (sinon). Enfin, tous les autres sommets de G' sont dominés de la même façon que dans G. Il s'en suit que D' est un dominant total de G' tel que $|D'| \leqslant |D|$.

2. Dans le cas où $\mathcal{D}_v \neq \emptyset$ et $\mathcal{D}_w \neq \emptyset$ (cas de l'item 2).

   Soit D dominant total de G. Nous construisons D' à partir de $D \cap V(G')$; pour chaque sommet $d \in D$ supprimé par la règle, nous ajoutons un sommet $d' \in N(v) \cap N(w)$ tel que $N(d) \setminus (N(v) \cap N(w)) \subseteq N(d') \setminus (N(v) \cap N(w))$. D'après le fait 4, $d \in \bigcup \mathcal{D}_v \cup \bigcup \mathcal{D}_w$ et donc d' existe.

   D'après le fait 1, nous pouvons supposer que $v \in D$ ou $w \in D$. D'une part, y est dominé dans G' par v ou w (s'il a été ajouté). D'autre part, si la règle a supprimé un sommet $d \in D$ alors il existe $d' \in D'$ tel que $N(d) \setminus (N(v) \cap N(w)) \subseteq N(d') \setminus (N(v) \cap N(w))$ et donc $N(d)$ est dominé dans G' par d'. Il s'en suit que D' est un dominant total de G' tel que $|D'| \leqslant |D|$.

3. Dans le cas où $\mathcal{D}_v \neq \emptyset$ et $\mathcal{D}_w = \emptyset$ (cas de l'item 3).

   Soit D dominant total de G. Nous construisons D' à partir de $D \cap V(G')$; pour chaque sommet $d \in D$ supprimé par la règle, nous ajoutons un sommet $d' \in N(v)$ tel que $N(d) \setminus N(v) \subseteq N(d') \setminus N(v)$. D'après le fait 4, $d \in \bigcup \mathcal{D}_v$ et donc d' existe.

D'après le fait 2 nous pouvons supposer que $v \in D$. D'une part, v' est dominé dans G' par v (s'il a été ajouté). D'autre part, si la règle a supprimé un sommet $d \in D$, alors il existe $d' \in D'$ tel que $N(d) \setminus N(v) \subseteq N(d')$ et donc $N(d)$ est dominé dans G' par v, d'. Il s'en suit que D' est un dominant total de G' tel que $|D'| \leqslant |D|$.

4. Symétriquement à l'item 3.

Inversement, nous considérons D' un dominant total de G' afin de construire D un dominant total de G (de même taille). Nous prouvons indépendamment chacun des cas de la règle. Soit D' dominant total de G'

1. Nous construisons D à partir de $D' \setminus \{v', w', y\}$; selon que, respectivement, $v' \in D'$, $w' \in D'$, ou que $y \in D'$, nous ajoutons les sommets $u_v \in N(v) \setminus N(w)$, $u_w \in N(w) \setminus N(v)$, $u \in N(v) \cap N(w)$. Les sommets ajoutés existent d'après le fait 2.

   D'après le fait 3, $v, w \in D'$, et donc $v, w \in D$. D'une part, la règle ne supprime que des voisins de v ou w qui sont donc dominés dans G par v ou par w. D'autre part, les sommets v, w sont dominés dans G, ou bien par u_v, u_w, ou u (si l'un d'eux a été ajouté dans D), ou bien par $N_1(v, w)$ (sinon). Il s'en suit que D est un dominant total de G tel que $|D| = |D'|$.

2. Nous construisons D à partir de $D' \setminus \{y\}$; si $y \in D'$, nous ajoutons un sommet $u \in N(v) \cap N(w)$. Le sommet ajouté existe.

   D'après le fait 3, $v \in D'$ ou $w \in D'$ et donc $v \in D$ ou $w \in D$. D'une part, la règle ne supprime que des voisins communs de v et w qui sont donc dominés dans G. D'autre part, v et w sont dominés dans G, ou bien par u (s'il a été ajouté à D), ou bien par des sommets non modifiés (sinon). Il s'en suit que D est un dominant total de G tel que $|D| = |D'|$.

3. Nous construisons D à partir de $D' \setminus \{v'\}$; si $v' \in D'$, nous ajoutons un sommet $u \in N(v)$. Le sommet ajouté existe.

   D'après le fait 3 $v \in D'$, et donc $v \in D$. D'une part, la règle ne supprime que des voisins de v qui sont donc dominés dans G par v. D'autre part, v est dominé dans G, ou bien par u (s'il a été ajouté à D), ou bien par des sommets non modifiés (sinon). Il s'en suit que D est un dominant total de G tel que $|D| = |D'|$.

4. Symétriquement à l'item 3.

□

Nous décrivons maintenant la règle auxiliaire qui complète la règle 9. Comme nous l'avons déjà dit, intuitivement cette règle doit réduire la partie de $N_3(v, w)$ qui n'a pas été réduite par règle 9 (cf. figure 2.18). Pour énoncer

cette règle nous avons besoin de supposer que le graphe est plongé dans le plan ; car nous utilisons des 2-régions. Nous appelons les 2-régions des régions simples afin de conserver la dénomination d'Alber, Fellows et Niedermeier [3]. Nous rappelons qu'il s'agit d'une région dont tous les sommets sont des voisins communs de v et w (cf. figure 2.19).

Définition

Une *région simple* R est une 2-région. Ce qui implique que :

— les chemins délimitants sont de longueur au plus 2 ;

— ^{et} $V(R) \setminus \{v, w\} \subseteq N(v) \cap N(w)$.

⌟

Règle 10

Soit v, w deux sommets. S'il existe une vw-région simple R, telle que $|N_3(v, w) \cap V(R)| \geqslant 1$:

— supprimer $N_3(v, w)$

— ajouter un sommet z et les arêtes $\{v, z\}, \{w, z\}$

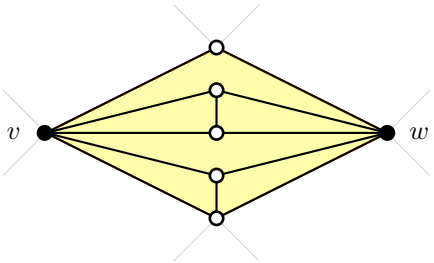


FIGURE 2.19 – Illustration d'une régions simple réduite par la règle 10 (cf. item 2.3.1.

Remarquons que nous utilisons une règle spécifique pour borner la taille des régions simples, tandis que pour la DOMINATION [3], cette borne est une conséquence de la seconde règle (sur un paire de sommets).

Nous montrons maintenant la correction de la règle 10.

Lemme 13

Soit G un graphe plan, soit $v, w \in V(G)$ et R une vw-région simple. Si G' est le graphe obtenu par application de la règle 10 sur R, alors G admet un dominant total de taille k si et seulement si G' en admet un.

⌞ ⌟

Démonstration. Soit D un dominant total de G. Nous construisons D' un dominant total de G' tel que : si la règle supprime des sommets de D, alors $D' = D \setminus (N_3(v, v) \cap V(R)) \cup \{z\}$; sinon, $D' = D$.

Puisque $N_3(v, w) \cap V(R) \neq \emptyset$ nous pouvons supposer que v ou w est un sommet dominant. En effet, tout dominant partiel doit contenir soit v soit w soit 2 sommets de $N(v, w)$. Dans le dernier cas, sans perte de généralité, les deux sommets peuvent être remplacés par v et un voisin de v. Supposons que $v \in D$. Si un sommet $u \in D$ est supprimé par la règle alors les voisins de u sont dominés dans G' par v, sauf v, w qui sont dominés par z. De plus z est dominé par v. Si la règle ne supprime pas de sommet alors v, w sont dominés dans G' par des sommets de $N_{1,2}(v, w)$. Dans les deux cas, D' est un dominant total de G' tel que $|D'| \leqslant |D|$.

Inversement, soit D' un dominant total de G'. nous construisons D tel que : si $z \in D'$ alors $D = D \setminus \{z\} \cup \{u\}$ avec $u \in N_3(v, w) \cap V(R)$; sinon, $D = D'$.

Puisque z doit être dominé $v \in D'$ ou $w \in D'$, sans perte de généralité supposons $v \in D'$. La règle n'a retiré que des sommets de $N_3(v, w) \cap V(R)$ qui sont dominés dans G par v. De plus, v, w sont dominés par u s'il a été ajouté à D, et par $N_{1,2}(v, w)$ sinon. Dans les deux cas, D est un dominant total de G tel que $|D| = |D'|$.

□

Pour en terminer avec la réduction du voisinage d'une paire de sommet, nous signalons quelques pistes qui pourraient permettre d'améliorer la réduction. D'abord, lorsque dans la définition 55, nous observons que quatre sommets au plus suffisent à dominer $N_3(v, w)$, nous devons préciser que v, w suffisent à dominer $N_3(v, w)$ et que les deux autres sommets servent seulement à dominer v, w. Ensuite, ayant une condition d'application plus contraignante, la règle 9, ne s'applique pas systématiquement, au contraire de la règle 6. Enfin, si la distance entre v et w est strictement inférieure à 3 alors nous avons besoin de moins de sommets pour dominer $N_3(v, w)$ et donc nous pouvons considérer des dominants totaux partiels plus petits. Nous pourrions formuler une règle qui prend en compte les ensembles de $\mathcal{D}$ et la distance. Néanmoins l'analyse du noyau (cf. proposition 6) nous laisse penser que ces formulations alternatives de la règle 9 n'apporteraient pas d'amélioration significative.

Complexité de la réduction

Nous montrons ici que la réduction est polynomiale en la taille du graphe. Nous reprenons l'ensemble de la réduction dans l'algorithme 2

Algorithme 2 : Réduction de Domination Totale

tant que G *a été modifié à l'étape précédente* **faire**
  pour chaque *sommet* $v \in V(G)$ **faire**
    appliquer la règle 8
  pour chaque *paire* $v, w \in V(G)$ **faire**
    appliquer la règle 9;
    soit $\mathcal{R}$ l'ensemble des vw-régions simples;
    pour chaque *région simple* $R \in \mathcal{R}$ **faire**
      appliquer la règle 10

Lemme 14

Soit G un graphe plan, G peut être réduit par les règles 8, 9, 10 en temps $O(|G|^3)$.

Démonstration. De nombreux points de cette preuve sont similaires à la preuve du lemme 10. Nous posons $n = |G|$.

La règle 8 s'applique en temps $O(\delta(v))$ sur un sommet v. En effet, la construction des trois voisinages se fait en $O(\delta(v))$. Construisons d'abord $N_1(v)$. Pour chaque sommet $u \in N(v)$, nous parcourons ses voisins jusqu'à trouver un sommet non voisin de v ; si un tel sommet existe alors $u \in N_1(v)$. La construction de $N_1(v)$ nécessite de parcourir au plus deux fois chaque arête de $G[N(v)]$ plus une arête hors de $G[N(v)]$ pour chaque sommet. Puisque $G[N(v)]$ est planaire, la construction de $N_1(v)$ se fait en $O(\delta(v))$. Construisons ensuite $N_2(v)$ et $N_3(v)$. Pour chaque sommet $u \in N(v) \setminus N_1(v)$, nous parcourons ses voisins jusqu'à trouver un sommet de $N_1(v)$; si un tel sommet existe alors $u \in N_2(v)$, sinon $u \in N_3(v)$. De même la construction de $N_2(v)$ et $N_3(v)$ se fait en $O(\delta(v))$. Ensuite, la suppression de $N_{2,3}(v)$ et l'ajout de v' se fait en $O(\delta(v))$.

La règle 9 s'applique en temps $O(\delta(v) + \delta(w))$. De même, la construction des trois voisinages et la modification de $N_{2,3}(v, w)$ se fait en $O(\delta(v) + \delta(w))$. La construction des ensembles $\mathcal{D}, \mathcal{D}_v, \mathcal{D}_w$ se fait avec un algorithme paramétré [2] en temps $2^{\sqrt{3}}(\delta(v) + \delta(w))$.

La règle 10 s'applique en temps $O(\delta(v) + \delta(w))$ En effet, la construction des vw-régions simples se fait en $O(\delta(v) + \delta(w))$: remarquons que les ordres cycliques autour de v et de w induisent un ordre cyclique partiel sur $V(G) \setminus \{v, w\}$; une région simple contient exactement les sommets de $N(v) \cap N(w)$ consécutifs.

Dans le pire cas, tester si la règle 8 s'applique sur tous les sommets se fait en temps $\sum_{v \in V(G)} O(\delta(v)) = O(n)$. Puis tester si les règles 9 et 10 s'ap-

pliquent sur chaque paire de sommets se fait en temps $\sum_{v,w \in V(G)} O(\delta(v) + \delta(w)) = O(n^2)$. Toujours dans le pire cas, une seule de ses règles a pu être appliquée et n'a supprimé qu'un seul sommet. La boucle principale peut donc être répétée au plus n fois. La réduction s'effectue donc en temps $O(n^3)$. □

2.3.2 Analyse de la taille du noyau

Dans cette sous-section nous montrons une borne (en fonction du paramètre k) sur la taille d'un graphe plan réduit par nos règles, ce qui en fait un noyau. Pour cela nous supposons que le graphe plan admet un dominant total D (de taille au plus k), et nous exhibons une D-décomposition en régions $\Re$ telle que :

— $\Re$ a au plus $3|D|$ régions,

— $\Re$ couvre tous les sommets sauf $122|D|$ d'entre eux,

— chaque région $R \in \Re$ contient au plus 159 sommets (plus ses pôles).

Comme pour le noyau de DOMINATION ROUGE-BLEU, nous montrons trois propositions qui correspondent aux trois caractéristiques ci-dessus. La proposition 4 découle des résultats de la section 2.1. La proposition 5 se prouve en couvrant les sommets restant par des régions simples. La proposition 6 se prouve avec la réduction par la règle 9.

Dans cette sous-section, nous considérons toujours un graphe G plan et réduit. Nous rappelons que nous considérons des 3-régions.

Nous montrons d'abord le lemme suivant qui borne la taille des régions simples. Nous utilisons ce lemme dans les propositions 5 et 6. Remarquons que, dans l'énoncé, nous supposons que le graphe est réduit, mais nous n'utilisons que la règle 10 dans la preuve.

Lemme 15

Soit G un graphe plan réduit. Toute région simple R contient au plus 5 sommets distincts de ses pôles.

Démonstration. Soit R une vw-région simple. Nous bornons séparément les sommets de R qui sont dans $N_1(v,w)$, $N_2(v,w)$, $N_3(v,w)$ (cf. figure 2.19).

Les sommets de $N_1(v,w) \cap V(R)$ sont nécessairement sur le bord de R, donc R en contient au plus 2. Les sommets de $N_2(v,w) \cap V(R)$ sont nécessairement adjacents à un sommet de $N_1(v,w) \cap V(R)$, par planarité chaque sommet de $N_1(v,w) \cap V(R)$ a au plus 1 tel voisin dans R. Les sommets de $N_3(v,w) \cap V(R)$ ont nécessairement été remplacés par 1 seul sommet car G est réduit par la règle 10.

□

Taille de la décomposition

La proposition suivante borne le nombre de régions dans la décomposition. Remarquons que dans cet énoncé nous n'avons pas à supposer que le graphe est réduit.

Proposition 4

Soit G un graphe plan. Soit D un dominant total de G de taille au moins 3. Il existe une D-décomposition maximale $\Re$ de G telle que $|\Re| \leqslant 3|D| - 6$.

Démonstration. Soit $\Re$ la décomposition maximale exhibée par le lemme 2. Nous prouvons que le multigraphe $G_\Re$ sous-jacent à $\Re$ est mince. D'après le lemme 4 $G_\Re$ est planaire. Il nous faut donc montrer que chacun des deux ouverts délimités par deux occurrences d'une arête multiple contient un sommet de D.

Nous montrons en fait que toute paire de vw-régions R_1, R_2 (avec des pôles identiques) délimite deux ouverts du plan contenant chacun un sommet de D. Remarquons d'abord que R_1 et R_2 ne peuvent pas partager un chemin du bord, sinon $R_1 \uplus R_2$ est une région, contradiction avec la maximalité de $\Re$. Soit O un de ces deux ouverts. Par l'absurde, supposons que O ne contienne pas de sommet dominant. Les sommets dans O (s'il y en a), sont dominés par v ou par w. Par conséquent, les régions R_1, R_2 et les régions de $\Re$ incluses dans O peuvent être remplacées par la région $R_1 \cup O \cup R_2$, contradiction avec la maximalité de $\Re$.

Puisque chaque arête de $G_\Re$ correspond à une région de $\Re$ et que la preuve du lemme 4 fournit un plongement qui ne change pas les positions des sommets relativement aux arêtes de $G_\Re$; le graphe $G_\Re$ est mince. D'après le lemme 3, $|\Re| \leqslant 3|D| - 6$. □

Extérieur de la décomposition

Nous bornons maintenant le nombre de sommets qui ne sont pas couverts par la décomposition, c'est-à-dire la taille de $V(G) \setminus V(\Re)$. Nous montrons, en utilisant la règle 8, que les sommets n'appartenant pas à une région peuvent être couverts par un petit nombre de régions simples, nous utilisons ensuite la règle 10. Remarquons que dans l'énoncé nous supposons le graphe réduit, mais nous n'utilisons pas la règle 9.

Proposition 5

Soit G un graphe plan réduit. Soit $D \neq \emptyset$ un dominant total de G. Soit $\Re$ une D-décomposition de G. Si $\Re$ est maximale alors $|V(G) \setminus V(\Re)| \leqslant 122|D|$.

Démonstration. Nous montrons en fait que $|V(G) \setminus V(\Re)| \leqslant 40 \cdot |\Re| + 2|D|$. La proposition 4 fournit la borne linéaire en $|D|$. Puisque D domine $V(G)$, nous pouvons considérer $V(G)$ comme $\bigcup_{v \in D} N(v)$; il nous suffit de borner $N(v) \setminus V(\Re)$ pour tout $v \in D$. Soit $v \in D$. Nous bornons séparément les tailles de $N_1(v)$, $N_2(v)$, et $N_3(v)$ privés de $V(\Re)$.

D'abord, considérons $N_1(v)$. Nous montrons que $N_1(v) \subseteq V(\Re)$ (l'argumentation est similaire à la preuve de la proposition 2). Soit $u \in N_1(v)$. Par l'absurde, supposons que $u \notin V(\Re)$. Nous montrons qu'il existe un chemin $p = (v, u, \dots w)$ avec $w \in D$ de longueur au plus 3 qui ne croise pas de région de $\Re$ (contredisant ainsi la maximalité de $\Re$).

Par définition de $N_1(v)$, il existe $y \in N(u) \setminus N[v]$.

— Supposons que $y \in D$. Par hypothèse $u \notin V(\Re)$, donc le chemin $p = (v, u, y)$ ne croise aucune région de $\Re$.

— Supposons $y \notin D$. Nous considérons $\mathfrak{P}_y$ l'ensemble des chemins délimitants de régions dans $\Re$, qui contiennent y.

  — Supposons que $\mathfrak{P}_y \neq \emptyset$. Soit $u_{\min} \in N(y)$ sur un chemin de $\mathfrak{P}_y$, tel que $u_{\min}$ soit minimum autour de y à partir de u (par hypothèse $u \neq u_{\min}$). Rappelons que y est dominé.

    — Supposons que $u_{\min} \in D$. Par construction, le chemin $p = (v, u, y, u_{\min})$ ne croise aucune région de $\Re$.

    — Supposons que $u_{\min} \notin D$. Soit $(v', u_{\min}, y, w_{\max}) \in \mathfrak{P}_y$ tel que $w_{\max}$ soit maximum autour de y à partir de u ($u_{\min}$ étant fixé). Par construction, le chemin $p = (v, u, y, w_{\max})$ ne croise aucune région de $\Re$.

  — Supposons que $\mathfrak{P}_y = \emptyset$. Puisque y est dominé, il existe $w \in D \cap N(y)$ distinct de v. Par hypothèse ni u ni y n'est dans $V(\Re)$, donc le chemin $p = (v, u, y, w)$ ne croise aucune région de $\Re$.

  Dans tous les cas, p est une région qui peut être ajoutée à $\Re$; contradiction avec la maximalité de $\Re$.

Donc $|\bigcup_{v \in D} N_1(v) \setminus V(\Re)| = 0$.

Ensuite, considérons $N_2(v)$. Nous montrons que, dans le pire cas, $N_2(v) \setminus V(\Re)$ peut être couvert par un ensemble $\mathcal{R}$ de $4\delta_{G_\Re}(v)$ régions simples ($\delta_{G_\Re}(v)$ étant le nombre de régions de $\Re$ dont v est un pôle).

Observons que, par définition de $N_2(v)$, tout sommet de $N_2(v) \setminus V(\Re)$ est, d'une part voisin de v d'autre part voisin d'un sommet dans $N_1(v)$. Observons aussi que, un tel sommet $u \in N_1(v)$ ainsi que l'arête $\{u; v\}$ est sur le bord d'une région de $\Re$ (dont v est un pôle). En effet, nous venons de montrer que $u \in V(\Re)$ donc u est sur le bord d'une région ; de plus, soit

$y \in N_2(v) \setminus V(\Re)$ voisin de u, supposons que $\{u; v\}$ n'est pas sur le bord d'une région, en considérant $\mathfrak{P}_u$ et par des arguments similaires à ceux ci-dessus, nous trouvons $w \in D$ tel que (v, y, u, w) peut être ajouté à $\Re$, contradiction avec la maximalité de $\Re$. Par conséquent, pour chaque région de $\Re$ il y a au plus deux sommets de $N_1(v)$ voisins de $N_2(v) \setminus V(\Re)$.

Soit $S = \{v\} \cup \{u \in N_1(v, w) \mid \{v; u\} \in \partial R, R \in \Re\}$, nous avons montré $|S| \leqslant 2\delta_{G_\Re}(v) + 1$. Considérons $\mathcal{R}$ une S-décomposition maximale en régions simples de $G[S \cup (N_2(v) \setminus V(\Re))]$. Un telle décomposition existe (d'après le lemme 2), couvre $N_2(v) \setminus V(\Re)$, (par les arguments des propositions 1 et 4, en remplaçant la notion de région par celle de région simple), et a un multigraphe sous-jacent $G_\mathcal{R}$ mince (par les arguments de la proposition 2 et ci-dessus, en remplaçant la notion de région par celle de région simple). De plus $G_\mathcal{R}$ est un multigraphe en étoile (de centre v). Nous pouvons facilement constater que $E(G_\mathcal{R}) \leqslant 2|V(G_\mathcal{R})| - 3 \leqslant 4\delta_{G_\Re}(v)$.

Observons maintenant que, à un replongement prés, les régions simples de $\mathcal{R}$ sont des régions simples de G. Soit $R' \in \mathcal{R}$ une région simple contenant un sous-graphe $G' = G[V(R') \setminus (S \cup N_2(v))]$. Par définition de $N_2(v)$, les sommets G' ne sont adjacents qu'aux pôles de R'. En utilisant cette propriété, il peut être démontré que le sous-graphe G' peut être replongé en dehors de R' en conservant le nombre linéaire de régions dans $\Re$. D'après le lemme 15, chacune des régions simples de $\mathcal{R}$ contient au plus 5 sommets (distincts des pôles).

Donc $|\bigcup_{v \in D} N_2(v) \setminus V(\Re)| \leqslant 5 \sum_{v \in D} 4\delta_{G_\Re}(v) \leqslant 40|\Re|$.

Enfin, considérons $N_3(v)$. Puisque G est réduit par règle 5, $N_3(v)$ contient au plus un sommet. Donc $|\bigcup_{v \in D} N_3(v) \setminus V(\Re)| \leqslant |D|$.

Par conséquent, en faisant l'union des trois voisinages pour chaque sommet de D, et en ajoutant l'ensemble D, nous obtenons $V(G) \setminus V(\Re) \leqslant 0 + 40|\Re| + |D| + |D|$.

□

Taille d'une région

Nous prouvons maintenant la proposition 6. Celle-ci borne le nombre de sommets dans une région. Grâce à la règle 9, nous pouvons affirmer que les sommets de la région, ou bien, ont été supprimés, ou bien, peuvent être dominés par un petit nombre de sommets. Dans le second cas, nous utilisons les dominants partiels pour construire des régions simples, ce qui, grâce à la règle 10 fournit la borne. Remarquons que dans l'énoncé nous supposons le graphe réduit, mais nous n'utilisons pas la règle 8.

Proposition 6

Soit G un graphe plan réduit. Soit D un dominant total de G et $v, w \in D$.

Toute vw-région R contient au plus 128 sommets distincts de ses pôles et des sommets de degré 1.

Démonstration. Soit R une vw-région. Il apparaîtra clairement au cours de la preuve que le pire cas est obtenu lorsque ∂R est composé de deux chemins disjoints de longueur 3, c'est-à-dire lorsqu'il contient le maximum de sommets, que nous appelons par la suite v, u_v, u_w, w, u'_w, u'_v. De plus nous notons v' et w' les deux sommets de degré 1 voisins respectivement de v et de w (s'ils existent).

Nous comptons d'abord les sommets de $N_1(v,w)$ dans R. Par définition, il y a au plus 6 sommets sur ∂R et les sommets de $N_1(v,w)$ sont sur ∂R. il y a donc au plus 4 sommets de $N_1(v,w)$.

Nous comptons maintenant les sommets de $N_{2,3}(v,w)$ dans R. Pour cela nous bornons la taille d'une décomposition maximale en régions simples qui couvre $N_2(v,w)$ et $N_3(v,w)$ dans R. Nous distinguons cinq cas : celui où la règle 9 n'est pas appliquée, plus les quatre cas de celle-ci.

Observons que, dans tous les cas, par définition de $N_2(v,w)$, tout sommet de $N_2(v,w) \cap V(R)$ (s'il y en a) est, d'une part, voisin de v ou w, d'autre part, voisin d'un sommet de $N_1(v,w) \cap V(R) \subseteq \{u_v, u_w, u'_w, u'_v\}$.

0. Supposons que $\mathcal{D} \neq \emptyset$ (la règle 9 ne s'applique pas).

   Soit $\{d_1, d_2, d_3\} \in \mathcal{D}$ (il apparaîtra clairement que le pire cas est obtenu pour un dominant partiel de taille 3). Observons que, par définition de $N_3(v,w)$, tout sommet de $N_3(v,w)$ est, d'une part voisin de v ou w, d'autre part voisin d'un d_i avec $i \in [1,3]$.

   Soit $S = (\{u_v, u_w, u'_w, u'_v\} \cup \{d_1, d_2, d_3\}) \cap V(R)$, nous savons $|S| \leqslant 7$. Considérons $\mathcal{R}$ une $\{v,w\} \cup S$-décomposition maximale en régions simples de $G[V(R)]$. Une telle décomposition existe (d'après l'algorithme du lemme 2, en remplaçant la notion de région par celle de région simple), couvre $N_{2,3}(v,w) \cap V(R)$ (par les arguments des propositions 2 et 5, en remplaçant la notion de région par celle de région simple) et a un multigraphe sous-jacent $G_{\mathcal{R}}$ mince (par les arguments des propositions 1 et 4, en remplaçant la notion de région par celle de région simple). De plus, d'après les observations sur le voisinage des sommets de $N_2(v,w)$ et de $N_3(v,w)$, nous pouvons construire $\mathcal{R}$ telle que $G_{\mathcal{R}}$ est un multigraphe biparti.

   D'après le lemme 3, $|E(G_{\mathcal{R}})| \leqslant 2|\{v,w\} \cup S| - 4 = 14$. Observons que les régions simples de $\mathcal{R}$ sont aussi des régions simples de G. Supposons par l'absurde qu'une région simple $R' \in \mathcal{R}$ contient un sommet de $V(G) \setminus V(R)$. Par construction les chemins délimitants de R' sont dans

R ; puisque $V(R')$ contient un sommet de $V(G) \setminus V(R)$, nécessairement R' contient ∂R, contradiction avec la définition de région.

Donc, d'après le lemme 15 une région simple contient au plus 5 sommets distincts de ses pôles. Or chaque arête de $G_{\mathcal{R}}$ correspond à une région simple de $\mathcal{R}$. Donc, dans ce cas R contient au plus $5 \cdot |\mathcal{R}| + |S| \leqslant 5 \cdot 14 + 7 = 77$ sommets distincts de v,w, v',w'.

1. Supposons que $\mathcal{D}_v = \emptyset$ et $\mathcal{D}_w = \emptyset$ (la règle 9 item 1 s'applique). Dans ce cas, $N_2(v, w)$ a été supprimé et $N_3(v, w)$ a été remplacé par 3 sommets.

   Donc, dans ce cas R contient au plus $4 + 1 = 5$ sommets distincts de v,w, v',w'.

2. Supposons que $\mathcal{D}_v \neq \emptyset$ et $\mathcal{D}_w \neq \emptyset$ (la règle 9 item 2 s'applique).

   Soit $\{v, d_v, d'_v\} \in \mathcal{D}_v$ et $\{w, d_w, d'_w\} \in \mathcal{D}_w$ (il apparaîtra que le pire cas est obtenu pour des dominants partiels de taille 3). Observons que, tout sommet de $N_3(v, w) \setminus (N(v) \cap N(w))$ est, d'une part, voisin de v ou w, d'autre part, voisin de v, w, d_v, d'_v, d_w, ou d'_w.

   Soit $S = (\{u_v, u_w, u'_w, u'_v\} \cup \{d_v, d'_v, d_w, d'_w\}) \cap V(R)$, nous savons $|S| \leqslant 8$. Considérons $\mathcal{R}$ une $\{v, w\} \cup S$-décomposition maximale en régions simples de $G[V(R)]$. Une telle décomposition existe (d'après le lemme 2), couvre $N_{2,3}(v, w) \setminus (N(v) \cap N(w)) \cap V(R)$ (similaire aux 2 et 5), et a un multigraphe sous-jacent $G_{\mathcal{R}}$ mince (similaire aux propositions 1 et 4).

   D'après le lemme 3, $|E(G_{\mathcal{R}})| \leqslant 3|\{v, w\} \cup S| - 6 = 24$. Observons que les régions simples de $\mathcal{R}$ sont aussi des régions simples de G (pour les mêmes raisons que dans l'item 0).

   Donc, d'après le lemme 15, dans ce cas R contient au plus $5 \cdot |\mathcal{R}| + |S| \leqslant 5 \cdot 24 + 8 = 128$ sommets distincts de v,w, v',w'.

3. Supposons que $\mathcal{D}_v \neq \emptyset$ et $\mathcal{D}_w = \emptyset$ (la règle 9 item 3 s'applique). Nous considérons séparément l'ensemble $N_3(v, w) \cap N(v)$ du reste.

   Nous bornons d'abord les tailles de $N_3(v, w) \setminus N(v)$ et de $N_2(v, w)$ dans R. Soit $\{v, d, d'\} \in \mathcal{D}_v$ (il apparaîtra que le pire cas est obtenu pour un dominant partiel de taille 3). Observons que, tout sommet de $N_3(v, w) \setminus N(v)$ est, d'une part, voisin de w, d'autre part, voisin de d, ou d'.

   Soit $S = (\{u_v, u_w, u'_w, u'_v\} \cup \{d, d'\}) \cap V(R)$, nous savons $|S| \leqslant 6$. Considérons $\mathcal{R}$ une $\{v, w\} \cup S$-décomposition maximale en régions simples de $G[V(R)]$. Une telle décomposition existe (d'après le lemme 2), couvre $((N(w) \cap N_3(v, w)) \cup N_2(v, w)) \cap V(R)$ (similaire aux 2 et 5), et a un multigraphe sous-jacent $G_{\mathcal{R}}$ mince (similaire aux propositions 1 et 4). De plus, d'après les observations sur le voisinage des sommets de

$N_2(v,w)$ et de $N_3(v,w)$, nous pouvons construire $\mathcal{R}$ telle que $G_{\mathcal{R}}$ est un multigraphe biparti.

D'après le lemme 3, $|E(G_{\mathcal{R}})| \leqslant 2|\{v,w\} \cup S| - 7 = 12$. Observons que les régions simple de $\mathcal{R}$ sont aussi des régions simples de G.

Nous bornons maintenant la taille de $N(v) \cap N_{2,3}(v,w)$. En deux étapes, nous montrons par l'absurde qu'il y a au plus 10 sommets. Rappelons que les sommets de $N(v) \cap N_{2,3}(v,w)$ sont dans $\bigcup \mathcal{D}_v$ et donc dans $N_1(v)$. Nous bornons le nombre de ces sommets dont le voisinage intersecte $N[d]$; nous utilisons ensuite la même argumentation sur ceux dont le voisinage intersecte $N[d']$. Par l'absurde, supposons qu'il y ait au moins 5 tels sommets que nous nommons u_1, u_2, u_3, u_4, u_5. Remarquons que les sommets de $N(d) \cap N(w)$ sont isolés de v et des u_i pour $i \in [1,5]$ par deux dw-chemins. Par conséquent, au moins trois u_i pour $i \in [1,5]$ sont adjacents à un même sommet dans $N(d)$, sans perte de généralité supposons qu'il s'agisse de u_1, u_2, u_3. Puisque G est réduit par la règle 9 item 3, les ensembles $N(u_i) \setminus N(v)$ pour $i \in [1,5]$ sont incomparables entre eux, et donc u_1, u_2, u_3 sont voisins de sommets dans $N(d') \cap N(v)$, ou dans $\{w, u_w, u'_w\}$. En contractant, d'une côté les arêtes de la forme $\{d; y\}$ avec $y \in N(d) \cap N(v)$ sur d, de l'autre côté, les arêtes de la forme $\{w; z\}$ avec $z \in N(w) \setminus N(d)$ sur w, nous obtenons un $K_{3,3}$ mineur de G ; contradiction avec la planarité ([46, 54] ; cf. figure 2.20).

Donc $|\bigcup \mathcal{D}_v| \leqslant 10$.

Donc, d'après le lemme 15, dans ce cas R contient au plus $5 \cdot |\mathcal{R}| + |S| + 1 \leqslant 5 \cdot 12 + 7 + 10 = 77$ sommets distincts de v,w, v',w'.

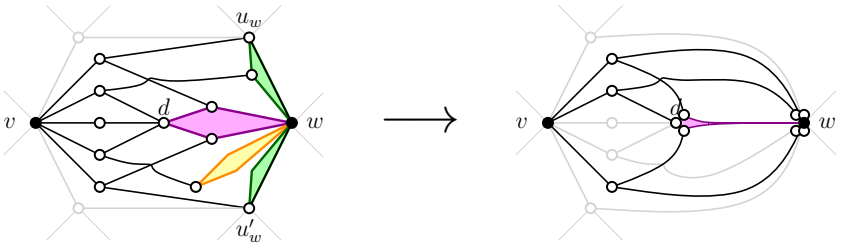


FIGURE 2.20 – Exhibition du mineur dans la proposition 6.

4. Supposons que $\mathcal{D}_v = \emptyset$ et $\mathcal{D}_w \neq \emptyset$ (la règle 9 item 4 s'applique). Symétriquement à l'item 3.

La région R contient au plus $\max\{77, 5, 128, 77\} = 128$ sommets distincts de v,w, v',w'.

□

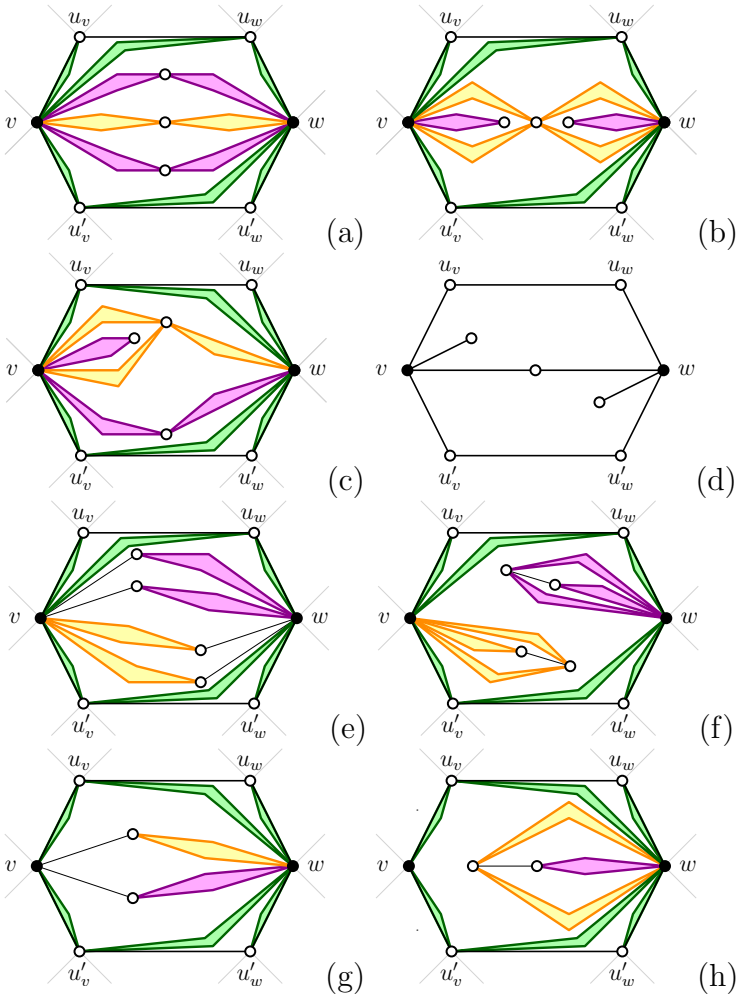


FIGURE 2.21 – Quelques illustrations de $\mathcal{R}$ couvrant $N_2(v,w)$ et $N_3(v,w)$ dans la proposition 6. (a,b,c) Dans dans le cas où la règle 9 ne s'applique pas, avec des pôles dans $\mathcal{D}$. (d) Dans le cas de l'item 1, $N_3(v,w)$, où tous les sommets sont remplacés par des gadgets. (e,f) Dans le cas de l'item 2, $N_3(v,w)$, avec des pôles dans $\mathcal{D}_v$ et $\mathcal{D}_w$. (g,h) Dans le cas de l'item 3, $N_3(v,w)$, avec des pôles dans $\mathcal{D}_v$.

Remarquons que, pour obtenir le pire cas, nous supposons que la région contient les trois sommets du dominant partiel (par exemple, $\{d_1, d_2, d_3\} \in \mathcal{D}$) ; en pratique, ces trois sommets sont nécessairement répartis entre les différentes vw-régions. Remarquons aussi que l'inégalité d'Euler pour les multigraphes bipartis ne nous donne qu'une borne grossière, nous savons que les multigraphes sous-jacents que nous considérons sont plus contraints (nous pourrions obtenir une meilleur borne par une analyse au cas par cas).

Noyau linéaire dans les graphes planaires

Nous pouvons maintenant utiliser les trois propositions pour prouver notre résultat.

Théorème 3
Le problème Domination Totale, restreint aux graphes planaires et paramétré par k la taille de la solution, admet un noyau linéaire de taille $508k$.

Démonstration. Nous montrons que l'itération des règles produit un algorithme polynomial qui pour chaque instance planaire (G, k), renvoie une instance planaire équivalente (G', k) telle que : G et G' sont des instances négatives ; ou $|G'| \leqslant 508k$.

Soit G le graphe plan de l'instance et G' le graphe réduit par les règles 8, 9, et 10. D'après les lemmes 11, 12, et 13 G' admet un dominant total de taille au plus k si et seulement si G en admet un. Et d'après le lemme 14 la réduction se fait en temps polynomial. Donc l'itération des règles forme bien une réduction polynomiale.

Supposons maintenant que G' admette une solution D'. Si $|D'| = 0$ alors G' est vide, $|D'| \neq 1$ car D' est total, si $|D'| = 2$ alors G' peut être décomposé en au plus une région. Sinon, $|D'| \geqslant 3$, nous pouvons appliquer les propositions 4, 5, et 6 pour montrer que la taille de G' est bornée par $128(3k - 6) + 122k + 2k = 508k$. Par contraposée, si G' a plus de sommets alors il n'a pas de dominant total de taille k. Donc l'instance réduite est un noyau.

Par conséquent, l'algorithme est bien une extraction de noyau. □

2.4 Discussion

Dans ce chapitre nous avons présenté et corrigé la méthode de décomposition en régions [3] et nous l'avons utilisée pour obtenir un noyau linéaire

pour deux variantes du problème Domination : un noyau de taille $43k$ pour Domination Rouge-Bleu et un noyau de taille $508k$ pour Domination Totale. Ces deux résultats sont constructifs au sens où nous décrivons les algorithmes d'extraction de noyaux ; de plus, nous donnons une borne explicite et raisonnable sur la taille des noyaux.

Rappelons nous que, après avoir décrit les règles de réduction, cette méthode analyse la taille d'un noyau en trois étapes :

— construire une décomposition avec un nombre linéaire de régions ;

— borner linéairement le nombre de sommets hors de la décomposition ;

— borner par une constante le nombre de sommets dans chaque région.

Comme nous l'avons souligné dans la section 2.1, seule une partie de la première étape est décrite indépendamment des problèmes étudiés. Le reste des preuves suppose la connaissance d'un ensemble solution. En cela, la méthode ne peut pas être considérée comme un cadre générique. Cependant, il est aisé de remarquer que nos preuves n'utilisent qu'une seule propriété de la solution : tout sommet est à distance bornée du reste de l'ensemble solution (distance au plus 4 pour un dominant rouge-bleu et au plus 3 pour un dominant total). Les règles de réduction, quant à elles, sont clairement conçues pour réduire les régions (même si elles n'y font pas référence explicitement), et sont basées sur cette même propriété. Notons que la règle pour une paire est construite sur un schéma particulier qui se dessine dans la section 2.2 (cf. règle 6) et se précise dans la section 2.3 (cf. règle 9) : celui-ci consiste à considérer toutes les solutions locales et à choisir une région gadget pour simuler toutes les possibilités. Ce schéma semble facilement transposable d'un problème à un autre. Il faut aussi signaler une certaine similarité (dont nous reparlerons plus en détail) entre ce schéma et la technique que nous présentant dans le chapitre 3.

Au vu de ces considérations, la généralisation de cette méthode en un cadre valable pour toute une famille de problème (sur les graphes planaires) semble naturelle. C'est un travail qui avait été commencé par Guo et Niedermeier [40], mais qui avait hérité des imperfections de son prédécesseur. Nous pensons que les nouvelles définitions et preuves présentes dans ce chapitre peuvent servir de fondation à une nouvelle démonstration du cadre proposé par Guo et Niedermeier.

Considérant les derniers méta-résultats sur les noyaux [8, 26, 43], ainsi que la contribution que nous y apportons au chapitre 3, l'élaboration d'un tel cadre peut paraître superflu d'un point de vue théorique. Nous pensons néanmoins qu'un cadre générique permettant de construire des noyaux dans les graphes planaires demeure un outil intéressant. En effet, étant restreint

aux graphes planaires, il pourrait utiliser plus finement les propriétés de cette classe et donc permettrait d'obtenir des bornes plus raisonnables.

Naturellement, nos deux noyaux peuvent être améliorés selon trois axes plus conventionnels : la complexité de l'extraction, la taille du noyau, et la classe de graphe du problème.

Concernant les complexités, nous n'avons pas particulièrement cherché à optimiser le temps d'exclusion (nous avons seulement pris en compte la planarité lors de l'analyse de complexité). Nos algorithmes et leur analyse sont donc très certainement améliorables. En particulier, un choix judicieux de l'ordre dans lequel sont appliquées les règles peut amélioré la complexité, du moins en pratique. De même lorsque qu'une règle est appliquée il convient de vérifier que le graphe est toujours réduit selon les règles précedentes ; comme chaque règle ne fait que des modification locale, la vérification peut être locale. Pour Domination Rouge-Bleu, nous avons fait la remarque que nos règles sont redondantes (cf. règles 2, 3 et 5), il y a là aussi matière à gagner du temps.

Quant aux tailles des noyaux, nous avons essayé d'être précis dans leur analyse, en particulier lorsque nous comptons les sommets d'une région. Nous avons cependant omis certaines améliorations afin de ne pas alourdir des preuves déjà longues et techniques. En particulier la taille du noyau pour Domination a été améliorée par Chen, Fernau, Kanj et Xia [12] ; leur idée consiste à bicolorer partiellement le graphe (c'est-à-dire, séparer les sommets dominants et les sommets dominés), pour appliquer des règles similaires à celles que nous énonçons pour Domination Rouge-Bleu. Si cette idée permettrait d'améliorer la borne du noyau pour Domination Totale, elle ne saurait être utilisée pour Domination Rouge-Bleu, dont les instances sont déjà bicolorées. Une autre approche permettant de diminuer nos constances consisterait à borner la taille des régions en moyenne. En effet, nous considérons les régions indépendamment les unes des autres et nous retenons le pire cas à chaque fois ; or certaines configurations ne peuvent exister simultanément dans toutes les régions (par exemple, si les sommets d'un dominant partiel sont tous dans une vw-région alors toutes les autres vw-régions n'en contiennent pas). Enfin, dans leur conclusion, Alber, Fellows et Niedermeier ont proposé d'étendre leur jeu de règles en considérant le voisinage de trois sommets, ou plus ; ce qui nécessiterai de définir une notion de région à trois pôles, ou plus. A notre connaissance cette voie n'a pas encore été explorée.

À propos de la classe des graphes, les résultats sur les graphes planaires sont le plus souvent extensibles aux classes de graphes peu denses. Il est peu probable que le cadre de Bodlaender, Fomin, Lokshtanov, Penninkx, Sau-

rabh et Thilikos [8] puisse s'appliquer tel quel à Domination Rouge-Bleu étant donné la forme des instances (bicolorées) de celui-ci ; en revanche, il permet de montrer l'existence d'un noyau pour Domination Totale dans les graphes de genre borné [34] ; si le cadre présenté dans le chapitre 3 s'applique à ce problème, le noyau est constructif. Dans les classes de graphes plus larges, Domination (et la plus part de ses variantes) ne satisfont pas les conditions des méta-théorèmes [26, 43]. Cependant en s'inspirant des noyaux pour Domination et Domination Connexe dans les graphes sans mineur [31] et sans mineur topologique [30], il devrait être possible d'étendre nos deux résultats à ces classes.

Comme nous l'avons vu un peu plus haut, nous avons peu de pistes pour améliorer la taille du noyau de Domination Rouge-Bleu. De part la nature même de ses instances (des graphes planaires et bipartis) Domination Rouge-Bleu semble être plus facile que la plus part des variantes de Domination. Cela soulève la question d'une borne inférieure sur la taille de son noyau. Il existe plusieurs techniques pour démontrer qu'un problème n'admet pas de noyau polynomial, mais les techniques pour démontrer qu'un problème n'admet pas de noyau linéaire avec une constante multiplicative inférieure à une certaine constante sont plus rares ; à notre connaissance il n'y a que le résultat de Chen, Fernau, Kanj et Xia utilisant la notion de dual paramétrique [12]. À défaut d'une borne inférieure théorique, il serait intéressant d'exhiber un pire cas pour notre algorithme.

Enfin, force est de constater que les preuves que nous proposons sont longues, pénibles et inélégantes (cf. propositions 3 et 6). Nous pensons que cela est dû à un manque d'outil. De nombreux outils existent pour traiter les graphes planaires. Or le sous-graphe induit par une région est planaire, mais cette propriété ne permet d'obtenir qu'une borne grossière sur la taille de la région. En effet, ce sous-graphe est beaucoup plus contraint : toutes ses arêtes doivent pouvoir être tracées à l'intérieur de la région. En quelque sorte, il s'agit d'un graphe planaire dont la face externe est fixée. Autrement dit, pour borner précisément la taille d'une région nous devons utilisé la topologie de la région, qui est une contrainte plus forte que la planarité, puisque nous ne pouvons pas dessiner à l'extérieur.

Chapitre 3

Méta-noyaux explicites

Comme nous l'avons mentionné, les travaux de Alber, Fellows et Niedermeier [3] ont ouvert la voie à de nombreux résultats plus généraux : Guo et Niedermeier [40] ont tenté de généraliser la méthode de décomposition en régions à tous les problèmes vérifiant une certaine propriété de distance ; ensuite Bodlaender, Fomin, Lokshtanov, Penninkx, Saurabh et Thilikos [8] ont démontré l'existence de deux méta-noyaux dans les graphes plongeables dans une surface de genre borné, pour tous problèmes vérifiant certaines propriétés (le premier, un noyau polynomial pour les problèmes recouvrable et exprimable en logique monadique ; le second, un noyau linéaire pour les problèmes quasi-recouvrable et d'index entier fini) ; puis Fomin, Lokshtanov, Saurabh et Thilikos [31] ont étendu ces résultats aux graphes sans mineur, en utilisant des propriétés de bidimensionnalité et de séparabilité ; enfin Kim, Langer, Paul, Reidl, Rossmanith, Sau et Sikdar [43] l'ont encore étendu aux graphes sans mineur topologique, en utilisant des propriétés bornant la largeur arborescente.

En résumé, ses méta-résultats garantissent l'existence de noyaux linéaires dans les graphes peu denses pour de nombreux problèmes satisfaisant certaines propriétés génériques. Mais cette généralité est accompagnée d'une perte de constructivité. Ces résultats ne fournissent que l'existence d'une extraction de noyau. Les preuves, non constructives, ne permettent pas d'exhiber un algorithme (du moins certaines étapes) avec sa complexité, et il n'y a pas de borne explicite sur la taille du noyau (sur la constante multiplicative).

L'extraction des méta-noyaux (et la preuve d'existence) s'effectue en deux étapes : premièrement, une décomposition en protrusion est construite ; deuxièmement, l'existence d'un représentant (c'est-à-dire une protrusion équivalente et de petite taille), pour chacune des protrusions est montrée. Les deux généralisations [31, 43] étendent les théorèmes initiaux des méta-noyaux [8] en améliorant la méthode de construction d'un décomposition, mais conserve

la même méthode (non constructive) pour remplacer une protrusion par son représentant.

Puisqu'il existe déjà plusieurs outils pour construire explicitement une décomposition en protrusions, nous ne développerons pas cet aspect de l'algorithme et nous nous focaliserons sur le remplacement d'une protrusion par son représentant. Nous construisons le représentant là où les résultats précédents énoncent seulement son existence. Pour cela nous avons recours à la programmation dynamique : nous parvenons à coder toute l'information nécessaire pour identifier les représentants dans le bord de la protrusion.

Pour concevoir notre cadre de remplacement de protrusions, nous nous inspirons essentiellement de la preuve du second théorème des méta-noyaux de Bodlaender, Fomin, Lokshtanov, Penninkx, Saurabh et Thilikos [8]. Combiné avec un outil de construction de décomposition en protrusions, notre cadre devient un algorithme générique d'extraction de noyaux. Pour nos applications, nous utiliserons les puissants outils fournies par les deux généralisations [31, 43] pour construire des décompositions en protrusions.

Dans la section 3.1 nous présentons notre cadre générique de remplacement de protrusions et prouvons sa correction ; le remplacement correspond à une règle de réduction. Dans la section 3.2 nous expliquons comment appliquer notre cadre générique et les outils nécessaires pour obtenir un noyau. Dans les dernières sections nous appliquons notre cadre à divers problèmes : r-Domination (section 3.3), r-Indépendance (section 3.4), $\mathcal{F}$-Paquetage (section 3.5).

Les méta-noyaux de Bodlaender, Fomin, Lokshtanov, Penninkx, Saurabh et Thilikos [8] se base sur une décomposition en protrusions des graphes. Cette notion est une généralisation de la décomposition en régions. Afin de donner une première intuition, considérons la méthode de décomposition en régions sous un point de vue un peu différent. Un fois décomposé, un graphe contient un ensemble séparateur (les sommets extérieurs aux régions plus le bords des régions) de taille linéaire, et un nombre linéaire de sous-graphes (les régions) avec une structure suffisamment contrainte pour qu'ils puissent être remplacés par des sous-graphes de taille bornée. Ce remplacement doit conserver le comportement du graphe par rapport au problème. Il faut donc que toutes les solutions possibles dans le sous-graphe d'origine puissent être simulées dans le représentant. Cette idée est présente dans les deux noyaux que nous présentons au chapitre 2, et particulièrement dans le cas de Domination Totale (section 2.3, définition 55). Remarquons que le bord de la région joue un rôle primordial, en effet, lorsque la solution globale est reconstituée, il faut s'assurer que les solutions internes aux

sous-graphes se recollent bien. Nous pouvons en effet constater que, dans le chapitre 2, le bord d'une région est composé des sommets pour lesquelles ont ne peut pas prédire l'appartenance à la solution ; par contre l'intérieure d'un région contient un nombre constant de sommets dominants, dont le choix est déterminé par le choix des sommets dominants sur le bord. Autrement dit, le bord contient l'information que au plus deux sommets (les pôles) sont dominants, un fois ces deux sommets identifiés, la solution est facilement reconstituée. La décomposition en protrusions fonctionne sur la même idée : la protrusion est un sous-graphe dans lequel la solution est aisément construite, un fois que l'information du bord est connue.

Remarquons que, pour la plupart des variantes de Domination (à l'exception de Domination Connexe [47]) les règles de réductions pouvaient être appliquées sans connaissance de la décomposition en régions. Il n'était donc pas nécessaire de la construire, la preuve de son existence suffit. Ici nous avons besoin de la décomposition en protrusions pour appliquer la règle de remplacement. La constructibilité de la décomposition en protrusions est donc un point fondamental.

3.1 Règle de remplacement de protrusions

Notre règle de remplacement se base sur les propriétés de la protrusion. Une protrusion est un sous-graphe dont le bord et la largeur arborescente sont bornés. Par conséquent, la protrusion est adaptée à l'utilisation de la programmation dynamique. La programmation dynamique utilise des tables dans lesquelles elle enregistre l'information nécessaire à la reconstruction d'une solution dans un sous-graphe. Dans le cas d'un graphe de largeur arborescente bornée, une table est associée à chaque séparateur d'une décomposition arborescente (un sac) ou, de manière équivalente, à chaque sous-graphe induit par un sous-arbre de la décomposition (le séparateur étant le bord du sous-graphe). La table considère toutes les manières dont une solution peut traverser le séparateur. En particulier, une table est associée au bord de la protrusion (qui correspond à la racine de la décomposition arborescente). Et donc, d'une certaine façon, cette table du bord mémorise le comportement de la protrusion vis à vis du problème : quelle est la taille des meilleures solutions partielles dans la protrusion et comment recoller les solutions partielles au reste du graphe.

Dans cette section, nous posons quelques notions préliminaires (graphes bordés, protrusions, équivalence canonique). Puis, nous définissons la notion d'encodeur qui formalise la programmation dynamique et nous permet de la manipuler de façon générique (indépendamment du problème). Nous introduisons ensuite une relation d'équivalence sur les protrusions, définie à partir de l'encodeur (c'est-à-dire, des tables de programmation dynamique). Enfin, nous montrons que si l'encodeur et la relation d'équivalente vérifient certaines propriétés, alors notre équivalence raffine l'équivalence canonique, autrement dit, deux protrusions équivalentes selon notre définition peuvent être interverties sans changer le comportement du graphe par rapport au problème. De plus, notre équivalence a un petit nombre de classes ; nous montrons qu'il est possible de construire un représentant pour chacune de ses classes, une fois encore en imitant la programmation dynamique. Ces deux propriétés (le raffinement de l'équivalence canonique et la constructibilité des représentants) prouvent la validité de notre règle ; nous pourrons donc l'appliquer sur une décompositions en protrusion par la suite. Puisque nous savons construire des décompositions linéaires, et que nos représentants ont taille constante (en fonction du paramètre), après réduction, nous obtenons des noyaux linéaires.

Nous rappelons que notre cadre s'applique à des problèmes d'optimisation sous forme décisionnelle paramétrés par la taille de la solution ; et que lorsque nous parlons d'un problème Π, nous supposons qu'il est de ce type (cf. définition 27).

3.1.1 Définitions préliminaires

Dans cette sous-section, nous posons les définitions de graphes bordés, de protrusions, et d'équivalence canonique. Toutes ses notions ont été introduites par Bodlaender, Fomin, Lokshtanov, Penninkx, Saurabh et Thilikos [8]. Les graphes bordés et les protrusions sont des objets très similaires ; tous deux servent à décomposer le graphe, en décoller une partie et en recoller une autre. Cependant le graphe bordé est un graphe (avec un bord) à part entière (décollé) alors que la protrusion est un sous-graphe (collé dans un graphe). La relation d'équivalence canonique est celle qui permet de prouver la correction d'un remplacement de protrusion ; elle conserve le comportement par rapport au problème.

Dans les définitions suivantes, nous donnons les notions de graphe bordé et de collage. Informellement, un graphe bordé est un graphe dans lequel un ensemble de sommets est spécifié ; ces sommets sont labellés. Cet ensemble spécifique de sommets est appelé le bord. Une fois encore, nous commettrons un abus en utilisant la notation $\partial \cdot$ pour le bord ; cet abus est justifié par le fait que le bord d'un graphe bordé correspondra systématiquement au bord d'une protrusion (cf. définition 5). Le collage de deux graphes bordés est l'identification des deux bords en respectant les labels. Par la suite, le collage nous permettra d'effectuer le remplacement d'une protrusion par une autre dans une graphe.

Définition 56 : graphe bordé

Un *graphe bordé* est un graphe G muni d'un ensemble $\partial G \subseteq V(G)$ et d'une fonction injective $\lambda_G : \partial G \to \mathbb{N}$. L'ensemble ∂G est le *bord* de G et la fonction λ_G définit les labels des sommets du bord. l'ensemble des labels est noté $\Lambda(G) = \lambda_G(\partial G)$.

Un graphe G est *t-bordé* si $\Lambda_G(G) \subseteq [1, t]$. Nous notons $\mathcal{B}_t$ l'ensemble des graphes t-bordés et $\mathcal{F}_t$ l'ensemble des graphes t-bordés de largeur arborescente au plus $t - 1$.

Une décomposition arborescente de $G \in \mathcal{F}_t$ est *bordée* si $X_r \supseteq \partial G$, où X_r est le sac de la racine de l'arbre.

⌟

Clairement, $\mathcal{F}_t \subseteq \mathcal{B}_t$. Remarquons qu'un graphe peut être vu comme un graphe 0-bordé. Par conséquent, nous utilisons le même alphabet Γ pour décrire les graphes et les graphes bordés.

Remarquons aussi que, à partir d'une bonne décomposition arborescente d'un graphe G t-bordé, nous pouvons facilement construire un bonne décomposition arborescente bordée en ajoutant ∂G à chaque sac. La largeur

arborescente est alors doublée (rappelons que dans notre cadre, la largeur arborescente est un constante, qui ne dépend que du problème).

Définition 57 : collage

Soit G_1 et G_2 deux graphes t-bordés. Le *collage* $G_1 \oplus G_2$ est le graphe obtenu par l'union disjointe de G_1 et G_2 puis l'identification des sommets de ∂G_1 avec les sommets de ∂G_2 ayant le même label.

Soit $G = G_1 \oplus G_2$ un graphe et G'_2 un graphe bordé. Le graphe $G' = G_1 \oplus G'_2$ est le graphe obtenu à partir de G en *remplaçant* G_2 par G'_2. ⌟

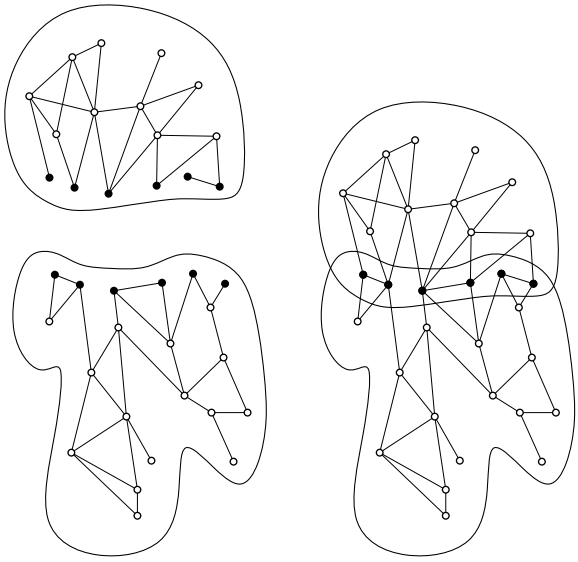


FIGURE 3.1 – Exemple de collage de deux graphe 6-bordés (cf. définition 57).

Remarquons que dans $G_1 \oplus G_2$ il y a une arête $\{v, u\}$ si elle existe dans G_1 ou dans G_2 ; c'est vrai en particulier pour v, u appartenant au bord. Le graphe $G_1 \oplus G_2$ peut être considéré ou bien comme un graphe ou bien comme un graphe bordé. Dans toutes nos utilisation, nous ne considérons pas le bord de G_1 et G_2 comme le bord de $G_1 \oplus G_2$. Par abus de langage, la notion de remplacement est aussi valable sur les protrusions.

Nous définissons maintenant la protrusion. Intuitivement, c'est un sous-graphe dont le bord et la largeur arborescente sont bornés. Dans de nombreux cas il est intéressant de trouver une décomposition en protrusions du graphe afin d'appliquer une programmation dynamique sur chaque protrusion.

Définition 58 : protrusion

Une *t-protrusion* Y dans un graphe G est un sous-graphe induit de G avec $|\partial Y| \leqslant t$ et $\mathrm{tw}(Y) \leqslant t-1$.

⌟

Remarquons qu'une région est bien un cas particulier de protrusion. Considérons R une r-région. D'une part, le bord ∂R contient au plus $2r$ sommets ; d'autre part, le graphe induit par $V(R)$ est planaire, de diamètre au plus r, donc de largeur arborescente au plus $O(r)$ [20]. Comme nous l'avons dans nos applications, la valeur de r est une constante fixée par le problème.

Notre règle de réduction correspond à un remplacement de protrusions. Afin de montrer que cette réduction amène à un noyau, une possibilité consiste à exhiber une décomposition en protrusions de l'instance (cf. corollaire 1). Intuitivement, il s'agit d'une partition du graphe, dans laquelle le nombre de protrusions et le nombre de sommets hors d'une protrusion sont contrôlés.

Définition 59 : décomposition en protrusions

Une (α, t)-*décomposition en protrusions* d'un graphe G est une partition $\mathcal{P} = Y_0 \uplus Y_1 \uplus \cdots Y_\ell$ de $V(G)$ telle que :

- $\ell \leqslant \alpha$ et $|V(Y_0)| \leqslant \alpha$;
- *et* pour tout $i \in [1, \ell]$, $N(Y_i) \subseteq Y_0$;
- *et* pour tout $i \in [1, \ell]$, $Y_i \cup N(Y_i)$ est une t-protrusion de G.

⌟

Nous voulons que le nombre de protrusions soit linéaire en fonction du paramètre (suffisamment petit) pour obtenir un noyau linéaire. Étant donnée une instance (G, k), lorsque nous disons qu'une décomposition en protrusion est linéaire, nous entendons que c'est une décomposition de G et que $\alpha = O(k)$. La valeur de t (qui borne la largueur arborescente) importe peu, car, dans nos applications, ce sera une constante ne dépendant que du problème.

Remarquons qu'une décomposition en régions peut être vue comme une décomposition en protrusions.

Comme nous l'avons déjà dit, les graphes bordés de largeur arborescente bornée et les protrusions sont très similaires. Leur distinction est essentiellement une question de point de vue : un graphe bordé est un objet en tant que tel alors qu'une protrusion est un sous-graphe. Nous souhaitons pouvoir manipuler et comparer les graphes bordés indépendamment de ce à quoi ils seront collés par la suite ; pour cela, il faut spécifier, par les labels, comment

procéder au collage (nous voulons, par exemple, que deux graphes bordés équivalents soient bien collés de la même façon). Tandis que nous considérons la protrusion comme la partie d'un graphe, et donc les sommets du bord de la protrusion sont aussi des sommets du graphe ; en d'autres termes, la façon dont la protrusion est collée au graphe est intrinsèque.

Par conséquent, une protrusion Y d'un graphe G peut être vue comme un graphe t-bordé de largeur arborescente $t-1$ en ajoutant des labels quelconques sur les sommets de son bord. Bien sûr, nous considérerons $(G \setminus Y) \cup \partial Y$ comme un graphe t-bordé avec les mêmes labels. Inversement, un graphe t-bordé $G_1 \in \mathcal{F}_t$ de largeur arborescente $t-1$ peut être vu comme la protrusion d'un graphe $G_1 \oplus G_2$ avec $G_2 \in \mathcal{B}_t$ quelconque. Notre algorithme d'extraction de noyau sélectionne une protrusion du graphe en entrée, la considère comme un graphe bordé, trouve son représentant (un graphe bordé) et le colle au graphe en entrée en tant que protrusion.

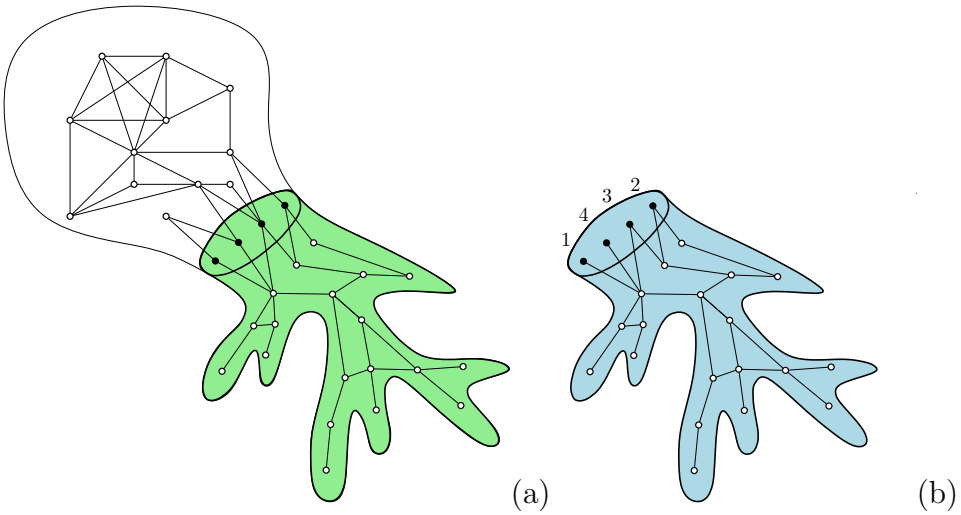


FIGURE 3.2 – Exemple de 4-protrusion et de graphe 4-bordé. (a) Une protrusion : sous-graphe dont le bord est de taille 4 et la largeur arborescente est de 2 (cf. définition 58). (b) Un graphe bordé : graphe avec un ensemble de 4 sommets labellés et une largeur arborescente de 2 (cf. définition 56).

Dans la sous-section 3.1.3 nous utiliserons la notion de graphe bordé car nous cherchons à les classer, indépendamment de ce à quoi ils pourraient être collés. Dans la sous-section 3.1.4 nous utiliserons la notion de protrusion car nous modifions un graphe en y remplaçant une protrusion par une autre.

Nous définissons maintenant l'équivalence canonique. Celle-ci sert à prouver la correction d'un remplacement de protrusion : un remplacement est va-

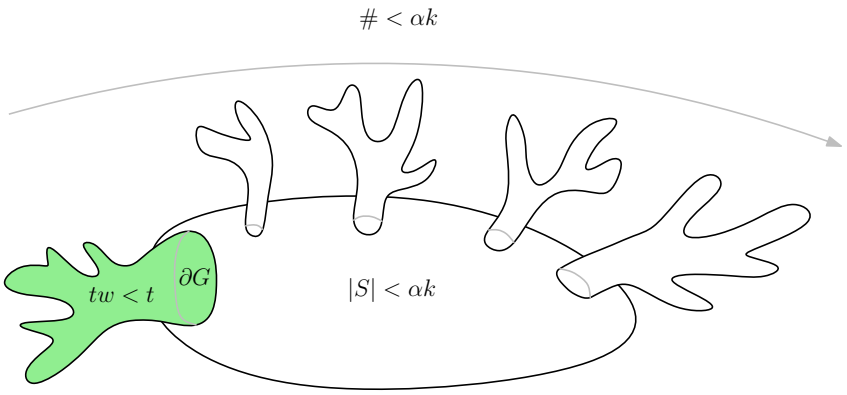


FIGURE 3.3 – Définition schématique d'une décomposition en protrusions (cf. définition 59).

lide s'il ne change pas le comportement du graphe par rapport au problème. En fait, l'équivalence porte sur les graphes bordés. Deux graphes bordés sont canoniquement équivalents si, pour chaque graphe bordé auquel ils peuvent être collés, ils admettent une solution simultanément ; de plus la différence entre la taille des solutions est constante (cf. figure 3.4). Cette notion correspond bien à la preuve de correction d'une règle de réduction : une règle est valide si l'instance initiale et l'instance réduite admettent des solutions simultanément.

Définition 60 : équivalence canonique

Soit Π un problème et $t \in \mathbb{N}$. Soient $G_1, G_2 \in \mathcal{B}_t$. Les graphes t-bordés G_1, G_2 sont *canoniquement équivalents*, $G_1 \equiv_\Pi G_2$, s'il existe une constante de transposition $\Delta_{\Pi,t}(G_1, G_2) \in \mathbb{Z}$ telle que quelque soit $H \in \mathcal{B}_t$ et $k \in \mathbb{Z}$, $(G_1 \oplus H, k) \in \Pi$ si et seulement si $(G_2 \oplus H, k + \Delta_{\Pi,t}(G_1, G_2)) \in \Pi$. ⌟

Dans la sous-section 3.1.3, nous définirons la $\mathcal{E}$-équivalence, qui raffine l'équivalence canonique. Cela nous autorisera à effectuer le remplacement d'une protrusion par une protrusion $\mathcal{E}$-équivalente.

Une des deux hypothèses dans le méta-théorème de Bodlaender, Fomin, Lokshtanov, Penninkx, Saurabh et Thilikos [8] est que le problème considéré a un index entier fini, c'est-à-dire que l'équivalence canonique à un nombre fini de classes. Cela leur permet de montrer l'existence d'un représentant de taille borné pour chaque classe. Nous mentionnons ici la définition simplement à titre informatif.

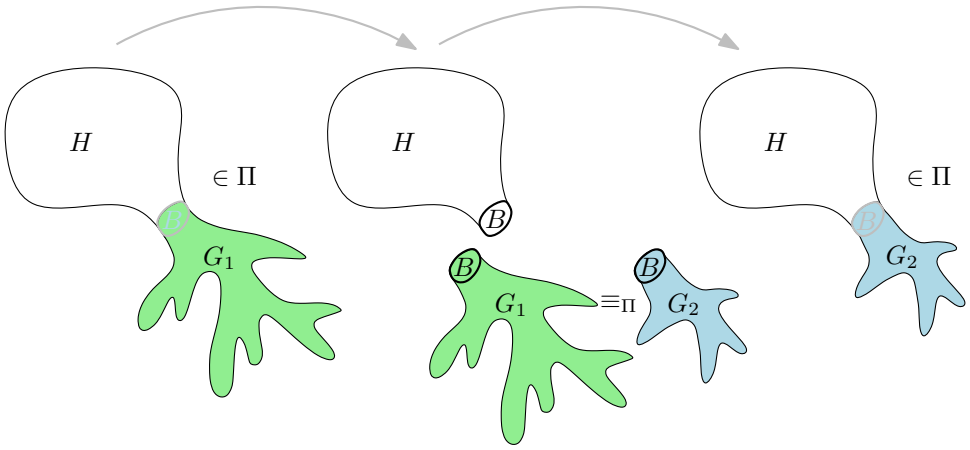


FIGURE 3.4 – Représentation schématique de l'équivalence canonique : en échangeant deux sous-graphes (par exemple, des protrusions) canoniquement équivalents, l'existence de solutions est conservée (cf. définition 60).

Définition : index entier fini [8]

Un problème Π a un *index entier fini* si et seulement si le nombre de classes de l'équivalence canonique sur $\mathcal{F}_t$ est fini, pour tout entier t.

⌟

Puisque nous fournissions notre propre $\mathcal{E}$-équivalence, et nos outils pour borner le nombre de ses classes d'équivalence, nous n'auront pas besoin de faire explicitement référence à la propriété d'indexation entière finie. Cependant, puisque notre $\mathcal{E}$-équivalence raffine l'équivalence canonique, cela implique que tous les problèmes qui entrent dans notre cadre ont un index entier fini.

3.1.2 Encodeur de programmation dynamique

Nous définissons ici la notion d'encodeur. Nous décrivons le cadre de travail pour des problèmes de minimisation sous forme décisionnelle, c'est-à-dire de la forme $\Pi = \{(G,k) \mid \exists S, m^\Pi(S) \leqslant k, (G,S) \in L^\Pi\}$ (cf. section 1.3), le cadre de travail pour un problème de maximisation en découle aisément.

Un encodeur est destiné à décrire les tables de programmation dynamique et comment les remplir. Nous donnons ensuite deux propriétés nécessaires pour que l'encodeur simule une programmation dynamique efficace qui résout le problème considéré.

Intuitivement, nous l'avons dit, un encodeur formalise la programmation dynamique pour un problème générique Π. Rappelons qu'une table est générée pour chacun des sous-graphes considérés par la programmation dynamique. Cette table à pour entrées les descriptions de toutes les façons dont une solution peut traverser le bord du sous-graphe ; et, pour chacune de ces entrées, la table enregistre la taille de la meilleure solution (traversant selon la façon décrite par l'entrée). Notre encodeur contiendra donc une fonction qui, étant donné un bord, génère les entrées de la table (nous les appellerons encodages), un langage qui permette de reconnaître les solutions partielles et une fonction qui détermine les valeurs à enregistrer (cf. figure 3.5).

Définition 61 : encodeur

Un *encodeur* est un triplet $\mathcal{E} = (\mathcal{C}^{\mathcal{E}}, L^{\mathcal{E}}, f^{\mathcal{E}})$, avec :

- $\mathcal{C}^{\mathcal{E}}$ une fonction calculable de $2^{\mathbb{N}} \to 2^{\Upsilon^*}$ qui, à chaque ensemble d'entiers $I \subseteq \mathbb{N}$, associe un ensemble $\mathcal{C}^{\mathcal{E}}(I)$ de chaînes de caractères sur un certain alphabet Υ (qui correspondront aux entrées d'une table) ; un *encodage* est une chaîne $R \in \mathcal{C}^{\mathcal{E}}(I)$;
- $L^{\mathcal{E}}$ un langage calculable qui reconnaît des triplets de la forme $(G, S, R) \in \Gamma^* \times \Sigma^* \times \Upsilon^*$, où G est un graphe bordé décrit sur Γ, S est un certificat décrit sur Σ (qui correspondra au certificat d'une solution partielle au problème), et $R \in \mathcal{C}^{\mathcal{E}}(\Lambda(G))$ est un encodage sur Υ ; si $(G, R, S) \in L^{\mathcal{E}}$, nous dirons que S *satisfait* l'encodage R (dans G) ;
- $f^{\mathcal{E}}$ une fonction calculable de $\Gamma^* \times \Upsilon^* \to \mathbb{N} \cup \{+\infty\}$ qui à un graphe bordé G et un encodage $R \in \mathcal{C}^{\mathcal{E}}(\Lambda(G))$ associe un entier (qui correspondra à la mesure d'une solution partielle optimale satisfaisant R).

La *taille* d'un encodeur est $s_{\mathcal{E}}(t) = \max\{|\mathcal{C}^{\mathcal{E}}(I)| \mid I \subseteq \{1, \ldots, t\}\}$. ⌟

Les encodages dans $\mathcal{C}^{\mathcal{E}}(I)$ correspondent aux entrées de la table pour un graphe dont le bord est labellé par I. Chaque encodage $R \in \mathcal{C}^{\mathcal{E}}(I)$ peut être appréhendé ou bien comme une façon dont une solution peut traverser le bord du graphe, ou bien comme une contrainte supplémentaire devant être vérifiée par une solution partielle. Un autre point de vue serait de faire correspondre à chaque encodage R un problème Π_R définie à partir du problème initial Π plus les contraintes imposées par R.

Le langage $L^{\mathcal{E}}$ identifie les certificats qui sont des solutions partielles satisfaisant sur le bord les conditions imposées par un encodage. D'un autre point de vue, nous pourrions dire que Π_R se base sur le langage $L^R = \{(G, S) \mid (G, S, R) \in L^{\mathcal{E}}\}$. Remarquons que d'un point de vu théorique, nous n'avons pas besoin de $L^{\mathcal{E}}$. Aucun des résultats de cette section n'y fait référence. Néanmoins nous avons jugé que ce langage joue un rôle important dans la

compréhension de ce qu'est un encodage. Soulignons que toutes nos applications utilisent un tel langage pour définir la fonction $f^{\mathcal{E}}$.

La fonction $f^{\mathcal{E}}$ peut être vue de deux façons. Si nous fixons G, la fonction $f^{\mathcal{E}}(G, \cdot) : R \mapsto f^{\mathcal{E}}(G, R)$ remplit la table de G. Si nous fixons R, la fonction $f^{\mathcal{E}}(\cdot, R) : G \mapsto f^{\mathcal{E}}(G, R)$ peut être vue comme la fonction d'optimisation du problème Π_R.

La taille de l'encodeur $s_{\mathcal{E}}(t)$ est une fonction qui, étant donnée la taille du bord (le sac considéré par la programmation dynamique), donne le nombre d'encodages définis sur un graphe t-bordé ; c'est-à-dire, le nombre d'entrées de la table.

Remarquons que nous demandons à ce que le générateur $\mathcal{C}^{\mathcal{E}}$ et le langage $L^{\mathcal{E}}$ et la fonction $f^{\mathcal{E}}$ soient simplement calculables (sans contraindre plus la complexité). Rappelons que $t = |\partial G|$ est une constante qui ne dépend que du problème. Donc, $\mathcal{C}^{\mathcal{E}}$, qui ne dépend que du bord, se calcule en temps constant. Par contre $f^{\mathcal{E}}$ dépend du graphe bordé ; cependant, nous montrons dans le lemme 19, qu'il nous suffit d'appliquer $f^{\mathcal{E}}$ sur des graphes de taille constante.

La propriété suivante assure qu'un encodeur est effectivement conçu pour résoudre un problème. Elle traduit l'idée que, arrivé à la fin de la programmation dynamique, la table associée au graphe entier (de bord vide) contient la réponse au problème. Elle nous servira de cas de base pour montrer le raffinement de l'équivalence canonique par notre équivalence (cf. définition 64).

Définition 62 : Π-encodeur
Un encodeur $\mathcal{E}$ est un Π-*encodeur* si $\mathcal{C}^{\mathcal{E}}(\emptyset)$ est un singleton $\{R_\emptyset\}$, tel que pour tout graphe 0-bordé $f^{\mathcal{E}}(G, R_\emptyset) = f^{\Pi}(G)$.
⌟

Au premier abord, il semblerait plus naturel d'avoir une condition imposant l'égalité entre $L^{\mathcal{E}}$ et L^{Π} : pour tout certificat S, $(G, S, R_\emptyset) \in L^{\mathcal{E}}$ si et seulement si $(G, S) \in L^{\Pi}$. Cette condition n'est ni nécessaire, ni suffisante. En effet, d'une part, puisque nous étudions des problèmes de décision, nous n'avons pas besoin de connaître toutes les solutions, il nous suffit d'en connaître une de taille optimum ; d'autre part, puisque la forme de la fonction $f^{\mathcal{E}}$ n'est pas contrainte, il est possible que $f^{\mathcal{E}}$ contredise l'intuition et ne soit pas défini à partir de $L^{\mathcal{E}}$.

La propriété de confinement ci-dessous assure qu'une table enregistrera peu de valeurs (cf. figure 3.5). Cela impliquera un petit nombre de classes pour la $\mathcal{E}$-équivalence (cf. définition 64). Elle joue un rôle similaire à celui de la *monotonicité* [8].

Définition 63 : confinement

Un encodeur $\mathcal{E}$ est *g-confiné* si il existe une fonction $g : \mathbb{N} \to \mathbb{N}$ telle que pour tout graphe t-bordé G avec $\Lambda(G) = I$:

— *ou* $\{R \in \mathcal{C}^{\mathcal{E}}(I) \mid f^{\mathcal{E}}(G, R) \neq +\infty\} = \emptyset$

— *ou* $\max_{R \in \mathcal{C}^{\mathcal{E}}(I)}\{f^{\mathcal{E}}(G, R) \neq +\infty\} - \min_{R \in \mathcal{C}^{\mathcal{E}}(I)}\{f^{\mathcal{E}}(G, R) \neq +\infty\} \leqslant g(t)$.

⌟

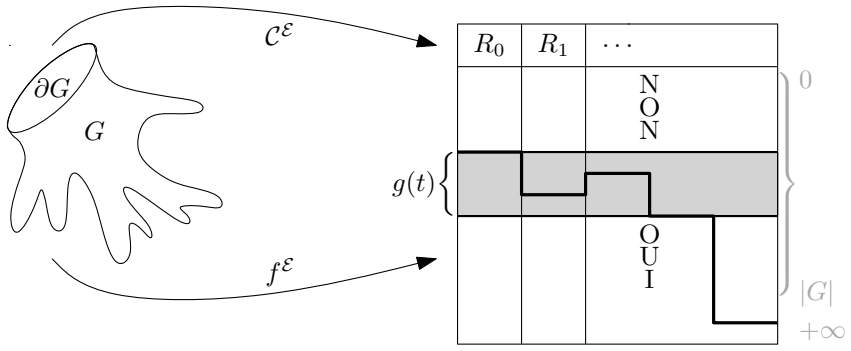


FIGURE 3.5 – Représentation schématique d'un encodeur confiné (cf. définitions 61 et 63). A partir d'un graphe t-bordé, la fonction $\mathcal{C}^{\mathcal{E}}$ génère les entrées de la table, la fonction $f^{\mathcal{E}}$ remplie cette table avec les tailles de solutions possibles, ou la valeur $+\infty$. Toutes les valeurs (sauf $+\infty$) sont comprises dans un intervalle de largeur $g(t)$. La valeur enregistrée pour un encodage R_i délimite l'intervalle pour lequel la réponse au problème Π_{R_i} est OUI de celui ou la réponse est NON.

3.1.3 Relation de $\mathcal{E}$-équivalence

Étant donné un encodeur $\mathcal{E}$, nous définissions maintenant la relation de $\mathcal{E}$-équivalence sur les graphes bordés, laquelle traduit le fait que deux graphes ont le même comportement vis à vis de la programmation dynamique. Nous voulons que cette $\mathcal{E}$-équivalence nous permette de faire des remplacements valides (comme avec l'équivalence canonique) et qu'elle soit facile à calculer (d'où l'usage de la programmation dynamique). En fait, pour des raisons techniques, nous définissons plusieurs relations dont les unes sont raffinées par les autres ; mais qui intuitivement joue le même rôle. Ensuite, grâce à la propriété de confinement, nous montrons que le nombre de classes de la $\mathcal{E}$-équivalence est petit (cf. lemme 16). Puis, nous définissons une propriété sur la $\mathcal{E}$-équivalence qui nous permet de prouver que la $\mathcal{E}$-équivalence est un raffinement de l'équivalence canonique (cf. lemme 17). Enfin, nous calculons

un représentant de petite taille pour chacune des classes d'équivalence (cf. lemme 18).

C'est ce représentant qui nous permettra d'effectuer le remplacement dans la sous-section suivante.

Intuitivement, deux graphes t-bordés sont $\mathcal{E}$-équivalents s'ils ont les mêmes tables à un décalage près, autrement dit, si la table du second peut être obtenue en ajoutant une constante à chaque valeur de la table du premier (cf. figure 3.6). Remarquons que pour effectuer le remplacement de protrusion, nous n'avons besoin de définir la $\mathcal{E}$-équivalence que sur $\mathcal{F}_t$, les graphes bordés de largeur arborescente bornée ; cependant pour des raisons techniques nous avons besoin d'une définition plus générale (dans la preuve du lemme 17).

Définition 64 : $\mathcal{E}$-équivalence

Étant donné un encodeur $\mathcal{E}$, deux graphes bordés $G_1, G_2 \in \mathcal{B}_t$ sont *$\mathcal{E}$-équivalents*, noté $G_1 \sim^*_{\mathcal{E},t} G_2$, si :

— $\Lambda(G_1) = \Lambda(G_2) = I$;

— *et* il existe un entier $\Delta_{\mathcal{E},t}(G_1, G_2)$ tel que pour tout encodage $R \in \mathcal{C}^{\mathcal{E}}(I)$, la fonction d'optimisation vérifie
$f^{\mathcal{E}}(G_1, R) = f^{\mathcal{E}}(G_2, R) - \Delta_{\mathcal{E},t}(G_1, G_2)$.

Nous notons $\cdot \sim_{\mathcal{E},t} \cdot$ la restriction de la relation $\cdot \sim^*_{\mathcal{E},t} \cdot$ lorsque les graphes G_1, G_2 appartiennent à $\mathcal{F}_t$.

La *fonction de transposition* est la fonction $\Delta_{\mathcal{E},t} : \mathcal{B}_t \times \mathcal{B}_t \to \mathbb{Z}$. Dans le cas où $f^{\mathcal{E}}(G_1, R) = f^{\mathcal{E}}(G_2, R) = +\infty$ pour tout encodage R, nous posons arbitrairement $\Delta_{\mathcal{E},t}(G_1, G_2) = 0$.

⌟

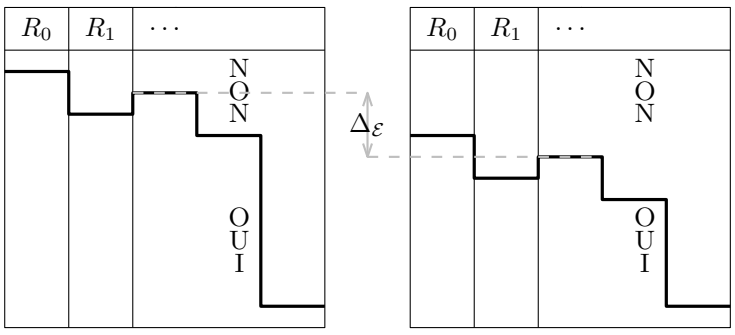


FIGURE 3.6 – Représentation schématique de la définition de $\mathcal{E}$-équivalence ((cf. définition 64)). Les valeurs diffèrent d'une constante $\Delta_{\mathcal{E},t}$ (sauf pour la valeur $+\infty$), les graphes bordés correspondants sont équivalents.

Nous avons besoin de raffiner la $\mathcal{E}$-équivalence par une classe de graphes. En effet, nous considérons des problèmes restreints à une classe de graphes, et, pour respecter la définition de noyau, nous voulons que le graphe réduit soit aussi dans la classe (sans quoi nous obtenons un binoyau) . Pour cela, il faut que le remplacement de protrusion préserve la classe de graphes.

Définition 65 : $\mathcal{E}$-équivalence sur $\mathcal{G}$

Étant donné un encodeur $\mathcal{E}$, et une classe de graphes $\mathcal{G}$, deux graphes bordés $G_1, G_2 \in \mathcal{B}_t$ sont *$\mathcal{G}$-équivalents* $G_1 \approx_{\mathcal{G},t} G_2$ si :

— pour tout $G^- \in \mathcal{B}_t$, $G^- \oplus G_1 \in \mathcal{G}$ si et seulement si $G^- \oplus G_2 \in \mathcal{G}$.

deux graphes bordés $G_1, G_2 \in \mathcal{B}_t$ sont *$\mathcal{E},\mathcal{G}$-équivalents* $G_1 \sim_{\mathcal{E},\mathcal{G},t} G_2$ si :

— $G_1 \sim^*_{\mathcal{E},t} G_2$,

— ^{et} $G_1 \approx_{\mathcal{G},t} G_2$.

Nous notons $\cdot \sim_{\mathcal{E},\mathcal{G},t} \cdot$ la restriction de la relation $\cdot \sim^*_{\mathcal{E},\mathcal{G},t} \cdot$ lorsque les graphes G_1, G_2 appartiennent à $\mathcal{F}_t$. ⌟

Dans la suite de ce chapitre, nous ne considérerons que des classes de graphes dont l'appartenance peut être décrite en logique monadique du second ordre ; comme par exemple les classes des graphes excluant un mineur. Par conséquent nous savons que $\cdot \approx_{\mathcal{G},t} \cdot$ a un nombre fini de classes d'équivalence [24, 19] ; appelons le $r_{\mathcal{G},t}$.

Nous pouvons donc énoncer le lemme suivant, qui donne une borne supérieure sur le nombre de classes de la $\mathcal{E}$-équivalence (c'est-à-dire, la relation a un index fini). Intuitivement, la preuve va de soi : le nombre d'encodages est borné par $s_{\mathcal{E}}(t)$ et le nombre de valeurs possible (à décalage prés) est borné par le g-confinement.

Lemme 16

Soit $\mathcal{G}$ une classe de graphes dont l'appartenance peut être décrite en logique monadique du second ordre. Soit $\mathcal{E}$ un encodeur, soit $g : \mathbb{N} \to \mathbb{N}$ une fonction, et $t \in \mathbb{N}$. Si $\mathcal{E}$ est g-confiné alors la relation $\cdot \sim^*_{\mathcal{E},\mathcal{G},t} \cdot$ a au plus $r(\mathcal{E}, g, t, \mathcal{G}) := (g(t)+2)^{s_{\mathcal{E}}(t)} \cdot 2^t \cdot r_{\mathcal{G},t}$ classes d'équivalence.

En particulier, la relation $\cdot \sim_{\mathcal{E},\mathcal{G},t} \cdot$ a au plus $r(\mathcal{E}, g, t, \mathcal{G})$ classes.

⌞ ⌟

Démonstration. Nous montrons d'abord que la relation $\cdot \sim^*_{\mathcal{E},t} \cdot$ a un index fini. Soit $I \subseteq \{1, \dots, t\}$. Par confinement, nous savons que, pour tout $G \in \mathcal{B}_t$ avec $\Lambda(G) = I$, la fonction $f^{\mathcal{E}}(G, \cdot)$ peut prendre au plus $g(t)+2$ valeurs distinctes ($g(t)+1$ valeurs entières et possiblement la valeur $+\infty$). Par conséquent, le nombre de classes de $\sim^*_{\mathcal{E},t}$ sur tous les graphes $G \in \mathcal{B}_t$

avec $\Lambda(G) = I$ est au plus de $(g(t)+2)^{|\mathcal{C}^{\mathcal{E}}(I)|}$. Puisque le nombre de sous-ensembles de $\{1,\dots,t\}$ est 2^t, nous en concluons que le nombre total de classes d'équivalence de $\cdot \sim^*_{\mathcal{E},t} \cdot$ est au plus de $(g(t)+2)^{s\varepsilon(t)} \cdot 2^t$.

Ensuite, nous avons déjà remarqué que $\cdot \approx_{\mathcal{G},t} \cdot$ a un index fini, car $\mathcal{G}$ peut être décrite en logique monadique du second ordre.

Enfin, puisque la relation $\cdot \sim^*_{\mathcal{E},\mathcal{G},t} \cdot$ est le produit cartésien des relations $\cdot \sim^*_{\mathcal{E},t} \cdot$ et $\cdot \approx_{\mathcal{G},t} \cdot$, le nombre de classes d'équivalence de $\cdot \sim^*_{\mathcal{E},\mathcal{G},t} \cdot$ est au plus de $(g(t)+2)^{s\varepsilon(t)} \cdot 2^t \cdot r_{\mathcal{G},t}$.

□

Maintenant, nous voulons nous assurer qu'un encodeur $\mathcal{E}$ est effectivement adapté pour exécuter une programmation dynamique sur un graphe dans $\mathcal{F}_t \cap \mathcal{G}$. Rappelons que, avec la programmation dynamique, la table associée à un séparateur (le bord d'un sous-graphe) se calcule à partir des tables associées aux séparateurs précédents (c'est-à-dire, aux descendants dans la décomposition arborescente). Un encodeur $\mathcal{E}$ nous fournit un cadre formel pour générer des tables ; nous voulons que ces tables puissent aussi être calculées à partir des descendants. Ce comportement est capturé par la propriété définie ci-dessous.

La définition d'équivalence adaptée nécessite de considérer un situation compliquée (cf. figure 3.7) et d'en nommer chaque élément. Cette situation devra être décrite à nouveau pour chacune de nos applications. Afin de faciliter la lecture des preuves, nous fixons d'abord des notations pour chaque élément. Par analogie avec la notation G_B pour désigner le sous-graphe de $G \in \mathcal{F}_t$ associé au sac B de la décomposition arborescente de B, nous noterons G_B un sous-graphe de $G \in \mathcal{B}_t$ séparé du reste de G par un bord B.

Définition 66

Soit $G \in \mathcal{B}_t$ un graphe bordé avec $\partial G = A$ et $B \subseteq V(G)$ un séparateur de G avec $|B| \leqslant t$ et $B \neq A$. Soient G_B et G^- deux sous-graphes de G considérés comme des graphes t-bordés de bord B, tels que $A \subseteq V(G^-)$ et $G_B \oplus G^- = G$. Soit $G'_B \in \mathcal{B}_t$ tel que $G_B \sim^*_{\mathcal{E},\mathcal{G},t} G'_B$. Enfin, soit $G' = G'_B \oplus G^-$ avec $\partial G' = A$ le graphe obtenu à partir G en remplaçant le sous-graphe G_B par G'_B.

⌟

Bien que cette description soit valable pour tout graphe bordé, il est plus facile d'en expliquer l'intuition sur un graphe $G \in \mathcal{F}_t$. Le séparateur B est un sac de la décomposition arborescente (enracinée en A) de G ; G_B est le graphe induit par le sous-arbre enraciné en B ; et G^- est le graphe induit par l'autre composante connexe de l'arbre (contenant A). Remarquons que,

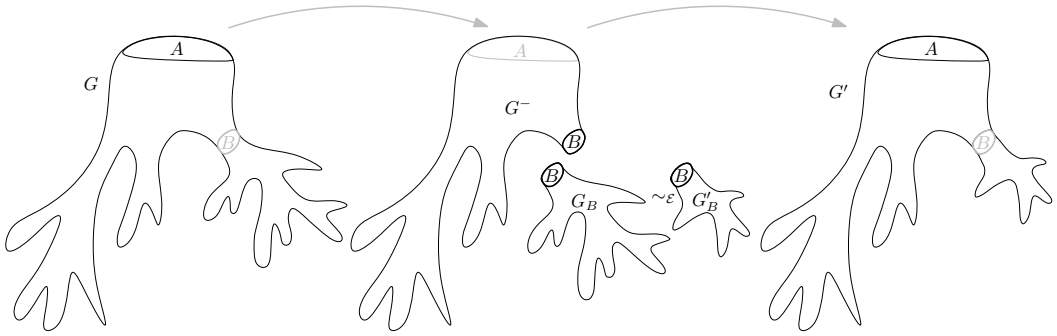


FIGURE 3.7 – Représentation schématique des notations posées dans la définition 66. Le graphe bordé G' est obtenu à partir de G en échangeant deux sous-graphe $\mathcal{E}$-équivalents G_B, G'_B. Intuitivement, G et G' seront deux protrusions et B sera un sac de leur décomposition arborescente.

d'après la contrainte $A \subseteq V(G^-)$, A peut intersecter B, mais ne doit pas le traverser (autrement dit, $A \cap V(G_B) \setminus B = \emptyset$).

Nous pouvons maintenant de finir une équivalence adaptée. Ici aussi, pour les mêmes raisons techniques (dans la preuve du lemme 17), nous posons une définition générale pour les graphes bordés de $\mathcal{B}_t$, alors que notre objectif et d'utiliser cette propriété sur les graphes de $\mathcal{F}_t$ (cf. figure 3.8).

Définition 67 : équivalence adaptée

Une relation d'équivalence $\cdot \sim^*_{\mathcal{E},\mathcal{G},t} \cdot$ est *adaptée* si pour tout graphe bordé $G \in \mathcal{B}_t$ avec A, B, G_B, G'_B, G' comme décrits dans la définition 66 :

— $G \sim^*_{\mathcal{E},\mathcal{G},t} G'$;

— *et* $\Delta_{\mathcal{E},t}(G, G') = \Delta_{\mathcal{E},t}(G_B, G'_B)$.

⌟

Le fait suivant énonce que pour montrer que $\cdot \sim^*_{\mathcal{E},\mathcal{G},t} \cdot$ est adaptée il suffit de montrer que $\cdot \sim^*_{\mathcal{E},t} \cdot$ est adaptée. Nous utiliserons ce fait dans nos applications (cf. sections 3.3, 3.4, 3.5) lorsqu'il s'agira de montrer que les encodeurs que nous proposons ont des équivalences adaptées, nous n'aurons pas à tenir compte de la classe de graphes considérée. Intuitivement, ce fait traduit l'idée que raffiner les relations $\cdot \sim^*_{\mathcal{E},t} \cdot$ et $\cdot \sim_{\mathcal{E},t} \cdot$ par $\cdot \approx_{\mathcal{G},t} \cdot$ (pour obtenir $\cdot \sim^*_{\mathcal{E},\mathcal{G},t} \cdot$ et $\cdot \sim_{\mathcal{E},\mathcal{G},t} \cdot$) est simplement une astuce pour appliquer notre cadre à des problèmes restreints à la classe de graphes $\mathcal{G}$, et que cela ne modifie pas les propriétés de l'encodeur.

Fait 5

Soit $G \in \mathcal{B}_t$ avec un séparateur B, soit $G_B \sim_{\mathcal{E},\mathcal{G},t} G'_B$ et $G' \in \mathcal{B}_t$ comme décrit dans la définition 66. Si $G \sim^*_{\mathcal{E},t} G'$ alors $G \sim^*_{\mathcal{E},\mathcal{G},t} G'$.

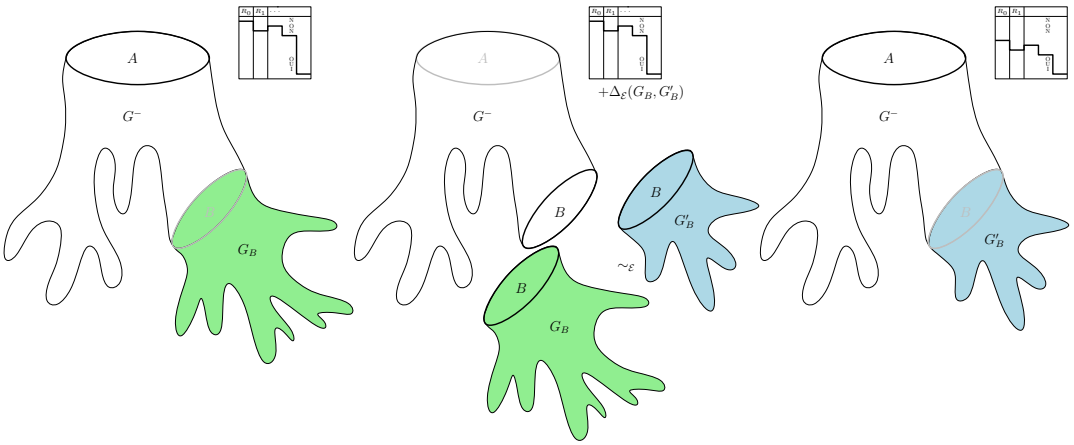


FIGURE 3.8 – Représentation schématique de la définition d'une équivalence adaptée (cf. définition 67). La table du graphe bordé G' peut être calculée à partir de la table de G en ajoutant la constante $\Delta_{\mathcal{E},t}(G_B, G'_B)$ qui est donnée par les tables de G_B et G'_B.

Démonstration. Nous rappellons que d'apres la définition 66, $G = G^- \oplus G_B$ et $G' = G^- \oplus G'_B$. Supposons que $G \sim^*_{\mathcal{E},t} G'$. Pour montrer que $G \sim^*_{\mathcal{E},\mathcal{G},t} G'$, il suffit de montrer que $G \approx_{\mathcal{G},t} G'$. Soit $H \in \mathcal{B}_t$. Nous voulons montrer que $G \oplus H \in \mathcal{G}$ si et seulement si $G' \oplus H \in \mathcal{G}$. Nous savons que $G \oplus H = (G_B \oplus G^-) \oplus H = G_B \oplus (G^- \oplus H)$ (et de même pour G'). Puisque $G_B \approx_{\mathcal{G},t} G'_B$ il s'en suit que $G \oplus H = G_B \oplus (G^- \oplus H) \in \mathcal{G}$ si est seulement si $G'_B \oplus (G^- \oplus H) = G' \oplus H \in \mathcal{G}$. □

Afin d'effectuer le remplacement de protrusion, nous devons vérifier que la relation $\cdot \sim^*_{\mathcal{E},t} \cdot$ est un raffinement de la relation $\cdot \equiv_\Pi \cdot$. Cela signifie que le remplacement d'une protrusion par une protrusion $\mathcal{E}$-équivalente conserve le comportement de l'instance par rapport au problème. Intuitivement, nous prouvons le lemme suivant par induction : pour les graphes (sans bord), le raffinement est contraint par la définition d'un Π-encodeur ; pour les graphes bordés, nous pouvons nous ramener au cas des graphes (sans bord) par la définition d'une relation adaptée.

Lemme 17

Soit Π un problème, soit $\mathcal{E}$ un encodeur, soit $g : \mathbb{N} \to \mathbb{N}$, et soit $G_1, G_2 \in \mathcal{B}_t$. Si $\mathcal{E}$ est un Π-encodeur g-confiné tel que $\sim^*_{\mathcal{E},\mathcal{G},t}$ est adaptée, alors $G_1 \sim^*_{\mathcal{E},\mathcal{G},t} G_2$ implique que :

— $G_1 \equiv_\Pi G_2$;

— *et* $\Delta_\Pi(G_1, G_2) = \Delta_{\mathcal{E},t}(G_1, G_2)$.

En particulier, c'est également vrai pour $\cdot \sim_{\mathcal{E},\mathcal{G},t} \cdot$ lorsque $G_1, G_2 \in \mathcal{F}_t$.

Démonstration. Soit $\mathcal{E} = (\mathcal{C}^\mathcal{E}, L^\mathcal{E}, f^\mathcal{E})$ un Π-encodeur et soit $G_1, G_2 \in \mathcal{B}_t$ tels que $G_1 \sim_{\mathcal{E},t} G_2$. Nous voulons montrer que pour tout $H \in \mathcal{B}_t$ et pour tout $k \in \mathbb{N}$, $(G_1 \oplus H, k) \in \Pi$ si et seulement si $(G_2 \oplus H, k + \Delta_{\mathcal{E},t}(G_1, G_2)) \in \Pi$.

Supposons que $(G_1 \oplus H, k) \in \Pi$ (par symétrie le même argument s'applique à G_2). Puisque $G_1 \oplus H$ est un graphe 0-bordé et que $\mathcal{E}$ est un Π-encodeur, nous avons $f^\mathcal{E}(G_1 \oplus H, R_\emptyset) = f^\Pi(G_1 \oplus H) \leqslant k$. Puisque $\cdot \sim^*_{\mathcal{E},\mathcal{G},t} \cdot$ est adaptée et que $G_1 \sim^*_{\mathcal{E},\mathcal{G},t} G_2$, nous avons $(G_1 \oplus H) \sim^*_{\mathcal{E},t} (G_2 \oplus H)$ et $\Delta_{\mathcal{E},t}(G_1 \oplus H, G_2 \oplus H) = \Delta_{\mathcal{E},t}(G_1, G_2)$. Par conséquent, $f^\mathcal{E}(G_2 \oplus H, R_\emptyset) = f^\mathcal{E}(G_1 \oplus H, R_\emptyset) + \Delta_{\mathcal{E},t}(G_1, G_2) \leqslant k + \Delta_{\mathcal{E},t}(G_1, G_2)$ (rappelons que $G_2 \oplus H$ est 0-bordé). Enfin, puisque $\mathcal{E}$ est un Π-encodeur, $f^\Pi(G_2 \oplus H) = f^\mathcal{E}(G_2 \oplus H, R_\emptyset)$ et donc $(G_2 \oplus H, k + \Delta_{\mathcal{E},t}(G_1, G_2)) \in \Pi$. □

A partir d'ici, nous avons besoin de considérer des graphes de $\mathcal{F}_t$ (c'est-à-dire, bordés, de largeur arborescente bornée, et tels que le bord soit inclus dans un sac). En effet, dans les preuves à venir (cf. lemme 18), nous avons besoin d'une décomposition arborescente des graphes bordés. Par conséquent nous n'utiliserons que des cas particuliers des lemmes 16 et 17.

Nous avons maintenant posé tous les outils pour pouvoir affirmer qu'un remplacement est valide. Il nous reste à choisir un représentant approprié, c'est-à-dire un représentant de taille constante, et qui ne fait pas croître le paramètre (afin de respecter la définition 35 des noyaux). La contrainte sur le paramètre est formalisée par la définition ci-dessous. Dans le lemme qui suit, nous montrons que la contrainte sur la taille est vérifiée, c'est-à-dire, nous montrons qu'un tel représentant existe et donnons une borne de sa taille (une borne supérieure sur la taille d'un plus petit représentant progressif).

Intuitivement, la preuve se base sur le fait que, étant donnée une décomposition arborescente du graphe, chaque sac de la décomposition à une table distincte, sans quoi, un remplacement permettrait de décroître la taille de représentant ; par conséquent, la décomposition arborescente contient peu de nœuds et donc le graphe contient peu de sommets.

Définition 68 : représentant progressif

Étant donné un encodeur $\mathcal{E}$ et une classe d'équivalence $\mathfrak{C} \subseteq \mathcal{F}_t$ de la

relation $\cdot \sim_{\mathcal{E},\mathcal{G},t} \cdot$, un *représentant progressif* de $\mathfrak{C}$ est un graphe bordé G tel que pour tout $G' \in \mathfrak{C}$, la constance de transposition $\Delta_{\mathcal{E},t}(G, G')$ est négative.

⌟

Lemme 18

Soit $\mathcal{G}$ une classe de graphes dont l'appartenance peut être exprimé en logique monadique du second ordre. Soit $\mathcal{E}$ un encodeur, $g : \mathbb{N} \to \mathbb{N}$ une fonction, et t un entier. Si $\mathcal{E}$ est g-confiné et $\cdot \sim^*_{\mathcal{E},\mathcal{G},t} \cdot$ est adaptée, alors toute classe d'équivalence de $\cdot \sim_{\mathcal{E},\mathcal{G},t} \cdot$ admet un représentant progressif de taille au plus $b(\mathcal{E}, g, t, \mathcal{G}) := 2^{r(\mathcal{E},g,t,\mathcal{G})+1} \cdot t$, avec $r(\mathcal{E}, g, t, \mathcal{G})$ définie dans le lemme 16.

⌞ ⌟

Démonstration. Soit $\mathfrak{C}$ une classe d'équivalence de $\cdot \sim_{\mathcal{E},\mathcal{G},t} \cdot$. Nous montrons d'abord qu'il existe un représentent progressif dans $\mathfrak{C}$. Puisque $\Delta_{\mathcal{E},t}(G_1, G_2) = f^{\mathcal{E}}(G_1, R) - f^{\mathcal{E}}(G_2, R)$ pour tout encodage R (peu importe lequel, nous nous intéressons au décalage, qui est indépendant de R), $G \in \mathfrak{C}$ est progressif si $f^{\mathcal{E}}(G, R)$ est minimal dans l'ensemble $f^{\mathcal{E}}(\mathfrak{C}, R) = \{f(G, R) \mid G \in \mathfrak{C}\}$. Puisque $f^{\mathcal{E}}(\mathfrak{C}, R)$ est un sous-ensemble de $\mathbb{N}$, il admet un élément minimal (en d'autres termes, la relation d'ordre $G_1 \preccurlyeq G_2$, définie par $\Delta_{\mathcal{E},t}(G_1, G_2) \leqslant 0$, est bien fondée) ; par conséquent, il existe un représentant progressif de $\mathfrak{C}$.

Maintenant, soit $G \in \mathcal{G}$ un représentant progressif de $\mathfrak{C}$ minimum en nombre de sommets. Nous montrons que G a au plus $2^{r(\mathcal{E},g,t,\mathcal{G})+1} \cdot t$ sommets. Soit $(T, \mathcal{X})$ une bonne décomposition arborescente bordée de G, de largeur au plus $t - 1$.

D'abord, remarquons que pour chaque nœud x de T, le graphe G_x est un représentant progressif de sa classe, disons $\mathfrak{C}'$. Supposons par contradiction que ce n'est pas le cas, et soit G'_x un représentant progressif de $\mathfrak{C}'$. Puisque G'_x est progressif et G_x ne l'est pas, $\Delta_{\mathcal{E},t}(G'_x, G_x) < 0$. Soit G' le graphe obtenu en remplaçant G_x par G'_x dans G. Puisque $\cdot \sim_{\mathcal{E},\mathcal{G},t} \cdot$ est adaptée, il s'en suit que $G \sim_{\mathcal{E},\mathcal{G},t} G'$ et $\Delta_{\mathcal{E},t}(G', G) = \Delta_{\mathcal{E},t}(G'_x, G_x) < 0$; contradiction avec l'hypothèse que G est progressif.

Ensuite, nous montrons que $G_x \nsim_{\mathcal{E},\mathcal{G},t} G_y$ pour toute paire de nœuds x, y de T situés sur un chemin depuis la racine jusqu'à une feuille de T. Supposons par contradiction qu'il existe deux nœuds x, y sur un tel chemin et tels que $G_x \sim_{\mathcal{E},\mathcal{G},t} G_y$. Soit $\mathfrak{C}'$ la classe de G_x et G_y. D'après l'argumentation précédente, G_x et G_y sont progressifs. Supposons que $G_y \subsetneq G_x$ (c'est-à-dire, x est plus proche de la racine), et soit G' le graphe obtenu en remplaçant G_x par G_y dans G. Encore une fois, puisque $\cdot \sim_{\mathcal{E},\mathcal{G},t} \cdot$ est adaptée, il s'en suit que $G \sim_{\mathcal{E},\mathcal{G},t} G'$ et $\Delta_{\mathcal{E},t}(G', G) = \Delta_{\mathcal{E},t}(G_y, G_x) = 0$. Par conséquent G' est un représentant progressif de $\mathfrak{C}$ avec $|G'| < |G|$; contradiction avec la minimalité de G.

Enfin, puisque $G_x \nsim_{\mathcal{E},\mathcal{G},t} G_y$ pour toute paire de nœuds x, y de T sur un chemin depuis la racine jusqu'à une feuille, la hauteur de T est au plus le nombre de classes d'équivalence de $\cdot \sim_{\mathcal{E},\mathcal{G},t} \cdot$, c'est-à-dire au plus $r(\mathcal{E}, g, t, \mathcal{G})$ d'après le lemme 16. Puisque T est un arbre binaire, nous en déduisons que $|T| \leqslant 2^{r(\mathcal{E},g,t,\mathcal{G})+1} - 1$. Finalement, puisque $|G| \leqslant |T| \cdot t$, nous obtenons $|G| \leqslant 2^{r(\mathcal{E},g,t,\mathcal{G})+1} \cdot t$.

□

Remarquons que pour ce lemme, nous ne montrons pas la constructivité du représentant. En effet, nous n'avons besoin que de l'existence du représentant et d'une borne explicite sur sa taille (dans la preuve du lemme 19). Nous pourions vouloir dériver un algorithme (par programmation dynamique) de la preuve : les nœuds de la décomposition arborescente sont parcourus des feuilles jusqu'à la racine, à chaque nœud la table est calculée, et, si celle-ci est équivalente à une table déjà calculée, le remplacement est effectué. Cependant cet algorithme ne garantie pas que le représentant ainsi obtenu soit progressif.

3.1.4 Remplacement explicite et générique

Il nous reste à formuler notre théorème principal, qui prouve la correction de la règle de remplacement de protrusion. Nous avons montré que, sous certaines conditions, la relation $\cdot \sim_{\mathcal{E},\mathcal{G},t} \cdot$ raffinait $\cdot \equiv_\Pi \cdot$ et par conséquent que remplacer une protrusion par une protrusion $\mathcal{E}$-equivalente était valide (lemme 17). De plus, nous avons montré que chaque classe d'équivalence admet un représentant de petite taille qui n'augmente pas le paramètre (lemme 18). Il nous faut donc expliquer comment construite ce représentant. Puis nous pourrons rassembler tous les éléments pour prouver la correction de notre réduction.

Le lemme suivant indique comment, étant donnée une borne sur la taille d'un représentant, calculer une petit représentant afin d'effectuer une remplacement de protrusions.

Lemme 19

Soit $\mathcal{G}$ une classe de graphes, soit $\mathcal{E}$ un encodeur, $g : \mathbb{N} \to \mathbb{N}$ une fonction, et $t \in \mathbb{N}$ tels que $\mathcal{E}$ est g-confiné et $\cdot \sim^*_{\mathcal{E},\mathcal{G},t} \cdot$ est adaptée. Étant donnée une borne supérieure $b \geqslant t$ sur la taille des plus petits représentants progressifs, étant donnée une t-protrusion Y d'un graphe, une t-protrusion Y', de taille au plus b, tel que $Y \sim_{\mathcal{E},\mathcal{G},t} Y'$ et $\Delta_{\mathcal{E},t}(Y', Y) \leqslant 0$ peut être calculée en temps $O(|V(Y)|)$, avec une constante dépendante de $\mathcal{E}, g, b, \mathcal{G}, t$.

⌞ ⌟

Démonstration. Soit $\mathcal{E} = (\mathcal{C}^{\mathcal{E}}, L^{\mathcal{E}}, f^{\mathcal{E}})$ l'encodeur en question. Nous considérons d'abord l'ensemble $\mathfrak{R}$ contenant tous les graphes bordés de $\mathcal{F}_t$ de taille au plus $b+1$. L'ensemble de ces graphes, ainsi qu'une bonne décomposition arborescente bordée de largeur au plus $t-1$ pour chaque graphe, peuvent être générés en un temps ne dépendant que de b et t (qui seront des constantes une fois le problème fixé). Par hypothèse, la taille des plus petits représentants progressifs de toutes les classes est au plus b, donc, $\mathfrak{R}$ contient un représentant progressif de chaque classe. Nous partitionnons $\mathfrak{R}$ en classes d'équivalence de $\cdot \sim_{\mathcal{E},\mathcal{G},t} \cdot$: pour chaque graphe $G \in \mathfrak{R}$ et pour tout encodage $R \in \mathcal{C}^{\mathcal{E}}(\Lambda(G))$, nous calculons la valeur de $f^{\mathcal{E}}(G,R)$; puisque $L^{\mathcal{E}}$ et $f^{\mathcal{E}}$ sont calculables, en un temps dépendant de $\mathcal{E}, g, t$, et $b \geqslant |G|$ (c'est-à-dire, en temps constant, une fois Π et $\mathcal{E}$ fixés). Par conséquent, pour toute paire de graphes $G_1, G_2 \in \mathfrak{R}$ nous pouvons décider si $G_1 \sim_{\mathcal{E},\mathcal{G},t} G_2$, et, le cas échéant, calculer la constante de transposition $\Delta_{\mathcal{E},t}(G_1, G_2)$; en un temps dépendant de $\mathcal{E}, g, t, b$ et $\mathcal{G}$.

Étant donnée une t-protrusion de n sommets et de bord $\partial(Y)$, nous commençons par calculer une bonne décomposition arborescente bordée $(T, \mathcal{X}, r)$ de Y (avec $X_r = \partial Y$) en temps $f(t) \cdot |V(Y)|$; en utilisant l'algorithme de Bodlaender [6, 44]. En assignant les labels distincts dans $[1,t]$ aux sommets de ∂Y, nous pouvons naturellement considérer la protrusion Y comme un graphe t-bordé. Nous pouvons supposer que $\Lambda(Y) = [1,t]$. De la même façon, nous pouvons labeller arbitrairement chaque sac de la décomposition. Par conséquent, chaque nœud $x \in V(T)$ définit naturellement une protrusion Y_x dans Y, avec une bonne décomposition arborescente bordée (enracinée en x). Nous pouvons maintenant décrire la procédure de remplacement de protrusion.

Nous traitons les sacs de $(T, \mathcal{X}, r)$ des feuilles vers la racine, jusqu'au premier nœud $x \in V(T)$ tel que $|V(Y_x)| = b+1$ (remarquons que puisque $(T, \mathcal{X}, r)$ est bonne, lors du traitement des sacs, au plus un sommet est introduit à chaque étape, de plus $b \geqslant t$, donc un tel x existe). Puisque $|V(Y_x)| = b+1$ le graphe bordé Y_x est dans $\mathfrak{R}$, nous pouvons donc trouver en temps constant son représentant progressif Y'_x, avec au plus b sommets et une constante de transposition $\Delta_{\mathcal{E},t}(Y'_x, Y_x) \leqslant 0$. Soit Y'' le graphe obtenu en remplaçant Y_x par Y'_x dans Y. Nous avons $|V(Y'')| < |V(Y)|$ (remarquons que cette opération de remplacement permet de construire directement une bonne décomposition arborescente bordée de Y''). Puisque $\cdot \sim^*_{\mathcal{E},\mathcal{G},t} \cdot$ est adaptée, il s'en suit que $Y \sim_{\mathcal{E},\mathcal{G},t} Y''$ et $\Delta_{\mathcal{E},t}(Y'', Y) = \Delta_{\mathcal{E},t}(Y'_x, Y_x) \leqslant 0$.

Nous appliquons récursivement cette procédure de remplacement jusqu'à obtenir une protrusion Y' de taille au plus b et telle que $Y \sim_{\mathcal{E},\mathcal{G},t} Y'$. La constante de transposition correspondante $\Delta_{\mathcal{E},t}(Y', Y)$ peut être facilement calculée par sommation des constantes de transposition à chaque étape de

remplacement. En particulier, puisque chaque remplacement introduit un représentant progressif, nous avons bien $\Delta_{\mathcal{E},t}(Y'', Y) \leqslant 0$. Enfin, nous pouvons supposer que le nombre de nœuds dans une bonne décomposition arborescente de Y est en $O(n)$ [44], et donc, le temps d'exécution de l'algorithme est de l'ordre de $O(n)$ (la constante dépend uniquement de $\mathcal{E}, g, b, t$, et $\mathcal{G}$, qui sont fixés).

□

Remarquons que, lorsque nous labellons les sacs de la décomposition arborescente de la protrusion Y, nous pouvons choisir les labels de sorte à ce qu'ils soient cohérents. Les labels du sac racine peuvent être transférés aux sommets dans tous les sacs de $(T, \mathcal{X}, r)$ en effectuant une procédure de décalage standard [8] : dans tous les cas, les sacs fils sont labellés comme le sac père ; et lorsqu'un sommet est ajouté dans dans le sac fils, son label est un label inutilisé dans le sac père.

Nous pouvons maintenant assembler tous les éléments pour énoncer notre théorème principal. Nous affirmons que, étant donnée une instance d'un problème, si ce problème admet un encodeur (avec des propriétés adéquates) alors nous pouvons calculer une instance équivalente par remplacement de protrusion. Afin de rendre notre algorithme le plus constructif possible, il nous faut restreindre le problème à une classe de graphes excluant un mineur (ou un mineur topologique).

Théorème 4

Soit $\mathcal{G}$ une classe de graphes excluant un mineur (respectivement, un mineur topologique) H. Soit Π un problème restreint sur $\mathcal{G}$. Soit $\mathcal{E}$ un encodeur, $g : \mathbb{N} \to \mathbb{N}$, et soit $t \in \mathbb{N}$ tels que $\mathcal{E}$ est un Π-encodeur g-confiné et tels que $\cdot \sim^*_{\mathcal{E},\mathcal{G},t} \cdot$ est adaptée. Étant donnée une instance (G, k) de Π et une t-protrusion Y de G, une instance équivalente $(G - (Y - \partial(Y)) \oplus Y', k')$ de Π peut être calculée en temps $O(|V(Y)|)$, avec Y' une t-protrusion (de $(G - (Y - \partial(Y)))$ de taille au plus $b(\mathcal{E}, g, t, \mathcal{G})$ et avec $k' \leqslant k$.

Démonstration. D'après le lemme 16, le nombre de classes d'équivalence de la relation $\cdot \sim_{\mathcal{E},\mathcal{G},t} \cdot$ est fini, et d'après le lemme 18 la taille d'un plus petit représentant progressif d'une classe est au plus $b(\mathcal{E}, g, t, \mathcal{G})$. Par conséquent, nous pouvons appliquer le lemme 19 qui nous fournira, en temps $O|V(Y)|$, une t-protrusion Y' de taille au plus $b(\mathcal{E}, g, t, \mathcal{G})$ telle que $Y \sim_{\mathcal{E},\mathcal{G},t} Y'$, et la constante de transposition $\Delta_{\mathcal{E},t}(Y', Y)$ telle que $\Delta_{\mathcal{E},t}(Y', Y) \leqslant 0$. Par voie de conséquence, si nous posons $k' := k + \Delta_{\Pi,t}(Y', Y)$ nous obtenons que (G, k) et $((G - (Y - \partial(Y))) \oplus Y', k')$ sont deux instances équivalentes de Π avec $k' \leqslant k$ et $|V(Y')| \leqslant b(\mathcal{E}, g, t, \mathcal{G})$.

□

Notre procédure de remplacement de protrusion peut servir, en particulier, à une extraction de noyau. Par exemple, s'il existe une décomposition en protrusions de l'instance, l'extraction de noyau découle naturellement de notre résultat précédant : il suffit de remplacer chaque protrusion par son représentant (cf. figure 3.9).

Corollaire 1

Soit $\mathcal{G}$ une classe de graphes excluant un mineur (respectivement, un mineur topologique) H. Soit Π un problème restreint sur $\mathcal{G}$. Soit $\mathcal{E}$ un encodeur, soit $g : \mathbb{N} \to \mathbb{N}$, et $t \in \mathbb{N}$ tels que $\mathcal{E}$ est un Π-encodeur g-confiné et tels que $\cdot \sim^*_{\mathcal{E},\mathcal{G},t} \cdot$ est adaptée.

Étant donnée une instance (G, k) de Π et une $(\alpha k, t)$-décomposition en protrusions de G, un noyau linéaire, de taille au plus $(1 + b(\mathcal{E}, g, t, \mathcal{G}))\alpha \cdot k$, peut être calculé.

Démonstration. Pour $i \in [1, \alpha \cdot k]$, nous appliquons l'algorithme polynomial fourni par le théorème 4 afin de remplacer toutes les protrusions Y_i par des protrusions Y'_i de taille au plus $b(\mathcal{E}, g, t, \mathcal{G})$ et afin de mettre à jour le paramètre k en conséquence. De cette façon, nous obtenons une instance équivalente (G', k') telle que $G' \in \mathcal{G}$, $k' \leqslant k$ et $|G'| \leqslant |V(Y_0)| + \alpha \cdot k \cdot (b(\mathcal{E}, g, t, \mathcal{G}) + 1)$.

□

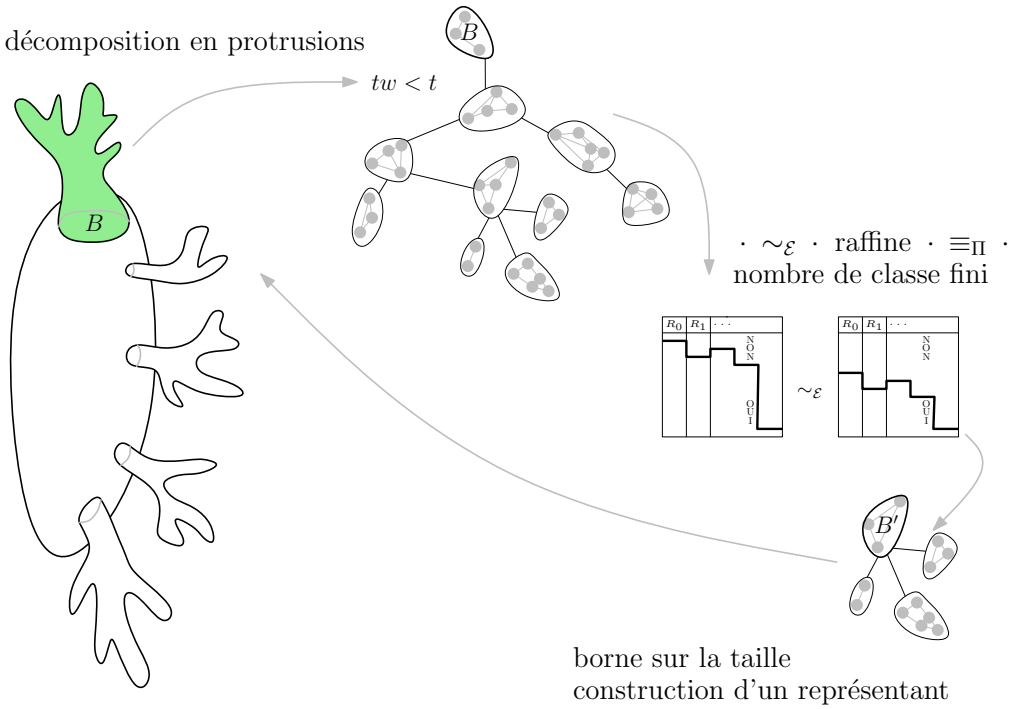


FIGURE 3.9 – Résumé schématique de l'algorithme d'extraction de noyau dans le corollaire 1. Dans une décomposition en protrusions, une protrusion, de bord B et de largeur arborescente t est sélectionnée. Une programmation dynamique est effectuée, sur sa décomposition arborescente, afin d'en déterminer la classe de $\mathcal{E}$-équivalence. Un représentant de cette classe peut être calculé grâce à la borne sur la taille des représentants. La protrusion est remplacée par son représentant dans la décomposition.

3.2 Outils pour les applications

Nous allons maintenant décrire comment utiliser notre cadre générique et les outils nécessaires pour pouvoir l'appliquer aux problèmes qui nous intéressent, assavoir : r-Domination, r-Indépendance, $\mathcal{F}$-Paquetage. Montrer qu'un problème entre dans notre cadre se fait en deux étapes : d'abord identifier une décomposition en protrusions de façon constructive pour chaque instance positive (ou négative, selon le problème) du problème, puis définir un encodeur du problème et montrer que l'équivalence associée est adaptée à la programmation dynamique. Pour conclure, il suffit finalement d'appliquer notre corollaire 1 qui fournit le noyau.

3.2.1 Théorème de décomposition en protrusions

Pour nos applications, nous avons besoin de construire une décomposition en protrusions des instances pour pouvoir appliquer le corollaire 1. Plus précisément, nous voulons que, dans une décomposition, le nombre de protrusions soit linéaire en fonction du paramètre (suffisamment petit) pour obtenir un noyau linéaire.

Comme nous l'avons annoncé, le théorème des méta-noyaux [8] utilise la notion de quasi-recouvrabilité. Intuitivement, un problème est quasi-recouvrable si l'ensemble solution (ou un ensemble défini à partir de la solution) recouvre un modulateur d'arborescence à distance radiale r. Cette notion utilise la distance radiale (la distance dans le graphe radial) qui nécessite de plonger le graphe. Si un problème est quasi-recouvrable, il est possible de construire une décomposition en protrusions pour toutes ses instances positives. Cette propriété joue à peu près le même rôle que les théorèmes qui suivent et qui sont plus génériques ; pour cette raison nous n'en donnons pas la définition formelle.

Nous énonçons maintenant un théorème démontré par Kim, Langer, Paul, Reidl, Rossmanith, Sau et Sikdar [43], que nous utiliserons dans chacune de nos applications pour construire la décomposition en protrusions. Étant donné un modulateur d'arborescence, ce théorème permet de construire une décomposition en protrusions ; qui est linéaire si le graphe exclut un mineur (fixé). Remarquons que, dans le théorème 4, nous nous restreignons déjà à une classe de graphes excluant un mineur.

Théorème 5 : [43]
Soit c, t deux entiers positifs. Soit H un graphe avec $|H| = h$ et G un

graphe sans mineur (respectivement, mineur topologique) H . Soit k un entier positif. Étant donné un modulateur $X \subseteq V(G)$ tel que $|X| \leqslant c \cdot k$ et $\mathrm{tw}(G - X) \leqslant t$, une $((\alpha_H \cdot t \cdot c) \cdot k, 2t + h)$-décomposition en protrusions peut être calculée en temps $O(|G|)$, avec $\alpha_H \leqslant 40h^2 2^{5h \log h}$ une constante ne dépendant que de H.

Remarquons que, tel que le théorème est formulé, il n'est pas nécessaire de distinguer les deux entiers c et k ; mais, dans les applications, nous utiliserons c comme une constante (dépendant du problème) et k comme le paramètre du problème.

Intuitivement, l'algorithme de décomposition en protrusions considère une décomposition arborescente du graphe privé de son modulateur, et il marque chacun des sacs en fonction du nombre de voisins que ce sac a dans le modulateur.

Pour obtenir notre décomposition, il nous suffit donc de trouver un modulateur d'arborescence. Pour ce faire nous utilisons les travaux de Fomin, Lokshtanov, Saurabh et Thilikos [27] sur la bidimensionnalité. Leur résultat permet de montrer l'existence d'un modulateur dans toutes instances positives d'un problème vérifiant deux propriétés assez génériques : la bidimensionnalité (par mineur ou par contraction) et la séparabilité linéaire. Nous définissons ces deux notions avant d'énoncer le théorème. Rappelons que nous décrivons le cadre de travail pour le cas des problèmes de minimisation. Dans nos applications nous n'utilisons le théorème 6 que sur des problèmes de minimisation.

Intuitivement, un problème est bidimensionnel si les instances contenant une grande grille (comme mineur) ou pseudo-grille (comme contraction) ont nécessairement un solution de grande taille.

Définition 69 : problème bidimensionnel [27]

Un problème Π est *bidimensionnel par mineur* si

— pour tout graphe G, $f^{\Pi}(G - v) \leqslant f^{\Pi}(G)$, pour $v \in V(G)$,
$f^{\Pi}(G \setminus e) \leqslant f^{\Pi}(G)$,
$f^{\Pi}(G/e) \leqslant f^{\Pi}(G)$, pour $e \in E(G)$;

— ^{et} pour la grille, $f^{\Pi}(\Gamma_r) \geqslant O(r^2)$.

Un problème est *bidimensionnel par contraction* si

— pour tout graphe G, $f^{\Pi}(G/e) \leqslant f^{\Pi}(G)$ pour $e \in E(G)$;

— ^{et} pour la pseudo-grille, $f^{\Pi}(\Gamma'_n) \geqslant O(n^2)$.

⌟

Le premier item impose que les opérations de mineur dans une instance font diminuer la taille de la solution. Le second item signifie que la solution optimale d'une grille a une taille (au moins) proportionnel au nombre de sommets.

Il faut signaler que cette définition reste identique pour les problème de minimisation et de maximisation.

Intuitivement, un problème est séparable si l'intersection d'une solution optimale avec un sous-graphe n'est pas trop éloignée d'une solution optimale dans le sous-graphe.

Définition 70 : problème séparable linéairement [27]

Un problème Π est *séparable linéairement* si pour tout graphe G et tout sous-graphe G_B de bord B, nous avons $|S \cap V(G_B)| \leqslant f^{\Pi}(G_B) + O(|B|)$; où $S \subseteq V(G)$ est une solution optimale dans G.

⌟

Remarquons que cette définition présente certaines similarités avec les notions de confinement (cf. définition 63) et de monotonicité [8]. Remarquons aussi que la séparabilité n'est définie que dans le cadre de problèmes certifiables par sommets, c'est-à-dire des problèmes dont la solution peut être décrite par un ensemble de sommets. Ce n'est pas un restriction contraignante car il est possible de décrire artificiellement les certificats de nombreux problèmes, comme $\mathcal{F}$-Paquetage, par un ensemble de sommets. De plus, dans les applications que nous présentons, nous n'appliquons le théorème 6 qu'à des problèmes naturellement certifiables par sommets.

Théorème 6 : [27]

Soit Π un problème, restreint à une classe de graphes excluant un mineur fixé (respectivement, un mineur apex), bidimensionnel par mineur (respectivement, par contraction) et séparable linéairement. Pour tout réel $c > 0$, il existe un entier t tel que dans toute instance positive $(G, k) \in \Pi$, G a un modulateur X avec $|X| \leqslant c \cdot k$ et $\mathrm{tw}(G - X) \leqslant t$.

⌞ ⌟

Sans entrer dans le détail, l'algorithme pour obtenir le modulateur consiste à construire une décomposition arborescente, trouver un sac de la décomposition de sorte que la solution soit balancée de part et d'autre de ce sac séparateur, et rechercher récursivement de tel sac dans les deux décompositions arborescentes obtenues.

Pour rendre l'algorithme de modulation constructif, nous avons besoin de construire une décomposition arborescente de l'instance. Il n'est pas possible

de construire un décomposition arborescente de largeur optimale en temps polynomial, mais nous pouvons nous contenter d'une approximation. Nous utilisons donc un algorithme d'approximation polynomial sur les graphes excluant un mineur ; cette approximation dérive d'une preuve de Demaine et Hajiaghayi [16]. A notre connaissance, il n'y a pas de formulation explicite de la constante d'approximation. Remarquons que toute amélioration de cette constante se répercuterait immédiatement sur la taille de nos noyaux.

Pour l'algorithme de modulation, nous avons également besoin d'une solution (approchée) du problème considéré. Pour toutes nos applications, il existe un schéma d'approximation FPT de nos problèmes restreints à une classe excluant un mineur ; ces approximations découlent du résultat de Fomin, Lokshtanov, Raman et Saurabh [26]. Nous pouvons arbitrairement choisir le ratio d'approximation.

L'influence des deux approximations sur la taille de nos noyaux disparaît dans les dominations asymptotiques $O(\cdot)$. Le problème considéré et la classe de graphes à laquelle il est restreint impactent le temps d'exécution mais pas le résultat de l'approximation ; ils n'affectent donc pas la taille de nos noyaux. Les corollaires 2 et 3 découlent du théorème précèdent.

La démonstration du théorème 6 se base sur la propriété que si un graphe a une grande largeur arborescente alors il contient nécessairement un grande grille (la grille Γ_r comme mineur, ou la pseudo-grille Γ'_r comme contraction). Par conséquent, les valeurs des constantes c et t du théorème 6 dépendent de cette relation.

Dans le cas de nos applications, la relation entre la largueur arborescente et la taille de la grille est linéaire. Afin d'être le plus explicite possible dans l'expression des tailles de nos noyaux, nous avons besoin de préciser cette relation linéaire.

Proposition 7 : [16, 42]

Il existe une fonction f_m telle que, étant donnés un graphe H et un entier positif r, pour tout graphe G excluant H comme mineur si $tw(G) \geqslant f_m(|H|)$ alors G contient Γ_r comme mineur, avec $f_m(h) \leqslant 2^{O(h^2 \log h)}$.

Proposition 8 : [25]

Il existe une fonction f_c telle que, étant donné un graphe apex H et un entier positif r, pour tout graphe G excluant H comme mineur si $tw(G) \geqslant f_c(|H|)$ alors G contient Γ'_r comme contraction.

Remarquons que Kawarabayashi et Kobayashi [42] ont fourni une borne de la fonction f_m, mais que, à notre connaissance, il n'existe pas de borne

similaire pour la fonction f_c. En cela nos résultats des section 3.3 et section 3.4 ne seront pas totalement explicite. Remarquons aussi que toutes nouvelles bornes sur ces fonctions se répercutera directement sur nos résultats.

3.2.2 Choix de la fonction d'optimisation

Nous décrivons ici, de façon informelle, des formulations génériques de la fonction d'optimisation $f^{\mathcal{E}}$. Le choix de la fonction $f^{\mathcal{E}}$ est une étape cruciale dans l'application de notre cadre, en effet c'est cette fonction qui remplit les tables de programmation dynamique, et donc qui choisit, au fur et à mesure de l'exécution de l'algorithme, quelle est l'information pertinente à conserver. D'une part, si nous conservons trop d'information, nous obtenons des tables trop grandes, autrement dit, notre encodeur n'est pas confiné. D'autre part, si nous ne conservons pas assez d'information, nous ne pouvons pas calculer la solution optimale ; autrement dit, notre encodeur n'est pas adapté à la programmation dynamique. Il faut donc conserver juste assez d'information.

Nous proposons trois types de fonctions d'optimisation $f^{\mathcal{E}}$, que nous appelons respectivement fonction naturelle, fonction tronquée, et fonction pertinente. Chacune des trois sections suivantes est un exemple d'application d'un des trois types de fonctions. Ces trois types nous permettent d'appliquer notre cadre, et sont suffisamment génériques pour être utilisés sur un grand nombre de problèmes ; cependant, ils n'excluent pas tout autre type de fonction.

Au lieu de faire de $f^{\mathcal{E}}$ un élément de l'encodeur, nous aurions pu imposer la forme de cette fonction, ce qui nous aurait épargné la tâche de la redéfinir à chaque application, mais nous aurions été contraints de restreindre notre cadre à une catégorie particulière de problèmes.

L'intuition la plus naturelle consiste à définir $f^{\mathcal{E}}$ comme la fonction qui minimise la mesure d'une solution partielle ; autrement dit, $f^{\mathcal{E}}$ (définie sur les graphes bordés) est une extension de la fonction d'optimisation du problème f^{Π} (définie sur les graphes) . La notion de mesure d'une solution partielle est plus ou moins obvie selon le problème : pour r-Domination, la solution étant un ensemble de sommets, la mesure est évidement le nombre de sommets ; pour $\mathcal{F}$-Paquetage, la solution est un ensemble de modèles, le mesure est donc le nombre de modèles, mais une solution partielle contient aussi des modèles potentiels (en cours de réalisation), pour lesquels il faut établir une convention (en d'autres termes, il convient de correctement définir m^{Π_R} à partir de m^{Π}).

Définition

Étant donné un générateur d'encodages $\mathcal{C}^{\mathcal{E}}$ et un langage $L^{\mathcal{E}}$, nous définissons la fonction *naturelle* $f^{\mathcal{E}} : \Gamma^* \times \Upsilon^* \to \mathbb{N} \cup \{+\infty\}$ telle que :

$$f^{\mathcal{E}}(G, R) = \min\{k \mid \exists S \in \Sigma^*, m^{\Pi_R}(S) \leqslant k, (G, S, R) \in L^{\mathcal{E}}\}.$$

Si un tel S n'existe pas, nous posons $f^{\mathcal{E}}(G, R) = +\infty$.

L'encodeur naturel est donc $\mathcal{E} = (\mathcal{C}^{\mathcal{E}}, L^{\mathcal{E}}, f^{\mathcal{E}})$.

⌟

Cependant, dans de nombreux cas, l'encodeur naturel dérivé à partir d'un programme dynamique *ad hoc* n'est pas confiné. Dans ce cas, nous pouvons essayer de forcer le confinement. Autrement dit, la fonction naturelle n'est utilisable que si le problème est naturellement confiné (remarquons que les problèmes naturellement confinés correspondent aux problèmes monotones [8]). Sinon, il nous faut trouver un moyen de supprimer de l'information. Pour cela, nous introduisons la fonction tronquée, que nous notons $f_g^{\mathcal{E}}$, pour une certaine fonction g. Cette fonction tronquée élimine (dans les tables) toutes les valeurs à une distance supérieure à $g(t)$ de l'optimale. Intuitivement, nous voulons que, à chaque étape de la programmation dynamique, le programme cesse de considérer des solutions partielles qui sont trop éloignées d'un optimal partiel (calculé à cette étape) pour pouvoir être complétées en une solution optimale (à la fin de l'exécution). La difficulté réside dans le choix d'une fonction g qui préserve suffisamment de valeurs dans les tables pour exécuter le programme dynamique (c'est-à-dire, telle que l'encodeur soit adapté, cf. définition 67).

Définition

Étant donné un encodeur (naturel) $\mathcal{E} = (\mathcal{C}^{\mathcal{E}}, L^{\mathcal{E}}, f^{\mathcal{E}})$, et une fonction $g : \mathbb{N} \to \mathbb{N}$, nous définissons la fonction *tronquée* $f_g^{\mathcal{E}} : \Gamma^* \times \Upsilon^* \to \mathbb{N} \cup \{+\infty\}$ telle que :

$$f_g^{\mathcal{E}}(G, R) = \begin{cases} +\infty, & \text{si } f^{\mathcal{E}}(G, R) - g(t) > \\ & \min\{f^{\mathcal{E}}(G, R) \mid R \in \mathcal{C}^{\mathcal{E}}(\Lambda(G))\}, \\ f^{\mathcal{E}}(G, R), & \text{sinon.} \end{cases}$$

L'encodeur tronqué est donc $\mathcal{E}_g = (\mathcal{C}^{\mathcal{E}}, L^{\mathcal{E}}, f_g^{\mathcal{E}})$.

⌟

Néanmoins, pour certains problèmes il n'existe pas de fonction g qui permette l'application de la programmation dynamique (c'est-à-dire, que l'encodeur tronqué n'est pas adapté). Pour cette raison, nous proposons la fonction pertinente, que nous notons $\bar{f}_g^{\mathcal{E}}$. Pour remplir la table d'un nœud (dans une décomposition arborescente), cette fonction prend en compte les tables des

fils. Plus précisément, cette fonction permet de transmettre à la racine l'information que, quelque part parmi les descendants, une valeur à été tronquée (c'est-à-dire, un solution partielle a été éliminée). Intuitivement, l'usage de cette fonction est justifié par le fait que, si une solution partielle S a été éliminée à un certain moment, il n'y a pas raison qu'une solution construite à partir de S, redevienne utile (même si sa mesure retourne dans l'intervalle de confinement) ; cela revient à dire que l'usage de la fonction pertinente conserve la propriété d'être un Π-encodeur.

Pour pouvoir définir la fonction pertinente, nous avons besoin qu'une solution induise un encodage lorsqu'elle traverse un séparateur. Bien évidemment, il est difficile de définir un protocole générique qui décrive comment construire un encodage à partir d'une solution, pour tous les problèmes. Nous ne donnerons donc pas une définition formelle de ce que « une solution induit un encodage » veut dire ici, et nous le définirons de façon *ad hoc* dans chaque application.

Définition

Étant donné $G \in \mathcal{B}_t$ un graphe de bord A, un encodage R_A sur A est *non pertinent* s'il existe un certificat S tel que $(G, S, R_A) \in L^{\mathcal{E}}$ et $|S| = f^{\mathcal{E}}(G, R_A)$, et il existe un séparateur $B \subset V(G)$ tel que $B \neq A$ et $|B| \leqslant t$ de sorte que S « induise » un encodage R_B sur B avec $\bar{f}_g^{\mathcal{E}}(G_B, R_B) = +\infty$.

⌟

Définition

Étant donné un encodeur (naturel) $\mathcal{E} = (\mathcal{C}^{\mathcal{E}}, L^{\mathcal{E}}, f^{\mathcal{E}})$, et une fonction $g : \mathbb{N} \to \mathbb{N}$, nous définissons la fonction *pertinente* $\bar{f}_g^{\mathcal{E}} : \Gamma^* \times \Upsilon^* \to \mathbb{N} \cup \{+\infty\}$ telle que :

$$\bar{f}_g^{\mathcal{E}}(G, R) = \begin{cases} +\infty, & \text{si } f^{\mathcal{E}}(G, R) - g(t) > \\ & \min\{f^{\mathcal{E}}(G, R) \mid R \in \mathcal{C}^{\mathcal{E}}(\Lambda(G))\}, \\ & \text{ou si } R \text{ est non pertinent,} \\ f^{\mathcal{E}}(G, R), & \text{sinon.} \end{cases}$$

L'encodeur pertinent est donc $\bar{\mathcal{E}}_g = (\mathcal{C}^{\mathcal{E}}, L^{\mathcal{E}}, \bar{f}_g^{\mathcal{E}})$.

⌟

Remarquons que, par définition, un encodeur tronqué $\mathcal{E}_g$ ou un encodeur pertinent $\bar{\mathcal{E}}_g$ est nécessairement g-confiné. Remarquons que dans tous les faits, lemmes et théorèmes de la section 3.1 précédente un encodeur $\mathcal{E}$ peut en particulier être un encodeur tronqué ou pertinent.

3.3 Application à r-Domination

Dans cette section nous construisons un noyau linéaire pour le problème r-DOMINATION dans une classe des graphes excluant un mineur apex fixé. Nous utilisons notre cadre de remplacement de protrusion via la programmation dynamique. Pour un entier $r \geqslant 1$ fixé, le problème r-DOMINATION consiste à dominer à distance r un graphe donné, c'est-à-dire à trouver un sous-ensemble de sommets tel que tout le graphe est à distance au plus r de la solution. Notons que r-DOMINATION est un problème de minimisation (l'ensemble de tous les sommets est une solution).

Définition 71 : r-dominant
Un *r-dominant* est un ensemble $D \subseteq V(G)$ tel que $N^r[D] = V(G)$. ⌟

r-DOMINATION :
instance : un graphe $\mathcal{G}$ et un entier k ;
paramètre: l'entier k ;
question : existe-t-il un r-dominant de taille au plus k ?

Lorsque $r = 1$ ce problème est équivalent à DOMINATION. Par conséquent r-DOMINATION est NP-complet, en particulier dans les graphes planaires ; du point de vue paramétré, il est W[2]-complet [19].

Nous construisons un noyau linéaire pour r-DOMINATION lorsqu'il est restreint à une classe de graphes excluant un mineur apex fixé H. La décomposition en protrusions est obtenue à partir du résultat de Fomin, Lokshtanov, Saurabh et Thilikos [27]. L'encodeur que nous utilisons pour obtenir ce noyau est inspiré de la programmation dynamique de Demaine, Fomin, Hajiaghayi et Thilikos [15] ; nous le notons $\mathcal{E}_{r\mathrm{D}}$.

Théorème
Soit $r \geqslant 1$ un entier. Soit H un graphe apex et $\mathcal{G}$ la classe des graphes excluant H comme mineur. Le problème r-DOMINATION, restreint à $\mathcal{G}$, admet un noyau linéaire constructif et explicite.
⌞ ⌟

Comme nous l'avons annoncé dans la section 3.2, la démonstration se fait en deux étapes. Dans la sous-section 3.3.1 nous construisons une décomposition en protrusions linéaire. Et dans la sous-section 3.3.2 nous proposons un encodeur qui satisfait les conditions pour que nous puissions appliquer le corollaire 1 dans la sous-section 3.3.3.

3.3.1 Décomposition en protrusions d'une instance

Nous construisons ici une décomposition en protrusions. Pour obtenir un modulateur d'arborescence, nous reformulons le théorème 6 dans le cas particulier de r-Domination (en soulignant l'influence de r et de H). Pour cela, il nous suffit de vérifier que r-Domination est bidimensionnel par contractions et séparable linéairement, ce qui est clair : la bidimensionnalité découle de la définition du problème et la séparabilité se démontre avec les arguments du lemme 22.

Pour conclure, nous utilisons le théorème 5 afin d'obtenir la décomposition.

Corollaire 2

Soit $r \geqslant 1$ un entier. Soit H un graphe apex avec $|H| = h$. Si (G, k) est une instance positive de r-Domination, avec G sans mineur H, alors il existe un modulateur $X \subseteq V(G)$ tel que $|X| = O(r f_c(h) \cdot k)$ et $\mathrm{tw}(G - X) = O(r f_c(h)^2)$.

De plus, étant donné une instance (G, k), il existe un algorithme polynomial qui, ou renvoie un tel modulateur X, ou répond que (G, k) est une instance négative.

Lemme 20

Soit $r \geqslant 1$ un entier. Soit H un graphe apex avec $|H| = h$. Soit (G, k) une instance de r-Domination avec G sans mineur H. Il existe un algorithme polynomial qui calcule une $(O(2^{O(h \log h)} r^2 f_c(h)^3) \cdot k, O(r f_c(h)^2))$-décomposition en protrusions ou répond que (G, k) est une instance négative.

Démonstration. Nous utilisons l'algorithme de la corollaire 2 sur l'instance (G, k). Celui-ci répond que (G, k) est négative ou renvoie un modulateur X de taille $|X| = c \cdot k = O(r f_c(h)) \cdot k$ avec $\mathrm{tw}(G - X) = t = O(r f_c(h)^2)$. Ensuite, dans le second cas, nous appliquons l'algorithme du théorème 5 sur X, qui renvoie une $((\alpha_H \cdot tc) \cdot k, 2t + h)$-décomposition, avec $\alpha_H = O(2^{O(h \log h)})$. □

3.3.2 Encodeur pour r-Domination

Nous rappelons que, intuitivement, un programme dynamique cherche à décrire de quelles façons une solution peut traverser un séparateur (dans une décomposition arborescente) ; au cours de l'exécution, cela revient à identifier quelles contraintes sont remplies par la solution partielle déjà construite et quelles sont celles qui restent et que le programme espère remplir dans la partie du graphe qui n'a pas encore été explorée.

Dans le cas de r-Domination, un sommet du séparateur peut avoir trois états : soit dominé par un sommet plus haut dans la décomposition arborescente (autrement dit, il n'est pas encore dominé), soit dominant (dans la solution), soit dominé par un sommet plus bas (autrement dit, il est déjà dominé) ; et ce, à une certaine distance de l'ensemble solution. Pour décrire cela, nous apposons sur chaque sommet du séparateur une étiquette composée d'un symbole $\uparrow$ ou $\downarrow$ (pour indiquer s'il est dominé par en haut ou par en bas) et d'un entier dans $[0, r]$ (pour indiquer sa distance au dominant). Chaque encodage correspond donc à un étiquetage de ∂G, et $\mathcal{C}^{\mathcal{E}_{r\mathrm{D}}}$ génère tous les étiquetages possibles d'un séparateur.

Comme nous l'avons dit, notre encodeur s'inspire de la programmation dynamique proposée par Demaine, Fomin, Hajiaghayi et Thilikos [15]. La particularité de cette programmation réside dans le fait que les contraintes sur le séparateur sont optionnelles au sens où un sommet étiqueté i doit être à distance au plus i du dominant et non à distance exactement i. D'une part, cette astuce évite que certains encodages soient insatisfiables ou satisfaits par une solution partielle trop grande (autrement dit, il n'y a plus de valeur $+\infty$ dans la table et nous n'avons pas besoin de tronquer), d'autre part, elle nous permet de facilement identifier quel encodage est associé à la valeur minimale dans la table. Ces deux points jouent un rôle important dans la preuve du confinement, lemme 22. L'idée qui sous-tend la notion informelle d'encodage optionnel est que certaines contraintes sont intuitivement plus favorables à une petit solution, en utilisant des contraintes optionnelles, nous évitons qu'une qu'une contrainte intuitivement bonne devienne mauvaise à cause d'une configuration particulière à un graphe. Cela permet donc d'ordonner les contraints et fournie donc un ordre partiel sur les encodages.

Le générateur $\mathcal{C}^{\mathcal{E}_{r\mathrm{D}}}$

Soit $G \in \mathcal{B}_t$ de bord ∂G labellé par $\Lambda(G)$. La fonction $\mathcal{C}^{\mathcal{E}_{r\mathrm{D}}}$ envoie $\Lambda(G)$ sur un ensemble $\mathcal{C}^{\mathcal{E}_{r\mathrm{D}}}(\Lambda(G))$ d'encodages. Chaque encodage $R \in \mathcal{C}^{\mathcal{E}_{r\mathrm{D}}}(\Lambda(G))$ fait correspondre $\Lambda(G)$ avec un $|\Lambda(G)|$-uplet de $\{0, \uparrow 1, \downarrow 1, \dots \uparrow r, \downarrow r\}^{|\Lambda(G)|}$. Il s'en suit que les éléments du $|\Lambda(G)|$-uplet sont en bijection avec les sommets de ∂G. Pour un sommet $v \in \partial G$ nous notons $R(v)$ l'élément du $|\Lambda(G)|$-uplet correspondant.

Le langage $L^{\mathcal{E}_{r\mathrm{D}}}$

Soit $G \in \mathcal{B}_t$ de bord ∂G. Pour un ensemble de sommets S nous posons que $(G, S, R) \in L^{\mathcal{E}_{r\mathrm{D}}}$ (que S est un r-dominant partiel satisfaisant R) si :

— pour tout sommet $v \in V(G) \setminus \partial G$, ou $d(v, S) \leqslant r$, ou il existe $w \in \partial G$ tel que $R(w) = \uparrow j$ avec $d(v, w) + j \leqslant r$;

— *et* pour tout sommet $v \in \partial G$ tel que $R(v) = 0$, $v \in S$;

— *et* pour tout sommet $v \in \partial G$ tel que $R(v) =\downarrow i$, ou $d(v, S) \leqslant i$, ou il existe $w \in \partial G$ tel que $R(w) =\uparrow j$ avec $d(v, w) + j \leqslant i$.

Remarquons que si S est un r-dominant partiel satisfaisant R, alors S contient l'ensemble $\{v \mid R(v) = 0\}$, mais S peut contenir d'autres sommets de ∂G.

La fonction $f^{\mathcal{E}_{r\mathrm{D}}}$

Soit $G \in \mathcal{B}_t$ de bord ∂G. Pour r-DOMINATION, nous utilisons la fonction d'optimisation naturelle. Nous définissons $f^{\mathcal{E}_{r\mathrm{D}}}$ telle que :

$$f^{\mathcal{E}_{r\mathrm{D}}}(G, R) = \min\{k \mid \exists S \subseteq V(G), |S| \leqslant k, (G, S, R) \in L^{\mathcal{E}_{r\mathrm{D}}}\}.$$

La taille $s_{\mathcal{E}_{r\mathrm{D}}}$

Par définition de $\mathcal{C}^{\mathcal{E}_{r\mathrm{D}}}$, la taille de $\mathcal{E}_{r\mathrm{D}}$ peut facilement être bornée :

$$s_{\mathcal{E}_{r\mathrm{D}}}(t) \leqslant (2r + 1)^t. \tag{3.1}$$

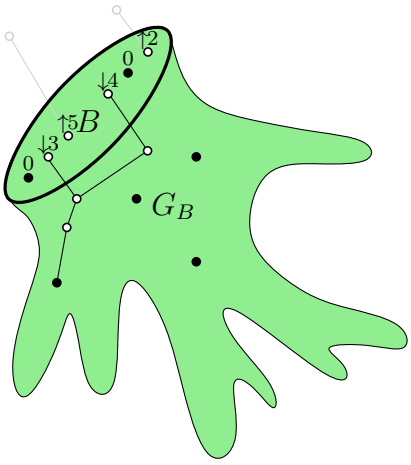


FIGURE 3.10 – Illustration de l'encodeur $\mathcal{E}_{r\mathrm{D}}$ pour r-DOMINATION définie dans cette sous-section 3.3.2.

Nous montrons maintenant que l'encodeur $\mathcal{E}_{r\mathrm{D}}$ défini ci-dessus vérifie toutes les conditions pour que notre cadre puisse être appliqué. Tout d'abord nous montrons que $\mathcal{E}_{r\mathrm{D}}$ est un r-DOMINATION-encodeur. Ensuite nous prouvons que $\mathcal{E}_{r\mathrm{D}}$ est g-confiné pour $g(t) = t$. Pour finir nous démontrons que l'équivalence $\cdot \sim^*_{\mathcal{E}_{r\mathrm{D}},\mathcal{G},t} \cdot$ est adaptée.

Lemme 21

L'encodeur $\mathcal{E}_{r\mathrm{D}} = (\mathcal{C}^{\mathcal{E}_{r\mathrm{D}}}, L^{\mathcal{E}_{r\mathrm{D}}} f^{\mathcal{E}_{r\mathrm{D}}})$ est un r-DOMINATION-encodeur.

Démonstration. Il y a un unique 0-uplet $R_\emptyset \in \mathcal{C}^{\mathcal{E}_{r\mathrm{D}}}(\emptyset)$, et $(G, S, R_\emptyset) \in L^{\mathcal{E}_{r\mathrm{D}}}$ si et seulement si S est un r-dominant de G, par définition de $L^{\mathcal{E}_{r\mathrm{D}}}$. De plus, $f^{\mathcal{E}_{r\mathrm{D}}}(G, R_\emptyset) = f^{\Pi}(G)$ par définition. □

Lemme 22

L'encodeur $\mathcal{E}_{r\mathrm{D}} = (\mathcal{C}^{\mathcal{E}_{r\mathrm{D}}}, L^{\mathcal{E}_{r\mathrm{D}}} f^{\mathcal{E}_{r\mathrm{D}}})$ est g-confiné pour $g : t \mapsto t$.

Démonstration. Soit G un graphe bordé. Soit $R \in \mathcal{C}^{\mathcal{E}_{r\mathrm{D}}}(\Lambda(G))$ un encodage quelconque sur ∂G, et soit R_0 l'encodage tel que $R_0(v) = 0$ pour tout $v \in \partial G$. Soit S_0 un r-dominant partiel satisfaisant R_0 et de taille minimale (c'est-à-dire, $(G, S_0, R_0) \in L^{\mathcal{E}_{r\mathrm{D}}}$ et $f^{\mathcal{E}_{r\mathrm{D}}}(G, R_0) = |S_0|$). Observons que S_0 satisfait également R. Par conséquent, $f^{\mathcal{E}_{r\mathrm{D}}}(G, R_0) = \max_R f^{\mathcal{E}_{r\mathrm{D}}}(G, R)$. Soit S un r-dominant partiel satisfaisant R et de taille minimale. Notons que $S \cup \partial G$ satisfait R_0. Donc, il s'en suit que $f^{\mathcal{E}_{r\mathrm{D}}}(G, R_0) - \min_R f^{\mathcal{E}_{r\mathrm{D}}}(G, R) \leqslant |\partial G| \leqslant t$, ce qui prouve le confinement. □

Lemme 23

L'équivalence $\cdot \sim^*_{\mathcal{E}_{r\mathrm{D}},\mathcal{G},t} \cdot$ associée à $\mathcal{E}_{r\mathrm{D}} = (\mathcal{C}^{\mathcal{E}_{r\mathrm{D}}}, L^{\mathcal{E}_{r\mathrm{D}}} f^{\mathcal{E}_{r\mathrm{D}}})$ est adaptée pour $\mathcal{G}$ une classe de graphe quelconque.

Démonstration. Pour rappel, nous avons posé dans la définition 66 que G est un graphe t-bordé de bord A, que B est un séparateur de G, que G_B est un sous-graphe t-bordé de G de bord B et que $G^- \oplus G_B = G$. Considérons G'_B tel que $G'_B \sim^*_{\mathcal{E}_{r\mathrm{D}},\mathcal{G},t} G_B$ et $G' = G^- \oplus G'_B$ (cf. figure 3.7).

Conformément à la définition 67, nous voulons montrer que $G \sim^*_{\mathcal{E}_{r\mathrm{D}},\mathcal{G},t} G'$ et que $\Delta_{\mathcal{E}_{r\mathrm{D}},t}(G, G') = \Delta_{\mathcal{E}_{r\mathrm{D}},t}(G_B, G'_B)$. D'après le fait 5 il nous suffit de montrer que $G \sim^*_{\mathcal{E}_{r\mathrm{D}},t} G'$ (sans tenir compte de la classe $\mathcal{G}$). Par conséquent, nous voulons prouver que $f^{\mathcal{E}_{r\mathrm{D}}}(G, R_A) = f^{\mathcal{E}_{r\mathrm{D}}}(G', R_A) + \Delta_{\mathcal{E}_{r\mathrm{D}},t}(G_B, G'_B)$ pour tout $R_A \in \mathcal{C}^{\mathcal{E}_{r\mathrm{D}}}(\Lambda(G))$.

Soit $R_A \in \mathcal{C}^{\mathcal{E}_{r\mathrm{D}}}(\Lambda(G))$.

Tout d'abord supposons que $f^{\mathcal{E}_{r\mathrm{D}}}(G, R_A) \neq +\infty$. Soit $S = D \cup D^-$ un r-dominant partiel satisfaisant R_A et de taille minimale (c'est-à-dire, $(G, S, R_A) \in L^{\mathcal{E}_{r\mathrm{D}}}$ et $f^{\mathcal{E}_{r\mathrm{D}}}(G, R_A) = |S|$) partitionné en $D \subseteq V(G_B)$ et $D^- \subseteq V(G^-) \setminus B$. Pour commencer, à partir de S, nous construisons un encodage $R_B \in \mathcal{C}^{\mathcal{E}_{r\mathrm{D}}}(\Lambda(G_B))$ sur B, tel que D satisfait R_B. Par la suite, nous montrons que l'encodage R_B contient suffisamment d'information, c'est-à-dire que un r-dominant partiel D' satisfaisant R_B dans G'_B peut être recollé

avec D^- pour former un r-dominant S' dans G'. A la fin, puisque G_B et G'_B sont équivalents, nous savons qu'ils ont les mêmes tables à décalage près ; ce décalage se propage sur les tables de G et G' (c'est-à-dire, les tailles de S et S' différent de la valeur du décalage).

Nous commençons par observer comment S traverse B. Nous posons R_B (satisfait par D) tel que, pour tout $v \in B$:

- si $v \in S$ alors $R_B(v) = 0$;
- sinon, si il existe un plus court chemin , dont la première arête est dans G_B, ou de v à S de longueur i ou de v à $w \in A$ avec $R_A(w) =\uparrow j$ de longueur $i - j$, alors $R_B(v) =\downarrow i$;
- sinon, $R(v) =\uparrow i$ avec $i = d_G(v, S)$ ou $i = d_G(v, w) + j$ avec $w \in A$ et $R_A(w) =\uparrow j$ (notons que tout plus court chemin de v à S ou à A a nécessairement sa première arête dans G^-).

Soit D' un r-dominant partiel de G'_B satisfaisant R_B et de taille minimale. Soit $S' = D' \cup D^-$.

Nous montrons maintenant que S' est bien un r-dominant partiel de G' satisfaisant R_A ; c'est-à-dire $(G', S', R_A) \in L^{\mathcal{E}_{r\mathrm{D}}}$. Conformément à la définition de $L^{\mathcal{E}_{r\mathrm{D}}}$ nous vérifions deux types de contraintes : celles qui s'appliquent sur les sommets dans $V(G') \setminus A$ et celles sur les sommets de A. Commençons par les sommets de $V(G') \setminus A$. Pour tout $v \in V(G') \setminus (A \cup S')$ nous construisons un chemin de v à S' longueur au plus r ou un chemin de v à $w \in A$ de longueur au plus $r - i$ avec $R_A(w) =\uparrow i$. Nous utilisons pour cela la procédure récursive qui suit. À l'étape j, la procédure identifie un sommet $v_j \in B$ (ainsi qu'un $v_{j-1}v_j$-chemin, partie du chemin cherché). Nous initialisions $v_0 = v$. Si $v_0 \in V(G'_B)$ alors nous pouvons supposer que $d_{G'_B}(v_0, D') > r$ (sinon le chemin cherché existe). Puisque D' satisfait R_B, cela signifie que B contient un sommet v_1 tel que $R_B(v_1) =\uparrow i_1$ et que $d_{G'_B}(v_0, v_1) + i_1 \leqslant r$. De même, si $v_0 \in V(G^-) \setminus B$ alors nous pouvons supposer que $d_{G^-}(v_0, D^-) > r$ et que $d_{G^-}(v_0, w) > r - i$ pour tout $w \in A$ avec $R_A(w) =\uparrow i$ (sinon le chemin cherché existe). Puisque $S = D^- \cup D$ est un r-dominant partiel satisfaisant R_A, cela signifie que tout chemin de longueur au plus r entre v_0 et S ainsi que tout chemin de longueur au plus $r - i$ avec $w \in A$ et $R_A(w) =\uparrow i$ passe par une arête dans G_B dont une extrémité est dans B. Soit v_1 le premier de ces sommets. Par définition de R_B nous avons $d_{G^-}(v_0, v_1) + i_1 \leqslant r$. Nous avons terminé l'initialisation, commençons la boucle. Considérons que la procédure a trouvé le sommet v_j (avec $j \geqslant 1$) et notons l_j la longueur du v_0v_j-chemin construit jusque là. Nous devons montrer que $l_j + i_j \leqslant r$ (ou respectivement, $l_j + i_j \leqslant r - i$ dans le cas où nous atteignons un sommet de A). Nous l'avons montré pour $j = 1$. Par récurrence, supposons cela vrai pour j quelconque, et montrons-le pour $j + 1$. Nous distinguons deux cas.

— $R_B(v_j) =\downarrow i_j$. Nous pouvons supposer que $d_{G'_B}(v_j, D') > i_j$; sinon la procédure termine puisque par construction $l_j + i_j \leqslant r$ (respectivement, $l_j + i_j \leqslant r - i$). Puisque D' est un r-dominant partiel satisfaisant R_B, il existe un sommet $v_{j+1} \in B$ avec $R_B(v_{j+1}) =\uparrow i_{j+1}$ et $d_{G'_B}(v_i, v_{j+1}) + i_{j+1} \leqslant i_j$. Comme $l_{j+1} = l_j + d_{G'_B}(v_i, v_{j+1})$, il s'en suit que $l_{j+1} + i_{j+1} \leqslant r$ (respectivement, $l_{j+1} + i_{j+1} \leqslant r - i$) (cf. figure 3.11(b)).

— $R_B(v_j) =\uparrow i_j$. Nous pouvons supposer que $d_{G^-}(v_j, D^-) > i_j$ (et respectivement, $d_{G^-}(v_j, w) > i_j - i$ pour tout $w \in A$ avec $R_A(w) =\uparrow i$) ; sinon la procédure termine puisque par construction $l_j + i_j \leqslant r$ (respectivement, $l_j + i_j \leqslant r - i$). Puisque par définition de R_B, $d_G(v_j, S) = i_j$, tout chemin de v_j à S (respectivement, à $w \in A$) de longueur au plus r passe par une arête dans G_B dont une extrémité est dans B. Soit v_{j+1} le premier de ses sommets. Nous avons $R_B(v_{j+1}) =\downarrow i_{j+1}$ avec $d_{G^-}(v_j, v_{j+1}) + i_{j+1} \leqslant i_j$. Comme $l_{j+1} = l_j + d_{G^-}(v_i, v_{j+1})$, il s'en suit que $l_{j+1} + i_{j+1} \leqslant r$ (respectivement, $l_j + i_j \leqslant r - i$).

Remarquons que la procédure termine puisque $r-(l_j+i_j)$ décroit strictement. Considérons maintenant les sommets de A. Remarquons que $A \subseteq V(G^-)$. Soit $v \in A$. Si $R(v) = 0$ alors $v \in D^-$ (car S satisfait R_A) et donc $v \in S'$. Si $R(v) =\downarrow i$, la procédure ci-dessus construit un chemin de v à S' de longueur au plus r ou un vw-chemin de longueur au plus $r - i - j$ avec $w \in A$ et $R_A(w) =\uparrow j$. Par conséquent, S' est un r-dominant partiel de taille au plus $f^{\mathcal{E}_{r\mathrm{D}}}(G, R_A) + \Delta_{\mathcal{E}_{r\mathrm{D}},t}(G_B, G'_B)$ satisfaisant R_A.

Il nous reste à montrer que la taille de S' est celle attendue (autrement dit, le décalage se propage bien). Par construction de R_B, $|D| \geqslant f^{\mathcal{E}_{r\mathrm{D}}}(G_B, R_B)$. Puisque $G'_B \sim^*_{\mathcal{E}_{r\mathrm{D}},t} G_B$, nous savons que $|D'| = f^{\mathcal{E}_{r\mathrm{D}}}(G'_B, R_B) = f^{\mathcal{E}_{r\mathrm{D}}}(G_B, R_B)$ $+\Delta_{\mathcal{E}_{r\mathrm{D}},t}(G_B, G'_B)$ et donc $|S' = D' \cup D^-| = f^{\mathcal{E}_{r\mathrm{D}}}(G_B, R_B) + \Delta_{\mathcal{E}_{r\mathrm{D}},t}(G_B, G'_B) +$ $|D^-| \leqslant f^{\mathcal{E}_{r\mathrm{D}}}(G, R_A) + \Delta_{\mathcal{E}_{r\mathrm{D}},t}(G_B, G'_B)$.

Enfin supposons que $f^{\mathcal{E}_{r\mathrm{D}}}(G, R_A) = +\infty$. Il s'en suit que $f^{\mathcal{E}_{r\mathrm{D}}}(G', R_A) =$ $+\infty$ aussi. En effet, supposons que ce ne soit pas le cas. Il existe alors un r-dominant partiel de G' satisfaisant R_A, et d'après l'argumentation qui précède nous pouvons construire un r-dominant partiel de G ; contradiction avec $f^{\mathcal{E}_{r\mathrm{D}}}(G, R_A) = +\infty$.

En conséquence de quoi, nous pouvons conclure que $G \sim^*_{\mathcal{E}_{r\mathrm{D}},t} G'$ et donc que la relation d'équivalence est adaptée.

□

3.3.3 Noyau pour r-Domination

Nous pouvons maintenant énoncer le noyau linéaire pour r-DOMINATION.

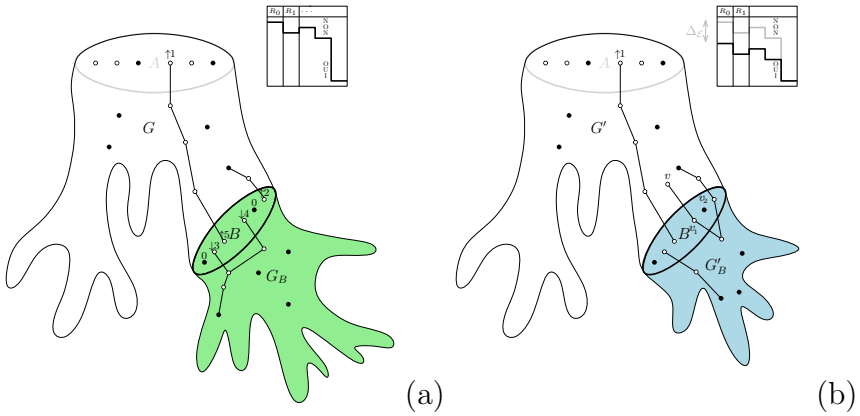


FIGURE 3.11 – Illustration du remplacement de protrusion et de la reconstruction d'un r-dominant dans le lemme 23. (a) La protrusion initiale G. (b) La protrusion après remplacement G'. Par exemple, le sommet v est a distance au plus 5 du r-dominant puisque nous pouvons construire un vd-chemin passant par v_1 et v_2 et composé de trois sous-chemins.

Théorème 7

Soit $r \geqslant 1$ un entier. Soit H un graphe apex avec $|H| = h$ et $\mathcal{G}$ la classe des graphes excluant H comme mineur. Le problème r-DOMINATION, restreint à $\mathcal{G}$, admet une noyau linéaire de taille $O(b(\mathcal{E}_{r\mathrm{D}}, g, O(rf_c(h)^2), \mathcal{G}) \cdot 2^{O(h \log h)} r^2 f_c(h)^3) \cdot k$.

Démonstration. Soit (G, k) une instance de r-DOMINATION avec G sans mineur H. D'après le lemme 20, ou (G, k) est une instance négative, ou nous pouvons construire une $(\alpha_H tc \cdot k, 2t+h)$-décomposition en protrusions. Dans le premier cas, nous renvoyons une instance trivialement négative (par exemple, $(G = (\{v\}, \emptyset), k = 0)$). Dans le second cas, nous considérons l'encodeur $\mathcal{E}_{r\mathrm{D}}$ défini dans la sous-section 3.3.2. D'après les lemmes 21, 22, et 23 $\mathcal{E}_{r\mathrm{D}}$ est un r-DOMINATION-encodeur g-confiné (pour $g : t \mapsto t$) et l'équivalence $\cdot \sim^*_{\mathcal{E}_{r\mathrm{D}}, \mathcal{G}, t} \cdot$ est adaptée. Nous savons que $s_{\mathcal{E}_{r\mathrm{D}}}(t) \leqslant (2r+1)^t$. Par conséquent, nous pouvons appliquer le corollaire 1 qui construit un noyau linéaire pour r-DOMINATION de taille :

$$\alpha_H tc \cdot (b(\mathcal{E}_{r\mathrm{D}}, g, t, \mathcal{G}) + 1) \cdot k,$$

avec

- $b(\mathcal{E}, g, t, \mathcal{G})$ la borne sur la taille des représentants (lemme 18) ;
- $t = rf_c(h)^2$ la borne sur la largeur arborescente (corollaire 2) ;

— $c = r f_c(h)$ la constante du modulateur (corollaire 2) ;
— $\alpha_H = O(2^{O(h \log h)})$ la constante de la décomposition (théorème 5) ;
— $f_c(h)$ la fonction liant largeur et pseudo-grille (proposition 8).

□

3.4 Application à r-Indépendance

Dans cette section nous utilisons notre cadre pour obtenir un noyau linéaire pour le problème r-Indépendance dans une classe des graphes excluant un mineur apex fixé. Étant donné un graphe, le problème r-Indépendance consiste en la sélection de sommets indépendants à distance r, c'est-à-dire des sommets à distance au moins $r+1$ (séparés par au moins $r+1$ arêtes, si le graphe est connexe) les uns des autres. Notons que r-Indépendance est un problème de maximisation (l'ensemble vide de sommets est une solution).

Définition 72 : r-indépendant
Un *r-indépendant* est un ensemble $I \subseteq V(G)$ tel que pour toute paire $v, w \in I$, $d(v, w) > r$.

⌟

r-Indépendance :
instance : un graphe G et un entier k ;
paramètre: l'entier k ;
question : existe-t-il un r-indépendant de taille au moins k ?

Lorsque $r = 1$ ce problème est équivalent à Indépendance. Par conséquent r-Indépendance est NP-complet, en particulier dans les graphes planaires ; du point de vue paramétré, il est W[1]-complet [19].

Nous construisons un noyau linéaire pour r-Indépendance lorsqu'il est restreint à une classe de graphes excluant un mineur apex fixé H. La décomposition en protrusions est obtenue à partir de celle pour r-Domination. L'encodeur est défini en nous basant sur une programmation dynamique classique ; nous le notons $\mathcal{E}_{r\mathrm{I}}$. Afin de montrer qu'il est adapté, nous nous inspirons de la preuve de Bodlaender, Fomin, Lokshtanov, Penninkx, Saurabh et Thilikos [8] qui prouve que ce problème a un index entier fini.

Théorème
Soit $r \geqslant 1$ un entier. Soit H un graphe apex et $\mathcal{G}$ la classe des graphes excluant H comme mineur. Le problème r-Indépendance, restreint à $\mathcal{G}$, admet un noyau linéaire constructif et explicite.
⌞ ⌟

Dans la sous-section 3.4.1 nous construisons une décomposition en protrusions linéaire. Et dans la sous-section 3.4.2 nous proposons un encodeur qui satisfait les conditions pour que nous puissions appliquer le corollaire 1 dans la sous-section 3.4.3.

3.4.1 Décomposition en protrusions d'une instance

Nous construisons ici la décomposition en protrusions. Pour cela, nous utilisons simplement la propriété (de type Erdős-Pósa) de dualité entre r-Indépendance et r-Domination [8]. Cela nous permet d'utiliser de nouveau le corollaire 2 pour obtenir un modulateur.

Lemme 24 : [8]
Si (G, k) est une instance négative de r-Indépendance, alors (G, k) est une instance positive de r-Domination.

Démonstration. À partir de (G, k) une instance négative de r-Indépendance, nous construisons D un r-dominant de G de taille k. Pour $i \in [1, k]$ nous ajoutons à D le sommet v_i arbitrairement choisi dans $G - \bigcup_{j<i} N^r(v_j)$. Puisque il n'existe pas de r-indépendant de taille supérieure à k, le graphe $G - \bigcup_{j<k} N^r(v_j)$ est vide, donc D est un r-dominant. □

Lemme 25
Soit $r \geqslant 1$ un entier. Soit H un graphe apex avec $|H| = h$. Soit (G, k) une instance de r-Indépendance avec G sans mineur H. Il existe un algorithme polynomial qui calcule une $(O(2^{O(h \log h)} r^2 f_c(h)^3) \cdot k, O(r f_c(h)^2))$-décomposition en protrusions ou répond que (G, k) est une instance positive.

Démonstration. D'après le lemme 20 il existe un algorithme qui calcule une $((\alpha_H \cdot tc) \cdot k, 2t + h)$-décomposition de G, avec $\alpha_H = O(2^{O(h \log h)})$, $t = O(r f_c(h)^2)$,et $c = O(r f_c(h))$, ou répond que (G, k) est une instance négative de r-Domination, c'est-à-dire une instance positive de r-Indépendance d'après le lemme 24. □

3.4.2 Encodeur pour r-Indépendance

Nous rappelons que, intuitivement, un encodage cherche à décrire comment une solution peut traverser un séparateur. Pour r-Indépendance, nous apposons sur chaque sommet du séparateur une étiquette composée d'un vecteur dont la première valeur indique la distance (dans le graphe déjà parcouru) du sommet à la solution, les autres valeurs indiquent la distance aux autres sommets du séparateur.

Il peut paraître inutile de mémoriser les distances entre les sommets du séparateur puisque cette information est contenue dans le graphe (considéré

par la programmation), et qu'elle ne dépend pas de la forme de la solution. Cependant, nous en avons besoin dans notre cadre. En effet, lorsque nous comparons deux graphes pour effectuer un remplacement, nous ne comparons que les tables ; si nous voulons contraindre deux graphes équivalents à avoir les mêmes distances entre sommets du séparateur, il nous faut ajouter cette contrainte dans l'encodage. Il résulte de ce choix d'encodeur que (pour un graphe donné) de nombreux encodages seront inutiles, au sens où il n'existera pas de solutions partielles les satisfaisant (en d'autre termes, la valeur $-\infty$ leur sera affectée).

Le générateur $\mathcal{C}^{\mathcal{E}_{r\mathrm{I}}}$

Soit $G \in \mathcal{B}_t$ de bord ∂G labellé par $\Lambda(G)$. La fonction $\mathcal{C}^{\mathcal{E}_{r\mathrm{I}}}$ envoie $\Lambda(G)$ sur un ensemble $\mathcal{C}^{\mathcal{E}_{r\mathrm{I}}}(\Lambda(G))$ d'encodages. Chaque encodage $R \in \mathcal{C}^{\mathcal{E}_{r\mathrm{I}}}(\Lambda(G))$ fait correspondre $\Lambda(G)$ avec un $|\Lambda(G)|$-uplet. Les éléments de ce $|\Lambda(G)|$-uplet sont en bijection avec les sommets de ∂G. Encore une fois, nous notons $R(v)$ l'élément du $|\Lambda(G)|$-uplet correspondant à v. Chaque élément $R(v)$ est un $(|\Lambda(G)|+1)$-uplet $(d_S, d_{v_1}, \dots d_{v_{|\Lambda(G)|}}) \in \{0, \dots r+1\}^{|\Lambda(G)|+1}$.

Le langage $L^{\mathcal{E}_{r\mathrm{I}}}$

Soit $G \in \mathcal{B}_t$ de bord ∂G. Pour un ensemble de sommets S nous posons que $(G, S, R) \in L^{\mathcal{E}_{r\mathrm{I}}}$ (que S est un r-indépendant partiel satisfaisant R) si :

- pour toute paire de sommets $v, w \in S$, $d_G(v, w) > r$;
- pour tout sommet $v \in \partial G$, $d_G(v, S) > d_S$ et pour tout $w \in \partial G$, $d_G(v, w) > d_{\lambda(w)}$.

Remarquons qu'une solution partielle est également un r-indépendant.

La fonction $f_g^{\mathcal{E}_{r\mathrm{I}}}$

Soit $G \in \mathcal{B}_t$ de bord ∂G. Pour r-Indépendance, nous utilisons la fonction d'optimisation tronquée par $g : t \mapsto 2t$. Nous définissons $f_g^{\mathcal{E}_{r\mathrm{I}}}$ telle que :

$$f_g^{\mathcal{E}_{r\mathrm{I}}}(G, R) = \begin{cases} -\infty, & \text{si } f^{\mathcal{E}_{r\mathrm{I}}}(G, R) + g(t) > \\ & \max\{f^{\mathcal{E}_{r\mathrm{I}}}(G, R) \mid R \in \mathcal{C}^{\mathcal{E}_{r\mathrm{I}}}(\Lambda(G))\}, \\ f^{\mathcal{E}_{r\mathrm{I}}}(G, R), & \text{sinon;} \end{cases}$$

avec $f^{\mathcal{E}_{r\mathrm{I}}}(G, R) = \max\{k \mid \exists S \subseteq V(G), |S| \leqslant k, (G, S, R) \in L^{\mathcal{E}_{r\mathrm{I}}}\}$.

La taille $s_{\mathcal{E}_{r\mathrm{I}}}$

Par définition de $\mathcal{C}^{\mathcal{E}_{r\mathrm{I}}}$, la taille de $\mathcal{E}_{r\mathrm{I}}$ peut facilement être bornée :

$$s_{\mathcal{E}_{r\mathrm{I}}}(t) \leqslant (2r+1)^{t(t+1)}.$$

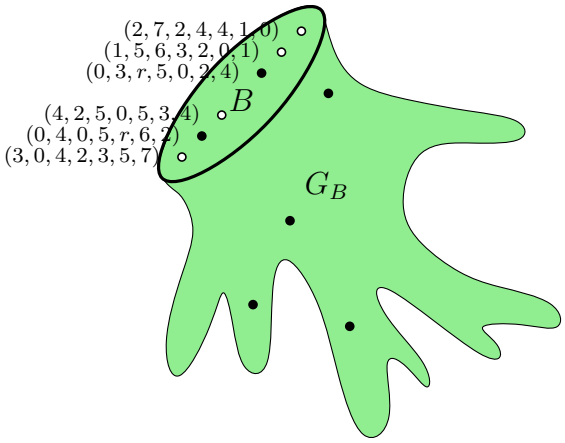


FIGURE 3.12 – Illustration de l'encodeur $\mathcal{E}_{r\text{I}}$ pour r-INDÉPENDANCE définie dans cette sous-section 3.4.2.

Nous montrons maintenant que l'encodeur $\mathcal{E}_{r\text{I}}$ défini ci-dessus vérifie toutes les conditions pour que notre cadre puisse être appliqué. Tout d'abord nous montrons que $\mathcal{E}_{r\text{I}}$ est un r-INDÉPENDANCE-encodeur. Ensuite nous prouvons que $\mathcal{E}_{r\text{I}}$ est g-confiné pour $g(t) = 2t$. Pour finir nous démontrons que l'équivalence $\cdot \sim^*_{\mathcal{E}_{r\text{I}},\mathcal{G},t} \cdot$ est adaptée.

Lemme 26

L'encodeur $\mathcal{E}_{r\text{I}} = (\mathcal{C}^{\mathcal{E}_{r\text{I}}}, L^{\mathcal{E}_{r\text{I}}} f_g^{\mathcal{E}_{r\text{I}}})$ est un r-INDÉPENDANCE-encodeur.

Démonstration. Il y a un unique 0-uplet $R_\emptyset \in \mathcal{C}^{\mathcal{E}_{r\text{I}}}(\emptyset)$, et $(G, S, R_\emptyset) \in L^{\mathcal{E}_{r\text{I}}}$ si et seulement si S est un r-indépendant de G, par définition de $L^{\mathcal{E}_{r\text{I}}}$. De plus, $f_g^{\mathcal{E}_{r\text{I}}}(G, R_\emptyset) = f^{\Pi}(G)$ par définition. □

Lemme 27

L'encodeur $\mathcal{E}_{r\text{I}} = (\mathcal{C}^{\mathcal{E}_{r\text{I}}}, L^{\mathcal{E}_{r\text{I}}} f_g^{\mathcal{E}_{r\text{I}}})$ est g-confiné pour $g : t \mapsto 2t$.

Démonstration. Par définition de $f_g^{\mathcal{E}_{r\text{I}}}$, $\mathcal{E}_{r\text{I}}$ est g-confiné pour $g(t) = 2t$. □

Lemme 28

L'équivalence $\cdot \sim^*_{\mathcal{E}_{r\text{I}},\mathcal{G},t} \cdot$ associée à $\mathcal{E}_{r\text{I}} = (\mathcal{C}^{\mathcal{E}_{r\text{I}}}, L^{\mathcal{E}_{r\text{I}}} f_g^{\mathcal{E}_{r\text{I}}})$ est adaptée pour $\mathcal{G}$ une classe de graphe quelconque.

Démonstration. Nous rappelons que nous avons posé G, G' de bord A et G^-, G_B, G'_B de bord B dans la définition 66 (cf. figure 3.7).

Conformément à la définition 67, nous voulons montrer que $G \sim^*_{\mathcal{E}_{r\mathrm{I}},\mathcal{G},t} G'$ et que $\Delta_{\mathcal{E}_{r\mathrm{I}},t}(G,G') = \Delta_{\mathcal{E}_{r\mathrm{I}},t}(G_B,G'_B)$. D'après le fait 5 il nous suffit de montrer que $G \sim^*_{\mathcal{E}_{r\mathrm{I}},t} G'$ (sans tenir compte de la classe $\mathcal{G}$). Par conséquent, nous voulons prouver que $f_g^{\mathcal{E}_{r\mathrm{I}}}(G,R_A) = f_g^{\mathcal{E}_{r\mathrm{I}}}(G',R_A) + \Delta_{\mathcal{E}_{r\mathrm{I}},t}(G_B,G'_B)$ pour tout $R_A \in \mathcal{C}^{\mathcal{E}_{r\mathrm{I}}}(\Lambda(G))$.

Soit $R_A \in \mathcal{C}^{\mathcal{E}_{r\mathrm{I}}}(\Lambda(G))$.

Tout d'abord supposons que $f_g^{\mathcal{E}_{r\mathrm{I}}}(G,R_A) \neq -\infty$. Soit $S = I \cup I^-$ un r-indépendant partiel satisfaisant R_A et de taille maximale (c'est-à-dire, $(G,S,R_A) \in L^{\mathcal{E}_{r\mathrm{I}}}$ et $f^{\mathcal{E}_{r\mathrm{I}}}(G,R_A) = |S|$) partitionné en $I \subseteq V(G_B)$ et $I^- \subseteq V(G^-)\setminus B$. Comme dans la preuve du lemme 23, à partir de S, nous construisons un encodage $R_B \in \mathcal{C}^{\mathcal{E}_{r\mathrm{I}}}(\Lambda(G_B))$ sur B, tel que I satisfait R_B. Contrairement au lemme 23, nous devons ajouter une argumentation pour démontrer que la valeur de R_B vérifie $f_g^{\mathcal{E}_{r\mathrm{I}}}(G_B,R_B) \neq -\infty$, en effet, il faut nous assurer que, si I a pu être étendu en S (et que S est possiblement utile) alors la valeur de I n'a pas été tronquée (autrement dit, nous n'avons pas besoin de forcer la propagation des valeurs tronquées). Par rapport à l'application précédente, cette étape est nouvelle, mais les arguments qu'elle contient correspondent peu ou prou à ceux utilisés dans le lemme 22 pour montrer que l'encodeur pour r-Domination était confiné. Par la suite, nous montrons que un r-indépendant partiel I' satisfaisant R_B dans G'_B peut être recollé avec D^- pour former un r-indépendant S' dans G'. A la fin, nous démontrons que le décalage entre G_B et G'_B se propage sur G et G'.

Nous commençons par observer comment S traverse B. Nous posons R_B (satisfait par I) tel que pour tout $v \in B$, $R_B(v) = (d_S, d_1, \dots d_{|B|})$ avec :

— $d_S = d_{G_B}(v,I)$;

— ^{et} pour tout $w \in \partial G_B$, $d_w = \min\{d_{G_B}(v,w), r+1\}$.

Nous montrons maintenant que $f_g^{\mathcal{E}_{r\mathrm{I}}}(G_B,R_B) \neq -\infty$ pour $g : t \mapsto 2t$. Considérons d'abord R_0 tel que $R_0(v) = (0,0,\dots 0)$ pour tout $v \in B$. Observons que tout r-indépendant satisfaisant un $R \in \mathcal{C}^{\mathcal{E}_{r\mathrm{I}}}(\Lambda(G_B))$ quelconque satisfait aussi R_0. Par conséquent, $f^{\mathcal{E}_{r\mathrm{I}}}(G_B,R_0) = \max_R f^{\mathcal{E}_{r\mathrm{I}}}(G_B,R)$. Soit I^0 satisfaisant R_0 et de taille maximale. Nous posons $I_* = I \setminus N^{\frac{r}{2}}(B)$, $I^0_* = I^0 \setminus N^{\frac{r}{2}}(B)$, et $I^-_* = I^- \setminus N^{\frac{r}{2}}(B)$. Clairement, $|I^0_*| \geqslant |I^0| + t$ (sans quoi B contiendrait un sommet à distance au plus $r/2$ de deux sommets distincts de I^0). De même $|I^-_*| \geqslant |I^-| + t$. Remarquons que $I^0_* \cup I^-_*$ est un r-indépendant de G satisfaisant R_A et donc nous avons $|I^0_* \cup I^-_*| \leqslant |S|$ (car S est maximal). Puisque $S = I \cup I^-$ nous avons aussi $|S| \leqslant |I \cup I^-_*| + t$. D'après ces deux inégalités nous obtenons que $|I^0_*| \leqslant |I| + t$ et donc que $|I^0| \leqslant |I| + 2t$. Il s'en suit que $f_g^{\mathcal{E}_{r\mathrm{I}}}(G_B,R_B) = f^{\mathcal{E}_{r\mathrm{I}}}(G_B,R_B)$, autrement dit, la valeur de R_B n'a pas été tronquée.

Soit I' un r-indépendant de G'_B satisfaisant R_B de taille maximale. Soit $S' = I' \cup I^-$.

Nous montrons maintenant que S' est bien un r-indépendant partiel de G' satisfaisant R_A. Conformément à la définition de $L^{\mathcal{E}_{r\text{I}}}$, nous vérifions deux types de contraintes : celles qui s'appliquent sur les sommets dans S' et celles sur les sommets de A. Commençons par les sommets de S'. Soit p un plus court vw-chemin dans G' avec $v, w \in S'$. Nous partageons p en sous-chemins maximaux $p_1, \dots p_q$ tels que p_j pour $j \in [1, q]$ est, ou bien un chemin dans G'_B, ou bien un chemin dans G^-. Si $q = 1$ alors $d_{G'}(v, w) > r$ car I' et I^- sont des r-indépendants (un r-indépendant partiel est en particulier un r-indépendant). Supposons que $q > 1$. Observons que tous les sous-chemins dans G^- sont aussi des chemins dans G et que tous les sous-chemins dans G'_B ont une longueur supérieure ou égale à la distance entre ses extrémités dans G_B (par définition de R_B). Nous considérons trois cas.

- Supposons $v, w \in V(G^-) \setminus B$. D'après l'observation ci-dessus, la longueur de p est au moins $d_G(v, w) > r$.
- Supposons $v \in V(G^-) \setminus B$ et $w \in V(G'_B)$. Soit u le premier sommet de p_q. Pour les mêmes raisons que dans le premier item, nous avons $d_{G'}(v, u) \geqslant d_G(v, u)$. Ensuite, par construction de I' nous avons $d_{G'_B}(u, w) \geqslant d_{G_B}(u, I)$. Donc p a une longueur supérieure ou égale à la distance entre v et I, c'est-à-dire $d_{G'}(v, w) > r$.
- Supposons $v, w \in V(G'_B)$. Soit u_1 et u_q le dernier sommet de p_1 et le premier sommet p_q, respectivement. Comme dans le premier item, nous avons $d_{G'}(u_1, u_q) \geqslant d_G(u_1, u_q)$. Ensuite, comme dans le second item, par construction de S', nous avons $d_{G'_B}(v, u_1) \geqslant d_{G_B}(I, u_1)$ et $d_{G'_B}(u_q, w) \geqslant d_{G_B}(u_q, I)$. Donc p a une longueur supérieure ou égale à la distance entre deux sommets dans I, donc $d_{G'}(v, w) > r$.

Considérons maintenant les sommets dans A. Remarquons que $A \subseteq V(G^-)$. Soit $v \in A$ avec $R(v) = (d_S, d_{v_1}, \dots d_{v_{|A|}})$. Considérons un plus court vw-chemin avec $w \in S'$; comme dans l'argumentation précédente (les deux premiers items) $d_{G'}(v, w) > d_S$. Considérons maintenant un plus court vv_i-chemin avec $v_i \in A$; comme dans l'argumentation précédente (le premier item) $d_{G'}(v, v_i) > d_{v_i}$.

Il nous reste à montrer que S' a la taille attendue. Par construction de R_B, $|I| \leqslant f_g^{\mathcal{E}_{r\text{I}}}(G_B, R_B)$. Puisque $G'_B \sim^*_{\mathcal{E}_{r\text{I},t}} G_B$, nous savons que $|I'| = f_g^{\mathcal{E}_{r\text{I}}}(G_B, R_B) + \Delta_{\mathcal{E}_{r\text{I},t}}(G_B, G'_B)$ et donc $|S' = I' \cup I^-| = f_g^{\mathcal{E}_{r\text{I}}}(G_B, R_B) + \Delta_{\mathcal{E}_{r\text{I},t}}(G_B, G'_B) + |I^-| \geqslant f_g^{\mathcal{E}_{r\text{I}}}(G, R_A) + \Delta_{\mathcal{E}_{r\text{I},t}}(G_B, G'_B)$.

Enfin supposons que $f_g^{\mathcal{E}_{r\text{I}}}(G, R_A) = -\infty$. Il s'en suit que $f_g^{\mathcal{E}_{r\text{I}}}(G', R_A) = -\infty$ aussi. En effet, supposons que ce ne soit pas le cas. Il existe alors un r-indépendant partiel de G' satisfaisant R_A, et d'après l'argumentation qui

précède nous pouvons construire un r-indépendant partiel de G ; contradiction avec $f_g^{\mathcal{E}_{r\mathrm{I}}}(G, R_A) = -\infty$.

En conséquence de quoi, nous pouvons conclure que $G \sim^*_{\mathcal{E}_{r\mathrm{I}},t} G'$ et donc que la relation d'équivalence est adaptée.

□

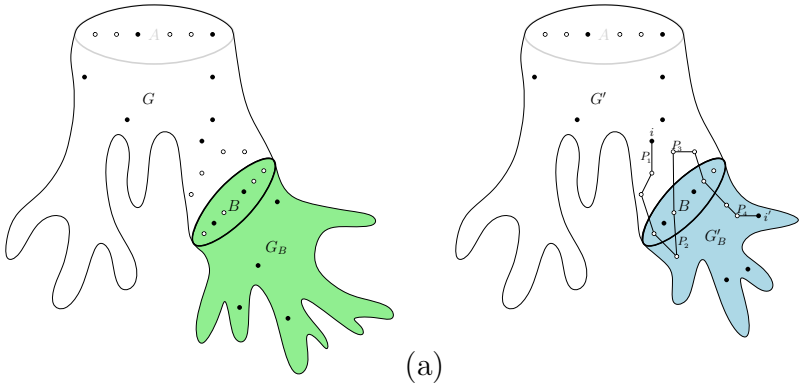


FIGURE 3.13 – Illustration du remplacement de protrusion et de la reconstruction d'un r-indépendant dans le lemme 28. (a) La protrusion initiale G. (b) La protrusion après remplacement G'. Par exemple, les sommets i, i' sont à distance au moins 11 l'un de l'autre puisque nous pouvons construire un ii'-chemin composé des quatre sous-chemins P_1, P_2, P_3, P_4.

3.4.3 Noyau pour r-Indépendance

Nous pouvons maintenant énoncer le noyau linéaire pour r-INDÉPENDANCE.

Théorème 8

Soit $r \geqslant 1$ un entier. Soit H un graphe apex avec $|H| = h$ et $\mathcal{G}$ la classe des graphes excluant H comme mineur. Le problème r-INDÉPENDANCE, restreint à $\mathcal{G}$, admet une noyau linéaire de taille $O(b(\mathcal{E}_{r\mathrm{I}}, g, O(r f_c(h)^2), \mathcal{G}) \cdot 2^{O(h \log h)} r^2 f_c(h)^3) \cdot k$.

Démonstration. Soit (G, k) une instance de r-INDÉPENDANCE avec G sans mineur H. D'après le lemme 25, ou (G, k) est une instance positive, ou nous pouvons construire une $(\alpha_H tc \cdot k, 2t+h)$-décomposition en protrusions. Dans le premier cas, nous renvoyons une instance trivialement positive (par exemple, $(G = (\{v\}, \emptyset), k = 1)$). Dans le second cas nous considérons l'encodeur $\mathcal{E}_{r\mathrm{I}}$ défini dans la sous-section 3.4.2. D'après les lemmes 26, 27, et 28 $\mathcal{E}_{r\mathrm{I}}$ est

un r-INDÉPENDANCE-encodeur g-confiné (pour $g : t \mapsto 2t$) et l'équivalence $\cdot \sim^*_{\mathcal{E}_{r\mathrm{I}},\mathcal{G},t} \cdot$ est adaptée. Nous savons que $s_{\mathcal{E}_{r\mathrm{I}}}(t) \leqslant (2r+1)^{t(t+1)}$. Par conséquent, nous pouvons appliquer le corollaire 1 qui construit un noyau linéaire pour r-INDÉPENDANCE de taille :

$$\alpha_H tc \cdot (b(\mathcal{E}_{r\mathrm{I}}, g, t, \mathcal{G}) + 1) \cdot k,$$

avec

- $b(\mathcal{E}, g, t, \mathcal{G})$ la borne sur la taille des représentants (lemme 18) ;
- $t = r f_c(h)^2$ la borne sur la largeur arborescente (corollaire 2) ;
- $c = r f_c(h)$ la constante du modulateur (corollaire 2) ;
- $\alpha_H = O(2^{O(h \log h)})$ la constante de la décomposition (théorème 5) ;
- $f_c(h)$ la fonction liant largeur et pseudo-grille (proposition 8).

□

3.5 Application à $\mathcal{F}$-Paquetage

Dans cette section nous utilisons notre cadre pour obtenir un noyau linéaire pour le problème $\mathcal{F}$-Paquetage dans une classe de graphes excluant un mineur fixé. Étant donné un graphe, le problème $\mathcal{F}$-Paquetage consiste à trouver un ensemble de modèles disjoints de $\mathcal{F}$, c'est-à-dire à trouver un ensemble de sous-graphes de l'instance tel que chaque sous-graphe peut être contracté en un graphe dans $\mathcal{F}$; l'instance admet donc un ou plusieurs graphes de $\mathcal{F}$ comme mineur. Notons que $\mathcal{F}$-Paquetage est un problème de maximisation (l'ensemble vide de modèles est une solution).

Nous devons d'abord définir quelle est la structure d'un certificat de $\mathcal{F}$-Paquetage. Étant donné un graphe, un solution est un paquet de modèles. Avant de définir ce qu'est un paquet, nous rappelons la définition de modèle.

Définition 73

Un *modèle* d'un graphe F dans une graphe G est une application Φ, qui assigne à chaque sommet $v \in V(F)$ un sous-graphe $\Phi(v)$ de G connexe et non vide et à chaque arête $e \in E(F)$ une arête $\Phi(e) \in E(G)$ tels que :

- les sous-graphes $\Phi(v)$ (pour $v \in V(F)$) sont deux à deux sommet-disjoints ;
- les arêtes $\Phi(e)$ (pour $e \in E(F)$) sont deux à deux distinctes ;
- pour $\{v; w\} \in E(F)$, $\Phi(\{v; w\})$ a une extrémité dans $\Phi(v)$ et l'autre dans $\Phi(w)$.

Nous notons $\Phi(F)$ le sous-graphe de G obtenu par union (disjointe) des sous-graphes $\Phi(v)$ pour $v \in V(F)$ plus les arêtes $\Phi(e)$ pour $e \in E(F)$.

⌟

Définition 74 : $\mathcal{F}$-paquet

Un *$\mathcal{F}$-paquet* est un ensemble M de modèles tel que les sous-graphes $\Phi(F)$ sont sommet-disjoints, pour $\Phi \in M, F \in \mathcal{F}$.

⌟

$\mathcal{F}$-Paquetage :
instance : un graphe G et un entier k ;
paramètre: l'entier k ;
question : existe-t-il un $\mathcal{F}$-paquet de taille au moins k ?

Pour des raisons techniques, dans le lemme 29 (et le reste de la sous-section 3.5.1) et le fait 6 (et la sous-section 3.5.2), nous aurons besoin de supposer que la famille de graphes $\mathcal{F}$ ne contient que des graphes connexes et au moins un graphe planaire. Pour être exact dans la dénomination,

nous devrions appeler notre problème TOUS-CONNEXES UN-PLANAIRE $\mathcal{F}$-PAQUETAGE ; par simplicité nous continuerons à le nommer $\mathcal{F}$-PAQUETAGE.

Lorsque $\mathcal{F} = \{K_3\}$ ce problème est équivalent à PAQUETAGE DE CYCLES. Par conséquent $\mathcal{F}$-PAQUETAGE est NP-complet ; du point de vue paramétré, il est W[1]-complet [19].

Nous construisons un noyau linéaire pour $\mathcal{F}$-PAQUETAGE lorsqu'il est restreint à une classe de graphes excluant un mineur fixé H. La décomposition en protrusions est obtenue à partir d'une décomposition pour le problème $\mathcal{F}$-DÉLÉTION et d'une variante de la propriété d' Erdős-Pósa [28]. L'encodeur que nous utilisons est inspiré de la programmation dynamique pour le problème PAQUETAGE DE CYCLES [8], que nous généralisons grâce aux outils d'Adler, Dorn, Fomin, Sau et Thilikos [1] ; nous le notons $\mathcal{E}_{\mathcal{F}\mathrm{P}}$.

Théorème

Soit $\mathcal{F}$ une famille finie de graphes connexes dont un est planaire. Soit H un graphe et $\mathcal{G}$ la classe des graphes excluant H comme mineur. Le problème $\mathcal{F}$-PAQUETAGE, restreint à $\mathcal{G}$, admet un noyau linéaire constructif et explicite.

Dans la sous-section 3.5.1 nous construisons une décomposition en protrusions linéaire. Et dans la sous-section 3.5.2 nous proposons un encodeur qui satisfait les conditions pour que nous puissions appliquer le corollaire 1 dans la sous-section 3.5.2.

3.5.1 Décomposition en protrusions d'une instance

Nous construisons ici la décomposition en protrusions. Nous reformulons d'abord le théorème 6 pour le cas du problème $\mathcal{F}$-DÉLÉTION ; nous obtenons ainsi un modulateur d'arborescence. Nous utilisons ensuite une variante de la propriété d'Erdős-Pósa qui établi une relation linéaire entre une instance de $\mathcal{F}$-DÉLÉTION et une instance de $\mathcal{F}$-PAQUETAGE. Cela nous permet d'utiliser le modulateur pour $\mathcal{F}$-DÉLÉTION dans l'application à $\mathcal{F}$-PAQUETAGE. Avec le théorème 5, ce modulateur nous permet enfin de construire une décomposition.

Nous commençons par définir le problème $\mathcal{F}$-DÉLÉTION. Ce problème consiste à trouver un ensemble de sommets qui intersecte tous les mineurs de $\mathcal{F}$ dans l'instance, c'est-à-dire des sommets qui, s'ils sont retirés, suppriment tous les modèles de $\mathcal{F}$.

$\mathcal{F}$-DÉLÉTION :

instance : un graphe G et un entier k' ;
paramètre: l'entier k' ;
question : existe-t-il un ensemble $D \subseteq V(G)$ de taille au plus k' tel que $G - D$ soit sans mineur de la famille $\mathcal{F}$?

Nous reformulons maintenant le théorème 6 afin d'obtenir un modulateur pour les instances positives de $\mathcal{F}$-Délétion. Il nous suffit de vérifier que $\mathcal{F}$-Délétion est bidimensionnel par mineur et séparable linéairement, ce qui est clair : la bidimensionnalité découle du fait que $\mathcal{F}$ contient un graphe planaire et la preuve de séparabilité suit le schéma du lemme 22.

Corollaire 3
Soit $\mathcal{F}$ une famille finie de graphes dont un F est planaire avec $|F| = r$. Soit H un graphe avec $|H| = h$. Si (G, k') est une instance positive de $\mathcal{F}$-Délétion, avec G sans mineur H, alors il existe un modulateur $X \subseteq V(G)$ tel que $|X| = k'$ et $\mathrm{tw}(G - X) = O(r\sqrt{r} f_m(h)^3)$. De plus, étant donné une instance (G, k'), il existe un algorithme polynomial qui ou renvoie un tel modulateur X, ou répond que (G, k') est une instance négative.
⌞ ⌟

Afin de nous ramener au problème $\mathcal{F}$-Paquetage, nous avons besoin de la propriété d'Erdős-Pósa. Cette propriété permet de transformer une instance positive de $\mathcal{F}$-Délétion en une instance négative de $\mathcal{F}$-Paquetage avec un ajustement linéaire du paramètre ; en d'autres termes, le modulateur obtenu pour une instance positive de $\mathcal{F}$-Délétion dans la corollaire 3 est également un modulateur pour une instance négative de $\mathcal{F}$-Paquetage.

Nous rappelons la définition de la propriété d'Erdős-Pósa [21], puis nous formulons le cas particulier qui relie les problèmes $\mathcal{F}$-Délétion et $\mathcal{F}$-Paquetage.

Définition 75 : propriété d'Erdős-Pósa
Une classe de graphe $\mathcal{M}$ satisfait la propriété d'Erdős-Pósa s'il existe une fonction f telle que pour tout entier k et pour tout graphe G,

— ou bien G contient k sous-graphes sommet-disjoints, tous isomorphes à un graphe de $\mathcal{M}$;
— ou bien il existe un ensemble $S \subseteq V(G)$ de taille au plus $f(k)$ tel que $G - S$ ne contient aucun sous-graphe de $\mathcal{M}$.

⌟

Étant donné un graphe F, nous appelons $\mathcal{M}(F)$ la classe des graphes qui peuvent être contractés en F. Robertson et Seymour [51] ont démontré que

la classe $\mathcal{M}(F)$ satisfait la propriété d'Erdős-Pósa si et seulement si F est planaire et connexe. Autrement dit, étant donné un graphe (nécessairement) planaire F, pour tout graphe G, ou bien G contient k copies de F comme mineur, ou bien tous les modèles de F dans G peuvent être supprimés avec $f(k)$ sommets. Plusieurs travaux se sont attelés à améliorer la fonction f, dont le plus récent est dû à Chekuri et Chuzhoy [11]. Dans le cas spécifique où le graphe G est restreint à une classe de graphes excluant un mineur fixé, Fomin, Saurabh et Thilikos [28] ont montré que la fonction f est linéaire (pour tout graphe planaire connexe F). Ces résultats se généralisent facilement si, au lieu d'un graphe planaire F, nous considérons une famille $\mathcal{F}$ finie de graphes connexes et contenant au moins un graphe planaire.

De tout cela découle le théorème suivant, dans lequel nous précisons une borne supérieure pour la fonction linéaire f.

Lemme 29 : [28]

Soit $\mathcal{F}$ une famille finie de graphes dont un F est planaire avec $|F| = r$. Soit H un graphe avec $|H| = h$. Si (G, k), avec G sans mineur H, est une instance négative de $\mathcal{F}$-Paquetage, alors $(G, O(r2^{15h+8h\log h}) \cdot k)$ est une instance positive de $\mathcal{F}$-Délétion.

Nous sommes maintenant en mesure de construire la décomposition en protrusions. Comme pour r-Indépendance nous partons d'un modulateur pour un autre problème ; en utilisant la propriété d'Erdős-Pósa, nous revenons à $\mathcal{F}$-paquet ; il ne nous reste plus qu'à utiliser le théorème 5 pour obtenir la décomposition en protrusions.

Lemme 30

Soit $\mathcal{F}$ une famille finie de graphes dont un F est planaire avec $|F| = r$. Soit H un graphe avec $|H| = h$. Soit (G, k) une instance de $\mathcal{F}$-Paquetage avec G sans mineur H. Il existe un algorithme polynomial qui calcule une $O(h^2 2^{O(h\log h)} r^{5/2} f_m(h)^3) \cdot k, O(r\sqrt{r} f_m(h)^3)$-décomposition en protrusions ou répond que (G, k) est une instance positive.

Démonstration. Nous utilisons l'algorithme de la corollaire 3 sur l'instance (G, k'), avec $k' = O(r2^{15h+8h\log h}) \cdot k$. Celui-ci répond que (G, k') est une instance négative de $\mathcal{F}$-Délétion, ou renvoie un modulateur X de taille $|X| = c \cdot k' = k'$ avec $\mathrm{tw}(G - X) = t = O(r\sqrt{r} f_m(h)^3)$. Dans le premier cas, d'après le lemme 29, nous pouvons conclure que (G, k) est une instance positive de $\mathcal{F}$-Paquetage. Dans le second cas, nous appliquons l'algorithme du théorème 5 qui renvoie une $((\alpha_H \cdot tc) \cdot k', 2t + h)$-décomposition, avec $\alpha_H = O(2^{O(h\log h)})$. □

3.5.2 Encodeur pour $\mathcal{F}$-Paquetage

Nous rappelons que, intuitivement, notre encodeur doit décrire comment une solution peut traverser un séparateur. Pour le cas de $\mathcal{F}$-Paquetage, la structure d'une solution est relativement compliquée : il s'agit d'un paquet de modèles, c'est-à-dire, informellement, un ensemble de sous-graphes. Par conséquent, un séparateur peut intersecter des modèles. Décrire comment la solution traverse le séparateur revient à décrire les modèles intersectés. Au cours de l'exécution de la programmation dynamique, cela revient à réserver de la place pour les modèles qui pourraient potentiellement être réalisés plus tard. Il faut noter une différence majeure entre cette application et les deux précédentes : les contraintes imposées par l'encodeur ne sont pas locales, en effet, dans certains cas, un modèle potentiel peut descendre profondément dans le sous-graphe considéré (c'est-à-dire, utilise des sommets à une distance arbitraire du bord). Dans une certaine mesure, cela veux dire qu'il est difficile de définir une version optionnelle de l'encodeur ; au sens où il n'y a pas de façon intuitive d'ordonner les encodages et d'identifier l'encodage associé à la valeur minimale ou maximale de la table.

Pour décrire comment les modèles potentiels intersectent le séparateur, nous utilisons la notion de enracinement [1] (qui est une adaptation pour la programmation dynamique du folio de Robertson et Seymour [52]). Remarquons que les enracinements sont initialement définis pour les décompositions branchantes, mais peuvent facilement être transférés aux décompositions arborescentes. Pour simplifier, supposons que $\mathcal{F} = \{F\}$ est un singleton ; la généralisation à une famille $\mathcal{F}$ finie de graphes connexes s'induit facilement.

Définition 76 : enracinement

Soit F un graphe connexe. Soit G un graphe de bord B. Un *enracinement* de F sur B est un quintuplet $(\mathcal{A}, S_F^*, S_F, \psi, \chi)$ tel que :

- $S_F \subseteq S_F^*$ sont tous deux des sous-ensembles de $V(F)$;
- $\mathcal{A}$ est une collection (possiblement vide) de sous-ensembles non vides de B deux à deux disjoints ;
- $\psi : \mathcal{A} \to S_F$ est une application surjective ;
- $\chi : S_F \times S_F \to \{0, 1\}$ est une fonction binaire symétrique.

⌟

Définition 77 : modèle potentiel

Soit F un graphe connexe. Soit G un graphe de bord B. Soit ($\mathcal{A}$, S_F^*, S_F, ψ, χ) un enracinement de F sur B. Un *modèle potentiel* satisfaisant ($\mathcal{A}$, S_F^*, S_F, ψ, χ) est une application partielle Φ qui assigne

— à chaque sommet $v \in S_F$ un sous-graphe $\Phi(v)$ de G non vide et tel que $\{A \in \mathcal{A} \mid \psi(A) = v\}$ est l'ensemble des intersections de B avec les composantes connexes de $\Phi(v)$;

— à chaque sommet $v \in S_F^*$ un sous-graphe $\Phi(v)$ de G connexe, non vide ;

— à chaque arête $e \in \{e \in E(F) \mid \chi(e) = 1 \vee e \in S_F^* \times (S_F^* \setminus S_F)\}$ une arête $\Phi(e)$ de G ;

tel que Φ vérifie que :

— les sous-graphes $\Phi(v)$ (pour $v \in V(F)$) sont deux à deux sommet-disjoints ;

— les arêtes $\Phi(e)$ (pour $e \in E(F)$) sont deux à deux distinctes ;

— pour $\{v; w\} \in E(F)$, $\Phi(\{v; w\})$ a une extrémité dans $\Phi(v)$ et l'autre dans $\Phi(w)$.

Nous notons $\Phi(F)$ le sous-graphe de G obtenu par union (disjointe) des sous-graphes $\Phi(v)$ pour $v \in V(F)$ plus les arêtes $\Phi(e)$ pour $e \in E(F)$. ⌟

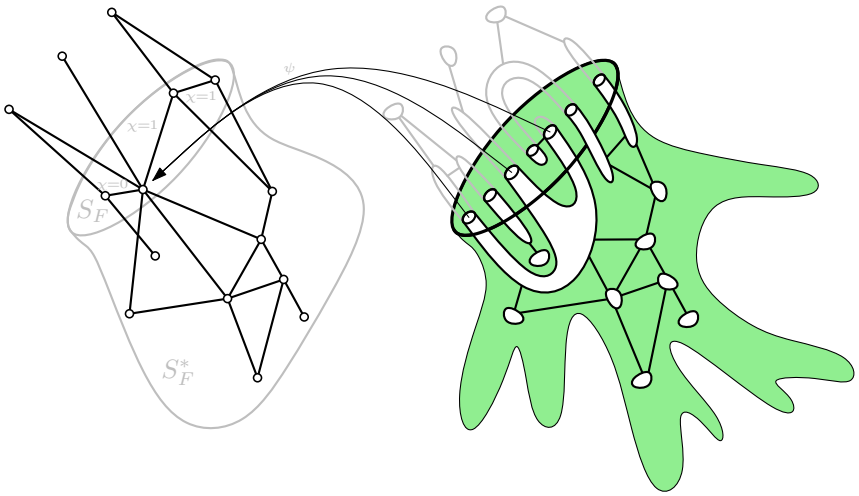


FIGURE 3.14 – Définition schématique d'un enracinement et du modèle potentiel qui le satisfait (cf. définitions 76 et 77).

La collection $\mathcal{A}$ représente les intersections des composantes connexes du modèle potentiel avec le séparateur B. Les sous-ensembles $S_F^*, S_F \subseteq V(F)$ et la fonction χ indiquent que nous considérons un modèle potentiel de $F[S_F^*]$

dans G contenant les arêtes de $S_F \times S_F$ désignées par χ. En d'autres termes, la fonction χ désigne quelles arêtes (sur le bord) ont déjà été trouvées à cette étape de la programmation. Puisque les modèles des sommets sont possiblement non connexes, nous avons besoin d'identifier leurs composantes ; l'ensemble S_F indique quels modèles de sommets intersectent B et l'application ψ associe les ensembles de $\mathcal{A}$ aux sommets de S_F. Autrement dit, ψ peut être vu comme un étiquetage des ensembles de $\mathcal{A}$ tel que deux ensembles avec la même étiquette appartiennent au même modèle de sommet (mais seront connectés plus tard au cours de la programmation dynamique).

Il a été démontré [1] que les enracinements permettent d'utiliser la programmation dynamique dans le but de déterminer si un graphe G contient un mineur F. Il est facile de montrer que le nombre d'enracinements possibles, sur un séparateur de taille t d'un mineur F avec $V(F) = r$, est au plus de $f(t, F) = 2^{t \log t} r^t 2r^2$. Remarquons que, en particulier, cela prouve que, la relation $\cdot \approx_{\mathcal{G},t} \cdot$ a au plus $2^{t \log t} r^t 2^{r^2}$ classes d'équivalence lorsque $\mathcal{G}$ est la classe des graphes excluant H comme mineur avec $V(H) = h$.

Le générateur $\mathcal{C}^{\mathcal{E}_{\mathcal{F}\text{P}}}$

Soit $G \in \mathcal{B}_t$ de bord ∂G labellé par $\Lambda(G)$. La fonction $\mathcal{C}^{\mathcal{E}_{\mathcal{F}\text{P}}}$ envoie $\Lambda(G)$ sur un ensemble $\mathcal{C}^{\mathcal{E}_{\mathcal{F}\text{P}}}(\Lambda(G))$ d'encodages. Chaque encodage $R \in \mathcal{C}^{\mathcal{E}_{\mathcal{F}\text{P}}}(\Lambda(G))$ est un ensemble de au plus $|\Lambda(G)|$ enracinements $\{(\mathcal{A}_i, S^*_{F_i}, S_{F_i}, \psi_i, \chi_i) \mid F_i \in \mathcal{F}\}$ dans lequel chaque enracinement code le modèle potentiel d'un mineur $F_i \in \mathcal{F}$ (plusieurs occurrence d'un même mineur sont possibles).

Le langage $L^{\mathcal{E}_{\mathcal{F}\text{P}}}$

Soit $G \in \mathcal{B}_t$ de bord ∂G. Pour un $\mathcal{F}$-paquet $\mathcal{S}$ nous posons que $(G, \mathcal{S}, R) \in L^{\mathcal{E}_{\mathcal{F}\text{P}}}$ (que $\mathcal{S}$ est un $\mathcal{F}$-paquet partiel satisfaisant R) s'il existe un paquet de modèles potentiels satisfaisant les enracinements dans R dans $G - \bigcup_{\Phi \in \mathcal{S}} \Phi(F)$.

Remarquons que nous autorisons les modèles entiers à intersecter ∂G, mais ils ne doivent pas empiéter sur les modèles potentiels imposés par l'encodage R. Remarquons aussi qu'une solution partielle est également un $\mathcal{F}$-paquet.

La fonction $\mathcal{F}$

Pour $\mathcal{F}$-Paquetage, nous utilisons la fonction d'optimisation pertinente, avec $g : t \mapsto t$. Comme il a été discuté dans la section 3.2, nous avons besoin de définir de quelle façon une solution induit un encodage lorsqu'elle traverse un séparateur. Nous voulons que les enracinements dans l'encodage correspondent aux intersections des modèles (de la solution) avec le séparateur.

Définissons formellement la signification de induire. Soit G de bord A et G_B de bord B tels que dans la définition 66. Soit Φ un modèle de F (respectivement, un modèle potentiel). Nous définissons l'enracinement $(\mathcal{A}, S_F^*, S_F, \psi, \chi)$ sur B, induit par Φ tel que :

— $\mathcal{A}$ contient les éléments de la forme $B \cap C$ où C est une composante connexe du graphe $G_B[V(\Phi(v))]$ avec $v \in V(F)$;

— ψ étiquette chaque élément de $\mathcal{A}$ avec le sommet correspondant de F ;

— S_F^*, S_F correspondent aux sommets de F dont les modèles intersectent G_B et B respectivement.

Soit R_A un encodage sur A. Soit $\mathcal{S}$ un $\mathcal{F}$-paquet partiel satisfaisant R_A et $\mathcal{P}$ l'ensemble des modèles potentiels satisfaisants les enracinements dans R_A. Nous définissons l'encodage R_B comme l'ensemble des enracinements induits par les modèle de $\mathcal{S} \cup \mathcal{P}$. Clairement, le paquet M des modèles de $\mathcal{S}$ entièrement réalisés dans G_B est une solution partielle satisfaisant R_B.

Nous pouvons maintenant définir formellement un encodage non pertinent. Soit $G \in \mathcal{B}_t$ de bord A. Soit R_A un encodage sur A. Un encodage est *non pertinent* si il existe un certificat $\mathcal{S}$ satisfaisant R_A et un séparateur B avec $|B| \leqslant t$ et $B \neq A$ tel que $\mathcal{S}$ induise un encodage R_B sur B avec $\bar{f}_g^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}(G_B, R_B) = -\infty$.

Soit $G \in \mathcal{B}_t$ de bord ∂G. Nous définissons $f^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}$ telle que :

$$\bar{f}_g^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}(G,R) = \begin{cases} -\infty, & \text{si } f^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}(G,R) + g(t) > \\ & \max\{f^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}(G,R) \mid R \in \mathcal{C}^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}(\Lambda(G))\}, \\ & \text{ou si } R \text{ est non pertinent,} \\ f^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}(G,R), & \text{sinon.} \end{cases}$$

avec $f^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}(G,R) = \max\{k \mid \exists \mathcal{S}, |\mathcal{S}| \leqslant k, (G,\mathcal{S},R) \in L^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}\}$.

Remarquons que nous avons choisi de ne pas inclure les modèles potentiels dans une solution partielle, et donc que $f^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}$ compte le nombre maximum de modèles entièrement réalisables dans G en satisfaisant R.

La taille $s_{\mathcal{E}_{\mathcal{F}\mathrm{P}}}$

Rappelons que $f(t,F) = 2^{t \log t} r^t 2^{r^2}$ est le nombre d'enracinements d'un mineur F avec $|F| = r$ sur un bord B avec $|B| = t$. Pour $\mathcal{F}$ une famille finie de graphes connexes, nous posons $r = \max_{F \in \mathcal{F}} |F|$. Par définition de $\mathcal{C}^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}$, la taille de $\mathcal{E}_{\mathcal{F}\mathrm{P}}$ peut être bornée par :

$$s_{\mathcal{E}_{\mathcal{F}\mathrm{P}}}(t) \leqslant (\sum_{j \in J} 2^{j \log j} r^j 2^{r^2}) \leqslant (\sum_{j \in J} 2^{j \log j} r^j) 2^{r^2} \leqslant t \cdot 2^{t \log t} r^t 2^{r^2},$$

avec J un ensemble quelconque d'entiers tel que $\sum_{j \in J} j \leqslant t$.

Remarquons qu'une autre possibilité consiste à définir les encodages comme un unique enracinement d'un mineur (non connexe) formé par l'union disjointe d'au plus t mineurs de $\mathcal{F}$ (dans ce cas nous pouvons borner la taille de $\mathcal{E}_{\mathcal{F}\mathrm{P}}$ par $s_{\mathcal{E}_{\mathcal{F}\mathrm{P}}}(t) \leqslant 2^{t \log t}(tr)^t 2^{(tr)^2}$).

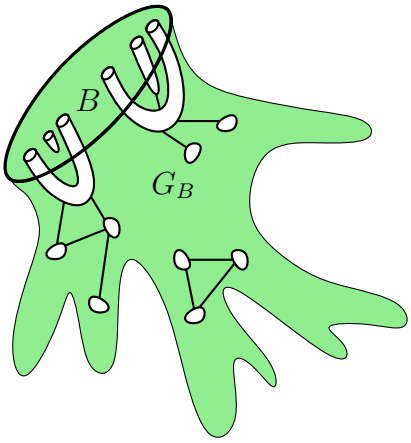


FIGURE 3.15 – Illustration de l'encodeur $\mathcal{E}_{\mathcal{F}\mathrm{P}}$ pour $\mathcal{F}$-PAQUETAGE définie dans cette sous-section 3.5.2.

Nous montrons maintenant que l'encodeur $\mathcal{E}_{\mathcal{F}\mathrm{P}}$ défini ci-dessus vérifie toutes les conditions pour que notre cadre puisse être appliqué. Tout d'abord nous montrons que $\mathcal{E}_{\mathcal{F}\mathrm{P}}$ est un $\mathcal{F}$-PAQUETAGE-encodeur. Ensuite nous prouvons que $\mathcal{E}_{\mathcal{F}\mathrm{P}}$ est g-confiné pour $g(t) = t$. Pour finir nous démontrons que l'équivalence $\cdot \sim^*_{\mathcal{E}_{\mathcal{F}\mathrm{P}},\mathcal{G},t} \cdot$ est adaptée.

Lemme 31

L'encodeur $\mathcal{E}_{\mathcal{F}\mathrm{P}} = (\mathcal{C}^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}, L^{\mathcal{E}_{\mathcal{F}\mathrm{P}}} \bar{f}_g^{\mathcal{E}_{\mathcal{F}\mathrm{P}}})$ est un $\mathcal{F}$-PAQUETAGE-encodeur.

Démonstration. Il y a un unique encodage $R_\emptyset \in \mathcal{C}^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}(\emptyset)$ (un ensemble vide d'enracinements), et $(G, S, R_\emptyset) \in L^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}$ si et seulement si S est un $\mathcal{F}$-PAQUETAGE de G, par définition de $L^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}$. Maintenant $\bar{f}_g^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}(G, R_\emptyset)$ peut prendre deux valeurs, ou bien $\bar{f}_g^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}(G, R_\emptyset) = f^{\Pi}(G)$; ou bien $\bar{f}_g^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}(G, R_\emptyset) = -\infty$. Nous montrons que le second cas n'a pas lieu. En effet, il convient de vérifier qu'une solution optimale n'a pas été construite à partir d'une solution partielle dont la valeur a été tronquée (autrement dit, en propageant une valeur tronquée, nous oublions comment construire les extensions de la solution partielle, il ne faut pas que ce soit le cas de la solution optimale). Remarquons que cette argumentation est nouvelle par rapport aux lemmes 21 et 26, cependant les arguments correspondent peu ou prou à ceux du lemme 22.

Soit G un graphe 0-bordé. Soit $\mathcal{S}$ un $\mathcal{F}$-paquet maximum (c'est-à-dire, $|\mathcal{S}| = f^{\Pi}(G)$) de G. Par l'absurde supposons que $\bar{f}_g^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}(G, R_\emptyset) = -\infty$. Par la forme inductive de la définition de $\bar{f}_g^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}$ nous pouvons dire qu'il existe un séparateur B avec $|B| \leqslant t$ et $B \neq A$ tel que $\mathcal{S}$ induit un encodage R_B et $f^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}(G_B, R_B) + t \leqslant \max\{f^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}(G, R) \mid R \in \mathcal{C}^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}(\Lambda(G_B))\}$ (autrement dit, la valeur de R_B a été tronquée lorsque la programmation dynamique considérait B, et cela s'est propagé jusqu'à la racine ; les valeurs des encodages induits par $\mathcal{S}$ en dessous de B ne sont, quant à elles, pas tronquées). Soit M l'ensemble des modèles de $\mathcal{S}$ entièrement réalisés dans G_B. Nous avons que $|M| = f^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}(G_B, R_B)$, sinon $\mathcal{S}$ n'est pas maximum. Soit M_B l'ensemble des modèles de $\mathcal{S}$ qui intersectent B. Nous avons $|M_B| \leqslant t$. Soit R_0 l'encodage vide sur B et M_0 satisfaisant R_0 et de taille maximum (c'est-à dire, M_0 est un $\mathcal{F}$-paquet maximum de G_B). Nous avons $|M_0| = f^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}(G_B, R_0) = \max\{f^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}(G, R) \mid R \in \mathcal{C}^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}(\Lambda(G_B))\}$. Remarquons que $\mathcal{S} \setminus (M \cup M_B) \cup M_0$ est un $\mathcal{F}$-paquet et donc nous avons $\mathcal{S} \setminus (M \cup M_B) \cup M_0 \leqslant |\mathcal{S}|$ (car $\mathcal{S}$ est maximum). D'après cette inégalité nous obtenons $|M_0| \leqslant |M| + t$, contradiction avec la définition de $\bar{f}_g^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}$. □

Lemme 32

L'encodeur $\mathcal{E}_{\mathcal{F}\mathrm{P}} = (\mathcal{C}^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}, L^{\mathcal{E}_{\mathcal{F}\mathrm{P}}} \bar{f}_g^{\mathcal{E}_{\mathcal{F}\mathrm{P}}})$ est g-confiné pour $g : t \mapsto t$.

Démonstration. Par définition de $\bar{f}_g^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}$, $\mathcal{E}_{\mathcal{F}\mathrm{P}}$ est g-confiné pour $g(t) = t$. □

Fait 6

Soit G, G' de bord A et G_B, G'_B de bord B tels que dans la définition 66. Soit Φ un modèle (respectivement, un modèle potentiel) d'un graphe F dans G. Soit $(\mathcal{A}, S_F^*, S_F, \psi, \chi)$ l'enracinement induit par Φ. Si G'_B admet un modèle potentiel Φ'_B satisfaisant $(\mathcal{A}, S_F^*, S_F, \psi, \chi)$, alors G' admet un modèle (respectivement, un modèle potentiel) de F.

Démonstration. Nous construisons un modèle Φ' de F dans G'.

Pour chaque sommet $v \in V(F) \setminus S_F^*$, nous posons $\Phi'(v) = \Phi(v)$. Pour chaque sommet $v \in S_F^* \setminus S_F$, nous posons $\Phi'(v) = \Phi'_B(v)$. Pour chaque sommet $v \in S_F$, nous posons $\Phi'(v) = \Phi(v)[V(G^-)] \oplus \Phi'_B(v)$. Puisque $\Phi(v)$ est connexe et que les composantes connexes de $\Phi'_B(v)$ ont les mêmes bords que celles de $\Phi(v)[V(G_B)]$ (par définition d'enracinement), il advient que $\Phi'(v)$ est connexe. Remarquons que, pour tout $u \in V(F)$, $\Phi'(v)$ n'intersectent pas $\Phi'(u)$, car $\Phi(v), \Phi'_B(v)$ n'intersecte pas $\Phi'(u)$.

Pour chaque arête $e \in V(F) \times (V(F) \setminus S_F^*)$ ou telle que $\chi(e) = 0$ nous posons $\Phi'(e) = \Phi(e)$. Pour chaque arête e in $S_F^* \times (S_F^* \setminus S_F)$ ou telle que $\chi(e) =$

1 nous posons $\Phi'(e) = \Phi'_B(e)$. Puisque B est un séparateur de G, S_F est un séparateur de F et donc, il n'y a pas d'arêtes dans $(V(F) \setminus S_F^*) \times (S_F^* \setminus S_F)$. Puisque Φ, Φ'_B sont respectivement un modèle et un modèle potentiel, les arêtes $\Phi'(e), e \in E(F)$ sont distinctes et si $e = \{u, v\}$, alors $\Phi'(e)$ a une extrémité dans $\Phi'(u)$ et l'autre dans $\Phi'(v)$.

□

Lemme 33

L'équivalence $\cdot \sim^*_{\mathcal{E}_{\mathcal{FP}},\mathcal{G},t} \cdot$ associée à $\mathcal{E}_{\mathcal{FP}} = (\mathcal{C}^{\mathcal{E}_{\mathcal{FP}}}, L^{\mathcal{E}_{\mathcal{FP}}} \bar{f}_g^{\mathcal{E}_{\mathcal{FP}}})$ est adaptée pour $\mathcal{G}$ une classe de graphe quelconque.

Démonstration. Nous rappelons que nous avons posé G, G' de bord A et G^-, G_B, G'_B de bord B dans la définition 66 (cf. figure 3.7).

Conformément à la définition 67, nous voulons montrer que $G \sim^*_{\mathcal{E}_{\mathcal{FP}},\mathcal{G},t} G'$ et que $\Delta_{\mathcal{E}_{\mathcal{FP}},t}(G, G') = \Delta_{\mathcal{E}_{\mathcal{FP}},t}(G_B, G'_B)$. D'après le fait 5 il nous suffit de montrer que $G \sim^*_{\mathcal{E}_{\mathcal{FP}},t} G'$ (sans tenir compte de la classe $\mathcal{G}$). Par conséquent, nous voulons prouver que $\bar{f}_g^{\mathcal{E}_{\mathcal{FP}}}(G, R_A) = \bar{f}_g^{\mathcal{E}_{\mathcal{FP}}}(G', R_A) + \Delta_{\mathcal{E}_{\mathcal{FP}},t}(G_B, G'_B)$ pour tout $R_A \in \mathcal{C}^{\mathcal{E}_{\mathcal{FP}}}(\Lambda(G))$.

Soit $R_A \in \mathcal{C}^{\mathcal{E}_{\mathcal{FP}}}(\Lambda(G))$.

Tout d'abord supposons que $\bar{f}_g^{\mathcal{E}_{\mathcal{FP}}}(G, R_A) \neq -\infty$. Soit $\mathcal{S} = M \cup M_B \cup M^-$ un $\mathcal{F}$-paquet partiel satisfaisant R_A et de taille maximale (c'est-à-dire, $(G, \mathcal{S}, R_A) \in L^{\mathcal{E}_{\mathcal{FP}}}$ et $f^{\mathcal{E}_{\mathcal{FP}}}(G, R_A) = |\mathcal{S}|$) partitionné en M l'ensemble des modèles entièrement dans G_B, M^- l'ensemble des modèles entièrement dans G^-, et M_B l'ensemble des modèles qui intersectent B. Soit $\mathcal{P}$ l'ensemble des modèles potentiels satisfaisant les enracinements dans R_A. Enfin, soit R_B l'encodage sur B induit par $\mathcal{S} \cup \mathcal{P}$. Remarquons que $\bar{f}_g^{\mathcal{E}_{\mathcal{FP}}}(G'_B, R_B) \neq -\infty$ (par définition de $\bar{f}_g^{\mathcal{E}_{\mathcal{FP}}}$). Contrairement aux lemmes 23 et 28, nous n'avons pas besoin de décrire comment construire un encodage à partir de $\mathcal{S}$ puisque nous l'avons déjà fait pour définir $\bar{f}_g^{\mathcal{E}_{\mathcal{FP}}}$. Il nous reste donc à montrer qu'un $\mathcal{F}$-paquet partiel M' satisfaisant R_B dans G'_B peut être recollé avec M^- pour former un $\mathcal{F}$-paquet S' dans G'. Il nous faut encore démontrer que le décalage entre G_B et G'_B se propage sur G et G'.

Soit M' un $\mathcal{F}$-paquet de G'_B satisfaisant R_B de taille maximale. Considérons les modèles potentiels satisfaisant R_B dans G'_B. Il y en a deux genres. Les premiers satisfont les enracinements dans R_B qui existent à cause de l'intersection de modèles de $\mathcal{S}$ avec B. Soit M'_B l'ensemble des modèles obtenus en recollant les modèles potentiels satisfaisant R_B avec les modèles potentiels définis par l'intersection des modèles de M_B avec G^-. Les autres satisfont les enracinements qui existent à cause de l'intersection de modèles potentiels de $\mathcal{P}$ avec B. Soit $\mathcal{P}'$ l'ensemble de modèles potentiels obtenu en recollant les modèles potentiels avec les modèles potentiels définis par l'intersection des modèles de $\mathcal{P}$ avec G^-. Soit $\mathcal{S}' = M' \cup M'_B \cup M^-$.

Nous montrons maintenant que $\mathcal{S}'$ est bien un $\mathcal{F}$-paquet partiel de G' satisfaisant R_A. Par définition, M' et M^- sont des $\mathcal{F}$-paquets. D'après le fait 6, M'_B contient bien des modèles disjoints. Observons que les modèles dans $M' \cup M^-$ sont disjoints (car les graphes G'_B et G^- sont disjoints), que les modèles dans $M'_B \cup M^-$ sont disjoints (car ceux de $M_B \cup M^-$ le sont), et que les modèles dans $M' \cup M'_B$ sont disjoints (car M' satisfait R_B). Par conséquent, $\mathcal{S}' = M' \cup M'_B \cup M^-$ est un $\mathcal{F}$-paquet. Vérifions maintenant que $\mathcal{S}'$ satisfait R_A. D'après le fait 6, $\mathcal{P}'$ contient bien des modèles potentiels disjoints. Observons que les modèles dans $\mathcal{P}' \cup M'$ sont disjoints (car M' satisfait R_B), que les modèles dans $\mathcal{P}' \cup M'_B$ sont disjoints (par définition de R_B), et que les modèles dans $\mathcal{P}' \cup M^-$ sont disjoints (car car $\mathcal{S}$ satisfait R_A). Par conséquent $\mathcal{S}$ satisfait R_A.

Il nous reste à montrer que $\mathcal{S}'$ a la taille attendue. Par construction de R_B, $|M'| = \bar{f}_g^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}(G'_B, R_B)$ (sinon $\mathcal{S}$ n'est pas maximum). Observons que $|M_B| = |M'_B|$. Puisque $G'_B \sim^*_{\mathcal{E}_{\mathcal{F}\mathrm{P}},t} G_B$ et $\bar{f}_g^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}(G'_B, R_B) \neq -\infty$, nous savons que $|M'| = \bar{f}_g^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}(G_B, R_B) + \Delta_{\mathcal{E}_{\mathcal{F}\mathrm{P}},t}(G_B, G'_B)$ et donc $|\mathcal{S}'| = |M' \cup M'_B \cup M^-| = \bar{f}_g^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}(G_B, R_B) + \Delta_{\mathcal{E}_{\mathcal{F}\mathrm{P}},t}(G_B, G'_B) + |M_B| + |M^-| \geqslant \bar{f}_g^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}(G, R_A) + \Delta_{\mathcal{E}_{\mathcal{F}\mathrm{P}},t}(G_B, G'_B)$.

Enfin supposons que $\bar{f}_g^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}(G, R_A) = -\infty$. Il s'en suit que $\bar{f}_g^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}(G', R_A) = -\infty$ aussi. En effet, supposons que ce ne soit pas le cas. Il existe alors un $\mathcal{F}$-paquet partiel de G' satisfaisant R_A, et d'après l'argumentation qui précède nous pouvons construire un $\mathcal{F}$-paquet partiel de G ; contradiction avec $\bar{f}_g^{\mathcal{E}_{\mathcal{F}\mathrm{P}}}(G, R_A) = -\infty$.

En conséquence de quoi, nous pouvons conclure que $G \sim^*_{\mathcal{E}_{\mathcal{F}\mathrm{P}},t} G'$ et donc que la relation d'équivalence est adaptée.

□

3.5.3 Noyau pour $\mathcal{F}$-Paquetage

Nous pouvons maintenant énoncer le noyau linéaire pour $\mathcal{F}$-Paquetage.

Théorème 9

Soit $\mathcal{F}$ une famille finie de graphes connexes dont un F est planaire avec $|F| = r$. Soit H un graphe avec $|H| = h$ et $\mathcal{G}$ la classe des graphes excluant H comme mineur. Le problème $\mathcal{F}$-Paquetage, restreint à $\mathcal{G}$, admet une noyau linéaire de taille $O(b(\mathcal{E}_{\mathcal{F}\mathrm{P}}, g, O(r\sqrt{r} f_m(h)^3), \mathcal{G}) \cdot 2^{O(h \log h)} r^{\frac{5}{2}} f_m(h)^3) \cdot k$.

Démonstration. Soit (G, k) une instance de r-Indépendance avec G sans mineur H. D'après le lemme 30, ou (G, k) est une instance positive, ou nous pouvons construire une $(\alpha_H tc \cdot k, 2t+h)$-décomposition en protrusions. Dans le premier cas, nous renvoyons une instance trivialement positive (par exemple,

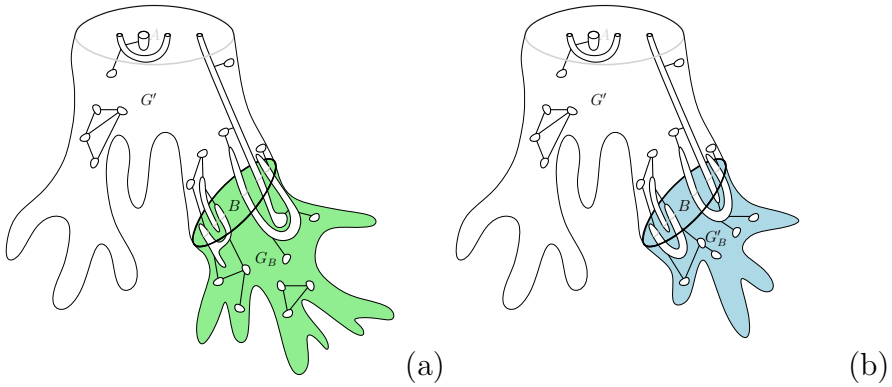


FIGURE 3.16 – Illustration du remplacement de protrusion et de la reconstruction d'un $\mathcal{F}$-PAQUETAGE dans le lemme 33. (a) La protrusion initiale G. (b) La protrusion après remplacement G'. Par exemple, les deux modèles qui traversent B peuvent être reconstruits car il G_B et G'_B partagent les mêmes enracinements.

$(G = F \in \mathcal{F}, k = 1)$). Dans le second cas nous considérons l'encodeur $\mathcal{E}_{\mathcal{FP}}$ défini dans la sous-section 3.5.2. D'après les lemmes 31, 32, et 33 $\mathcal{E}_{\mathcal{FP}}$ est un $\mathcal{F}$-PAQUETAGE-encodeur g-confiné (pour $g : t \mapsto t$) et l'équivalence $\cdot \sim^*_{\mathcal{E}_{\mathcal{FP}},\mathcal{G},t} \cdot$ est adaptée. Nous savons que $s_{\mathcal{E}_{\mathcal{FP}}}(t) \leqslant 2^{t \log t}(tr)^t 2^{tr^2}$. Par conséquent, nous pouvons appliquer le corollaire 1 qui construit un noyau linéaire pour r-INDÉPENDANCE de taille :

$$\alpha_H tc \cdot (b(\mathcal{E}_{\mathcal{FP}}, g, t, \mathcal{G}) + 1) \cdot k',$$

avec

- $b(\mathcal{E}, g, t, \mathcal{G})$ la borne sur la taille des représentants (lemme 18) ;
- $t = O(r\sqrt{r} f_m(h)^3)$ la borne sur la largeur arborescente (corollaire 2) ;
- $c = 1$ la constante du modulateur (corollaire 2) ;
- $\alpha_H = O(2^{O(h \log h)})$ la constante de la décomposition (théorème 5) ;
- $f_m(h) \leqslant 2^{O(h^2 \log h)}$ la fonction liant largeur et grille (proposition 7) ;
- $k' = O(r2^{O(h \log h)}) \cdot k$ le paramètre pour $\mathcal{F}$-DÉLÉTION (lemme 29).

□

3.6 Discussion

Dans ce chapitre nous avons élaboré une cadre générique de remplacement de protrusions et nous l'avons appliqué à trois problèmes : r-DOMINATION, r-INDÉPENDANCE, et $\mathcal{F}$-PAQUETAGE ; pour obtenir des noyaux linéaires sur

des classes sans mineur ou sans mineur apex. Ce cadre est complémentaire des méta-résultats du domaine [8, 26, 43], d'une part il utilise ces résultats pour obtenir des décompositions en protrusions (dont il a besoin pour construire les noyaux), d'autre part il rend constructifs et explicites certains noyaux dont seule l'existence été démontrée (en utilisant le remplacement de Bodlaender, Fomin, Lokshtanov, Penninkx, Saurabh et Thilikos [8]).

Signalons que, outre les trois exemples donnés dans ce chapitre, ce cadre a également été utilisé sur plusieurs autres problèmes. Il permet de construire une extraction de noyaux pour le problème $\mathcal{F}$-Délétion [33], dans les classe de graphes excluant un mineur, en supposant que $\mathcal{F}$ est une famille finie de graphes connexes dont un est planaire. En combinant les encodeurs de r-Domination et $\mathcal{F}$-Délétion d'une part, et de r-Indépendance et $\mathcal{F}$-Paquetage d'autre part, nous pouvons obtenir des noyaux pour les problèmes r-$\mathcal{F}$-Délétion (qui consiste à trouver un ensemble de sommets qui dominent à distance r tous les mineurs de $\mathcal{F}$) et r-$\mathcal{F}$-Paquetage [32] (qui consiste à trouver un paquet de mineurs de $\mathcal{F}$ à distance r les uns des autres), respectivement. Enfin, en adaptant l'encodeur de $\mathcal{F}$-Paquetage, nous parvenons à des noyaux pour diverses versions de ce problème [32].

Tous ces problèmes peuvent être regroupés en deux catégories : ceux dont le certificat est un ensemble de sommets et ceux dont le certificat est un ensemble de sous-graphe. Nous pensons que notre cadre n'est pas restreint à ces deux catégories, et qu'il permet d'aborder des problèmes dont la solution est un ensemble d'arêtes ou encore des problèmes sur des graphes pondérés ou les graphes orientés.

Remarquons aussi que notre cadre de remplacement de protrusions est indépendant des méthodes pour obtenir des décompositions en protrusions. Il peut être utilisé dès lors qu'il est possible d'identifier une protrusion. Par exemple, il nous permet de montrer une extraction non déterministe de $\mathcal{F}$-Délétion dans les graphes sans mineur topologique (en effet, nous ne connaissont pas de méthode pour trouver une décomposition dans cette classe, mais nous pouvons calculer des protrusions au fur et à mesure). Il peut également servir à extraire des noyaux polynomiaux (à partir de décompositions polynomiales).

Une amélioration conventionnelle de notre cadre consisterait à diminuer les bornes sur la taille des noyaux. La constructions de nos noyaux utilise un grand nombre de résultats intermédiaires, et l'amélioration de chacun d'eux peut entraîner une diminution notable de nos constantes. Remarquons que nous considérons des problèmes modulables par r ou $\mathcal{F}$ (qui peut être vu comme un second paramètre) que nous considérons une constante dont la

valeur est arbitraire (autrement dit, ces problèmes peuvent être considérés soit comme une famille dont chaque problème est défini par une valeur fixée ; soit comme un méta-problème paramétré par une seconde valeur) ce qui donne à nos applications une certaine généricité. De la même manière, nous nous restreignons aux graphes sans mineur H, où H est un graphe fixé, mais indéterminé. Par conséquent, la constante multiplicative de nos noyaux est fonction d'une part du problème (de r ou de $\mathcal{F}$), d'autre part de la classe de graphe (de H). Dans nos noyaux, la dépendance en r ou en $\mathcal{F}$, le second paramètre, est une triple exponentielle ($O(2^{2^{2^r}})$) tandis que la dépendance en H, le mineur exclu, comprend une fonction f_m ou f_c. Diminuer la taille de nos noyaux consiste à diminuer la dépendance en l'une ou l'autre de ces valeurs. En particulier, exhiber une borne sur la fonction f_c rendrait nos résultats complètement explicites.

A l'inverse il serait intéressant de voir dans quelle mesure la généricité de notre cadre rend inévitable des constantes multiplicatives larges dans la taille des noyaux. Autrement dit, il serait intéressant de monter une borne inférieure sur la taille des noyaux que peut produire notre cadre. Une telle démonstration se baserait probablement sur des outils de la théorie des automates ou sur des hypothèses de la théorie de la complexité (classique ou paramétrée). Plus généralement, la relation entre la programmation dynamique et les noyaux, telle qu'elle est mise en evidence par notre résultat, pourrait être un point de départ pour rapprocher les résultats négatifs de la programmation dynamique et les bornes inferieures sur les noyaux .

Enfin, nous pensons qu'il est possible d'accéder à une compréhension plus globale des différents méta-théorèmes sur les noyaux ; cela pourrait, entre autre, rendre plus abordables les preuves nécessaires pour vérifier qu'un problème entre dans notre cadre (en particulier la démonstration qu'un encodeur est adapté). En particulier, nous avons souligné dans la section 3.1 que notre notion de confinement jouait un rôle similaire à la notion de monotonie [8]. Il serait intéressant d'observer si un lien uni ces deux notions, par exemple si l'une pourrait impliquer l'autre ; notons néanmoins que le confinement concerne un encodeur alors que la monotonie est la propriété d'un problème, ce qui constitue une différence majeure. Avec une telle relation, nous pourrions espérer que certains corollaires de Bodlaender, Fomin, Lokshtanov, Penninkx, Saurabh et Thilikos [8] soient facilement transférables à notre cadre.

Bibliographie

[1] I. Adler, F. Dorn, F. Fomin, I. Sau, and D. Thilikos. Faster parameterized algorithms for minor containment. *Theoretical Computer Science*, 412(50) :7018–7028, 2011.

[2] J. Alber, H. L. Bodlaender, H. Fernau, T. Kloks, and R. Niedermeier. Fixed parameter algorithms for dominating set and related problems on planar graphs. *Algorithmica*, 33(4) :461–493, 2002.

[3] J. Alber, M. Fellows, and R. Niedermeier. Polynomial-time data reduction for Dominating Set. *Journal of the ACM*, 51(3) :363–384, 2004.

[4] U. Bertelè and F. Brioschi. *Nonserial Dynamic Programming*. Academic Press, Inc., Orlando, FL, USA, 1972.

[5] A. Bishnu, A. Ghosh, and S. Paul. Parameterized complexity of k-tuple and liar's domination. *CoRR*, abs/1309.5461, 2013.

[6] H. Bodlaender. A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM Journal on Computing*, 25(6) :1305–1317, 1996.

[7] H. Bodlaender, R. Downey, M. Fellows, and D. Hermelin. On problems without polynomial kernels. *Journal of Computer and System Sciences*, 75(8) :423 – 434, 2009.

[8] H. Bodlaender, F. Fomin, D. Lokshtanov, E. Penninkx, S. Saurabh, and D. Thilikos. (meta) kernelization. In *Proc. of the 50th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 629–638. IEEE Computer Society, 2009.

[9] H. Bodlaender, B. Jansen, and S. Kratsch. Cross-composition : A new technique for kernelization lower bounds. In *Proc. of the 28th International Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 9 of *LIPIcs*, pages 165–176, 2011.

[10] A. Braga de Queiroz, V. Garnero, and P. Ochem. On interval representations of graphs. *Discrete Applied Mathematics*, 202 :30–36, 2016.

[11] C. Chekuri and J. Chuzhoy. Large-treewidth graph decompositions and applications. In *Proc. of the 45th ACM Symposium on the Theory of Computing (STOC)*, pages 291–300, 2013.

[12] J. Chen, H. Fernau, I. Kanj, and G. Xia. Parametric duality and kernelization : lower bounds and upper bounds on kernel size. *SIAM Journal on Computing*, 37(4) :1077–1106, 2007.

[13] S. Cook. The complexity of theorem-proving procedures. In *Proceedings of the 3rd Annual ACM Symposium on Theory of Computing, May 3-5, 1971, Shaker Heights, Ohio, USA*, pages 151–158, 1971.

[14] B. Courcelle. The monadic second-order logic of graphs. I. recognizable sets of finite graphs. *Information and Computation*, 85(1) :12–75, 1990.

[15] E. Demaine, F. Fomin, M. Hajiaghayi, and D. Thilikos. Fixed-parameter algorithms for (k, r)-center in planar graphs and map graphs. *ACM Transactions on Algorithms*, 1(1) :33–47, 2005.

[16] E. Demaine and M. Hajiaghayi. Linearity of grid minors in treewidth with applications through bidimensionality. *Combinatorica*, 28(1) :19–36, 2008.

[17] R. Diestel. *Graph Theory*, volume 31. Springer-Verlag Berlin Heidelberg, 2006.

[18] M. Dom, D. Lokshtanov, and S. Saurabh. Incompressibility through colors and ids. In S. Albers, A. Marchetti-Spaccamela, Y. Matias, S. Nikoletseas, and W. Thomas, editors, *Automata, Languages and Programming*, volume 5555 of *Lecture Notes in Computer Science*, pages 378–389. Springer Berlin Heidelberg, 2009.

[19] R. Downey and M. Fellows. *Parameterized Complexity*. Springer, 1999.

[20] D. Eppstein. Diameter and treewidth in minor-closed graph families. *Algorithmica*, 27 :275–291, 2000. Special issue on treewidth, graph minors, and algorithms.

[21] P. Erdős and L. Pósa. On independent circuits contained in a graph. *Canadian Journal of Mathematics*, 17 :347–352, 1965.

[22] Eukleídês. Stoïkheïa. VII :123–126, IIIs av. J.C.

[23] L. Eulero. Solutio problematis ad geometriam situs pertinentis. VIII :128–140, 1741.

[24] J. Flum and M. Grohe. *Parameterized Complexity Theory*. Texts in Theoretical Computer Science. Springer, 2006.

[25] F. Fomin, P. Golovach, and D. Thilikos. Contraction obstructions for treewidth. *Journal of Combinatorial Theory, Series B*, 101(5) :302–314, 2011.

[26] F. Fomin, D. Lokshtanov, V. Raman, and S. Saurabh. Bidimensionality and EPTAS. In *Proc. of the 22nd ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 748–759. SIAM, 2011.

[27] F. Fomin, D. Lokshtanov, S. Saurabh, and D. Thilikos. Bidimensionality and kernels. In *Proc. of the 21st ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 503–510. SIAM, 2010.

[28] F. Fomin, S. Saurabh, and D. Thilikos. Strengthening Erdős-Pósa property for minor-closed graph classes. *Journal of Graph Theory*, 66(3) :235–240, 2011.

[29] F. Fomin and D. Thilikos. Dominating sets in planar graphs : Branch-width and exponential speed-up. *SIAM Journal on Computing*, 36(2) :281–309, 2006.

[30] F. V. Fomin, D. Lokshtanov, S. Saurabh, and D. M. Thilikos. Linear kernels for (connected) dominating set on graphs with excluded topological subgraphs. *CoRR*, abs/1210.0257, 2012.

[31] F. V. Fomin, D. Lokshtanov, S. Saurabh, and D. M. Thilikos. Linear kernels for (connected) dominating set on H-minor-free graphs. In *Proc. of the 23rd ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 82–93. SIAM, 2012.

[32] V. Garnero, C. Paul, I. Sau, and D. Thilikos. Explicit linear kernels for packing problems. 2015. Manuscript submitted for conference.

[33] V. Garnero, C. Paul, I. Sau, and D. Thilikos. Explicit linear kernels via dynamic programming. In *Proc. of the 31st International Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 25 of *LIPIcs*, pages 312–324, 2015.

[34] V. Garnero and I. Sau. A linear kernel for planar total dominating set. Manuscript submitted for publication available on ArXiv.

[35] V. Garnero, I. Sau, and D. Thilikos. A linear kernel for planar red-blue dominating set. In *Proc. of the 12th Cologne Twente Workshop on Graphs and Combinatorial Optimization (CTW)*, pages 117–120, 2013.

[36] V. Garnero and M. Weller. Parameterized certificate dispersal and its variants. *Theorical Computer Science*, 622 :66–78, 2016.

[37] A. Giannopoulou, M. Kamiński, and D. M. Thilikos. Forbidding kuratowski graphs as immersions. *Journal of Graph Theory*, 78(1) :43–60, 2015.

[38] M. Grohe. *Logic, graphs, and algorithm.* Texts in Logic and Games. Amsterdam University Press, 2007.

[39] Q. Gu and N. Imani. Connectivity is not a limit for kernelization : Planar connected dominating set. In *Proc. of the 9th Latin American Symposium on Theoretical Informatics (LATIN)*, volume 6034 of *LNCS*, pages 26–37, 2010.

[40] J. Guo and R. Niedermeier. Linear problem kernels for NP-hard problems on planar graphs. In *Proc. of the 34th International Colloquium on Automata, Languages and Programming (ICALP)*, volume 4596 of *LNCS*, pages 375–386, 2007.

[41] R. Halin. S-functions for graphs. *Journal of Geometry*, 8(1) :171–186, 1976.

[42] K. Kawarabayashi and Y. Kobayashi. Linear min-max relation between the treewidth of H-minor-free graphs and its largest grid. In *Proc. of the 29th International Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 14 of *LIPIcs*, pages 278–289, 2012.

[43] E. Kim, A. Langer, C. Paul, F. Reidl, P. Rossmanith, I. Sau, and S. Sikdar. Linear kernels and single-exponential algorithms via protrusion decompositions. In *Proc. of the 40th International Colloquium on Automata, Languages and Programming (ICALP)*, volume 7965 of *LNCS*, pages 613–624, 2013.

[44] T. Kloks. *Treewidth. Computations and Approximations.* Springer-Verlag LNCS, 1994.

[45] S. Kreutzer. *Parameterized and Exact Computation : Third International Workshop, IWPEC 2008, Victoria, Canada, May 14-16, 2008. Proceedings*, chapter Algorithmic Meta-theorems, pages 10–12. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[46] K. Kuratowski. Sur le problème des courbes gauches en topologie. *Fundamenta Mathematicae*, 15(1) :271–283, 1930.

[47] D. Lokshtanov, M. Mnich, and S. Saurabh. A linear kernel for planar connected dominating set. *Theoretical Computer Science*, 23(412) :2536–2543, 2011.

[48] D. Lubell. A short proof of Sperner's lemma. *Journal of Combinatorial Theory*, 1(2) :299 –, 1966.

[49] R. Niedermeier. *Invitation to fixed parameter algorithms*, volume 173. Oxford University Press, 2010.

[50] N. Robertson and P. Seymour. Graph minors. II. Algorithmic aspects of tree-width. *Journal of Algorithms*, 7(3) :309 – 322, 1986.

[51] N. Robertson and P. Seymour. Graph minors. V. excluding a planar graph. *Journal of Combinatorial Theory, Series B*, 41(1) :92–114, 1986.

[52] N. Robertson and P. Seymour. Graph minors. XIII. The disjoint paths problem. *Journal of Combinatorial Theory, Series B*, 63(1) :65–110, 1995.

[53] N. Robertson and P. Seymour. Graph minors. XX. Wagner's conjecture. *Journal of Combinatorial Theory, Series B*, 92(2) :325 – 357, 2004.

[54] K. Wagner. Über eine eigenschaft der ebenen komplexe. *Mathematische Annalen*, 114(1) :570–590, 1937.

[55] J. Wang, Y. Yang, J. Guo, and J. Chen. Planar graph vertex partition for linear problem kernels. *Journal of Computer and System Sciences*, 79(5) :609 – 621, 2013.

[56] R. Watrigant. *Approximation et complexité paramétrée de problèmes d'optimisation dans les graphes : partitions et sous-graphes*. PhD thesis, Université de Montpellier, 2014.

[57] K. Weihe. Covering trains by stations or the power of data reduction. In *ALEX*, pages 1–8, 1998.

[58] J. Zhu. Approximation for minimum total dominating set. In *Proc. of the 2nd International Conference on Interaction Sciences*, volume 403 of *ACM International Conference Proceeding Series*, pages 119–124, 2009.

Résumé

Résumé

En algorithmique et en complexité, la plus grande part de la recherche se base sur l'hypothèse que $\mathsf{P} \neq \mathsf{NP}$ (les classes, appelées en anglais *Polynomial time* et *Non deterministic Polynomial time*, sont différentes), c'est-à-dire qu'il existe des problèmes dont la solution peut être vérifiée mais non construite en temps polynomial. Si cette hypothèse est admise, de nombreux problèmes naturels ne sont pas dans P (c'est-à-dire, n'admettent pas d'algorithme efficace), ce qui a conduit au développement de nombreuses branches de l'algorithmique. L'une d'elles est la complexité paramétrée. Elle propose des algorithmes exacts, dont l'analyse est faite en fonction de la taille de l'instance et d'un paramètre. Ce paramètre permet une granularité plus fine dans l'analyse de la complexité. Un algorithme sera alors considéré comme efficace s'il est à paramètre fixé, c'est-à-dire, lorsque sa complexité est exponentielle en fonction du paramètre et polynomiale en fonction de la taille de l'instance. Ces algorithmes résolvent les problèmes de la classe FPT (en anglais, *Fixed Parameter Tractable*).

L'extraction de noyaux est une technique qui permet, entre autre, d'élaborer des algorithmes à paramètre fixé. Elle peut être vue comme un pré-calcul de l'instance, avec une garantie sur la compression des données. Plus formellement, une extraction de noyau est une réduction polynomiale depuis un problème vers lui même, avec la contrainte supplémentaire que la taille du noyau (l'instance réduite) est bornée en fonction du paramètre. Pour obtenir l'algorithme à paramètre fixé, il suffit de résoudre le problème dans le noyau, par exemple par une recherche exhaustive (de complexité exponentielle, en fonction du paramètre). L'existence d'un noyau implique donc l'existence d'un algorithme à paramètre fixé, la réciproque est également vraie. Cependant, l'existence d'un algorithme à paramètre fixé efficace ne garantit pas un petit noyau, c'est a dire un noyau dont la taille est linéaire ou polynomiale. Sous certaines hypothèses, il existe des problèmes n'admettant pas de noyau (c'est-à-dire hors de FPT) et il existe des problèmes de FPT n'admettant pas de noyaux polynomiaux.

Un résultat majeur dans le domaine des noyaux est la construction d'un noyau linéaire pour le problème Domination dans les graphes planaires, par Alber, Fellows et Niedermeier.

Tout d'abord, la méthode de décomposition en régions proposée par Al-

ber, Fellows et Niedermeier, a permis de construire de nombreux noyaux pour des variantes de Domination dans les graphes planaires. Cependant cette méthode comportait un certain nombre d'imprécisions, ce qui rendait les preuves invalides. Dans la première partie de notre thèse, nous présentons cette méthode sous une forme plus rigoureuse et nous l'illustrons par deux problèmes : Domination Rouge-Bleu et Domination Totale.

Ensuite, la méthode a été généralisée, d'une part, sur des classes de graphes plus larges (de genre borné, sans-mineur, sans-mineur-topologique), d'autre part, pour une plus grande variété de problèmes. Ces méta-résultats prouvent l'existence de noyaux linéaires ou polynomiaux pour tout problème vérifiant certaines conditions génériques, sur une classe de graphes peu denses. Cependant, pour atteindre une telle généralité, il a fallu sacrifier la constructivité des preuves : les preuves ne fournissent pas d'algorithme d'extraction constructif et la borne sur le noyau n'est pas explicite. Dans la seconde partie de notre thèse nous effectuons un premier pas vers des méta-résultats constructifs ; nous proposons un cadre général pour construire des noyaux linéaires en nous inspirant des principes de la programmation dynamique et d'un méta-résultat de Bodlaender, Fomin, Lokshtanov, Penninkx, Saurabh et Thilikos.

Summary

In the fields of Algorithmic and Complexity, a large area of research is based on the assumption that $\mathsf{P} \neq \mathsf{NP}$ (the classes*Polynomial time* and *Non deterministic Polynomial time* are different), which means that there are problems for which a solution can be verified but not constructed in polynomial time. Many natural problems are not in P, which means that they have no efficient algorithm. In order to tackle such problems, many different branches of Algorithmics have been developed. One of them is called Parametric Complexity. It consists of developing exact algorithms whose complexity is measured as a function of the size of the instance and of a parameter. Such a parameter allows a more precise analysis of the complexity. In this context, an algorithm will be considered to be efficient if it is fixed parameter tractable (fpt), that is, if it has a complexity which exponentially depends on the parameter and polynomially depends on the size of the instance. Problems that can be solved by such an algorithm form the FPT class.

Kernelisation is one technique that produces fpt algorithms, among others. It can be viewed as a preprocessing of the instance, with a guarantee on the compression of the data. More formally, a kernelisation is a polynomial reduction from a problem to itself, with the additional constraint that the size of the kernel, the reduced instance, is bounded by a function of the parameter. In order to obtain an fpt algorithm, it is sufficient to solve the problem in the reduced instance, by brute-force for example (which has exponential complexity, in the parameter). Hence, the existence of a kernelisiation implies the existence of an fpt algorithm. It holds that the converse is true also. Nevertheless, the existence of an efficient fpt algorithm does not imply a small kernel, meaning a kernel with a linear or polynomial size. Under certain hypotheses, it can be proved that some problems can not have a kernel (that is, are not in FPT) and that some problems in FPT do not have a polynomial kernel.

One of the main results in the field of Kernelisation is the construction of a linear kernel for the Dominating Set problem on planar graphs, by Alber, Fellows and Niedermeier.

To begin with, the region decomposition method proposed by Alber, Fellows and Niedermeier has been reused many times to develop kernels for variants of Dominating Set on planar graphs. Nevertheless, this method had quite a few inaccuracies, which has invalidated the proofs. In the first part of our thesis, we present a more thorough version of this method and we illustrate it with two examples : Red Blue Dominating Set and Total

Dominating Set.

Next, the method has been generalised to larger classes of graphs (bounded genus, minor-free, topological-minor-free), and to larger families of problems. These meta-results prove the existence of a linear or polynomial kernel for all problems verifying some generic conditions, on a class of sparse graphs. As a price of generality, the proofs do not provide constructive algorithms and the bound on the size of the kernel is not explicit. In the second part of our thesis, we make a first step to constructive meta-results. We propose a framework to build linear kernels based on principles of dynamic programming and a meta-result of Bodlaender, Fomin, Lokshtanov, Penninkx, Saurabh and Thilikos