Connexité dans les Réseaux et Schémas d'Étiquetage Compact d'Urgence

Pierre Halftermeyer

université de BORDEAUX

edmi BORDEAUX
ÉCOLE DOCTORALE DE MATHÉMATIQUES ET INFORMATIQUE

Thèse de doctorat

Discipline : informatique

présentée par

Pierre Halftermeyer

Connexité dans les Réseaux et Schémas d'Étiquetage Compact d'Urgence

Soutenue le 22 *septembre* 2014
devant le jury composé de :

Président :		
M. Robert Cori	Professeur	LaBRI, Université de Bordeaux
Rapporteurs :		
M. Patrice Ossona de Mendez	Chargé de recherche	CNRS
M. Laurent Viennot	Directeur de recherche	INRIA
Examinateur :		
M. Ioan Todinca	Professeur	LIFO, Université d'Orléans
Co-directeurs :		
M. Bruno Courcelle	Professeur émérite	LaBRI, Université de Bordeaux et IUF
M. Cyril Gavoille	Professeur	LaBRI, Université de Bordeaux et IUF

au Laboratoire Bordelais de Recherche en Informatique
351, cours de la Libération 33405 Talence

Résumé

L'objectif de cette thèse est d'attribuer à chaque sommet x d'un graphe G à n sommets une étiquette $L(x)$ de taille compacte $O(\log n)$ bits afin de pouvoir :

1. construire, à partir des étiquettes d'un ensemble de sommets en panne $X \subseteq V(G)$, une structure de donnée $S(X)$
2. décider, à partir de $S(X)$ et des étiquettes $L(u)$ et $L(v)$, si les sommets u et v sont connectés dans le graphe $G \setminus X$.

Nous proposons une solution à ce problème pour la famille des graphes 3-connexes de genre g (*via* plusieurs résultats intermédiaires).

— Les étiquettes sont de taille $O(g \log n)$ bits
— Le temps de construction de la structure de donnée $S(X)$ est $O(\textsc{Sort}(|X|, n))$.
— Le temps de décision est $O(\log \log n)$. Ce temps est optimal.

Nous étendons ce resultat à la famille des graphes excluant un mineur H fixé. Les étiquettes sont ici de taille $O(\text{polylog}\, n)$ bits.

Mots-clef : graphes sur les surfaces, mineur de graphe, connexité avec panne, schéma d'étiquetage, planification de l'urgence.

Abstract

We aim at assigning each vertex x of a n-vertices graph G a compact $O(\log n)$-bit label $L(x)$ in order to :

1. construct, from the labels of the vertices of a forbidden set $X \subseteq V(G)$, a data-structure $S(X)$
2. decide, from $S(X)$, $L(u)$ and $L(v)$, whether two vertices u and v are connected in $G \setminus X$.

We give a solution to this problem for the family of 3-connected graphs with bounded genus.

— We obtain $O(g \log n)$-bit labels.
— $S(X)$ is computed in $O(\textsc{Sort}(|X|, n))$ time.
— Connection between vertices is decided in $O(\log \log n)$ optimal time.

We finally extend this result to H-minor-free graphs. This scheme requires $O(\text{polylog}\, n)$-bit labels.

Keywords : graphs on surfaces, graph minor, forbidden-set connectity, labeling scheme, emergency planning.

Remerciements

Je tiens à remercier en premier lieu MM. Bruno Courcelle et Cyril Gavoille, qui m'ont fait l'honneur de diriger cette thèse et qui auront su incarner pour moi de brillants modèles de rigueur, de persévérance, d'honnêteté et d'enthousiasme.

Je remercie les rapporteurs de cette thèse, MM. Patrice Ossona de Mendez et Laurent Viennot pour le temps consacré à la lecture estivale du présent manuscrit.

Je remercie M. Ioan Todinca, membre du jury, et M. Robert Cori qui m'a fait l'honneur d'y présider.

Je remercie tous les personnels académiques du LaBRI. En particulier ceux qui m'ont prêté main forte, ceux avec qui j'ai enseigné, et ceux qui m'ont fait part de leurs idées quant à mes recherches. Je pense en particulier à Nicolas Bonichon, Pierre Castéran, Géraud Sénizergues, Anne Dicky, Adrien Boussicault, Nicolas Hanusse, David Ilcinkas, Sylvain Salvati, Lionel Clément, Christian Rétoré, Fabien Baldacci, Pascal Desbarats, André Raspaud, Paul Dorbec, et bien d'autres.

Je remercie l'équipe *Combinatoire et Algorithmes* et l'équipe *Méthodes Formelles* du LaBRI, les différents intervenants des réunions de l'ANR *Displexity*, ainsi que toutes les personnes que j'ai eu la chance de rencontrer dans le cadre de conférences.

Je remercie l'ensemble des personnes qui font tourner le LaBRI, de l'équipe technique à l'équipe administrative en passant par le personnel d'entretien.

Je remercie mes étudiants. Surtout ceux des deux premiers rangs.

Je remercie Jean-Pierre Petit de m'avoir autorisé à utiliser quelques unes de ses planches.

Je remercie Pierre-Henri Jondot pour quelques mots qui m'ont motivé à reprendre mes études.

Je remercie la Formation Continue de l'Université de Bordeaux, en particulier Patrick Loubet et Paola Barbosa, l'ASP (ex-CNASEA), la Mairie de Saint-Médard en Jalles, en particulier Isabelle Heisel, et *via* la Caisse d'Allocation Familiale, mon pays, la France, et son modèle social. Ils ont rendu cette reprise d'études possible.

Je remercie les *Kangourous* de Pessac.

Je remercie l'ensemble des doctorants du LaBRI avec qui j'ai eu la chance de partager bureaux, repas, cafés et discussions. En particulier : Eve Garnaud, Allyx Fontaine, Razanne Issa, Lorijn van Rooijen, Samah Bouzidi, Emilie Diot, Noémie-Fleur Sandillon-Reizer, Yi Ren, Clément Charpentier, Thomas Morsellino, Gabriel Renault, Christian Glacet, Vincent Autefage, le bureau 123, la *diaspora tunisienne*, l'AFODIB, *etc.*

Je remercie mes camarades et toutes les personnes de bonne volonté.

Je remercie mes amis de Bordeaux, de Madrid, d'ailleurs et de nulle part qui se reconnaîtront.

Je remercie mes parents, ma soeur, mon frère ainsi que toute ma famille sans qui je serais quelqu'un d'autre.

Je remercie Anaïs pour la sérénité qu'elle m'apporte.

Table des matières

Introduction aux travaux

Quelques définitions

Représentation implicite. Il existe une multitude de structures de données permettant de représenter un graphe. On pourrait, par exemple, utiliser une matrice ou une collection de listes d'adjacence. Le type de représentation pertinent pour nos travaux découle de la notion de *représentation implicite*.

R est une *représentation implicite* d'un graphe G [55] si elle est l'assignation à chaque sommet x de G, d'une étiquette $L(x)$, aussi petite que possible [1], telle qu'il existe un algorithme décidant de l'adjacence entre les sommets x et y à partir des seules entrées $L(x)$ et $L(y)$.

Par exemple, si l'on considère un arbre $T \in \mathcal{T}$ ($\mathcal{T}$ étant la classe des arbres.) à n sommets, on obtient une représentation implicite de ce graphe en étiquetant chaque sommet u de T par :

$$L(u) = i(u),\ i\big(p(u)\big)$$

où la fonction p associe à u son père (ou $\perp$ si u est racine) dans l'arbre considéré enraciné, et la fonction i associe à chaque sommet un identifiant unique compris entre 1 et n qui peut être représenté par une chaîne binaire de $\lceil \log n \rceil$ bits.

Il s'agit bien d'une représentation implicite car, données les étiquettes de deux sommets, on peut décider en temps constant de leur adjacence en testant si l'un est le père de l'autre. Les étiquettes sont de taille $2 \cdot \lceil \log n \rceil$

1. Idéalement de taille $O(\log n)$ bits.

bits.

On utilisera l'adjectif *compact* pour qualifier une représentation implicite, ou plus tard toute forme d'étiquetage, dont les étiquettes comptent $O(\log n)$ bits. On peut remarquer, par comptage, qu'il n'existe pas de manière compacte de représenter implicitement tous les graphes. En effet, le nombre de graphe à n sommets et m arêtes est :

$$N(n,m) = \binom{\binom{n}{2}}{m} = 2^{O(m \log(\frac{n^2}{m}))}$$

donc si $d = \frac{2n}{m}$ est le degré moyen du graphe :

$$N(n,m) = 2^{O(nd \log(\frac{n}{d}))}$$

Il faut donc coder les représentations de ces $N(n,m)$ graphes sur $O(n.d.\log(\frac{n}{d}))$ bits d'information [2]. Il est alors inévitable d'avoir au moins une étiquette de $O(d \log \frac{n}{d})$ bits pour l'un des graphes.

Etiquetage. Il est naturel de dériver de cette notion de représentation implicite, celle qui sera centrale dans cette thèse, d'*étiquetage* des sommets. Considérons

$$P(u_1, \cdots, u_m, X_1, \cdots X_l)$$

une propriété sur les graphes de la famille $\mathcal{F}$, dépendant des sommets $u_1, \cdots, u_m$ et des ensembles de sommets $X_1, \cdots, X_l$. Considérons également une fonction $t : \mathcal{F} \to \mathbb{N}$. On appelle alors *$t$-étiquetage* de la propriété P sur le graphe G à n sommets, l'assignation à tout sommet u de G d'une étiquette de taille au plus $t(G)$ de telle manière que l'on puisse décider, pour tous $u_1, \cdots, u_m \in V(G)$ et pour tous $X_1, \cdots, X_l \subseteq V(G)$, si la propriété $P(u_1, \cdots, u_m, X_1, \cdots X_l)$ est ou non vérifiée dans le graphe G, et ce en n'utilisant que l'information procédant des étiquette des sommets de $\{u_1, \cdots, u_m\} \cup X_1 \cup \cdots \cup X_l$.

On peut imaginer, par exemple, un étiquetage compact pour la propriété $P_1(x,y,z)$ décidant si, oui ou non, il existe un chemin de x à y, ne passant pas

2. Sans quoi on aura au moins une paire de graphes différents représentés par la même chaîne binaire, ce qui est impossible.

par le sommet z. Ou encore un autre étiquetage pour la propriété $P_2(x, y, X)$, décidant si oui ou non il existe un chemin de x à y n'empreintant que des sommets de X.

Cette notion se généralise, de manière naturelle, à l'étiquetage de sommets *et* d'arêtes. On peut ainsi traiter des propriétés telles que $P_3(x, y, X, Y)$, la propriété d'existence d'un chemin de x à y n'empreintant que des sommets de X et des arêtes de Y.

Un étiquetage peut aussi être destiné au calcul d'une valeur plutôt qu'à la decision d'une propriété. Par exemple, la fonction de distance entre deux sommets $d : V \times V \to \mathbb{N}$ peut donner lieu à un étiquetage. L'algorithme caculera la distance entre deux sommets x et y à partir de leurs étiquettes $L(x)$ et $L(y)$.

On remarquera aussi qu'un étiquetage des sommets pour la propriété d'adjacence $A(x, y)$ est équivalent à une représentation implicite du graphe. L'exemple de représentation implicite donné plus haut pour la famille des arbres est un t-étiquetage pour la propriété $A(x, y)$ avec $t : G \in \mathcal{T} \to 2 \cdot \lceil \log n(G) \rceil$ où $n(G)$ est le nombre de sommet du graphe G. On dira aussi plus simplement qu'on a un $O(\log n)$-étiquetage.

Schéma d'étiquetage. L'étude des performances des étiquetages nous amène à définir un paradigme d'analyse. Nous introduisons donc la notion de *schéma d'étiquetage*.

Pour une classe de graphes $\mathcal{C}$, et pour une propriété [3] $P(z_1, \cdots, z_m, Z_1, \cdots Z_l)$ sur des objets [4] et ensembles d'objets étiquetés du graphe, on définit un *schéma d'étiquetage* des objets de l'ensemble W, comme étant un couple $(\mathcal{L}, \mathcal{Q})$ où :

3. Ou une fonction. La définition s'adapte trivialement.

4. Sommets ou arêtes. On note W l'ensemble des objets étiquetés. Par exemple, $W = V$ dans le cas d'un étiquetage des sommets. $W = V \cup E$ dans le cas d'un t-étiquetage des sommets et des arêtes.

— $\mathcal{L}$ est l'*algorithme d'étiquetage* prenant en entrée un graphe $G \in \mathcal{C}$ et calculant pour tout $z \in W$ une étiquette $L_G(z)$ de taille au plus $t(G)$ bits.

— $\mathcal{Q}$ est l'*algorithme de requête*, il prend en entrée une chaîne binaire

$$b = L_G(z_1), \cdots, L_G(z_m), L_G(Z_1), \cdots, L_G(Z_l)$$

avec

$$L_G(Z_i) = L_G(z_{(i,1)}), \cdots L_G(z_{(i,|Z_i|)})$$

où $z_{(i,j)}$ est le j-ème objet de Z_i.

$\mathcal{Q}$ n'est nourri d'aucune autre information sur le graphe G que de son appartenance à $\mathcal{C}$ et de la chaîne b. En sortie, $\mathcal{Q}$ renvoit la valeur :

$$\mathcal{Q}(b) = P(z_1, \cdots, z_m, Z_1, \cdots Z_l)$$

On dit alors que $(\mathcal{L}, \mathcal{Q})$ est un *schéma d'étiquetage* de la propriété P sur la classe $\mathcal{C}$. Il est de *compacité* t. On dit que $(\mathcal{L}, \mathcal{Q})$ est un schéma d'étiquetage *compact* si $t = O(\log n)$, n étant la grandeur désignant la complexité combinatoire du graphe, par défaut de précision, son nombre de sommet.

Étiquetage *versus* oracle

Un *oracle* est une structure de données pré-calculée permettant de répondre efficacement à des requêtes. Un étiquetage est donc un cas particulier d'oracle où les données sont distribuées entre les sommets.

Un schéma d'étiquetage des sommets où les étiquettes sont de compacité t implique donc naturellement, par simple concaténation de toutes les étiquettes, un oracle de taille $S = n \cdot t$. En revanche, un oracle de taille S ne peut être considéré comme un étiquetage que si l'on peut le distribuer équitablement entre ses sommets, c'est-à-dire sous forme d'étiquettes de taille S/n, ce qui n'est pas possible dans le cas général.

L'intérêt de l'étiquetage saute aux yeux dès lors que l'on considère des applications distribuées, et *a fortiori* embarquées, impliquant une contrainte de stockage et de téléchargement minimal d'information. Cela dit, ce n'est pas leur seul atout. L'approche distribuée permet de ne donner qu'une vue partielle du réseau à un agent, ce qui peut être attrayant pour la protection des données sensibles et de la sécurité. De plus, certains résultats obtenus par schéma d'étiquetage sont *in se* compétitifs face aux approches centralisées. On citera [4] et [13]. En dernier lieu, on notera que de bons algorithmes dynamiques peuvent dériver de schémas d'étiquetages dont les temps de mise-à-jour sont performants.

On pourra donc comparer les performances de nos schémas d'étiquetage aux oracles de l'existant, mais sans perdre de vue le fait que le caractère distribué de notre modèle constitue :

— d'une part un atout pour l'utilisateur ;
— et de l'autre une contrainte forte pour la conception.

Connexité et Pannes - Définitions

La *connexité* est un des concepts basiques de la théorie des graphes. Soit G un graphe. On dit que ses sommets u et v sont *connectés* s'il existe une *chaîne*, c'est à dire une suite d'arêtes adjacentes du graphe, ayant pour extrémités u et v.

On peut définir le problème Connexité ainsi :

Problème 1 : Connexité
Entrée: $G = (V, E)$ un graphe, $u, v \in V$
Sortie: Conn(u, v) la valeur du prédicat « u et v sont dans la même composante connexe de G »

Si pour toute paire de sommets u et v de G, u et v sont connectés, on dit que le graphe G est un graphe *connexe*.

G n'étant pas forcément un graphe connexe, il est naturel de vouloir le partitionner en ses sous-graphes connexes maximaux, appelés les *composantes connexes* de G. Un graphe ayant une et seulement une composante connexe

est donc un graphe connexe.

On appelle *coupe* (*cut-set* en anglais) d'un graphe connexe G, un sous ensemble C de $V(G)$ tel que le graphe induit par $V(G) \setminus C$ n'est pas connexe.

On dit que G est *k-connexe* s'il n'existe pas de coupe C de moins de k sommets. C'est à dire qu'il faut au moins supprimer k sommets si l'on veut déconnecter le graphe G.

On s'intéressera dans ce document au problème de Connexité Avec Panne (Forbidden Set Connectivity en anglais). Ce problème prend en entrée un graphe G, deux de ses sommets u et v, un ensemble de ses sommets $X \subseteq V(G)$ et un ensemble d'arêtes $Y \subseteq E(G)$. Il s'agit de décider si, oui ou non, les sommets u et v sont connectés dans le sous-graphe de G dont l'ensemble de sommets est $V(G) \setminus X$ et dont l'ensemble d'arêtes est $E(G) \setminus Y$ (que l'on se permettra de noter $G \setminus (X \cup Y)$).

Problème 2 : Connexité Avec Panne
Entrée: $G = (V, E)$ un graphe, $u, v \in V$, $X \subseteq V$, $Y \subseteq E$
Sortie: $\textsc{Conn}(u, v, X, Y)$ la valeur du prédicat « u et v sont dans la même composante connexe de $G \setminus (X \cup Y)$ »

On considèrera tout particulièrement la version de ce problème où $Y = \emptyset$, que l'on nommera Connexité Avec Panne sur les Sommets ou abusivement Connexité Avec Panne lorsqu'il n'y aura pas d'ambiguïté.

Problème 3 : Connexité Avec Panne sur les Sommets
Entrée: $G = (V, E)$ un graphe, $u, v \in V$, $X \subseteq V$
Sortie: $\textsc{Conn}(u, v, X)$ la valeur du prédicat « u et v sont dans la même composante connexe de $G \setminus X$ »

$\textsc{Conn}(u, v, X, Y)$ (ou $\textsc{Conn}(u, v, X)$) est donc la propriété définie comme étant vraie si et seulement si u et v sont connectés dans le graphe en panne.

On considèrera parfois le problème Composante Connexe Avec Panne. $\textsc{CompConn}(u, X, Y)$ (respectivement $\textsc{CompConn}(u, X)$) désigne un identifiant de la composante connexe de u dans le graphe privé de l'ensemble

interdit. On peut remarquer que le problème Connexité Avec Panne se réduit à Composante Connexe Avec Panne : comparer les identifiants des composantes connexes de u et de v permet de décider de leur état de connexion. La réduction inverse n'est pas possible. Le problème Composante Connexe Avec Panne est donc un problème plus difficile.

Problème 4 : Composante Connexe Avec Panne
Entrée: $G = (V, E)$ un graphe, $u \in V$, $X \subseteq V$, $Y \subseteq E$
Sortie: CompConn(u, X, Y) un identifiant de la composante connexe de u dans $G \setminus (X \cup Y)$

Problème 5 : Composante Connexe Avec Panne Sur les Sommets
Entrée: $G = (V, E)$ un graphe, $u \in V$, $X \subseteq V$
Sortie: CompConn(u, X) un identifiant de la composante connexe de u dans $G \setminus X$

Étiquetage et connexité

On traitera ici, comme annoncé précédemment, de schémas d'étiquetage de graphes pour la propriété Conn(u, v, X). C'est-à-dire que nous tâcherons, pour une famille de graphes donnée, d'étiqueter les sommets en visant le critère de compacité, et ce de façon à pouvoir décider de la propriété Conn(u, v, X).

Nous nous intéresserons, au delà de la taille des structures de données, aux complexités temporelles des différentes étapes de calcul, à savoir l'étiquetage et la requête. Il conviendra d'étiqueter les sommets des graphes considérés en un temps raisonnable, polynomial en n, le nombre de sommet du graphe (on visera même un temps proportionnel à cette grandeur) et de répondre aux requête dans un temps ultra-rapide[5]. Nous introduirons plus loin le concept de *connexité dans l'urgence* qui nous permettra d'affiner l'analyse du temps de requête.

5. Le temps proportionnel à la taille de la requête est l'idéal.

Paradigme d'analyse

Dans tous les secteurs d'activité, il est nécessaire de réagir face à une situation de crise. Et si possible, de réagir vite et juste. Métaphoriquement, il convient d'*éteindre le feu* de la plus efficace des manières. La sagesse populaire et le dicton *mieux vaut prévenir que guérir* s'appliquent ici.

Le monde de l'entreprise parle de *processus de crise*, ou plutôt, puisque la langue anglaise a pignon sur cette rue, de *process de crise* ou encore d'*emergency management*. Une entreprise ou une formation politique doit savoir réagir vite et de manière coordonnée à un *tweet* la mettant en cause, à un fait divers pouvant influer sur son image, à une hausse brutale du cours d'une de ses matières premières ou à une cyber-attaque. D'où l'émergence du métier de *gestionnaire de communauté* (*Online community manager* en Anglais) et le renforcement des services de veille et de sécurité informatique [6]. Il faudra donc *prévoir l'urgence* et être *proactif*. On pourra citer les plans ORSEC en France, ou le *Disaster Mitigation Act* aux Etats-Unis comme plans d'urgence de grande envergure.

Rapprochons-nous du cœur de notre sujet et considérons un réseau électrique, d'importance critique, subissant les dommages d'une tempête.
L'administrateur de ce réseau porte une double-responsabilité :

— *primo*, il se doit de réparer les dommage au plus vite
— *secundo*, il lui faut maintenir le système opérationnel dans la plus large des mesures.

« $\textsc{Conn}(u, v, X)$ » ? Quoi de plus essentiel que de pouvoir s'assurer que les points du réseau qui n'ont pas été mises hors-service par l'attaque malveillante soient toujours interconnectées ? Dans une telle situation, *a fortiori* si le réseau est grand, on sera satisfait d'avoir engagé le technicien f (f comme *fourmi* [19]) ayant *planifié l'urgence* plutôt que son homologue c (c comme *cigale*) qui s'offrira le luxe de nombreuses et gourmandes requêtes de connexité en temps $O(n + m)$ alors que la moitié de la ville est plongé dans le noir [6].

Notre cher f, au prix d'une petite remise à jour périodique de sa struc-

6. Ce que d'ailleurs il ignore.

ture de données (lancée pendant la pause-déjeuner de chaque vendredi, par exemple), pourra s'offrir, selon la topologie de son réseau, des *turbo-requêtes* $\textsc{Conn}(u, v, X)$ (en temps quasi-constant, logarithmique ou sub-logarithmique), après un seul pré-calcul de la panne en temps quasi-linéaire en $|X|$, par exemple $O(|X| \cdot \log\log n)$. Le travail régulier de f et son état de veille technologique assidu lui permettront de ne pas se trouver *fort dépourvu*[7] une fois *la bise venue*

Nous nous mettons donc tout naturellement au service de notre f dans le cadre très spécifique du problème Connexité Avec Panne. Nous tâcherons donc de mettre au point pour lui des structures de données satisfaisant les exigences du paradigme d'analyse de connexité *dans l'urgence*. C'est ainsi qu'il nous faudra raffiner, comme évoqué juste avant, la notion de temps de requête communément utilisée dans les travaux d'étiquetage compact ou quasi-compact.

Temps de pré-requête et temps de turbo-requête *versus* temps de requête

Un schéma d'étiquetage $(\mathcal{L}, \mathcal{Q})$ est-il bon ? Les critères habituels permettant d'aborder cette question sont, comme nous l'avons vu précédemment, d'une part, la *compacité* des étiquettes t, et de l'autre, les différents temps de calcul : Le *temps d'étiquetage* et le *temps de requête*.

— Le temps d'étiquetage $T_{\mathcal{L}}(G)$ est le temps mis par l'algorithme d'étiquetage $\mathcal{L}$ pour assigner une étiquette à chacun des sommets du graphe G donné en entrée. On étudiera plus particulièrement la valeur asymptotique $T_{\mathcal{L}}(n)$ de ce temps en prenant le nombre de sommets n du graphe d'entrée comme paramètre. On ne s'interdit pas d'introduire d'autres paramètres.

— Le temps de requête $T_{\mathcal{Q}}(b)$ est le temps nécessaire à l'algorithme $\mathcal{Q}$ pour traiter une chaîne binaire de requête b. On s'intéresse souvent dans le cas particulier de notre problème au temps de requête

7. Au chômage.

$T_{\mathcal{Q}}(n, |X|)$ selon les deux paramètres que sont le nombre de sommet du graphe en panne et la taille de la panne en nombre de sommets.

L'approche nouvelle, ici présentée, consiste en une sorte de *curryfication*[8] de l'algorithme de requête. En effet, une requête $\textsc{Conn}(u, v, X)$ peut être vue comme deux étapes successives :

— une *pré-requête* $\textsc{Conn}(X)$ retournant une fonction ;
— puis l'application de cette fonction de *turbo-requête* aux sommets u et v : $\textsc{Conn}(X)(u, v)$

On notera plus élégament

— $\textsc{PreConn}(X)$ la première étape de la requête ;
— et $\textsc{Conn}_X(u, v)$ la seconde.

Cette granularisation semble naturelle dès lors que l'on imagine une situation de panne unique où de nombreuses requêtes sont nécessaires : il faut essayer de factoriser le traitement de la panne (et son analyse) afin de n'y avoir recours qu'une seule fois, plutôt que de l'intégrer naïvement au temps de calcul de chacune des requêtes.
Plutôt que : $\textsc{Conn}(X, u_1, v_1)$, $\textsc{Conn}(X, u_2, v_2)$, $\textsc{Conn}(X, u_3, v_3)$, *et cætera*.
On fera : $\textsc{PreConn}(X)$, $\textsc{Conn}_X(u_1, v_1)$, $\textsc{Conn}_X(u_2, v_2)$, $\textsc{Conn}_X(u_2, v_2)$, *et cætera*.

C'est donc dans ce paradigme dit de *connexité dans l'urgence* que nous travaillerons. On évaluera les performances en temps de nos schémas d'étiquetage (désormais désignés par des triplets $(\mathcal{L}, \mathcal{P}, \mathcal{Q})$ où $\mathcal{P}$ est l'algorithme de pré-traitement d'une panne X et $\mathcal{Q}$ celui de turbo-requête nécessitant l'utilisation d'une structure de donnée livrée par $\mathcal{P}$.) en fonction :

— du temps d'étiquetage $T_{\mathcal{L}}(G)$
— du temps de pré-requête $T_{\mathcal{P}}(b)$
— du temps de turbo-requête $T_{\mathcal{Q}}(T_{\mathcal{P}}(b), b)$

8. La curryfication est l'opération qui fait passer d'une fonction à plusieurs arguments à une fonction à un argument qui retourne une nouvelle fonction prenant le reste des arguments.

État de l'art

Schémas d'étiquetage

De nombreux schémas d'étiquetages de graphes ont été mis au point et nourrissent la littérature. De l'un à l'autre varient d'une part le *type d'informations* glanées par les requêtes et de l'autre les *classes de graphes* sur lesquelles ils sont valides. Nous présentons ici un survol bref et non-exhaustif de ce qui est réalisable dans le domaine.

Adjacence. Il est montré dans [37] et rappelé plus haut que la classe des arbres admet une représentation implicite dont les étiquettes comptent $2 \log n$ bits. On désigne un sommet-racine, puis on donne à chaque sommet un numéro d'identifiant entre 1 et n. L'étiquetage consiste alors à attribuer à chaque sommet une étiquette contenant son identifiant ainsi que celui de son père dans l'arbre (ou 0 pour la racine). L'algorithme de requête peut alors décider de l'adjacence. Deux sommets sont adjacents si l'un est le père de l'autre, c'est-à-dire si l'un figure dans l'étiquette de l'autre, ce qui est vérifiable en temps constant.

Une représentation équivalente est proposé [55] pour la classe des graphes d'intervalles. Chaque sommet reçoit pour étiquette deux entiers compris entre 1 et $2n$: les bornes de son intervalle. L'adjacence est testée en temps constant, deux sommets sont adjacent si l'une des bornes de l'un est dans l'intervalle de l'autre.

De la représentation implicite des arbres découle, toujours dans [37], des schémas d'étiquetage compact d'adjacence pour les graphes d'arboricité bornés. On trouve aussi dans [17], entre autres, un schéma compact pour les graphe de largeur de clique bornée (classe incluant les graphe de largeur arborescente bornée).

[30] améliore ces résultats pour les graphes planaires et les classes de graphes de largeurs arborescentes bornées.

Distance. [48] montre que les arbres dont les arêtes sont pondérées (poids sur m bits) admettent un schéma d'étiquetage de distance dont les étiquettes comptent $O(m \log n + \log^2 n)$. Ce résultat est prouvé optimal dans [31]. On

y montre aussi que les graphes de genre borné admettent des schémas en $O(\sqrt{n})$.

Plus proche ancêtre commun. Un schéma compact pour le problème Ancêtre [3] dont les étiquettes sont de compacité $\log_2 n + (\sqrt{\log n})$ décide en temps constant depuis leurs étiquettes si de deux sommets u et v d'un arbre, l'un est ancêtre de l'autre.

Routage. Dans [25] et [57], sont proposés des schéma d'étiquetage compacts des arbres permettant des requêtes pour le problème Routage. Il calculent en temps constant, à partir d'étiquettes de tailles $O(\log n)$ de deux sommets u et v d'un arbre, un identifiant du port à empreinter depuis u pour router un message de u à v.

Connexité Sans Panne

Savoir si deux sommets u et v d'un graphe quelconque G, sans panne, sont dans la même composante connexe est une question naturelle de la théorie des graphes. Elle peut être trivialement traitée en temps $O(n+m)$ en explorant la composante connexe de u, de proche en proche, jusqu'à rencontrer v (u et v sont donc connectés, on renvoie $\top$) ou jusqu'à épuisement de cette composante (on répond $\bot$). L'algorithme 1, basé sur un algorithme classique de parcours de graphe, constitue une esquisse d'implémentation de cette méthode.

On propose un schéma d'étiquetage d'urgence trivial de la classe des graphes généraux pour ce problème :

Proposition 1. *Il existe un schéma d'étiquetage d'urgence des graphes pour le problème* Connexité Sans Panne.

- *Les étiquettes sont de compacité* $\lceil \log n \rceil$.
- *Le temps d'étiquetage est* $O(n+m)$.
- *Le temps de pré-requête est nul.*
- *Le temps de turbo-requête est* $O(1)$.

Démonstration. On étiquète le graphe G en temps $O(n+m)$ comme le fait [9] l'algorithme 2. Il attribue à chacun des sommets un numéro identifiant

9. Même parcours classique que l'algorithme précédent

```
1: fonction Conn(G, u, v)
2:     file f ← FILE-VIDE
3:     démarquer tous les sommets de G
4:     enfiler u dans f
5:     marquer u
6:     tant que f non vide faire
7:         si sommet en tête de f = v alors retourner ⊤
8:         fin si
9:         pour tout x voisin du sommet en tête de f non marqué faire
10:            marquer x
11:            enfiler x dans f
12:        fin pour
13:    fin tant que
14:    défiler f
15:    retourner ⊥
16: fin fonction
```

Algorithme 1: Connexité Sans Panne

sa composante connexe. Les étiquettes sont donc de taille compacte $\log k$ où $k \leq n$ est le nombre de composantes connexes du graphe. On teste la connexion entre deux sommets u et v en vérifiant que leurs étiquettes $L(u)$ et $L(v)$ sont identiques.

□

Connexité Avec Panne

Appréhender l'évolution des composantes connexes d'un graphe dont certaines arêtes ou certains sommets sont supprimés ou interdits est une question plus difficile et de grande importance. Connexité Avec Panne sur les Arêtes a été résolu de manière presque optimale [51] dans les graphes généraux. Cela dit, le problème est encore plus compliqué quand la panne impacte des sommets. Pour en avoir l'intuition, on remarquera que la suppression d'une seule arête suppose l'augmentation du nombre de composantes, au plus, que d'une unité, alors que la suppression d'un sommet peut voir naître un nombre de composantes de l'ordre de son degré, soit $O(n)$ composantes dans les graphes généraux.

```
1: procedure LABELING(G)
2:     file f ← FILE-VIDE
3:     effacer toutes les étiquettes des sommets de G
4:     enfiler un sommet quelconque de G dans f
5:     c ← 1
6:     donner à l'étiquette du sommet en tête de f la valeur c
7:     tant que f non vide faire
8:         pour tout x voisin du sommet en tête de f non étiqueté faire
9:             donner à l'étiquette de x la valeur c
10:            enfiler x dans f
11:        fin pour
12:        défiler f
13:        si f est vide alors
14:            c ← c + 1
15:            soit s un sommet sans étiquette de G
16:            donner à l'étiquette du sommet s la valeur c
17:            enfiler s dans f
18:        fin si
19:    fin tant que
20:    défiler f
21:    retourner ⊥
22: fin procedure
```

Algorithme 2: CONNEXITÉ SANS PANNE - Étiquetage

```
1: fonction QUERY(u, v)
2:     si u et v ont la même étiqueté alors retourner ⊤
3:     sinon retourner ⊥
4:     fin si
5: fin fonction
```

Algorithme 3: CONNEXITÉ SANS PANNE - Requête

Le problème Connexité Avec Panne sur les Arêtes et les sommets peut se réduire à Connexité Avec Panne sur les Sommets en considérant, pour tout graphe G, le graphe G' résultant de la subdivision de chacune des arêtes du graphe G par un sommet de degré 2. Une arête $e = u_i - u_j$ donne donc naissance à deux arêtes consécutives $u_i - u_e - u_j$. Si l'on considère la fonction $s : E \to V'$ qui a chaque arête de G associe le sommet qui la subdivise dans G' ($u_e = s(e)$), alors une panne $X \cup Y$ de G équivaut à une panne $X \cup s(Y)$ sur les sommets de G'. On notera que cette réduction peut parfois supposer de sortir de la classe d'appartenance du graphe étudié [10] ou de modifier l'ordre de grandeur du nombre de sommet [11].

Nous allons présenter, dans le chapitre à venir, le schéma d'étiquetage proposé par [14][15] pour la classe des graphes planaires (dans un premier temps, 3-connexes). Ce travail étant le point de départ du notre, nous allons le présenter très largement. La lecture du paragraphe 1.2.2 montrant, à un niveau de détail microscopique, le mécanisme sous-jacent à une preuve d'existence d'un schéma ; ainsi que celle des derniers paragraphes 1.4 et 1.5 résumant ce travail et en soulignant certaines limites se heurtant à nos objectifs, devraient suffire au lecteur souhaitant entrer directement dans le vif du sujet de nos travaux.

10. Par exemple, la classe des graphes 3-connexes n'est pas stable par subdivision d'arête.
11. Pour la classe des cliques, $n' = \Omega(n^2)$

Chapitre 1

Schéma pour les graphes planaires de Courcelle et al.

Ce chapitre a pour objet la présentation des travaux constituant le point de départ des notres. A savoir un schéma d'étiquetage de connexité proposé par Courcelle *et al.* [14][15] pour la famille des graphes planaires 3-connexes, dans un premier temps, puis pour celle des graphes planaires sans la contrainte de connectivité, dans un second.

Voici l'énoncé de ces résultats :

Théorème 2 ([14])**.** *Le problème* Connexité Avec Panne *admet un schéma d'étiquetage compact pour la famille des graphes planaires* 3*-connexes.*

- *L'étiquetage du graphe se fait en temps* $O(n \log n)$*.*
- *Les étiquettes sont de taille* $O(\log n)$*.*
- *Les requêtes sont traitées en temps* $O(|X|^2)$ *avec* X *l'ensemble en panne.*

Théorème 3 ([15])**.** *Le théorème 2 s'étend à la famille des graphes planaires sans contrainte de connectivité.*

Nous allons, donc, tout au long de cette partie, présenter les grandes lignes de la preuve du théorème 2 et les intuitions de celle du théorème 3. Ce travail nous permettra d'exhiber une des limites de la stratégie de [14] en ce qui concerne le temps de requête, ce qui justifiera pour nous un changement de paradigme d'analyse et une nouvelle approche.

1.1 Intuition de la preuve

Le *plongement* dans le plan d'un graphe planaire $G = (V, E)$ est un couple d'applications $\varepsilon = (p, s)$ tel que $p : V \to \mathbb{R}^2$ associe à tout sommet u sa position $p(u)$ dans le plan et que $s : E \to \mathcal{P}(\mathbb{R}^2)$ associe à toute arête $e = (u, v)$ une courbe ayant pour extrémités $p(u)$ et $p(v)$; et tel que pour tout couple d'arêtes distinctes e et f, et pour tout $x \in \mathbb{R}^2$, on ait $x \in s(e) \cap s(f)$ si et seulement s'il existe $u \in V$ tel que $x = p(u)$ et u est incident à e et f. Si $s(e)$ est un segment de droite pour tout $e \in E$, on dit qu'on a un plongement dans le plan en lignes droites. Un graphe planaire *topologique* G est la classe d'équivalence des plongements d'un graphe planaire par homéomorphisme [1]. On le notera $G = (V, E, F)$, où F est l'ensemble fixé de ses faces.

Soit G un graphe planaire topologique. On notera G^+ (figure 2) le graphe obtenu en ajoutant un sommet au milieu de chaque face, et en le reliant à chacun des sommets incidents à cette face. On remarque que si G est 2-connexe, un sommet d'une face f n'apparait que dans un seul coin de cette face [2], G^+ est donc un graphe planaire simple. On peut donc, par l'algorithme de Schnyder [53] ou un algorithme équivalent [8], plonger ce graphe dans le plan, de telle manière que les sommets aient des coordonnées entières (de tailles linéaires par rapport à celle du graphe) et que les arêtes soient des lignes droites qui ne s'intersectent pas deux à deux. On parle de plongement planaire en lignes droites sur la grille. (figure 1) Considérons donc un tel plongement pour G^+.

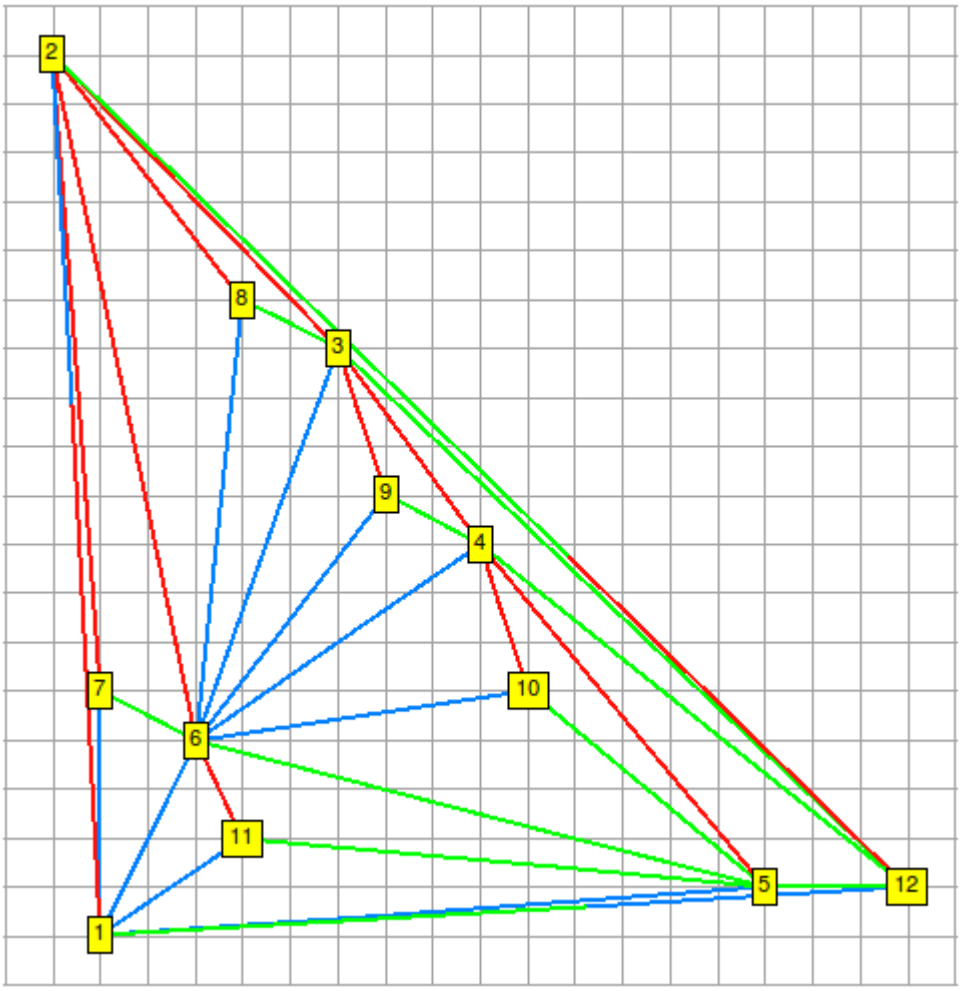
Pour $X \subseteq V(G)$, on définit sa barrière $\mathrm{Bar}(X)$ comme étant l'ensemble des arêtes de G^+ choisies telles que deux sommets $u, v \in V(G) \setminus X$ sont séparés par X dans G si et seulement si ils sont séparés dans le plan par $\mathrm{Bar}(X)$. (Figure 3)

Forts de ces définitions, le but est de construire un schéma d'étiquetage attribuant à tous les sommets une étiquette permettant :

1. Un homéomorphisme est une transformation continue et réversible conservant la topologie. L'objet est déformé comme du caoutchouc : un beignet torique est homéomorphe à une tasse possédant une anse.

2. C'est-à-dire que si on procède au parcours circulaire des arêtes de f, on ne voit un sommet donné qu'une seule fois par tour.

FIGURE 1 – Plongement en lignes droites d'un graphe planaire 3-connexe dans la grille par l'algorithme décrit dans [8]. L'image est générée avec `pigale`.



— de localiser dans le plan les sommets u et v dont la connexion est à tester.
— et de construire $\mathrm{Bar}(X)$ à partir de la liste des étiquttes des sommets de X.

Décider de la connexion de u et v est alors équivalent à répondre à la

question : Les coordonnées de u et v appartiennent-elles ou non à la même partie connexe de $\mathbb{R}^2 \setminus \mathrm{Bar}(X)$? On a recours à un algorithme de LOCALISATION DE POINT pour en décider en un temps $O(p \cdot \log p)$ où p est le nombre de segments formant la barrière.

1.2 Schéma d'étiquetage

1.2.1 préliminaires

On construit une étiquette $L_G(x)$ pour tout sommet x du graphe G. Cette étiquette est formée des coordonnées de x dans le plongement fixé de G^+ sur la grille et de celle d'un nombre borné de sommets de G^+. La preuve de l'algorithme impose que le nombre de faces incidentes à chaque paire de sommets soit borné, et c'est pour cette raison que l'on a restreint dans un premier temps le problème à la famille des graphes 3-connexes.

On note que la notion de graphe planaire topologique est une notion discrète. On a affaire à un graphe planaire $G = (V, E)$ à chaque sommet duquel on attribue la liste circulaire des arêtes incidentes $E^0(u)$ selon le sens inverse des aiguilles d'une montre. On pourra noter (e', u, e) un coin d'une face si e' suit e dans la liste $E^0(u)$.

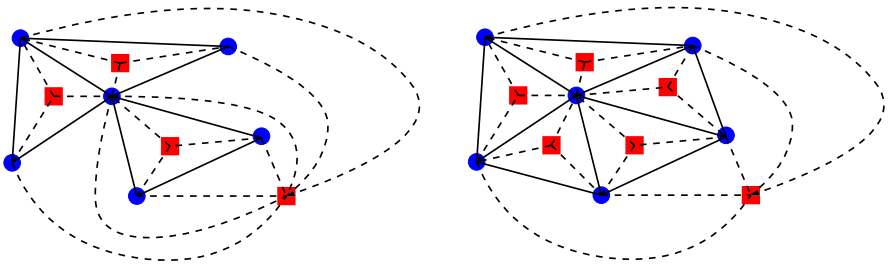
Étant donné qu'on plonge le graphe sur le plan et non sur la sphère, la face extérieure du graphe topologique est désignée par un de ses coins.

Soit $G = (V, E, F)$ un graphe planaire topologique connexe. On définit plus précisément le graphe planaire connexe $G^+ = (V \cup F, E \cup E')$, que l'on nomme toujours *graphe augmenté* de G, où E' est l'ensemble des arêtes reliant une face f à chaque sommet incident u, autant de fois qu'il existe de coin (e, u, e') donnant sur f (Il peut y avoir plusieurs coins d'un sommet donnant sur une face si c'est un sommet d'articulation.).

Un graphe G à n sommets a un nombre de faces maximal dans une configuration triangulée ($m = 2n - 4$). Le nombre de sommets de G^+ est donc

inférieur à $3n-4$. Tout plongement ε de G peut être prolongé en un plongement ε^+ de G^+ en attribuant à une face f une position $p(f)$ dans l'ouvert de $\mathbb{R}^2$ correspondant et en reliant ce point au sommet par une ligne coupant le coin correspondant. On notera que G^+ est triangulé. (Figure 2.) On notera aussi que G^+ est simple si et seulement si G est 2-connexe.

FIGURE 2 – Graphes augmentés de deux graphes. Le second graphe est 2-connexe : son augmenté est simple.



Soit $G = (V, E, F)$ et $G^+ = (V \cup F, E \cup E')$ le graphe augmenté correspondant. On peut définir $\mathrm{Bar}(X)$ pour tout $X \subseteq V$ comme l'ensemble des arêtes de G^+ qui relient une face f et un sommet x tel qu'il existe dans G^+ une autre arête reliant f à un sommet $y \in X$. (On aura éventuellement $x = y$ dans le cas d'un graphe non 2-connexe.)

Si ε^+ est un plongement de G^+ et que pour tout objet combinatoire o du graphe, $s(o)$ est son objet géométrique associé [3] dans le plongement, alors on pose :

$$\mathrm{Bar}(X, \varepsilon^+) = \bigcup_{e \in Bar(X)} s(e)$$

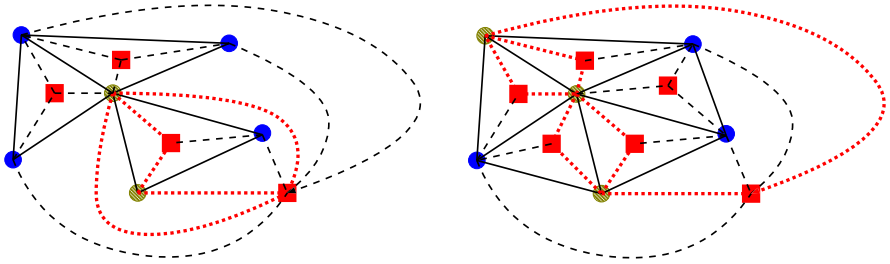
Le lieu géométrique $\mathrm{Bar}(X, \varepsilon^+)$ est un sous-ensemble fermé borné de $\mathbb{R}^2$. Deux points x et y de $\mathbb{R}^2$ sont dits séparés par $Bar(X, \varepsilon^+)$ s'ils sont dans deux composantes connexes distinctes de $\mathbb{R}^2 \setminus \mathrm{Bar}(X, \varepsilon^+)$. (Voir figure 3)

Les auteurs démontrent le lemme suivant :

Lemme 4 ([14]). *Pour tout graphe G planaire topologique dont le graphe augmenté G^+ admet ε^+ pour plongement planaire, pour tout $X \subseteq V(G)$,*

3. Il nous arrivera à l'avenir d'omettre cette distinction.

FIGURE 3 – Barrières de nos deux graphes. Deux sommets sont en pannes dans le premier, 3 sommets dans le second. (les sommets en pannes sont grisés.)



pour tous $u, v \in V(G) \setminus X$ *:*
les sommets u *et* v *sont séparés dans* G *si et seulement si les points du plan leur correspondant sont séparés par* $\mathrm{Bar}(X, \varepsilon^+)$.

Ce lemme constitue la pierre angulaire de la réduction du problème purement combinatoire qui incombe aux auteurs à un problème de géométrie algorithmique.

On doit donc construire des étiquettes permettant de retrouver, pour chaque paire de sommets $x, y \in X$, les coordonnées des faces auxquelles ils sont tous deux incidents. Ce qui nous permettra de construire, à partir des étiquettes relatives à une panne, la barrière correspondante.

1.2.2 Méta-preuve

Pour prouver l'existence d'un schéma d'étiquetage compact, il est nécessaire de controler la quantité d'information par sommet impacté sur laquelle on peut compter. On peut grossièrement considérer chaque information apparaissant dans l'étiquette d'un sommet x comme l'image précalculée $f(x)$ d'une fonction f appliquée à x. Les auteurs définissent plus formellement les limites de ce qui est étiquetable.

Si $\mathcal{F}$ est un ensemble fini de fonctions unaires et $\mathcal{X}$ est un ensemble fini de variables, on note $\mathcal{B}(\mathcal{F}, \mathcal{X})$ l'ensemble des formules sans quantification qui sont des combinaisons booléennes de formules atomiques de formes $x = y$, $f(x) = y$ $f(x) = g(y)$, pour $x, y \in \mathcal{X}$ et $f, g \in \mathcal{F}$. (Avec éventuellement $f = g$.) Les formules de la forme $x = f(g(y))$ sont proscrites. $\mathcal{B}(\mathcal{F}, \mathcal{X})$ n'est donc pas l'ensemble des formules non-quantifiées sur $\mathcal{F}$ et $\mathcal{X}$.

Soit V un ensemble et soit $\bar{f} : V \to V$ une fonction totale pour tout $f \in \mathcal{F}$. On notera $\bar{\mathcal{F}}$ la famille $(\bar{f})_{f \in \mathcal{F}}$ et $\mathcal{X} = \{x_1, \cdots, x_m\}$. Toute formule $\varphi \in \mathcal{B}(\mathcal{F}, \mathcal{X})$ définit alors une relation m-aire $R_\varphi \subseteq V^m$ de la forme :

$$R_\varphi = \{(a_1, \cdots, a_m) \in V^m \mid \varphi(a_1, \cdots, a_m)\}$$

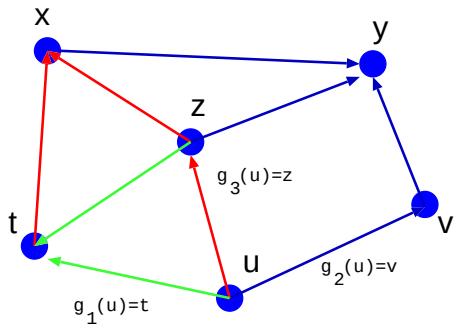
On dit que R_φ est représentée par les fonctions de $\bar{\mathcal{F}}$ et la formule φ.

On dit qu'une fonction multivaluée d'arité m, *i.e.* une fonction $g : V^m \to \mathcal{P}(V)$, est représentée par les fonctions de $\bar{\mathcal{F}}$ et une formule φ si la relation $(m+1)$-aire $y \in g(x_1, \cdots, x_m)$ est représentée par $\bar{\mathcal{F}}$ et φ, avec φ une disjonction de formules de la forme $\psi \wedge (y = f(x_i))$ or $\psi \wedge (y = x_i)$ avec $\psi \in \mathcal{B}(\mathcal{F}, \{x_1, \cdots, x_m\})$. Pour une classe $\mathcal{C}$ de graphes (ou de graphes topologiques) G, on considèrera des relations $P, Q, \cdots$ et des fonctions $g, h, \cdots$ sur l'ensemble $X(G) = V(G)$ (ou même sur $X(G) = V(G) \cup E(G) \cup F(G)$) comme, par exemple, l'adjacence de deux sommets, l'incidence à une face commune, etc. On dira que $P, Q, \cdots, g, h, \cdots$ sont représentables par k fonctions dans les graphes de $\mathcal{C}$ s'il existe des formules $\varphi_P, \varphi_Q, \cdots, \varphi_g, \varphi_h, \cdots$ bien formées telles que pour tout $G \in \mathcal{C}$ il existe un k-uplet $\mathcal{F}$ de fonctions totales unaires sur $X(G)$ qui représentent $P, Q, \cdots, g, h, \cdots$ dans G par le biais de, respectivement, $\varphi_P, \varphi_Q, \cdots, \varphi_g, \varphi_h, \cdots$

Dans les constructions à suivre, on utilisera des fonction partielles $\bar{f} : X(G) \to X(G)$ où $\bar{f}(x) \neq x$ pour tout x. On les rendra totales en posant, par convention $\bar{f}(x) = x$ au lieu de « $\bar{f}(x)$ n'est pas défini ».

Pour nous entraîner à manipuler ces concepts, les auteurs démontrent le lemme suivant :

FIGURE 4 – L'adjacence représentée par 3 fonctions dans un graphe planaire.



Lemme 5 ([14]). *L'adjacence dans la famille des graphes planaires $G = (V, E)$ peut être représentée par trois fonctions de $V \to V$. (Figure 4.)*

Démonstration. On peut se limiter aux graphes simples car la multiplicité des arêtes n'a pas d'impact sur l'adjacence.

Soit $G = (V, E)$ un graphe planaire simple. E peut être partitionné en trois ensembles E_1, E_2, E_3 tels que $G[E_i]$ soit une forêt pour chaque i. On peut considérer ces forêts enracinées (On oriente chaque arête en direction de la racine de l'arbre auquel elle appartient.) Un tel ensemble de forêt est appelé *bois de Schnyder*.

Pour $x, y \in V$, on pose $g_i(x) = y$ si et seulement si y est le père de x dans $G[E_i]$. On étend cette fonction partielle à une fonction totale de la manière proposée plus haut. Alors, x et y sont adjacents si et seulement si :

$$x \neq y \wedge \bigvee_{1 \leq i \leq 3} (x = g_i(y) \vee y = g_i(x))$$

La formule ci-dessus et les 3 fonctions qu'elle utilise *représentent* la propriété d'adjacence dans les graphes planaires et c'est ce qu'il fallait démontrer. □

1.2.3 Preuve

Pour toute couple (x, y) de sommets distincts d'un graphe planaire topologique, $\text{Faces}(x, y)$ désignera l'ensemble des faces auxquelles à la fois x et y sont incidents. On dira qu'un graphe planaire topologique est *m-face-borné* si $|\,\text{Faces}(x, y)| \leq m$ pour tout $x, y \in V(G)$, $x \neq y$.

On remarque que les graphes 2-connexes obtenus à partir de graphes simple 3-connexe par subdivision d'arêtes (et qui ont un plongement topologique unique sur la sphère) sont 2-face-bornés. Dans un tel graphe, deux sommets sont adjacents à deux faces distinctes si et seulement si ils sont adjacents entre eux ou reliés par un chemin dont tous les sommets intermédiaires sont de degré 2.

On peut définir aussi la propriété

$$\text{sf}(x, y) \iff |\,\text{Faces}(x, y)| \geq 1$$

pour x et y d'avoir au moins une face incidente en commun, et

$$m\text{face}(x, y) \iff |\,\text{Faces}(x, y)| \leq m$$

celle pour x et y d'avoir au plus m faces incidente en commun.

Un m-uplet de fonctions de *sélection de face* est un m-uplet $(\text{Select}_i(x, y))_{i\in[m]}$ de fonctions partielles $V(G) \times V(G) \to F(G)$ tel que, pour tout $x, y \in V(G)$:

— $\text{Select}_i(x, y) \neq \text{Select}_j(x, y)$ pour tout $i \neq j$
— $\text{Select}_i(x, y) \in \text{Faces}(x, y)$ pour tout i
— $\text{Faces}(x, y) = \{\text{Select}_1(x, y), \cdots, \text{Select}_m(x, y)\}$ si $|\,\text{Faces}(x, y)| \leq m$

Ces fonctions nous permettent d'énumérer les faces communes à deux sommets. On remarquera que $Select_i(x, y)$ et $Select_i(y, x)$ ne sont pas nécessairement égales.

Pour des sommets voisins x et y, on note $\text{left}(x, y)$ la face à gauche de l'arête $x - y$ traversée de x vers y. $\text{left}(y, x)$ est alors la face à droite de cette arête. On a égalité si $x - y$ est un isthme. On appellera cette fonction la

fonction *face-gauche*.

Proposition 6 ([14])**.** *Pour tout graphe planaire topologique, simple et connexe, on peut représenter :*

- *l'adjacence et la fonction face-gauche avec 9 fonctions sur* $V(G) \cup F(G)$.
- *l'adjacence et la propriété face-commune avec 18 fonctions.*
- *Pour tout* m*, on peut définir un* m*-uplet de sélection et le représenter avec* $18 + 3m$ *fonctions incluant les 18 fonctions de la propriété face-commune.*

Démonstration. Soit $G = (V, E, F)$ un graphe planaire topologique simple et connexe. Soit g_1, g_2, g_3 les trois fonctions partielles construites dans la démonstration du lemme 5. Elle permettent de représenter l'adjacence. On utilise également la fonction *face-gauche*. Soient $g_i^\alpha, (i, \alpha) \in \{1, 2, 3\} \times \{\text{left}, \text{right}\}$, les six fonctions partielles $V \to F$ telles que :

$$g_i^{\text{left}} = \text{left}(x, g_i(x))$$

$$g_i^{\text{right}} = \text{left}(g_i(x), x)$$

avec $g_i^\alpha(x)$ non définie si $g_i(x)$ ne l'est pas.

On a alors :

$$\begin{aligned} f = \text{left}(x, y) \iff & \bigvee_{1 \le i \le 3} \left(y = g_i(x) \wedge f = g_i^{\text{left}}(x)\right) \\ & \vee \bigvee_{1 \le i \le 3} \left(x = g_i(y) \wedge f = g_i^{\text{right}}(y)\right) \end{aligned}$$

Cette représentation utilise 9 fonctions.

La propriété de face-commune utilise le graphe planaire $G^+ = (V \cup F, E \cup E')$. Soient g_i^+ pour $i \in \{1, 2, 3\}$ les trois fonctions partielles $V \cup F \to V \cup F$ représentant l'adjacence dans G^+. La propriété face-commune peut être

représentée ainsi (figure 6) pour $x, y \in V, x \neq y$:

$$\bigvee_{1\leq i,j\leq 3} g_i^+(x) = g_j^+(y) \in F \qquad (1a)$$

$$\vee \bigvee_{1\leq i,j\leq 3} g_i^+(x) \in F \wedge g_j^+(g_i^+(x)) = y \qquad (1b)$$

$$\vee \bigvee_{1\leq i,j\leq 3} g_i^+(x) \in F \wedge g_j^+(g_i^+(y)) = x \qquad (1c)$$

$$\vee \quad \exists f \in F\Big(\bigvee_{1\leq i,j\leq 3} g_i^+(f) = x \wedge g_j^+(f) = y\Big) \qquad (1d)$$

Pour manipuler l'assertion « $g_i^+(x) \in F$ », on utilisera les fonctions partielles $g_i' : V \to F$ définie par :

$$g_i'(x) = \texttt{si} \quad g_i^+(x) \in F \quad \texttt{alors} \quad g_i^+(x) \quad \texttt{sinon} \quad \perp$$

Pour manipuler l'assertion « $g_i^+(x) \in F \wedge g_j^+(g_i^+(x)) = y$ », on utilisera les fonctions partielles $g_{i,j} :' V \to F$ définie par :

$$\begin{aligned} g_{i,j}'(x) = &\texttt{si} \quad g_i^+(x) \in F \quad \texttt{et} \quad g_j^+(g_i^+(x)) \neq \perp \\ &\texttt{alors} \quad g_j^+(g_i^+(x)) \\ &\texttt{sinon} \quad \perp \end{aligned}$$

Il s'agit maintenant d'éliminer la quantification existentielle « $\exists f \in F(\cdots)$ » dans la formule $(1d)$. Pour ce faire, on définit un graphe auxiliaire H (figure 5) où $V(H) = V(G)$ et où $x - y$ est une arête si et seulement s'il existe $i, j \in \{1, 2, 3\}$ et $f \in F$ tels qu'on ait $g_i^+(f) = x$ et $g_j^+(f) = y$ (si x et y sont pères d'une même face, chacun dans au moins une des trois forêts). Une telle arête peut être dessinée à l'intérieur de la face f dans un plongement ε de G. (G reste planaire. On ajoute au maximum trois arêtes par face.)

On pose h_1, h_2, h_3 les trois fonctions relatives à l'adjacence dans H. On peut alors réécrire la condition (1d) de cette manière :

$$\bigvee_{1\leq i\leq 3} h_i(x) = y \vee h_i(y) = x$$

On peut donc bien représenter les propriétés d'adjacence et de face-commune par 18 fonctions, à savoir $g_i, g_i', g_{i,j}', h_i$ pour $i, j \in \{1, 2, 3\}$.

FIGURE 5 – Construction du graphe H à partir du graphe G^+ muni d'un bois de Schnyder.

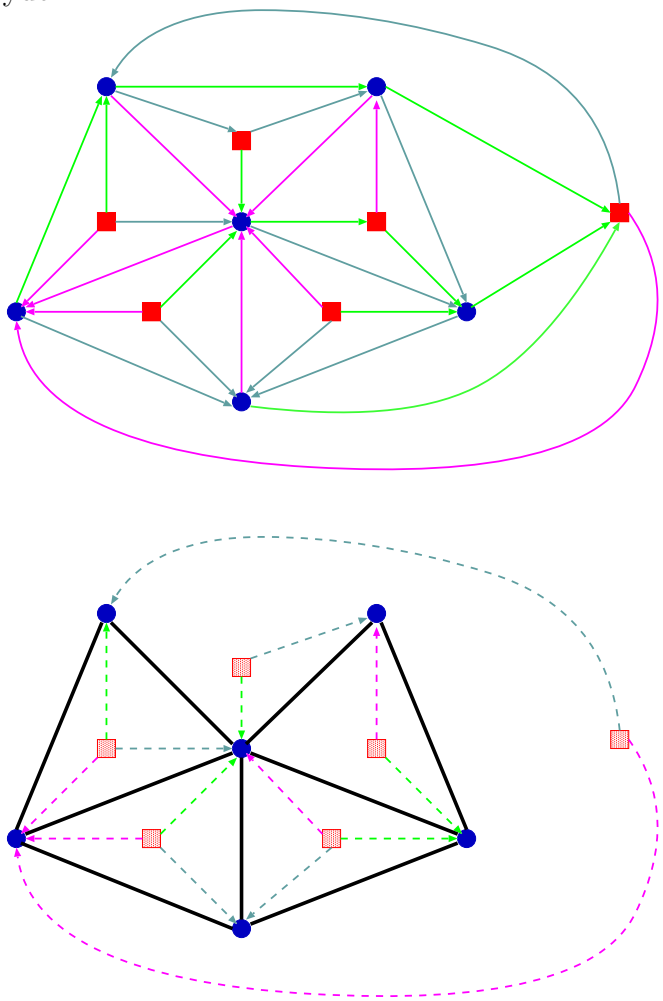


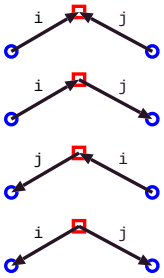
FIGURE 6 – Les différents types de faces communes.

$$\bigvee_{1\le i,j\le 3} g_i^+(x) = g_j^+(y) \in F \qquad (1a)$$

$$\vee \bigvee_{1\le i,j\le 3} g_i^+(x) \in F \wedge g_j^+(g_i^+(x)) = y \qquad (1b)$$

$$\vee \bigvee_{1\le i,j\le 3} g_i^+(x) \in F \wedge g_j^+(g_i^+(y)) = x \qquad (1c)$$

$$\vee \quad \exists f \in F\Big(\bigvee_{1\le i,j\le 3} g_i^+(f) = x \wedge g_j^+(f) = y\Big) \qquad (1d)$$



Il nous reste à définir le moyen de représenter un m-uplet de fonctions de sélection de face. On utilisera les cas $(1a)$ à $(1d)$ qui caractérisent la propriété de face commune (figure 6). On remarquera dans un premier temps que ces cas s'excluent mutuellement. Chaque face de Faces(x, y) est spécifiée dans un et un seul de ces cas.

On fixe un ordre sur $F(G)$. Soit $x, y \in V(G)$ tels que $x \neq y$ et soit $f \in F(G)$. On dit que f est de (x, y)-*type* t si $f \in$ Faces(x, y) et si l'une des conditions suivante est avérée :

- (a) $f = g'_i(x) = g'_j(y)$ et $t = (a, i, j)$
- (b) $f = g'_i(x)$, $y = g'_{i,j}(x)$ et $t = (b, i, j)$
- (c) $f = g'_i(y)$, $x = g'_{i,j}(y)$ et $t = (c, i, j)$
- (d) f appartient à $F(x, y)$ qu'on définit comme l'ensemble des faces de Faces(x, y) qui ne sont pas de forme (a), (b) ou (c) ; on fixe une énumération $\{f_1, \cdots, f_p, \cdots\}$ de cet ensemble à partir de l'ordre sur $F(G)$. On dit que le j-ème élément de $F(x, y)$ est de (x, y)-type $t = (d, j)$.

On remarquera que le (y, x)-type de f se déduit immédiatement de son (x, y)-type. Les types $(a, j, i), (c, i, j), b(i, j), (d, j)$ donneront, respectivement, $(a, i, j), (b, i, j), (c, i, j), (d, j)$. Et on a $F(x, y) = F(y, x)$.

On peut maintenant définir les fonctions partielles de $V(G) \to F(G)$ pour $i \in \{1, 2, 3\}$ et $j \geq 1$: $h_{i,j}(x) = f$ si $h_i(x)$ est définie et f est le j-ème élément de $F(x, h_i(x))$.

Pour tous $x, y \in V(G), x \neq y$ and $j \geq 1$, il y a, au plus, une face de (x, y)-type (d, j) et elle est caractérisée par la condition :

$$\bigvee_{1 \leq i \leq 3} \big((f = h_i, j(x) \wedge y = h_i(x)) \vee (f = h_i, j(y) \wedge y = h_i(y))\big)$$

De manière analogue, on constate que, pour tout $t \in \{a, b, c\} \times \{1, 2, 3\} \times \{1, 2, 3\}$, il existe, au plus une face f de (x, y)-type t, et elle est caractérisée par une condition similaire. Par exemple, si $t = (c, 1, 3)$, la condition est :

$$f = g'_1(y) \wedge x = g'_{,1,3}(y)$$

En appliquant un ordre lexicographique sur les types des faces, on obtient un ordre sur Faces(x, y). Soit Select$_i(x, y)$ le i-ème élément de cet ensemble. Il est clair qu'on pourra exprimer, pour tout $i \leq m$, Select$_i(x, y)$ par une disjonction de formules de la forme :

$$f = g(z) \wedge \psi$$

Avec $z \in \{x, y\}$, $\mathcal{F}_m = \{g_i, g'_i, g'_{i,j}, h_i, h'_{i,l} | i, j \in \{1, 2, 3\}, l\{1, \cdots, m\}\}$, $\psi \in \mathcal{B}(\mathcal{F}_m, x, y)$ et $g \in \mathcal{F}_m$.

L'ensemble $\mathcal{F}_m$ compte $18 + 3m$ fonctions et permet de représenter un m-uplet de fonctions de sélection de face. □

Comme suggéré dans la preuve, il conviendra donc dans notre cas de créer une étiquette, pour tout sommet $x \in V(G)$, contenant les valeurs de $f(x)$ pour tout $f \in \mathcal{F}_2$ pour pouvoir lister les faces communes à toute paire de sommets en panne et ainsi construire la barrière correspondante. L'étiquette de tout x contiendra également les coordonnées de x dans le plongement afin de pouvoir tester la connexion d'une paire de sommet (x, y) dans $\mathbb{R}^2 \setminus \mathrm{Bar}(X, \varepsilon^+)$.

On considèrera donc dans la partie à venir traitant des requêtes que ces informations sont accessibles à partir de l'étiquette. On peut par exemple imaginer une étiquette de la forme :

$$\mathrm{Lab}(x) = p(x) \quad p\big(f_1(x)\big) \quad \cdots \quad p\big(f_{24}(x)\big)$$

Où $p(x)$ représente la position d'un sommet ou d'une face x dans le plongement ε^+ de G^+ et où les f_i sont les fonctions de $\mathcal{F}_2$.

On notera qu'une telle étiquette est de taille (en nombre de bits) inférieure à

$$25 \cdot \max_{(x \in V^+)} \big(\text{taille}(p(x)) \big) + K$$

(où K est une petite constante due à la nécessité de séparateurs) et que taille$(p(x))$, la quantité d'information nécessaire à la représentation de la position du point x dans le plan, est inférieure à (en bits) $2 \cdot \log(g_{\varepsilon^+})$ où g_ε^+ représente la plus grande dimension de la grille sur laquelle est appliqué le plongement.

L'algorithme présenté dans [8] plonge, par ε^+, le graphe $G^+ = (V \cup F, E \cup E')$ dans une grille telle que $g_{\varepsilon^+} \leq |V \cup F| - 2$. On rappelle que $|V \cup F| \leq 3n - 4$. On a donc une étiquette de moins de $50 \cdot \log(3n - 4) + O(1)$ bits soit

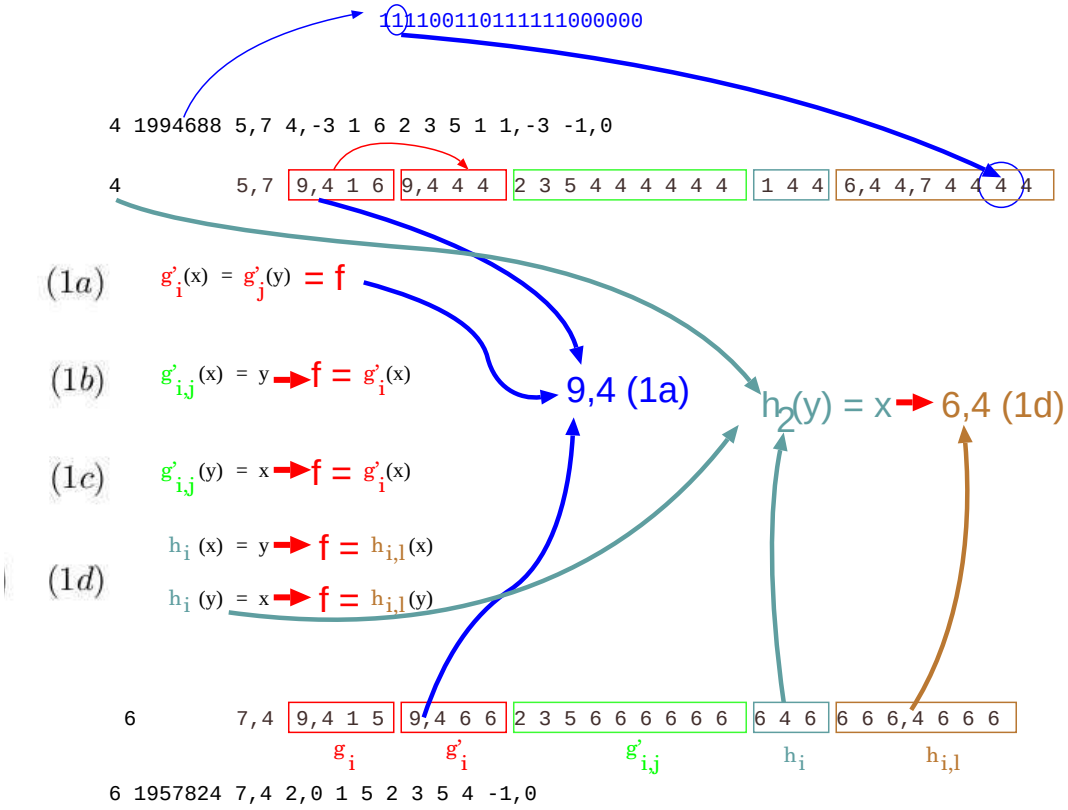
$$50 \cdot \log n + O(1)$$

1.3 Requête

Soit un graphe G planaire et 3-connexe. On considèrera dans cette partie qu'une requête q est une liste d'étiquettes $(L_G(u), L_G(v), L_G(x_1) \cdots, L_G(x_k))$ où les u, v et les x_i sont des sommets tous distincts de G dont les étiquettes ont été calculées comme énoncé dans la partie précédente. u et v sont les sommets dont on veut tester la connexion. Les x_i sont les sommets en panne.

Par construction des étiquettes, on sait que l'information contenue dans q nous permet de calculer les coordonnées de u et de v (selon ε^+) ainsi que celles des extrémités des segments qui composent $\text{Bar}(X, \varepsilon^+)$ (figure 7). Par définition de la barrière, ces segments ne se croisent pas, *i.e.* leurs intersections sont réduites à des extrémités.

FIGURE 7 – A partir de leurs étiquettes, on trouve deux faces adjacentes communes aux sommets x et y.



Un ensemble Y fini de segments de $\mathbb{R}^2$ ne se croisant pas deux à deux est appelé *subdivision du plan*. L'union $\bigcup Y$ de ces segments est un sous-ensemble fermé de $\mathbb{R}^2$. Notre problème de connexion peut être réduit à ce nouveau problème :

Entrée. Une subdivision Y du plan composée de segments dont les extrémités sont dans $\mathbb{N}^2$ et $u \in \mathbb{N}^2 \setminus \bigcup Y$.

Sortie. Composante connexe de u dans $\mathbb{R}^2 \setminus \bigcup Y$?

Ce problème est appelé Localisation d'un Point dans le Plan (En anglais : Planar Point Location). Un résultat classique de géométrie algorithmique est que pour Y une subdivision du plan composée de m segments définis sur une grille $n \times n$. On peut construire en un temps $O(m \cdot log(m))$ une structure de donnée de taille $O(m)$ à partir de laquelle on peut obtenir en temps $O(\log m)$ la composante connexe dans $\mathbb{R}^2 \setminus \bigcup Y$ d'un point du plan [18]. On peut donc faire appel en boîte noire à ce résultat pour répondre à notre requête de connexité en comparant les composantes connexes de u et de v.

Le résultat est généralisé aux graphes planaires. Cette partie est plus technique, en effet, en perdant la propriété *m-faces-borné*, on s'interdit de pouvoir avoir accès à l'ensemble des coordonnées des segments de la barrière. Les auteurs remplace l'ensemble des occurences d'un chemin de la forme $x - f - y$ par seulement l'une d'elle et construisent une barrière réduite. Ils traitent ensuite les faux-positifs en utilisant une décomposition en composante 3-connexes du graphe et une technique issue de [17] et utilisant la logique du second ordre monadique.

1.4 Résumé

Ce travail propose une méthode d'étiquetage compact [4], prenant un temps $O(n \log n)$, des graphes planaires 3-connexes. Celle-ci exploite leur arboricité

4. Une implémentation de ce schéma d'étiquetage pour les graphes planaires 3-connexes constituant le noyau de notre mémoire de Master valide empiriquement le résultat.

bornée et le fait que deux sommets x et y ne peuvent être incidents qu'à un nombre borné de faces communes dans un plongement du graphe.

Le traitement d'une panne consiste en la création d'une subdivision du plan géométrique, appelée *barrière*, en temps $O(|X|^2)$, et en la création d'une structure de donnée de requête de Localisation d'un Point dans le Plan en temps $O(|X| \log |X|)$. Le test (à proprement parler) est réduit à une paire de requêtes de Localisation d'un Point dans le Plan dans cette structure. Elle est effectuée en temps $O(\log |X|)$. Le résultat est ensuite généralisé aux graphes planaires, sans restriction de connectivité.

1.5 Limites

La borne $O(\log |X|)$ n'a d'intérêt que si l'on considère un nombre de requêtes important pour une seule et même panne, sinon, elle est négligeable devant la borne $O(|X|^2)$ nécessaire à la construction de la barrière. C'est pour cela que nous introduirons un paradigme d'analyse des schémas d'étiquetage de connexité distingant le temps de pré-requête du temps de requête.

La borne $O(\log |X|)$ est inhérente aux possibilités offertes par les algorithmes optimaux de Localisation d'un Point dans le Plan. Elle constitue donc un *goulot d'étranglement* de la stratégie de réduction à un problème géométrique.

Chapitre 2

Des graphes planaires aux graphes de genres bornés

2.1 Résumé de notre contribution

Les figures 39 et 40 du chapitre de conclusion de ce document pourront aider le lecteur à s'orienter dans la lecture du présent chapitre. Elles le résument graphiquement.

Nous présentons ici un schéma d'étiquetage original pour le problème Connexité Avec Panne pour la famille des graphes 3-connexes plongeables sur une surface fermée[1], orientable[2] ou non[3], paramétrée par différentes grandeurs, toutes entre elles liées :

— Le *genre* est le nombre maximum de courbes fermées simples, sans points communs, pouvant être tracées à l'intérieur de cette surface sans la déconnecter. Le genre d'une sphère est donc 0, car une courbe fermée la déconnecte fatalement en deux disques. Un tore est de genre 1, car on peut tracer un cercle *faisant le tour du trou*, qui a pour complémentaire un cylindre topologique à deux bords. Une bouteille de Klein est de genre 2 car deux courbes fermées bien choisies la laisse connexe. Le genre caractérise à la fois les surfaces fermées :

1. Espace topologique dont tout point possède un voisinage homéomorphe à un plan.
2. Qui possède deux faces distinctes que l'on pourrait peindre, l'une en bleu, l'autre en rouge.
3. A l'inverse, unilatère.

  - orientables : le tore à g *trous* est de *genre orientable* g.
  - non-orientables : la sphère muni de k *bonnets-croisés*[4] est de *genre non-orientable* k.
- Le *genre eulérien* d'une surface $\mathcal{S}$ est défini ainsi :

$$\mathrm{eg}(\mathcal{S}) = \begin{cases} 2g & \text{si la surface est orientable} \\ k & \text{si la surface est non-orientable} \end{cases}$$

- La *caractéristique d'Euler-Poincaré*[5] χ est reliée au genre pour les surfaces closes :
  - $\chi = 2 - \mathrm{eg}(\mathcal{S})$
  - D'où $\chi = 2 - 2g$ (si $\mathcal{S}$ orientable)
  - et $\chi = 2 - k$ (si $\mathcal{S}$ non-orientable)

On dit d'un graphe qu'il est :

- de genre orientable g s'il ne peut être plongé sans croisement d'arête sur une surface orientable que si elle est de genre supérieur ou égal à g.
- de genre non-orientable k s'il ne peut être plongé sans croisement d'arête sur une surface non orientable que si elle est de genre supérieur ou égal à k.
- de genre eulérien $\mathrm{eg}(G)$ que s'il ne peut être plongé sans croisement d'arête sur une surface $\mathcal{S}$ que si $\mathrm{eg}(\mathcal{S}) \geq \mathrm{eg}(G)$.

Le décor étant planté, nous énonçons ci-dessous le théorème principal de ce chapitre.

Théorème 7. *La famille des graphes* 3*-connexes admet un schéma d'étiquetage d'urgence pour* Connexité Avec Panne. *Pour un graphe G à n sommets :*

- *Les étiquettes sont de taille $O(\mathrm{eg}(G) \cdot \log n)$ bits.*
- *La complexité d'étiquetage est polynomiale. (linéaire dans le cas planaire)*
- *La complexité de pré-requête est $O(\textsc{Sort}(|X|, n) + \textsc{PrePred}(|X|, n))$.*

4. Construction en deux étapes. On fait un trou dans la sphère, puis on colle au bord circulaire crée celui d'un ruban de Möbius.

5. voir annexe 3.6.

— *La complexité de turbo-requête est* $O(\textsc{Pred}(|X|, n))$.

Où l'on définit comme suit les complexités utilisées pour décrire celle du problème qui nous intéresse :

— $\textsc{Sort}(k, N)$ *la complexité du tri d'un ensemble de* k *entiers issus d'un univers* U *de taille* N.
— $\textsc{PrePred}(k, N)$ *la complexité de construction d'une structure de données adaptée au traitement du problème* Prédécesseur *(qui sera défini plus bas, paragraphe 2.2.1) pour un ensemble de recherche de* k *éléments issus d'un univers* U *de taille* N.
— $\textsc{Pred}(k, N)$ *est le temps de requête pour ce même problème.*

Il est connu que l'on peut, de manière déterministe, réaliser un tri en temps $k \log\log k$ en utilisant par exemple l'algorithme de Han [32].

On sait aussi qu'il est possible de calculer le prédécesseur en utilisant une structure inspirée des *arbres de Van Emde Boas* telle qu'un *fast-tries* de Willard [62][50], en temps $O(\log\log n)$ après création de la structure de donnée de taille $O(k)$ en temps $O(k \log\log n)$.

D'autres structures perfectionéés, comme celle de Thorup et Pătraşcu, de complexité spaciale linéaire en N, permettent des requêtes en temps

$$O\left(\min\left\{\frac{\log\log n}{\log\log\log n}, \sqrt{\frac{\log|X|}{\log\log|X|}}\right\}\right)$$

Ce temps est optimal pour une structure de taille linéaire.

Nous ne pouvons malgré tout nous permettre leur temps de construction polynomial en N aucours de notre phase de pré-requête. Si nous n'étions pas limité en temps et en espace à cette étape-là, nous pourrions même précalculer l'ensemble des requête dans l'univers des possible et répondre aux requêtes en temps $O(1)$ par simple lecture dans un tableau.

Pour la lecture de ce document, le lecteur pourra considérer que :
— $\textsc{Sort}(k, N) = O(k \log\log k)$.
— $\textsc{PrePred}(k, N) = O(k \log\log N)$
— $\textsc{Pred}(k, N) = \log\log N$

Démonstration. La preuve du théorème 7 fera l'objet de tout ce chapitre.

Elle s'appuiera sur quelques résultats préalables, à savoir :

— un schéma d'étiquetage d'urgence pour Connexité Avec Panne sur les arbres,
— un second schéma d'étiquetage d'urgence pour Connexité Avec Panne sur les graphes planaires extérieurs utilisant le schéma précédent,
— une preuve intermédiaire pour le cas particulier des graphes planaires,
— une décomposition des graphes 3-connexes en un graphe planaire extérieur, un arbre couvrant de degré borné $O(\text{eg}(G))$ et un graphe constitué de $O(\text{eg}(G))$ arêtes,
— un méta-schéma d'étiquetage permettant d'articuler ces résultats et d'obtenir notre schéma global.

2.2 Schéma pour la famille des arbres

Nous présentons ici un schéma répondant à un problème plus difficile que Connexité Avec Panne, à savoir Composante Connexe Avec Panne, défini précédemment. On rappelle que l'expression $\textsc{CompConn}(u, X)$, constituant la sortie du problème, doit avoir pour valeur un identifiant de la composante connexe de u dans $G \setminus X$. On peut facilement réduire le problème Connexité Avec Panne au problème Composante Connexe Avec Panne en comparant, pour les sommets u et v sur lesquels le problème se pose, les composantes connexes solutions de Composante Connexe Avec Panne. En d'autres termes, le prédicat « $\textsc{Conn}(u, v, X)$ » est équivalent à « $\textsc{CompConn}(u, X) = \textsc{CompConn}(v, X)$ ».

Théorème 8. Composante Connexe Avec Panne Sur Les Arêtes et les Sommets *admet un schéma d'étiquetage d'urgence pour la famille des arbres à n sommets.*

— *Les étiquettes sont de taille $O(\log n)$ bits.*
— *La complexité d'étiquetage est linéaire.*
— *La complexité de pré-requête est* $\textsc{Sort}(|X|, n) + \textsc{PrePred}(|X|, n)$
— *La complexité de turbo-requête est* $O(\textsc{Pred}(|X|, n)$

Résumé de la preuve. Une réduction par subdivision des arêtes d'un arbre T à n sommets à un arbre T' à $2n-1$ sommets permet de s'abstenir, sans perdre de généralité, de traiter les pannes sur les arêtes.
On montre qu'on peut obtenir un schéma d'étiquetage compact d'urgence pour Composante Connexe Avec Panne en composant deux autres schémas compacts répondant aux problèmes de Plus Proche Ancêtre et Routage dans la famille des arbres.
On obtient des étiquettes compactes par concaténation des étiquettes des deux schémas auxiliaires.
Les complexités de calcul font références à celles du problème Prédécesseur car on y réduit le problème Plus Proche Ancêtre en numérotant les sommets de l'arbre selon un parcours en profondeur.
Les complexités relatives au problème Routage ne sont pas mentionnées dans l'énoncé du résultat car l'existant nous offre des solutions suffisamment efficaces pour qu'elles soient négligeables.

Démonstration. Le schéma va s'appuyer sur deux schémas auxiliaires que nous présentons ci-après.

2.2.1 Schéma auxiliaire : Plus Proche Ancêtre

On définit le problème suivant :

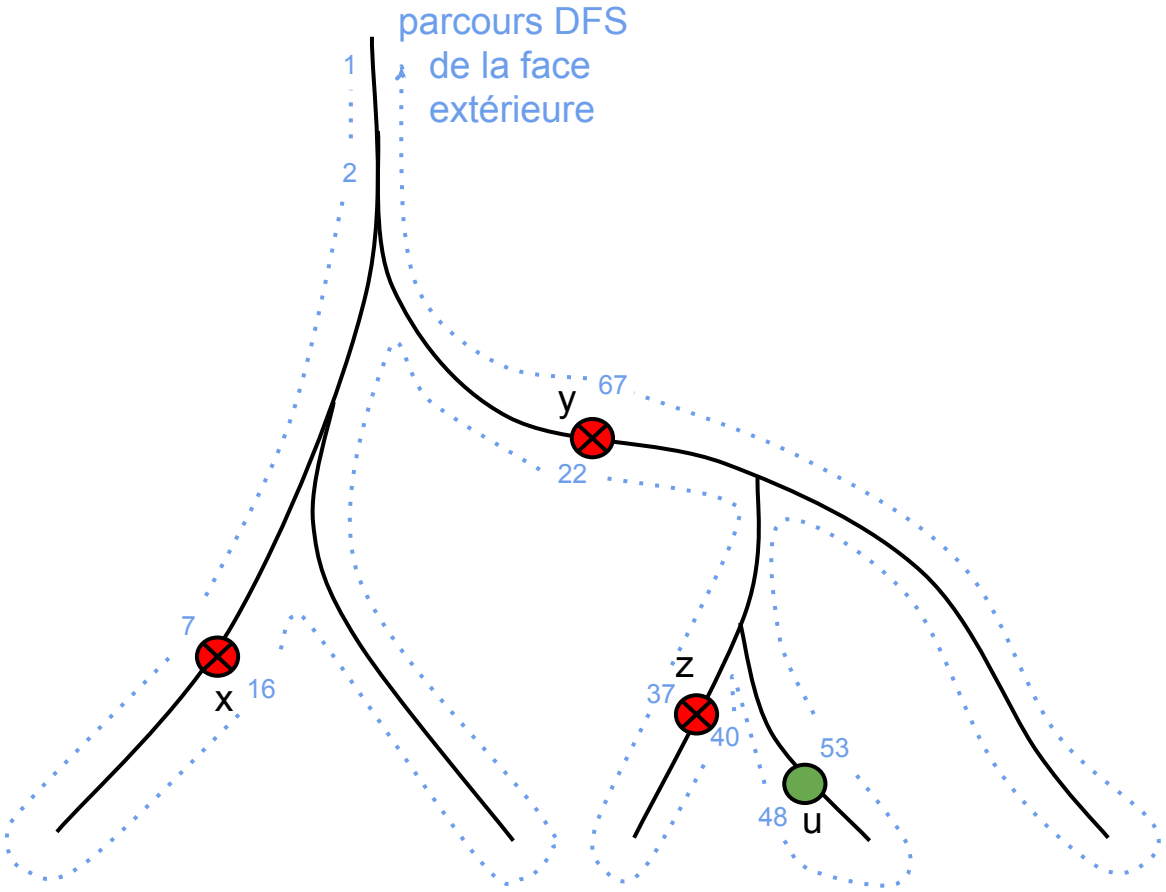
Problème 6 : Plus Proche Ancêtre
Entrée: Arbre $T = (V, E)$ enraciné en $\mathrm{r}(T)$, $X \subseteq V$, $u \in V \setminus X$
Sortie: $\textsc{prAnc}(u, X)$ le plus proche ancêtre de u dans l'arbre T parmi les sommets de X ou $\perp$ si aucun sommet de X n'est ancêtre de u

Proposition 9. Plus Proche Ancêtre *admet un schéma d'étiquetage d'urgence.*

- *Les étiquettes sont de taille* $O(\log n)$ *bits.*
- *La complexité d'étiquetage est linéaire.*
- *La complexité de pré-requête est* $\textsc{Sort}(|X|, n) + \textsc{PrePred}(|X|, n)$
- *La complexité de turbo-requête est* $O(\textsc{Pred}(|X|, n)$

Démonstration. Considérons un parcours en profondeur de l'arbre T au cours duquel on numérote les sommets visités selon l'ordre de passage. Chaque

FIGURE 8 – Parcours DFS de T pour PLUS PROCHE ANCÊTRE de u parmi $\{x, y, z\}$. Les sommets non concernés par la requête ne sont pas dessinés.



```
1: fonction DFSNum(T)
2:     c ← 1
3:     DFSNumAux(r(T), c)
4: fin fonction
5: fonction DFSNumAux(u, c par référence)
6:     α(u) ← c
7:     c ← c + 1
8:     pour tout x fils de u faire
9:         appeler récursivement DFSNumAux(x, c)
10:    fin pour
11:    ω(u) ← c
12:    c ← c + 1
13: fin fonction
```

Algorithme 4: DFS NUMBERS

sommet u reçoit un numéro $\alpha(u)$ de première visite et un numéro $\omega(u)$ de dernière visite. (*cf.* algorithme 4 et figure 8)

Par construction-même de cette numérotation, on pourra noter [6] que pour tout couple de sommets (u, v), les énoncés suivant sont équivalents :

— u est un ancêtre de v
— $[\![\alpha(v), \omega(v)]\!] \subset [\![\alpha(u), \omega(u)]\!]$

ainsi que ceux-ci :

— u et v sont tels qu'aucun n'est ancêtre de l'autre.
— $[\![\alpha(v), \omega(v)]\!] \cap [\![\alpha(u), \omega(u)]\!] = \emptyset$

Pour un ensemble $X = \{x_1, x_2, \ldots, x_k\}$, on peut considérer l'ensemble de $2k$ couples :

$$\left\{ \begin{pmatrix} \alpha(x_1) \\ x_1 \end{pmatrix}, \begin{pmatrix} \omega(x_1) \\ x_1 \end{pmatrix}, \begin{pmatrix} \alpha(x_2) \\ x_2 \end{pmatrix}, \begin{pmatrix} \omega(x_2) \\ x_2 \end{pmatrix}, \ldots, \begin{pmatrix} \alpha(x_k) \\ x_k \end{pmatrix}, \begin{pmatrix} \omega(x_k) \\ x_k \end{pmatrix} \right\}$$

et le trier dans l'ordre croissant de la première valeur pour obtenir une séquence de couples entier/sommet :

$$W(X) = \left(\begin{pmatrix} w_1 \\ x(w_1) \end{pmatrix}, \begin{pmatrix} w_2 \\ x(w_2) \end{pmatrix}, \ldots, \begin{pmatrix} w_{2k} \\ x(w_{2k}) \end{pmatrix} \right)$$

6. $[\![\cdot, \cdot]\!]$ est la notation adoptée pour repésenter un intervalle d'entiers.

Ce tableau est assimilable à un mot de parenthèses bien formé $z(X)$ où si, pour les entiers i et j tels que $i < j$ et $x(w_i) = x(w_j)$, les couples $\begin{pmatrix} w_i \\ x(w_i) \end{pmatrix}$ et $\begin{pmatrix} w_j \\ x(w_j) \end{pmatrix}$ constituent une paire de parenthèses associées.

Par exemple, un ensemble de sommet $X = \{x, y, z\}$ tel que :
- $I(x) = [\![\alpha(x), \omega(x)]\!] = [\![7, 16]\!]$
- $I(y) = [\![22, 67]\!]$
- $I(z) = [\![37, 40]\!]$

produira la séquence $W(X)$ et le mot de parenthèse $z(X)$ suivants :

$$W(X) = \begin{pmatrix} 7 \\ x \end{pmatrix}, \begin{pmatrix} 16 \\ x \end{pmatrix}, \begin{pmatrix} 22 \\ y \end{pmatrix}, \begin{pmatrix} 37 \\ z \end{pmatrix}, \begin{pmatrix} 40 \\ z \end{pmatrix}, \begin{pmatrix} 67 \\ y \end{pmatrix}$$

$$z(X) = \quad (\qquad) \qquad (\qquad (\qquad) \qquad)$$

Connaître le plus proche ancêtre x d'un sommet u parmi les sommets de X revient à trouver le plus petit intervalle $I(x)$ incluant $[\![\alpha(u), \omega(u)]\!]$.

Proposition 10. *On peut réduire cette question au problème* PRÉDÉCESSEUR

Problème 7 : PRÉDÉCESSEUR
Entrée: Un ensemble E (fourni trié) de n éléments parmi un univers $(U, \preceq)$ bien ordonné de taille N, x de type U
Sortie: k, plus grand élément de E tel que $k \preceq x$

Pour réaliser cette réduction, on crée un tableau $S(X)$ de la forme :

$$S(X) = \begin{pmatrix} 0 & w_1 & w_2 & \dots & w_{2|X|} \\ \perp & s(w_1) & s(w_2) & \dots & s(w_{2|X|}) \end{pmatrix}$$

où $s(w_i)$ est un identifiant du sommet $x \in X$ tel que $I(x)$ est le plus petit intervalle englobant l'intervalle $[\![w_i, w_{i+1}[\![$ (où $\perp$ si un tel intervalle n'existe pas), soit l'image inverse de w_i pour PRÉDÉCESSEUR.

Dans l'exemple présenté plus haut, on a :

$$S(X) = \begin{pmatrix} 0 & 7 & 16 & 22 & 37 & 40 & 67 \\ \perp & \mathrm{Id}(x) & \perp & \mathrm{Id}(y) & \mathrm{Id}(z) & \mathrm{Id}(y) & \perp \end{pmatrix}$$

On peut construire $S(X)$ en temps $O(|X|)$ (*cf.* algorithme 5).

On définit la relation d'ordre $\preceq$ (évaluable en temps constant) sur : $[\![\ 0,\ 2n\ ]\!]\ \times\ (X\ \cup\ \{\perp\})$ par :

$$\begin{pmatrix} w_i \\ Id(s_i) \end{pmatrix} \preceq \begin{pmatrix} w_j \\ Id(s_j) \end{pmatrix} \iff w_i \leq w_j$$

Remarque 11. *Si la taille des objets* $\begin{pmatrix} w_i \\ Id(s_i) \end{pmatrix}$ *est* $O(\log n)$ *bits, ils pourront éventuellement être «* transtypés par forçage *» (*i.e *« castés » pour utiliser l'anglicisme répandu) en objets de type* entier *sans que leur ordre ne soit modifié. On peut donc envisager des solutions algorithmiques pour le tri et pour* Prédécesseur *utilisant des manipulations astucieuses, dites* transdichotomiques *[12], non exclusivement basées sur des comparaisons ordinales.*

Soit une instance de Plus Proche Ancêtre d'entrée (u, X). La sortie du problème Prédécesseur pour l'entrée

$$\left(S(X), \begin{pmatrix} \alpha(u) \\ \perp \end{pmatrix}\right)$$

est de la forme

$$\begin{pmatrix} w \\ i \end{pmatrix}$$

où i est un identifiant du plus proche ancêtre de u parmi X (ou $\perp$ s'il n'existe pas un tel ancêtre). On peut donc utiliser ce i comme sortie pour Plus Proche Ancêtre. Notre problème est donc réduit.

Dans le cas particulier de l'exemple précédent (celui de la figure 8), si nous cherchons le Plus Proche Ancêtre d'un sommet u tel que $\alpha(u) = 48$ parmi les sommets de X, on a une entrée correspondante

$$\left(\begin{pmatrix} 0 & 7 & 16 & 22 & 37 & 40 & 67 \\ \perp & \mathrm{Id}(x) & \perp & \mathrm{Id}(y) & \mathrm{Id}(z) & \mathrm{Id}(y) & \perp \end{pmatrix}, \begin{pmatrix} 48 \\ \perp \end{pmatrix}\right)$$

1: **fonction** CONSTRUIRES(tableau W) ▷ avec $W_i = \begin{pmatrix} w_i \\ x(w_i) \end{pmatrix}$. Un sommet peut être codé de façon arbitraire, on peut l'imaginer représenté par une étiqueté compacte.
2: $S_0 = \begin{pmatrix} 0 \\ \perp \end{pmatrix}$
3: empiler $\perp$ dans la pile vide p
4: **pour** $i \in [\![1,\ 2 \cdot |X|]\!]$ **faire**
5: **si** $x(w_i) = \text{sommetPile}(p)$ **alors** ▷ Parenthèse fermante
6: dépiler p
7: $S_i = \begin{pmatrix} w_i \\ \text{sommetPile}(p) \end{pmatrix}$
8: **sinon** ▷ Parenthèse ouvrante
9: empiler $x(w_i)$ dans p
10: $S_i = \begin{pmatrix} w_i \\ x(w_i) \end{pmatrix}$
11: **fin si**
12: **fin pour**
13: **fin fonction**

Algorithme 5: CRÉATION DE S

pour PRÉDÉCESSEUR. La sortie correspondante est :

$$\begin{pmatrix} 40 \\ Id(y) \end{pmatrix}$$

on peut donc utiliser $Id(y)$ comme sortie pour le problème PLUS PROCHE ANCÊTRE.

On peut donc construire le schéma d'étiquetage (voir l'algorithme d'encodage 6 et les algorithmes de décodage 7 et 8). Ce qui prouve la proposition 9. □

1: **fonction** ÉTIQUETERPROCHEANCETRE(arbre $T = (V, E)$ à n sommets.)
2: DFSNum(T)
3: **pour tout** $s \in V$ **faire**
4: label(s)= $\langle \alpha(s), \omega(s) \rangle$ de taille $2 \log n$ bits.
5: **fin pour**
6: **fin fonction**

Algorithme 6: ÉTIQUETAGE PLUS PROCHE ANCÊTRE

1: **fonction** PREPROCPROCHEANCETRE($L(u) \cup L(X)$)
2: construire trivialement et trier $w(X)$ en temps SORT($|X|, n$)
3: construire $S(X)$ en appelant CONSTRUIRES($w(X)$) en temps $O(|X|)$
4: construire l'éventuelle (selon choix algorithmique) structure de donnée relative au prétraitement de $S(X)$ en vue de requêtes PRÉDÉCESSEUR, en temps PREPRED($|X|, n$)
5: **fin fonction**

Algorithme 7: PRÉ-REQUÊTE PLUS PROCHE ANCÊTRE

1: **fonction** TURBOQUERYPROCHEANCETRE($S(X), label(u)$)
2: trouver $\begin{pmatrix} w \\ i \end{pmatrix}$, le prédécesseur de $\begin{pmatrix} \alpha(u) \\ \perp \end{pmatrix}$ dans $S(X)$ en temps PRED($|X|, n$)
3: **retourner** i
4: **fin fonction**

Algorithme 8: TURBO-REQUÊTE PLUS PROCHE ANCÊTRE

2.2.2 Schéma auxiliaire : ROUTAGE pour la famille des arbres

Soit le problème suivant :

Problème 8 : ROUTAGE
Entrée: Un graphe $G = (V, E)$, $(s, t) \in V^2$
Sortie: NEXTHOPPORT(s, t) le port à emprunter depuis s pour aller de s à t par le plus court chemin.

Théorème 12 (Fraigniaud et Gavoille [25] Thorup et Zwick [57])**.** *Il existe, pour la famille des arbres à n sommets, des schémas d'étiquetage répondant au problème* Routage*. Ils peuvent être vu comme des schémas d'étiquetage d'urgence tel que :*

- *Les étiquettes sont de taille $O(\log n)$ bits*
- *La complexité d'étiquetage est linéaire*
- *La complexité de pré-requête est 0 (problème sans panne)*
- *La complexité de turbo-requête est $O(1)$, le port récupéré est un entier entre 1 et le degré du sommet source.*

2.2.3 Schéma principal : Composante Connexe Avec Panne pour la famille des arbres

Proposition 13. *On peut réduire, dans la famille des arbres, le problème* Composante Connexe Avec Panne *à une composition de* Plus Proche Ancêtre *et de* Routage *dans un nouvel arbre T' augmenté d'un sommet ρ, on a :*

$$\textsc{CompConn}_T(u, X) =$$

$$\Big(\textsc{prAnc}_{T'}(u, X \cup \{\rho\}), \textsc{nextHopPort}_{T'}\big(\textsc{prAnc}_{T'}(u, X \cup \{\rho\}), u\big)\Big) .$$

Démonstration. Soit un arbre $T = (V, E)$ auquel on ajoute un sommet *super-racine* ρ, déclaré nouvelle racine, relié uniquement à la racine de T. On obtient ainsi l'arbre T'.

Soit une panne $X \subseteq V$ à laquelle on ajoute également ρ. On constate que pour tout $u \in V \setminus X$, l'arête la plus haute dans T' (*i.e* la plus proche de ρ) de la composante connexe de u est l'arête e incidente à $\textsc{prAnc}_{T'}(u, X \cup \{\rho\})$ située sur le plus court chemin entre $\textsc{prAnc}_{T'}(u, X \cup \{\rho\})$ et u. Identifier e revient à identifier la composante connexe recherchée elle même. Or e est identifiée de façon non équivoque par le couple :

$$\Big(\textsc{prAnc}_{T'}(u, X \cup \rho), \textsc{nextHopPort}_{T'}\big(\textsc{prAnc}_{T'}(u, X \cup \{\rho\}), u\big)\Big) .$$

□

On obtient donc le schéma d'étiquetage du théorème 8 en concaténant les étiquettes des schémas des proposition 9 et 12 appliqués à l'arbre T' de la démonstration précédente. On a :

$$\textsc{Label}(x) = \langle \alpha(x), \omega(x), \lambda(x) \rangle$$

avec $\lambda(x)$ l'étiquette de routage du sommet x. A noter que $\lambda(x)$ est utilisé comme identifiant $\mathrm{Id}(x)$ pour le schéma de Plus Proche Ancêtre. L'étiquette $\lambda(\textsc{prAnc}(u, X))$ est donc disponible au moment de l'éventuelle requête de Routage qui nous donne $\textsc{nextHopPort}(\textsc{prAnc}(u, X), u)$.

Sachant que le schéma de Routage proposé dans [57] construit des étiquettes de taille $\log(n) + o(\log n)$, notre schéma admet des étiquettes de taille $3 \cdot \log n + o(\log n)$.

Les requêtes de routage étant traitées en temps constant, le temps de requête de notre schéma est toujours $O(\textsc{Pred}(|X|, n))$.

□

2.3 Schéma Composante Connexe pour les planaires extérieurs

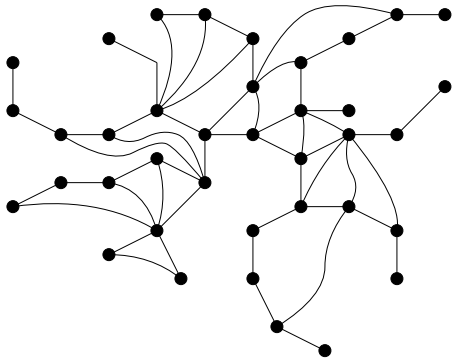
Un graphe est dit *planaire* s'il peut être *plongé*, c'est à dire *immergé*[7] (ou encore *dessiné*) sans croisement d'arêtes dans le plan. Un graphe planaire est dit planaire *extérieur* s'il en existe un plongement tel que tous les sommets du graphe sont incidents à la face extérieure. Voir premier dessin de la figure 9.

Théorème 14. *Tout graphe planaire extérieur à n sommets admet un schéma d'étiquetage d'urgence pour* Composante Connexe Avec Panne *tel que :*

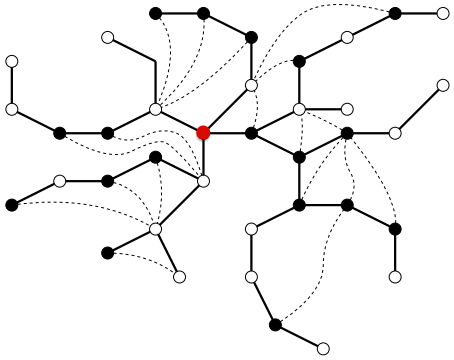
- *Les étiquettes sont de taille $O(\log n)$ bits.*
- *La complexité d'étiquetage est linéaire.*
- *La complexité de pré-requête est* $\textsc{Sort}(|X|, n) + \textsc{PrePred}(|X|, n)$.
- *La complexité de turbo-requête est* $O(\textsc{Pred}(|X|, n))$.

7. A chaque sommet u est associé un point $p(u)$ du plan P et à chaque arrête $e = (x, y)$, on associe $p(e) = p_e([0, 1])$ où $p_e : [0, 1] \to P$ est une fonction continue telle que $p_e(0) = x$ et de $p_e(1) = y$.

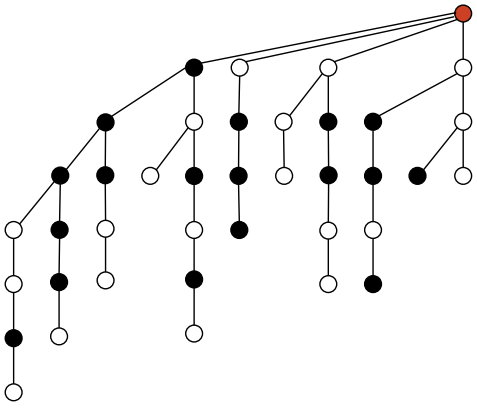
FIGURE 9 – Graphes-étapes de la construction du schéma pour les graphes planaires extérieurs.



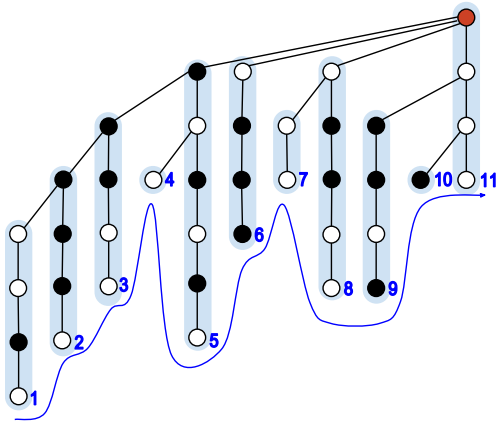
1. *G graphe planaire extérieur dessiné sur le plan.*



2. *T et c génèrent G*

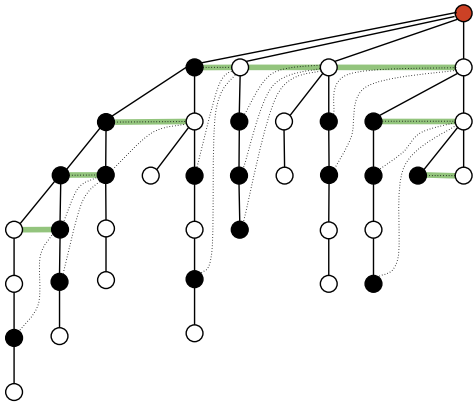


3. *T vu comme arbre planté*

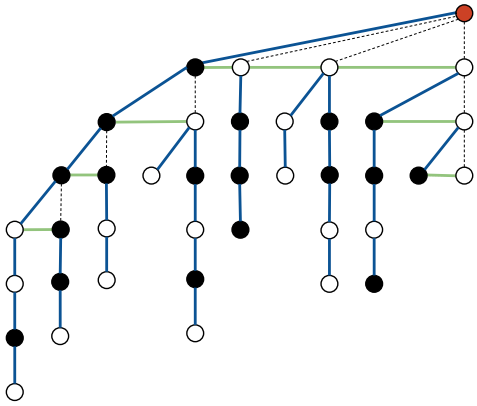


4. *Branches dans T*

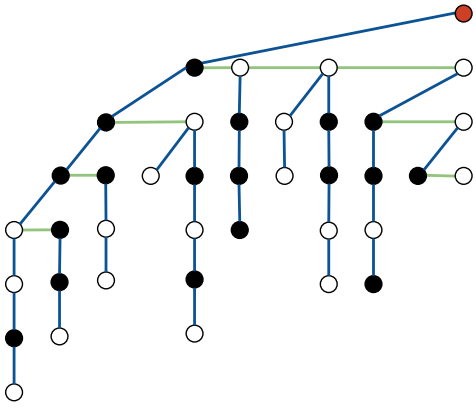
FIGURE 10 – Graphes-étapes de la construction du schéma pour les graphes planaires extérieurs. (suite)



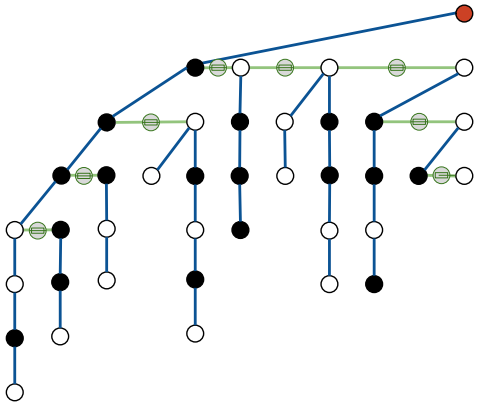
1. *Construction des arêtes témoins de type E*



2. *On conserve les arêtes de type F*



3. $\tilde{T}'$ *est un arbre*



4. T' *et ses sommets-témoins*

Résumé de la preuve. Pour construire ce schéma, on s'appuie sur une construction proposée par [9] décomposant un graphe planaire extérieur G en un arbre et un ensemble d'arêtes supplémentaires. A partir de cette construction, on crée un autre arbre auxiliaire $T'(G)$ nous permettant de réduire le problème posé dans les graphes planaires extérieurs à une généralisation de celui traité en 2.2 dans les arbres.

Démonstration. Soit G un graphe planaire extérieur à n sommets.

2.3.1 Construction auxiliaire : $T(G)$

De l'arbre T au graphe planaire extérieur G. Considérons un arbre (figure 9-2) enraciné[8] $T = (V, E)$ dont une carte planaire nous est fournie ; *i.e.* on considère ordonné l'ensemble des fils de chaque sommet. On dit alors que l'arbre est *planté*. On peut numéroter les feuilles de l'arbre[9], de gauche à droite (on considère que la racine est en haut[10]). On note $l(u)$ le numéro attribué à u si u est une feuille. $l(u) = \perp$ sinon.

A partir de la fonction l, on peut définir une *branche* de T comme étant une classe d'équivalence de la fonction b définie récursivement ainsi :

$$b(u) = \begin{cases} l(u) \text{ si } l(u) \neq \perp \\ b(d(u)) \text{ sinon.} \end{cases}$$

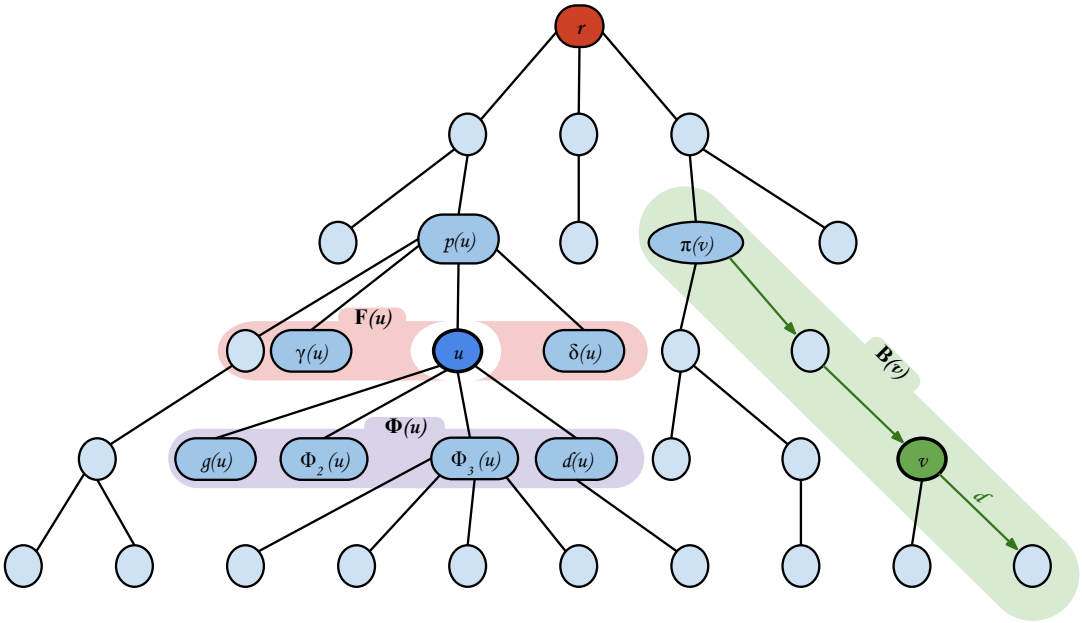
où $d(u)$ est le fils droit, c'est à dire le dernier fils, de u. On note $B(u)$ la branche du sommet u.

On définit une bonne fois pour toutes (voir figure 11) les notations des relations de parenté dans l'arbre T que nous utiliserons :

- $g(u)$ et $d(u)$ sont les fils gauche et fils droit du sommet u
- $\gamma(u)$ et $\delta(u)$ sont les frère gauche et frère droit du sommet u
- $\Phi(u)$ est l'ensemble des fils de u dont le i-ème est noté[11] $\Phi_i(u)$
- $F(u)$ est l'ensemble des frères de u ($u \notin F(u)$)
- $p(u)$ est le père du sommet u.

8. La racine est le sommet rouge sur la figure 9-2 .
9. Le sommet le plus bas du dessin 9-2 est la première feuille.
10. Le sens gauche→droite est donc le sens trigonométrique autour de la racine sur le dessin 9-2.
11. « *Tu quoque mi* Φ_i. »

FIGURE 11 – Parenté dans un arbre



— $\pi(u)$ est le plus haut sommet de la branche de u

Ces fonctions prennent la valeur $\perp$ lorsqu'elle ne sont pas définies. Par exemple, si r est le sommet racine : $p(r) = \perp$. Et si f est une feuille, on a : $\Phi(f) = \emptyset$ et donc $\Phi_3(f) = \perp$.

Considérons c une bicoloration des sommets de T, disons en noir et blanc, telle que tous les sommets de la dernière branche sont blancs [12]. On dira d'une telle coloration qu'elle est une *bonne coloration* de T, ou encore que $c(T)$ est un arbre *bien* coloré.

Considérons maintenant le graphe P construit à partir de $c(T)$ en ajoutant, pour chaque sommet noir u, une *arête de passage* $a(u)$ unissant le

12. On n'exige pas une coloration propre.

sommet u et son *sommet de passage* $\sigma(u) = \delta(\pi(u))$, soit le frère droit du plus haut sommet de la branche de u. Nous nous permettront d'écrire, par commodité, que $\sigma(B(u)) = \sigma(u)$ étant donné que deux sommets de la même branche admettent le même *sommet de passage*. (Voir les arêtes en pointillés des figures 9-2 et 10-1.)

On dira que l'arbre planté T et la coloration c *génèrent* le graphe P.

Théorème 15 (Bonichon, Gavoille, et Hanusse [9])**.** *Un graphe P ainsi généré est un graphe planaire extérieur. Il y a une bijection sous-jacente entre l'ensemble des arbres bien coloriés et celui des graphes planaires extérieur.*

Du graphe planaire extérieur G à l'arbre $T(G)$.

Théorème 16 ([9])**.** *On peut construire à partir d'un graphe G planaire extérieur, à n sommets, en temps $O(n)$, un arbre planté T et une bonne coloration associée c qui génèrent G.*

2.3.2 Construction auxiliaire : $T'(G)$

Soit un couple $\big(T, c\big)$ qui génère G comme défini dans le théorème 16.

On construit un premier graphe auxiliaire intermédiaire $\tilde{T'}$ (Figure 10-3).

Nous obtenons $\tilde{T'}$ depuis T en détachant un sommet u de son père $p(u)$ si u reçoit au moins une arête de passage de la part de la branche de son frère gauche $\gamma(u)$. Dans ce cas, on remplace l'arête supprimée $\big(p(u), u\big)$ par une arête $\big(\gamma(u), u\big)$ dite *témoin* de ces arêtes de passage.

Plus formellement, $\tilde{T'} = (V, E \cup\ F)$ avec :

— $V = V(T)$

— L'ensemble E est construit ainsi :
pour tout x ayant k fils (dans T, les relations de parenté sont considérées dans T uniquement, sauf indication contraire explicite), pour tout $i \in [\![1, k-1]\!]$:

$\big(\Phi_i(x), \Phi_{i+1}(x)\big) \in E \iff \Phi_i(x)$ et $\Phi_{i+1}(x)$ sont connectés dans $G \backslash \{x\}$.

Intuitivement, on peut se dire qu'une arête $e = \big(\Phi_i(x), \Phi_{i+1}(x)\big)$ élément de l'ensemble E *témoigne* de l'existence d'au moins une arête de passage issue d'une branche donnée : celle dont $\Phi_i(x)$, qui n'est pas un fils droit, est le plus haut sommet.

— L'ensemble F est construit ainsi :
pour tout x ayant k fils dans T, pour tout $i \in [\![1, k]\!]$:

$$\big(x, \Phi_i(x)\big) \in F \iff i = 1 \vee \big(\Phi_{i-1}(x), \Phi_i(x)\big) \notin E \ .$$

Intuitivement, on peut se dire qu'une arête de F permet de garder un sommet connecté à son père après modification, s'il ne l'est pas *via* ses frères gauches, par une arête de type E.

A partir du graphe $\tilde{T'}$, on construit le graphe $T' = (V \cup W,\ E' \cup F)$ en subdivisant toutes les arêtes de E. (Figure 9-4)

On obtient un ensemble W de *sommets témoins* intermédiaires[13] tels que $e = \big(\Phi_i(x), \Phi_{i+1}(x)\big)$ est remplacée par $e'_1 = \big(\Phi_i(x), w_i(x)\big)$ et $e'_2 = \big(w_i(x), \Phi_{i+1}(x)\big)$. On dit que $w_i(x)$ témoigne pour la branche du i-ème fils de x. On notera donc $w_i(x) = w(\Phi_i(x)) = w(B(\Phi_i(x)))$.

Proposition 17. *[9] T' est un arbre.*

Il a semblé nécessaire à l'auteur d'expliciter la preuve à venir, mais elle est fastidieuse. Sa lecture pourra facilement être omise par le lecteur convaincu, à la lumière de la 10, que T' est un arbre.

Démonstration. Un subdivision d'arête ne pouvant créer de cycle ni déconnecter le graphe, prouver que T' est un arbre revient à prouver que $\tilde{T'}$ en est un.

On prouve d'abord que $\tilde{T'}$ est connexe.

Par construction, tout sommet u de $\tilde{T'}$ est voisin, soit (alternative exclusive) de $\gamma(u)$, qui est son frère gauche dans T, soit de $p(u)$, qui est son père dans T.

13. Que l'on utilisera comme fusibles.

Supposons par l'absurde que u est $p(u)$ ne sont pas connectés dans $\tilde{T'}$. Alors il ne sont, *a fortiori*, pas voisins. u est donc voisin de $\gamma(u)$. $\gamma(u)$ n'est donc pas connecté à $p(u)$, ni non plus, par une récurrence immédiates, aucun des sommets frères de u à gauches de celui-ci. Le premier frère de u, $\Phi_1(p(u))$ n'est donc pas connecté à $p(u)$, et *a fortiori* pas voisin de $p(u)$, ce qui est contradictoire, par construction de l'ensemble des arêtes de l'ensemble F.

On en déduit que tout sommet x est connecté à $p(x)$, et donc par récurrence, que tout sommet est connecté à la racine. $\tilde{T'}$ est donc connexe.

On prouve maintenant par induction structurelle que $\tilde{T'}$ est un arbre.

Pour tout sommet u, notons $\tilde{T'}[u]$ le sous graphe de $\tilde{T'}$ induit par la composante connexe de u dans le graphe $\tilde{T'}$ privé des éventuelles arêtes-témoins incidentes à u et à ses frères, et de celle qui pourrait relier u à son père. (On peut voir $\tilde{T'}[u]$ comme la partie de $\tilde{T'}$ située « sous u » et sur laquelle on appliquera l'hypothèse d'induction ascendante (*bottom-up*).)

Soient x et y deux sommets frères de $\tilde{T'}$. Supposons que pour toute paire de sommet (z,t) telle que $p(z), p(t) \in \{x,y\}$ on ait :

$$\tilde{T'}[z] \text{ et } \tilde{T'}[t] \text{ sont des arbres et n'ont pas de sommet en commun .}$$

Montrons alors par l'absurde que $\tilde{T'}[x]$ et $\tilde{T'}[y]$ sont aussi des arbres dont l'intersection est réduite à l'ensemble vide.

On suppose que $\tilde{T'}[x]$ possède un cycle $\mathcal{C}$. L'hypothèse d'induction impose que $\mathcal{C}$ est réductible à un cycle sur les sommets de $\{x\} \cup \Phi(x)$ de la forme :

$$\mathcal{C}' = (x, \Phi_i(x), \Phi_{i+1}(x), \cdots, \Phi_{j-1}(x), \Phi_j(x), x) \ .$$

Ce qui impose que le sommet $\Phi_j(x)$ est voisin à la fois de son frère gauche et de son père, ce qui est contradictoire. $\tilde{T'}[x]$ est donc un arbre. De même, on prouve que $\tilde{T'}[y]$ est un arbre.

De plus, par construction, on n'a pas pu créer d'intersection entre $\tilde{T'}[x]$ et $\tilde{T'}[y]$.

On peut initialiser trivialement l'induction pour les sommets qui sont des feuilles dans T, et ainsi conclure la preuve de la proposition 17. □

2.3.3 Composante Connexe avec Panne et Réparation

On ne peut *a priori* réduire le problème sur les graphes planaires extérieurs à celui sur les arbres[14], par l'intermédiaire de T', sans faire appel à des réparations *ad-hoc* des pannes de l'arbre.

On doit donc, pour la famille des arbres, être capable de traiter un problème un peu plus difficile :

Problème 9 : Composante Connexe avec Panne et Réparation
Entrée: Graphe G, sous-ensemble X des sommets de G dits *en panne*, ensemble Y d'arêtes de $V(G) \times V(G)$ dites *de réparation*, sommet u
Sortie: Un identifiant de la composante connexe de u dans $(G \setminus X) \cup Y$

Les arêtes de réparations peuvent être traitées en utilisant une structure de données pour le problème Trouver et Unir[15]. On considèrera les identifiants des composantes connexes de $G \setminus X$ comme ceux d'ensembles disjoints atomiques qu'on pourra réunir en classes s'ils sont connectés par les arêtes de réparations.

Lemme 18. Composante Connexe Avec Panne et Réparation *admet un schéma d'étiquetage d'urgence pour la famille des arbres à n sommets.*

- *Les étiquettes sont de taille $O(\log n)$ bits.*
- *La complexité d'étiquetage est linéaire.*
- *La complexité de pré-requête est*

$$\textsc{Sort}(|X|, n) + \textsc{PrePred}(|X|, n) + O(|Y| \cdot \textsc{Pred}(|X|, n)) + \textsc{MakeSet}(|Y|)$$

- *La complexité de turbo-requête est* $O(\textsc{Pred}(|X|, n) + \textsc{SetFind}(|Y|)$

Où :

- $\textsc{SetUnion}(k)$ *est le temps nécessaire à la création d'une structure de données de recherche pour le problème* Union-Find *en k opérations* MakeSet, Find *et* Union. *Et où*

14. Ce qui sera l'objet de la proposition 20.

15. Ou Union and Find.. Il s'agit d'une structure de données permettant de construire et de maintenir des classes d'équivalences via les opérations MakeSet, qui crée une classe sous forme d'un singleton, et Union qui permet d'unir deux classes. On peut ensuite connaître la classe d'équivalence d'un objet grâce à l'opération Find.

— SetFind(k) *est le temps nécessaire à la réalisation d'une requête* Find *dans cette structure de donnée.*

Démonstration. On construit un schéma à partir du schéma pour le problème sans réparations :

1. On étiquette l'arbre passé en entrée pour le problème Composante Connexe Avec Panne sans réparation.
2. On effectue la pré-requête selon l'algorithme 9.
3. On effectue la requête selon l'algorithme 10.

```
1: fonction PREPROCCC((LABEL(x))_{x∈X}, (LABEL(y_1 1), LABEL(y_2 2))_{y∈Y})
2:     Effectuer la pré-requête de (LABEL(x))_{x∈X} pour COMPOSANTE
   CONNEXE AVEC PANNE ▷ en temps SORT(|X|, n) + PREPRED(|X|, n)
3:     pour tout y ∈ Y faire
4:         c_1 = COMPCONN(y_1, X)
5:         c_2 = COMPCONN(y_2, X)
6:         si DSFIND(c_1 = ⊥) alors
7:             DSMAKESET(c_1)
8:         fin si
9:         si DSFIND(c_2 = ⊥) alors
10:            DSMAKESET(c_2)
11:        fin si
12:        DSUNION(DSFind(c_1), DSFind(c_2))
13:    fin pour ▷ Boucle en temps O(|Y| · PRED(|X|, n)) + MAKESET(|Y|)
14: fin fonction
```

Algorithme 9: Pré-requête Composante Connexe Avec Panne et Réparation pour les arbres à n sommets

```
1: fonction CompConn(Label(u))
2:     c = CompConn(u, X)
3:     c' = DSFind(c)
4:     si c' = ⊥ alors
5:         retourner c
6:     sinon
7:         retourner c'
8:     fin si
9: fin fonction
```

Algorithme 10: Requête Composante Connexe Avec Panne et Réparation pour les arbres à n sommets

□

Corollaire 19. Composante Connexe Avec Panne et Réparation *admet un schéma d'étiquetage d'urgence pour la famille des arbres à n sommets.*

- *Les étiquettes sont de taille $O(\log n)$ bits.*
- *La complexité d'étiquetage est linéaire.*
- *La complexité de pré-requête est*

$$\textsc{Sort}(|X|, n) + \textsc{PrePred}(|X|, n) + O(|Y| \cdot \textsc{Pred}(|X|, n))$$

- *La complexité de turbo-requête est $O(\textsc{Pred}(|X|, n)$*

Démonstration. D'après [26], on peut avoir des complexités :

- $\textsc{SetUnion}(k) = O(k \cdot \alpha(k))$
- $\textsc{SetFind}(k) = O(\alpha(k))$

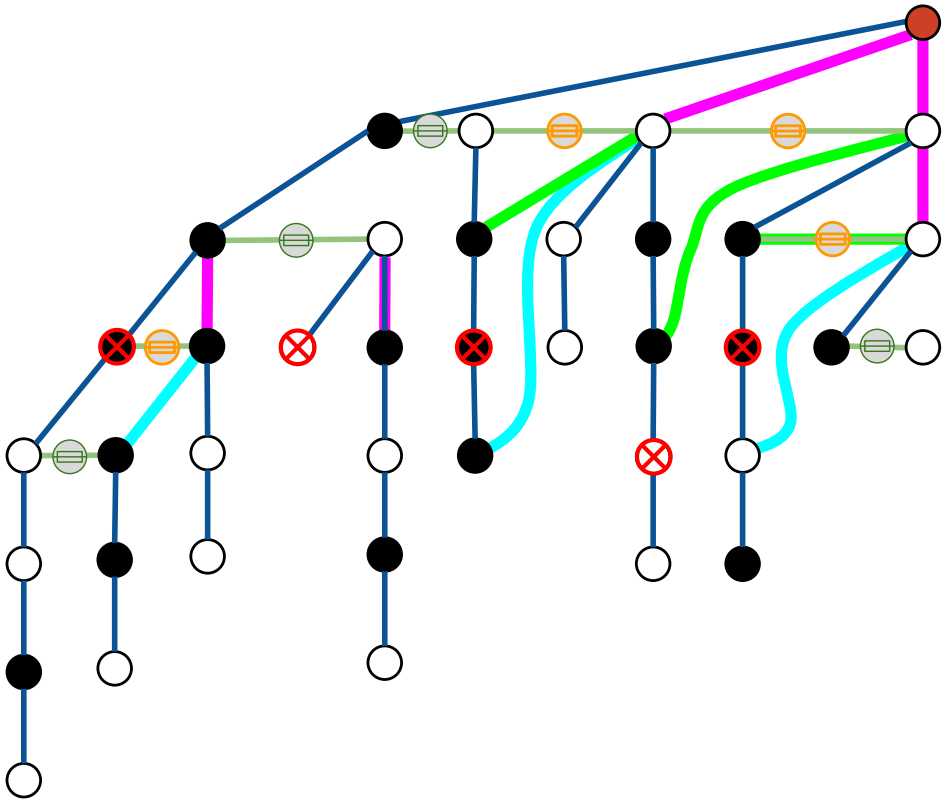
où α est la *fonction inverse d'Ackermann*, fonction qui croit extrêmement lentement [16] et qui est négligeable devant $O(\textsc{Pred}(|X|, n)$. □

2.3.4 Réduction

Proposition 20. *Le problème* Composante Connexe avec Panne *dans le graphe planaire extérieur G se réduit à* Composante Connexe avec Panne et Réparation *dans l'arbre $T'(G)$.*

16. $\alpha(n) \leq 4$ Pour tout n *de taille pratique.*

FIGURE 12 – Panne et réparation dans T'. Les sommets témoins de Δ sont en orange. Les arêtes de réparation de ∇, des trois types, sont en traits épais, et, dans l'ordre de définition, en rose, vert et bleu.



Démonstration. On pose (Voir figure 12.) :

— pour tout $X \subseteq V(G), \Delta(X) = \{w(B(x)) \mid x \in X\}$. Cette panne supplémentaire neutralise les arêtes-témoins de passages des branches impactées, en touchant les sommets qui les subdivisent.

— pour tout $X \subseteq V(G), \nabla(X)$ contient chaque arête (dans la mesure où elle est définie et qu'aucune de ses extrémités n'est en panne) de la forme :

— $\big(p(\pi(x)), \sigma(x)\big)$ où $x \in X$. Elle sert à reconnecter $\sigma(x)$, le frère droit du sommet en haut de branche du sommet en panne, à son père, après la mise en panne du témoin. En effet, si $\sigma(x)$ était connecté à son père $p(\sigma(x)) = p(\pi(x))$ dans T' via x, cette arête de réparation permet de rétablir la connexion que la panne de x à détruite dans T'. Sinon, cette arête existe déjà est elle est donc inutile, mais pas gênante.

— $\big(p(x), \sigma(x)\big)$ où $x \in X$ et si $p(x) \in B(x)$ (on rappelle que $B(u)$ est l'ensemble des sommets de la branche numéro $b(u)$) est tel qu'il existe un sommet noir au sein de la composante connexe de $p(x)$ dans $B(x) \setminus X$. Un tel sommet noir justifie de reconnecter une partie de la branche au sommet de passage après que la connexion a été détruite lors de la mise en panne de $w(b(x))$.

— $\big(d(x), \sigma(x)\big)$ où $x \in X$ et s'il existe un sommet noir au sein de la composante connexe de $d(x)$ dans $B(x) \setminus X$. Un tel sommet noir justifie de reconnecter une partie de la branche au sommet de passage, de même que précédemment.

Il s'avère[17] que l'arbre en panne $T'(G)$, impacté par la panne $X \cup \Delta(X)$, puis réparé par $\nabla(X)$, a les même propriétés de connexion que le graphe planaire extérieur en panne G subissant la panne X. Autrement dit :

$$x \in G \setminus X \iff x \in (T'(G) \setminus (X \cup \Delta(X))) \cup \nabla(X) \ .$$

L'entrée (u, X) du problème Composante Connexe avec Panne dans le graphe planaire extérieur G se réduit donc à l'entrée $(u, X \cup \Delta(X), \nabla(X))$ du problème Composante Connexe avec Panne et Réparation dans l'arbre $T'(G)$.

17. Démonstration triviale par induction structurelle des feuilles vers la racine de $T(G)$.

On constatera facilement que $|V(T'(G))| \leq 2 \cdot n$, que $|\Delta(X)| \leq |X|$ et que $\nabla(X) \leq 2 \cdot |X|$, ce qui revient à affirmer que les paramètres d'analyse de complexité avant et après la réduction sont de tailles asymptotiquement équivalentes.

□

Cela étant dit, il reste à montrer comment, à partir des étiquettes des sommets de X qui nous sont fournies, on peut obtenir, en temps convenable, pendant la pré-requête, les étiquettes des sommets de $\Delta(X)$ et impliqués dans la réparation $\nabla(X)$.

Proposition 21. *On peut modifier le schéma d'étiquetage compact d'urgence pour* Composante Connexe avec Panne *dans les arbres, de telle manière qu'il soit toujours un schéma d'étiquetage compact d'urgence dont les complexités et dont la taille des étiquettes sont équivalentes, et de telle sorte que l'on puisse calculer, en temps $O(|X|)$, les étiquettes des sommets de l'ensemble $\Delta(X)$ et des extrémités des arêtes de $\nabla(X)$*

Démonstration. On pose pour tout $u \in V(G)$,

$$\Lambda(u) = \{u, p(u), d(u), \sigma(u), \pi(u), p(\pi(u)), w(b(u))\} \ .$$

Il est évident que si l'on a en main toutes les étiquettes des sommets de $\cup_{x \in X} \Lambda(x)$, on peut obtenir en temps X tous les sommets intervenant dans $\Delta(X)$ et $\nabla(X)$

Lemme 22. *On peut décider en temps $O(X)$, pour tout $u \in p(X) \cup d(X)$, de la propriété $P(u)$: « Il existe un sommet noir dans la composante connexe du sommet u dans $B(u) \setminus X$. » à partir des étiquettes des sommets de $\Lambda(X)$ à condition de les enrichir chacune de $1 + 4 \log n$ bits d'information.*

Démonstration. Soient les fonctions :

— $\iota : V \to \mathbb{N}$ qui à un sommet associe un entier indiquant sa position dans la branche. On a donc $\forall x$, $\iota(\pi(x)) = 1$ et $\iota(d(x)) = 1 + \iota(x)$

— $\mu : V \to \mathbb{N}$ qui à un sommet u associe le nombre de sommet noirs au dessus de lui dans sa branche :

$$\mu(u) = |\{v \in B(u) \mid c(v) = noir \wedge \iota(v) < \iota(u)\}|$$

— $\nu : V \to \mathbb{N}$ qui à un sommet u associe le nombre de sommet noirs en dessous de lui dans sa branche :

$$\nu(u) = |\{v \in B(u) \mid c(v) = noir \wedge \iota(v) > \iota(u)\}|$$

Considérons un schéma d'étiquetage $\mathcal{S}^+$ pour Composante Connexe avec Panne dans les arbres où pour tout sommet x l'étiquette est définie par :

$$\textsc{Label}^+(x) = \langle b(x), \iota(x), \textsc{Label}(x), \mu(x), \nu(x), c(x)\rangle$$

alors si les étiquettes des sommets en panne, vus comme éléments d'une séquence $(x_i)_{x_i \in X}$ triée [18] selon l'ordre lexicographique de $\big(b(x_i), \iota(x_i)\big)_{x_i \in X}$:

— On peut calculer $P(p(x_i))$, en temps constant, en effectuant quelques opérations arithmétiques élémentaires :

$$P(p(x_i)) \iff \begin{array}{c} i = 1 \ \wedge \ \mu(x_i) > 0 \\ \vee \\ \mu(x_i) - \big(\mu(x_{i-1}) + \mathbb{1}_{c(x_{i-1})=noir}\big) > 0 \end{array}$$

— ainsi que $P(d(x_i))$, en temps constant également :

$$P(d(x_i)) \iff \begin{array}{c} i = \max\{i | x_i \in B(x_i)\} \ \wedge \ \nu(x_i) > 0 \\ \vee \\ \nu(x_i) - \big(\nu(x_{i+1}) + \mathbb{1}_{c(x_{i+1})=noir}\big) > 0 \end{array}$$

18. On peut supposer la séquence triée car la pré-requête du schéma de base exige déjà un temps $\textsc{Sort}(|X|, n)$. On remarque au passage que les étiquettes sont *castable* en un type assimilable à un entier, on n'a pas non plus ici d'obstacle à un tri transdichotomique.

avec :

$$\mathbb{1}_P = \begin{cases} 1 & \text{si } P \text{ est vraie} \\ 0 & \text{sinon.} \end{cases}$$

Cette construction démontre le lemme 22. □

Les modifications apportées au schéma d'origine ne modifiant pas ses performances, (Les étiquettes demeurent compactes) la proposition 21 est bien démontrée. □

Les propositions 21, 20 et le corollaire 19 nous permettent de conclure la preuve du théorème 14. □

Les complexités temporelles restent inchangées par rapport au schéma dans la famille des arbres. L'étiquette de u est de taille $|\Lambda(u)| = 7$ fois celle d'une étiquette modifiée, soit $49 \log n + o(\log n)$, mais on peut les réduire à $25 \log n + o(\log n)$ en remarquant que l'utilisation du $\textsc{Label}_+(x)$ n'est nécessaire que si x est en panne et est dans une branche (pas sommet-témoin), c'est donc sa propre étiquette qui nous livre cette information supplémentaire.

L'étiquette est alors de la forme :

$$\textsc{Label}_G(u) = \left\langle \textsc{Label}_{+T'}(u), (\textsc{Label}_{T'}(v))_{v \in \Lambda(u) \setminus u} \right\rangle .$$

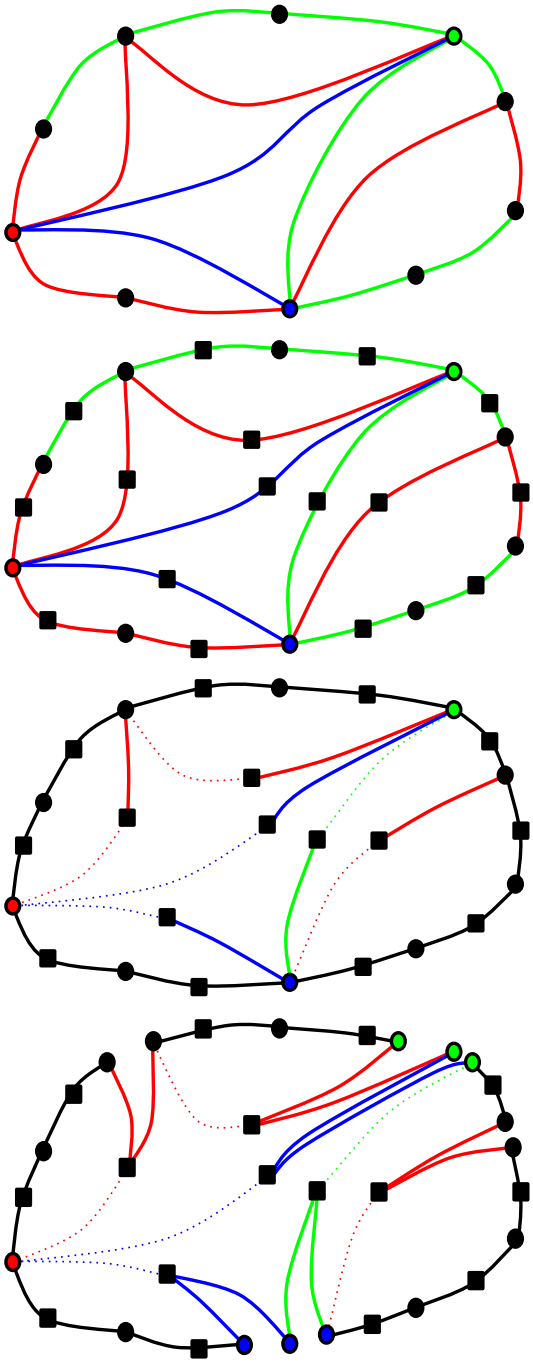
2.3.5 Cas 2-connexe et panne sur les arêtes

Nous présentons ici une réduction permettant d'étendre le théorème 14 au problème Composante Connexe Avec Panne sur Les Sommets et les Arêtes.

Théorème 23. *Tout graphe planaire extérieur 2-connexe à n sommets admet un schéma d'étiquetage d'urgence pour* Composante Connexe Avec Panne sur Les Sommets et les Arêtes *tel que :*

- *Les étiquettes sont de taille $O(\log n)$ bits.*
- *La complexité d'étiquetage est linéaire.*
- *La complexité de pré-requête est* $\textsc{Sort}(|X|, n) + \textsc{PrePred}(|X|, n)$.
- *La complexité de turbo-requête est* $O(\textsc{Pred}(|X|, n))$.

FIGURE 13 – Construction de G^+ et G' à partir de G muni d'un bois de Schnyder.



Démonstration. Soit G un graphe planaire extérieur 2-connexe à n sommets. On va construire en temps $O(n)$ un graphe G' planaire extérieur à $O(n)$ sommets tel qu'une requête $\textsc{Conn}(u, v, X, Y)$ dans G puisse être réduite à une requête $\textsc{Conn}'(u', v', X')$ dans G', avec $|X'| = O(|X| + |Y|)$. On pourra s'aider de la figure 13 pour suivre la construction.

G etant un graphe 2-connexe, il n'a pas de point d'articulation. On peut en déduire que sa face extérieur est bordée par un cycle sans répétition de sommets. G est donc l'union d'un cycle et de cordes de ce cycle qui ne se croisent pas.

G étant planaire, il admet un bois de Schnyder[19] qu'on peut construire en temps $O(n)$. On peut considérer chaque arbre de chaque forêt du bois de schnyder comme étant enracinée.

Considérons maintenant G^+ le graphe obtenu en subdivisant chacune des arêtes $x - z$ de G en deux arêtes $x - y - z$. Il est clair que la requête $\textsc{Conn}(u, v, X, Y)$ peut être réduite à une requête $\textsc{Conn}^+(u, v, X \cup \tilde{Y})$ dans G^+ où $\tilde{Y}$ est l'ensemble des sommets nés dans G^+ des arêtes de G. Cela dit, G^+ n'est pas planaire extérieur et on ne peut pas appliquer le théorème 14 directement.

Considérons maintenant C un sous graphe de G^+ tel que $V(C) = V(G^+)$ et que $E(C)$ contient

— toutes les arêtes issue du cycle extérieur de G ;
— une arête sur les deux issues de chaque corde de G. On choisira toujours celle des deux étant la plus éloignée de la racine de l'arbre de Schnyder auquel elle appartient.

On note que, par construction, C est de degré maximum 5.

Considérons un disque topologique $\mathcal{D}$ sur lequel est plongé G^+ est dont le bord correspond au cycle extérieur de G. Coupons[20] $\mathcal{D}$ selon les arêtes de C qui ne sont pas sur le bord. On obtient un nouveau disque $\mathcal{D}'$ et la carte

19. Ses arêtes peuvent être couverte par trois forêts. Voir chapitre 1.
20. Les arêtes et les sommets du bord concernés sont dupliqués lors de la coupe.

topologique du nouveau graphe G'.

G' est un graphe planaire extérieur. Chaque arête e de G a donné naissance à un ensemble $R(e)$ d'arête de G^+. $|R(e)| \leq 2$. Chaque sommet s de G^+ a donné naissance à un ensemble $R(s)$ d'arête de G'. $|R(s)| \leq 4$.

Une instance $\text{Conn}^+(u, v, X \cup \tilde{Y})$ du problème Composante Connexe Avec Panne sur Les Sommets et les Arêtes pour le graphe G^+ est équivalente à une instance[21] $\text{Conn}^+(R_1(u), R_1(v), R(X \cup \tilde{Y}))$ Composante Connexe Avec Panne sur Les Sommets sur le graphe planaire exterieur G'. Le théorème 14 s'applique. □

2.4 Méta-schéma auxiliaire

Nous avons construit dans les paragraphes précédents (Partie 2.3) un schéma d'étiquetage d'urgence pour le problème Composante Connexe avec Panne pour la famille des graphes planaires extérieurs à n sommets, tel que nous l'annoncions dans le théorème 14. Nous ne perdons pas de vue qu'il s'agit là d'une étape dans la démonstration du théorème 7 traitant d'un schéma similaire pour la famille des graphes 3-connexes de genre eulérien g. Nous allons ci-dessous présenter un méta-schéma auxiliaire qui constitue l'étape suivante de cette preuve en trois actes.

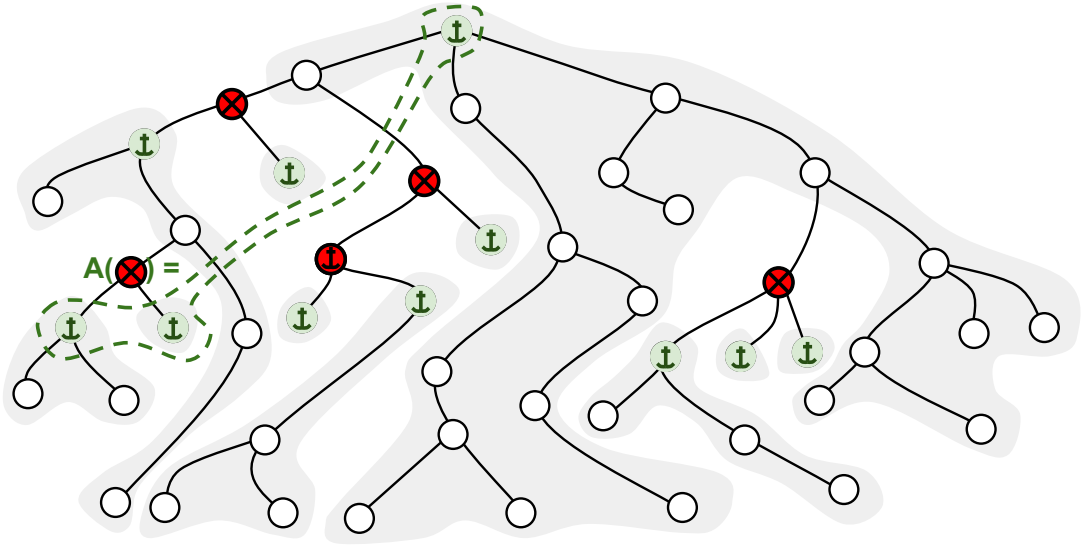
Beaucoup de symboles ont été utilisés dans la partie précédente, en particulier pour désigner toutes sortes de relations de parenté dans l'arbre ; nous invitons le lecteur à les oublier pour ce qui vient. En effet, il lui faudrait sinon se plonger dans les études passionnantes mais hors-sujet des alphabets berbère ou éthiopien.

2.4.1 Ancrage

Définition 24. *Soit G un graphe. On appelle* fonction d'ancrage *une fonction $A : V(G) \to \mathcal{P}(V(G))$ telle que pour tout $X \subseteq V(G)$, pour toute compo-*

21. $R_1(u)$ est un sommet quelconque de $R(u)$.

FIGURE 14 – Ancrage dans l'arbre. Chaque sommet en panne fournit ses fils et la racine de l'arbre comme ancres.



sante connexe C *de* $G \setminus X$*,*

$$C \cap \left(\bigcup_{x \in X} A(x) \right) \neq \emptyset$$

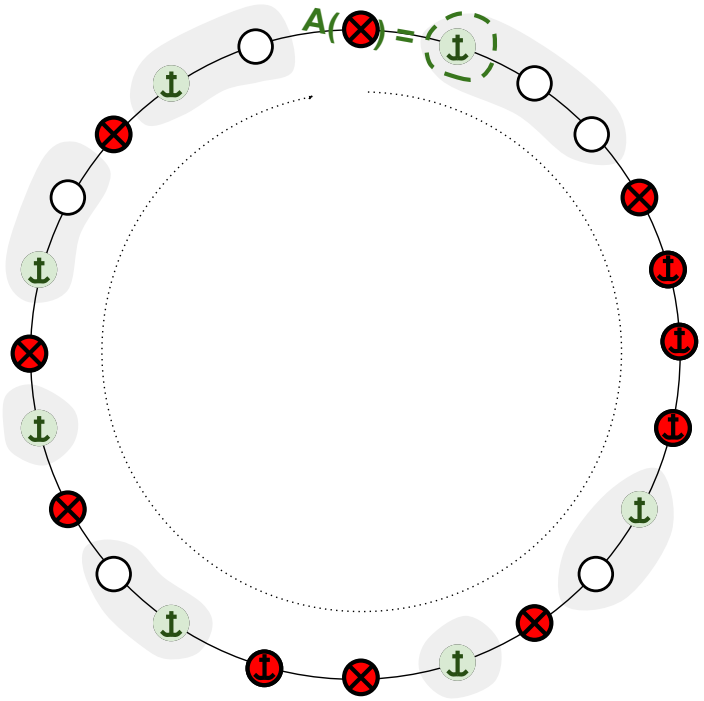
c'est à dire que l'union des résultats de l'application de la fonction A *à chacun des sommets d'une panne contient au moins un sommet*[22] *dans chaque composante connexe du graphe résultant de cette panne. On verra les figures 14 et 15.*

Soit $\mathcal{A}(G)$ *l'ensemble des fonction d'ancrage d'un graphe. On appelle* nombre d'ancrage *le nombre* $\gamma(G)$ *défini ainsi :*

$$\gamma(G) = \min_{A \in \mathcal{A}(G)} \max_{x \in V(G)} |A(x)| \ .$$

22. Dit *sommet-ancre*.

FIGURE 15 – Ancrage dans le cycle. Chaque sommet en panne fournit son successeur dans un ordre circulaire comme ancre.



On peut étendre[23] *cette notion à une famille de graphe* $\mathcal{F}$ *:*

$$\gamma(\mathcal{F}) = \max_{G\in\mathcal{F}}\{\gamma(G)|G \in \mathcal{F}\} \ .$$

2.4.2 Couture

Soit un ensemble de δ graphes distincts $\mathcal{M} = \{M_1, \dots, M_\delta\}$ et soit C un autre graphe n'ayant de sommet commun avec aucun M_i.

Soit un ensemble de δ *fonctions de couture* $\mathcal{K} = \{\kappa_1, \dots, \kappa_\delta\}$ de la forme $\kappa_i : V(M_i) \to V(C) \cup \{\perp\}$. On note, sans indice :

$$\kappa : \bigcup_{1\leq i\leq\delta} V(M_i) \to V(C) \cup \{\perp\}$$

la fonction associant, à un sommet u d'un des graphes de $\mathcal{M}$, son image par la fonction κ_i où i est l'indice de son graphe d'appartenance. On note :

$$\kappa^{-1} : V(C) \to \prod_{1\leq i\leq\delta} \mathcal{P}(V(M_i))$$

la fonction associant à un sommet v de C le δ-uplet

$$\kappa^{-1}(v) = ((\kappa^{-1}(v))_1, \dots, (\kappa^{-1}(v))_\delta) = (\kappa_1^{-1}(v), \dots, \kappa_\delta^{-1}(v))$$

produit cartésien de ses images réciproques[24] par les κ_i.

Définition 25. Coudre *le graphe* M_i *sur le graphe* C *signifie créer un nouveau graphe, noté* $M_i \rtimes_{\kappa_i} C$ *résultant, pour tout* $u \in M_i$*, de la fusion*[25] *du sommet* u *dans le sommet* $\kappa_i(u)$ *s'il est défini. Les arêtes multiples éventuellement créées sont simplifiées et les boucles sont ignorées.*[26] *(Voir figure 16.) On se permettra de noter* $\mathcal{M} \rtimes_\kappa C$ *le graphe résultant de la couture successive*[27] *de tous les graphes de* $\mathcal{M}$ *sur* C*. On appelle* épaisseur de couture*, noté*

23. $\gamma(\mathcal{F})$ n'étant défini que si $\{\gamma(G)|G \in \mathcal{F}\}$ est borné. Ce n'est pas le cas par exemple dans la famille des arbres.

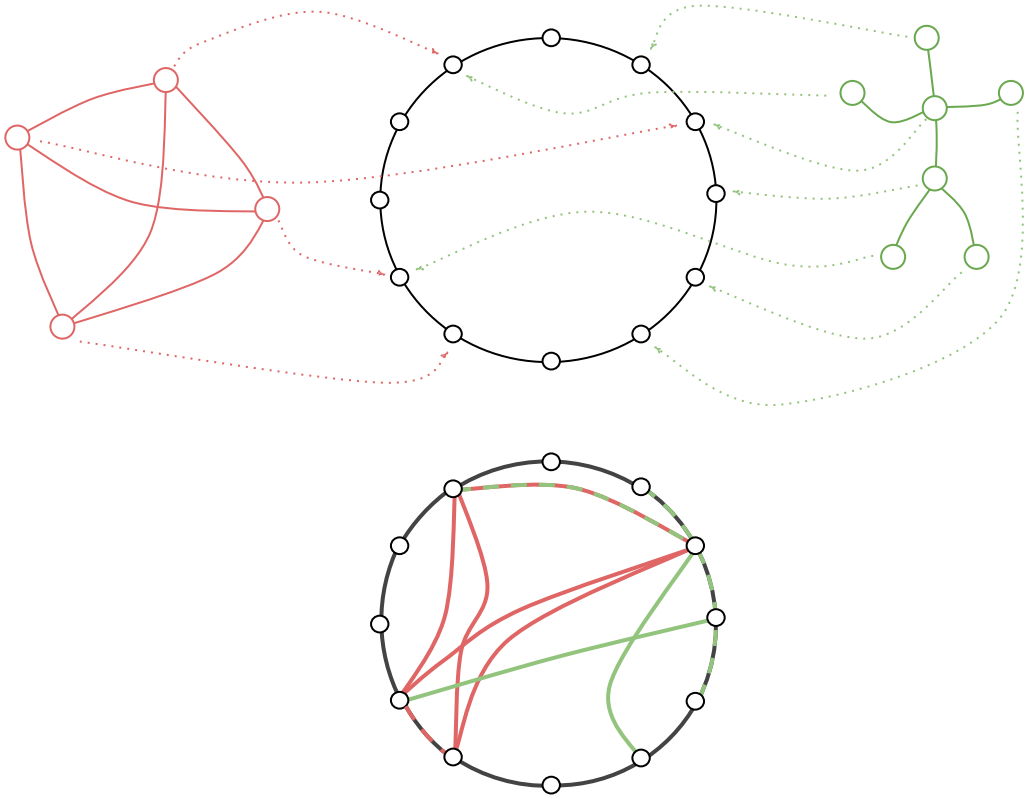
24. éventuellement, $(\kappa^{-1}(v))_i = \emptyset$ pour un certain i si aucun sommet de M_i n'a pour image v par κ_i.

25. Une sorte de *point* de couture.

26. On rappelle que boucles et arêtes multiples ne sont pas pertinentes quand de connexité et de pannes sur les sommets il s'agit.

27. Peu importe l'ordre des opérations. On utilisera l'opérateur « $\rtimes$ », sans indice, s'il n'y a pas d'ambigüité.

FIGURE 16 – Couture du graphe rouge et du graphe vert sur le cycle noir.



$\rho(M_i, \kappa_i, C)$ ou s'il n'y a pas de risque d'ambigüité, simplement $\rho(M_i \rtimes C)$ ou même ρ_i, la valeur

$$\rho_i = \max_{y \in C} |\kappa_i^{-1}(y)|$$

et donc, naturellement, on note

$$\rho = \rho(\mathcal{M} \rtimes C) = \rho(\mathcal{M}, \kappa, C) = \max_{y \in C} \max_{M_i \in \mathcal{M}} |\kappa_i^{-1}(y)| \ .$$

Définition 26. *Soit $\mathcal{F}$ une famille de graphes telle que, pour tout $G \in \mathcal{F}$, on peut obtenir (en temps $\text{UNSEW}_{\mathcal{F}}$) un triplet $(\mathcal{M}, C, \mathcal{K})$ tel que :*

— $\mathcal{M} = \{M_1, \ldots, M_\delta\}$

— *Les M_i sont deux à deux distincts.*
— *Chaque M_i est élément d'une famille $\mathcal{M}_i$ admettant un schéma d'étiquetage compact d'urgence $S_{\mathcal{M}_i}$ pour* Composante Connexe avec Panne.
— *C est un graphe distinct de chaque M_i*
— *C est élément d'une famille $\mathcal{C}$ admettant un schéma d'étiquetage compact d'urgence $S_{\mathcal{C}}$ pour* Composante Connexe avec Panne.
— *La famille $\mathcal{C}$ admet un nombre d'ancrage $\gamma(\mathcal{C})$ et sa fonction d'ancrage A est calculable en temps constant pour tout sommet.*
— *$\mathcal{K} = \{\kappa_1, \ldots, \kappa_\delta\}$ est un ensemble de fonction de couture et les ρ_i correspondants sont bornés pour la famille $\mathcal{M}_i$ par $\rho_{\mathcal{M}_i}$.*
— *$G = \mathcal{M} \rtimes C$*

alors on dit que $\mathcal{F}$ se découd favorablement *dans $\langle (\mathcal{M}_i)_{1\leq i\leq\delta}, \mathcal{C}\rangle$ en temps* Unsew$_{\mathcal{F}}$.

2.4.3 Méta-Schéma

Théorème 27. *Soit $\mathcal{F}$ une famille de graphes qui se découd favorablement dans $\langle (\mathcal{M}_i)_{1\leq i\leq\delta}, \mathcal{C}\rangle$.*

Alors la famille $\mathcal{F}$ admet un schéma compact d'étiquetage d'urgence S pour Composante Connexe avec Panne.

— *Les étiquettes sont de taille (en bits) :*

$$\lambda_S = \gamma(\mathcal{C}) \cdot (\lambda_{S_{\mathcal{C}}} + \sum_{1\leq i\leq\delta} \rho_{\mathcal{M}_i} \cdot \lambda_{S_{\mathcal{M}_i}})$$

où λ_σ est la taille maximale en bits d'une étiquette pour un schéma quelconque σ. On utilisera les notations L_σ, P_σ et Q_σ pour désigner les temps d'étiquetage, pré-requête et turbo-requête d'un schéma σ.

— *L'étiquetage se fait en temps :*

$$L_S = \textsc{Unsew}(\mathcal{F}) + L_{S_{\mathcal{C}}} + \sum_{1\leq i\leq\delta} (L_{S_{\mathcal{M}_i}} + \rho_{\mathcal{M}_i}) + O(\gamma(\mathcal{C}) \cdot n_{\mathcal{C}})$$

— *La pré-requête d'une panne X demande un temps :*

$$P_S(|X|) = P_1 + P_2$$

avec :

$$P_1 = |X| \cdot \gamma(\mathcal{C}) \cdot \left(Q_{S_\mathcal{C}} + \sum_{1 \leq i \leq \delta} (\rho_{\mathcal{M}_i} \cdot Q_{S_{\mathcal{M}_i}}) \right)$$

$$P_2 = \textsc{SetUnion}\left(|X| \cdot \gamma(\mathcal{C}) \cdot \left(1 + \sum_{1 \leq i \leq \delta} \rho_{\mathcal{M}_i} \right) \right)$$

— *On effectue une turbo-requête en temps :*

$$Q_S(|X|) = Q_{S_\mathcal{C}} + \textsc{SetFind}\left(|X| \cdot \gamma(\mathcal{C}) \cdot \left(1 + \sum_{1 \leq i \leq \delta} \rho_{\mathcal{M}_i} \right) \right)$$

Démonstration. Soit $G \in \mathcal{F}$ tel que $G = M_i \rtimes_{\kappa_i} C$.

Cette preuve consiste en la construction du méta-schéma. Elle s'appuie sur le fait que les M_i peuvent être utilisés pour « réparer » les pannes qui déconnectent des composantes de C. Cette preuve ressemble donc en certains points à celle du lemme 18, notamment, on utilisera une structure de donnée Unir et Trouver pour matérialiser les fusions de composantes connexes.

L'étiquette d'un sommet u de G est de la forme :

$$\textsc{Label}_S(u) = \left\langle (\textsc{Label}'_S(a))_{a \in \{u\} \cup A(u)} \right\rangle$$

avec

$$\textsc{Label}'_S(a) = \textsc{Label}_{S_\mathcal{C}}(a), \left(\left(\textsc{Label}_{S_{\mathcal{M}_i}}(x) \right)_{x \in \kappa_i^{-1}(a)} \right)_{1 \leq i \leq \delta} .$$

Pour tout $u \in V(G)$, les étiquettes sont donc bien de taille (en bits) au plus :

$$\lambda_S = \gamma(\mathcal{C}) \cdot \lambda'_S$$

où :

$$\lambda'_S = \lambda_{S_\mathcal{C}} + \sum_{1 \leq i \leq \delta} \rho_{\mathcal{M}_i} \cdot \lambda_{S_{\mathcal{M}_i}} .$$

On peut étiqueter G en temps :

$$L_S = \textsc{Unsew}(\mathcal{F}) + L_{S_\mathcal{C}} + \sum_{1 \leq i \leq \delta} (L_{S_{\mathcal{M}_i}} + \rho_{\mathcal{M}_i}) + O(\gamma(\mathcal{C}) \cdot n_\mathcal{C})$$

en décousant G, en étiquetant chacun des graphes intervenant, puis en attribuant les étiquettes aux sommets de C selon les fonctions de couture, puis en les distribuant selon la fonction d'ancrage. [28]

On peut traiter la pré-requête d'une panne X en temps :

$$P_S(|X|) = P_1 + P_2$$

avec :

$$P_1 = |X| \cdot \gamma(\mathcal{C}) \cdot \left(Q_{S_\mathcal{C}} + \sum_{1 \leq i \leq \delta} (\rho_{\mathcal{M}_i} \cdot Q_{S_{\mathcal{M}_i}}) \right)$$

$$P_2 = \textsc{SetUnion}\left(|X| \cdot \gamma(\mathcal{C}) \cdot \left(1 + \sum_{1 \leq i \leq \delta} \rho_{\mathcal{M}_i} \right) \right)$$

en pré-calculant les composantes connexes de tous les représentants de toutes les ancres de la panne, dans les graphes où ils interviennent (en temps P_1), le tout en tenant à jour une structure de donnée Unir et Trouver (en temps P_2). Cette structure nous permet de fusionner en une seule les composantes d'un graphe M_i et du graphe C correspondant à une même ancre considérée comme point de couture.

On traite alors une requête :

— Si $u \in C$:

$$\textsc{CompConn}_S(\textsc{Label}_S(u)) = \text{DSFind}(\textsc{CompConn}_{S_\mathcal{C}}(\textsc{Label}_{S_\mathcal{C}}(u))$$

28. Un sommet u appartenant à un M_i dont l'image par la fonction de coûture κ_i est $\perp$ recevra les étiquettes d'un ensemble de sommet intersectant toutes les composantes connexes de C avant panne, ensemble qu'on pourrait noté $A(\emptyset)$ est dont la cardinalité est forcément inférieure au nombre d'ancrage.

— Sinon (u est dans un et un seul M_i) :

$$\textsc{CompConn}_S(\textsc{Label}_S(u)) = \textsc{DSFind}(\textsc{CompConn}_{S_{\mathcal{M}_i}}(\textsc{Label}_{S_{\mathcal{M}_i}}(u))$$

Si cette donnée n'est pas définie, c'est que u n'est plus connecté à C, on peut alors retourner $(i, \textsc{Label}_{S_{\mathcal{M}_i}}(u))$ comme identifiant de sa composante connexe.

On a donc pour résultat, soit l'identifiant d'une composante connexe d'un M_i non connectée à C (c'est donc une composante connexe du graphe en panne si on l'associe à i pour éviter les doublons), soit l'identifiant d'une classe d'équivalence de composantes connexes reliées via C.

La requête prend un temps :

$$Q_S(|X|) = Q_{S_C} + \textsc{SetFind}\left(|X| \cdot \gamma(\mathcal{C}) \cdot \left(1 + \sum_{1 \leq i \leq \delta} \rho_{\mathcal{M}_i}\right)\right) .$$

Ce qui établit le théorème 27. □

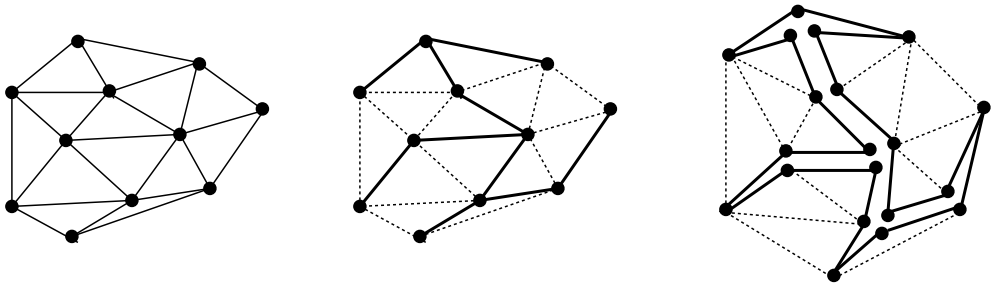
2.5 Connexité Avec Panne pour les graphes planaires 3-connexes

Théorème 28. *La famille des graphes planaires 3-connexes à n sommets admet un schéma d'étiquetage d'urgence pour* Connexité Avec Panne.

- *Les étiquettes sont de taille $O(\log n)$ bits.*
- *La complexité d'étiquetage est $O(n)$.*
- *La complexité de pré-requête est $O(\textsc{Sort}(|X|, n)+\textsc{PrePred}(|X|, n)$.*
- *La complexité de turbo-requête est $O(\textsc{Pred}(|X|, n)$.*

Démonstration. On va montrer comment on peut découdre un graphe planaire 3-connexes pour lui appliquer très simplement le méta-schéma du théorème 27 et obtenir le théorème 28.

FIGURE 17 – Découpage d'un graphe planaire 3-connexe.



2.5.1 Découpe d'un graphe planaire 3-connexes

Proposition 29. *On peut découdre favorablement en temps linéaire un graphe G planaire 3-connexe à n sommets en un arbre T à n sommets et de degré borné Δ et en un graphe planaire extérieur $2-$connexe P d'au plus $2n-2$ sommets.*

Démonstration.

Théorème 30 ([5] Barnette)**.** *Tout graphe planaire 3-connexe admet un arbre couvrant de degré 3.*

Théorème 31 ([56] Strothmann)**.** *On peut construire l'arbre du théorème 30 en temps linéaire.*

Soit G une graphe planaire 3-connexe à n sommets. On peut le plonger sur une sphère $\mathcal{S}$. Soit T un arbre couvrant de G de degré $\Delta = 3$ obtenu selon le théorème 31. Si on découpe $\mathcal{S}$ le long les arêtes de T tel qu'illustré sur la figure 17, on obtient un nouveau graphe P, toujours planaire car plongé sur le disque topologique résultant de l'incision de la sphère. On constate que tous les sommets de P sont sur le bord de ce disque : P est donc un graphe planaire extérieur 2-connexe à $2n-2$ sommets (il suffit de compter les arêtes de T pour s'en convaincre).

Chaque sommet u de G, au cours de cette construction, a donné naissance à exactement $d_T(u)$ sommets de P, où $d_T(u)$ est le degré du sommet u dans l'arbre couvrant T. On rappelle que $d_T(u) \leq \Delta$ pour tout u de G.

2.5.2 Reconstruction de G par couture

Soit la fonction de couture[29] $\kappa : V(P) \to V(T)$ qui à chaque sommet de P associe le sommet de T qui lui a donné naissance.

Coudre P sur T selon la fonction de couture revient à opérer la construction inverse de celle du paragraphe 2.5.1. On a donc $G = P \rtimes T$.

Pour tout $y \in V(T)$, on a $|\kappa^{-1}(y)| \leq \Delta$ car y a donné naissance à autant de sommets de P qu'il admettait de coins dans T. Donc $\rho = \Delta = 3$.

Soit la fonction $A : V(T) \to \mathcal{P}(V(T))$ qui à chaque sommet de T associe l'ensemble de ses voisins. Il est aisé de constater que c'est une fonction d'ancrage, car T étant un arbre[30], pour tout $X \subseteq V(G)$, toute composante connexe de $G \setminus X$ possède un sommet voisin de X. Le nombre d'ancrage de A est le degré de T, $\gamma(T) = \Delta = 3$.

On pose $\mathcal{G}_3(n)$ la famille des graphes planaires 3-connexes à au plus n sommets, $\mathcal{T}_3(n)$ la famille des arbres de degré au plus 3 et d'au plus n sommets et $\mathcal{P}(2n-2)$ la famille des graphes planaires extérieurs d'au plus $2n-2$ sommets. Alors, $\mathcal{T}_3(n)$ et $\mathcal{P}(2n-2)$ admettant des schémas d'étiquetage d'urgence pour Composante Connexe avec Panne d'après les théorèmes 8 et 14, on peut dire que $\mathcal{G}_3(n)$ se découd favorablement dans $\langle \mathcal{P}(2n-2), \mathcal{T}_\Delta(n) \rangle$ avec $\gamma(\mathcal{T}_3(n)) = \rho = 3$.

Cela prouve la proposition 29. □

2.5.3 Application du méta-schéma

On peut alors appliquer le théorème 27 en disant que la famille des graphes planaires 3-connexes à n sommets, admet un schéma d'étiquetage d'urgence S pour Connexité Avec Panne.

— Les étiquettes sont de taille (en bits) :

$$\lambda_S = 3 \cdot \lambda'_S$$

où :

$$\lambda'_S = \lambda_{S_{\mathcal{T}_3(n)}} + 3 \cdot \lambda_{S_{\mathcal{P}(2n-2)}}$$

29. Dite *canonique*.
30. Et donc un graphe connexe.

donc :

$$\lambda_S = O(\log n)$$

— L'étiquetage se fait en temps linéaire :

$$L_S = \textsc{Unsew}(\mathcal{G}_3(n)) + L_{S_{\mathcal{T}_3(n)}} + L_{S_{\mathcal{P}(2n-2)}} + O(n)$$

— La pré-requête d'une panne X demande un temps :

$$P_S(|X|) = P_1 + P_2$$

avec :

$$P_1 = 3 \cdot |X| \cdot \left(Q_{S_{\mathcal{G}_3(n)}} + 3 \cdot Q_{S_{\mathcal{P}(2n-2)}}\right)$$

$$P_2 = \textsc{SetUnion}\,(12 \cdot |X|)$$

donc [31] :

$$P_S(|X|) = O(|X| \cdot \textsc{Pred}(|X|, n))$$

— On effectue une turbo-requête en temps :

$$Q_S(|X|) = Q_{S_{\mathcal{T}_3(n)}} + \textsc{SetFind}\,(12 \cdot |X|)$$

soit [32] :

$$Q_S(|X|) = O(\textsc{Pred}(|X|, n))$$

Ce qui prouve le théorème 28. □

2.5.4 Pannes sur les sommets et les arêtes.

On peut généraliser le théorème 28 aux pannes sur les sommets et les arêtes.

Théorème 32. *La famille des graphes planaires 3-connexes à n sommets admet un schéma d'étiquetage d'urgence pour* Connexité Avec Panne sur les Sommets et les Arêtes.

— *Les étiquettes sont de taille $O(\log n)$ bits.*
— *La complexité d'étiquetage est $O(n)$.*
— *La complexité de pré-requête est $O(\textsc{Sort}(|X|, n)+\textsc{PrePred}(|X|, n)$.*

31. En négligeant encore, devant celui de Pred, le temps de SetFind dont la complexité est proportionnelle à la fonction inverse d'Ackermann.
32. *idem.*

— *La complexité de turbo-requête est* $O(\textsc{Pred}(|X|, n)$.

Démonstration. On prouve cette généralisation en reconsidérant la même preuve pour une subdivision[33] G' du graphe de départ G. G' possèdant $O(n)$ sommets d'après la formule d'Euler. (On peut montrer que $m \leq 3(n-2)$ si G planaire.)

On utilise la subdivision T' de T comme graphe de coupe. T' est toujours un arbre mais il n'est plus couvrant. Après découpe, on obtient P' une subdivision de P, graphe planaire extérieur 2-connexe. On sait la traiter (voir preuve du théorème 23). □

2.6 Connexité Avec Panne pour les graphes 3-connexes de genre eulérien g

2.6.1 Découpe

Proposition 33. *On peut découdre favorablement, en temps polynomial, un graphe 3-connexe G, à n sommets, de genre eulérien g, en :*

— *un arbre T à n sommets et de degré $\Delta = \lceil \frac{4+2\cdot g}{3} \rceil$*
— *un graphe planaire extérieur 2-connexe P d'au plus $\Delta \cdot n$ sommets*
— *un graphe à $O(g)$ arêtes*[34]

Le résultat se base sur les théorème suivants :

Théorème 34 (Ozeki [47])**.** *Tout graphe 3-connexe plongeable sur une surface fermée dont la caractéristique d'Euler est χ admet un arbre couvrant de degré $\lceil \frac{8-2\chi}{3} \rceil$.*

Théorème 35 (Fürer et Raghavachari [29])**.** *Pour tout graphe G à n sommet et m arêtes, on peut calculer un arbre couvrant de degré maximal borné par $\delta + 1$, où δ est la borne optimale, en temps $O(n \cdot m \cdot \log n \cdot \alpha(n))$.*

33. On rajoute un sommet, qu'on peut mettre en panne, au milieu de chaque arête.
34. Sans sommets isolés.

On peut donc, en temps polynomial, obtenir à partir de la carte topologique $\mathcal{C}$ d'une surface $\mathcal{S}$ de genre g, correspondant à un graphe G 3-connexe, un arbre couvrant T de degré borné par $\Delta + 1$. On peut ensuite, d'après [24], obtenir en temps $O(n + g)$, un graphe E contenant $O(g)$ arêtes[35] de $G \setminus T$, tel que le complémentaire $\mathcal{D}$ de $T \cup E$ dans la surface $\mathcal{S}$ soit un disque topologique. On dit que la carte planaire de $T \cup E$ est un *graphe de coupe* de la surface $\mathcal{S}$. La carte P issue de la découpe[36] de $\mathcal{S}$ est donc une carte planaire dont tous les sommets forment un cycle sur le bord[37] de $\mathcal{D}$. P est donc la carte d'un graphe planaire extérieur 2-connexe. On pourra voir la figure 18.

2.6.2 Reconstruction de G par couture

Soit la fonction de couture canonique $\kappa : V(P) \cup V(E) \to V(T)$. Alors par construction de T et E, il est clair que $G = \{P, E\} \rtimes_\kappa T$ et que $\rho = \Delta$.

2.6.3 Schéma *ad-hoc* pour E

Nous avons d'ores et déjà à disposition dans notre boîte à outils des schémas d'étiquetage d'urgence pour la familles des arbres à n sommets et celles des graphes planaires extérieurs à $O(n+g)$ sommets. Nous avons donc besoin de construire un schéma d'étiquetage pour Composante Connexe avec Panne pour la famille des graphes contenant $O(g)$ arêtes et leurs extrémités.

Proposition 36. *La famille $\mathcal{E}(m)$ des graphes à m arêtes, sans sommets isolés, admet un schéma d'étiquetage d'urgence trivial $\mathcal{S}_{\mathcal{E}(m)}$ pour* Composante Connexe Avec Panne *où :*

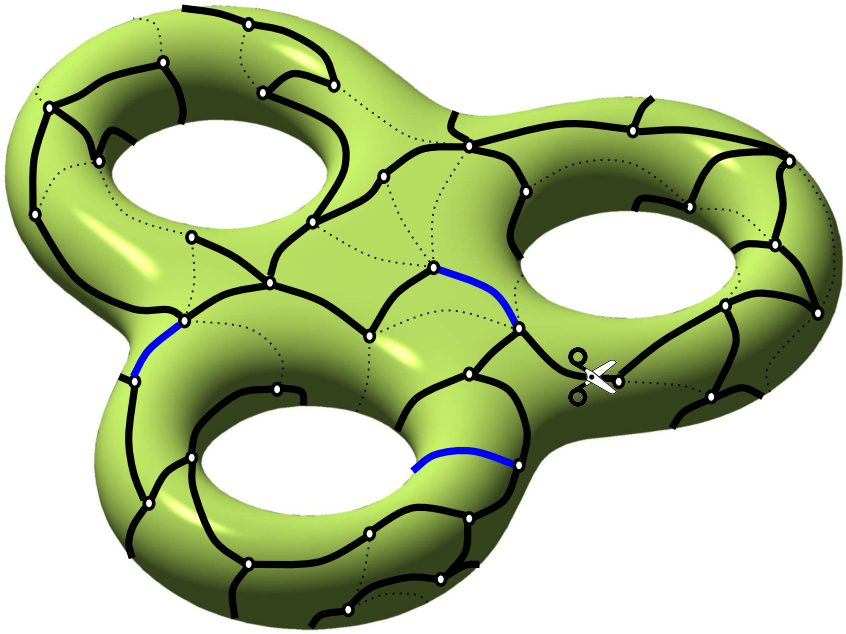
- *Les étiquettes sont de taille $O(m \log m)$ bits*
- *La complexité d'étiquetage est $O(m)$*
- *La complexité de pré-requête est*

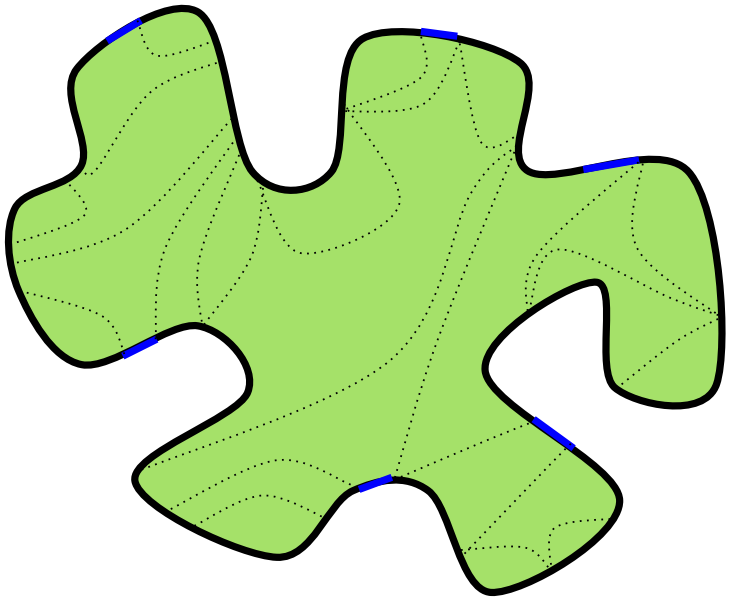
$$\textsc{MakeSet}(m)$$

- *La complexité de turbo-requête est* $\textsc{SetFind}(m)$

35. Et bien sûr leurs sommets-extrémités.
36. Idem que dans la partie 2.5.1.
37. Car T est couvrant.

FIGURE 18 – Découpage d'un graphe 3-connexe plongé sur une surface fermé. Ici un triple-tore. Les arêtes de l'arbre couvrant T sont en trait épais noir et les arêtes de E en bleu.





Démonstration. Chaque sommet du graphe est étiqueté par la concaténation d'un identifiant de ce sommet [38], et de la liste entière des arêtes du graphes [39]. Une panne X est traitées en temps $O(m)$. En effet, on peut remarquer que $|X| \leq 2 \cdot m$ puis construire $G \setminus X$ et calculer toutes ses composantes connexes [40], le tout en temps

$$\textsc{MakeSet}(m) \ .$$

Une turbo-requête se fait donc naturellement en temps $\textsc{SetFind}(m)$. □

2.6.4 Application du méta-schéma

D'après le théorème 27, la famille des graphes à n sommets, 3-connexes, de genre eulérien g, admet un schéma d'étiquetage d'urgence S tel que :

— Les étiquettes sont de taille (en bits) :

$$\lambda_S = \Delta \cdot \lambda'_S$$

où :

$$\lambda'_S = \lambda_{S_{\mathcal{T}_\Delta(n)}} + \Delta \cdot \lambda_{S_{\mathcal{P}(n+\mathrm{eg})}} + \Delta \cdot \lambda_{S_{\mathcal{E}(\mathrm{eg})}}$$

donc [41] :

$$\lambda_S = O(\log(n + \mathrm{eg})) = O(\log n)$$

— L'étiquetage se fait en temps :

$$L_S = \textsc{Unsew}(\mathcal{G}_{(\mathrm{eg},3)}(n)) + L_{S_{\mathcal{T}_\Delta(n)}} + L_{S_{\mathcal{P}(O(n))}} + L_{S_{\mathcal{E}(\mathrm{eg})}} + 2 \cdot \Delta + O(\Delta \cdot n)$$

Soit en un temps polynomial décrit dans l'énoncé du théorème 35.

— La pré-requête d'une panne X demande un temps :

$$P_S(|X|) = P_1 + P_2$$

avec :

$$P_1 = |X| \cdot \Delta \cdot \left(Q_{S_{\mathcal{G}_{(\mathrm{eg},3)}(n)}} + \Delta \cdot Q_{S_{\mathcal{P}(n+\mathrm{eg})}} + \Delta \cdot Q_{S_{\mathcal{E}(\mathrm{eg})}} \right)$$

38. De taille $O(\log m)$ bits.
39. Sous formes de couples des identifiants de leurs extrémités.
40. Qui sont des classes d'équivalence de sommets connectés par des arêtes, d'où l'utilisation d'Unir et Trouver.
41. En considérant eg et Δ comme des constantes pour notre schéma.

$$P_2 = \textsc{SetUnion}\left(|X| \cdot \Delta \cdot (1 + 2 \cdot \Delta)\right)$$

donc [42] :

$$P_S(|X|) = \textsc{Sort}(|X|, n) + |X| \cdot \textsc{Pred}(|X|, n)$$

— On effectue une turbo-requête en temps :

$$Q_S(|X|) = Q_{S_{\mathcal{T}_\Delta(n)}} + \textsc{SetFind}\left(|X| \cdot \Delta \cdot (1 + 2 \cdot \Delta)\right)$$

soit [43] :

$$Q_S(|X|) = O(\textsc{Pred}(|X|, n)$$

Ce qui met fin à la preuve du théorème 7. □

2.6.5 Pannes sur les sommets et les arêtes.

On peut généraliser [44] le théorème 7 aux pannes sur les sommets et les arêtes.

Théorème 37. *La famille des graphes 3-connexes admet un schéma d'étiquetage d'urgence pour* Connexité Avec Panne sur les Sommets et les Arêtes. *Pour un graphe G a n sommets :*

- *Les étiquettes sont de taille $O(\mathrm{eg}(G) \cdot \log n)$ bits.*
- *La complexité d'étiquetage est polynomiale. (linéaire dans le cas planaire)*
- *La complexité de pré-requête est $O(\textsc{Sort}(|X|, n+g)+\textsc{PrePred}(|X|, n))$.*
- *La complexité de turbo-requête est $O(\textsc{Pred}(|X|, n + g))$.*

Démonstration. Une application simple de la formule d'Euler nous montre que dans un graphe G connexe de genre eulérien g, $m = O(n + g) = O(n^2)$.

On peut donc construire la subdivision G' à $O(n+g)$ sommets du graphe G tel que dans la preuve du théorème 32 afin de procéder à une réduction équivalente.

Les schémas utilisés pour les graphes de coupe et pour le graphe planaire extérieur 2-connexes sont stables par subdivision, ils peuvent encore être utilisés. □

42. En négligeant une fois de plus le temps de SetFind dont la complexité est proportionnelle à la fonction inverse d'Ackermann.

43. *idem.*

44. On notera l'apparition du paramètre g dans certains temps de traitement.

Nous pouvons donc conclure ce chapitre 2.

2.7 Conclusion du chapitre

Tout au long de ce chapitre, nous avons mis en évidence l'existence de schémas d'étiquetage d'urgence aux temps de requête PRED(|X|, n), pour le problème COMPOSANTE CONNEXE AVEC PANNE dans diverses familles de graphes, à savoir, les arbres, les graphes planaires extérieurs, les graphes planaires 3-connexes et les graphes de genres eulériens bornés, également 3-connexes.

Il convient de remarquer que ce temps est optimal pour les familles de graphes que nous étudions car, pour le chemin de longueur n, inclus dans la famille des arbres et dans celles des graphes planaires à n sommets, une solution du problème PRÉDECESSEUR est également une solution de COMPOSANTE CONNEXE AVEC PANNE, donc notre temps de requête est en $\Omega(\text{PRED}(|X|, n))$. De plus, d'après [50], il existe des instances X [45] du problème prédécesseur tel que, si le temps de pré-requête est polynomial, $\text{PRED}(|X|, n) = \Omega(\log \log n)$.

Nous pouvons comparer ces résultats à l'existant dans le cas planaire 3-connexes :

Schéma	Courcelle et al. [14]	Théorème 28
Taille étiquettes	$O(\log n)$ bits	$O(\log n)$ bits
Temps étiquetage	$O(n)$	$O(n)$
Temps pré-requête	$O(\|X\|^2)$	$\text{SORT}(\|X\|, n) + \text{PREPRED}(\|X\|, n)$
Temps turbo-requête	$\text{PLOCQUERY}(\|X\|, n)$	$\text{PRED}(\|X\|, n)$

Ce qui peut donner, en explicitant les complexités :

45. Où $|X|$ est polynomial en n.

Schéma	Courcelle et al. [14]	Théorème 28
Taille étiquettes	$50 \log n + o(\log n)$ bits	$234 \log n + o(\log n)$ bits
Temps étiquetage	$O(n)$	$O(n)$
Temps pré-requête	$O(\|X\|^2)$	$\|X\| \log \log n$
Temps turbo-requête	$\log n$	$\log \log n$

On constate que la complexité d'étiquetage est améliorée aussi bien pour le paradigme *schéma d'étiquetage* que pour *schéma d'étiquetage d'urgence*.

Dans le premier, on passe d'un temps de requête quadratique en $|X|$, la taille de la panne, à un temps quasi-linéaire.

Dans le second paradigme, on passe d'une réduction à une autre. De Composante Connexe avec Panne $\rightarrow$ Point Location, on passe à Composante Connexe avec Panne $\rightarrow$ Prédécesseur, ce qui permet des turbo-requêtes exponentiellement plus rapide.

En revanche, la solution à laquelle nous nous comparons construit des étiquettes presque cinq fois plus petites et se généralise à la classe toute entière des graphes planaires. Ce n'est pas le cas de la notre et ce sera l'un des objectifs du chapitre à venir.

Chapitre 3

Graphes simplement connexes et graphes sans mineur H

3.1 Résumé de notre contribution

Dans ce chapitre, nous allons étendre, en relâchant la contrainte de compacité à des étiquettes de taille polylogarithmique en n, le résultat du chapitre précédent aux graphes qui ne sont pas 3-connexes et aux graphes n'admettant pas un graphe arbitraire H pour mineur. Pour ce faire, nous allons considérer, d'une part, les graphes 1-connexes comme des structures arborescentes de graphes de connectivités supérieures, et, d'autre part, nous utiliserons une structuration des graphes excluant un mineur fixé H (décrite dans [52]) en une arborescence de graphes presque-plongeables sur une surface où H ne l'est pas.

Les figures 41 et 42 du chapitre de conclusion de ce document pourront aider le lecteur à s'orienter dans la lecture du présent chapitre. Elles le résument graphiquement.

Nous allons dans un premier temps mettre au point un méta-schéma pour les graphes structurés en arborescences. Il aura pour corollaire immédiat un schéma pour les graphes de largeur arboréscente bornée. Nous ferons usage ensuite de notre méta-schéma pour obtenir les résultats escomptés sur les graphes 1-connexes ou excluant un mineur.

3.2 Méta-schéma pour les k-arborescences

Une *décomposition arborescente* d'un graphe G est un arbre T dont les sommets, appelés *sacs*, sont des sous-graphes de G et tel que, si un sommet appartient à deux sacs s_1 et s_2 dont l'un est ancêtre de l'autre, alors il appartient également à tous les sacs du plus court chemin dans T entre s_1 et s_2. Et par union de tous les sacs de T, on doit retrouver le graphe G.

Soit G un graphe à n sommet. S'il admet une décomposition arborescente T telle que :

- l'intersection entre deux sacs voisins dans T compte au plus k sommets ;
- que chaque sac de T est un graphe de la classe $\mathcal{C}$;
- et qu'on reste dans la classe $\mathcal{C}$ en prenant un sac s de T auquel on ajoute un nombre arbitraire d'arêtes simples unissant des paires de sommets appartenant, deux à deux, à une même intersection avec un sac voisin.

alors on dit que G est une *k-arborescence sur la classe $\mathcal{C}$.*.

Cette section a pour objet de prouver le théorème suivant :

Theorème 38. *Si la classe $\mathcal{C}$ admet un schéma d'étiquetage d'urgence quasi-compact*[1] *pour* Composante Connexe avec Panne sur les Sommets et les Arêtes*, alors la classe des k-arborescences sur $\mathcal{C}$ en admet également un.*

On peut remarquer que ce théorème aura pour corrolaire immédiat[2] l'existence d'un schéma d'étiquetage d'urgence (temps de requêtes optimaux) quasi-compact pour les graphes de largeur arborescente k.

3.2.1 Travail préliminaire

Branches lourdes. Pour tout sac s d'une arborescence T à N sacs, tel que s n'est pas une feuille, on peut désigner un de ses fils, mettons f comme

1. Les étiquette sont de taille poly-logarithmique.
2. En posant que $\mathcal{C}$ est la famille des graphes dont le nombre de sommets est inférieur à k.

étant son *fils lourd*. L'arrête $s - f$ est dite *arête lourde*.

D'après [57], il est possible de choisir en temps $O(N)$ tous les fils lourds de l'arborescence de telle sorte que pour tout sac t, le nombre d'arête non lourdes du plus court chemin entre t et la racine de T est $O(\log N)$.

Les sacs de l'arborescence T peuvent alors être partitionnés en branches dites *lourdes*. Ces branches lourdes sont les chemins maximaux de T dont toutes les arêtes sont lourdes.

Si un sac s_1 d'une branche lourde B_1 est fils d'un sac s_2 d'une autre branche lourde B_2, on dit que B_1 est fille de B_2 dans *l'arbre des branches lourdes*. Cet arbre a une hauteur $O(\log N)$. On peut noter que s_1 ne peut être que le sac de B_1 le plus proche de la racine de T. (Voir fig. 19)

Pseudo-arête. Dans cette partie, nous allons souvent travailler sur notre arborescence *via* des traitements *bottom-up*[3] et ce à partir d'une quantité d'information limitée, donnée par exemple, par les étiquettes des sommets en pannes. Nous aurons donc parfois besoin de savoir si deux sommets sont connectés dans une région du graphe qui nous sera méconnue.

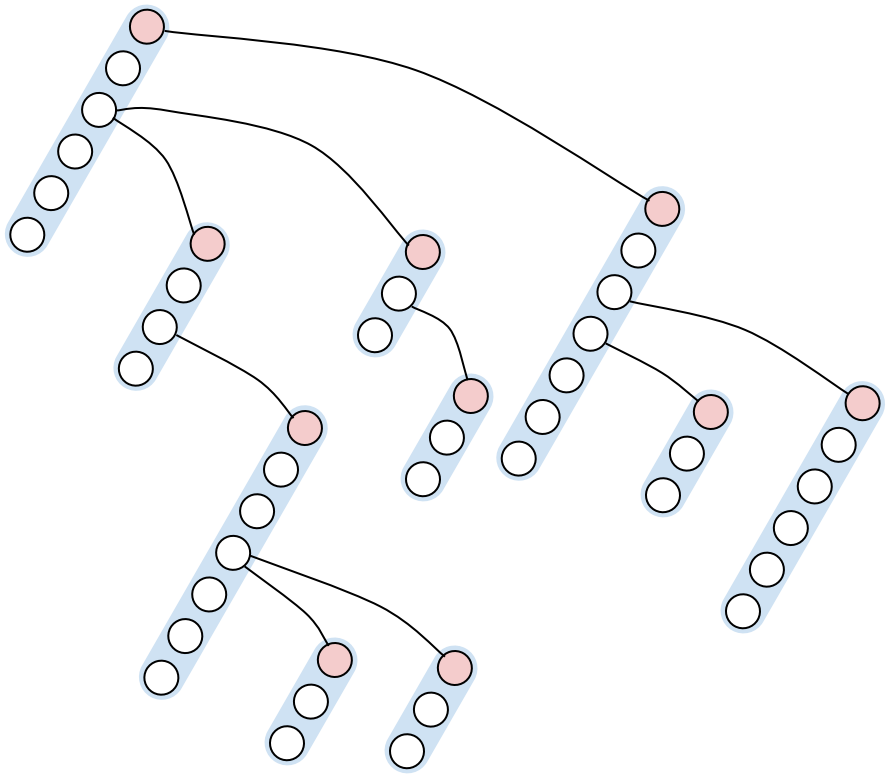
C'est pourquoi nous aurons besoin d'un outil permettant, intuitivement, de « résumer » le fait qu'il existe un chemin entre deux sommets d'un sac dans des sacs inférieurs « non contrôlés » (*i.e.* au sujet desquels peut d'information nous est fournise, par des pannes par exemple.)

On connectera donc une paire de sommets $\{x, y\}$, contenue dans l'intersection d'un sac S avec ses sacs-fils, par une *pseudo-arête* dès lors qu'il existera un chemin de x à y n'utilisant que des arêtes contenues dans des sacs strictement plus éloignés de la racine de l'arborescence T que S.

Une pseudo-arête sera pondérée par le nombre de fils non-lourds de S par où passe un chemin de x à y n'utilisant pas les arêtes de S ou de sacs plus

3. Du bas vers le haut.

FIGURE 19 – k-arborescence structurée en arbre de branches lourdes. Deux sacs voisins d'une branche ou deux sacs reliés par une arêtes partagent une k-intersection.



proches de la racine de T. Ce poids sera désigné comme étant la *force* de la pseudo-arête.

On pourra construire récursivement l'ensemble des pseudo-arêtes pondérées selon un traitement *bottom-up* de T en temps $O(k^2 \log n)$, où n est le nombre de sommets du graphe G.

On note S^+ un sac S augmenté de ses pseudo-arêtes.

Sacs remarquables pour le sommet u. Soit G un graphe structuré en une k-arborescence sur $\mathcal{C}$. Soit u un de ses sommets. On pourra lire les définitions suivantes en se référant à la figure 20.

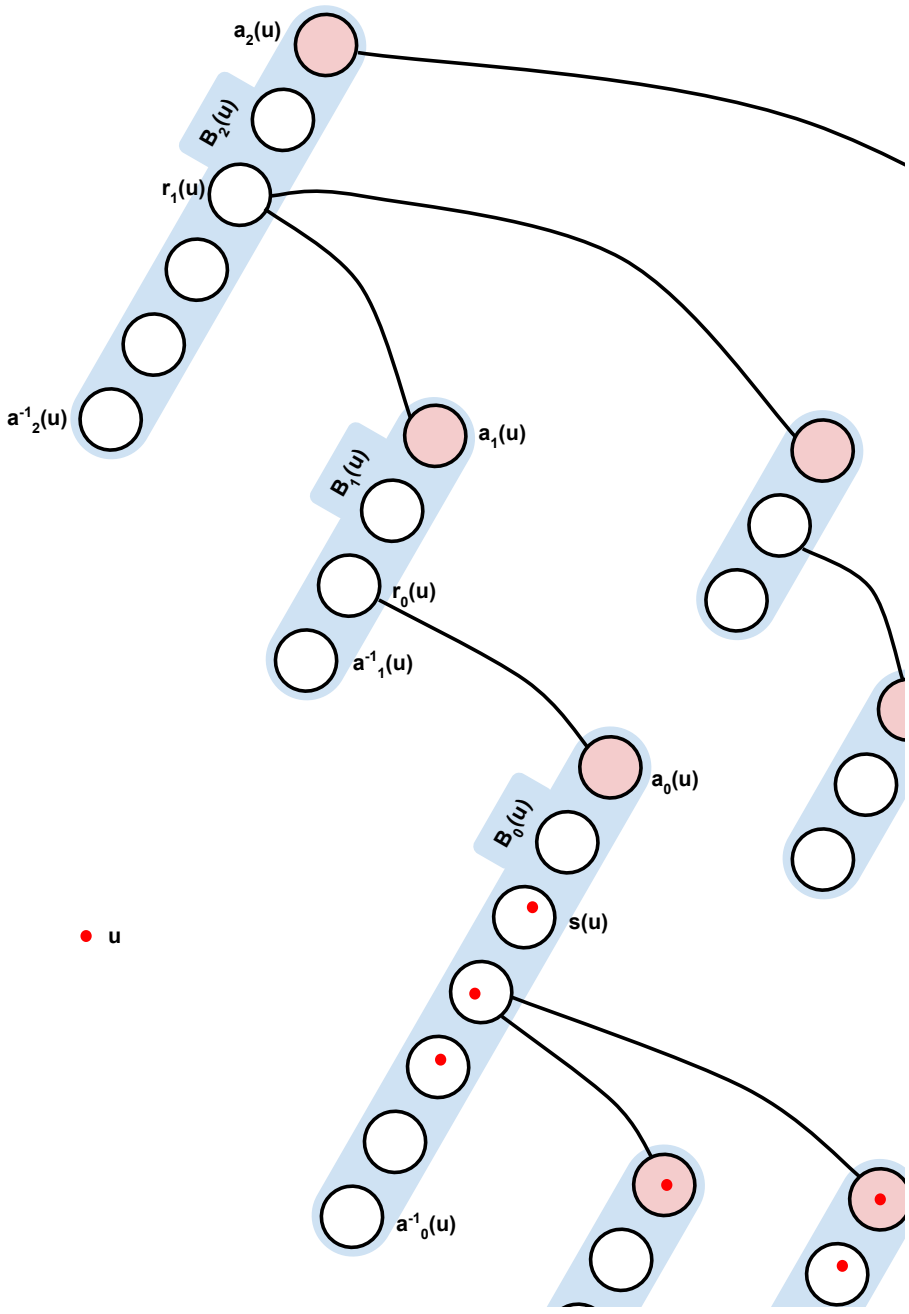
- On note $p(S)$ le père d'un sac S dans l'arborescence.
- On note $s(u)$ le sac le plus proche de la racine de l'arborescence dans lequel u apparait.
- On note $B(s(u))$ la branche lourde contenant u.
- On note $a(B(s(u))$ le sac de la branche le plus proche de la racine (qu'on appelle *apex*) et $a^{-1}(B(s(u))$ le sommet le plus éloigné.
- par commodité, on écrira :
  - $B_0(u) = B(s(u))$
  - $a_0(u) = a(B(s(u)))$
  - $a_0^{-1}(u) = a^{-1}(B(s(u)))$
  - $r_0(u) = p(a(B(s(u))))$
- Et pour tout $i > 0$:
  - $a_i(u) = a(r_{i-1}(u))$
  - $a_i^{-1}(u) = a^{-1}(r_{i-1}(u))$
  - $r_i(u) = p(a_i(u))$
  - $B_i(u) = B(a_i(u))$

Sacs-fantômes. Nous allons définir pour chaque sac d'une branche lourde un sous-ensemble des sacs de la branche en utilisant le lemme suivant :

Lemme 39. *Soit E_n l'ensemble des entiers positifs inférieurs à n. On peut attribuer à tout $x \in E_n$ un sous-ensemble de E_n, de cardinalité $O(\log n)$, noté $E(x)$ tel que pour toute paire $\{i, j\}$ de E_n,*

$$(E(i) \cap E(j)) \ \cap \ [i, j] \neq \emptyset \ .$$

FIGURE 20 – Sacs remarquables pour le sommet u dans la k-arborescence.



E_x peut être calculé en temps $O(\log n)$ et on peut déterminer un élément de cette intersection en temps $O(1)$.

Démonstration. Soit x un entier compris entre 0 et n. On considère $b(x) = b_0 \cdot b_1 \cdots b_m$ son écriture binaire sur $m+1$ bits avec $m = \lceil \log n \rceil - 1$.
Pour tout k de 0 à m, on note $p_k(x)$ le préfixe de $b(x)$ de longueur k bits, et on note les deux complétions suivantes de $p_k(x)$ sur m bits :

— $a_k(x) = p_k(x) \cdot \underbrace{0 \cdots 0}_{m-k}$

— $b_k(x) = p_k(x) \cdot 1 \underbrace{0 \cdots 0}_{m-(k+1)}$

On pose :

$$E(x) = \bigcup_{0 \leq k \leq m} \{a_k(x), b_k(x)\} \ .$$

On peut supposer que cet ensemble est structuré en deux tables $A(x)$ et $B(x)$ pour chaque entier x.

Soient i et j deux entiers inférieurs à n tels que (sans perte de généralité) $i \leq j$. Et soit $k \leq m$ le plus grand entier tel que $p_k(i) = p_k(j)$. On notera que $k = \textsc{FirstOne}(b(i) \oplus b(j))$ avec :

— $\oplus$ est l'opération *XOR bit à bit*

— FirstOne est l'opération qui donne l'indice entre 0 et $m-1$ du 1 le plus faible d'un registre de taille m.

Ces opérations peuvent être considérées élémentaires.

On remarque que le $(k+1)$-ième bit de $b(i)$ est à 0 et que celui de $b(j)$ est à 1. On en déduit que l'entier z d'écriture binaire

$$b(z) = p_k(i) \cdot 1 \underbrace{0 \cdots 0}_{m-(k+1)}$$

est tel que :

— $i \leq z \leq j$

— $z = b_k(i)$

— $z = a_{k+1}(j)$

On a donc défini, pour tout x inférieur à n, un ensemble $E(x)$ contenant $2 \cdot \lceil \log n \rceil$ entiers, de telle façon que pour tout i et j inférieurs à n, on obtient

FIGURE 21 – Les sacs s_1 et s_2 et leurs fantômes dans la branche lourde. On notera qu'ils partagent l'un d'entre eux sur le chemin dont ils sont extrémités.

en temps constant [4] un entier z compris entre i et j et élément à la fois de $E(i)$ et de $E(j)$.

□

Le lemme 39 nous permet d'affirmer de manière analogue qu'on peut attribuer à chaque sac s d'une branche lourde B un sous-ensemble (dit de *sacs-fantômes*) $F(s)$ des sacs de B, de cardinalité $O(\log |B|)$, tel que pour toute paire $\{s_1, s_2\}$ de sacs de B, $F(s_1) \cap F(s_2)$ contient un sac situé sur le chemin B entre s_1 et s_2 (éventuellement l'une des extrémités). On verra la figure 21.

Graphe de reconnexion de s à t. Soient s et t deux sacs d'une même branche lourde B tels que les autres sacs du chemin $s - t$ ne sont pas impactés par des pannes [5]. On appelle *graphe de reconnexion de s à t* le graphe dont l'ensemble des sommets V est l'union d'au plus 4 ($\leq k$)-intersections, à savoir, si elles existent :

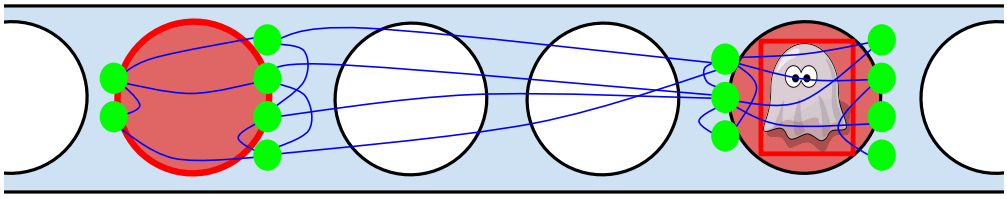
— l'intersection de s avec son fils-lourd
— l'intersection de s avec son père
— l'intersection de t avec son fils lourd
— l'intersection de t avec son père

On met une arête de reconnexion entre deux des sommets de V :

— s'ils sont connectés dans s^+ quand tous les autres sommets de V sont

4. Par exemple en constatant que $z = i \leq j$? $B(i)[\text{FIRSTONE}(b(i) \oplus b(j))]$: $B(j)[\text{FIRSTONE}(b(i) \oplus b(j))]$.
5. Sur des sommets non présents dans s et t.

FIGURE 22 – Graphe de reconnexion entre un sommet et (ici) l'un de ses fantômes dans une branche lourde (dessinée horizontalement car la définition est symétrique).



en panne[6] ;
— ou s'ils sont connectés dans t^+ quand tous les autres sommets de V sont en panne ;
— ou s'ils sont connectés dans I^+ quand tous les autres sommets de V sont en panne.

Avec I l'union des sac intermédiaires entre s et t (non inclus) et I^+ l'union de ces même sacs augmentés par les éventuelles pseudo-arêtes.

On verra la figure 22.

Méthode des sacs fantômes. Si deux sacs s et t partagent un fantôme f et qu'on connaît les graphes de reconnexion $s - f$ et $t - f$, alors on peut en déduire le graphe de reconnexion $s - t$ de manière triviale en temps $O(k)$.

3.2.2 Graphe auxiliaire de panne

Soit X l'ensemble des sommets en panne dans le graphe G dont une k-arboresence structurée en branche lourde nous est livrée. Nous allons ici contruire un graphe auxiliaire $P(G, X)$ nous permettant de mettre à jour les connexions dans le graphe entre les sacs contenant les plus hauts[7] représentants des sommets en panne. Cette construction sera une étape du pré-calcul

6. Celà permet d'observer une éventuelle déconnection entre deux sommets d'intersection qui serait due à la panne d'un troisième.
7. Plus proche de la racine de T.

de la panne X dans le méta-schéma d'étiquetage qui fait l'objet de la partie 3.2.

Notons $S_0(X) = \{s(x) \ ; \ x \in X\}$ l'ensemble des sacs les plus hauts où apparaissent les sommets de X.

Nous allons donner une définition par induction structurelle *bottom-up*[8] du *graphe de panne*.

Cas d'une branche lourde feuille. Si B est une branche lourde en position de feuille dans la structuration arborescente, on considère l'ensemble de sacs $E(B, X) = (B \cap S_0(X)) \cup \{a(B), a^{-1}(B)\}$ triés du bas vers le haut.[9]

Le *graphe de panne* $P(B, X)$ de la branche B est l'union des graphes de reconnexion de chaque paire consécutive de sacs de E. Si $B \cap S_0(X)$ est vide, on ne traite pas la branche et son graphe de connexion est vide.

Cas d'une branche lourde possédant des branches-filles On considère l'ensemble de sacs

$$E(B, X) = (B \cap S_0(X)) \cup \{a(B), a^{-1}(B)\} \cup F$$

où F est l'ensemble des sacs de B dont au moins un des fils non-lourds est apex d'une branche lourde traitée préalablement.[10]
On met a jour chaque sac augmenté de F en décrémentant les forces des pseudo arêtes lorsque la connexion n'est plus assurée dans l'un des fils, et en mettant en panne, le cas échéant celles dont la valeur associée atteint 0. On pourra voir la figure 23.[11]

8. *I.e* du bas vers le haut de l'arborescence.
9. Soient des sacs touchés par la panne et les extrémités de la branche.
10. On dira que la branche *hérite* d'une panne. Il s'agit d'un héritage *bottom-up*. L'analogie avec la généalogie serait plus évidente si l'on y dessinait les arbres dans le même sens.
11. Cette méthode nous permet de maintenir effectif le pouvoir de connexion des branches non-lourdes non impactées sans devoir disposer d'information sur elles. (Ce qui imposerait un facteur multiplicateur égal au degré maximal de l'arborescence, à savoir n dans le pire cas, à la taille de nos étiquettes.)

On procède ensuite comme dans le cas de la branche-feuille à ceci près que les sommets d'intersection issus des fils non-lourds traités préalablement sont ajoutés au graphe de reconnexion au sein de chaque sac f^+ issu de F.
On transmet en dernier lieu à l'apex de la branche située au dessus de B les mises à jour éventuelles des pseudo-arêtes et leurs forces initiales.

Le graphe de panne $P(G, X)$ est l'union des graphes de panne de toutes les branches traitées.

3.2.3 Constructions auxiliaires

Pour connaître la composante connexe d'un sommet dans le graphe $G \setminus X$, on tâchera en général de le relier à un sommet de $P(G, X)$ auquel il est connecté. Cela dit, dans des cas où nous nous intéresserions à l'état de connexion d'un sommet « éloigné » de $P(G, X)$, il nous faudra utiliser une méthode analogue à celle de la fonction de routage utilisée dans le schéma construit pour la famille des arbres. [12]

Fonction d'aiguillage. Soit A une fonction qui prend en entrée un couple sac-sommet (s, u) et lui associe i un identifiant de taille $O(\log n)$ de la composante connexe de u dans $G \setminus s$. On appelle *fonction d'aiguillage* cette fonction A. (Voir figure 24.)

Arbre d'aiguillage. La fonction précédante ne peut suffire car on ne pourra stocker dans l'étiquette d'un sommet u la valeur de $A(s, u)$ que pour un nombre limité de sac s [13]. On va devoir construire, pour chaque branche B, un arbre d'au plus $O(n^2)$ sommets que l'on pourra étiqueter pour le problème ROUTAGE comme vu auparavant (théorème 12) afin d'obtenir une information équivalente.

On définit pour toute branche B un arbre $\mathcal{A}(B)$ dont les sommets sont des couples sacs-ensemble-de-sommets de $B \times \mathcal{P}(V(G))$, tels que (On verra la figure 25.) :

12. On rappelle que la composante connexe d'un sommet u était le résultat du routage depuis le plus proche ancêtre de u dans X vers u lui-même.
13. Par exemple, pour les apices du sac de u.

FIGURE 23 – Graphe de panne, étape d'induction. La couleur rouge correspond à une panne. La couleur orange, à une panne héritée d'une sous-branche. Chaque arête rouge dans une intersection d'apex constitue un ordre de décrémentation de la force d'une pseudo-arête. Une fois réalisées ces décrémentations et les mises en panne correspondantes, on peut procéder comme dans une branche-lourde sans filles. Les sacs-pères des sacs impactés ne sont pas indiqués pour ne pas alourdir le dessin.

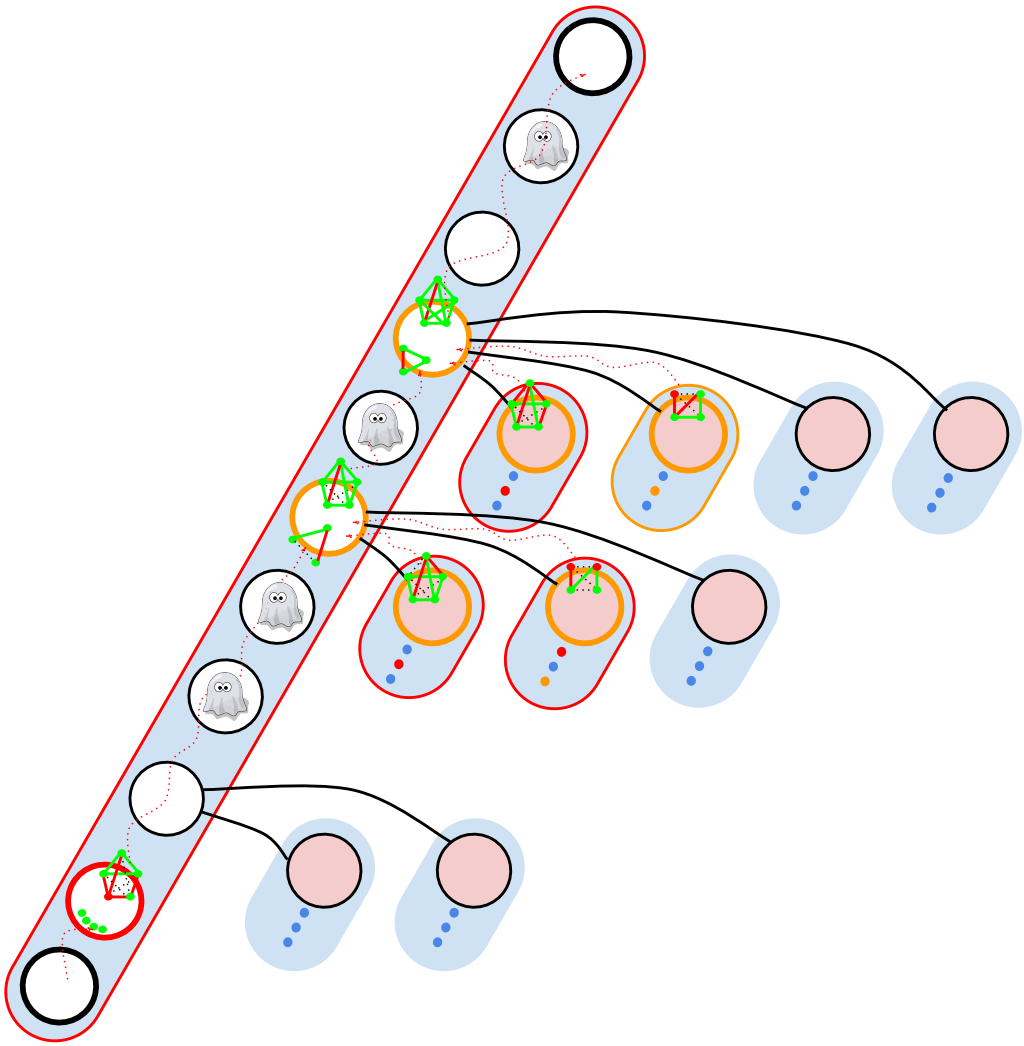


FIGURE 24 – Fonction d'aiguillage A : Tous les sommets du sac s sont en panne. $A(s, u)$ est la composante connexe de u dans $G \setminus s$

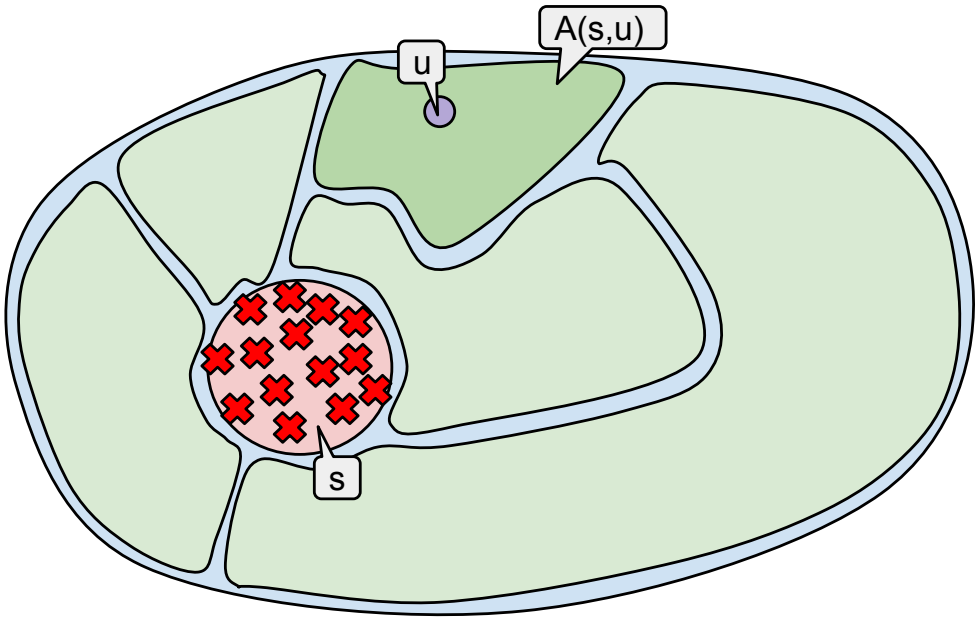
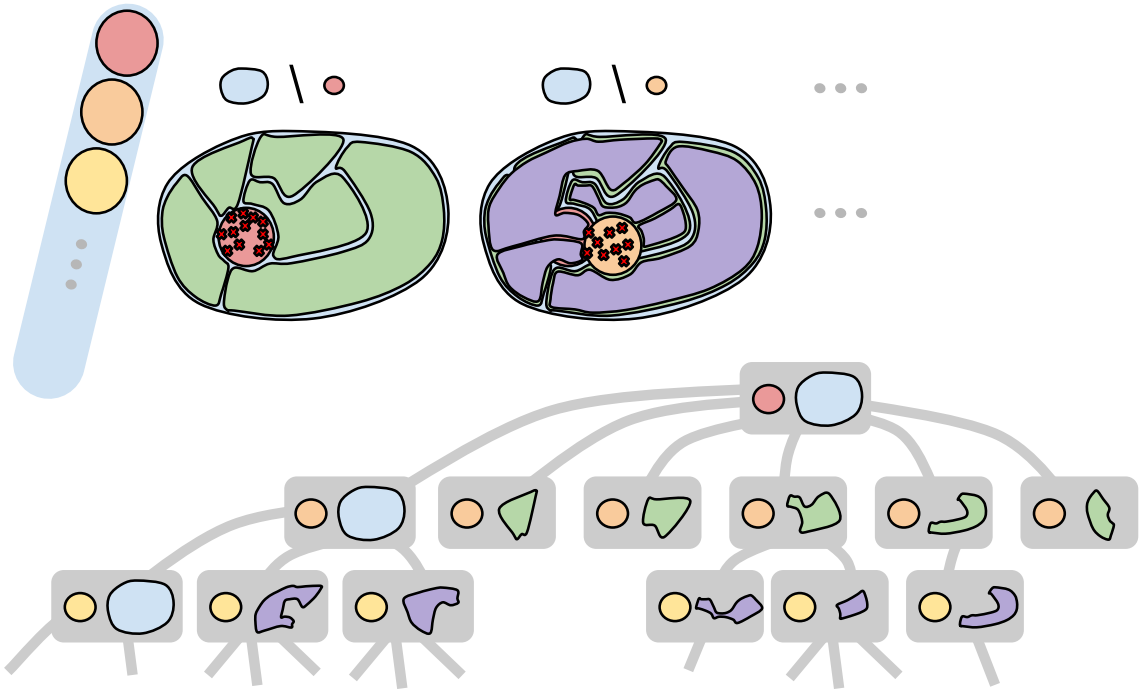


FIGURE 25 – Arbre d'aiguillage d'une branche.



— $(a(B), V(G))$ est la racine [14]
— pour tout sac s_i de B avec $i \geq 1$
  — $(s_i, V(G))$ est un sommet fils de $(s_{i-1}, V(G))$
  — pour toute composante connexe C de $G \backslash s_{i-1}$, (s_i, C) est un sommet dont le père est (s_{i-1}, C') telle que C' est le plus petit ensemble [15] contenant C parmi les sommets disponibles.

Soit u un sommet du sac s. On note $\mathcal{A}(s, u)$ le sommet (s, E) de $\mathcal{A}$ tel que $u \in E$ et que pour tout sommet (s, F) tel que $u \in F$ on a $E \subseteq F$. E peut être égal à $V(G)$ dans le cas où u appartient également au sac-père de s.

14. $a(B) = s_0$, les sacs de B sont notés s_i où l'indice i évolue de haut en bas.
15. Plus petit au sens de l'inclusion.

3.2.4 Structure de recherche de composante connexe

A partir du graphe de panne $P(G, X)$ et en utilisant une structure de donnée `Union-Find`, on peut construire une structure de recherche de la composante connexe d'un sommet.

On procède encore par traitement *bottom-up*. Pour tout sommet x d'un sac s du graphe de panne, on introduit (opération `Make-Set`), si ce n'est déjà fait, dans la structure `Union-Find` son identifiant $i(x)$ et sa composante connexe dans le sac traité[16] $c(x, s^+, X)$. On unit ensuite :

$$\texttt{Union}(\texttt{Find}(i(x)), \texttt{Find}(c(x, s^+, X))) \ .$$

Pour tout voisin déjà introduit v de x dans $P(G, X)$:

$$\texttt{Union}(\texttt{Find}(i(x)), \texttt{Find}(i(v)) \ .$$

3.2.5 Calcul d'une composante connexe

Soit un sommet u dont on désire connaître la composante connexe dans $G \setminus X$. On considère son occurence dans le sac $s = s(u)$, c'est-à-dire le sac le plus proche de la racine de l'arborescence où il est présent. Soit $B = B(s)$ la branche lourde correspondante.

— Si B n'a pas été traitée lors de la construction de $P(G, X)$, on considère son plus proche ancêtre B' parmis les branches traitées[17]. Comme vu dans les premiers chapitres de ce document, c'est équivalent à traiter une requête `Prédécesseur`. On peut tenter de relier u à un sommet v du sac s' de B', père d'un des apices de u, en temps $O(k^2)$, par la méthode des sacs fantômes. (Premier cas, figure 26.) On retourne alors la composante connexe de v (elle-même calculée comme dans le cas suivant où la branche à été traitée).
Si aucun sommet de l'intersection de s' vers le sac de u n'est accessible[18], on retourne le couple $(s', A(s', u))$ qui identifie la composante connexe. (Deuxième cas, figure 27.)

16. Pannes et pseudo-pannes.
17. S'il n'y en a pas, c'est qu'il n'y a pas de panne dans le graphe car la branche-racine hérite de toute panne.
18. Comme si tout s' était en panne.

— Si B a été traitée, on cherche le prédecesseur p de s et son successeur p' dans l'ensemble des sacs traités de B trié du plus éloigné au plus proche de la racine.
  — si $p = s$ (Troisième cas.), on renvoit le résultat de la recherche dans la structure `Union-Find` de la composante connexe du sommet x dans le sac augmenté s^+ en panne. Ce résultat se note $\texttt{Find}(c(x, s^+, X))$.
  — sinon on tente, en temps $O(k^2)$, par la méthode des sacs-fantômes, de relier u à un sommet v de p ou p'. En cas de succès, on renvoit $\texttt{Find}(i(v))$. (Quatrième cas, figure 28.) En cas d'échec, on déduit que u est déjà déconnecté du graphe de panne par les seules pannes de p' [19]. On retourne pour identifiant de la composante connexe de notre sommet : $\texttt{Routage}_{\mathcal{A}(B)}(\mathcal{A}(v', p'), \mathcal{A}(u, s))$ où v' est un sommet de l'intersection inférieure de p' (éventuellement en panne) relié à u *via* un chemin n'utilisant pas d'arête de p'. v' existe car G est connexe et il peut être déterminé par la méthode des sacs fantômes. (Cinquième cas, figure 29.)

3.2.6 Méta-étiquetage

Etiquetage. L'étiquette d'un sommet u doit contenir de l'information sur le sac $s(u)$ et ses $O(\log n)$ fantomes ainsi que sur ses $O(\log n)$ apices et les $O(\log n)$ fantômes de chacun d'entre eux. Apparaît donc le coefficient multiplicateur $O(\log^2 n)$ pour la taille des étiquettes.

La méthode des sacs fantôme impose $O(k \log n + k^2)$ bits d'information par sac connu. Il faut également disposer des étiquettes internes [20] de $O(k)$ sommets d'intersection par sac et de $O(k^2 \log n)$ bits d'onformation pour la gestion des pseudo-arêtes de chaque apex.

On doit aussi connaître les étiquettes de routage (de taille $O(\log n)$) dans l'arbre d'aiguillage de la branche de $O(k)$ sommets par sac.

19. u est déconnecté de p par les seules pannes de p' car les sommets de l'intersection supérieure de p ne sont pas en panne. S'ils l'étaient, p ne serait pas le prédécesseur de s car son père serait en panne. Les pannes à considérer dans p n'ont donc pas d'incidence sur la composante connexe de u.

20. Celles de l'étiquetage utilisé en boite noire.

FIGURE 26 – Calcul de la composante connexe de u : premier cas.

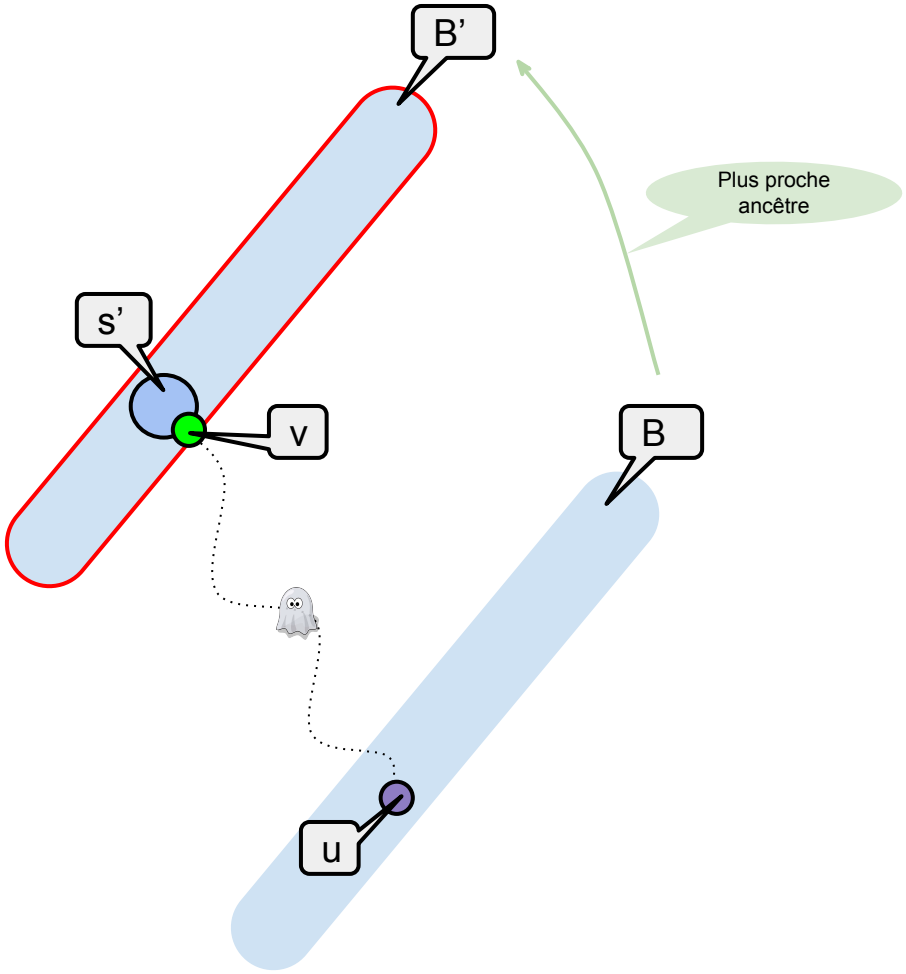


FIGURE 27 – Calcul de la composante connexe de u : deuxième cas.

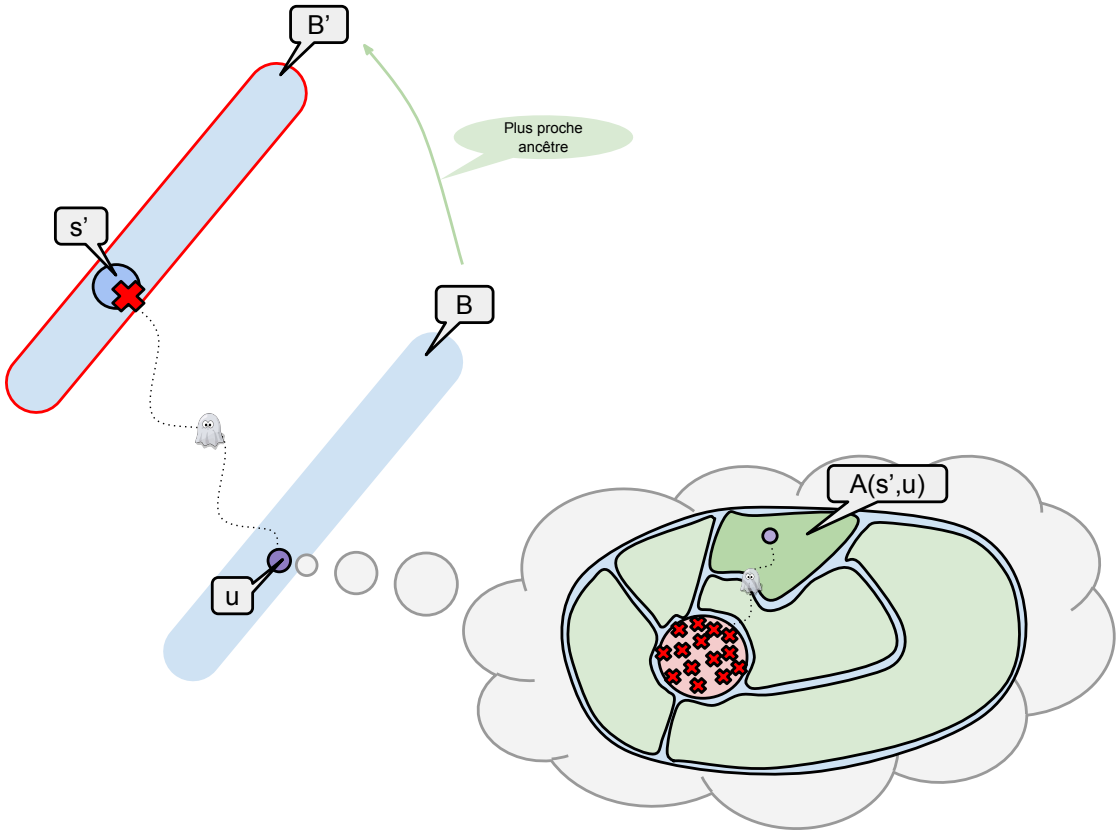


FIGURE 28 – Calcul de la composante connexe de u : quatrième cas.

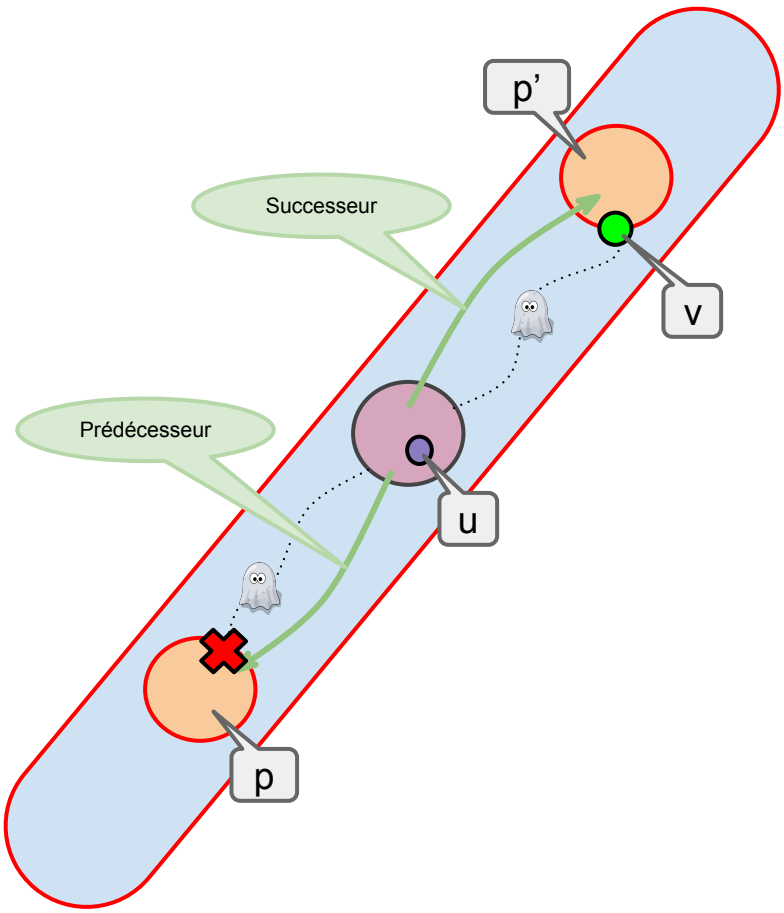
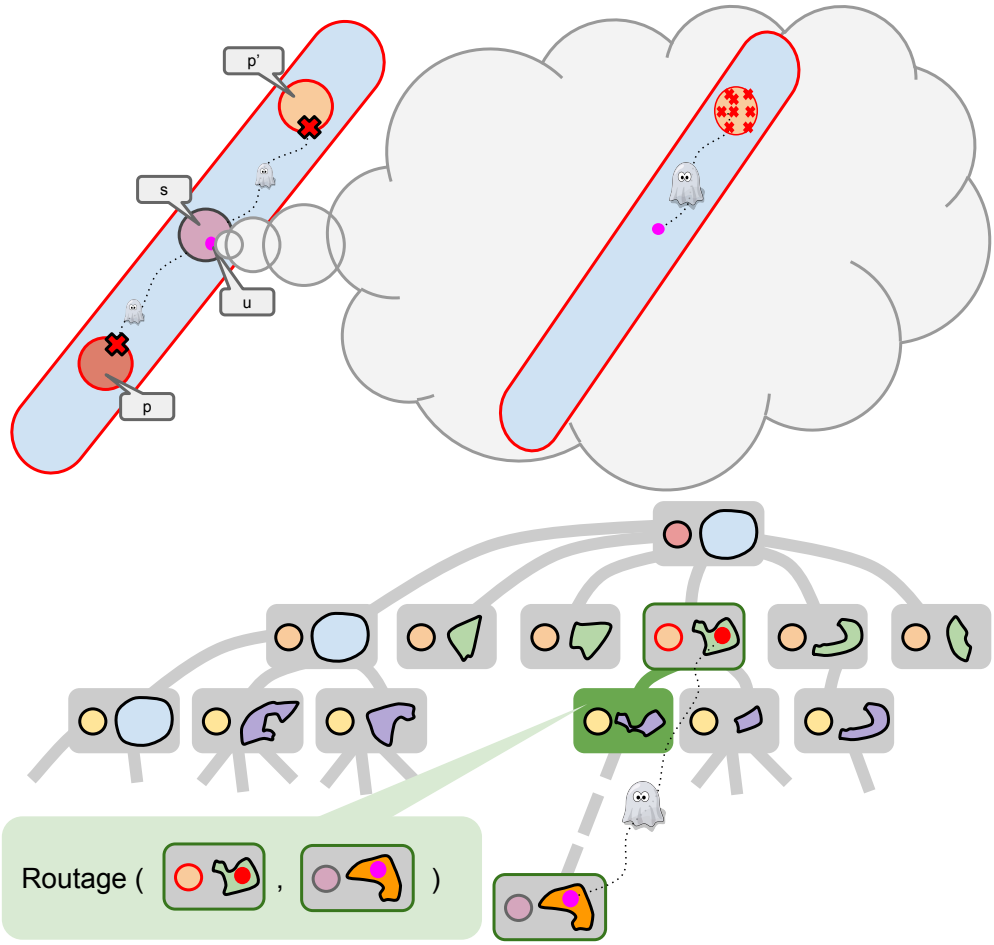


FIGURE 29 – Calcul de la composante connexe de u : cinquième cas.



On construit donc des étiquettes de taille $O(k^2 \cdot \log^2 n \cdot \lambda)$ où lambda est la taille maximale (au moins $\log n$) d'une étiquette interne. On n'a donc plus à proprement parler des étiquettes compactes. Cette contrainte est relâchée : on parle maintenant d'étiquetages *quasi-compacts* tant que les étiquettes demeurent de taille poly-logarithmiques.

Cet étiquetage peut être réalisé en temps polynomial :
— Structurations du graphe en branches lourdes et tris en temps polynomial.
— Construction des sacs augmentés en temps linéaire.
— Etiquetage interne en temps polynomial de chaque sac.
— Construction des graphes de reconnexion en temps linéaire.
— Construction des arbres et fonctions d'aiguillage en temps quadratique.

Pré-requête. Soit X une panne. On peut construire le graphe de panne en temps [21] :

$$|X| \cdot (p_\lambda(k^2) + k \cdot q_\lambda(|X|, n))$$

et la structure de recherche `Union-Find` en temps négligeable devant le précédent.

Turbo-requête. On peut alors effectuer une turbo-requête en temps :

$$\textsc{Pred}(|X|, n) + O(k^2) + \textsc{Find}(k \cdot x) + q_\lambda(|X|, n)) + \textsc{routage}(n^2) \ .$$

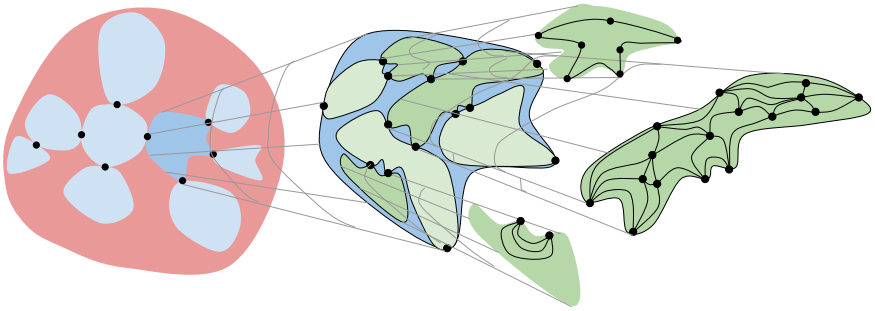
Ce qui constitue bien une turbo-requête au sens où on l'a défini car tous les termes de la somme sont négligeables devant $q_\lambda(|X|, n))$, à considérer que $\textsc{Pred}(|X|, n)$ est négligeable devant $(q_\lambda(|X|, n)))$ et que $k^2 = O(1)$.

3.2.7 Pannes sur les arêtes

Rien ne s'oppose à ce que notre méta-schéma supporte les pannes sur les arêtes, en revanche, si les schémas de départ ne fonctionnent que pour des pannes sur les sommets, alors le traitement des pseudo-arêtes n'est pas assuré

21. $p_\lambda(x)$ est le temps nécessaire au traitement d'une panne interne de x sommets et q_λ le temps de requête correspondant.

FIGURE 30 – On zoome progressivement de gauche à droite : (a) un graphe connexe et sa décomposition arboréscente en composante bi-connexes (b) l'une d'entre elles et sa décomposition arborescente en composantes 3-connexes, *bonds* et cycles



et on ne peut rien conclure du tout. On exige donc que les schémas utilisés en boîtes noires fonctionnent pour les pannes sur les arêtes.

3.2.8 Conclusion

Le théorème 38 est désormais établi. En découlera un corollaire immédiat pour la famille des graphes de largeur arborescente k : elle admet un schéma d'étiquetage d'urgence quasi-compact pour le problème CONNEXITÉ AVEC PANNE SUR LES SOMMETS ET LES ARÊTES. Les étiquettes sont de taille $O(k^2 \cdot \log^2 n)$ bits.

Ces résultats sont à comparer avec, d'une part [17], qui obtient des étiquettes de tailles $f(k) \cdot \log n$, où f est une fonction très grande obtenue à partir d'une méta-construction générale pour toutes les propriétés exprimables en logique du second ordre monadique sur les graphes de largeur arborescentes bornés (les pannes sur les arêtes n'y sont pas traitées). De l'autre, [16] explicite une construction dont les étiquettes sont de tailles $k^2 \log n$. Dans les deux cas, le traitement des pannes n'est pas adaptable à celui d'un schéma d'urgence. On pourra voir le tableau comparatif du chapitre de conclusion de ce document pour plus de précision.

3.3 Passage aux graphes 1-connexes

3.3.1 Arbre des composantes biconnexes

On appelle *composante biconnexe* (ou 2-*connexe*) d'un graphe, un sous-ensemble maximal de ses sommets tel que le sous-graphe qu'il induit est un graphe 2-connexe.

Si un graphe G est connexe, on peut construire un arbre dont les sommets sont ses composantes biconnexes et dont toute arête $C_i - C_j$ correspond à un sommet que les composantes biconnexes C_i et C_j ont en commun dans G.

On appelle un tel arbre *arbre des composantes biconnexes* de G. On notera qu'il n'est pas défini de manière unique. Par exemple, trois composantes biconnexes C_i, C_j et C_k partageant un sommet commun admettront les arbres $C_i - C_j - C_k$, $C_i - C_k - C_j$ et $C_j - C_i - C_k$ comme *arbre des composantes biconnexes*.

3.3.2 Arbre des composantes triconnexes

D'une manière analogue, on peut obtenir pour tout graphe 2-connexe la *décomposition de Tutte*, c'est-à-dire une arborescence de graphes triconnexes, de cycles et de *bonds* (graphes à deux sommets et au moins une arête). Une arête $C_i - C_j$ de l'arborescence correspond alors à l'intersection, réduite à un ensemble de 1 ou 2 sommets, entre les composantes 3-connexes (ou cycles, ou *bonds*) C_i et C_j.

3.3.3 Décomposition

Comme esquissé sur la figure 30, on pourra appréhender tout graphe, même non connexe, comme un ensemble de ses composantes connexes (dominante rouge sur la figure), chacune d'entre elles étant structurée en arbre des composantes biconnexes (bleues), elles-mêmes structurées en arbres des composantes triconnexes (vertes). Cette structuration nous permettra d'utiliser le théorème 38 pour étendre un schéma d'étiquetage des graphes 3-connexes

aux graphes de toutes connectivités.

3.4 Famille de graphes excluant H comme mineur

3.4.1 Théorème de structuration de graphe

3.4.1.1 Définitions préalables

Ces définitions sont celle présentées dans [52].

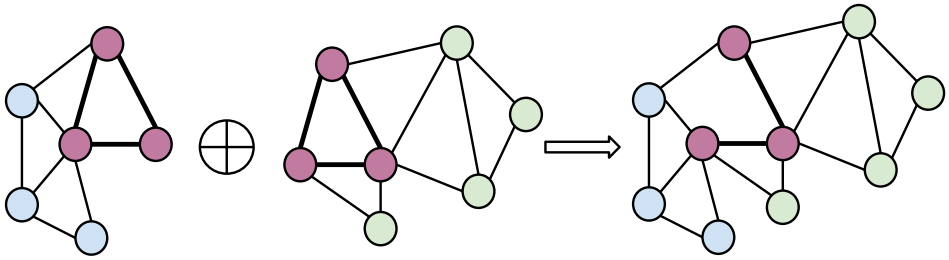
Clique-somme. Soit G un graphe dont (H_1, H_2) est une séparation (c'est-à-dire que H_1 et H_2 sont des sous-graphes de G sans arêtes en commun tels que $H_1 \cup H_2 = G$). Soit K le graphe complet tel que $V(K) = V(H_1) \cap V(H_2)$ avec $|V(K)| \leq k$. Soit $G_i = H_i \cup K$ pour $i \in \{1, 2\}$. On dit alors que G est une *k-clique-somme* de G_1 et G_2. On notera $G = G_1 \oplus_k G_2$ ou $G = G_1 \oplus G_2$ si la précision n'est pas nécessaire.

Réaliser une *k-clique-somme* entre G_1 et G_2 revient donc à identifier une clique de G_1 à une clique de G_2 de même taille au plus k, et à enlever un sous-ensemble arbitraire (éventuellement vide) des arêtes de cette clique dans le graphe obtenu. On verra la figure 31. On pourra aussi remarquer que $\oplus$ n'est pas un opérateur bien défini étant donné que des graphes différents peuvent résulter de la clique-somme de deux graphes donnés, par exemple en collant selon des cliques différentes ou en supprimant des sous-ensembles différents d'arêtes de la clique après collage.

Graphe h-presque-plongeable. On dit qu'un graphe G est *h-presque-plongeable* (voir figure 32) dans une surface $\mathcal{S}$ s'il existe un ensemble de sommets (dits *apices*[22], *apex* au singulier. Nous adopterons ici la désinence latine.) $A \subseteq V(G)$ de cardinalité au plus h et tel que le graphe $G \setminus A$ soit l'union d'un graphe G_0 plongé dans $\mathcal{S}$ et d'au plus h graphes disjoints $G_1, G_2, \cdots, G_k$

22. Ce mot est redéfini ici. On pourra oublier la définition donnée dans la preuve du théorème 38.

FIGURE 31 – Graphe obtenu par *clique-somme* selon un K_3 dont on conserve deux arêtes.



(avec $k \leq h$) appelés *vortices*, de *largeur-linéaire*[23] au plus h.

$G \setminus A$ peut donc s'écrire $G_0 \cup G_1 \cup \cdots \cup G_k$ (où $k \leq h$). La manière dont les vortices sont unis au graphe G_0 n'est pas arbitraire.

Vortices. Soit F une face d'un graphe G plongé sur une surface. Et soient $v_0, v_1, \cdots v_{m-1}, v_m$ les sommets bordant cette face (dans cet ordre circulaire) avec $v_0 = v_m$. Un *intervalle circulaire* pour F est un ensemble de sommets $\{v_a, v_{a+1}, \cdots, v_s\}$ où a et s sont des entiers et où les indices des v_i sont réduits modulo m. Soit Λ une liste finie d'intervalles circulaires de F.
Nous complétons le graphe G ainsi (voir fig 33) :

1. pour chaque intervalle circulaire $L \in \Lambda$, on ajoute un nouveau sommet v_L
2. pour tout $v_i \in L$, on peut ajouter ou non une arête $v_i - v_L$
3. pour toute paire d'intervalles de L et M de Λ, on ajoute une arête $v_L - v_M$ si $L \cap M$ est non vide.

Sous réserve qu'aucun des sommets du bord de F n'apparait dans plus de k intervalles de Λ, on dit qu'on a ajouté un *vortex de profondeur au plus* k au graphe G.

Les vortices sont des graphes de pathwidth bornée, c'est-à-dire qu'ils admettent une décomposition en chaîne de lageur bornée. Une *décomposition*

23. Ou *pathwidth*. La largeur linéaire d'un graphe est la largeur minimale d'une décomposition arborescente de ce graphe qui soit telle que son arbre-support est un chemin. Voir figure 34 et définition à venir.

FIGURE 32 – Graphe presque-plongeable dans le tore à 3 trous

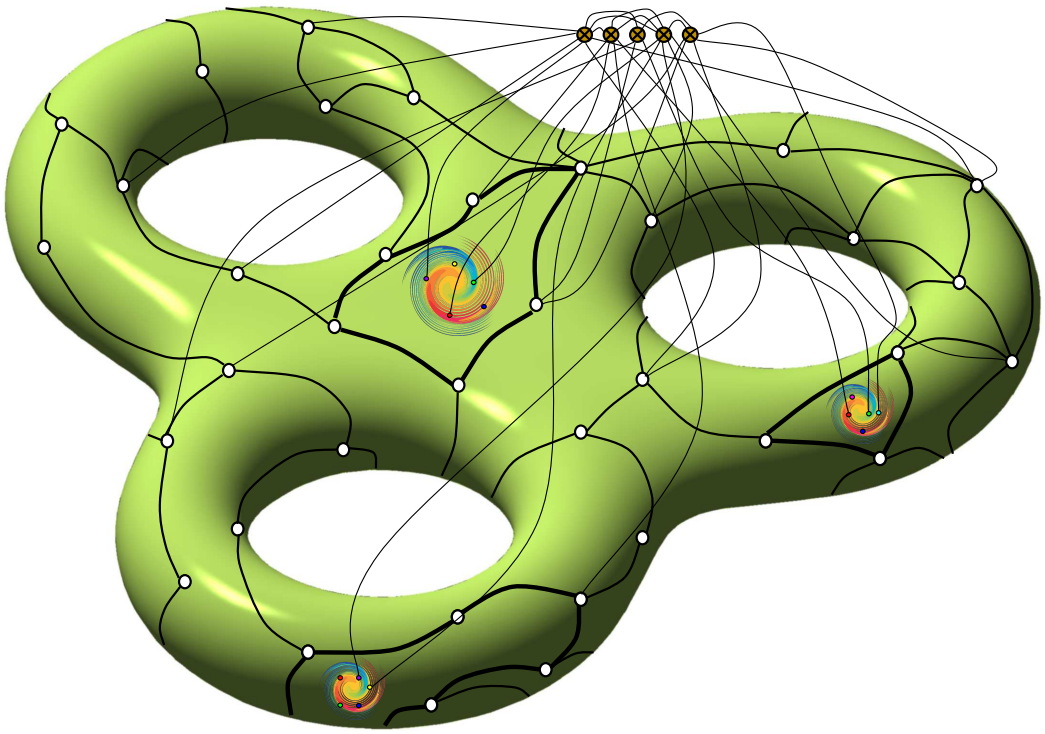


FIGURE 33 – Un vortex greffé sur une face F. Les sommets du bord de la face sont représentés en gris (forme oblongue) et ses arêtes en trait noir très épais (elles forment l'anneau). Un intervalle circulaire L est représenté par une bande de même couleur que son sommet v_L correspondant. Les arêtes construites au point *2* de la définition sont également de la même couleur que l'intervalle correspondant. Celles construites au point *3* sont épaisses et dessinées en noir. Nous utilisons le symbole du tourbillon de couleur apparaissant au centre de la figure pour résumer cette construction dans d'autres images.

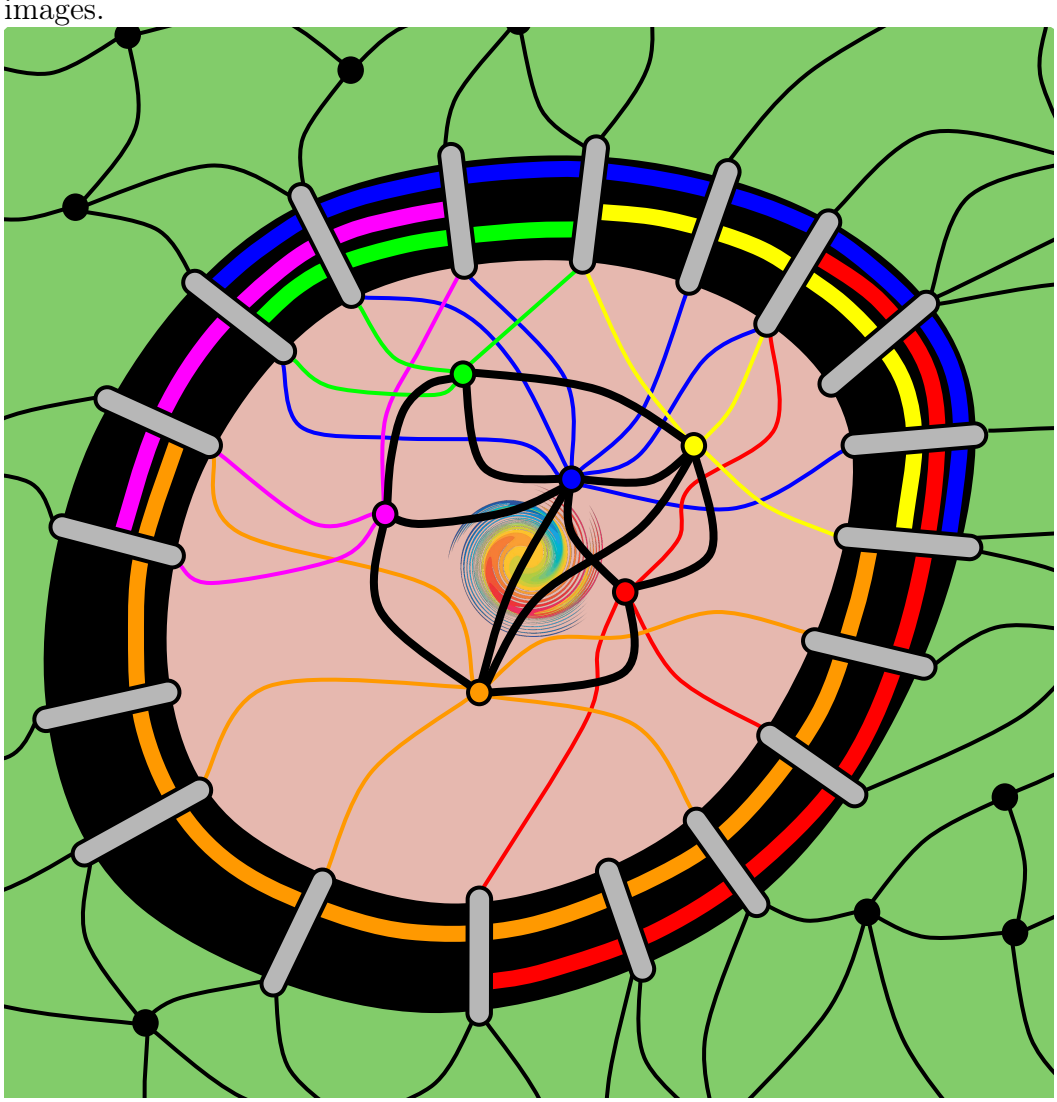
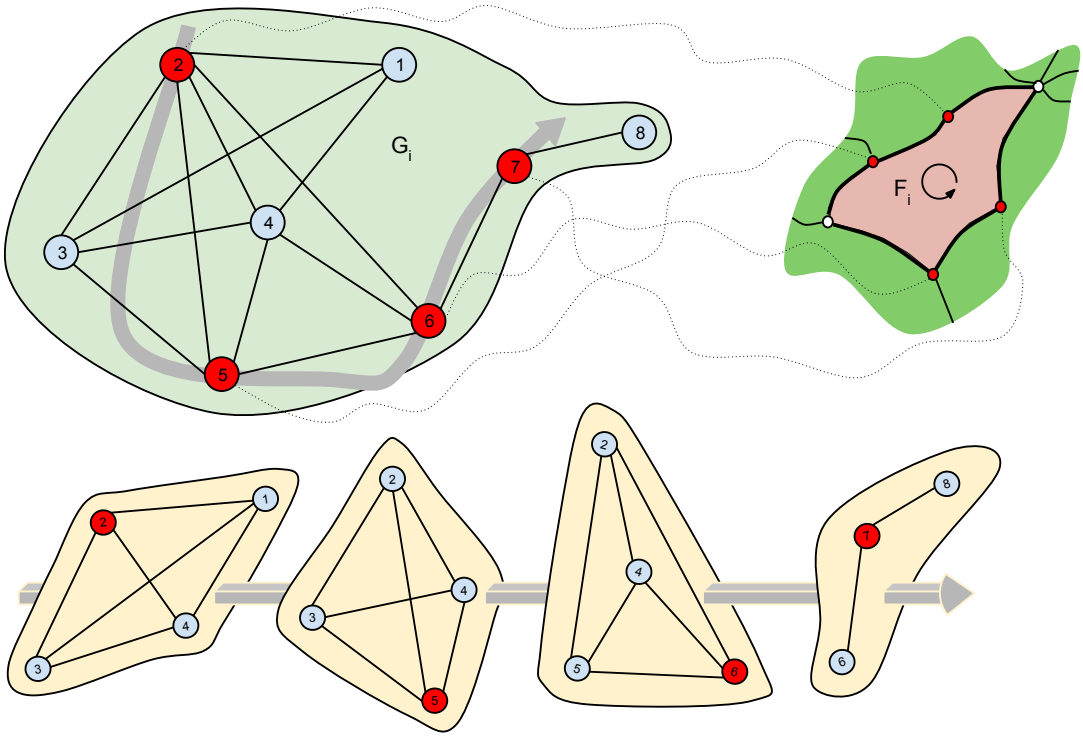


FIGURE 34 – Décomposition linéaire du vortex G_i et union à G_0 selon des sommets du bord de F_i. On s'affranchit ici de la définition basée sur les intervalles ciculaire pour ne voir en un vortex qu'un graphe de largeur linéaire bornée.



en chaîne (figure 34) d'un graphe G est une séquence $X_1, \ldots, X_l$ de sous-ensembles (appelés *sacs*) de $V(G)$ telle que :

- pour toute arête (u, v) de E, u et v soient tous deux contenus dans l'un des sacs
- les sacs dans lesquels un sommet u apparaît forment une sous-séquence de sacs consécutifs.

Pour chaque G_i, il existe une face F_i de G_0 dans $\mathcal{S}$ telle qu'on puisse associer injectivement au j-ème chaque sac X^i_j de la décomposition en chaîne de G_i, un sommet du bord de F_i, noté x^i_j et que la séquence[24] des $(x^i_j)_{1 \leq j \leq d_i}$ respecte l'ordre circulaire du bord de F_i.

24. avec d_i le nombre de sacs de la décomposition en chaîne du graphe G_i.

L'union de G_i à G_0 se fait par identification de (x^i_j) à l'un des sommets du sac X^i_j. (figures 34 et 35)

La définition du vortex et sa caractérisation en graphe de pathwidth bornée nous seront toutes deux utiles ultérieurement.

Apices. Les *apices* sont donc, comme évoqué précedemment, les sommets n'appartenant pas à l'union de G_0 et des vortices. ils sont potentiellement connectés à n'importe quel sommet du graphe G, qu'il soit élément d'un G_i ou même un autre apex. On pourra retourner à la figure 32 et y observer les 5 sommets arborant une croix en X et semblant être source de désordre : ce sont les apices.

Structure arborescente Si un graphe G peut être construit par k-clique-sommes successives de graphes appartenant à la classe $\mathcal{C}$, on dit que G est une *k-structure-arborescente* sur $\mathcal{C}$.

3.4.1.2 Énoncé du théorème de structuration

Le *Théorème de structuration des graphes sans mineur* [52] affirme que pour tout graphe H, il existe un entier $k = k(H)$ tel que tout graphe G n'admettant pas H comme mineur peut être obtenu, par k-clique-sommes successives, à partir d'une liste de graphes dont chaque élément est k-presque-plongeable dans une surface où H n'est pas plongeable.

3.5 Méta-schémas auxiliaires pour les graphes h-presque-plongeable

3.5.1 h-vortex + h apices = (h, h)-vortex

Un h-vortex d'au plus n sommets a une pathwidth bornée. Sa décomposition linéaire peut donc être vue comme un cas particulier de h-arborescence. De même, un h-vortex auquel on rajoute k sommets quelconques et arbitrairement reliés (entre eux et aux sommets de départ) peut être vu comme une

FIGURE 35 – Un vortex G_i vu comme un graphe de largeur linéaire bornée (chaque sommet rouge est ici représentant d'un des sacs de la décomposition de la figure 34.) greffé sur la face F_i qui lui correspond.

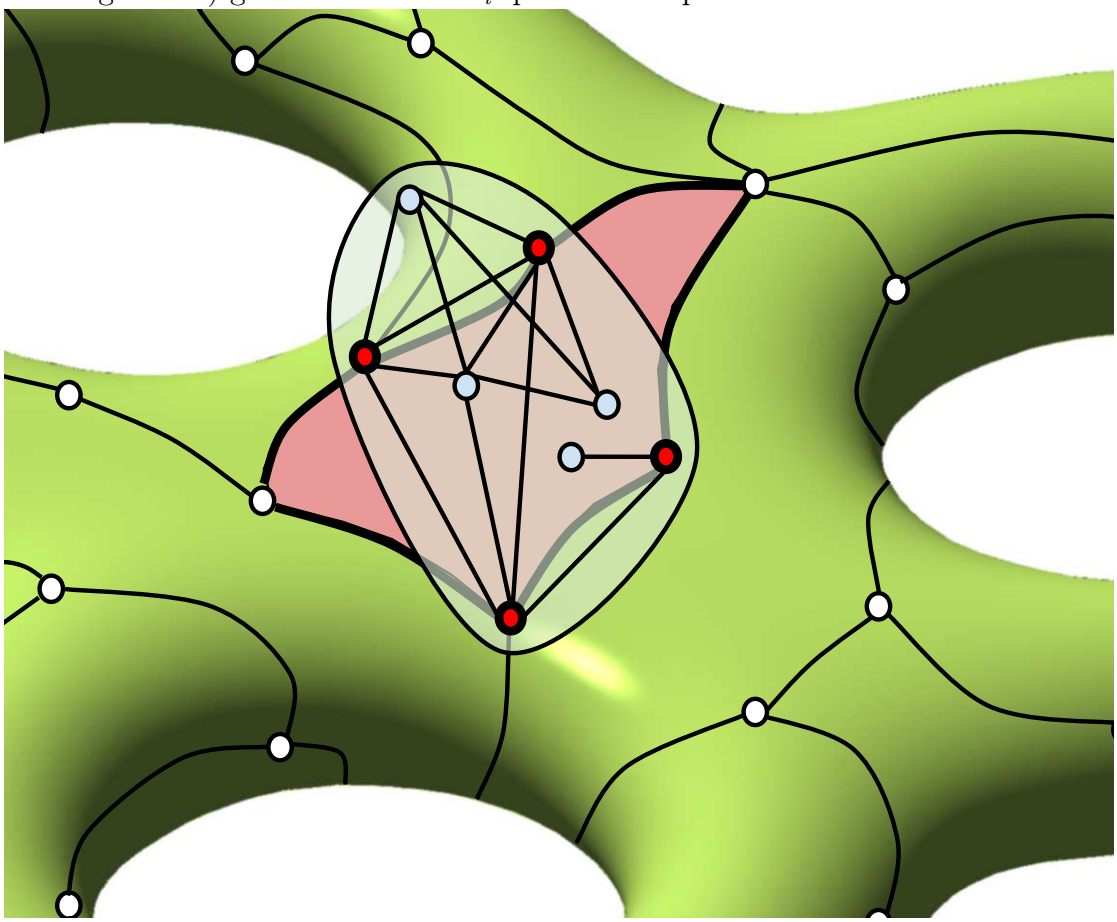
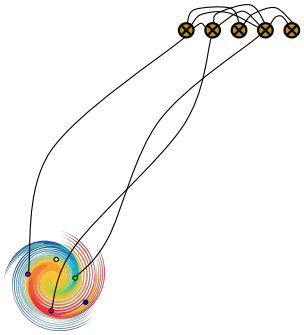


FIGURE 36 – (h, h)-vortex



$(h + k)$-arborescence. Il suffit pour celà d'ajouter les k sommets dans chacun des sacs. En particulier, si on ajoute les h apices d'un graphe presque-plongeable à un de ses vortices, on obtient un (h, h)-*vortex* (fig. 36) et sa décomposition linéaire

— dont les sacs sont des graphes dont le nombre de sommet est borné par la constante $2 \cdot h$ et admettent un schéma d'étiquetage compact d'urgence trivial [25] dont les étiquettes sont de taille constante $2^{O(h)}$, dont le temps de pré-requête est constant [26] et dont le temps de turbo-requête est également constant. [27]

— dont le nombre de sacs est borné par n.

Donc d'après le théorème 38 :

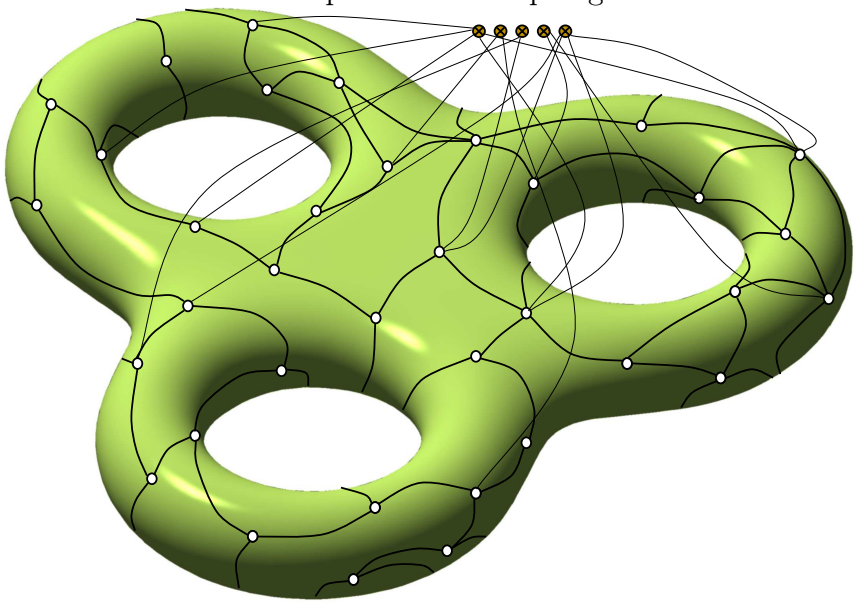
Théorème 40. *Pour tout h fixé, la classe des (h, h)-vortices à n sommets admet un schéma d'étiquetage d'urgence quasi-compact.*

3.5.2 3-connexe plongé + h apices = 3-connexe plongé h-enrichi

D'après le théorème 34, on sait qu'un graphe G_0 3-connexe plongé sur une surface admet un arbre couvrant T de degré borné par une constante Δ proportionnelle au genre du graphe. On peut donc enrichir un tel graphe de

25. Chaque étiquette contient la liste de toutes les pannes possibles associées à la composante de chacun des sommets.
26. Choix d'une panne dans la liste.
27. Lecture de la composante connexe.

FIGURE 37 – Graphe 3-connexe plongé et h-enrichi



h apices $(a_i)_{1\leq i\leq h}$ et des arêtes qui l'unissent à eux (voir fig. 37) en utilisant le méta-schéma du théorème 27.

On utilise le graphe $\mathcal{C} = T$ sur lequel on coud $M_0 = G_0$ et chaque étoile M_i de sommet central a_i. Notons que la famille des étoiles est incluse dans celle des arbres et admet donc un schéma d'étiquetage d'urgence d'après le théorème 8.

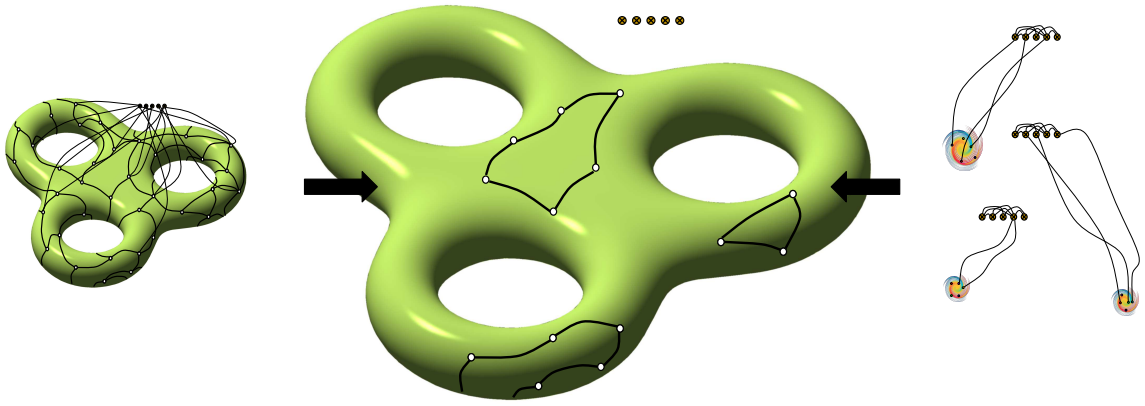
Théorème 41. *La classe des graphe* 3*-connexes plongés sur une surface de genre borné et* h*-enrichis admet un schéma d'étiquetage compact.*

3.5.3 3-connexe plongé h-enrichi + h (h,h)-vortices = h-presque-3-connexe-plongeable

Un graphe G h-presque-3-connexe-plongeable[28] se découd favorablement (voir fig. 38) selon un graphe $\mathcal{C}$ d'au plus n sommets, union de h cycles vorticiels et des h apices isolés, dont le nombre d'ancrage est $2 \cdot h + 2$. On obtient par découpage les graphes :

28. La partie plongée est 3-connexe.

FIGURE 38 – Obtention par coûture d'un graphe h-presque-3-connexe-plongeable



— M_0, graphe 3-connexe plongé h-enrichi
— et, pour tout $1 \leq i \leq h$, M_i un (h, h)-vortex.

Ceci, avec le théorème 27, établit le théorème suivant :

Theorème 42. *La famille des graphes h-presque-3-connexe-plongeables admet un schéma d'étiquetage compact d'urgence pour* COMPOSANTE CONNEXE AVEC PANNE SUR LES SOMMETS.

De plus, étant donné que toutes ces constructions sont généralisables aux pannes sur les arêtes par subdivision, on obtient le théorème :

Theorème 43. *La famille des graphes h-presque-3-connexe-plongeables admet un schéma d'étiquetage compact d'urgence pour* COMPOSANTE CONNEXE AVEC PANNE SUR LES SOMMETS ET LES ARÊTES.

3.6 Conclusions du chapitre

En considérant la famille des graphes h-presque-plongeables comme étant des $(k+2)$-arborescences de graphes h-presque-3-connexe-plongeables, le théorème 38 nous permet d'étendre le théorème 43 au théorème suivant :

Theorème 44. *La famille des graphes h-presque-plongeables admet un schéma d'étiquetage quasi-compact d'urgence pour* Composante Connexe avec Panne sur les Sommets et les Arêtes.

Et donc, en utilisant à nouveau le théorème 38 sur les h-structures-arborescentes de graphes h-presque-plongeables, on établit le théorème :

Theorème 45. *La famille des graphes excluant un mineur fixé admet un schéma d'étiquetage quasi-compact d'urgence pour* Composante Connexe avec Panne sur les Sommets et les Arêtes.

De plus, en utilisant la structuration du paragraphe 3.3.3, nous pouvons aussi étendre le schéma du théorème 7 a un schéma quasi-compact pour les graphes de toutes connectivités.

Theorème 46. *La famille des graphes de genre eulérien borné admet un schéma d'étiquetage quasi-compact d'urgence pour* Composante Connexe avec Panne sur les Sommets et les Arêtes.

Et comme annoncé au paragraphe 3.2.8, nous obtenons également un résultat sur les graphes de largeur arborescente bornée :

Theorème 47. *La famille des graphes de largeur arborescente bornée admet un schéma d'étiquetage quasi-compact d'urgence pour* Composante Connexe avec Panne sur les Sommets et les Arêtes.

Conclusion

Résumé de notre contribution

Notre contribution scientifique est concentrée dans deux des chapitres de cette thèse, à savoir le chapitre 2 et le chapitre 3. Dans chacune de ces divisions est introduit un *méta-schéma* d'étiquetage permettant d'assembler les résultats obtenus sur des classes de graphes structurellement peu complexes et d'obtenir ainsi des résultats sur des classes structurellement plus élaborées.

Ces résultats sont compilés et comparés à l'existant dans le tableau qui suit. Pour chacun des résultats (ceux dont nous ne sommes pas auteur sont grisés), le tableau indique :

— la famille de graphe sur lequel il est effectif
— la nature de la panne supportée (S pour sommets, A pour arêtes). Les complexités sont indiquée pour des pannes sur les sommets dans le tableau.
— le temps d'encodage de la structure de donnée
— la taille en bits d'une étiquette ou de la structure totale si non pertinent
— le temps de pré-requête suite à la déclaration d'une panne
— le temps de turbo-requête

Th.	**Famille**	**Panne**	**Encodage**	**Etiquette**	**Pré-calcul de X**	**Calcul** $\textsc{Conn}_{S(X)}(u,v)$
[51]	arbitraires	A	$n+m$	oracle	$\lvert X\rvert \log^2 n \log\log n$	$\log^2 n \log\log n$
[10]	planaires	S(A)	n	oracle	$\lvert X\rvert \log\log n$	$\log\log n$
[14]	planaires 3-c	S(A)	n	$\log n$	$\lvert X\rvert^2$	$\log n$
[15]	planaires	S(A)	n	$\log n$	$\lvert X\rvert^2$	$\log n$
[16]	tw k	S(A)	$k^2 n \log n$	$k^2 \log n$	$\lvert X\rvert^2$	$\log n$
8	arbres	S(A)	n	$\log n$	$\lvert X\rvert \log\log n$	$\log\log n$
23	planaires ext.	S(A)	n	$\log n$	$\lvert X\rvert \log\log n$	$\log\log n$
32	planaires 3-c	S(A)	n	$\log n$	$\lvert X\rvert \log\log n$	$\log\log n$
37	genre g 3-c	S(A)	$(n+g)n$	$g \log n$	$\lvert X\rvert \log\log n$	$\log\log n$
46	genre g	S(A)	n^2	$g \log^3 n$	$\lvert X\rvert \log\log n$	$\log\log n$
47	treewidth k	S(A)	n^2	$k^2 \log^2 n$	$\lvert X\rvert \log\log n$	$\log\log n$
45	H-minor-free	S(A)	n^2	$f(H) \log^5 n$	$\lvert X\rvert \log\log n$	$\log\log n$

Nous avons voulu, pour conclure ce document, illustrer graphiquement les constructions successives que nous avons été amené à réaliser. Ces illustrations seront accompagnées, à chaque étape, des numéros de théorème correspondant.

Tout d'abord, la figure 39 nous sert à matérialiser l'utilisation du théorème 27. Sur fond rouge, nous précisons quelles sont les schémas de départ que nous utilisons et qui nous permettent d'obtenir un nouveau schéma d'étiquetage d'urgence de connexité pour une famille obtenue par coûture.

Nous pouvons alors illustrer ainsi les résultats du chapitre 2 tel qu'effectué sur la figure 40. Nous voyons, en parcourant ce schéma de bas en haut, comment l'objectif de ce chapitre, à savoir de construire un schéma d'étiquetage compact de connexité des graphes 3-connexes plongés sur des surfaces, est réduit aux problèmes PREDECESSEUR dans un ensemble d'entiers triés et à ROUTAGE dans un arbre.

Quant à elle, la figure 41 matérialise le méta-schéma du théorème 38. Il nous permet d'obtenir un schéma d'étiquetage d'urgence de connexité quasi-compact pour une famille structurée en k-arborescence de graphes d'une autre famille, pour laquelle un schéma compact ou quasi-compact nous est

FIGURE 39 – Méta-schéma du chapitre 2. On obtient un schéma d'étiquetage d'urgence de connexité pour la famille des graphes obtenus par coûture. Le phylactère indique un numéro de théorème. « CPSA » indique un schéma d'urgence compact pour panne sur les sommets et arêtes.

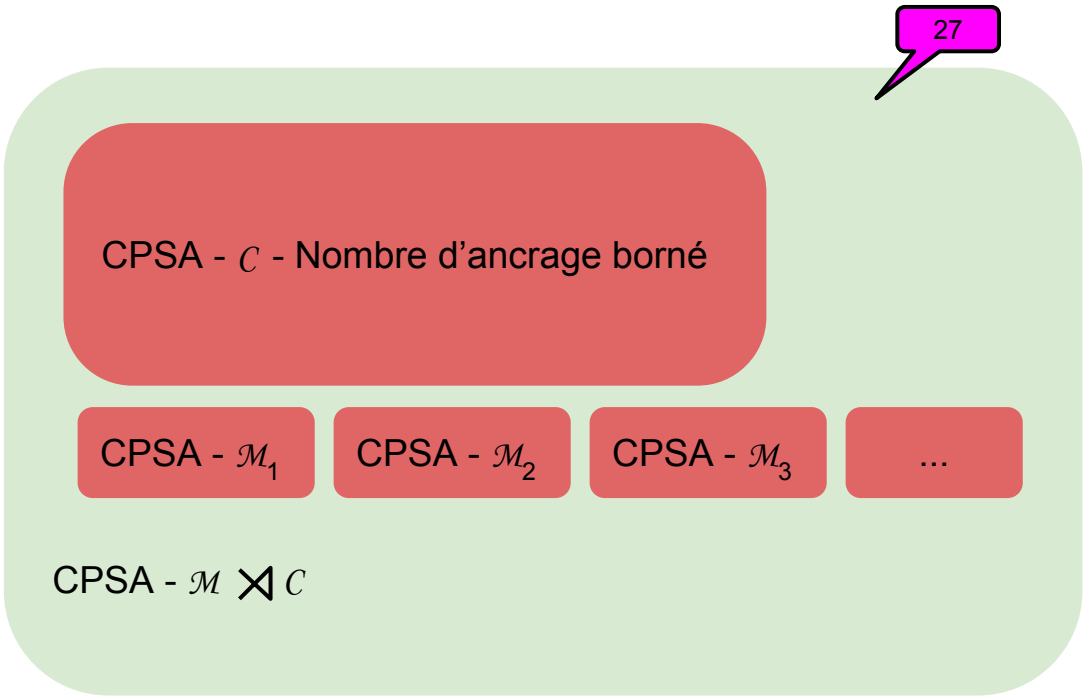


FIGURE 40 – Schéma de dépendance des resultats du chapitre 2. « CPS » indique un schéma sur les sommets uniquement.

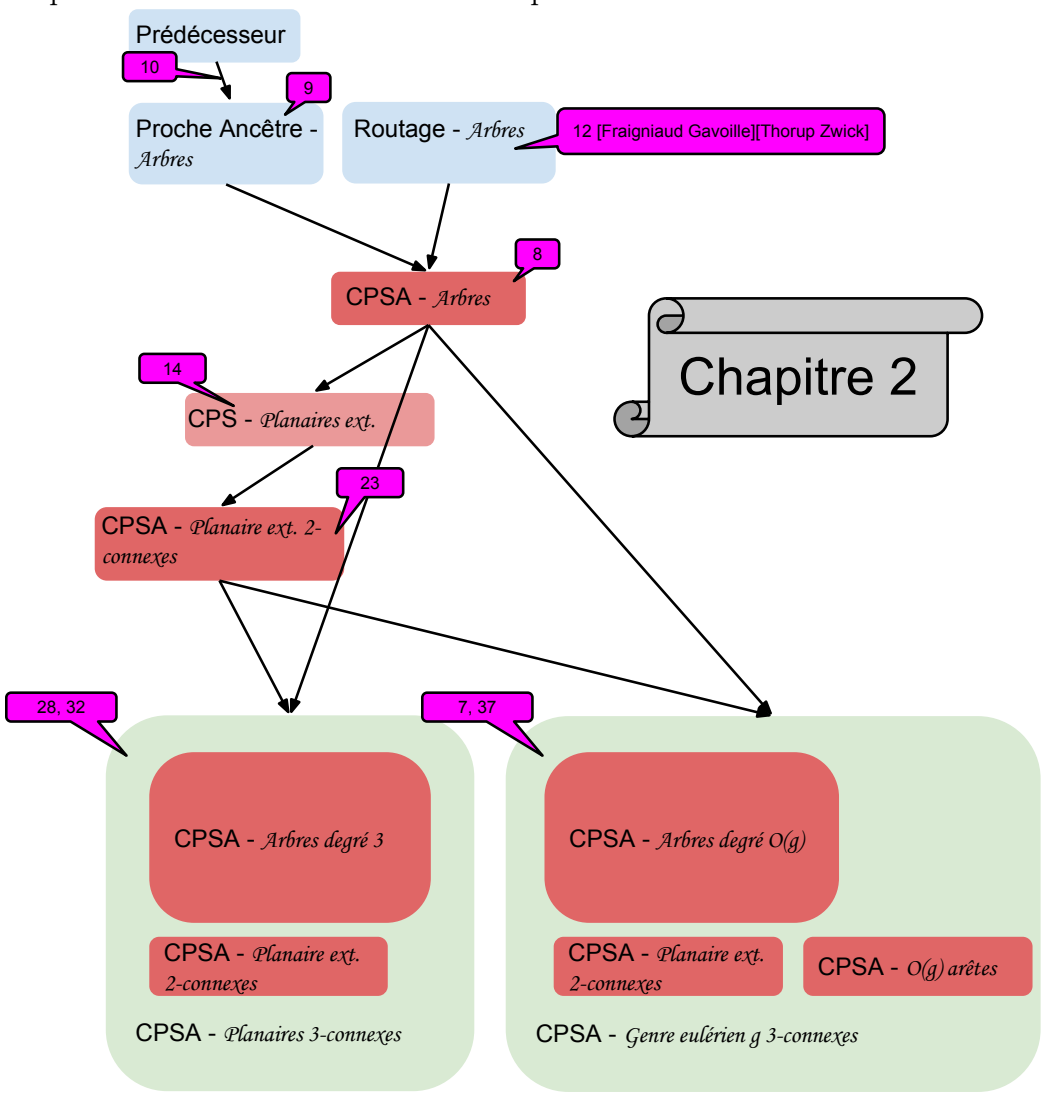
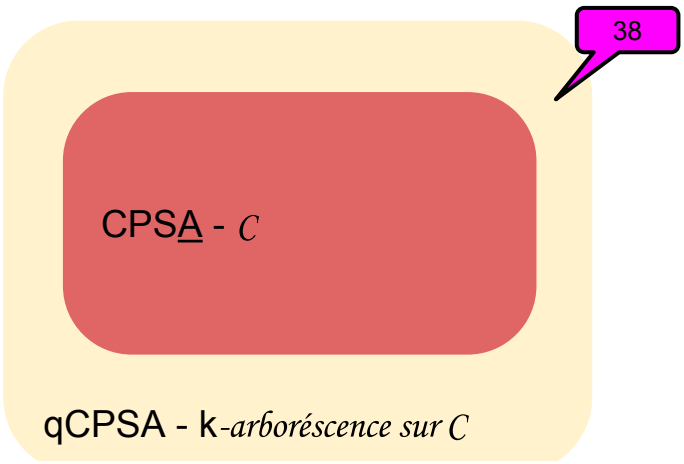


FIGURE 41 – Méta-schéma du chapitre 3 sur les k-arboréscences.



livré.

On peut alors représenter (figure 42) les résultats du chapitre 3 comme nous l'avons fait pour le précédent. Nous voyons comment nous nous appuyons sur les résultats précédents pour obtenir un schéma d'étiquetage d'urgence de connexité, quasi-compact, pour la famille des graphes n'acceptant pas le graphe H comme mineur.

Enfin, nous *branchons* les résultats des deux chapitres 2 et 3 pour obtenir une cartographie de l'ensemble de nos résultats.

Limites et perspectives

Implémentation. Nous avons dans cette thèse construit des schémas d'étiquetages d'urgence de connexité pour diverses familles de graphes. Ces schémas sont des construction théoriques, on peut s'interroger sur la faisabilité de leur implémentation.

Le schéma de Courcelle *et al.*[14] pour les graphes planaires 3-connexes avait été implémenté par nos soins[29]. L'implémentation de notre schéma

29. Ce résultat constituant l'essentiel de notre stage de fin de Master.

FIGURE 42 – Schéma de dépendance des resultats du chapitre 3. « qCPSA » indique que le schéma est quasi-compact.

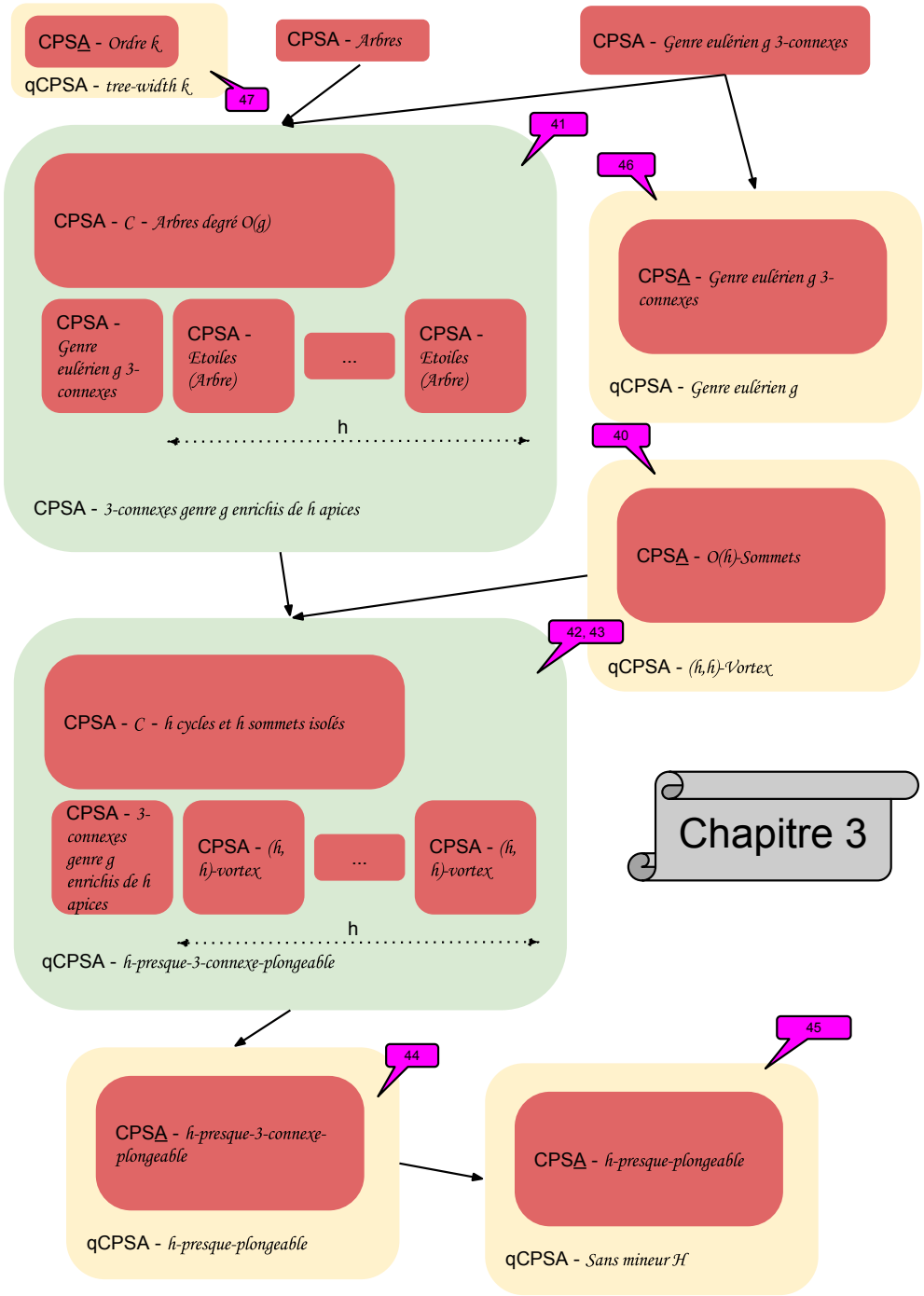
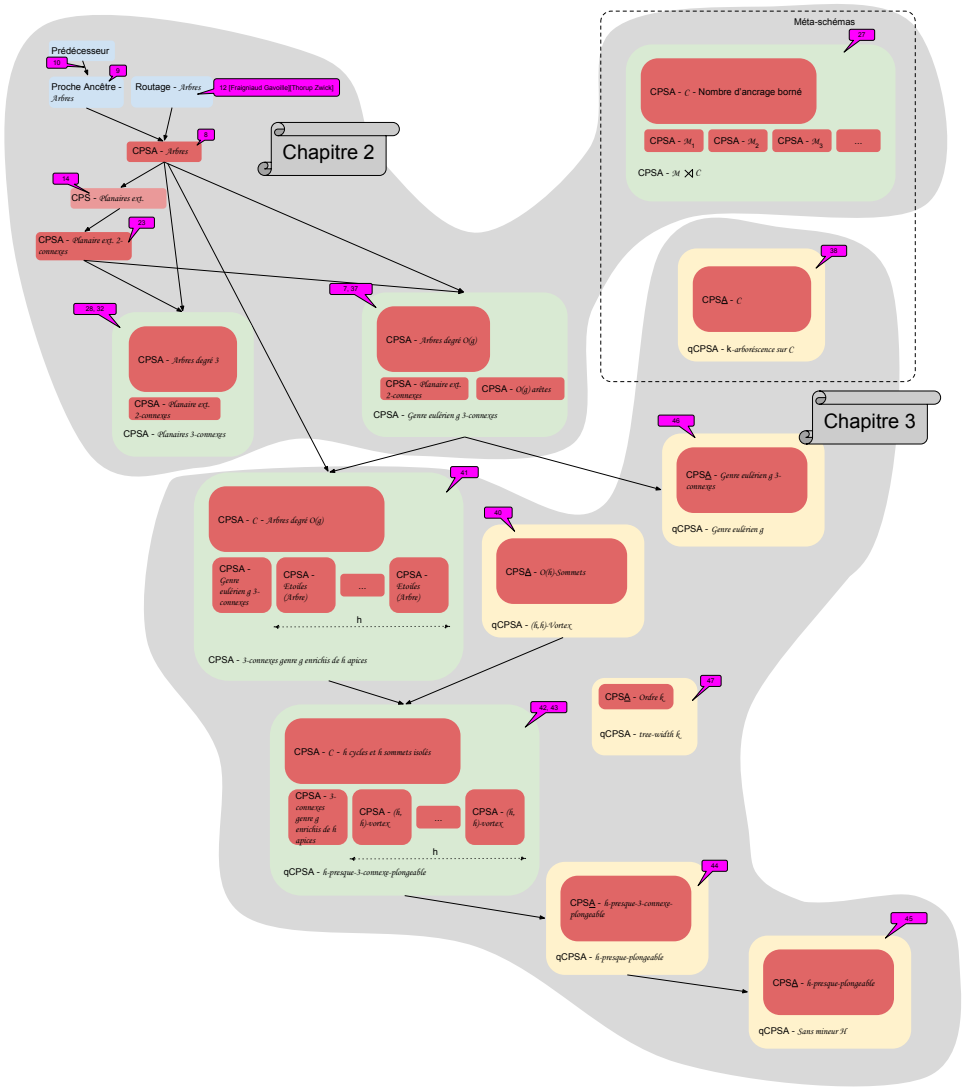


FIGURE 43 – Schéma de dépendance des resultats de ce document.



d'urgence pour la famille des arbres semble assez élémentaire et constitue un de nos objectifs à court terme, de même que celui, un peu moins trivial sur les graphes planaires extérieurs.

En revanche, les résultats sur les classes de graphes plus complexes[30] semblent requérir des outils algorithmiques perfectionnés, ne serait-ce que pour découper les graphes plongés sur les surfaces. Un état de l'art des algorithmes disponibles devra précéder l'implémentation de nos schémas.

Améliorations. Nos schémas d'urgence ont un temps de turbo-requête optimal $\textsc{Pred}(|X|, n)$. De plus, l'étape de tri pour la pré-requête semble difficile à éviter, et le temps $\textsc{Sort}(|X|, n)$ semble incompressible. C'est sur la taille des étiquettes que nos espoirs d'amélioration se tournent. Nous laissons donc ouverte la qustion : existe-t-il un schéma d'étiquetage *d'urgence* et *compact* pour les familles où nous n'avons obtenus qu'un schéma *quasi-compact* ? Nous tendons à croire que oui, mais qu'un changement de stratégie s'impose : il faut repenser nos méta-schémas.

Une autre question : nos constructions, moyennant quelques adaptations éventuelles, permettent-elles de généraliser nos schémas à d'autres problèmes ? Par exemple Distance avec Panne, Nombre de Composante connexe avec Panne ou Tailles ces Composantes Connexes avec Panne seraient peut-être de bons candidats.

Un autre axe d'amélioration : certains de nos schémas de connexités tolèrent la réparation de pannes ou l'ajout d'arêtes et de sommets. On peut imaginer une structure de données pseudo-dynamique tenant à jours sous forme de pannes et réparations les évolutions de l'état d'un oracle de connexité jusqu'à un re-calcul total périodique. C'est une direction que nous aimerions suivre également.

30. Complexes au sens de la profondeurs des imbrications de résultats dans nos preuves.

Médiagraphie

[1] *Warhol photo exhibition, Stockholm, 1968 : Kaplan, Justin, ed., Bartlett's Familiar Quotations, 16th Ed.* Little, Brown & Co, 1992. x

[2] Crazy german kid, 2007. http://www.youtube.com/watch?v=M8pR1rZZHEs (disponible au 01/09/2013 et mise en ligne par Dunney http://www.youtube.com/user/Dunney). vii

[3] Serge Abiteboul, Stephen Alstrup, Haim Kaplan, Tova Milo, and Theis Rauhe. Compact labeling schemes for ancestor queries. *SIAM Journal on Computing*, 35(6) :1295–1309, 2006. xii

[4] Takuya Akiba, Yoichi Iwata, and Yuichi Yoshida. Fast exact shortest-path distance queries on large networks by pruned landmark labeling. In *SIGMOD Conference*, pages 349–360, 2013. v

[5] David Barnette. Trees in polyhedral graphs. *Canad. J. Math*, 18 :731–736, 1966. 58

[6] A. Beky. Sécurité informatique : le marché mondial en croissance de 8 *silicon.fr*, 2013. http://www.silicon.fr/logiciel-securite-informatique-marche-2012-86583.html (disponible au 01/09/2013). viii

[7] G. Bernanos. *La France contre les robots*. Plon, 1944. ix, xi

[8] Nicolas Bonichon, Stefan Felsner, and Mohamed Mosbah. Convex drawings of 3-connected plane graphs. *Algorithmica*, 47(4) :399–420, 2007. 2, 3, 15

[9] Nicolas Bonichon, Cyril Gavoille, and Nicolas Hanusse. Canonical decomposition of outerplanar maps and application to enumeration, coding and generation. In 29^{th} *International Workshop on Graph-Theoretic Concepts in Computer Science (WG)*, volume 2880 of Lecture Notes in Computer Science, pages 81–92. Springer-Verlag, June 2003. 34, 36, 37

[10] Glencora Borradaile, Seth Pettie, and Christian Wulff-Nilsen. Connectivity oracles for planar graphs. In *Algorithm Theory–SWAT 2012*, pages 316–327. Springer, 2012. 104

[11] L. Carroué. Industrie, socle de la puissance. *Le Monde Diplomatique, mars 2012*, 2012. http://www.monde-diplomatique.fr/2012/03/CARROUE/47485 (disponible au 01/09/2013). x

[12] Timothy M. Chan and Mihai Pătraşcu. Transdichotomous results in computational geometry, I : Point location in sublogarithmic time. *SIAM Journal on Computing*, 39(2) :703–729, June 2009. 27

[13] James Cheng, Silu Huang, Huanhuan Wu, and Ada Wai-Chee Fu. Tf-label : a topological-folding labeling scheme for reachability querying in a large graph. In *SIGMOD Conference*, pages 193–204, 2013. v

[14] Bruno Courcelle, Cyril Gavoille, Mamadou Mustapha Kanté, and David Andrew Twigg. Connectivity check in 3-connected planar graphs with obstacles. In *International Conference on Topological & Geometric Graph Theory*, volume 31, pages 151–155. Electronic Notes in Discrete Mathematics, August 2008. xv, 1, 5, 8, 10, 66, 67, 104, 107

[15] Bruno Courcelle, Cyril Gavoille, Mamadou Mustapha Kanté, and David Andrew Twigg. Optimal labelling for connectivity checking in planar networks with obstacles. Technical Report hal-00367746, HAL, March 2009. xv, 1, 104

[16] Bruno Courcelle and David Andrew Twigg. Constrained-path labellings on graphs of bounded clique-width. *Theory of Computer Systems*, February 2009. 90, 104

[17] Bruno Courcelle and Rémi Vanicat. Query efficient implementation of graphs of bounded clique-width. *Discrete Applied Mathematics*, 131 :129–150, 2003. xi, 17, 90

[18] M. De Berg, O. Cheong, M. Van Kreveld, and M. Overmars. *Computational geometry : algorithms and applications*. 17

[19] J. de La Fontaine. La cigale et la fourmi, 1668. in *Fables*, Livre I. viii

[20] R. Debray. *Eloge des frontières*. Plon, 2010. x

[21] J. Duportail. Femen : des seins nus pour quel message ? *lefigaro.fr*, 2013. http://www.lefigaro.fr/actualite-france/2013/02/26/01016-20130226ARTFIG00668-femen-des-seins-nus-pour-quel-message.php (disponible au 01/09/2013). x

[22] J. Duriez. La lutte contre la solitude «Grande cause nationale». *la-croix.com*, 2010. http://www.la-croix.com/Semaine-en-images/La-lutte-contre-la-solitude-Grande-cause-nationale-_NG_-2010-11-23-559304 (disponible au 01/09/2013). vii

[23] J. Duriez. Après Google, une panne à plusieurs millions de dollars pour Amazon. *clubic.com*, 2013. http://pro.clubic.com/entreprises/amazon/actualite-578414-panne-amazon-millions-dollars-pertes.html (disponible au 01/09/2013). viii

[24] Jeff Erickson and Sariel Har-Peled. Optimally cutting a surface into a disk. *Discrete & Computational Geometry*, 31(1) :37–59, 2004. 62

[25] Pierre Fraigniaud and Cyril Gavoille. Routing in trees. Research Report RR-1252-01, LaBRI, University of Bordeaux 1, 351, cours de la Libération, 33405 Talence Cedex, France, January 2001. xii, 30

[26] Michael L. Fredman and Michael E. Saks. The cell probe complexity of dynamic data structures. In 21^{st} *Annual ACM Symposium on Theory of Computing (STOC)*, pages 345–354. ACM Press, May 1989. 41

[27] A.L. Frémont. Un historien d'extrême droite se suicide à notre-dame. *lefigaro.fr*, 2013. http://www.lefigaro.fr/actualite-france/2013/05/21/01016-20130521ARTFIG00477-un-homme-se-suicide-dans-notre-dame-de-paris.php (disponible au 01/09/2013). x

[28] F. Fukuyama and A. Bloom. *The end of history ?* National Affairs, Incorporated, 1989. ix

[29] M. Furer and B. Raghavachari. Approximating the minimum-degree steiner tree to within one of optimal. *Journal of Algorithms*, 17(3) :409–423, 1994. 61

[30] Cyril Gavoille and Arnaud Labourel. Shorter implicit representation for planar graphs and bounded treewidth graphs. In Lars Arge and Emo Welzl, editors, 15^{th} *Annual European Symposium on Algorithms (ESA)*, volume 4698 of Lecture Notes in Computer Science, pages 582–593. Springer, October 2007. xi

[31] Cyril Gavoille, David Peleg, Stéphane Pérennès, and Ran Raz. Distance labeling in graphs. Research Report RR-1239-00, LaBRI, University of Bordeaux 1, 351, cours de la Libération, 33405 Talence Cedex, France, April 2000. xi

[32] Yijie Han. Deterministic sorting in o (n log log n) time and linear space. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 602–608. ACM, 2002. 21

[33] S. Hessel. *Indignez-vous !* Indigène éditions, 2010. x

[34] F. Hollande, J.M. Ayrault, C. Taubira, M. Touraine, and D. Bertinotti. Loi 2013-404 du 17 mai 2013 ouvrant le mariage aux couples de personnes de même sexe. 2013. http://www.legifrance.gouv.fr/affichTexte.do?cidTexte=JORFTEXT000027414540 (disponible au 01/09/2013). x

[35] M. Huret and Milcent B. Faut-il interdire la fessée ? *lexpress.fr*, 2001. http://www.lexpress.fr/actualite/societe/famille/faut-il-interdire-la-fessee_491939.html (disponible au 01/09/2013). x

[36] A. Huxley. *Brave new world*. Chatto and Windus, 1932. ix

[37] Sampath Kannan, Moni Naor, and Steven Rudich. Implicit representation of graphs. In 20^{th} *Annual ACM Symposium on Theory of Computing (STOC)*, pages 334–343. ACM Press, May 1988. xi

[38] Kean and Thomas. *The 9/11 commission report : Final report of the national commission on terrorist attacks upon the United States*. 2011. 1-46. viii

[39] J. Lennon. Imagine, 1971. ix

[40] E. Luttwak. *Turbo-Capitalism : Winners And Losers In The Global Economy*. HarperCollins, 1999. xi

[41] D. Mark. Return of a missing gold ring solves wwii mystery. *ABC news*, 2013. http://www.abc.net.au/news/2013-08-20/return-of-a-missing-gold-ring-solves-wwii-mystery/4900666 (disponible au 01/09/2013). x

[42] Abraham Harold Maslow. A theory of human motivation. *Psychological review*, 50(4) :370, 1943. viii

[43] M. McLuhan. *The Medium is the mMssage*. Bantam books, 1967. ix

[44] Ph. Muray. *L'empire du bien*. Les Belles Lettres, 1991. ix

[45] Ph. Muray. *Festivus, festivus : Conversations avec Elisabeth Lévy*. Fayard, 2005. ix

[46] G. Orwell. *Nineteen Eighty-four*. Secker and Warburg, 1949. x

[47] Kenta Ozeki. Spanning trees with bounded maximum degrees of graphs on surfaces. *SIAM Journal on Discrete Mathematics*, 27(1) :422–435, 2013. 61

[48] David Peleg. Proximity-preserving labeling schemes and their applications. In Peter Widmayer, Gabriele Neyer, and Stephan Eidenbenz, editors, 25^{th} *International Workshop on Graph-Theoretic Concepts in Computer Science (WG)*, volume 1665 of Lecture Notes in Computer Science, pages 30–41. Springer, June 1999. xi

[49] Jean-Pierre Petit. *Le topologicon*. Belin, 1985. xiii

[50] Mihai Pătraşcu and Mikkel Thorup. Time-space trade-offs for predecessor search. In 38^{th} *Annual ACM Symposium on Theory of Computing (STOC)*, pages 232–240. ACM Press, May 2006. 21, 66

[51] Mihai Pătraşcu and Mikkel Thorup. Planning for fast connectivity updates. In 48^{th} *Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 263–271. IEEE Computer Society Press, October 2007. xiii, 104

[52] Neil Robertson and Paul D. Seymour. Graph minors. XVI. Excluding a non-planar graph. *Journal of Combinatorial Theory, Series B*, 89(1) :43–76, 2003. 69, 92, 97

[53] W. Schnyder. Embedding planar graphs on the grid. In *Proceedings of the first annual ACM-SIAM symposium on Discrete algorithms*, pages 138–148. Society for Industrial and Applied Mathematics, 1990. 2

[54] S. Spielberg. Minority report, 2002. ix

[55] Jeremy P. Spinrad. *Efficient Graph Representations*, volume 19 of *Fields Institute Monographs*. American Mathematical Society, 2003. i, xi

[56] W. Strothmann. *Bounded degree spanning trees*. PhD thesis, Universitat-Gesamthochschule Paderborn, 1997. 58

[57] Mikkel Thorup and Uri Zwick. Compact routing schemes. In 13^{th} *Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, pages 1–10. ACM Press, July 2001. xii, 30, 31, 71

[58] M. Turchi. Politique, industrie, médias : les méthodes de la dynastie dassault. *mediapart.fr*, 2009. http://www.mediapart.fr/journal/france/011009/politique-industrie-medias-les-methodes-de-la-dynastie-dassault (disponible pour les abonnés au 01/09/2013). ix

[59] H. Verneuil. Mille milliards de dollars, 1982. ix

[60] P. Virilio. *Vitesse et Politique : essai de dromologie*. Galilée, 1977. xi

[61] S. Watson. Angelina jolie's prophylactic mastectomy a difficult decision. *Harvard Health Publications*, 2013. http://www.health.harvard.edu/blog/angelina-jolies-prophylactic-mastectomy-a-difficult-decision-201305156255 (disponible au 01/09/2013). ix

[62] Dan E Willard. Log-logarithmic worst-case range queries are possible in space θ (n). *Information Processing Letters*, 17(2) :81–84, 1983. 21

Annexe - Digression

Connexion

« Suis-je connecté ? » Combien de fois par jour cette question m'effleure-t-elle ? Mon téléphone portable est-il relié au réseau ? Et au Wi-Fi ? Suis-je joignable en vacances ? Puis-je changer d'opérateur téléphonique ou d'hébergeur de courrier électronique sans disparaître socialement ? Puis-je rentrer à Bordeaux depuis Clermont en train ?

Cette angoisse de la connexion rompue peut faire ressembler le plus aguerri des hommes d'affaire du XXIème siècle à une adolescente attendant depuis trois jours l'appel du jeune *Kévin*. On le voit faire les cent-pas, dans un aéroport ou un hôtel, en costume impeccable, mais le nez collé compulsivement à son *téléphone intelligent* de dernière génération ou à sa tablette. Ce pionnier du néo-nomadisme, figure-clef de l'élite mondialisée, n'a plus de patrie que le réseau dématérialisé de ses contacts électroniques. Sans cet *e*-cordon ombilical symbolique, il n'est nulle part et il est seul. (Nottons que la lutte contre la solitude constitue la cause nationale pour l'année 2011 du gouvernement français élu [22]).

La génération dont nous sommes a eu la chance de connaître l'avant et l'après, aussi bien en ce qui concerne l'Internet qu'au sujet du téléphone portable. Nous savons donc que nous sommes capable de survivre déconnectés, nous pratiquons même la déconnexion dès que l'occasion le permet, mais ce ne sera sans doute plus le cas des générations à venir. C'est d'ailleurs au tour de l'adolescente de ressembler à notre homme d'affaire et de se sentir nue sans le talisman électronique qui est greffé à sa main. On retrouvera facilement sur un site d'hébergement de vidéos, la crise de nerfs d'un jeune allemand, violentant le PC qui lui refuse l'accès à un jeu en ligne massivement-multijoueur [2].

Suis-je connecté ? (*bis*) Cette question est centrale, certes, mais quelle en est la réponse ? Si vous lisez en ce moment une version électronique de ce document, vous pouvez peut-être apercevoir dans un coin de votre écran une petite icône qui indique, à l'aide d'arcs de cercles concentriques, si oui ou non, et dans quelle mesure, vous êtes connecté à votre router Wi-Fi. Votre téléphone portable arborera lui aussi, quelques barres verticales témoignant de son état et de son niveau de connexion. Quant à votre *router*, il saura vous indiquer s'il a *accroché* le réseau d'un simple clin de LED.

La patience, lorsqu'il s'agit de connaître son état de connexion, n'est pas de mise. L'attente est hautement anxiogène. La réponse à la question qui taraude le candidat à la connexion doit donc être disponible sans délai. Elle est devenue un besoin quasi-vital pour l'homme moderne et s'est progressivement glissé sous sa *pyramide de Maslow* [42].

La structure qui m'emploie est-elle connectée ? *S*i, *a fortiori*, nous faisons parti de l'équipe d'administration-réseau d'un grand moteur de recherche américain ou d'un site de vente en ligne pour lesquels l'impact de chaque minute de panne s'évalue en centaines de milliers de dollars de manque à gagner [23], ou si nous avons en charge la sécurité civile ou militaire d'une nation entière, alors dans ces cas là, impatience et exigence sont des vertus. Le temps perdu inutilement lors d'une situation de crise donne lieu à des chasses aux responsables ou à des restructurations légitimes comparables à celle de la NORAD un lendemain de 11 Septembre 2001 [38]. Se maintenir connecté, ou maintenir son client client connecté, est, en conclusion, un défi incontournable pour tout professionnel consciencieux.

Distribution des données

L'humain, le robot et le *big data*. A l'heure des Systèmes d'information, des *big data*, des Google et Facebook, de la Nouvelle Université de

Bordeaux ou de l'Union Européenne ; à l'heure de la gouvernance globale, de Jacques Attali, de Bernard-Henri Levy, de la guerre globale contre le terrorisme menée par *l'Empire du Bien* [44], de la surveillance globale, et du Patriot Act, information et pouvoir se fédèrent et s'entrelacent chaque jour un peu plus [58]. Les monopoles finissent par s'imposer à coups de rachats et de *mille milliards de dollars* [59], les états-providences mamouths faussement libéraux et hyper-dirigistes sont désormais capables de tout nous donner ou de tout nous prendre, la liberté individuelle disparait derrière son ersatz électoral, festif et libidineux [45], derrière le *tittytainement* cher à *Zig* Brzezinski. Le résistant, le mal-pensant et l'*ennemi de la liberté* sont traqués. Nos lois *encadrent* si bien la *liberté d'expression* que les condamnations politiques, en Occident-Démocratique, deviennent légions.

Et aux grands maux les grands remèdes ! Le *Bien* doit s'armer. C'est tout de même de notre sécurité dont il s'agit. Alors on emmagasine les données personnelles, on les recoupe, on les torture et on les fait parler ; le pouvoir centralisé me connait mieux que je ne me connais moi-même, et cela bien sûr dans mon plus grand intérêt. Le jeu en vaut en effet la chandelle : les groupes terroristes sont dissouts et les fanatiques emprisonnés avant même leur passage à l'acte, *le meilleur des mondes* [36] des cyber-Torquemada et de *Minority Report* [54] est enfin une réalité ; mon cancer est diagnostiqué vingt ans avant sa manifestation grâce à une formule statistique [61] ; mon fils est orienté vers la filière d'insertion professionnelle qui convient le mieux à son profil. Sécurité contre liberté. Que suis-je prêt à abandonner pour éviter la mort violente, la maladie ou la misère ? Suis-je prêt à donner à un pouvoir géant, à ses mathématiciens, à ses joueurs de poker, à ses algorithmes, la vision de toutes les cartes de mon jeu, et donc lui offrir l'immortalité ? C'est de la *fin de l'Histoire* [28] des hommes et donc du début de la dictature des *robots* [7] dont il s'agit.

Données et sphères concentriques. Nous vivons une époque de l'ouverture. Il faut être ouvert d'esprit, il faut faire tomber les mûrs et les barrières. *Open-mind*, *open-space*, rien ne doit s'opposer au *libre-échangisme* des idées, de la marchandise et des humains. La grande orgie multicolore et globale du *village monde* [43] n'admet ni entrave ni coup de frein. Tout le vocabulaire *johnlennonisant* [39] qui sert à décrire les notions d'ouverture est porteur de

charges émotionnelles positives très fortes. Au contraire, les mots permettant de faire référence à des concepts cloisonnés sont drapés des oripeaux de la haine de l'autre, du fanatisme, du sexisme, du racisme, de l'autarcie et des tous les épouvantails *godwiniens* étapes du *reductio ad Hitlerum* et de la disqualification mécanique de tout *criminel de la pensée* et futur *non-être* [46].

Intime, privée ou publique, l'information dont je dispose peut être mentalement disposée autour de moi selon des sphères concentriques. Ma boîte crânienne est l'incarnation symbolique d'une barrière séparant deux d'entre ces sphères. Mais voilà, à une époque où les frontières sont dépassés [20], à l'époque de la télé-réalité, à celle où l'on légifère sur la célébration légale de l'amour [34], sur l'éducation des enfants, à une époque où on interdit la fessée [35], à une époque de puritanisme impudique où l'on doit hurler en montrant ses seins [21] ou se suicider dans une cathédrale [27] pour être écouté, l'intime, le privé et la *chose publique* se mélangent. L'exhibitionnisme, symptôme s'il en est de la modernité narcissique des réseaux sociaux et de la quête de *quart d'heure de célébrité* [1], fait tomber tous les murs. La victime enthousiaste de ce casse du nouveau siècle, atteinte du syndrome de Stockholm, se livre d'elle même aux trompettes prédatrices de la renommée et du *buzz*. Elle offre ce qu'elle a de plus intime en échange d'un succédané d'existence, comme un État de la vieille Europe livrerait son or à la Réserve Fédérale Américaine contre une palette de billets verts fiduciaires frappés du *Grand Sceau*, de l'*In God we trust* et empreints de leur dose de rêve hollywoodien [41] ; ou encore comme la recherche publique française pourrait servir de département R&D, aux frais du contribuable, à quelques entreprises privées du CAC40 qui délocalisent malgré tout leur production en Asie du Sud-Est alors que le chômage et la pauvreté battent leur plein dans l'hexagone [11]. Cette dérégulation enthousiaste et naïve de ce qui a trait à la vie privée et à la vie publique, à toutes les échelles, ouvre la porte à toutes les inquisitions et tous les pillages intellectuels. La question : « Que craindre si vous n'avez rien à cacher ? », la sacralisation de la *transparence*, l'impératif d'hurler avec les loups et de s'émouvoir avec les brebis (*Indignez-vous !* [33] C'est un ordre.), ou en somme l'allégeance publique obligatoire à l'esprit du temps, tous constituent autant de *shahâda* de la religion du moderne.

La transparence comme faiblesse. Une fois les portes de mon esprit et de mon intimité ouvertes, il n'y a plus qu'à pomper. La pieuvre n'a guère plus à faire que d'immiscer son tentacule, de se servir et de stocker . Nous parlons de pieuvre car le pouvoir repose sur la centralisation de l'information. L'humanité est réduite à l'ensemble des cellules sensorielles d'un gigantesque *système d'information* qui la connaît en masse et en détail, la gouverne, l'asservi et n'oublie jamais.
Le *S.I.* contrôle alors le passé qu'il peut écrire, l'avenir qu'il peut prédire et façonner, et donc par voie de conséquence, le présent de ce monde Orwellien. On nous opposera la notion de *gouvernance*. Comment les humains peuvent-ils être asservi par une machine elle-même gouvernée par des humains ? Nous répondrons en parlant de *la vitesse* [60]. Aux prises de décision d'une machine, de turbo-gouvernement ou de turbo-finance [40], ayant intégré l'ensemble des données sociales du réseau-humanité ne pourra être opposé aucun argument rationnel humain en temps utile. L'homme d'intuition et l'homme de foi pourront éventuellement se dresser contre la froideur inhumaine, mais les machines et eux ne parlent malheureusement pas le même langage.

Collaborer ? Fuir ? Détruire ? Que faire si l'on pense que l'humanité ne doit pas abandonner son destin aux mains de sa création ? Doit-on se refuser à l'inévitable *progrès* et au dogmatisme des meilleurs lendemains ? Doit-on vivre en Amish ? Doit-on attendre comme le messie un nouvel Oppenheimer qui inventera une machine capable de détruire toutes les autres [7] ? Peut-être. Nous prendrons ici, car nous sommes né moderne, malgré tout, le parti optimiste de ne pas nous opposer à l'aventure technologique et à l'esprit de la modernité en bloc ; le parti de supposer qu'il existe un moyen pour l'homme de dompter les Titans à qui il a donné le jour ; le parti de se rallier aux hommes de bonne volonté. Pour que la post-humanité des mutants et des transhumains augmentés soit une humanité tout de même, et que l'éthique ait encore un sens dans cet avenir imprévisible, où l'initiative humaine n'est plus guidé par rien mais aspirée par le néant.

Il nous faudra donc respirer l'air du temps, vivre au sein des machines et, si possible, en symbiose avec elles. Dont acte. L'un des défis qui se présentent à notre humanité est donc la nécessité d'une réaction à l'égard de ce qui tend à centraliser, globaliser, mondialiser, toujours. Pour lutter contre la centra-

lisation, on songe naturellement à *distribuer*. Le *Robin des bois* moderne ne doit plus se limiter à la redistribution des richesses matérielles puisque c'est à un nouveau mode de pillage que nous sommes aujourd'hui en proie.

En conclusion. L'algorithmique et surtout les bases de données distribués ont, potentiellement, une dimension émancipatrice, et elles constituent potentiellement une des cordes à l'arc de ceux qui cherchent un bouclier à la turbo-dictature de la froide rationalité des machines. Nous n'avons surtout pas eu ici la prétention de servir la cause de l'Humanité, et encore moins de manière décisive, mais nous aurons eu au moins l'apaisement de travailler dans le cadre éthique d'un moindre mal.

Annexe - Fougasse et topologie

Nous reproduisons ici, avec l'aimable autorisation de l'auteur, quelques pages extraites de l'une des bandes dessinées [49] de Jean-Pierre Petit. Elles expliquent brillamment ce qu'est la *caractéristique d'Euler-Poincaré* et font le lien avec le *genre* via l'exemple de la fougasse à n trous.

Effectivement, sa mésaventure l'a confronté à une situation qu'il ne pouvait plus assumer.

Eh oui, une remise en question brutale de son moi profond !

Bref, la seule solution est de trouver où est passé ce fichu pôle Sud

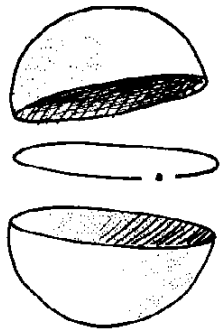
DÉCOMPOSITION CELLULAIRE

Tout objet géométrique sera décomposé en éléments, en cellules **CONTRACTILES** de toutes dimensions : POINTS, SEGMENTS, SURFACES, VOLUMES, ETC...

Et le POINT est de quelle dimension ?

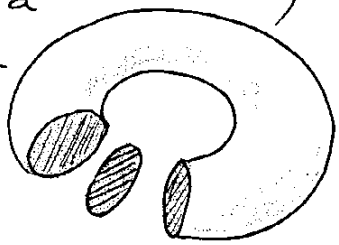
Par extension, nous dirons que le POINT est de dimension **ZÉRO**.

D'autre part, pour décomposer un cercle, il suffit que je le considère comme un segment fermé sur lui-même en un **POINT**. Si je retire ce point, il reste donc le segment.

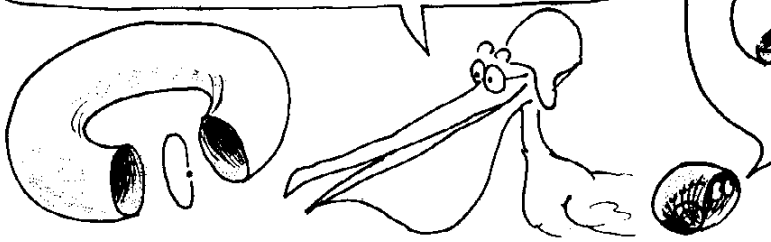


Une "SPHÈRE SURFACE" **S2** peut être décomposée en deux calottes et un segment fermé sur un point.

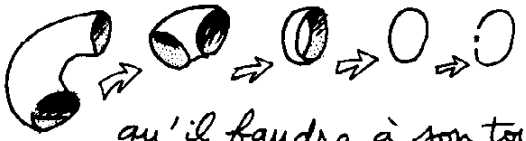
Un "TORE-VOLUME" ? Voyons, je n'ai qu'à le découper à l'aide d'un disque



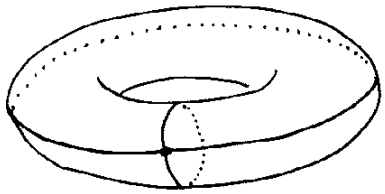
Et pour le "TORE-SURFACE" ? Voyons... je coupe par un cercle lui-même coupé en un point



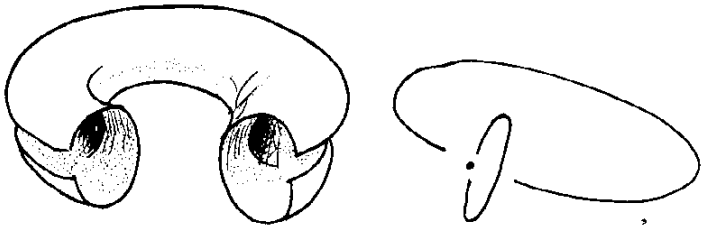
Le Tore ainsi coupé va se contracter selon un cercle :

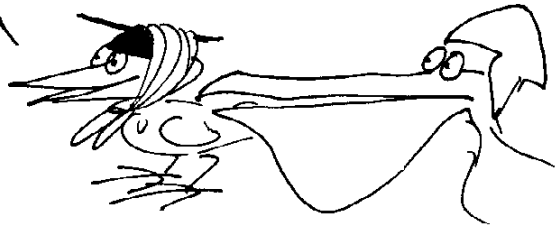


qu'il faudra à son tour décomposer selon un segment et un point.

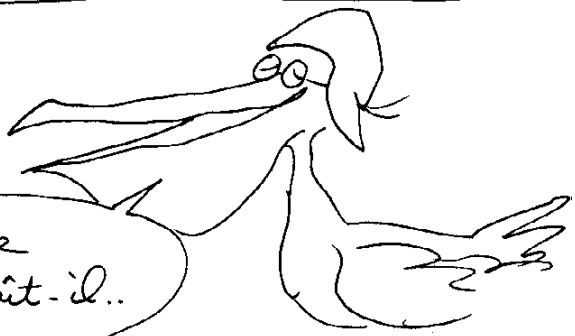


Voici une autre solution avec un point, deux segments et une face unique, où tous les éléments sont d'emblée contractiles





Bon, et que va-t-on faire de tout cela ?



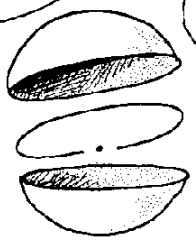
Comprendre le monde, paraît-il..

LA CARACTÉRISTIQUE D'EULER-POINCARÉ

Un objet étant ainsi décomposé, nous allons fabriquer un nombre χ qui sera égal au nombre de points, moins le nombre de segments, plus le nombre d'éléments de surface contractiles, moins le nombre de volumes contractiles (*), et on appellera ce nombre χ la CARACTÉRISTIQUE D'EULER-POINCARÉ.

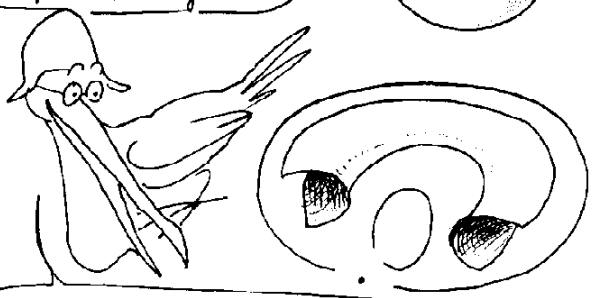
Ainsi pour le cercle $\chi = 1 - 1 = 0$

1 point, 1 segment



Pour la SPHÈRE-SURFACE $\chi = 1 - 1 + 2 = 2$

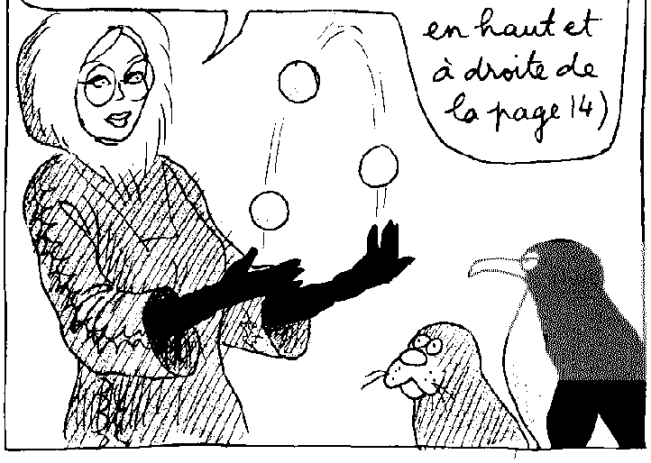
un point, un segment, deux calottes



Pour le Tore-surface, voyons... un point, deux segments, un élément de surface $\chi = 1 - 2 + 1 = 0$

C'est-à-dire 1 point, 2 segments et 1 élément de surface contractile

La caractéristique de la SPHÈRE-VOLUME est évidemment -1, alors que celle du TORE-VOLUME est $1 - 1 = 0$ (Voir le dessin en haut et à droite de la page 14)



(*) Ce qui s'étend immédiatement à un nombre de dimensions supérieur à trois (c'est une somme alternée).

Et maintenant, écoutez bien : cette caractéristique χ est **INDÉPENDANTE DU MODE DE DÉCOMPOSITION** (en cellules contractiles) !!

Par exemple, cette courbe fermée a été coupée en 8 segments, réunis par 8 points, et sa caractéristique est toujours nulle.

effectivement.

Voyons cette décomposition de la sphère : 4 sommets, 6 segments, 4 faces. Je retrouve $\chi = 4 - 6 + 4 = 2$.

Et là, 8 sommets, 12 segments, 6 faces $\chi = 8 - 12 + 6 = 2$

Tu peux essayer tout ce que tu voudras, tu retomberas toujours sur $\chi = 2$

Boufre de boufre

Étonnant, non ?

Voici un Théorème utile : Si un objet est la réunion de deux objets sa caractéristique est la somme de celles des 2 objets le composant.

La Direction

* Sorte de pain du midi de la France

Rien à voir ! la FOUGASSE-SURFACE ne peut pas se contracter selon un disque à N trous, enfin !

planté...

→ On peut passer de la SPHÈRE-SURFACE (caractéristique 2) au TORE-SURFACE (caractéristique zéro) en ajoutant une anse. Donc l'ajout d'une anse diminue la caractéristique d'une surface de 2 unités.

Donc la caractéristique de la FOUGASSE-SURFACE est égale à 2 moins deux fois le nombre de trous !

La SURFACE d'un morceau de gruyère à N trous est constituée de N sphères-surface plus la sphère extérieure. Donc sa caractéristique est

$$\chi = 2(1+N)$$

Alors que pour construire le GRUYÈRE-VOLUME, on part d'une sphère pleine ($\chi = -1$) et on enlève N ensembles SPHÈRE-VOLUME + SPHÈRE-SURFACE ($\chi = 2-1 = 1$). La caractéristique du GRUYÈRE-VOLUME est donc égale à $-(1+N)$.

Croyez-vous qu'avec de telles âneries nous arriverons à guérir ce pauvre Amundsen de sa géonévrose ?!!