Super-Arbre d'Accord Maximum

Vincent Berry

Super-arbre d'accord maximum*
RR-LIRMM 03040

Vincent Berry et François Nicolas

2003

Toutes choses sont muables et proches de l'incertain
Pierre Michon

Résumé

Given a collection of leaf-labeled trees, it is a well-known problem to infer a single *consensus* tree that represents them as best as possible. The problem finds direct application in bioinformatics and more particularly in phylogeny construction. Recent research in consensus of phylogenies concerns *supertree* inference, which arises when the input trees have different leaf sets [44, 45, 8, 49]. Input trees are often gene trees subject to different histories or duplication/loss events resulting in conflicts on the position of common leaves between different input trees.

We show here that the maximum agreement subtree (MAST) problem [22, 48, 19, 2, 10, 34, 35], designed for the case of input trees on equal leaf sets, can be extended for building supertrees. This problem concerns the inference of a tree with a largest set of leaves on which all input trees agree, i.e., all source trees are identical when restricted to this set of leaves. In contrast, in case of conflicting input trees, most current supertree methods output a tree which conflicts with some input tree(s)[8, 39], or output a tree not conflicting with any input tree but doing so, the tree contains many multifurcations, ie poor information [27, 7]. We show that even in case of conflicting input trees the maximum agreement supertree (*SuperMast*) is likely to contain a non-trivial number of input leaves.

We proove that the maximum agreement supertree problem is NP-hard, even if the input trees are rooted triples. This contrasts with the MAST problem which is polynomial for bounded degree trees or trees with a bounded number of leaves. We also prove that the maximum agreement supertree problem is $W[2]$-hard for the parameter p, where p is the minimum number of leaves to remove from source trees to obtain an agreement. In comparison, the MAST problem is FPT in p.

We give a $O(n + N)$ time algorithm for the special case of two input (rooted or unrooted) trees of unbounded degree, where N is the time bound for solving the MAST problem (currently $N = O(n^{1.5})$ [40, 34, 35]).

*avec le soutien de l'*Action Incitative Informatique-Mathématique-Physique en Biologie Moléculaire* [ACI IMP-Bio].

Table des matières

1 Introduction

1.1 La méthode *MAST* dans le contexte général

Les arbres dont les noeuds portent des étiquettes sont utilisés dans de nombreux domaines scientifiques dont la chimie, la linguistique, la vision par ordinateur, la reconnaissance de motifs ou encore les bases de données textuelles structurées (des références pour ces domaines peuvent être trouvées dans [35]) et surtout l'inférence phylogénétique qui nous intéresse plus particulièrement ici [48, 19, 2].

Une méthode très répandue pour mesurer la similitude de plusieurs arbres étiquetés par *un même ensemble d'étiquettes* est la méthode du sous-arbre d'accord maximum (*MAST*), qui produit le plus grand sous-arbre (en nombre d'étiquettes) homéomorphiquement inclus dans tous les arbres donnés en entrée.

1.2 Situation de *MAST* dans le contexte phylogénétique

Dans le contexte de la reconstruction de *phylogénies* (aussi appelées *arbres d'évolution*), les arbres considérés sont étiquetés uniquement aux feuilles, chacune ayant une étiquette distincte des autres. Les étiquettes correspondent le plus souvent à des espèces, mais aussi parfois à des gènes. La méthode *MAST* a été introduite dans ce domaine par Finden et Gordon [22], qui proposaient un algorithme heuristique. Il a fallu attendre le travail de [48] pour obtenir le premier algorithme exact en temps polynomial, s'appliquant au cas de deux arbres sources. De nombreux travaux se sont ensuite succédés, aboutissant

- dans le cas de *deux* arbres *enracinés* à un algorithme en $O(\sqrt{d}n \log n)$ [40] et un algorithme en $O(\sqrt{d}n \log^2 \frac{n}{d})$ [35], où d est le degré des arbres sources[1].
- dans le cas de *deux* arbres *non-enracinés* à un algorithme en $O(n^{1.5})$ [34].

Le problème *MAST* a été montré NP-difficile dès que l'on dispose de trois arbres sources [2], bien que résolvable en temps polynomial si le degré d'un des arbres sources est borné [2, 19, 10].

1.3 Inférence de super-arbres en reconstruction phylogénétique

Dans le domaine de la reconstruction phylogénétique, de nombreuses raisons, dont les contraintes de collecte de données homogènes pour un grand nombre d'espèces différentes, ont récemment porté l'attention sur la construction de phylogénies sur la base d'un ensemble d'arbres de donnés, appelés *arbres sources*, définis sur des ensembles de feuilles partiellement disjoints. L'arbre construit est appelé *super-arbre*, au sens où il s'agit d'une sorte de méta-analyse des données initiales : cet arbre est produit à partir d' arbres eux-mêmes issus de premières analyses [44]. Cette approche a pour avantage de pouvoir intégrer (indirectement) des données de différents types (morphologique, moléculaire, etc) et de palier le manque de données disponibles pour certaines espèces ; elle semble être la technique clef pour obtenir l'*arbre de la vie* mettant en relation l'ensemble des espèces vivantes [44, 51, 8].

Plusieurs méthodes existent pour inférer un super-arbre depuis un ensemble d'arbres sources [27, 5, 42, 45, 39, 14, 15, 49]. Quand les arbres sources sont compatibles, les méthodes de super-arbres produisent un résultat similaire et satisfaisant [51, 8]. Malheureusement, dans la plupart des cas, les arbres sources sont en conflit [51, 8] et ce d'autant plus que le nombre d'arbres sources et de feuilles partagées augmente [44]. En cas de conflit entre les arbres sources, deux alternatives sont possibles :

[1] [35] remarque que leur résultat améliore celui de [40] pour $d \geq n/2^{O(\sqrt{\log n})}$, et que pour tout d on a $\sqrt{d}n \log^2 \frac{n}{d} = O(n^{1.5})$.

1. une approche de type *consensus* qui enlève les conflits en choisissant par exemple de contracter les arêtes des arbres sources impliquées dans des conflits [27] ou d'enlever les feuilles responsables des conflits ; citons aussi le travail de [49].
2. une approche de type *optimisation* qui a tendance à résoudre d'une façon ou d'une autre les conflits en choisissant l'alternative qui maximise un certain critère de reconstruction [5, 42, 45, 39, 14]. Un certain nombre de travaux récents ont noté qu'une telle résolution des conflits n'est pas toujours justifiée ou désirable, et que le comportement de ces méthodes n'est pas encore bien compris [41, 43, 7, 51, 39].

Comme le font remarquer [27, 51, 8], en enlevant quelques feuilles dont le placement est problématique, on peut souvent éviter la plupart des multifourches proposées par les méthodes de super-arbres (dans les deux approches ci-dessus). Techniquement, l'arbre ainsi obtenu peut toujours être considéré comme un *super-arbre*, car il inclut plus de feuilles que n'importe quel arbre source. Toutefois, à notre connaissance, bien que cette approche a été suggérée plusieurs fois, aucune méthode de ce type n'a encore été proposée (dans le contexte de la reconstruction phylogénétique). L'objet de ce papier est de proposer une telle méthode.

1.4 La méthode *SuperMAST* et son intérêt

Nous proposons ici une méthode d'inférence de super-arbre apparentée à l'approche consensus (première catégorie de méthodes décrite ci-dessus). Dans cette catégorie, la méthode la plus souvent utilisée est celle de strict consensus qui propose un super-arbre contenant "*l'information sur laquelle les arbres sources sont en complet accord*" [27]. La contrainte d'obtenir un accord complet est très forte et la méthode est souvent critiquée pour le peu de structure présente dans le super-arbre inféré en pratique [51, 8].

Nous proposons d'adapter la méthode *MAST* au contexte des super-arbres, obtenant une méthode que nous appelons *SuperMAST*, pour inférer un super-arbre correspondant au plus grand ensemble de feuilles sur le placement desquelles les arbres sources ne sont pas en désaccord[2]. En retenant *l'information des arbres sources sur laquelle aucun arbre source n'est en désaccord*, cette méthode a toutes les chances de produire un super-arbre contenant plus de structure que celui donné par la méthode de strict consensus,

Comme la méthode de consensus strict, $SuperMAST$ propose un arbre ne contenant aucune information contredite par les données. Cette propriété est fortement désirable car en pratique les arbres sources ont souvent des structures partiellement contradictoires, sans qu'une information extérieure ne permette de remettre en cause les parties de phylogénies impliquées dans une contradiction.

Bien que $SuperMast$ soit une méthode de type consensus, il est aussi possible de l'utiliser dans une approche de type optimisation (deuxième catégorie de méthodes de super-arbres décrite au paragraphe 1.3). On peut en effet utiliser l'arbre qu'elle propose comme un *squelette* (*backbone tree*) sur lequel on peut greffer les feuilles restantes des arbres sources en accord avec un critère d'optimisation choisi (parcimonie, distance, combinatoire, etc).

Outre son intérêt propre pour l'estimation d'une phylogénie sous-jacente à une collection d'arbres sources, la méthode $SuperMAST$ peut s'avérer utile pour la méthode MRP (*Matrix Representation with Parsimony*) dans le cas où les arbres sources ont peu de feuilles en commun. Dans un tel cas, la méthode MRP donne des résultats insatisfaisants (grand nombre d'arbres les plus parcimonieux, très faible résolution de l'arbre final, faible adéquation à l'arbre correct) [9]. Pour améliorer les résultats de

[2] Notons qu'il est possible qu'un tel ensemble de feuilles ne soit pas unique, mais une légère modification des algorithmes permet de les identifier tous.

MRP dans un tel cas, [9] recommande d'insérer un arbre *graine* contenant des feuilles de nombreux arbres sources. Nous montrons dans la section 3.3 que le super-arbre d'accord maximum contient fort probablement des feuilles de tous les arbres sources, et constitue donc un arbre graine de choix.

Enfin, comme son homologue *MAST*, la méthode MAT trouve aussi une utilisation dans la comparaison d'arbres. Le nombre de feuilles du super-arbre qu'elle produit permet de mesurer le degré d'accord d'arbres dont les ensembles de feuilles sont partiellement disjoints et ainsi d'estimer, dans une certaine mesure, la confiance qu'on peut avoir dans leur super-arbre. Ceci fait echo à [39, 26] qui déplorent que de nombreuses super-arbres soient proposés depuis quelques années sans que l'on dispose de mesure pour évaluer leur adéquation aux données.

La faible complexité de cette méthode pour deux arbres la rend aussi attractive dans le cas d'études de simulations sur les méthodes de super-arbres, où le degré d'accord entre les arbres sources constitue un paramètre d'importance pour interpréter les résultats [9, 15]. En appliquant l'algorithme de la section 6 à tout couple d'arbres sources, on peut rapidement obtenir une mesure du degré d'accord moyen entre arbres sources.

1.5 Résultats

Les résultats présentés dans ce papier sont les suivants :

- une définition du problème de super-arbre d'accord maximum d'un ensemble d'arbres sources aux feuilles partiellement disjointes. Cette définition est une extension naturelle de celle de *MAST*, au sens où l'on retrouve cette dernière définition dans le cas où les arbres sources ont tous les mêmes feuilles ;
- des propositions tendant à montrer que les feuilles spécifiques, i.e., qui n'apparaissent que dans un seul arbre source, sont généralement incluses dans l'arbre résultat, et donc qu'il est vraisemblable que l'arbre produit contienne un nombre non-trivial de feuilles ainsi qu'il contienne des feuilles de tous les arbres sources ;
- une preuve de NP-difficulté pour le problème général du super-arbre d'accord maximal, même dans le cas où les arbres sources sont de degré borné ou ont un nombre borné de feuilles (cas pourtant polynomiaux pour le problème MAST).
- en ce qui concerne la complexité paramétrique pour le problème MAT, nous montrons qu'il appartient à la classe de problèmes $W[2]$-difficiles pour le paramètre p, le nombre minimum de feuilles qu'il faut enlever des arbres sources pour qu'ils soient en accord.
- Dans le cas particulier de *deux* arbres sources (enracinés ou non-enracinés), nous donnons un algorithme de complexité $O(n + N)$, où $O(N)$ est la complexité nécessaire pour résoudre le problème MAST (actuellement $N = min\{O(\sqrt{d}n \log n)\ ,\ O(\sqrt{d}n \log^2 \frac{n}{d})\}$ [40, 35] pour les arbres enracinés et $N = O(n^{1.5})$ pour les arbres non-enracinés [34]).
- Nous montrons que l'algorithme précédent ne peut être étendu à plus de deux arbres sources et nous donnons un deuxième algorithme ne souffrant pas de ce problème. Il s'agit d'un algorithme de programmation dynamique inspiré de l'algorithme de [48] qui permet de calculer le super-arbre d'accord maximum de deux arbres sources enracinés en $O(n^{4.5})$.

Bien que ces travaux n'aient pas de rapport avec ceux présentés ici, signalons que récemment [28] ont étudié une variante de *MAST* proposant un *super-arbre*. Toutefois, le problème qu'ils proposent de résoudre et leur définition de *super-arbre* ne s'adaptent pas à la reconstruction de phylogénies.

2 Préliminaires

2.1 Notion de conflit entre arbres sources

Les méthodes d'inférence de super-arbres font l'hypothèse que les arbres sources sont des estimations partielles d'une phylogénie sous-jacente aux données. Il est possible que ces estimations soient partiellement erronées en raison de données de qualité ou de quantité insuffisante, ou bien encore d'artéfacts de la méthode les produisant. De telles erreurs amènent les arbres sources à se contredire partiellement. On dit alors qu'ils sont *incompatibles, inconsistents,* ou *en conflit*. Toute incompatibilité d'un ensemble d'arbres enracinés se traduit par une incompatibilité de *triplets* [12, 10, 45] ou de façon équivalente par une incompatibilité dans le placement de noeuds représentant les plus petits ancêtres communs de deux feuilles [1, 36] : pour un même ensemble de trois feuilles ou pour deux plus petis ancêtres communs, les arbres sources induisent au moins deux agencements différents. Par exemple,

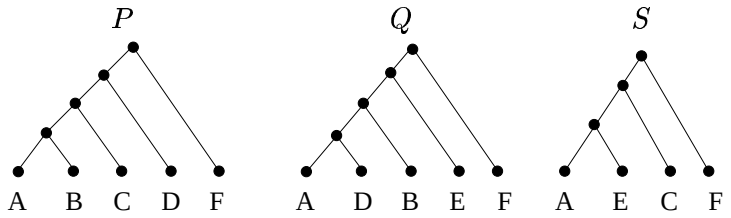

FIG. 1 – Exemple d'arbres sources incompatibles

sur la figure 1, les arbres P et Q sont en conflit sur le triplet $\{A, B, D\}$: leurs restrictions respectives à cet ensemble de trois feuilles, `((A,B),D)` et `((A,D),B)` (en utilisant la notation parenthésée pour décrire les arbres), sont des arbres différents. Certains conflits peuvent être moins directement visibles. Par exemple l'ensemble des trois arbres sources de la figure 1 est en conflit sur le triplet $\{A, E, C\}$, bien qu'un seul de ces trois arbres possède les trois feuilles à la fois. Ce conflit indirect est expliqué par le fait que dans l'arbre S le plus petit ancêtre commun de A et E (noté $lca(A, E)$) est un descendant du plus petit ancêtre commun de A et C (noté $lca(A, C)$). Or d'après Q, $lca(A, E)$ est un ancêtre de $lca(A, D)$. Donc tout super-arbre qui respecte les informations de Q et S doit induire, par transitivité, que $lca(A, D)$ est descendant de $lca(A, C)$, ce qui induit une contradiction avec l'arbre P.

En présence de conflits, les méthodes existantes de super-arbres choisissent soit la résolution proposée par l'un des arbres sources (contredisant d'autres arbres sources), soit proposent une multifourche traduisant une irrésolution du positionnement des sous-arbres impliqués dans les conflits. Par exemple, la méthode de strict consensus [27] propose le super-arbre `((A,B,C,D,E),F)` pour l'ensemble des arbres sources de la figure 1. Malheureusement la présence de plusieurs conflits, impliquant parfois de mêmes feuilles, amène souvent à des irrésolutions importantes. Nous explorons dans ce travail une troisième alternative, consistant à supprimer certaines feuilles de façons à éviter tout conflit. Bien-sûr, nous voulons inférer un super-arbre contenant autant de feuilles que possible. Plus précisément, en étendant la méthode du *MAST* au contexte des super-arbres, nous chercherons le plus grand sous-ensemble de feuilles des arbres sources qui peut être conservé en évitant tout conflit.

2.2 Définitions et notations

La définition des arbres que nous utiliserons est celle correspondant à un arbre d'évolution (*evolutionary tree*), aussi appelé *phylogénie* :

DÉFINITION 2.1 (ARBRE)
Un *arbre* (phylogénétique) est un graphe connexe acyclique $T = (V(T), E(T))$, tel qu'il existe une bijection $l_T : F(T) \mapsto S$ de l'ensemble $F(T)$ des feuilles de T vers un ensemble d'étiquettes S. Par abus de notation, $F(T)$ désignera les feuilles de T et en même temps les étiquettes associées aux feuilles de T.

Un arbre *non-enraciné* ne contient aucun sommet de degré deux et aucune orientation particulière.

Un arbre est *enraciné* s'il possède un noeud particulier appelé *racine* et une orientation de toutes ses arêtes de la racine vers ses feuilles. Seule le noeud racine est autorisé à avoir un degré égal à deux. La racine peut toutefois avoir un degré plus important.

La *taille* d'un arbre T est définie comme le nombre de feuilles qu'il possède : $|T| \triangleq |F(T)|$.

DÉFINITION 2.2 (RESTRICTION D'UN ARBRE)
Soit T un arbre, et soit S un ensemble d'étiquettes, on note $T|S$ la restriction de T à S, ie le plus petit sous-graphe connexe de T reliant les feuilles de $F(T)$ étiquetées dans S et dont les sommets de degré deux (autres que la racine éventuelle) sont supprimés. Notons que S pourra ou non être inclus dans $F(T)$.

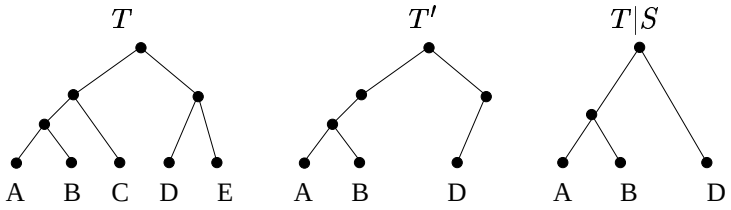

FIG. 2 – Exemple de restriction d'un arbre T à un ensemble de feuilles $S = \{A, B, D, F\}$. T' est le plus petit sous-graphe de T reliant les feuilles de T étiquetées dans S, $T|S$ est obtenu depuis T' en supprimant les sommets de degré 2.

Dans la suite de ce document, les arbres considérés seront le plus souvent enracinés, i.e., possédant un sommet racine et une orientation implicite des arêtes de la racine vers les feuilles. Ainsi, tout sommet, ou *noeud*, est descendant de la racine.

DÉFINITION 2.3 (DESCENDANT/ASCENDANT)
Soit T un arbre enraciné, et soit $v, v' \in V(T)$, on note $v <_T v'$, resp. $v \leq_T v'$, si v' est sur le chemin de la racine de T à v' exclus, resp. inclus.

DÉFINITION 2.4 (LEAST COMMON ANCESTER (LCA))
Soit T un arbre enraciné et $V \subseteq V(T)$, alors $lca_T(V)$ est le noeud $l \in V(T)$ tel que
(1) $\forall v \in V$, on a $v \leq_T l$
(2) tout noeud $l' \in V(T), l' \neq l$ qui satisfait (1) est tel que $l <_T l'$.

DÉFINITION 2.5 (INCLUSION HOMÉOMORPHIQUE D'ARBRES)
Soient T_1 et T_2 deux arbres, on dit que T_1 est *homéomorphiquement inclus* dans T_2, que l'on note

$$T_1 \subseteq_h T_2$$

ssi $T_1 = T_2|F(T_1)$, ie s'il existe un isomorphisme de graphe $\phi : V(T_1) \mapsto V(T_2|F(T_1))$ préservant les étiquettes des feuilles : $\forall f \in F(T),\ l_{T_1}(f) = l_{T_2}(\phi(f))$.

On note $\theta : V(T_1) \mapsto V(T_2)$ la fonction d'homéomorphisme entre T_1 et T_2. Les feuilles étant identifiées à leur étiquette, les feuilles portant la même étiquette entre T_1 et T_2 sont désignées sous un même nom f et nous aurons donc $\theta(f) = f$.

Dans le cas d'arbres enracinés, on ajoute la contrainte suivante [48] à la définition de $T_1 \subseteq_h T_2$: soit $v, v' \in V(T_1)$, alors

$$\theta(lca_{T_1}(v, v')) \quad = \quad lca_{T_2}(\theta(v), \theta(v')) \ . \tag{1}$$

Plus précisément dans le cas de deux feuilles $f, f' \in F(T_1) \cap F(T_2)$ on a

$$\theta(lca_{T_1}(f, f')) \quad = \quad lca_{T_2}(f, f') \ . \tag{2}$$

REMARQUE 2.1
Soit T un arbre est S un ensemble d'étiquettes, on a

$$T|S \ \subseteq_h \ T$$

REMARQUE 2.2
La relation $\subseteq_h$ est transitive, i.e.,

$$T_1 \subseteq_h T_2 \text{ et } T_2 \subseteq_h T3 \quad \Rightarrow \quad T_1 \subseteq_h T_3$$

REMARQUE 2.3
Soit T_1, T_2 deux arbres t.q. $T_1 \subseteq_h T_2$, soit $\theta : V(T_1) \mapsto V(T_2)$ l'homéomorphisme associé et soit $v, v' \in V(T_1)$

$$v <_{T_1} v' \quad ssi \quad \theta(v) <_{T_2} \theta(v') \tag{3}$$
$$v =_{T_1} v' \quad ssi \quad \theta(v) =_{T_2} \theta(v') \tag{4}$$

Cette remarque découle directement du fait que θ soit un homéomorphisme.

DÉFINITION 2.6 (ENSEMBLE DES ÉTIQUETTES DES ARBRES SOURCES)
Dans la suite du document, $\mathcal{T} = \{T_1, T_2, ..., T_k\}$ désigne une collection d'arbres sources depuis lesquelles on veut construire un super-arbre. On note

$$F(\mathcal{T}) = \cup_1^k F(T_i) \ .$$

DÉFINITION 2.7 (SOUS-ARBRES)
Soit P un arbre, on appelle *sous-arbre* de P l'ensemble de noeuds descendants d'un noeud $p \in V(P)$. Par abus de notation, p désignera le sous-arbre.

Si p est un sous-arbre de l'arbre P, on note $p^1, p^2, ..., p^r$ les arbres obtenus en supprimant la racine de p. Ces arbres sont appelés *sous-arbres fils* de p.

DÉFINITION 2.8 (FEUILLES ET SOUS-ARBRES SPÉCIFIQUES)
Soit $\mathcal{T} = \{T_1, \ldots, T_k\}$ une colection d'arbres, une feuille $x \in F(\mathcal{T})$ est dite *spécifique* si elle n'apparaît que dans l'un des arbres de $\mathcal{T}$. On note $I\!F(T_i)$ les feuilles spécifiques d'un arbre T_i et

$$I\!F(\mathcal{T}) = \cup_1^k I\!F(T_i)$$

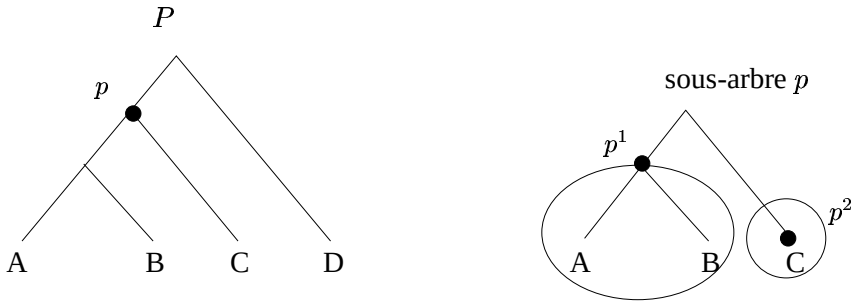

FIG. 3 – Exemple de sous-arbre p d'un arbre P. Les sous-arbres fils de p sont p^1 et p^2.

l'ensemble des feuilles spécifiques d'une collection. On s'interessera parfois aux feuilles spécifiques d'un arbre T_i plus particulièrement situées dans l'un de ses sous-arbres p :

$$\mathbb{F}(p) = F(p) - \cup_{j \neq i} F(T_j)$$

ainsi qu'aux feuilles spécifiques d'un sous-arbre $p \in T_i$ en excluant celles présentes dans un sous-arbre fils p^a de p :

$$\mathbb{F}_{\setminus p^a}(p) = \mathbb{F}(p) - \mathbb{F}(p^a).$$

Un sous-arbre de $T_i \in \mathcal{T}$ est dit *spécifique* s'il ne contient que des feuilles spécifiques.

2.3 Mast : sous-arbre d'accord maximum

Nous rappelons ici la définition du problème $Mast$ (*maximum agreement subtree)* [22], avant d'en donner une généralisation dans le contexte des super-arbres (cf section 3).

DÉFINITION 2.9 (SOUS-ARBRE D'ACCORD)
Soit $\mathcal{T} = \{T_1, \ldots, T_k\}$ une collection d'arbres sur le même ensemble F de feuilles, on appelle *sous-arbre d'accord* de $\mathcal{T}$ tout arbre T' t.q.

$$T' \subseteq_h Ti, \forall i \in [1, k] \quad (5)$$

$$F(T') \subseteq F \quad (6)$$

DÉFINITION 2.10 (SOUS-ARBRE D'ACCORD MAXIMUM)
Soit $\mathcal{T}_{saa}$ l'ensemble des sous-arbres d'accord d'une collection d'arbres $\mathcal{T}$ sur le même ensemble de feuilles. Alors $T' \in \mathcal{T}_{saa}$ est appelé *sous-arbre d'accord maximum* ssi

$$|T'| = max_{T_x \in \mathcal{T}_{saa}} |T_x| \ .$$

On note dans ce cas $T' = MAST_l(\mathcal{T})$.

On note $F' = MAST_F(\mathcal{T})$ tout ensemble de feuilles F' t.q. $F' = F(T')$ et $T' = MAST_l(\mathcal{T})$,

Le nombre de sous-arbres arbres d'accord maximaux peut être exponentiel, mais $|T'|$ est cependant unique.

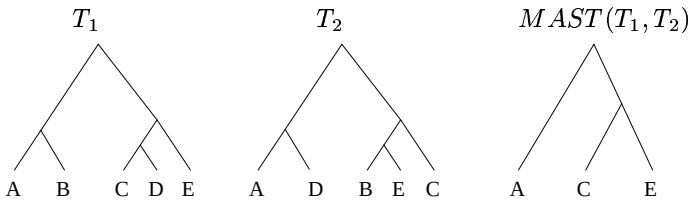

FIG. 4 – Exemple de sous-arbre d'accord maximum

3 Super-arbre d'accord maximum

3.1 Définition

DÉFINITION 3.1 (SUPER-ARBRE D'ACCORD)
Soit $\mathcal{T}$ une collection d'arbres définis sur $F(\mathcal{T})$. Alors T est un *super-arbre d'accord* de $\mathcal{T}$ ssi il vérifie les trois conditions suivantes :

$$T|F(T_i) \subseteq_h T_i \ , \forall i \in [1,k] \tag{7}$$

$$F(T) \cap F(T_i) = \emptyset \text{ ou } F(T) \cap (F(T_i) - F(T_i)) \neq \emptyset \ , \forall i \in [1,k] \tag{8}$$

$$F(T) \subseteq F(\mathcal{T}) \tag{9}$$

Remarquons que tout sous-arbre d'un super-arbre d'accord T d'une collection $\mathcal{T}$ est encore un super-arbre d'accord de $\mathcal{T}$.

DÉFINITION 3.2 (SUPER-ARBRE D'ACCORD MAXIMUM)
Soit $\mathcal{T}_{Saa}$ l'ensemble des super-arbres d'accords de $\mathcal{T}$. Alors $T \in \mathcal{T}_{Saa}$ est un *super-arbre d'accord maximum* de $\mathcal{T}$ ssi

$$|T| = max_{T_x \in \mathcal{T}_{Saa}} |T_x| \ .$$

Le nombre de super-arbres d'accord maximum peut être exponentiel, cependant $|T|$ est unique, et sera noté dans la suite $SuperMast(\mathcal{T})$. Un super-arbre d'accord maximum d'une collection $\mathcal{T}$ sera, lui, noté $SuperMast_t(\mathcal{T})$.

On notera aussi $F = SuperMast_F(\mathcal{T})$ tout ensemble F de feuilles t.q. $F = F(T)$ et $T = SuperMast_t(\mathcal{T})$.

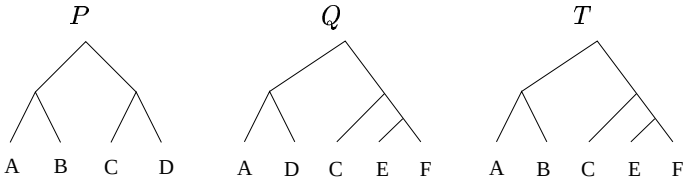

FIG. 5 – Exemple de super-arbre d'accord maximum (T) de deux arbres (P, Q).

REMARQUE 3.1
Si $\mathcal{T}$ est une collection d'arbres n'ayant aucune feuille en commun, ie $F(\mathcal{T}) = \cup_{T_i \in \mathcal{T}} F(T_i)$, alors $SuperMast(\mathcal{T}) = 0$ car il n'existe pas d'arbre sur $F(\mathcal{T})$ qui puisse vérifier la condition (8).

La condition (8) de la définition 3.1 est automatiquement vérifiée pour le MAST dans le contexte de reconstruction phylogénétique classique. Toutefois dans le contexte super-arbre, où les arbres sources ont des feuilles spécifiques, il est nécessaire de donner cette précision afin que le super-arbre traduise effectivement un *accord* entre les arbres sources. Par exemple, tout arbre défini sur $\cup_{1\ldots k} I\!F(T_i)$ a bien une restriction homéomorphiquement incluse dans tout arbre source, cependant il ne traduit aucun accord entre les arbres sources pour le placement des feuilles car aucune d'entre elle n'est partagée par plusieurs arbres sources. Cette condition (8) est nécessaire pour que le théorème 22 soit valide.

Cette condition (8) implique aussi que si un arbre source ne contient que des feuilles spécifiques (i.e., aucune feuille qu'on retrouve dans un autre arbre source), il sera ignoré pour le calcul du super-arbre d'accord maximum. Dans la suite du document nous supposerons que tout arbre source de la collection $\mathcal{T}$ possède au moins une feuille en commun avec un autre arbre source. Si on note $T := SuperMast_t(\mathcal{T})$, Ceci ne signifie pas pour autant que $\forall T_i \in \mathcal{T}, F(T_i) \cap F(T) \neq \emptyset$, comme le montre l'exemple suivant :

$\mathcal{T} = \{T_1, T_2, T_3\}$
$T_1 = (X, Y, A)$
$T_2 = ((A, (B, U_1, U_2, U_3)), (C, V_1, V_2, V_3))$
$T_3 = ((A, (C, V_1, V_2, V_3)), (B, U_1, U_2, U_3))$
$T = ((C, V_1, V_2, V_3), (B, U_1, U_2, U_3))$

où T_1 n'a aucune feuille en commun avec $T = SuperMAST_T(\{T_1, T_2, T_3\})$, en raison de l'incompatibilité des triplets, pourtant $A \in F(T_1) \cap F(T_2) \cap F(T_3)$.

REMARQUE 3.2
Si T est un super-arbre d'accord de la collection $\mathcal{T}$, alors $\{T_1|F(T), \ldots, T_k|F(T)\}$ est une collection d'arbres compatibles.

3.2 Lien entre $MAST(\mathcal{T})$ et $SuperMast(\mathcal{T})$

La définition de *SuperMAST* que nous donnons ci-dessus est une généralisation naturelle de celle de *MAST* au cas des super-arbres (ensemble de feuilles potentiellement différents).

En effet, dans le cas où tous les arbres sources contiennent le même ensemble de feuilles, les deux définitions coïncident :

- puisque $F = F(T_1) = \ldots = F(T_k) = F(\mathcal{T})$, et les équations (6) et (9) coïncident.
- Si T est un arbre t.q. $F(T) \subseteq F = F(\mathcal{T})$, alors $\forall i, 1 \leq i \leq k, T|F(T_i) = T$ et les équations (5) et (7) coïncident.
- la condition (8) devient caduque car vérifiée pour tout arbre dont les feuilles sont dans F. En effet : $I\!F(T_i) = \emptyset$.

Donc dans un tel cas, chercher le plus grand arbre vérifiant les conditions (5)-(6) est équivalent à chercher le plus grand arbre vérifiant les conditions (7)-(9).

3.3 Propriétés du super-arbre d'accord maximum

3.3.1 Intégration des feuilles spécifiques

Dans le cas du MAST, si p et q sont resp. des sous-arbres des arbres T_1, resp. T_2, les feuilles de p et q qui ne sont pas dans $F(p) \cap F(q)$ sont écartées car elles se trouvent ailleurs dans les arbres T_1 et T_2, autrement dit, les deux arbres sont en désaccord sur leur placement. Dans le cas des super-arbres, des feuilles peuvent appartenir à l'un des deux sous-arbres (p ou q) et pas à l'autre, sans qu'un désaccord soit en cause : il s'agit des feuilles spécifiques, pour lesquelles un seul arbre donne l'information de placement.

PROPOSITION 1
Soit $T := SuperMast_t(\mathcal{T})$*, alors* $\forall T_i \in \mathcal{T}$ *t.q.* $F(T_i) \cap F(T) \neq \emptyset$ *on a* $\mathbb{F}(T_i) \subseteq F(T)$.

PREUVE :
Soit T_i t.q. $F(T_i) \cap F(T) \neq \emptyset$, on montre par l'absurde que toute feuille de $\mathbb{F}(T_i)$ est dans T, sinon on peut la greffer dans T en conservant ses propriétés et donc T n'est pas de cardinalité maximum, une contradiction.

Plus précisément, soit $a \in \mathbb{F}(T_i) - F(T)$, et soit u le plus petit noeud ancêtre de f dans T_i t.q. $u = lca_{T_i}(f, a)$ avec $f \in F(T_i) \cap F(T)$ (un tel noeud existe forcément car $F(T_i) \cap F(T) \neq \emptyset$). Deux cas sont alors possibles suivant qu'il existe ou pas un sommet $f' \neq f,\ f' \in F(T_i) \cap F(T)$ t.q. $u = lca_{T_i}(f', a)$.

- si un tel f' existe alors soit $v = lca_T(f, f')$ et soit T' l'arbre T auquel on ajoute l'arête (v, a).
- sinon soit v le sommet interne de T auquel la feuille f est attachée, et soit T' l'arbre T auquel on greffe la feuille $\{a\}$ a un nouveau noeud interne créé entre v et f.

Dans les deux cas, on a $T'|F(T_i) \subseteq_h T_i$, en raison de $T|F(T_i) \subseteq_h T_i$ et du fait que l'arête ajoutée ne remet pas ceci en question. Pour tout autre arbre $T_j \neq T_i,\ T_j \in \mathcal{T}$, on a $T'|F(T_j) = T|F(T_j) \subseteq_h T_j$, donc T' est super-arbre d'accord de $\mathcal{T}$ et en même temps $|T'| > |T|$, une contradiction avec la définition de T. □

Note : le résultat précédent tient aussi pour des arbres non-enracinés. Il suffit d'enraciner T_i et T sur l'arête menant à une feuille commune avant de procéder à la greffe.

Quand on n'a que deux arbres sources (ayant au moins une feuille en commun), on est sûr que le super-arbre d'accord maximum contient au moins une feuille commune aux deux arbres, ce qui nous permet de savoir qu'il contient toutes les feuilles spécifiques des arbres sources :

COROLLAIRE 2
Si $\mathcal{T} = \{T_1, T_2\}, F(T_1) \cap F(T_2) \neq \emptyset$ *et* $T = SuperMast_t(\mathcal{T})$ *alors* $\mathbb{F}(\mathcal{T}) \subseteq F(T)$.

PREUVE :
Supposons que $T := SuperMast_t(T_1, T_2)$ ne contienne aucune feuille de T_1 alors soit $a \in F(T_1) \cap F(T_2)$, en procédant exactement de la même façon que dans la preuve précédente on peut montrer qu'il existe un arbre T' t.q. $F(T') = F(T) \cup \{a\}$ et T' est arbre d'accord de T_1 et T_2, et comme $|T'| > |T|$ on obtient une contradiction avec la définition de T. Ce qui montre que T possède au moins une feuille de $F(T_1) \cap F(T_2)$ donc d'après la proposition précédente, que $\mathbb{F}(T_1) \subseteq F(T)$ et $\mathbb{F}(T_2) \subseteq F(T)$, donc que $\mathbb{F}(\mathcal{T}) \subseteq F(T)$. □

Dans le cas où on a plus de deux arbres sources, la proposition précédente ne tient pas, car il existe des cas où un T_i peut ne voir aucune de ses feuilles communes retenue dans T. Toutefois en pratique, la probabilité d'un tel événement est très faible, car il faudrait que toutes ses feuilles apparaissant dans d'autres arbres soient systématiquement dans des triplets en conflits avec d'autres triplets contenant

d'autres feuilles (n'apparaissant pas dans T_i) et que ce soit ces feuilles qui soient choisis pour rester dans T. Autrement dit, dans la grande majorité des cas on aura $I\!F(\mathcal{T}) \subseteq F(T)$, ce qui est un trait intéressant du super-arbre d'accord maximum, nous garantissant qu'il n'aura pas une taille triviale.

3.3.2 Consistence de la méthode

Si l'on considère le problème MAT comme une méthode pour obtenir une estimation de la phylogénie sous-jaçente aux arbres sources, on peut montrer que cet estimateur est consistent :

DÉFINITION 3.3 (COMPATIBILITÉ D'ARBRES)
Une collection d'arbres $\mathcal{T}$ est dite *compatible* ou *consistent* s'il existe un arbre T t.q. $\forall T_i \in \mathcal{T}, T_i \subseteq_h T$.

Comme [36, 2] on exige ici que $T|F(T_i) = T_i$, ie que chaque arbre source soit un sous-arbre homéomorphique de l'arbre T. Cette définition de la compatibilité est plus forte qu'une autre définition rencontrée dans la littérature (nommée parfois *compatibilité faible - weak compatibility*) [46, 12] qui exige que $T|F(T_i) \rightarrow T_i$, ie que tout T_i puisse être obtenu depuis $T|F(T_i)$ en contractant éventuellement certaines arêtes.

Le résultat ci-dessous montre que si $\mathcal{T}$ est une collection d'arbres compatibles alors tout super-arbre d'accord maximum de $\mathcal{T}$ conserve l'ensemble des feuilles de $\mathcal{T}$:

PROPOSITION 3
Si $\mathcal{T}$ est une collection d'arbres compatibles, $\forall T = SuperMast_t(\mathcal{T})$, $F(T) = F(\mathcal{T})$.

PREUVE :
Par définition, si $\mathcal{T}$ est compatible, il existe un arbre T (pour lequel on peut sans perte de généralité supposer $F(T) \subseteq F(\mathcal{T})$) t.q. pour tout $T_i \in \mathcal{T}$ on a $T_i \subseteq_h T$. Donc T est un super-arbre d'accord de $\mathcal{T}$ et il est de taille maximum car $\cup_{T_i \in \mathcal{T}} F(T_i) = F(\mathcal{T}) \subseteq F(T) \subseteq F(\mathcal{T})$. □

Si $\mathcal{T}$ est compatible en un seul arbre T, la méthode MAT désigne cet arbre comme unique arbre solution.

Dans le cas d'une collection d'arbres sources incompatibles, le super-arbre d'accord excluera certaines feuilles mais ne sera jamais en désaccord avec les arbres de données. En terme de triplets, on a le résultat suivant :

PROPOSITION 4
Si $xy|z \subseteq_h T$ où $T := SuperMast_t(\mathcal{T})$, alors $\forall T_i \in \mathcal{T}$, $xz|y \not\subseteq_h T_i$ et $yz|x \not\subseteq_h T_i$.

PREUVE :
Il suffit de remarquer que sinon on n'aurait pas $T_i \subseteq_h T$, contredisant le fait que T soit un super-arbre d'accord de $\mathcal{T}$. □

4 NP-difficulté du problème MAT

Soit $\mathcal{T} = \{T_1, \ldots, T_k\}$ une collection d'arbres. Nous notons MAT le problème consistant à calculer $SuperMAST(\mathcal{T})$. Dans le cas où tous les arbres de $\mathcal{T}$ sont définis sur le même ensemble de feuilles, nous notons *MAST* le problème consistant à calculer $MAST(\mathcal{T})$.

Le problème *MAST* étant un cas particulier du problème MAT, ce dernier est au moins aussi difficile à résoudre. Nous montrons ci-dessous qu'en fait le problème MAT est plus difficile.

Le problème *MAST* est NP-difficile pour même trois arbres sources de degré non borné [2] (par réduction du problème *Three-Dimensional Matching*). Cependant, il devient polynomial pour un nombre quelconque d'arbres sources dès que l'un d'entre eux a un degré borné [2, 19].

Nous allons montrer que le problème MAT est lui aussi NP-difficile dans le cas général, et que borner le degré des arbres sources ne suffit pas à rendre le problème facile, puisque dans la réduction, tous les arbres sources sont des triplets enracinés (arbres binaires sur 3 feuilles), autrement dit les arbres non-triviaux les plus simples.

Le problème de décision associé au calcul du super-arbre d'accord maximum est le suivant :

Maximum Agreement super-Tree (MAT)
instance: Une collection $\mathcal{T} = \{T_1, \ldots, T_k\}$ d'arbres enracinés sur un ensemble de feuilles $F(\mathcal{T})$ et un entier $m \geq 0$
question: existe-t-il un sous-ensemble $F' \subseteq F(T)$ d'au plus m feuilles t.q.

$$\exists T, F(T) \subseteq F(\mathcal{T}) \text{ et } \forall T_i \in \mathcal{T}, T_i|F' \subseteq_h T$$

autrement dit, T est un super-arbre d'accord de $\mathcal{T}$ d'au moins $n - m$ feuilles ?

Pour montrer que ce problème est NP-complet, nous allons effectuer une réduction à partir du problème suivant :

3-Hitting Set (3HS)
instance: une collection $\mathcal{C}$ de sous-ensembles de 3 éléments choisis parmi un ensemble fini S de n éléments, un entier $m \geq 0$
question: existe-t-il un sous-ensemble $S' \subseteq S$ de taille au plus m t.q. S' couvre $\mathcal{C}$, i.e., S' contient au moins un élément de tout sous-ensemble présent dans $\mathcal{C}$?

Ce problème est NP-complet [25]. Par équivalence à une variante du problème SET COVER [3], la version d'optimisation admet un algorithme d'approximation de facteur 3 [4, 32]. Ce problème est aussi *fixed-parameter tractable*, le plus performant algorithme exact en date [37] permettant de le résoudre en $O(2.270^m + kn^3)$ où k est le nombre d'arbres sources. Signalons aussi au passage qu'un article de Downey et al [18] mentionne un papier non publié [11] contenant une réduction du problème *MAST* vers ce même problème $3HS$ (donc une réduction dans le sens inverse de celle présentée ici et concernant le problème *MAST* et non MAT).

THEOREME 5
Le problème MAT *est NP-complet.*

PREUVE :
Ce problème est trivialement dans NP : étant donné un arbre T, vérifier les conditions de la définition demande un temps polynomial : la condition (9) se vérifie en un parcours de T ; un parcours conjoint de T avec chaque $T_i \in \mathcal{T}$, tour à tour, permet de vérifier à chaque fois que les conditions (8) et (7) sont vérifiées.

Nous allons montrer comment construire une instance du problème MAT depuis une instance du problème 3HS. On supposera que l'instance de 3HS est telle que le graphe biparti qui la représente est

connexe (cf figure 6) (si ce n'est pas le cas, le problème 3HS peut être résolu indépendamment pour chaque composante connexe, en utilisant à chaque fois une instance de MAT).

La transformation de 3HS vers MAT est la suivante : à chaque sous-ensemble $X \in \mathcal{C}$, on associe deux triplets incompatibles sur les trois éléments de X (soit $A, B, C \in S$), par exemple $T_X = ((A, B), C)$ et $T'_X = ((A, C), B)$ (notation parenthésée des arbres). La donnée du problème MAT est constituée de la collection $\mathcal{T} = \cup_{X \in \mathcal{C}} \{T_X, T'_X\}$ et du paramètre m. Notons que $F(\mathcal{T}) = S$.

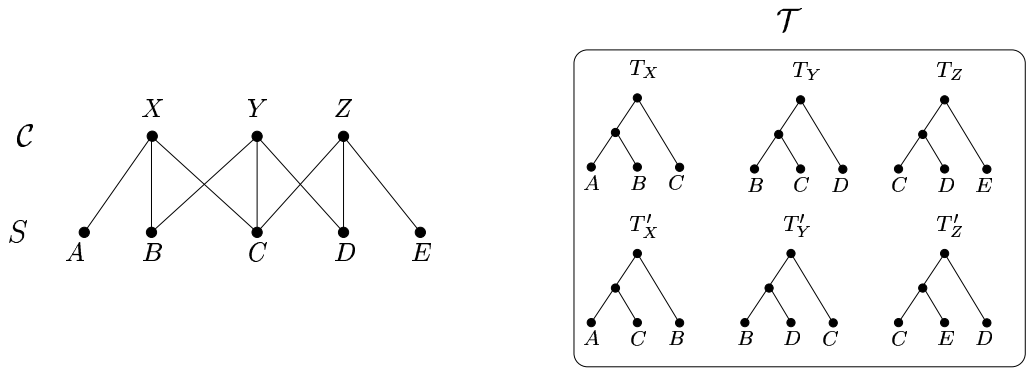

FIG. 6 – Illustration de la réduction. *(i)* Une instance de 3HS représentée sous la forme d'un biparti montrant l'appartenance des éléments de S aux ensembles de la collection $\mathcal{C}$. *(ii)* L'instance de MAT correspondante, chaque sous-ensemble de $\mathcal{C}$ (ex X) donne deux arbres de la collection $\mathcal{T}$ (ex T_X, T'_X).

L'instance de MAT s'obtient en temps polynomial depuis l'instance de 3HS (et est donc aussi de taille polynomiale). Nous montrons maintenant qu'il existe un super-arbre d'accord T de $\mathcal{T}$ t.q. $|T| \geq n - m$ ssi il existe un sous-ensemble $S' \subseteq S$ couvrant $\mathcal{C}$ t.q. $|S| \leq m$.

Soit T un super arbre d'accord de $\mathcal{T}$ t.q. $|T| \geq n - m$, et posons $S' = S - F(T)$. On a $|S'| \leq m$, de plus $S' \subseteq S$ recouvre $\mathcal{C}$ car l'existence de $X \in \mathcal{C}$ non couvert par S' signifie que les 3 sommets A, B, C de S contenus dans X sont dans $F(T)$, ce qui est impossible car cela signifierait $T_X = T|F(T_X) = T|F(T'_X) = T'_X$ (puisque $F(T_X) = F(T'_X) = \{A, B, C\} \subseteq F(T)$ et $T|F(T_X \subseteq_h T_X, T|F(T'_X) \subseteq_h T'_X$) contredisant $T_X \neq T'_X$[3].

Dans l'autre sens, supposons qu'il existe $S' \subseteq S$ couvrant $\mathcal{C}$ t.q. $|S'| \leq m$. Posons $F' = S - S'$. Le fait que S' couvre $\mathcal{C}$ garantit que $\forall X \in \mathcal{C}$, au moins un élément de X (donc une feuille de T_X et T'_X) n'est pas dans F'. Donc tout arbre T sur les feuilles F' contient au plus deux feuilles de tout arbre $T_i \in \mathcal{T}$, ce qui assure $T|F(T_i) \subseteq_h T_i$ donc que T vérifie la condition (7) de la définition 3.1 d'un super-arbre d'accord de $\mathcal{T}$[4]. La condition (8) est aussi vérifiée, par le fait que $\forall T_i \in \mathcal{T}, \not\exists F(T_i) = \emptyset$ (tout ensemble $X \in \mathcal{C}$ donne deux arbre sur les mêmes ensembles de feuilles, donc aucune feuille de T_i n'est spécifique). Enfin, la condition (9) est trivialement vérifiée puisque $F(T) = F' \subseteq S = F(\mathcal{T})$. Ainsi T est un super-arbre d'accord de $\mathcal{T}$ t.q. $|T| = |S| - |F'| \geq n - m$. Donc $SuperMast(\mathcal{T}) \geq n - m$. □

[3] T_X et T'_X sont différents et définis sur les mêmes trois feuilles donc incompatibles.

[4] Le fait que $|T_1|F(T)| \leq 2, \ldots, |T_k|F(T)| \leq 2$ garantit que cet ensemble d'arbres est compatible. Pour qu'il y ait incompatibilité, même indirecte, il faut qu'elle soit traduite sur des triplets.

Il en résulte que le problème consistant à trouver un super-arbre d'accord maximum est NP-difficile en général, même dans le cas où les arbres sources sont des triplets, les arbres non-triviaux les plus simples. En regard, notons que le problème MASTest trivialement polynomial dans le cas où les arbres sources sont des triplets et de façon plus générale qu'il est polynomial si le degré des arbres sources est borné [2].

La section suivante montre que la difficulté du problème MAT est plus grande encore par réduction au problème HS.

5 Complexité paramétrique du problème MAT

Nous nous intéressons maintenant à la complexité paramétrique de la version décision du problème de super-arbre d'accord maximum. Nous montrons que pour le paramètre p (le nombre de feuilles de $F(\mathcal{T})$ non incluses dans un super-arbre d'accord maximum de $\mathcal{T}$) ce problème est $W[2]$-complet. Il est donc, de ce point de vue aussi, strictement plus difficile que le problème MAST qui est lui FPT pour le paramètre p [11, 18].

5.1 Difficulté du problème HITTING SET

Soient H un ensemble et $\mathcal{C}$ une collection d'ensembles. On dit que H est un *hitting set* de $\mathcal{C}$ lorsque H contient au moins un élément de chaque ensemble appartenant à $\mathcal{C}$.

La complexité algorithmique du problème consistant à trouver le plus petit hitting set d'une collection d'ensembles donnée a été exhaustivement étudiée.

5.1.1 Résolutions exactes

Considérons le problème de décision

Hitting Set (HS)
instance: Une collection finie $\mathcal{C}$ d'ensembles finis et non vides et un entier $p \geq 0$.
question: La collection $\mathcal{C}$ admet-elle un hitting set de cardinal au plus p ?

Ce problème est NP-complet et W[2]-complet pour le paramètre p. En effet, HS n'est qu'une formulation alternative du problème SET COVER qui est notoirement NP-complet [17] et W[2]-complet pour le paramètre correspondant à p [13] [21, Proposition 10].

D'autre part, notons d-HS la restriction du problème HS aux instances $(\mathcal{C}, p)$ telles que tous les ensembles de $\mathcal{C}$ sont de cardinal d. Quel que soit $d \geq 2$ fixé, d-HS est NP-complet (remarquons que 2-HS est une formulation alternative du problème VERTEX COVER) mais est FPT pour le paramètre p. En effet, on peut facilement construire un algorithme retournant pour chaque instance $(\mathcal{C}, p)$ de d-HS un hitting set de cardinal p de $\mathcal{C}$ en temps $O(d^p \#\mathcal{C})$ (si il existe). Mais, on peut faire significativement mieux : on trouvera dans [38] le meilleur algorithme connu à l'heure actuelle.

5.1.2 Résolutions approchées

Considérons le problème de minimisation

Minimum Hitting Set (MHS)
instance: Une collection finie $\mathcal{C}$ d'ensembles finis.
solution: Tout hitting set H de $\mathcal{C}$.
measure: Le cardinal de H.

Ce problème a les mêmes propriétés que SET COVER du point de vue de l'approximation. Il admet ainsi un algorithme d'approximation polynomial de borne $1 + \ln(\#\mathcal{C})$ [33, 17]. C'est même le mieux que l'on peut espérer sauf si NP $\subseteq$ DTIME $\left(n^{\ln \ln n}\right)$ [20]. Nous utiliserons le fait que HS admet un algorithme d'approximation polynomial de borne constante si et seulement si P = NP [6].

Puis nous allons montrer que le problème du super-arbre d'accord maximum est un problème complexe du point de vue algorithmique. Formellement nous montrons que le problème de décision

Agreement Super-Tree (AGREEMENT SUPER-TREE)
instance: Une collection finie d'arbres $\mathcal{T}$ et un entier $p \geq 0$.
question: Existe-t-il un super-arbre d'accord de $\mathcal{T}$ de taille au moins $\#F(\mathcal{T}) - p$?

est à la fois NP-complet et W[2]-complet pour le paramètre p même si l'on se restreint aux instances $(\mathcal{T}, p)$ pour lesquelles tous les arbres de $\mathcal{T}$ sont de taille 3 (théorème 8).

Notons que le problème (de décision associé au problème) MAST n'est autre que la restriction du problème AGREEMENT SUPER-TREE aux instances $(\mathcal{T}, p)$ pour lesquelles tous les arbres de $\mathcal{T}$ sont sur le même ensemble de feuilles. Ainsi, AGREEMENT SUPER-TREE est NP-complet même si l'on se restreint aux instances $(\mathcal{T}, p)$ pour lesquelles $\mathcal{T}$ ne contient que trois arbres sur un même ensemble de feuilles [2].

Néanmoins, notre résultat de NP-complétude est non trivial car le problème MAST restreint aux instances $(\mathcal{T}, p)$ pour lesquelles tous les arbres de $\mathcal{T}$ sont de taille 3 se résoud en temps constant (il n'y a qu'un nombre fini d'instances !). La W[2]-difficulté de AGREEMENT SUPER-TREE est encore plus surprenante puisque MAST, dans sa forme générale, est FPT pour le paramètre p [18].

Nous montrons aussi que le problème d'optimisation

Maximum Agreement super-Tree (MAT)
instance: Une collection finie d'arbres $\mathcal{T}$.
solution: Tout arbre d'accord T^* de $\mathcal{T}$.
measure: $\#F(\mathcal{T}) - \#F(T^*)$.

n'admet pas d'algorithme d'approximation de facteur constant, si $P \neq NP$. Un résultat similaire pour le problème MASTest obtenu par [31], montrant qu'il n'existe pas d'algorithme polynomial d'approximation de facteur $2^{\log^{\delta} n}$ pour $\delta < 1$ sauf si $NP \subseteq DTIME[2^{polylog\ n}]$.

5.2 Difficulté du problème de super-arbre d'accord

Cette section est organisé comme suit. Tout d'abord, nous donnons une condition suffisante pour qu'une collection d'arbre soit incompatible (lemme 6). Ensuite, nous introduisons une notation naturelle pour décrire les "rateaux d'arbres". Enfin, dans le lemme 7, nous décrivons le gadget que nous utilisons pour réduire HS à AGREEMENT SUPER-TREE et nous démontrons les propriétés qui nous seront utiles. La réduction proprement dite est décrite dans la démonstration du théorème 8.

Nous considérons maintenant une représentation bien-connue, sous forme de graphe, d'un ensemble d'arbres [1, 12, 45].

DÉFINITION 5.1
Soit $\mathcal{T}$ une collection non vide d'arbres et S un ensemble de feuilles t.q. $S \subseteq F(\mathcal{T})$. On définit le graphe (non orienté) $[\mathcal{T}, S]$ de la façon suivante :

- les sommets sont les éléments de S
- les arêtes sont les paires $\{u, v\} \subseteq S$ pour lesquelles il existe $T \in \mathcal{T}$ et $x \in S$ tels que $x|uv$ soit un sous-arbre de T.

LEMME 6
Soit $\mathcal{T}$ une collection d'arbres, si $[\mathcal{T}, F(\mathcal{T})]$ est connexe alors $\mathcal{T}$ est incompatible.

La preuve est une implication directe du théorème 2 de [12].

DÉFINITION 5.2 (RATEAU)
On définit récursivement une fonction rake associant un arbre à chaque séquence non vide d'arbres sans feuilles communes par :

- $\mathrm{rake}(T) = T$ pour tout arbre T et
- $\mathrm{rake}(T_1, T_2, \ldots, T_k) = (T_1, rake(T_2, \ldots, T_k))$
  pour toute séquence d'arbres T_1, T_2, …, T_k de longueur $k \geq 2$ telle que les $F(T_i)$ ($i \in [1, k]$) soient deux à deux disjoints.

Autrement dit, étant donnée une séquence d'arbres T_1, T_2, …, T_k telle que les $F(T_i)$ ($i \in [1, k]$) soient deux à deux disjoints, on a :

$$\mathrm{rake}(T_1, T_2, \ldots, T_k) = (T_1, (T_2, (T_3, \ldots, (T_{k-2}, (T_{k-1}, T_k)) \ldots)))$$

comme on peut le voir sur la figure 7.

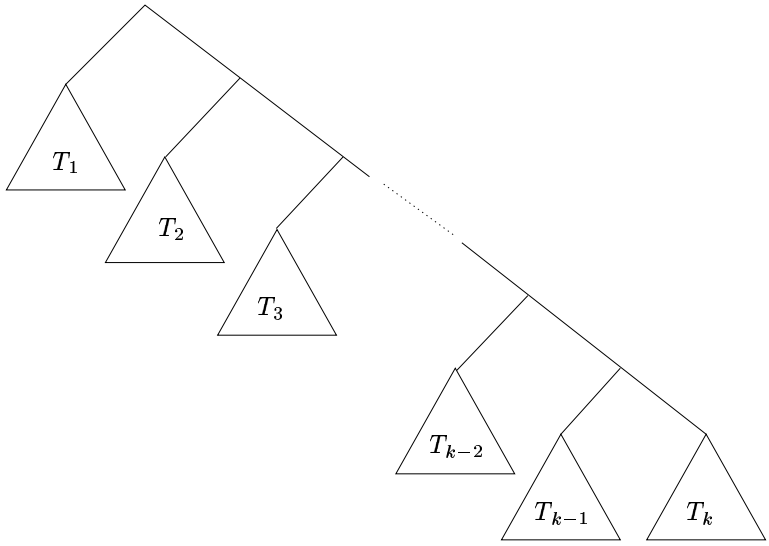

FIG. 7 – $\mathrm{rake}(T_1, \ldots, T_k)$ de rateau d'arbres

Dans cette section, nous n'utiliserons cette notation que dans les preuves du lemme 7 et du théorème 8 et seulement dans le cas où les $k-1$ premier arbres T_1, T_2, …, T_{k-1} sont triviaux[5].

Passons à la description du gadget. Supposons données des étiquettes x^1, x^2, …, x^m.

DÉFINITION 5.3 (GADGET)
Soient un entier $m \geq 1$, et un ensemble x^1, x^2, …, x^m, y^1, y^2, …, y^m d'étiquettes deux à deux distinctes, on définit $\mathcal{G}(x^1, x^2, \ldots, x^m, y^1, y^2, \ldots, y^m)$ la collection d'arbres :

$$\left\{ y^h | y^{h+1} x^{h+1}, y^h | x^{h+1} x^{h+2} \right\}_{h \in [1,m]}$$

où l'on a posé $x^{m+1} := x^1$, $x^{m+2} := x^2$ et $y^{m+1} := y^1$.

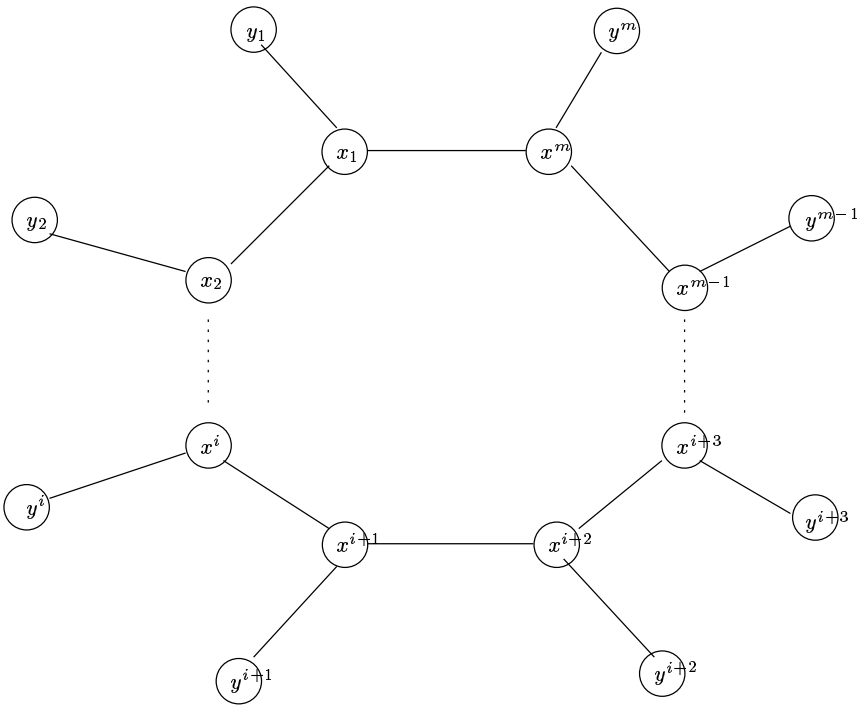

FIG. 8 – Illustration du gadget

Le lemme 7 ci-dessous montre que la collection d'arbres $\mathcal{G}$ définie par le gadget est incompatible, mais que si l'on retire un élément x^j quelconque, alors on peut trouver un arbre dont les feuilles sont $\left(\cup_{i\in[1,m]}\{x^i, y^i\}\right)\setminus\{x^j\}$ et qui est un super-arbre d'accord de $\mathcal{G}$, et dont la restriction à $\left\{x^1, x^2, \ldots, x^m\right\}\setminus\{x^j\}$ est de topologie quelconque.

LEMME 7
Soit $\mathcal{G} := \mathcal{G}(x^1, x^2, \ldots, x^m, y^1, y^2, \ldots, y^m)$.

1. *les* $2m$ *arbres de* $\mathcal{G}$ *ont tous exactement* 3 *feuilles appartenant à* $\{x^1, x^2, \ldots, x^m, y^1, y^2, \ldots, y^m\}$,
2. *les arbres de* $\mathcal{G}$ *sont incompatibles et,*
3. $\mathcal{G}$ *admet pour super-arbres d'accord, tous les arbres de la forme :*

$$\mathrm{rake}(y^j, y^{j+1}, \ldots, y^m, y^1, y^2, \ldots, y^{j-1}, T^*)$$

où j *est un élément de* $[1, m]$ *et* T^* *est un arbre quelconque sur* $\left\{x^1, x^2, \ldots, x^m\right\}\setminus\{x^j\}$.

PREUVE :
L'assertion 1 est vérifiée par construction de $\mathcal{G}$.

L'assertion 2 se déduit du lemme 6 car il est facile de montrer que le graphe $[\mathcal{G}, F(\mathcal{G})]$ associé à $\mathcal{G}$ (illustré sur la figure 8) est connexe.

Reste à démontrer l'assertion 3. Comme on ne modifie pas $\mathcal{G}$ en faisant subir aux séquences x^1, x^2, …, x^m et y^1, y^2, …, y^m un même décalage circulaire, on peut supposer que $j = 1$. Fixant un arbre T^* quelconque sur $\{x^2, x^3, \ldots, x^m\}$, il s'agit de vérifier que l'arbre $T := \mathrm{rake}(y^1, y^2, \ldots, y^m, T^*)$ sur $F(\mathcal{G})\setminus\{x^1\}$ est un super-arbre d'accord de $\mathcal{G}$. Pour cela, on distingue dans $\mathcal{G}$ les arbres qui ne contiennent pas x^1 de ceux qui contiennent cette feuille :

[5] C'est à dire réduit chacun à une feuille.

– il est immédiat que $\forall h \in [1, m-1]$, l'arbre $T_h := y^h|y^{h+1}x^{h+1} \in \mathcal{G}$ est t.q. $T|F(T_h) \subseteq_h T_h$, par construction de T. Il en est de même $\forall h \in [1, m-2]$ pour l'arbre $T_h := y^h|x^{h+1}x^{h+2} \in \mathcal{G}$. Donc que la condition (7) est vérifiée pour T par ces arbres de $\mathcal{G}$.
– $x^1 = x^{m+1}$ est une feuille de $y^m|y^{m+1}x^{m+1}$ et des $y^h|x^{h+1}x^{h+2}$ pour $h \in \{m-1, m\}$. Par suite, les restrictions de chacun de ces trois arbres à $F(\mathcal{G}) \setminus \{x^1\}$ sont des arbres à 2 feuilles, donc trivialement, la condition (7) est aussi vérifiée pour ces arbres.

On a ainsi montré que pour tout $t \in \mathcal{G}$, $t \mid F(T)$ était sous-arbre de T. Ceci termine la preuve de l'assertion 3 et la preuve du lemme. □

Tout est maintenant réuni pour nous permettre de prouver le résultat principal de cette section.

THEOREME 8
Le problème AGREEMENT SUPER-TREE *est :*
– NP*-complet et*
– W[2]*-difficile pour le paramètre* p
même si on le resteint aux instances $(\mathcal{T}, p)$ *pour lesquelles les arbres de* $\mathcal{T}$ *sont de taille* 3.

PREUVE :
Vérifions que AGREEMENT SUPER-TREE **est dans** NP**.** Étant donnée une instance $(\mathcal{T}, p)$ de AGREEMENT SUPER-TREE et un arbre T, on peut vérifier en temps polynomial si T est un super-arbre d'accord de $\mathcal{T}$ de taille au moins $\#F(\mathcal{T}) - p$. En effet, pour tout $T_i \in \mathcal{T}$, on peut tester si $T_i \mid F(T)$ est un sous-arbre de T en temps polynomial en construisant les restrictions $T_i \mid F(T)$ et $T \mid F(T_i)$ puis en testant si elles sont homéomorphes [29, 50]. On en déduit que AGREEMENT SUPER-TREE est dans NP.

Montrons que AGREEMENT SUPER-TREE **est** NP**-difficile et** W[2]**-difficile pour le paramètre** p**.** Pour cela, on va réduire polynomialement et paramétriquement à AGREEMENT SUPER-TREE, le problème HS qui est notoirement NP-complet et W[2]-complet pour le paramètre p (voir section 5.1.1).

Soit $(\mathcal{C}, p)$ une instance de HS.

Notons c le cardinal de $\mathcal{C}$ et $X_1, X_2, \ldots, X_c$ les éléments de $\mathcal{C}$. Pour chaque $i \in [1, c]$, notons m_i le cardinal de X_i et $x_i^1, x_i^2, \ldots, x_i^{m_i}$ les éléments de X_i. De manière synthétique, on peut écrire :

$$\begin{aligned} \mathcal{C} &= \{X_1, X_2, \ldots, X_c\} \\ &= \left\{\left\{x_1^1, x_1^2, \ldots, x_1^{m_1}\right\}, \left\{x_2^1, x_2^2, \ldots, x_2^{m_2}\right\}, \ldots, \left\{x_c^1, x_c^2, \ldots, x_c^{m_c}\right\}\right\}. \end{aligned}$$

Soit alors (y_i^j) une famille injective d'étiquettes n'appartenant pas à $X_1 \cup X_2 \cup \ldots \cup X_c$, indexée sur l'ensemble des couples (i, j) pour lesquels on a $i \in [1, c]$ et $j \in [1, m_i]$. On fabrique ainsi une collection d'ensembles

$$\left\{\left\{y_1^1, y_1^2, \ldots, y_1^{m_1}\right\}, \left\{y_2^1, y_2^2, \ldots, y_2^{m_2}\right\}, \ldots, \left\{y_c^1, y_c^2, \ldots, y_c^{m_c}\right\}\right\}$$

jumelle de $\mathcal{C}$ dont les éléments sont disjoints entre eux et des éléments de $\mathcal{C}$.

Posons :
– $\mathcal{G}_i := \mathcal{G}(x_i^1, x_i^2, \ldots, x_i^{m_i}, y_i^1, y_i^2, \ldots, y_i^{m_i})$ pour tout $i \in [1, c]$ et
– $\mathcal{T} := \mathcal{G}_1 \cup \mathcal{G}_2 \cup \ldots \cup \mathcal{G}_c$.

Au vu de la forme des gadgets $\mathcal{G}_i$ ($i \in [1, c]$) qui est explicitée dans l'énoncé du lemme 7, on se convainct facilement que l'on peut transformer l'instance $(\mathcal{C}, p)$ du problème HS en l'instance $(\mathcal{T}, p)$

du problème AGREEMENT SUPER-TREE en temps polynomial[6]. De plus, si l'on paramétrise les deux problèmes par p notre réduction est de façon évidente paramétrique. Comme l'assertion 1 du lemme 7 garantit que tous les arbres de $\mathcal{T}$ sont de taille 3, il ne reste plus qu'à vérifier que notre réduction est valide c'est-à-dire que les deux assertions suivantes sont équivalentes :

(1) $\mathcal{C}$ admet un hitting set de cardinal au plus p

(2) il existe un super-arbre d'accord de $\mathcal{T}$ de taille au moins $\#F(\mathcal{T}) - p$.

Montrons que $(2) \Rightarrow (1)$**.** Supposons (2) et soit T un super-arbre d'accord de $\mathcal{T}$ de taille au moins $\#F(\mathcal{T}) - p$ ce qui garantit que $H := F(\mathcal{T}) \setminus F(T)$ est de cardinal au plus p.

De plus, soit $i \in [1, c]$. Si l'on avait $F(\mathcal{G}_i) \subseteq F(T)$ alors T serait un super-arbre commun à tous les arbres de $\mathcal{G}_i$ contredisant l'assertion 2 du lemme 7. Il en résulte que l'un au moins des éléments de $F(\mathcal{G}_i)$ n'est pas feuille de T donc est dans H. Ainsi H est un hitting set de $\{F(\mathcal{G}_1), F(\mathcal{G}_2), \ldots, F(\mathcal{G}_c)\}$.

Transformons H de la manière suivante : pour chaque couple (i, j) tel que $y_i^j \in H$, enlevons de H l'élément y_i^j (qui ne peut "toucher" que l'ensemble $F(\mathcal{G}_i)$) et remplaçons-le par un élément quelconque de X_i. Cette opération n'a pas fait augmenter le cardinal de H et l'a changé en un hitting set de $\mathcal{C}$.

On a ainsi démontré (1).

Montrons que $(1) \Rightarrow (2)$**.** Pour tout $i \in [1, c]$ et tout $j \in [1, m_i]$, on note σ_i^j la séquence correspondant au $(j-1)$-ème décalé circulaire (cyclic shift) de la séquence $y_i^1, y_i^2, \ldots, y_i^{m_i}$:

$$\sigma_i^j := y_i^j, y_i^{j+1}, \ldots, y_i^{m_i}, y_i^1, y_i^2, \ldots, y_i^{j-1} .$$

Supposons (1).

Alors, il existe un hitting set H de $\mathcal{C}$ de cardinal au plus p. Pour chaque $i \in [1, c]$, H contient au moins un élément de X_i qui sécrit sous la forme $x_i^{j_i}$ avec $j_i \in [1, m_i]$. Soit T^* un arbre quelconque sur $(X_1 \cup X_2 \cup \ldots \cup X_c) \setminus H$ et

$$T := \mathrm{rake}(\sigma_1^{j_1}, \sigma_2^{j_2}, \ldots, \sigma_c^{j_c}, T^*) .$$

Par construction T est un arbre sur $F(\mathcal{T}) \setminus H$ donc de taille au moins $\#F(\mathcal{T}) - p$.

De plus, pour chaque $i \in [1, c]$, la restriction de T à $F(\mathcal{G}_i)$ n'est autre que $\mathrm{rake}(\sigma_i^{j_i}, T^* \mid X_i)$ qui est, par l'assertion 3 du lemme 7, un super-arbre d'accord de $\mathcal{G}_i$ (remarquons que $x_i^{j_i}$ qui est dans H n'est pas feuille de T donc à plus forte raison n'est pas feuille de $T^* \mid X_i$).

On en déduit que T est un super-arbre d'accord de $\mathcal{T}$.

On a ainsi démontré (2). □

5.3 Non-approximabilité à un facteur constant en temps polynomial

Nous montrons maintenant que le problème du super-arbre d'accord maximum est non approximable par L-réduction depuis MHS.

THEOREME 9
Le problème de minimisation MAT *n'admet pas d'algorithme polynomial d'approximation de borne constante sauf si* $\mathrm{P} = \mathrm{NP}$.

[6]Pour tout $i \in [1, c]$, $\mathcal{G}_i$ est de cardinal $2m_i$ donc $\mathcal{T}$ est de cardinal $2(m_1 + m_2 + \cdots + m_c)$.

PREUVE :
Supposons qu'il existe un algorithme polynomial d'approximation A de borne constante $\rho \in [1, \infty[$ pour le problème MAT.

Nous construisons alors à partir de A un algorithme B pour MHS. Étant donnée une instance $\mathcal{C}$ de MHS, l'algorithme B procède de la manière suivante :

1. On transforme $\mathcal{C}$ en l'instance $\mathcal{T}$ de MAT décrite dans la preuve du théorème 8.
2. On fait tourner l'algorithme A sur l'instance $\mathcal{T}$ et récupére un super-arbre d'accord T tel que $\#F(\mathcal{T}) - \#F(T)$ soit au plus ρ fois le minimum.
3. On calcule $H := F(\mathcal{T}) \setminus F(T)$ puis on transforme H comme dans la preuve du théorème 8 (paragraphe $(2) \Rightarrow (1)$) pour obtenir un hitting set de $\mathcal{C}$.
4. On retourne H.

Par construction, B est un algorithme d'approximation pour MHS de borne ρ et peut être implémenté en temps polynomial. Or, on trouve dans [6] la preuve que ceci n'est possible que si P = NP (remarquer que MHS n'est qu'une formulation alternative de MINIMUM SET COVER où l'on a échangé les rôles entre les ensembles et les éléments). □

Les résultats de cette section s'appliquent aussi au contexte non-enraciné, au sens où l'on peut réduire en temps polynomial le problème MAT *enraciné* au problème MAT *non-enraciné*. Il suffit de greffer à la racine de tout arbre source $T_i \in \mathcal{T}$ un même sous-arbre T_X (comportant de nouvelles feuilles, en nombre suffisant) puis de désenraciner les arbres obtenus, pour obtenir une collection $\mathcal{T}_u$ d'arbres non-enracinés. De façon évidente, $T_X \subseteq_h T$, où $T := SuperMast_t(\mathcal{T}_u)$ et en supprimant T_X de T, puis en l'enracinant au noeud qui menait à T_X, on obtient un arbre T' t.q. $T' := SuperMast_t(\mathcal{T})$.

6 Algorithme polynomial pour le problème MAT dans le cas de deux arbres sources

Nous venons de voir que dans le cas général le problème MAT est NP-difficile. Toutefois, nous montrons dans cette section qu'il devient polynomial dans le cas où l'on dispose uniquement de deux arbres sources, quel que soit leur degré. Pour ce cas particulier, nous montrons d'abord qu'un super-arbre d'accord maximal peut être obtenu par extension d'un sous-arbre d'accord maximal des deux arbres sources réduits à leurs feuilles communes. Nous montrons ensuite qu'on peut ajouter à ce squelette d'arbre les feuilles spécifiques en temps linéaire. Comme résoudre le problème MAST demande un temps polynomial pour deux arbres [34], le problème MAT peut donc être résolu en temps polynomial dans le cas de deux arbres sources.

6.1 Obtention d'un squelette d'arbre par résolution du problème MAST

LEMME 10
Soit $\mathcal{T} = \{T_1, T_2\}$ *et* $F_{12} = F(T_1) \cap F(T_2) \neq \emptyset$.
$\forall T' = MAST_t(T_1|F_{12}, T_2|F_{12}), \exists T = SuperMast_t(T_1, T_2)$ *t.q.* $T|F_{12} = T'$.

PREUVE :
$T' \neq \emptyset$ (tout arbre sur une ou deux feuilles de F_{12} est un sous-arbre de T_1 et de T_2) et T' contient uniquement des feuilles de F_{12} (en raison de (6)). Par définition de T' (condition (5)) on a $T' \subseteq_h T_1$ et

$T' \subseteq_h T_2$ et de la même façon que dans le corollaire 2 on peut lui greffer toutes les feuilles de $\mathbb{F}(\mathcal{T})$ pour obtenir un arbre T t.q.

- $T|F(T_1) \subseteq_h T_1$ et $T|F(T_2) \subseteq_h T_2$ (donc respectant la condition (7)) ;
- $F(T) \subseteq \mathbb{F}(\mathcal{T}) \cup F_{12} \subseteq F(\mathcal{T})$ (donc respectant la condition (9)) ;
- $F(T) \cap F(T_1) \cap F(T_2) = F(T') \neq \emptyset$ (donc respectant la condition (8)).

Donc T est un super-arbre d'accord de $\mathcal{T}$. Il est de taille maximum car supposons qu'il existe $T" = SuperMast_t(\mathcal{T})$ t.q. $|T"| > |T|$, comme $F(T) = \mathbb{F}(\mathcal{T}) \cup F(T')$, on a forcément

$$|F(T") \cap F_{12}| > |F(T) \cap F_{12}| = |F(T') \cap F_{12}| = |T'|.$$

Mais la condition (7) appliquée à $T"$ indique $T"|F(T_1) \subseteq_h T_1$ et $T"|F(T_2) \subseteq_h T_1$ ce qui induit $T"|F_{12} \subseteq_h T_1|F_{12}$ et $T"|F_{12} \subseteq_h T_2|F_{12}$. Donc l'arbre $T"|F_{12}$ serait un sous-arbre d'accord de $T_1|F_{12}$ et $T_2|F_{12}$, de taille $|F(T") \cap F_{12}| > |T'|$, une contradiction avec la définition de T'. □

La figure 9 illustre la situation décrite ci-dessus.

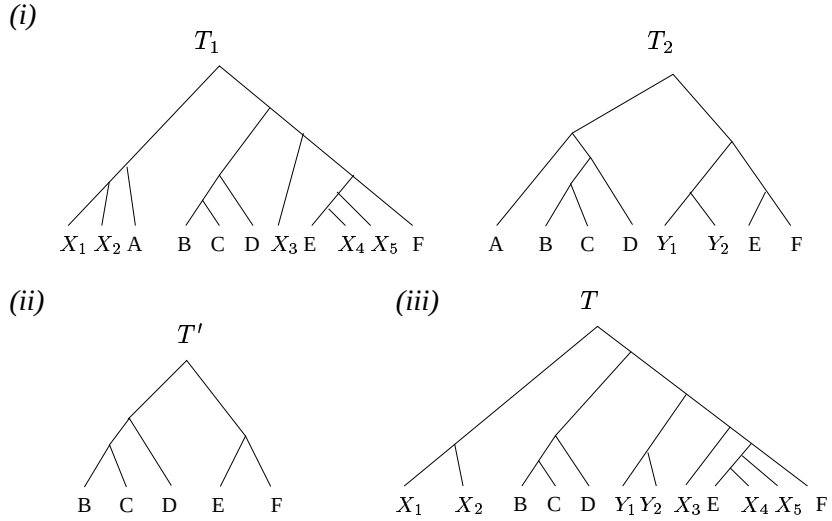

FIG. 9 – (i) Une collection $\mathcal{T} = \{T_1, T_2\}$ de deux arbres sources ; (ii) T' un sous-arbre d'accord maximum des arbres de $\mathcal{T}$ restreints à leurs feuilles communes ; (iii) un super-arbre d'accord maximum T des arbres sources, incluant T' comme sous-arbre.

6.2 Algorithme de construction du super-arbre

Etant donnée une collection $\mathcal{T} = \{T_1, T_2\}$ de deux arbres sources, nous avons décrit ci-dessus comment connaître un ensemble F de feuilles t.q. il existe un super-arbre d'accord maximum de $\mathcal{T}$ ayant F pour ensemble de feuilles. La façon de procéder implique la greffe successive des feuilles spécifiques sur T', le sous-arbre d'accord maximal obtenu depuis les restrictions de T_1 et T_2 à leurs feuilles communes. En pratique, pour être plus rapide, plutôt que de greffer les feuilles spécifiques les unes après les autres, on peut procéder par greffes de *sous-arbres spécifiques* (sous-arbres maximaux composés uniquement de feuilles spécifiques).

Notons que sur certaines arêtes de T' il est possible de greffer à la fois des sous-arbres spécifiques de T_1 et des sous-arbres spécifiques de T_2. Par définition de ces sous-arbres, aucun arbre source ne précise leur agencement respectif (ou même leur imbrication) à partir de cette arête. On voit ainsi

qu'en raison de telles situations, il est possible d'obtenir un grand nombre de super-arbres d'accord maximum.

L'algorithme que nous donnons ci-dessous permet de construire *un* super-arbre d'accord maximum. Toutefois, il est facile de le modifier pour obtenir *tous* les super-arbres d'accord maximum de la collection $\{T_1, T_2\}$ (en considérant aussi pour celà tous les sous-arbres d'accord maximum de $\{T'_1, T'_2\}$).

Avant de donner les algorithmes, introduisons quelques notations utiles à leur description :

DÉFINITION 6.1
Soit deux arbres sources T_1, T_2, on note $F_{12} = F(T_1) \cap F(T_2)$ et $\not\!F_{12} = \not\!F(T_1) \cup \not\!F(T_2) = F(\mathcal{T}) - F_{12}$.

Pour un noeud n d'un arbre, $F(n)$ désignera l'ensemble des feuilles du sous-arbre enraciné en n.

Etant donné $F_m \subseteq F_{12}$, toute feuille $f \in F_m$ est dite *commune* à T_1 et T_2. On dit qu'un noeud n de T_1 (resp. T_2) est *commun* (wrt F_m), si c'est une feuille commune ou si $\exists f, f' \in F_m$ t.q. $n = lca_{T_1}(f, f')$. Dans ce cas on sait qu'il existe un noeud n' *jumeau* dans T_2 (resp. T_1), i.e. tel que $n' = lca_{T_2}(f, f')$ (resp. $n' = lca_{T_1}(f, f')$).

Si n est un noeud d'un arbre T, on désignera par $S(n)$ le sous-arbre complet de T enraciné en n.

L'algorithme 1 (cf figure) décrit la procédure de construction du super-arbre d'accord maximum de deux arbres sources enracinés.

Algorithme 1: Calcul du super-arbre d'accord de deux arbres sources enracinés

Données : Deux arbres enracinés T_1, T_2

Résultat : Un arbre $T := SuperMast_t(T_1, T_2)$

Parcourir récursivement T_1 et T_2 calculer F_{12} et $\not\!F_{12}$

Parcourir récursivement T_1 et T_2 pour obtenir $T'_1 = T_1|F_{12}$ et $T'_2 = T_2|F_{12}$

Calculer $T_m = MAST_t(T'_1, T'_2)$ et $F(T_m)$

Soit $F_{comm} = F(T_m) \cup \not\!F_{12}$

$T_1^{"} \leftarrow T_1|F_{comm}$ et $T_2^{"} \leftarrow T_2|F_{comm}$

pour chaque *noeud commun n dans un parcours en profondeur de $T_1^{"}$ puis $T_2^{"}$* **faire**

- $NbComm(n) \leftarrow$ nb de sous-arbres fils de n contenant des feuilles de F_{comm}
- $Ppac(n) \leftarrow$ le plus proche noeud ancêtre de n commun aux deux arbres
- $Ppdc(n, i) \leftarrow$ le plus proche noeud commun descendant de n dans son i^{eme} sous-arbre fils

pour chaque *paire (n_1, n_2) de noeuds communs jumeaux dans $T_1^{"}$ et $T_2^{"}$* **faire**

- ordonner "de façon compatible" les sous-arbres fils de n : $Fils(n, i)$ et $Fils(n', i)$

/ ajout d'une racine artificielle à T_1 et T_2 pour pouvoir traiter les sous-arbres spécifiques branchant au dessus de $lca(F(T_m))$ */*

$T_1 \leftarrow T_1 \cup (a, r(T_1))$; $T_2 \leftarrow T_2 \cup (a_2, r(T_2))$ et déclarer a et a' noeuds communs jumeaux

/ Greffe des sous-arbres spécifiques à T_2 */*

retourner `Construire(`$lca_T(F(T_m)), lca_{T_2}(F(T_m))$`)`

Ordonner les sous-arbres fils de n et n' noeuds jumeaux dans un "ordre compatible" signifie que les $x = NbComm(n) = NbComm(n')$ premiers sous-arbres fils dans l'ordre sont ceux contenant des feuilles communes et que pour $i \in [1, h]$ on ait

$$F(Fils(n, i)) \cap F(T_m) = F(Fils(n', i)) \cap F(T_m)$$

L'algorithme récursif `Construire` utilisé dans l'algorithme précédent établit un super-arbre d'accord de T_1 et T_2.

Pour ce faire, il doit déterminer comment greffer les sous-arbres spécifiques de T_1 et T_2 sur T' le sous-arbre d'accord maximal obtenu depuis les restrictions de T_1 et T_2 à leurs feuilles communes. La greffe des sous-arbres spécifiques se déroule de façon similaire à la méthode de strict consensus de superarbres [27], sauf dans le cas 4. Toute arête de T' correspond à un chemin dans T_1 et dans T_2 (entre une et plusieurs arêtes). Pour chaque arête de T', il y a quatre situations possibles (cf figure 9) :

1. aucun sous-arbre spécifique n'est connecté au chemin correspondant dans T_1 et T_2 : ne rien greffer dans le super-arbre sur cette arête (par exemple c'est le cas de l'arête menant à la feuille F dans la figure 9 (ii) et (iii)) ;
2. un ou plusieurs sous-arbres spécifiques dans T_1 (mais pas dans T_2) sont connectés au chemin correspondant à l'arête : les greffer dans le même ordre sur l'arête dans le super-arbre construit (par exemple les sous-arbres $\{X_4\}$ et $\{X_5\}$ sur l'arête menant à la feuille E de T' dans la figure) ;
3. situation symétrique où le rôle de T_1 et T_2 est inversé : agir de façon symétrique ;
4. des sous-arbres spécifiques de T_1 *et* de T_2 sont connectés au chemin correspondant : greffer ces sous-arbres avec un interclassement quelconque respectant leur apparitions respectives dans T_1 et T_2. L'algorithme choisit de greffer ceux apparaissant dans T_2 au dessus de ceux apparaissant dans T_1 (mais d'autres possibilités sont possibles, comme évoqué précédemment). Par exemple sur la figure 9, sur l'arête connectant la racine de T' au noeud $lca_{T'}(E, F)$, le sous-arbre (Y_1, Y_2), spécifique à T_2, est greffé au dessus du sous-arbre contenant la feuille X_3, spécifique à T_1.

Les structures de données `Ppac` et `Pddc` initialisées dans l'algorithme 1 permettent de déterminer quels noeuds de T_1 et T_2 correspondent aux extrémités des arêtes de T' (permettant d'effectuer un parcours simultané des parties de T_1 et T_2 se correspondant) et de déterminer dans lequel des quatre cas évoqués ci-dessus l'on se trouve à un moment donné.

L'algorithme 2 (cf figure) montre comment `Construire(`n_1, n_2`)` est réalisé pour deux noeuds communs ($n_1 \in T_1, n_2 \in T_2$). Il engendre lui-même des appels récursifs sur des couples de noeuds communs descendants de n_1 et n_2. Les sous-arbres branchant entre ces deux paires de deux noeuds communs sont pris en compte à la fin de chacun de ces appels récursifs.

La correction de l'algorithme repose sur le lemme 10 (et le corollaire 2).

THEOREME 11
Etant donné une collection $\mathcal{T} = \{T_1, T_2\}$ de deux arbres sources enracinés, l'algorithme 1 renvoit un arbre T t.q. $T := SuperMast_t(\mathcal{T})$.

PREUVE :
Clairement l'arbre T construit par l'algorithme 1 est un super-arbre d'accord de $\{T_1, T_2\}$, i.e. $T|F(T_1) \subseteq_h T_1, T|F(T_2) \subseteq_h T_2$. D'autre part, $T_m \subseteq_h T$ pour $T_m := MAST_t(T'_1, T'_2)$ et $\mathbb{F}(\mathcal{T}) \subseteq F(T)$. Donc T est maximum (sinon sa restriction à F_{12} serait un sous-arbre d'accord de T'_1 et T'_2 plus grand que T_m, ce qui n'est pas possible par définition de T_m). □

Algorithme 2: CONSTRUIRE(n_1, n_2)

Données : Deux noeuds $n_1 \in T_1, n_2 \in T_2$ communs à deux arbres T_1 et T_2.

Résultat : Le sous-arbre de l'arbre $SuperMast_t(T_1, T_2)$ correspondant à $S(n_1), S(n_2)$ et aux sous-arbres spécifiques branchant entre ces noeuds et leurs $Ppac$.

/* 1 - Obtention de la partie de l'arbre correspondant à $S(n_1), S(n_2)$ */

si n_1 *(et* n_2*) sont une même feuille* f **alors** $T \leftarrow \{f\}$

sinon $T \leftarrow$ l'arbre dont la racine a pour sous-arbre fils :

- les arbres resultant de `Construire(`$Ppdc(n_1, i), Ppdc(n_2, i)$`)`, t.q. $i \leq NbComm(n_1)$
- les sous-arbres (spécifiques) $Fils(n_1, i)$, pour $i > NbComm(n_1)$
- les sous-arbres (spécifiques) $Fils(n_2, i)$, pour $i > NbComm(n_2)$

/* **2 - Prise en compte des sous-arbres spécifiques branchant au dessus de** n_1 **et** n_2 */

+

si $Ppac(n_1) \neq Pere(n_1)$ **alors**

  /* greffe de sous-arbres spécifiques venant de T_1 */
  Soit n_i le noeud fils de $Ppac(n_1)$ t.q. $n_1 \in S(n_i)$
  $T \leftarrow S(n_i)$ où $S(n_1)$ est remplacé par T

si $Ppac(n_2) \neq Pere(n_2)$ **alors**

  /* greffe de sous-arbres spécifiques venant de T_2 */
  Soit n_j le noeud fils de $Ppac(n_2)$ t.q. $n_2 \in S(n_j)$
  $T \leftarrow S(n_j)$ où $S(n_2)$ est remplacé par T'

retourner T

6.3 Complexité de l'algorithme

THEOREME 12

La complexité de l'algorithme 1 est en $O(n + N)$, où N dénote la complexité nécessaire au calcul d'un sous-arbre d'accord maximum de deux arbres enracinés.

PREUVE :

Toutes les opérations autres que l'obtention de T_m sont en temps linéaire.

- Toute restriction d'un T^* à un sous-ensemble $L \subseteq F(T^*)$ de feuilles se fait par un simple parcours en profondeur en $O(|L| + |T^*|)$. Dans les algorithmes ci-dessus on a toujours $|L| = O(n)$ et $|T^*| = O(n)$ (par exemple cf [16], bien qu'il soit possible de faire plus simple) . On demande par ailleurs un nombre constant de telles restrictions.
- Déterminer les noeuds de T_1 et T_2 qui sont *communs* et les paires de *jumeaux* entre ces noeuds se fait en temps linéaire : on utilise pour celà les $lca(f, f')$, avec $f, f' \in F(T_m)$. Pour tout arbre T_i de $O(n)$ feuilles, par un preprocessing en $O(n)$ on peut ensuite déterminer en $O(1)$ le noeud $lca_{T_i}(f, f')$ pour $f, f' \in F(T_m)$ [30]. Pour tout noeud n_m de T_m on sait $n_m = lca_{T_m}(f, f')$, en $O(1)$ on peut donc localiser le noeud n_i de $T_i \in \mathcal{T}$ t.q. $n_1 = lca_{T_i}(f, f')$ (on sait $f, f' \in F(T_i)$). Un tel noeud n_i est *commun* et on sait même que n_1 et n_2 sont des noeuds *jumeaux*.
  Ainsi par un seul parcours en profondeur de T_m on identifie tous les noeuds communs et jumeaux de T_1 et T_2 en $|F(T_m)| = O(n)$.
- Les structures de données $Ppac$, $Ppdc$ et $NbComm$ sont ensuite initialisées en un seul parcours en profondeur de $T_i \in \mathcal{T}$.
- On peut ordonner les sous-arbres fils de n et n' noeuds jumeaux en un "ordre compatible" en

$O(n)$ globalement pour les deux arbres en agissant de la façon suivante :

Fixer $\nu(f)$ une numérotation des feuilles de $F(T\_m)$ dans l'ordre gche/dte avec lequel elles apparaissent dans T_1 (i.e., les sous-arbres de T_1 communs sont déjà dans le bon ordre, il ne reste plus qu'a mettre à la fin les sous-arbres spécifiques).

Parcourir récursivement T_1 puis T_2 afin de fixer pour tout noeud n commun $Min[n] = min\_\{f \in F(n) \cap F\_comm\}\nu(f)$ ou $= \infty$ *si* $F(n) \cap F\_comm = \emptyset$

De cette facon on dispose d'un representant commun a T_1 et T_2 pour les sous-arbres fils de n et de n'.

Parcourir T_1 en ordre postfixe T_1 afin de déterminer pour tout noeud n commun et son jumeau n' :
$\vec{S}(n) = Fils[n,1], \ldots, Fils[n, NbComm[n]]$
la liste ordonnée par $Min[S\_i]$ croissant des sous-arbres $S\_i \in S(n)$ t.q. $Min[S\_i] \neq \infty$.
Compléter cette liste à la fin par la liste des sous-arbres $S\_i \in S(n)$ spécifiques (dans n'importe quel ordre).

Pour le parcours ci-dessus, on stocke pour chaque feuille f representante (ie celle qui est min) un pointeur vers le sous-arbre de T_1 et un pointeur vers le sous-arb de T_2 qui sont les plus petits sous-arbres non encore ordonnés la contenant (initialement ce sont des feuilles).

Le parcours est guidé par T_1. et se fait en qq sorte conjoitement dans T_2 en se positionnant sur le noeud n' pointé dans T_2 par le representant f de n.

Puis quand on traite conjointement ces noeuds, on met a jour les pointeurs de f et des autres feuilles communes qu'ils contiennent vers ces nouveaux noeuds n et n'.

– Dans l'algorithme CONSTRUIRE, la greffe d'un sous-arbre spécifique se fait en temps proportionnel à sa taille. Donc globalement, greffer tous les sous-arbres spécifiques se fait en $O(|T_1| + |T_2|) = O(n)$. Toutes les autres opérations de cet algorithme effectuant un parcours en profondeur conjointement dans T_1 et T_2 sont trivialement en temps constant.

□

Notons qu'actuellement $N = min\{O(\sqrt{d}n \log n)\ ,\ O(\sqrt{d}n \log^2 \frac{n}{d})\}$ [40, 35] pour les arbres enracinés et $N = O(n^{1.5})$ pour les arbres non-enracinés [34].

6.4 Modification pour le contexte de la reconstruction phylogénétique

Comme nous l'avons noté précédemment, sur certaines arêtes de T' il est possible de greffer à la fois des sous-arbres spécifiques de T_1 et des sous-arbres spécifiques de T_2. Dans un tel cas, les algorithmes précédents choisissent arbitrairement un interclassement correct des sous-arbres spécifiques, parmi tous ceux possibles pour obtenir un super-arbre d'accord, sans se soucier du sens des clades induits par cet interclassement.

Dans le domaine de la reconstruction phylogénétique, le super-arbre produit peut être vu comme un estimateur de la phylogénie sous-jacente aux arbres sources. Aussi ce super-arbre doit-il posséder le moins possible de clades inférés artificiellement et on préfèrera remplacer l'interclassement des sous-arbres spécifiques par une multifourche M les regroupant.

Deux possibilités subsistent pour la multifourche M évoquée :

cas 1 connecter à M les feuilles des sous-arbres spécifiques en question. Ainsi, toute phylogénie sous-jacente aux données (tout interclassement ou imbrication des sous-arbres X non contredit par T_1 et T_2) peut être obtenu par une certaine résolution de la multifourche M de l'arbre produit. Cette alternative rend la procédure de greffe de sous-arbres spécifiques identique à celle de [27] pour la construction d'un super-arbre de consensus strict.

cas 2 dans une perspective où le super-arbre produit servira d'arbre *graine* à la méthode MRP, on peut adopter une position moins stricte que celle du point précédent et conserver la résolution des sous-arbres spécifiques en question, afin que celle-ci soit présente sous forme de caractères dans la super-matrice constuite par MRP. Dans ce cas de figure, on connectera à M non pas les feuilles des sous-arbres spécifiques, mais les sous-arbres eux-mêmes.

Remarquons que dans ces deux cas, le super-arbre construit ne respecte pas (en général) la définition exacte d'un super-arbre d'accord de T_1 et T_2, car il fusionne en un seul noeud M un ensemble de noeuds de T_1 et/ou T_2.

Une légère modification des algorithmes donnés dans la section précédente permet de construire un arbre suivant les deux points de vue précédents, sans en changer la complexité.

Dans l'algorithme 2, à l'endroit où apparaît la marque + , il suffit d'insérer les lignes suivantes :

---

si $Ppac(n_1) \neq Pere(n_1)$ *et* $Ppac(n_2) \neq Pere(n_2)$ **alors**
renvoyer l'arbre dont la racine est une multifourche M connectée :
- à T
- aux [feuilles des] sous-arbres branchant entre n_1 (exclus) et $Ppac(n_1)$ (exclus)
- aux [feuilles des] sous-arbres branchant entre n_2 (exclus) et $Ppac(n_2)$ (exclus)

Ci-dessus la mention "[feuilles des]" correspond a l'alternative où on veut construire un arbre le plus general possible (cas 1).

La phylogénie sous-tendant les arbres sources est généralement supposée binaire et la présence de multifourches dans une estimation de cette phylogénie traduit une incertitude et non la proposition d'un phénomène de multispéciation. Toute multifourche inférée *spécifiquement* dans le super-arbre d'accord maximum (i.e., non présente dans les arbres sources) par les algorithmes précédents, indique un manque de recouvrement entre arbres sources pour une partie de la phylogénie sous-jacente. Toute méthode combinatoire d'inférence de super-arbre autre que celle-ci aura la même difficulté à interclasser les sous-arbres spécifiques dans un tel cas. Les multifourches présentes dans le super-arbre d'accord maximum indiquent explicitement les parties de la phylogénie pour lesquels il est nécessaire de collecter de nouvelles données.

6.5 Cas de deux arbres sources non-enracinés

Dans le domaine de la reconstruction phylogénétique, Steel et al [47] ont montré que le problème de la construction d'un super-arbre n'admet pas de solution satisfaisante dans le contexte enraciné. Toutefois, dans d'autres domaines on peut souhaiter calculer un super-arbre d'accord d'arbres non-enracinés.

Le corollaire 2 montre comment dans le cas de deux arbres sources enracinés, toutes les feuilles spécifiques sont incluses dans le super-arbre d'accord maximum. Ceci résulte du fait que le placement d'une feuille spécifique qu'indique l'arbre source qui la contient, ne peut entraîner de contradiction avec l'autre arbre source car celui-ci ne la contient pas.

La même explication s'applique aussi naturellement au cas d'arbres non-enracinés, pour lequel un équivalent du lemme 10 peut être dérivé facilement. Aussi pour obtenir un super-arbre d'accord maximum de deux arbres sources $\{T_1, T_2\}$ non-enracinés on peut procéder ainsi :

Soit $T' := MAST_t(T_1|F_{12}, T_2|F_{12})$
Enraciner T', T_1, T_2 en une feuille f commune aux trois arbres
Soit T l'arbre renvoyé par l'algorithme `Construire(`*n1,n2*`)` où $n1$, resp. $n2$, est le noeud de T_1 correspondant à la feuille f
Renvoyer l'arbre non-enraciné obtenu en désenracinant T

Soit $O(N')$ le temps nécessaire pour calculer T' le sous-arbre d'accord non-enraciné des arbres non-enracinés $T_1|F_{12}$ et $T_1|F_{12}$. Enraciner les trois arbres en une feuille commune, appliquer l'algorithme CONSTRUIRE(n_1, n_2) et désenraciner l'arbre T sont des opérations nécessitant un temps $O(n)$. On peut donc obtenir un super-arbre d'accord maximum de deux arbres sources non-enracinés en temps $O(n + N')$ (actuellement $N' = n^{1.5}$ [34]).

6.6 Impossibilité de l'extension de cet algorithme pour $k > 2$

L'obtention d'un super-arbre depuis k arbres sources en étendant les algorithmes 1 et 2 est tout à fait envisageable, donnant un algorithme de complexité $O(kn+N)$. Malheureusement cette approche ne garantit pas d'obtenir un *super-arbre d'accord maximum* des k arbres :

- l'ensemble $\cap_{T_i \in \mathcal{T}} F(T_i)$ des feuilles communes à *tous* les arbres sources peut être vide (ce qui n'empêche pas l'existence d'un super-arbre d'accord de $\mathcal{T}$ mais rend le passage par l'algorithme $MAST$ impraticable : $F_{MAST} = \emptyset$).
- plus grave, un sous-arbre d'accord maximum des arbres sources (réduits à leurs feuilles communes) n'est pas forcément inclus dans un de leur super-arbre d'accord maximum, comme le montre l'exemple de la figure 10 pour trois arbres sources. Dans cet exemple, aucun sous-arbre d'accord maximum des arbres sources réduits aux feuilles communes ($\{A, B, C, D, E, F\}$) n'est inclus homéomorphiquement dans le seul arbre T qui soit super-arbre d'accord maximum de $\mathcal{T}$.

7 Algorithme alternatif pour l'obtention du super-arbre d'accord maximum de deux arbres enracinés

L'algorithme de la section précédente ne pouvant être étendu aux collections de plus de deux arbres sources, nous donnons maintenant un autre algorithme pour le cas particulier de deux arbres sources. Cet algorithme est lui susceptible de pouvoir être étendu à plus de deux arbres sources (avec

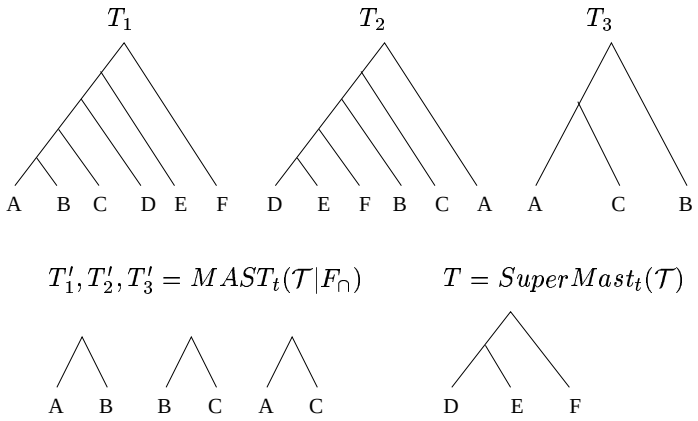

FIG. 10 – Une collection $\mathcal{T} = \{T_1, T_2, T_3\}$ de trois arbres sources pour lesquels $MAST_t(T_1|F, T_2|F, T_3|F) \not\subseteq_h SuperMast_T(\mathcal{T})$ où $F = F(T_1) \cap F(T_2) \cap F(T_3)$.

une complexité non-polynomiale toutefois, en raison de la la NP-difficulté du problème MAST pour plus de deux arbres sources). Il s'agit d'un algorithme de programmation dynamique inspiré de celui proposé par Steel et Warnow [48] pour le problème du *sous*-arbre d'accord maximum de deux arbres.

Ci-dessous nous rappelons l'algorithme *MAST* de [48], puis nous précisons le lien entre *MAST* et *SuperMast,* ce qui nous amène à un algorithme de programmation dynamique pour le calcul de `SuperMast`.

7.1 Calcul du MAST [48]

$MAST$ est un algorithme qui renvoit le taille d'un sous-arbre d'accord maximum de deux arbres (que les arbres soient enracinés ou non enracinés) qui procède par exploration conjointe de leurs sous-arbres, en partant de ceux correspondant aux feuilles.

Soient P et Q deux arbres sources, $p \in P$ et $q \in Q$ deux de leurs sous-arbres. L'algorithme *MAST* est un algorithme de programmation dynamique : la valeur $MAST(p, q)$ pour deux sous-arbres est obtenue en faisant référence aux valeurs $MAST(p^a, q^b), MAST(p^a, q), MAST(p, q^b)$ où p^a, resp. q^b, est l'un sous-arbre fils de p, resp. q.

Soit $G(p, q)$ le graphe biparti dont toute arête (a, b) a une valuation $w(a, b) = MAST(p^a, q^b)$, et soit $W(p, q)$ la valeur d'un couplage de poids maximum de ce graphe biparti. La valeur de $MAST(p, q)$ est alors donnée par la formule suivante :

$$
\begin{aligned}
MAST(p,q) \quad &= 0 \quad && \text{si } F(p) \cap F(q) = \emptyset && (10)\\
&= 1 \quad && \text{si } p = \{x\} \subseteq F(q) \text{ ou } q = \{x\} \subseteq F(p) && (11)\\
&= max \quad && \{ && (12)\\
& && W(p,q), && (13)\\
& && MAST(p, q^1),\ \ldots\ , MAST(p, q^s)\ , && (14)\\
& && MAST(p^1, q),\ \ldots\ , MAST(p^r, q)\ \} \text{ sinon} && (15)
\end{aligned}
$$

où $p^1, p^2, ..., p^r$ sont les sous-arbres fils de p, et $q^1, q^2, ..., q^s$ ceux de q.

7.2 L'algorithme SMAST

Nous allons maintenant introduire un algorithme pour le calcul de la taille d'un super-arbre d'accord maximum de deux arbres sources P et Q. Il s'agit d'un algorithme de programmation dynamique résultant d'une formule de calcul récursive produisant une valeur que nous noterons $SMAST$ pour des couples de sous-arbres (p, q) avec $p \in P, q \in Q$. Cet algorithme est dérivé de l'algorithme *MAST* auquel on apporte plusieurs modifications pour l'adapter au contexte des super-arbres où les arbres sources sont définis sur des ensembles de feuilles différents. Ce contexte plus complexte fait que pour deux *sous-arbres* $p \in P$ et $q \in Q$ on n'a pas forcément égalité entre $SMAST(p, q)$ (la valeur calculée) et $SuperMast(p, q)$ (notre objectif). Toutefois, la relation entre ces deux valeurs (montrée dans le théorème 22) permet d'établir que pour P et Q, les *arbres entiers*, $SMAST(P, Q) = SuperMast(P, Q)$ (cf corollaire 23).

Les modifications apportées à la formule du *MAST* pour obtenir celle de *SMAST* sont essentiellement dues aux feuilles spécifiques des arbres sources qu'il faut intégrer de différentes façons suivant les cas de figure. La section ci-dessous montre que ces feuilles spécifiques peuvent être intégrées. Les sections suivantes font le tour des modifications nécessaires pour passer du calcul de *MAST* à celui de *SMAST*.

Nous montrons ci-dessous comment la prise en compte des feuilles spécifiques nécessite de modifier la formule de *MAST* pour en obtenir une calculant le *SuperMAST* de deux arbres sources P et Q. Nous distinguons les différents cas de figure dans l'appariement d'un sous-arbre de P à un sous-arbre de Q.

7.2.1 Appariement d'un sous-arbre à une feuille

PROPOSITION 13
Si $\mathcal{T} = \{P, Q\}, P = \{x\}, x \in F(Q)$ *alors* $Q = SuperMast_t(\mathcal{T})$.

PREUVE :
En effet, Q vérifie toutes les conditions d'un super-arbre d'accord de P et Q :

- $F(Q) \subseteq I\!F(\mathcal{T}) = F(Q) \cup F(P)$ (condition (9)) ;
- $x \in F(Q) \cap F(P)$ et $x \in F(Q)$ (condition (8)) ;
- on a trivialement $Q|F(Q) = Q \subseteq_h Q$, et comme P n'est qu'une feuille appartenant à Q on a aussi $Q|F(P) \subseteq_h P$ (condition (7)).

De plus $x \in F(Q)$ d'où $F(Q) = F(\mathcal{T})$, ce qui montre qu'il ne peut pas exister d'arbre d'accord de P et Q plus grand (sinon il ne vérifierait pas la condition (9)). □

Cette proposition nous sera utile dans les preuves suivant la définition de SMAST. Toutefois, dans le cadre de l'algorithme de programmation dynamique, nous ne pouvons pas l'appliquer directement pour modifier l'équation (11), réglant le cas où l'un des deux sous-arbres considérés est réduit à une feuille, présente dans l'autre sous-arbre.

En effet, nous nous intéressons non pas à deux arbres, mais à deux sous-arbres $p \in P, q \in Q$. Cette différence a pour conséquence que nous ne pouvons pas intégrer toutes les feuilles de q dans le cas où $p = \{x\} \subseteq F(q)$, car les feuilles de $F(q) - I\!F(q)$ sont forcément dans l'arbre P ailleurs que dans p, ce qui signifie qu'il est trop tôt pour les intégrer au moment où on considère p. En conséquence, nous fixerons $SMAST(p, q) = 1 + I\!F(q)$ dans ce cas (équations (19) et sa symétrique (20)).

7.2.2 Appariement d'un sous-arbre à un sous-arbre fils de l'autre sous-arbre

Une autre modification nécessaire concerne le cas où l'un des deux sous-arbres de données, par exemple p, est apparié à un sous-arbre fils (ex q^b) du second sous-arbre, q. Dans le MAST classique (cf section 7.1) on exclut totalement les feuilles des autres sous-arbres de q. Toutefois dans le cas des super-arbres, il est possible de conserver les feuilles spécifiques de ces sous-arbres (comme nous venons de le voir aussi précédemment).

En conséquence, une autre adaptation de la formule de MAST pour SMAST est dans l'équation (14) d'ajouter un terme de type $|I\!F_{\setminus q^b}|$ à toute association de p à un sous-arbre fils q^b de q. De même dans l'équation (15), on ajoute un terme de type $|I\!F_{\setminus p^a}|$ à toute association de q à un sous-arbre fils p^a de p. La preuve du théorème 22 montrera que l'ajout de ce terme est justifié.

7.2.3 Appariement entre sous-arbres fils

La modification suivante concerne le cas où les sous-arbres fils de p sont associés à ceux de q, ie le cas où on recourt à un graphe biparti (équation (13) dans le cas du *MAST*).

Imaginons qu'on construise le biparti de façon identique au problème du MAST, en donnant un poids $w(i,j) = SMAST(i,j)$ (et non plus $MAST(i,j)$) aux arêtes entre sous-arbres p^a et q^b. Une fois obtenu un couplage de poids maximum, il faut maintenant en plus tenir compte des sommets *isolés*, ie non joints par le couplage. Chacun d'entre eux représente un sous-arbre d'un arbre source et peut contenir des feuilles spécifiques à cet arbre source, donc incorporables dans le super-arbre. La valeur renvoyée pour $SMAST$ serait donc la valeur du couplage proprement dit (ie la somme du poids de ses arêtes), à laquelle on ajoute le nombre de feuilles spécifiques à chaque sous-arbre correspondant à un sommet isolé du graphe dans le couplage.

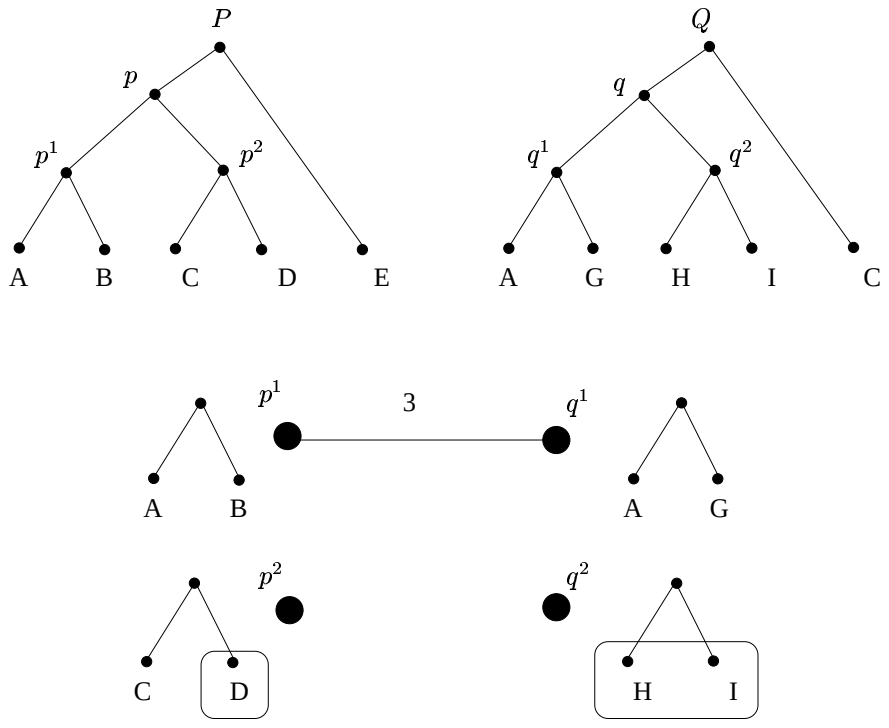

FIG. 11 – Association entre sous-arbres fils de p et de q.

Ainsi dans l'exemple de la figure 9, où l'on cherche à calculer $SMAST(p,q)$, la valeur du couplage en-lui même vaut trois, mais il faut tenir compte des sommets p_2 et q_2 qui sont isolés. p_2 contient une feuille spécifique à P, en l'occurence D, et q_2 contient deux feuilles spécifiques à Q, en l'occurence H et I.

La valeur de $SMAST(p,q)$ correspondante est donc de 6. La figure 12 montre le super-arbre d'accord maximum de p et q correspondant à cette valeur.

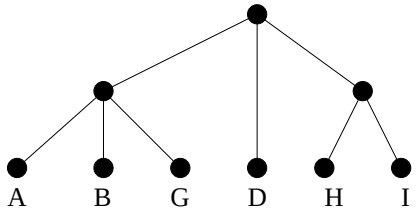

FIG. 12 – Arbre associé au couplage précédent

En fait, la prise en compte des sommets isolés change le pbm du biparti qu'on cherche à résoudre pour (p,q), car il existe des cas où l'on obtient une plus forte valeur en laissant volontairement des sommets isolés dans le couplage (et tel que aucun autre terme de la définition de SMAST ne permet d'obtenir un score aussi élevé).

Ainsi sur l'exemple $P = ((A,(H,H')),(B,B'),I)$ et $Q = ((A,(B,B')),I)$ où I est un sous-arbre résolu de 10 feuilles, un couplage maximum a pour valeur 13 (il ne prend pas (B,B')), alors qu'en ne prenant pas A on obtient une valeur de 14. Les autres termes permettant d'obtenir une valeur de $SMAST$ donnent des valeurs inférieures.

Ainsi, dans le cas des super-arbres il semble que l'on doit résoudre un problème d'optimisation quelque différent du cas du MAST. Toutefois, on peut se ramener au problème du couplage maximum en modifiant quelque peu le graphe biparti sur lequel on travaille : soit $p^1,\ldots,p^r$ les sous-arbres fils et $q^1,\ldots,q^s$ ceux de q, on définit le graphe ainsi : $G(p,q) = (X,Y,E)$ avec

$$X = \{p^1,\ldots,p^r,v_1,\ldots,v_s\}$$

$$Y = \{v'_1,\ldots,v'_r,q^1,\ldots,q^s\}$$

les arêtes E et leur valuation sont les suivantes :

$\{(p^a,q^b)$ de valeur $SMAST(p^a,q^b)$ pour tout p^a et $q^b\}$,
$\cup$ $\{(p^a,v'_a)$ de valeur $|I\!F(p^a)|$ pour tout $p^a\}$,
$\cup$ $\{(v_b,q^b)$ de valeur $|I\!F(q^b)|$ pour tout $q^b\}$.

Les sommets v, v' sont des sommets *virtuels* destinés à ne laisser aucun sous-arbre fils de p ou q non inclus dans un couplage maximum (sauf éventuellement s'il ne contient aucune feuille spécifique). L'utilisation d'une arête (p^a,v'_a) dans un tel couplage signifie que p^a n'est associé à aucun sous-arbre de q, donc on ne garde de p^a que les feuilles spécifiques, $I\!F_{p^a}$, la valeur de l'arête (p^a,v'_a). On notera $\widetilde{W}(p,q)$ la valeur renvoyée par un couplage maximum sur ce graphe $G(p,q)$.

PROPOSITION 14
Si il existe un couplage maximum de $G(p,q)$ qui n'utilise pas un sommet p^a, resp. q^b, alors $I\!F(p^a) = \emptyset$, resp. $I\!F(q^b) = \emptyset$.

PREUVE :
Supposons que p^a ne soit pas connecté par une arête dans le couplage, alors v_a n'est pas connecté

non plus car il est uniquement relié à p^a dans $G(p,q)$. Il est donc possible d'ajouter l'arête (p^a, v_a) au couplage. Cette arête est forcément de poids nul car sinon le couplage ne serait pas maximum car il ne contient pas cette arête. Or par définition, le poids de cette arête est égal à $|\mathbb{F}(p^a)|$. La preuve est identique dans le cas d'un sommet q^b. □

7.2.4 Exception à la prise en compte des feuilles spécifiques

Il est important de noter que le cas où $F(p) \cap F(q) = \emptyset$ est une exception à la règle de prise en compte des feuilles spécifiques. En effet dans ce cas, l'arbre $(p|\mathbb{F}(p), q|\mathbb{F}(q))$ (notation parenthésée) vérifie bien les axiomes (7) et (9) des super-arbres, mais il ne vérifie pas l'axiome d'accord entre arbres sources (8) qui impose que tout arbre source dont le super-arbre utilise des feuilles voit au moins une de ses feuilles non-spécifiques inclue dans le super-arbre.

Cette condition est nécessaire pour ancrer dans le super-arbre les feuilles spécifiques de l'arbre source. Si l'on décidait de se passer de cette condition et d'instaurer ainsi par exemple que

$$\begin{aligned} SMAST(p,q) &= |F(p)| + |F(q)| \text{ si } F(p) \text{ est spécifique et } F(p) \cap F(q) = \oslash & (16)\\ &= 0 \text{ dans les autres cas où } F(p) \cap F(q) = \oslash & (17) \end{aligned}$$

alors sur l'exemple $P = (C, p = (A, B))$ $Q = (q = (C, A), q' = (G, B))$ le résultat obtenu est incorrect, même en limitant l'application de (16) au cas où p est une feuille. Dans ce cas en effet, $SMAST(A, G) = 2$ donc $SMAST(p, q') = 3$ (biparti) et ensuite $SMAST(P, Q) = 4$ (biparti) ce qui est impossible car ça signifie conserver toutes les feuilles de P, Q, donc les 3 feuilles communes $\{A, B, C\}$, or P et Q sont incompatibles sur ces 3 feuilles.

La source du problème vient du fait que quand fixe $SMAST(p,q) = |F(p)| + |F(q)|$ on fait abstraction du fait que $F(q)$ qui n'est pas spécifique, ne sera peut-etre pas gardé car en conflit avec un autre arbre source. Or le sous-arbre spécifique p n'a peut-etre rien à voir avec q, aucun point d'ancrage commun à P et Q ne permet de savoir que c'est ici qu'il faut intégrer ce sous-arbre spécifique. Ainsi, on surélève artificiellement $SMAST(p,q)$ ce qui encourage à le garder, induisant plus tard des conflits.

Ainsi dans le cas où $F(p) \cap F(q) = \emptyset$, on fixera $SMAST(p,q) = 0$ comme dans le cas du *MAST*

7.2.5 Formule SMAST pour calculer $SuperMAST(P,Q)$

Tous les ingrédients sont maintenant réunis pour donner la formule permettant de calculer SMAST par un algorithme de programmation dynamique :

$$SMAST(p,q) = \begin{cases} 0 & \text{si } F(p) \cap F(q) = \emptyset & (18)\\ 1 + \mathbb{F}(q) & \text{si } F(p) = \{x\} \subseteq F(q) & (19)\\ 1 + \mathbb{F}(p) & \text{si } F(q) = \{x\} \subseteq F(p) & (20)\\ max & \{ & (21)\\ & \quad \widetilde{W}(p,q), \\ & \quad SMAST(p, q^1) + |\mathbb{F}_{\setminus q^1}(q)|, \ldots, SMAST(p, q^s) + |\mathbb{F}_{\setminus q^s}(q)|, \\ & \quad SMAST(p^1, q) + |\mathbb{F}_{\setminus p^1}(p)|, \ldots, SMAST(p^r, q) + |\mathbb{F}_{\setminus p^r}(p)| \ \} \text{ sinon} \end{cases}$$

7.3 Lien entre SMAST et SuperMAST

Dans le déroulement de l'algorithme SMAST, pour deux sous-arbres p et q de deux arbre P et Q, on n'obtient généralement pas l'égalité entre $SMAST(p,q)$ et $SuperMast(p,q)$. Cela provient du fait que l'on ne peut pas considérer p et q de façon indépendante du reste des arbres auxquels ils appartiennent (ici P et Q). Ainsi dans l'exemple de la figure 14, la feuille C n'est pas conservée, alors que si l'on avait considéré p et q comme étant des arbres à part entière, elle aurait dû apparaître. Le fait que cette feuille n'ait pas été conservée vient du fait qu'elle apparait dans Q en dehors de q.

Cependant, si l'on exclut ce type de feuilles des sous-arbres considérés, il existe une relation naturelle entre $SMAST$ et $SuperMast$ qui sera l'objet du théorème 22. Nous définissons ci-dessous formellement les ensembles de feuilles auxquels nous désirons nous restreindre. Rappellons que les feuilles sont identifiées à leurs étiquettes pour simplifier les notations, ce qui explique qu'on puisse avoir des *feuilles* appartenant à plusieurs arbres.

DÉFINITION 7.1
Soit P et Q deux arbres sources pour lesquels on veut obtenir un super-arbre d'accord maximum et soit $p \in P$ et $q \in Q$ deux de leurs sous-arbres. En considérant conjointement p, q, P, Q on peut définit les sous-ensembles de feuilles suivants :

- $F(\bar{p}) = F(P) - F(p)$
- $F(\bar{q}) = F(Q) - F(q)$
- $F(p\backslash\bar{q}) = F(p) - F(\bar{q}) = I\!F(p) \cup (F(p) \cap F(q))$
- $F(q\backslash\bar{p}) = F(q) - F(\bar{p}) = I\!F(q) \cup (F(p) \cap F(q))$

Autrement dit, $F(p\backslash\bar{q})$ dénote l'ensemble des feuilles du sous-arbre p qui lui sont spécifiques ou qui apparaissent aussi dans le sous-arbre q.

Une remarque importante est que dans la suite, tous les ensembles de feuilles (et donc les arbres restreints) que nous considèrerons seront définis respectivement à P et Q les deux arbres sources pour lesquels on veut calculer $SuperMast(P,Q)$. Ceci permet d'alléger les notations en ne fisant pas figurer explicitement P et Q dans les notations $F(p\backslash\bar{q}), F(q\backslash\bar{p}), p_{\backslash\overline{q}}, q_{\backslash\overline{p}}$ alors qu'ils interviennent dans leur définition. Ainsi, que l'on parle du couple (p,q) ou (p,q^b), où q^b est un sous-arbre fils de q, dans les deux cas, $I\!F(p)$ est calculé respectivement à Q.

Depuis les ensembles de feuilles précédents, on peut définir les arbres suivants :

DÉFINITION 7.2

- $p_{\backslash\overline{q}} = p|F(p\backslash\bar{q})$
- $q_{\backslash\overline{p}} = q|F(q\backslash\bar{p})$
- $p^a{}_{\backslash\overline{q}}$ pour $p^a|F(p^a\backslash\bar{q})$
- $q^b{}_{\backslash\overline{p}}$ pour $q^b|F(q^b\backslash\bar{p})$

où p^a, resp. q^b, est sous-arbre fils de $p \in P$, resp. $q \in Q$.

Autrement dit, on passe de p à $p_{\backslash\overline{q}}$, resp. $p_{\backslash\overline{q^b}}$, en lui enlevant les feuilles qui apparaissent dans Q ailleurs que dans son sous-arbre q, resp. q^b.

Les figures 13 et 14 illustrent ces définitions. Dans la figure 13, la feuille G du sous-arbre p ne fait pas partie de $F(p\backslash\bar{q})$ car elle apparaît en dehors du sous-arbre q dans l'arbre Q. De même, $D \in F(q)$ mais $D \notin F(q\backslash\bar{p})$ car elle apparaît en dehors du sous-arbre p dans l'arbre P. Si l'on s'intéresse au couple de sous-arbres (p,q^b), où q^b est le sous-arbre fils de q défini sur la figure, pour passer de p à $p_{\backslash\overline{q^b}}$, on perd aussi la feuille B qui fait partie de q mais pas de q^b.

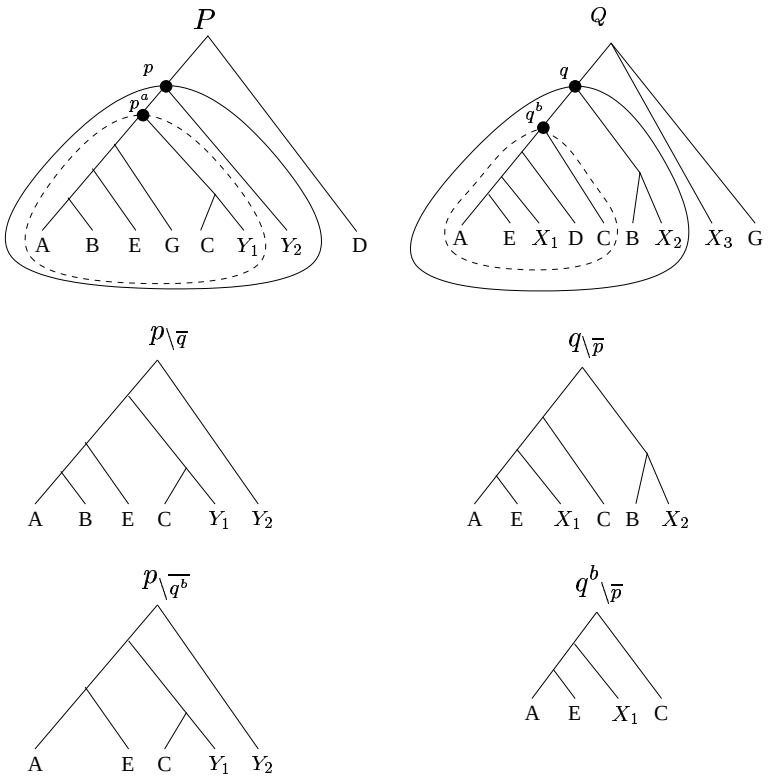

FIG. 13 – Exemple d'arbres P et Q pour lesquels on veut calculer un super-arbre d'accord. Pour tout couple de sous-arbres $(p \in P, q \in Q)$ on s'intéresse aux sous-arbres $p_{\backslash\overline{q}}$ et $q_{\backslash\overline{p}}$ qu'ils induisent (cf définitions). Ici $F(p\backslash\bar{q}) = \{A, B, C, E, Y_1, Y_2\}$ et $F(q\backslash\bar{p}) = \{A, B, C, E, X_1, X_2\}$. Pour un sous-arbre fils q^b de q, on s'intéresse aussi conjointement aux couples de sous-arbres $p_{\backslash\overline{q^b}}$ et $q^b{}_{\backslash\overline{p}}$ où q^b est sous-arbre fils de q.

Nous nous intéresserons plus particulièrement à certains sommets dans l'arbre constituant le super-arbre d'accord maximum de $p_{\backslash\overline{q}}$ et $q_{\backslash\overline{p}}$. On définit :

DÉFINITION 7.3
- T pour $SuperMast_t(p_{\backslash\overline{q}}, q_{\backslash\overline{p}})$
- r la racine de T,
- $r_{p_{\backslash\overline{q}}} = lca_T(F(p_{\backslash\overline{q}}) \cap F(T))$
- $r^i_{p_{\backslash\overline{q}}} = lca_T(F({p_{\backslash\overline{q}}}^i) \cap F(T))$ où ${p_{\backslash\overline{q}}}^i$ est sous-arbre fils de $p_{\backslash\overline{q}}$
- $r_{q_{\backslash\overline{p}}} = lca_T(F(q_{\backslash\overline{p}}) \cap F(T))$
- $r^j_{q_{\backslash\overline{p}}} = lca_T(F({q_{\backslash\overline{p}}}^j) \cap F(T))$ où ${q_{\backslash\overline{p}}}^j$ est sous-arbre fils de $q_{\backslash\overline{p}}$

Les notations précédentes sont définies dès que p et q ont au moins une feuille en commun, comme le montre le lemme suivant :

LEMME 15
Si $F(p) \cap F(q) \neq \emptyset$ *alors*
(i) $|T| \geq 1$
(ii) $F(T) \cap F(p) \neq \emptyset$ *et* $F(T) \cap F(q) \neq \emptyset$

PREUVE :
(i) : si $F(p) \cap F(q) \neq \emptyset$, soit x une feuille de $F(p) \cap F(q)$. Par définition de $p_{\setminus\overline{q}}$ et $q_{\setminus\overline{p}}$ on a $x \in F(p_{\setminus\overline{q}}) \cap F(q_{\setminus\overline{p}})$, donc l'arbre-feuille $\{x\}$ est un super-arbre d'accord de $p_{\setminus\overline{q}}$ et $q_{\setminus\overline{p}}$. Donc dans ce cas on est sûr que $|T| = SuperMast(p_{\setminus\overline{q}}, q_{\setminus\overline{p}}) \geq |\{x\}| = 1$.
(ii) : en appliquant l'axiome (9) à T on a

$$F(T) \subseteq F(p_{\setminus\overline{q}}) \cup F(q_{\setminus\overline{p}}) = I\!F(p) \cup I\!F(q) \cup (F(p) \cap F(q))$$

donc on ne peut avoir $F(T) \cap F(p) = \emptyset$ car sinon $F(T) \subseteq I\!F(q)$ une contradiction avec l'axiome (8) de la définition de T vis-à-vis de l'arbre source $q_{\setminus\overline{p}}$. □
Donc si $F(p) \cap F(q) \neq \emptyset$, T est défini et comporte des feuilles de p et q, donc $r, r_{p_{\setminus\overline{q}}}, r_{q_{\setminus\overline{p}}}$ sont définis.

Il existe de nombreuses fonctions d'homéomorphismes entre les arbres que nous considérons. La définition suivante les détaille :

DÉFINITION 7.4
- $\theta_{T,p_{\setminus\overline{q}}} : V(T|F(p_{\setminus\overline{q}})) \mapsto V(T)$ l'homéomorphisme associé au fait que $T|F(p_{\setminus\overline{q}}) \subseteq_h T$
- $\theta_{p_{\setminus\overline{q}}} : V(T|F(p_{\setminus\overline{q}})) \mapsto V(p_{\setminus\overline{q}})$ l'homéomorphisme associé au fait que $T|F(p_{\setminus\overline{q}}) \subseteq_h p_{\setminus\overline{q}}$
- $\theta_p : V(p_{\setminus\overline{q}}) \mapsto V(p)$ l'homéomorphisme associé au fait que $p_{\setminus\overline{q}} \subseteq_h p$
- $\phi_{p_{\setminus\overline{q}}} : V(T) \mapsto V(p_{\setminus\overline{q}}) \cup \emptyset$ l'application partielle résultant de la composition de $\theta^{-1}_{T,p_{\setminus\overline{q}}}$ et $\theta_{p_{\setminus\overline{q}}}$. En fait, si l'on se restreint aux sommets de T ayant un antécédant dans $T|F(p_{\setminus\overline{q}})$, on a un morphisme de $T|\theta^{-1}_{T,p_{\setminus\overline{q}}}(T)$ dans $p_{\setminus\overline{q}}$. Mais comme tout sommet de T n'a pas forcément d'antécédent par $\theta^{-1}_{T,p_{\setminus\overline{q}}}$, $\phi_{p_{\setminus\overline{q}}}$ n'est qu'une application partielle (préservant toutefois la structure).
- $\phi_p : V(T) \mapsto V(p) \cup \emptyset$ l'application partielle résultant de la composition de $\phi_{p_{\setminus\overline{q}}}$ et θ_p.
- $\theta_{T,\hat{q}} : V(T|F(q_{\setminus\overline{p}})) \mapsto V(T)$ l'homéomorphisme associé au fait que $T|F(q_{\setminus\overline{p}}) \subseteq_h T$
- $\theta_{q_{\setminus\overline{p}}} : V(T|F(q_{\setminus\overline{p}})) \mapsto V(q_{\setminus\overline{p}})$ l'homéomorphisme associé au fait que $T|F(q_{\setminus\overline{p}}) \subseteq_h q_{\setminus\overline{p}}$
- $\theta_q : V(q_{\setminus\overline{p}}) \mapsto V(q)$ l'homéomorphisme associé au fait que $q_{\setminus\overline{p}} \subseteq_h q$
- $\phi_{q_{\setminus\overline{p}}} : V(T) \mapsto V(q_{\setminus\overline{p}}) \cup \emptyset$ l'application partielle résultant de la composition de $\theta^{-1}_{T,\hat{q}}$ et $\theta_{q_{\setminus\overline{p}}}$.
- $\phi_q : V(T) \mapsto V(q) \cup \emptyset$ l'application partielle résultant de la composition de $\phi_{q_{\setminus\overline{p}}}$ et θ_q.

Le résultat suivant montre une correspondance entre les sous-arbres fils de p et l'arbre $p_{\setminus\overline{q}}$ ou ses sous-arbres fils :

LEMME 16
(i) Si $\exists\, p^a$ sous-arbre fils de p t.q. $F(p_{\setminus\overline{q}}) \subseteq F(p^a)$ alors $p_{\setminus\overline{q}} = {p^a}_{\setminus\overline{q}}$;
(ii) Si $\nexists\, p^a$ sous-arbre fils de p t.q. $F(p_{\setminus\overline{q}}) \subseteq F(p^a)$, ie $F(p_{\setminus\overline{q}})$ possède des feuilles de plusieurs sous-arbre fils de p, alors $\forall {p_{\setminus\overline{q}}}^i$ sous-arbre fils de $p_{\setminus\overline{q}}$, $\exists\, p^a$ sous-arbre fils de p t.q. ${p_{\setminus\overline{q}}}^i = {p^a}_{\setminus\overline{q}}$.

PREUVE :
(i) Supposons $\exists p^a$ sous-arbre fils de p t.q. $F(p_{\setminus\overline{q}}) \subseteq F(p^a)$. Nous montrons d'abord $F(p_{\setminus\overline{q}}) = F({p^a}_{\setminus\overline{q}})$ par double inclusion :
- on sait

$$F({p^a}_{\setminus\overline{q}}) = F(p^a \setminus \bar{q}) = I\!F(p^a) \cup (F(p^a) \cap F(q))$$

et

$$F(p_{\setminus\overline{q}}) = F(p \setminus \bar{q}) = I\!F(p) \cup (F(p) \cap F(q))$$

or $I\!F(p^a) \subseteq I\!F(p)$ et $(F(p^a) \cap F(q)) \subseteq (F(p) \cap F(q))$, donc $F({p^a}_{\setminus\overline{q}}) \subseteq F(p_{\setminus\overline{q}})$.

– $F(p_{\setminus\overline{q}}) - F(p^a{}_{\setminus\overline{q}})$

$$\begin{array}{lll} \subseteq & F(p^a) - F(p^a{}_{\setminus\overline{q}}) & \text{puisque } F(p_{\setminus\overline{q}}) \subseteq F(p^a) \\ \subseteq & F(p^a) - (I\!F(p^a) \cup (F(p^a) \cap F(q))) & \text{par définition de } p^a{}_{\setminus\overline{q}} \\ \subseteq & F(p^a) \cap F(\bar{q}) & \text{car } F(p^a) = I\!F(p^a) \cup (F(p^a) \cap F(q)) \cup (F(p^a) \cap F(\bar{q})) \end{array}$$

or $F(p_{\setminus\overline{q}}) \cap F(\bar{q}) = \emptyset$ par définition de $p_{\setminus\overline{q}}$ donc $F(p_{\setminus\overline{q}}) - F(p^a{}_{\setminus\overline{q}}) = \emptyset$, i.e., $F(p_{\setminus\overline{q}}) \subseteq F(p^a{}_{\setminus\overline{q}})$.

On a donc $F(p_{\setminus\overline{q}}) = F(p^a{}_{\setminus\overline{q}})$, ce qui permet de conclure :

$$\begin{array}{rcll} p_{\setminus\overline{q}} & = & p|F(p_{\setminus\overline{q}}) & \text{car par définition on a } F(p_{\setminus\overline{q}}) = F(p\setminus\bar{q}) \\ & = & p|F(p^a{}_{\setminus\overline{q}}) & \text{puisque } F(p_{\setminus\overline{q}}) = F(p^a{}_{\setminus\overline{q}}) \\ & = & p^a|F(p^a{}_{\setminus\overline{q}}) & \text{puisque par définition } F(p^a{}_{\setminus\overline{q}}) \subseteq F(p^a) \\ & = & p^a{}_{\setminus\overline{q}} & \text{car par définition on a } F(p^a{}_{\setminus\overline{q}}) = F(p^a\setminus\bar{q}) \end{array}$$

(ii) Supposons $\nexists\, p^a$ sous-arbre fils de p t.q. $F(p_{\setminus\overline{q}}) \subseteq F(p^a)$, i.e., il existe deux étiquettes x, y dans $F(p)$, et deux sous-arbre fils p^a, p^b de p t.q.

$$x \in F(p_{\setminus\overline{q}}) \cap F(p^a) \text{ et } y \in F(p_{\setminus\overline{q}}) \cap F(p^b) \ .$$

Ceci implique $lca_p(x, y) = r(p)$, la racine de p, et donc que $r(p)$ à un antécédent par θ_p dans $p_{\setminus\overline{q}}$. Cet antécédent est forcément $r(p_{\setminus\overline{q}})$ car $p_{\setminus\overline{q}} \subseteq_h p$, i.e. on ne peut avoir de sommet de $p_{\setminus\overline{q}}$ au dessus de $\theta_p^{-1}(r(p))$. Donc

$$r(p_{\setminus\overline{q}}) = \theta_p^{-1}(r(p))$$

Ceci joint à $p_{\setminus\overline{q}} \subseteq_h p$ induit aussi que $\forall {p_{\setminus\overline{q}}}^i$ sous-arbre fils de $p_{\setminus\overline{q}}$ il existe un sous-arbre fils p^a de p t.q. ${p_{\setminus\overline{q}}}^i \subseteq_h p^a$ ie par définition de $\subseteq_h$ on a

$${p_{\setminus\overline{q}}}^i = p^a|F({p_{\setminus\overline{q}}}^i)$$

(notons au passage que dans ce cas on a $F({p_{\setminus\overline{q}}}^i) \subseteq F(p^a)$). Par ailleurs on a par définition

$$p^a{}_{\setminus\overline{q}} = p^a|F(p^a\setminus\bar{q})$$

Donc pour montrer ${p_{\setminus\overline{q}}}^i = p^a{}_{\setminus\overline{q}}$, il nous reste à montrer que $F({p_{\setminus\overline{q}}}^i) = F(p^a\setminus\bar{q})$. On le fait par double inclusion :

- De $F({p_{\setminus\overline{q}}}^i) \subseteq F(p^a)$ et $F({p_{\setminus\overline{q}}}^i) \subseteq F(p_{\setminus\overline{q}})$ on déduit $F({p_{\setminus\overline{q}}}^i) \subseteq I\!F(p^a) \cup (F(p^a) \cap F(q))$ or par définition $I\!F(p^a) \cup (F(p^a) \cap F(q)) = F(p^a\setminus\bar{q})$, donc on a $F({p_{\setminus\overline{q}}}^i) \subseteq F(p^a\setminus\bar{q})$.
- Soit $x \in F(p^a{}_{\setminus\overline{q}}) = F(p^a\setminus\bar{q})$ on a $x \in I\!F(p^a)$ ou $x \in F(p^a) \cap F(q)$ donc $x \in F(p_{\setminus\overline{q}})$. Par ailleurs, comme on a vu que $F({p_{\setminus\overline{q}}}^i) \subseteq F(p^a)$, on sait $\exists y \in F({p_{\setminus\overline{q}}}^i) \cap F(p^a)$.
  Supposons maintenant que $x \notin F({p_{\setminus\overline{q}}}^i)$, alors

$$lca_{p_{\setminus\overline{q}}}(y, x) = r(p_{\setminus\overline{q}}) \ . \tag{22}$$

  Or dans l'arbre p,

$$lca_p(x, y) \leq_p r(p^a) <_p r(p)$$

  donc par (1) et (3), on a $\theta_p^{-1}(lca_p(x, y)) = lca_{p_{\setminus\overline{q}}}(x, y) <_{p_{\setminus\overline{q}}} \theta_p^{-1}(r(p)) = r(p_{\setminus\overline{q}})$ une contradiction avec (22). Donc forcément $\forall x \in F(p^a{}_{\setminus\overline{q}})$ on a $x \in F({p_{\setminus\overline{q}}}^i)$, i.e., $F(p^a{}_{\setminus\overline{q}}) \subseteq F({p_{\setminus\overline{q}}}^i)$.

□

Le lemme précédent tient de façon symétrique pour un sous-arbre fils q^b de q et un sous-arbre fils $q_{\setminus\overline{p}}{}^j$ de $q_{\setminus\overline{p}}$.

Il est facile de montrer qu'au moins l'un des deux arbres $p_{\setminus\overline{q}}, q_{\setminus\overline{p}}$ se projette à la racine de T (par la fonction $\phi^{-1}_{p_{\setminus\overline{q}}}$, resp. $\phi^{-1}_{q_{\setminus\overline{p}}}$) :

LEMME 17
Supposons $F(p) \cap F(q) \neq \emptyset$.
On ne peut pas avoir à la fois $r \neq r_{p_{\setminus\overline{q}}}$ *et* $r \neq r_{q_{\setminus\overline{p}}}$.

PREUVE :
Si $r \neq r_{p_{\setminus\overline{q}}}$ et $r \neq r_{q_{\setminus\overline{p}}}$, alors deux situations sont possibles :

1. $\exists S$ sous-arbre fils de r t.q. $F(S) \cap (F(p_{\setminus\overline{q}}) \cup F(q_{\setminus\overline{p}})) = \emptyset$. Donc $F(T) \not\subseteq F(p_{\setminus\overline{q}}) \cup F(q_{\setminus\overline{p}})$, ce qui contredit le fait que T soit un super arbre d'accord maximum de $p_{\setminus\overline{q}}$ et $q_{\setminus\overline{p}}$ (condition (9) non respectée).
2. $r_{p_{\setminus\overline{q}}}$ et $r_{q_{\setminus\overline{p}}}$ sont les deux fils de r. Alors $F(p_{\setminus\overline{q}}) \cap F(q_{\setminus\overline{p}}) \cap F(T) = \emptyset$. Comme $T = SuperMast_t(p_{\setminus\overline{q}}, q_{\setminus\overline{p}})$, la condition (9) induit alors que $F(T) \subseteq I\!F(p_{\setminus\overline{q}}) \cup I\!F(q_{\setminus\overline{p}})$, ce qui contredit le fait que T soit un super arbre d'accord maximum de $p_{\setminus\overline{q}}$ et $q_{\setminus\overline{p}}$ (condition (8) non respectée).

Dans les deux cas possibles on aboutit à une contradiction, donc on ne peut avoir à la fois $r \neq r_{p_{\setminus\overline{q}}}$ et $r \neq r_{q_{\setminus\overline{p}}}$. □

La remarque suivante établit que si les racines de $p_{\setminus\overline{q}}$ et $q_{\setminus\overline{p}}$ se projettent à la racine de T, alors on peut s'arranger pour qu'aucun sous-arbre fils de T ne contienne des feuilles spécifiques de P et des feuilles spécifiques de Q sans contenir des feuilles de $P \cap Q$ (utile pour la preuve du théorème 22).

REMARQUE 7.1
Si $\exists T = SuperMast_t(p_{\setminus\overline{q}}, q_{\setminus\overline{p}})$ t.q. $r = r_{p_{\setminus\overline{q}}} = r_{q_{\setminus\overline{p}}}$ alors

$$\exists T' = SuperMast_t(p_{\setminus\overline{q}}, q_{\setminus\overline{p}}) \text{ t.q. } \forall T^x \text{sous-arbre fils de } T' :$$

$$\left.\begin{array}{l} F(T^x) \cap F(p) \neq \emptyset \text{ et} \\ F(T^x) \cap F(q) \neq \emptyset \end{array}\right\} \Rightarrow \quad F(T^x) \cap F(p) \cap F(q) \neq \emptyset\ . \quad (23)$$

PREUVE :
Soit $T = SuperMast_t(p_{\setminus\overline{q}}, q_{\setminus\overline{p}})$. Si T contient des sous-arbres fils T^x ne vérifiant pas la propriété (23), alors soit T' l'arbre obtenu en remplaçant dans T tout T^x problématique (ie contenant à la fois des feuilles de $F(p)$ et de $F(q)$ sans contenir de feuilles de $F(p) \cap F(q)$) par deux sous-arbres T^x_p et T^x_q fils de la racine définis ainsi :
- $T^x_p = T^x | F(p)$
- $T^x_q = T^x | F(q)$

Il est facile de vérifier que $T'|F(p_{\setminus\overline{q}}) \subseteq_h p_{\setminus\overline{q}}$ et $T'|F(q_{\setminus\overline{p}}) \subseteq_h q_{\setminus\overline{p}}$ puisque c'est le cas de T et que la transformation précédente ne change pas cet état de fait, puisque $T^x \cap F(p) \subseteq I\!F(p)$ et $T^x \cap F(q) \subseteq I\!F(q)$. Donc T' vérifie la condition (7) pour être super-arbre d'accord de $p_{\setminus\overline{q}}$ et $q_{\setminus\overline{p}}$. Par ailleurs $F(T') = F(T)$ donc les propriétés (8) et (9) de T sont elles aussi conservées pour T'. Puisque $|T| = |T'|$ on conclut $T' = SuperMast_t(p_{\setminus\overline{q}}, q_{\setminus\overline{p}})$.

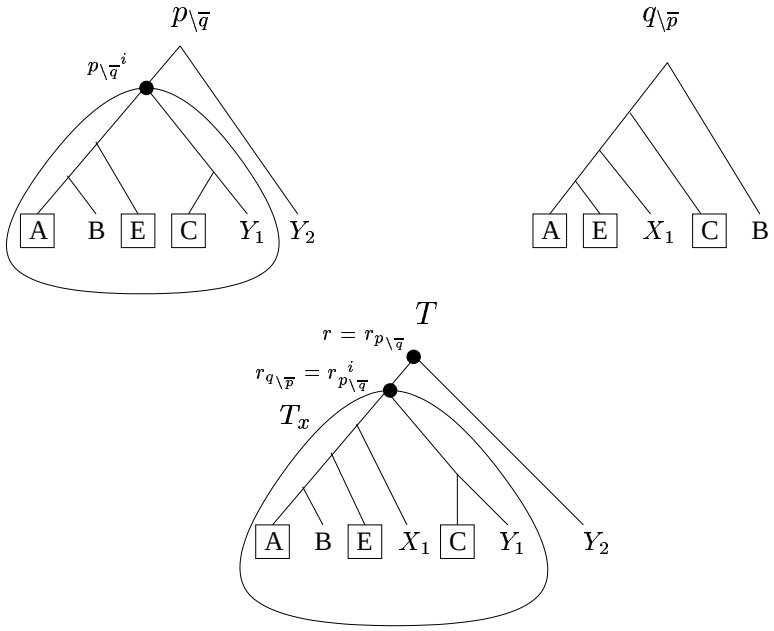

FIG. 14 – T est un super-arbre d'accord maximum de $p_{\setminus\overline{q}}$ et $q_{\setminus\overline{p}}$, les sous-arbres obtenus par restriction de sous-arbres p et q d'arbres sources P et Q (cf fig 13).

□

Le lemme suivant s'applique quand l'arbre T est obtenu par appariement d'un sous-arbre ($q_{\setminus\overline{p}}$ ou $p_{\setminus\overline{q}}$) avec un sous-arbre fils de l'autre, i.e., quand toutes les feuilles de $p_{\setminus\overline{q}}$ (ou $q_{\setminus\overline{p}}$) présentes dans T viennent d'un seul de ses sous-arbres fils.

LEMME 18
Supposons $F(p_{\setminus\overline{q}}{}^i) \cap F(q) \neq \emptyset$.
Si $r^{\,i}_{p_{\setminus\overline{q}}} = r_{p_{\setminus\overline{q}}}$ *alors* T *est un super-arbre d'accord de* $p_{\setminus\overline{q}}{}^i$ *et* $q_{\setminus\overline{p}}$ *(et donc* $|T| \leq SuperMast(p_{\setminus\overline{q}}{}^i, q_{\setminus\overline{p}})$*).*

PREUVE :
On montre dans un premier temps que :

$$\forall p_{\setminus\overline{q}}{}^{i'} \neq p_{\setminus\overline{q}}{}^i \text{ sous-arbre fils de } p_{\setminus\overline{q}} \quad \text{on a} \quad F(T) \cap F(p_{\setminus\overline{q}}{}^{i'}) = \emptyset \tag{24}$$

En effet : $\forall g \in F(p_{\setminus\overline{q}}{}^{i'})$, on ne peut pas avoir $g \in F(T)$ car sinon par définition de $r_{p_{\setminus\overline{q}}}$, on aurait $g \leq_T r_{p_{\setminus\overline{q}}}$ et donc $g \leq_T r_{p_{\setminus\overline{q}}} = r^{\,i}_{p_{\setminus\overline{q}}}$. Mais $g \leq_T r^{\,i}_{p_{\setminus\overline{q}}}$ est en contradiction avec le fait que $g \not\leq_{p_{\setminus\overline{q}}} r(p_{\setminus\overline{q}}{}^i) = lca_{p_{\setminus\overline{q}}}(F(p_{\setminus\overline{q}}{}^i))$ et que les morphismes $\theta_{p_{\setminus\overline{q}}}$ et $\theta_{T,p_{\setminus\overline{q}}}$ garantissent la conservation de structure entre $p_{\setminus\overline{q}}$ et $T = SuperMast_t(p_{\setminus\overline{q}}, q_{\setminus\overline{p}})$ (remarque 2.3).

On montre ensuite que les conditions (7),(8),(9) sont vérifiées par T pour les arbres $p_{\setminus\overline{q}}{}^i$ et q :

- par définition de T on a $F(T) \subseteq F(p_{\setminus\overline{q}}) \cup F(q_{\setminus\overline{p}})$ ce qui associé à (24) donne $F(T) \subseteq F(p_{\setminus\overline{q}}{}^i) \cup F(q_{\setminus\overline{p}})$, ie que la condition (9) est vérifiée pour que T soit un super-arbre d'accord de $p_{\setminus\overline{q}}{}^i$ et $q_{\setminus\overline{p}}$.
- Par définition de T on a $T|F(p_{\setminus\overline{q}}) \subseteq_h p_{\setminus\overline{q}}$ donc en se restreignant aux feuilles de $p_{\setminus\overline{q}}{}^i$ on a

$$(T|F(p_{\setminus\overline{q}}))|F(p_{\setminus\overline{q}}{}^i) = T|F(p_{\setminus\overline{q}}{}^i) \subseteq_h p_{\setminus\overline{q}}|F(p_{\setminus\overline{q}}{}^i) = p_{\setminus\overline{q}}{}^i$$

et comme on sait par définition de T que $T|F(q_{\setminus\overline{p}}) \subseteq_h q_{\setminus\overline{p}}$, la condition (7) est remplie pour que T soit un super-arbre d'accord de ${p_{\setminus\overline{q}}}^i$ et $q_{\setminus\overline{p}}$.

– Par définition de T, la condition (8) est vérifiée pour $q_{\setminus\overline{p}}$ et pour $p_{\setminus\overline{q}}$. Puisque par hypothèse $F({p_{\setminus\overline{q}}}^i) \cap F(q) \neq \emptyset$, on a $F(p) \cap F(q) \neq \emptyset$ et par le lemme 15, on en déduit $F(T) \cap F(p) \neq \emptyset$. Donc par définition de T, la condition (8) indique que $F(T) \cap (F(p_{\setminus\overline{q}}) - I\!F(p_{\setminus\overline{q}})) \neq \emptyset$. Avec (24) on en conclut que $F(T) \cap (F({p_{\setminus\overline{q}}}^i) - I\!F({p_{\setminus\overline{q}}}^i)) \neq \emptyset$, ce qui montre que la condition (8) est respectée pour ${p_{\setminus\overline{q}}}^i$ (elle l'est aussi pour $q_{\setminus\overline{p}}$ par définition de T).

On en déduit que T est un super-arbre d'accord de ${p_{\setminus\overline{q}}}^i$ et $q_{\setminus\overline{p}}$. Par définition de $SuperMast$ on en déduit que $|T| \leq SuperMast({p_{\setminus\overline{q}}}^i, q_{\setminus\overline{p}})$. □

Le lemme suivant montre que si la racine des arbres T et $p_{\setminus\overline{q}}$ se correspondent alors il y a une correspondance univoque entre sous-arbres fils de T et de $p_{\setminus\overline{q}}$: tout sous-arbre fils de $p_{\setminus\overline{q}}$, resp. T, correspond à au plus un sous-arbre fils de l'autre arbre[7].

LEMME 19
Supposons $F(p) \cap F(q) \neq \emptyset$.
Si $r_{p_{\setminus\overline{q}}} = r$ *alors*

(i) si $r_{p_{\setminus\overline{q}}} \neq {r_{p_{\setminus\overline{q}}}}^i$ *et* $F(T^x) \cap F({p_{\setminus\overline{q}}}^i) \neq \emptyset$ *alors* $\forall T^y \neq T^x$ *on a* $F(T^y) \cap F({p_{\setminus\overline{q}}}^i) = \emptyset$
(ii) si $F(T^x) \cap F({p_{\setminus\overline{q}}}^i) \neq \emptyset$ *alors* $\forall {p_{\setminus\overline{q}}}^{i'} \neq {p_{\setminus\overline{q}}}^i$ *on a* $F(T^x) \cap F({p_{\setminus\overline{q}}}^{i'}) = \emptyset$
(iii) si $F(T^x) \cap F(p_{\setminus\overline{q}}) \neq \emptyset$ *alors il existe* ${p_{\setminus\overline{q}}}^i$ *tel que* $T^x|F(p_{\setminus\overline{q}}) \subseteq_h {p_{\setminus\overline{q}}}^i$

où T^x, T^y, *resp.* ${p_{\setminus\overline{q}}}^i, {p_{\setminus\overline{q}}}^{i'}$, *sont des sous-arbres fils de* T, *resp.* $p_{\setminus\overline{q}}$.

PREUVE :
La preuve des deux premiers points repose sur la remarque 2.3 et la contrainte (2).

PREUVE DE (i) : Supposons $\exists f \in F(T^x) \cap F({p_{\setminus\overline{q}}}^i)$. Du fait que r est la racine de T et que $r = r_{p_{\setminus\overline{q}}} \neq {r_{p_{\setminus\overline{q}}}}^i$ on déduit

$$ {r_{p_{\setminus\overline{q}}}}^i <_T r \tag{25} $$

Alors pour tout $T^y \neq T^x$ il ne peut exister de feuille $f' \in F(T^y) \cap F({p_{\setminus\overline{q}}}^i)$ car sinon $r = lca_T(f, f')$ et donc par définition de ${r_{p_{\setminus\overline{q}}}}^i$, on aboutit à $r \leq_T {r_{p_{\setminus\overline{q}}}}^i$, une contradiction avec (25).

PREUVE DE (ii) : nous montrons qu'on ne peut pas avoir à la fois $r = r_{p_{\setminus\overline{q}}}$, $F(T^x) \cap F({p_{\setminus\overline{q}}}^i) \neq \emptyset$ et $F(T^x) \cap F({p_{\setminus\overline{q}}}^{i'}) \neq \emptyset$, donc que si les deux premières conditions sont remplies alors la 3ème ne peut être vérifiée.

Si $F(T^x) \cap F({p_{\setminus\overline{q}}}^i) \neq \emptyset$ et $F(T^x) \cap F({p_{\setminus\overline{q}}}^{i'}) \neq \emptyset$ alors il existe deux feuilles $f \in F(T^x) \cap F({p_{\setminus\overline{q}}}^i), f' \in F(T^x) \cap F({p_{\setminus\overline{q}}}^{i'})$. Comme $r = r_{p_{\setminus\overline{q}}}$, il existe aussi une feuille $g \in F(T^y) \cap F(p_{\setminus\overline{q}})$ pour un sous-arbre fils $T^y \neq T^x$ de T.

La situation de ces trois feuilles dans T est telle que

$$ lca_T(f, f') \leq_T r(T^x) <_T r = lca_T(f, g) \tag{26} $$

où $r(T^x)$ est la racine de T^x. Comme f, f', g appartiennent à T et $p_{\setminus\overline{q}}$, elles sont aussi dans $T|F(p_{\setminus\overline{q}})$ et comme nous avons $T|F(p_{\setminus\overline{q}}) \subseteq_h T$ (par l'homéomorphisme $\theta_{T,p_{\setminus\overline{q}}}$), de (2) et (26) nous déduisons

$$ \theta_{T,p_{\setminus\overline{q}}}(lca_{T|F(p_{\setminus\overline{q}})}(f, f')) <_T \theta_{T,p_{\setminus\overline{q}}}(lca_{T|F(p_{\setminus\overline{q}})}(f, g)) $$

[7] si un sous-arbre fils de $p_{\setminus\overline{q}}$ ne correspond à aucun sous-arbre fils de T c'est qu'aucune de ses feuilles n'a été retenue dans le super-arbre d'accord ; si un sous-arbre fils de T ne correspond à aucun sous-arbre fils de $p_{\setminus\overline{q}}$, c'est qu'il est issu de l'arbre $q_{\setminus\overline{p}}$.

et en appliquant l'équation (3),

$$lca_{T|F(p_{\setminus\overline{q}})}(f,f') <_{T|F(p_{\setminus\overline{q}})} lca_{T|F(p_{\setminus\overline{q}})}(f,g) \ . \tag{27}$$

Dans l'arbre $p_{\setminus\overline{q}}$ de racine $r(p_{\setminus\overline{q}})$, comme $f \in F({p_{\setminus\overline{q}}}^i)$ et $f' \in F({p_{\setminus\overline{q}}}^{i'})$ on sait $lca_{p_{\setminus\overline{q}}}(f,f') = r(p_{\setminus\overline{q}})$. Ce noeud étant la racine de l'arbre on a forcément

$$lca_{p_{\setminus\overline{q}}}(f,g) \leq_{p_{\setminus\overline{q}}} lca_{p_{\setminus\overline{q}}}(f,g)$$

Comme $T|F(p_{\setminus\overline{q}}) \subseteq_h p_{\setminus\overline{q}}$ (par définition de T), comme ci-dessus on peut appliquer les équations (2) et (3) cette fois sur l'homéorphisme $\theta_{p_{\setminus\overline{q}}}$ et obtenir

$$lca_{T|F(p_{\setminus\overline{q}})}(f,g) \leq_{T|F(p_{\setminus\overline{q}})} lca_{T|F(p_{\setminus\overline{q}})}(f,f')$$

une contradiction avec (27).

PREUVE DE (iii) : Si $F(T^x) \cap F(p_{\setminus\overline{q}}) \neq \emptyset$ alors $\exists\, {p_{\setminus\overline{q}}}^i$ sous-arbre fils de $p_{\setminus\overline{q}}$ t.q. $F(T^x) \cap F({p_{\setminus\overline{q}}}^i) \neq \emptyset$.

Par ailleurs, $T = SuperMast_t(p_{\setminus\overline{q}}, q_{\setminus\overline{p}})$ donc $T|F(p_{\setminus\overline{q}}) \subseteq_h p_{\setminus\overline{q}}$. On en déduit $T|F({p_{\setminus\overline{q}}}^i) \subseteq_h p_{\setminus\overline{q}}|F({p_{\setminus\overline{q}}}^i)$ puisque $F({p_{\setminus\overline{q}}}^i) \subset F(p_{\setminus\overline{q}})$. Or $p_{\setminus\overline{q}}|F({p_{\setminus\overline{q}}}^i) = {p_{\setminus\overline{q}}}^i$, donc on a

	$T\|F({p_{\setminus\overline{q}}}^i) \subseteq_h {p_{\setminus\overline{q}}}^i$	
et	$T^x\|F({p_{\setminus\overline{q}}}^i) \subseteq_h T\|F({p_{\setminus\overline{q}}}^i)$	car T^x est sous-arbre fils de T
donc	$T^x\|F({p_{\setminus\overline{q}}}^i) \subseteq_h {p_{\setminus\overline{q}}}^i$	par transitivité de la relation $\subseteq_h$ (remarque 2.2)

L'assertion (ii) prouvée ci-dessus combinée à $F(T^x) \cap F({p_{\setminus\overline{q}}}^i) \neq \emptyset$ implique que $F(T^x) \cap (F(p_{\setminus\overline{q}}) - F({p_{\setminus\overline{q}}}^i)) = \emptyset$, ie que $T^x|F({p_{\setminus\overline{q}}}^i) = T^x|F(p_{\setminus\overline{q}})$ et donc que

$$T^x|F(p_{\setminus\overline{q}}) \subseteq_h {p_{\setminus\overline{q}}}^i \ ,$$

le résultat souhaité. □

Le lemme précédent s'applique bien entendu aussi aux sous-arbres de Q et T dans le cas où $r_{q_{\setminus\overline{p}}} = r$ (les définitions et preuves ne supposent pas d'ordre particulier dans la collection $\mathcal{T} = \{P, Q\}$).

LEMME 20
Supposons $F(p) \cap F(q) \neq \emptyset$

(i) Si $r_{q_{\setminus\overline{p}}} \in T^x$ et ${r_{p_{\setminus\overline{q}}}}^i \neq r_{p_{\setminus\overline{q}}}$, alors $T^x = SuperMast_t({p_{\setminus\overline{q}}}^i, q_{\setminus\overline{p}})$.

(ii) Si $r_{p_{\setminus\overline{q}}} \in T^x$ et $r_{{q_{\setminus\overline{p}}}^j} \neq r_{q_{\setminus\overline{p}}}$ alors $T^x = SuperMast_t(p_{\setminus\overline{q}}, {q_{\setminus\overline{p}}}^j)$.

où T^x, resp. ${p_{\setminus\overline{q}}}^i$ et ${q_{\setminus\overline{p}}}^j$, est un sous-arbre fils de T, resp. de $p_{\setminus\overline{q}}$ et de $q_{\setminus\overline{p}}$.

Par exemple, sur la figure 14, le sous-arbre fils de r nommé T^x contient $r_{q_{\setminus\overline{p}}}$ et on a $T^x = SuperMast_t({p_{\setminus\overline{q}}}^i, q_{\setminus\overline{p}})$.
PREUVE :
(i) Notons d'abord que $r = r_{p_{\setminus\overline{q}}}$ en raison de $r_{q_{\setminus\overline{p}}} \in T^x$ et du lemme 17. Comme noté précédemment (lemme 15), $F(p) \cap F(q) \neq \emptyset$ implique $T \neq \emptyset$, et par l'axiome (8) de la définition 3.1 appliquée à $T = SuperMast_t(p_{\setminus\overline{q}}, q_{\setminus\overline{p}})$ on sait $\exists f \in F(T) \cap F(p) \cap F(q_{\setminus\overline{p}})$. Comme $r_{q_{\setminus\overline{p}}} \in T^x$, on en déduit

$$\exists f \in F(T^x) \cap F(p_{\setminus\overline{q}}) \cap F(q_{\setminus\overline{p}}) \tag{28}$$

et puisque $r = r_{p_{\setminus\overline{q}}}$, le lemme 19 (iii) indique que $\exists\, {p_{\setminus\overline{q}}}^i$ sous-arbre fils de $p_{\setminus\overline{q}}$ t.q. $T^x|F(p_{\setminus\overline{q}}) \subseteq_h {p_{\setminus\overline{q}}}^i$, donc $T^x|F({p_{\setminus\overline{q}}}^i) \subseteq_h {p_{\setminus\overline{q}}}^i$. Par ailleurs on a aussi $T^x|F(q_{\setminus\overline{p}}) \subseteq_h q_{\setminus\overline{p}}$ par définition de T et par

$r_{q_{\setminus\overline{p}}} \in T^x$. L'arbre T^x vérifie ainsi l'axiome (7) de la définition d'un super-arbre d'accord de ${p_{\setminus\overline{q}}}^i$ et $q_{\setminus\overline{p}}$.

L'axiome (8) de cette définition est lui aussi vérifié puisque $\exists\, f \in F(T^x) \cap F({p_{\setminus\overline{q}}}^i) \cap F(q_{\setminus\overline{p}})$ en raison de (28) et de $T^x|F(p_{\setminus\overline{q}}) \subseteq_h {p_{\setminus\overline{q}}}^i$.

L'axiome (9) est aussi vérifié : par définition, $F(T^x) \subset F(T) \subseteq F(p_{\setminus\overline{q}}) \cup F(q_{\setminus\overline{p}})$, et comme $\exists\, f \in F(T^x) \cap F({p_{\setminus\overline{q}}}^i) \cap F(q_{\setminus\overline{p}})$ le point (ii) du lemme 19 permet de conclure que $F(T^x) \subseteq F({p_{\setminus\overline{q}}}^i) \cup F(q_{\setminus\overline{p}})$.

On conclut ainsi que T^x est un super-arbre d'accord de $q_{\setminus\overline{p}}$ et ${p_{\setminus\overline{q}}}^i$.

Pour montrer qu'il est *maximum*, il faut nous servir de $r_{p_{\setminus\overline{q}}} \neq r_{{p_{\setminus\overline{q}}}^i}$: si T^x n'était pas maximum alors T ne serait pas maximum pour $p_{\setminus\overline{q}}$ et $q_{\setminus\overline{p}}$. En effet, s'il existe un arbre $T' = SuperMast_t({p_{\setminus\overline{q}}}^i, q_{\setminus\overline{p}})$ avec $|T'| > |T^x|$ alors on considère l'arbre T'' obtenu depuis T en remplaçant T^x par T'. Ce remplacement n'amène pas d'étiquettes déjà présentes dans T en dehors de T^x puisque

$$\begin{array}{ll} & F(T') \subseteq F(q_{\setminus\overline{p}}) \cup F({p_{\setminus\overline{q}}}^i) \quad \text{par définition de } T' \text{ ,} \\ & (F(T) - F(T^x)) \cap F({p_{\setminus\overline{q}}}^i) = \emptyset \quad \text{puisque } {r_{p_{\setminus\overline{q}}}}^i <_T r(T^x) \text{ (par } r_{p_{\setminus\overline{q}}} \neq {r_{p_{\setminus\overline{q}}}}^i \text{ et } F(T^x) \cap F({p_{\setminus\overline{q}}}^i) \neq \emptyset) \\ \text{et} & (F(T) - F(T^x)) \cap F(q_{\setminus\overline{p}}) = \emptyset \quad \text{puisque } r_{q_{\setminus\overline{p}}} \leq_T r(T^x) \text{ .} \end{array}$$

On a $|T''| > |T|$ et on peut aisément montrer que T'' est un super-arbre d'accord de $p_{\setminus\overline{q}}$ et $q_{\setminus\overline{p}}$, ce qui contredit $T = SuperMast_t(p_{\setminus\overline{q}}, q_{\setminus\overline{p}})$. Ainsi il n'existe pas de super-arbre d'accord de $q_{\setminus\overline{p}}$ et ${p_{\setminus\overline{q}}}^i$ de taille strictement plus grande que $|T^x|$, donc $T^x = SuperMast_t({p_{\setminus\overline{q}}}^i, q_{\setminus\overline{p}})$.

(ii) La preuve de ce point est symétrique. □

Le lemme suivant sera utile pour pouvoir appliquer l'induction dans la preuve du théorème 22.

LEMME 21
Soit p^a un sous-arbre fils de p.
Si T est un super-arbre d'accord de ${p^a}_{\setminus\overline{q}}$ et $q_{\setminus\overline{p}}$, et $F(T) \cap (F(p) - F(p^a)) = \emptyset$ alors
- *T est un super-arbre d'accord de ${p^a}_{\setminus\overline{q}}$ et $q_{\setminus\overline{p^a}}$,*
- *et donc $|T| \leq SuperMast({p^a}_{\setminus\overline{q}}, q_{\setminus\overline{p^a}})$.*

PREUVE :
Il suffit de montrer que T vérifie les trois axiomes d'un super-arbre d'accord de ${p^a}_{\setminus\overline{q}}$ et $q_{\setminus\overline{p^a}}$:
- Par définition de T (condition (9)) on a

$$F(T) \subseteq F({p^a}_{\setminus\overline{q}}) \cup F(q_{\setminus\overline{p}}) \text{ ,}$$

$$\text{donc } F(T) \subseteq F(p^a) \cup F(q) \cup F(q) \cap F(p)$$

  mais par hypothèse, $F(T) \cap F(p^{a'}) = \emptyset$ pour tout sous-arbre fils $p^{a'}$ de p autre que p^a, donc

$$F(T) \subseteq F(p^a) \cup F(q) \cup F(p^a) \cap F(q) \text{ ,}$$

$$\text{i.e., } F(T) \subseteq F({p^a}_{\setminus\overline{q}}) \cup F(q_{\setminus\overline{p^a}})$$

  ce qui montre que T vérifie (9) pour les arbres sources ${p^a}_{\setminus\overline{q}}$ et $q_{\setminus\overline{p^a}}$;
- Par définition de T (condition (8)) on sait

$$\begin{array}{rl} & \exists f \in F(T) \cap F({p^a}_{\setminus\overline{q}}) \cap F(q_{\setminus\overline{p}}) \\ \Rightarrow & \exists f \in F(T) \cap F(p^a) \cap F(q) \\ \Rightarrow & \exists f \in F(T) \cap F({p^a}_{\setminus\overline{q}}) \cap F(q_{\setminus\overline{p^a}}) \end{array}$$

  ce qui montre que T vérifie la condition (8) pour ${p^a}_{\setminus\overline{q}}$ et $q_{\setminus\overline{p^a}}$;

– Par définition de T (condition (7)), on sait

$$\begin{array}{rcl} T|F(q_{\setminus\overline{p}}) & \subseteq_h & q_{\setminus\overline{p}} \\ \text{donc } \big(T|F(q_{\setminus\overline{p}})\big)|F(q_{\setminus\overline{p^a}}) & \subseteq_h & q_{\setminus\overline{p}}|F(q_{\setminus\overline{p^a}}) \end{array}$$

or $q_{\setminus\overline{p}}|F(q_{\setminus\overline{p^a}}) = q_{\setminus\overline{p^a}}$, et $\big(T|F(q_{\setminus\overline{p}})\big)|F(q_{\setminus\overline{p^a}}) = T|F(q_{\setminus\overline{p^a}})$ (puisque $F(T) \cap (F(p) - F(p^a)) = \emptyset$), donc

$$T|F(q_{\setminus\overline{p^a}}) \quad \subseteq_h \quad q_{\setminus\overline{p^a}}\,.$$

Par ailleurs la définition de T (condition (7)) induit que $T|F(p^a{}_{\setminus\overline{q}}) \subseteq_h p^a{}_{\setminus\overline{q}}$. Ces deux points montrent que T vérifie la condition (7) pour $p^a{}_{\setminus\overline{q}}$ et $q_{\setminus\overline{p^a}}$.

□

THEOREME 22 (RELATION ENTRE SMAST ET SUPERMAST)
Soient P et Q deux arbres enracinés, et soient p et q deux de leurs sous-arbres respectifs, alors

$$SMAST(p,q) = SuperMast(p_{\setminus\overline{q}}, q_{\setminus\overline{p}})$$

Autrement dit, $SMAST(p,q)$ est la taille d'un arbre d'accord maximum de p et q si l'on fait abstraction des feuilles de p apparaissant en dehors de q dans Q et vice-versa.

PREUVE :
La démarche est similaire à celle de [48] mais nous raisonnons dans le cadre de T et non dans celui de p et q (à la différence de *MAST*, T lui-même n'est pas inclus homéomorphiquement dans p ou q, seule sa restriction à $F(p)$, resp. $F(q)$ l'est). Nous détaillons avant tout le cas particulier où $F(p) \cap F(q) = \emptyset$. Dans le cas général, où $F(p) \cap F(q) \neq \emptyset$, la preuve se fait par induction sur $l = |F(p)| + |F(q)|$ et nous détaillons dans l'ordre :
– le cas trivial de l'induction : $l = 2$ en raison de $|F(p)| = |F(q)| = 1$;
– le cas où $l > 2$ mais $min(|F(p)|, |F(q)|) = 1$
– le cas le plus général où $l > 2$ et $|F(p)| > 1, |F(q)| > 1$.

• Si $F(p) \cap F(q) = \emptyset$, alors $F(p_{\setminus\overline{q}}) \cap F(q_{\setminus\overline{p}}) = \emptyset$ et les conditions (8) et (9) de la définition 3.1 d'un super-arbre d'accord maximum font que $T = SuperMast_t(p_{\setminus\overline{q}}, q_{\setminus\overline{p}})$ ne peut contenir aucune feuille, donc $SuperMast(p_{\setminus\overline{q}}, q_{\setminus\overline{p}}) = 0$. Par ailleurs, si $F(p) \cap F(q) = \emptyset$, l'équation (18) conclut aussi $SMAST(p,q) = 0$.

Nous utilisons maintenant l'induction pour le cas $F(p) \cap F(q) \neq \emptyset$.

• Dans le cas trivial où $l = 2$, on a $p = q = \{x\}$ et $SMAST(p,q) = 1$ (19) (puisque $\not\!F(p) = \not\!F(q) = \emptyset$). On a $p_{\setminus\overline{q}} = q_{\setminus\overline{p}} = \{x\}$ et l'arbre-feuille $T = \{x\}$ vérifie bien les conditions (7)-(9) d'un super-arbre d'accord de $p_{\setminus\overline{q}}$ et $q_{\setminus\overline{p}}$. Il ne peut y avoir de super-arbre d'accord plus grand en raison de l'axiome (9), donc $T = SuperMast_t(p_{\setminus\overline{q}}, q_{\setminus\overline{p}})$ et $|T| = 1 = SMAST(p,q)$.

• Considérons maintenant le cas où $l > 2$ et $min(|F(p)|, |F(q)|) = 1$. Supposons sans perte de généralité que $F(p) = \{x\}$ et $|F(q)| > 1$. Puisque $F(p) \cap F(q) \neq \emptyset$, on a $x \in F(q)$. Par l'équation

(19), $SMAST(p,q) = |\not\!F(q)| + 1$. Par leur définition, il est facile de voir qu'ici $p_{\setminus\overline{q}} = \{x\}$ et $q_{\setminus\overline{p}} = \not\!F(q) \cup \{x\}$. La proposition 13 implique donc que $SuperMast_t(p_{\setminus\overline{q}}, q_{\setminus\overline{p}}) = q_{\setminus\overline{p}}$ et ainsi

$$SuperMast(p_{\setminus\overline{q}}, q_{\setminus\overline{p}}) = |F(q_{\setminus\overline{p}})| = |\not\!F(q) \cup \{x\}| = |\not\!F(q)| + 1 = SMAST(p,q)\ .$$

Le cas où $F(q) = \{x\}$ et $|F(p)| > 1$ se règle de façon symétrique.

• La suite de la preuve s'applique aux cas où $|F(p)| > 1, |F(q)| > 1$ (et donc $l > 2$).

Dans une première étape nous allons montrer que $SuperMast(p_{\setminus\overline{q}}, q_{\setminus\overline{p}}) \leq SMAST(p,q)$.

Le cas le plus simple est quand $\exists p^a$ sous-arbre fils de p t.q. $F(p_{\setminus\overline{q}}) \subseteq F(p^a)$:

$SuperMast(p_{\setminus\overline{q}}, q_{\setminus\overline{p}})$	$=$	$SuperMast({p^a}_{\setminus\overline{q}}, q_{\setminus\overline{p}})$	puisque $p_{\setminus\overline{q}} = {p^a}_{\setminus\overline{q}}$ par le lemme 16 *(i)*
	$=$	$SuperMast({p^a}_{\setminus\overline{q}}, q_{\setminus\overline{p^a}})$	puisque $q_{\setminus\overline{p}} = q_{\setminus\overline{p^a}}$ car $F(p) \cap F(q) = F(p^a) \cap F(q)$, résultant de $p_{\setminus\overline{q}} = {p^a}_{\setminus\overline{q}}$ (lemme 16 *(i)*)
	$=$	$SMAST(p^a, q)$	par induction puisque $\|p^a\| < \|p\|$
	$\leq$	$SMAST(p,q)$	par (21)

Le cas symétrique où $\exists q^b$ sous-arbre fils de q t.q. $F(q_{\setminus\overline{p}}) \subseteq F(q^b)$ aboutit de la même façon à $SuperMast(p_{\setminus\overline{q}}, q_{\setminus\overline{p}}) \leq SMAST(p,q)$. La suite de la preuve s'applique aux cas restants, ie $\not\exists p^a$ sous-arbre fils de p et q^b sous-arbre fils de q t.q. $F(p_{\setminus\overline{q}}) \subseteq F(p^a)$ ou $F(q_{\setminus\overline{p}}) \subseteq F(q^b)$.

Le premier cas à traiter est celui où les feuilles de $p_{\setminus\overline{q}}$ (ou de $q_{\setminus\overline{p}}$) conservées dans T sont issues d'un seul de ses sous-arbre fils. Supposons ainsi que pour un sous-arbre fils ${p_{\setminus\overline{q}}}^i$ de $p_{\setminus\overline{q}}$ on ait ${r_{p_{\setminus\overline{q}}}}^i = r_{p_{\setminus\overline{q}}}$, alors :

1. T est super-arbre d'accord de ${p_{\setminus\overline{q}}}^i$ et $q_{\setminus\overline{p}}$ (par le lemme 18) ;
2. donc T est super-arbre d'accord de ${p^a}_{\setminus\overline{q}}$ et $q_{\setminus\overline{p}}$ pour p^a sous-arbre fils de p (par le lemme 16*(ii)*) ;
3. De plus, ${r_{p_{\setminus\overline{q}}}}^i = r_{p_{\setminus\overline{q}}}$ induit $F(T) \cap F({p_{\setminus\overline{q}}}^{i'}) = \emptyset$ pour tout sous-arbre fils ${p_{\setminus\overline{q}}}^{i'}$ de $p_{\setminus\overline{q}}$ autre que ${p_{\setminus\overline{q}}}^i$ (comme déjà montré en (24)). En ajoutant que par définition $T|F(p_{\setminus\overline{q}}) \subseteq_h p_{\setminus\overline{q}}$, on déduit que $F(T) \cap (F(p) - F(p^a)) = \emptyset$ (où p^a est le sous-arbre désigné au point 2 ci-dessus).

Des points 2 et 3 ci-dessus, on déduit par le lemme 21 que

$\|T\|$	$\leq$	$SuperMast({p^a}_{\setminus\overline{q}}, q_{\setminus\overline{p^a}})$	
	$\leq$	$SMAST(p^a, q)$	par induction
	$\leq$	$SMAST(p,q)$	en raison de (21)

Si il existe un sous-abre fils ${q_{\setminus\overline{p}}}^j$ de q t.q. ${r_{q_{\setminus\overline{p}}}}^j = r_{q_{\setminus\overline{p}}}$, la preuve suit le même raisonnement.

Dans la suite nous supposons maintenant que ${r_{p_{\setminus\overline{q}}}}^i \neq r_{p_{\setminus\overline{q}}}$ et ${r_{q_{\setminus\overline{p}}}}^j \neq r_{q_{\setminus\overline{p}}}$ pour tout sous-arbre fils ${p_{\setminus\overline{q}}}^i$, resp. ${q_{\setminus\overline{p}}}^j$, de p, resp. q. D'après le lemme 17, il y a trois configurations possibles pour les noeuds $r, r_{p_{\setminus\overline{q}}}, r_{q_{\setminus\overline{p}}}$ dans l'arbre $T = SuperMast_t(p_{\setminus\overline{q}}, q_{\setminus\overline{p}})$:

($\overrightarrow{A}$) $r = r_{p_{\setminus\overline{q}}}$ et $r_{q_{\setminus\overline{p}}}$ est dans un sous-arbre fils T^x de T

($\overrightarrow{B}$) $r = r_{q_{\setminus\overline{p}}}$ et $r_{p_{\setminus\overline{q}}}$ est dans un sous-arbre fils T^x de T

($\overrightarrow{C}$) $r = r_{p_{\setminus\overline{q}}} = r_{q_{\setminus\overline{p}}}$

Les deux premiers cas correspondent à l'appariement d'un arbre source à un sous-arbre fils de l'autre arbre source, et le troisième cas correspond à un appariement entre sous-arbres fils des deux arbres sources.

($\overrightarrow{A}$) $\boxed{r = r_{p_{\setminus\overline{q}}} \neq r_{q_{\setminus\overline{p}}}}$

Si $r_{q_{\setminus\overline{p}}} \in T^x$, le lemme 20 *(i)* dit qu'il existe T^x un sous-arbre fils de $T = SuperMast_t(p_{\setminus\overline{q}}, q_{\setminus\overline{p}})$ t.q. $T^x = SuperMast_t({p_{\setminus\overline{q}}}^i, q_{\setminus\overline{p}})$ pour ${p_{\setminus\overline{q}}}^i$ un sous-arbre fils de $p_{\setminus\overline{q}}$. On a $F(T^x) \cap F({p_{\setminus\overline{q}}}^i) \neq \emptyset$ (car sinon $F(T^x) \subseteq I\!F(q)$ et avec $r_{q_{\setminus\overline{p}}} \in T^x$ on déduirait $F(p) \cap F(q) \cap F(T) = \emptyset$, une contradiction avec l'axiome (9) pour T). Donc en appliquant le lemme 19(*ii*), on sait

$$\forall {p_{\setminus\overline{q}}}^{i'} \neq {p_{\setminus\overline{q}}}^i \text{ sous-arbre fils de } p_{\setminus\overline{q}} \text{ on a } F(T^x) \cap F({p_{\setminus\overline{q}}}^{i'}) = \emptyset \ . \tag{29}$$

En appliquant le lemme 16 *(ii)* (i.e., ${p_{\setminus\overline{q}}}^i = {p^a}_{\setminus\overline{q}}$), on déduit $T^x = SuperMast_t({p^a}_{\setminus\overline{q}}, q_{\setminus\overline{p}})$ et avec (29) et $T|F(p_{\setminus\overline{q}}) \subseteq_h p_{\setminus\overline{q}}$, on déduit $F(T^x) \cap (F(p) - F(p^a)) = \emptyset$. Donc on peut appliquer le lemme 21 à T^x pour déduire que

$$|T^x| \leq SuperMast({p^a}_{\setminus\overline{q}}, q_{\setminus\overline{p^a}}) \ . \tag{30}$$

En ce qui concerne l'ensemble F' des feuilles de T apparaissant en dehors de T^x, on a

$$\begin{aligned} F' = F(T) - F(T^x) &\subseteq F(p_{\setminus\overline{q}}) \cup F(q_{\setminus\overline{p}}) - F(q_{\setminus\overline{p}}) - F({p_{\setminus\overline{q}}}^i) \\ &\subseteq I\!F_{{p_{\setminus\overline{q}}}^i}(p_{\setminus\overline{q}}) \end{aligned} \tag{31}$$

La première équation résulte de la définition de T (union des deux premiers termes), du fait que $r_{q_{\setminus\overline{p}}} \in T^x$ (premier terme soustrait) et du fait que $T^x = SuperMast_t({p_{\setminus\overline{q}}}^i, q_{\setminus\overline{p}})$ associé au point (i) du lemme 19 (deuxième terme soustrait). La deuxième équation résulte de ce qui reste de $F(p_{\setminus\overline{q}})$ quand on a enlevé les termes indiqués (il ne reste rien de $F(q_{\setminus\overline{p}}) \cap F(p_{\setminus\overline{q}})$ puisque rien de $F(q_{\setminus\overline{p}})$).

Comme remarqué ci-dessus ${p_{\setminus\overline{q}}}^i = {p^a}_{\setminus\overline{q}}$ pour p^a un sous-arbre fils de p. On a ainsi $I\!F_{{p_{\setminus\overline{q}}}^i}(p_{\setminus\overline{q}}) = I\!F_{{p^a}_{\setminus\overline{q}}}(p) = I\!F_{\setminus p^a}(p)$ donc avec (31) on déduit

$$|F'| \leq |I\!F_{\setminus p^a}(p)| \ , \tag{32}$$

ce qui permet de conclure :

$$\begin{array}{rcll} SuperMast(p_{\setminus\overline{q}}, q_{\setminus\overline{p}}) & = & |T| & \text{par définition de } T, \\ & = & |T^x| + |F'| & \text{par définition de } T^x \text{ et } F', \\ & \leq & SuperMast({p^a}_{\setminus\overline{q}}, q_{\setminus\overline{p^a}}) + |I\!F_{\setminus p^a}(p)| & \text{par (30) et (32)} \\ & \leq & SMAST(p^a, q) + |I\!F_{\setminus p^a}(p)| & \text{par induction} \\ & \leq & SMAST(p, q) & \text{en raison de (21).} \end{array}$$

($\overrightarrow{B}$) $\boxed{r = r_{q_{\setminus\overline{p}}} \neq r_{p_{\setminus\overline{q}}}}$

On obtient le résultat de façon symétrique au cas ($\overrightarrow{A}$).

($\overrightarrow{C}$) $\boxed{r = r_{p_{\setminus\overline{q}}} = r_{q_{\setminus\overline{p}}}}$

Ce cas correspond au recours à un couplage du graphe biparti $G(p, q)$ dans l'obtention de la valeur $SMAST$. Dans ce graphe, les sommets de la première partie correspondent aux sous-arbres fils p^a de p et aux alter-ego v_b des sous-arbres fils q^b de q, et les sommets de la deuxième partie correspondent aux sous-arbres fils q^b de q et aux alter-ego v'_a des sous-arbres fils de p. Nous allons construire un couplage C_T de valeur w_T en passant en revue les sous-arbres fils T^x de T et en ajoutant pour chacun une arête à C_T. Chaque T^x possède au moins une feuille en commun avec $p_{\setminus\overline{q}}$ ou $q_{\setminus\overline{p}}$ (car sinon T ne respecte pas l'axiome (9)) et la remarque 7.1 indique que T peut être choisi t.q. tout T^x vérifie la condition (23). Pour chaque T^x, il y a trois situations possibles :

1. $\exists i$ t.q. $F(T^x) \cap F(p_{\setminus \overline{q}}{}^i) \neq \emptyset$ et $F(T^x) \cap F(q_{\setminus \overline{p}}) = \emptyset$ (i.e., $\forall j, F(T^x) \cap F(q_{\setminus \overline{p}}{}^j) = \emptyset$) : on ajoute à C_T l'arête (p^a, v'_a) de valeur $|\mathbb{F}(p^a)|$ où p^a est le sous-arbre fils de p correspondant à $p_{\setminus \overline{q}}{}^i$ (i.e., t.q. $p^a{}_{\setminus \overline{q}} = p_{\setminus \overline{q}}{}^i$ dans le cas *(ii)* du lemme 16).
2. $\exists j$ t.q. $F(T^x) \cap F(q_{\setminus \overline{p}}{}^j) \neq \emptyset$ et $F(T^x) \cap F(p_{\setminus \overline{q}}) = \emptyset$ (i.e., $\forall i, F(T^x) \cap F(p_{\setminus \overline{q}}{}^i) = \emptyset$) : on ajoute à C_T l'arête (q^b, v_b) de valeur $|\mathbb{F}(q^b)|$ où q^b est le sous-arbre fils de q correspondant à $q_{\setminus \overline{p}}{}^j$ (lemme 16 *(ii)*).
3. $\exists i, j$ t.q. $F(T^x) \cap F(p_{\setminus \overline{q}}{}^i) \neq \emptyset$ et $F(T^x) \cap F(q_{\setminus \overline{p}}{}^j) \neq \emptyset$: on ajoute à C_T l'arête (p^a, q^b) de valeur $SMAST(p^a, q^b)$ où p^a et q^b sont les sous-arbres fils de p, resp. q correspondant à $p_{\setminus \overline{q}}{}^i$, resp. $q_{\setminus \overline{p}}{}^j$ (lemme 16 *(ii)*).

Le couplage C_T est un couplage correct de $G(p, q)$ (i.e., tout sommet du graphe ne participe au maximum qu'à une arête de C_T) :

- tout sous-arbre T^x de T correspond à au plus un sous-arbre $p_{\setminus \overline{q}}{}^i$ de $p_{\setminus \overline{q}}$ et au plus un sous-arbre $q_{\setminus \overline{p}}{}^j$ de $q_{\setminus \overline{p}}$ (lemme 19 (*ii*))
- tout sous-arbre $p_{\setminus \overline{q}}{}^i$ de $p_{\setminus \overline{q}}$ (resp. $q_{\setminus \overline{p}}{}^j$ de $q_{\setminus \overline{p}}$) ne correspond qu'à au plus un seul sous-arbre fils T^x de T (lemme 19 (*i*))
- tout sous-arbre $p_{\setminus \overline{q}}{}^i$, resp. $q_{\setminus \overline{p}}{}^j$, est associé à un (et un seul) sous-arbre p^a de p, resp. q^b de q (lemme 16 *(ii)*).

Bien-sûr il peut y avoir des sous-arbres fils de p ou q sans correspondance dans $p_{\setminus \overline{q}}$ et $q_{\setminus \overline{p}}$, donc C_T n'est pas forcément un couplage maximum du biparti, i.e. $w_T \leq \widetilde{W}(p, q)$.

Nous montrons maintenant que chaque arête ajoutée à C_T en raison d'un T_x (cf cas 1.,2.,3. ci-dessus), a une valeur supérieure ou égale à $|T^x|$:

1.

$F(T^x) \subseteq$	$F(T) \subseteq$	$F(p_{\setminus \overline{q}}) \cup F(q_{\setminus \overline{p}})$	par (9) appliquée à T
$\Rightarrow$	$F(T^x) \subseteq$	$F(p_{\setminus \overline{q}})$	car dans le cas présent $F(T^x) \cap F(q_{\setminus \overline{p}}) = \emptyset$
$\Rightarrow$	$F(T^x) \subseteq$	$F(p_{\setminus \overline{q}}{}^i)$	par $F(T^x) \cap F(p_{\setminus \overline{q}}{}^i) \neq \emptyset$ et le lemme 19 (ii)
$\Rightarrow$	$F(T^x) \subseteq$	$F(p^a{}_{\setminus \overline{q}})$	pour un sous-arbre p^a de p (lemme 16 *(ii)*)

   On a $F(p^a{}_{\setminus \overline{q}}) = \mathbb{F}(p^a) \cup (F(p^a) \cap F(q))$ et comme $F(T^x) \cap F(q_{\setminus \overline{p}}) = \emptyset$ donc $F(T^x) \cap F(q) = \emptyset$, on déduit $F(T^x) \subseteq \mathbb{F}(p^a)$. Donc $|T^x| \leq |\mathbb{F}(p^a)|$, la valeur de l'arête ajoutée à C_T dans ce cas là.
2. Une explication similaire montre que $|T^x| \leq |\mathbb{F}(q^b)|$, la valeur de l'arête ajoutée à C_T dans ce deuxième cas.
3. Une preuve similaire[8] à celle du lemme 21 (basée sur $F(T^x) \cap F(p_{\setminus \overline{q}}{}^i) \neq \emptyset$, $F(T^x) \cap F(q_{\setminus \overline{p}}{}^j) \neq \emptyset$, le fait que T^x vérifie (23), sur les axiomes du lemme 19 et le lemme 16 *(ii)*) permet de montrer que T^x est super-arbre d'accord de $p^a{}_{\setminus \overline{q^b}}$ et $q^b{}_{\setminus \overline{p^a}}$, donc que

$$\begin{aligned} |T_x| &\leq SuperMast(p^a{}_{\setminus \overline{q^b}}, q^b{}_{\setminus \overline{p^a}}) \\ &\leq SMAST(p^a, q^b) \text{ (en appliquant ensuite l'induction)}, \end{aligned}$$

   la valeur de l'arête ajoutée à C_T dans ce cas.

En conséquence

$SuperMast(p_{\setminus \overline{q}}, q_{\setminus \overline{p}}) = \lvert T \rvert = \sum_{T^x} \lvert T^x \rvert$	$\leq w_T$	$\forall T^x$, l'arête ajoutée étant de valeur $\leq \lvert T^x \rvert$
	$\leq \widetilde{W}(p, q)$	par définition de $\widetilde{W}(p, q)$
	$\leq SMAST(p, q)$	par la formule (21)

[8] détaillée en annexe

Donc dans toutes les situations possibles, on a montré $SuperMast(\hat{p},\hat{q}) \leq SMAST(p,q)$.

Il nous faut maintenant montrer que $|T| \geq SMAST(p,q)$ pour obtenir l'égalité souhaitée dans le théorème. On montre que pour toutes les façons dont la valeur $SMAST(p,q)$ peut être fixée dans l'équation (21) (la seule qui s'applique dans le cas $|p|>1, |q|>1, F(p)\cap F(q) \neq \emptyset$), on peut construire un arbre T' t.q. $|T'| = SMAST(p,q)$ et qui soit un super-arbre d'accord de $p_{\setminus\overline{q}}$ et $q_{\setminus\overline{p}}$, et donc que forcément

$$SMAST(p,q) = |T'| \leq |T| = SuperMast(p_{\setminus\overline{q}}, q_{\setminus\overline{p}}) \ .$$

Il y a trois cas à traiter suivant que dans l'équation (21), un arbre source est apparié à un sous-arbre fils de l'autre arbre source (cas $\overleftarrow{A}$ et $\overleftarrow{B}$ ci-dessous) ou que les sous-arbres fils des arbres sources sont appariés entre eux (cas $\overleftarrow{C}$ ci-après).

($\overleftarrow{A}$) Cas où $SMAST(p,q) = SMAST(p,q^b) + |I\!F_{\setminus q^b}(q)|$, q^b étant sous-arbre fils de q

Par induction, on sait $SMAST(p,q^b) = SuperMast(p_{\setminus\overline{q^b}}, {q^b}_{\setminus\overline{p}})$. Soit T'' un super-arbre d'accord maximum de $p_{\setminus\overline{q^b}}$ et ${q^b}_{\setminus\overline{p}}$ on montre d'abord $T''|F(p_{\setminus\overline{q}}) \subseteq_h p_{\setminus\overline{q}}$ et $T''|F(q_{\setminus\overline{p}}) \subseteq_h q_{\setminus\overline{p}}$:

– commençons par montrer

$$T''|F({q^b}_{\setminus\overline{p}}) = T''|F(q_{\setminus\overline{p}}) \tag{33}$$

En effet, si cette égalité n'est pas vérifiée, alors $\exists f \in F(T'') \cap (F(q_{\setminus\overline{p}}) - F({q^b}_{\setminus\overline{p}}))$ ce qui est impossible car par (9) $F(T'') \subseteq I\!F(p) \cup I\!F(q^b) \cup (F(p) \cap F(q^b))$.
Par ailleurs, en appliquant les définitions, on a $T''|F({q^b}_{\setminus\overline{p}}) \subseteq_h {q^b}_{\setminus\overline{p}} \subseteq_h q_{\setminus\overline{p}}$. En combinant avec (33) on obtient

$$T''|F(q_{\setminus\overline{p}}) \subseteq_h q_{\setminus\overline{p}} \tag{34}$$

– on commence par montrer

$$T''|F(p_{\setminus\overline{q^b}}) = T''|F(p_{\setminus\overline{q}}) \tag{35}$$

En effet, si cette égalité n'est pas vérifiée, alors $\exists f \in F(T'') \cap (F(p_{\setminus\overline{q^b}}) - F(p_{\setminus\overline{q}}))$ ce qui est impossible (toujours en raison de (9) appliquée à T'').
Par définition de T'' on a $T''|F(p_{\setminus\overline{q^b}}) \subseteq_h p_{\setminus\overline{q^b}}$ donc avec (35) on déduit $T''|F(p_{\setminus\overline{q}}) \subseteq_h p_{\setminus\overline{q^b}}$. Comme on passe de $p_{\setminus\overline{q^b}}$ à $p_{\setminus\overline{q}}$ en ajoutant uniquement des feuilles n'appartenant pas à T'' (i.e., des feuilles de $(F(q) - F(q^b)) \cap F(p)$), on en déduit

$$T''|F(p_{\setminus\overline{q}}) \subseteq_h p_{\setminus\overline{q}} \tag{36}$$

De (9) on sait $F(T'') \subseteq I\!F(p) \cup I\!F(q^b) \cup (F(p) \cap F(q^b))$. On en déduit

$$F(T'') \subseteq I\!F(p) \cup I\!F(q) \cup (F(p) \cap F(q))$$

puisque $I\!F(q^b) \subseteq I\!F(q)$ et $F(p) \cap F(q^b) \subseteq F(p) \cap F(q)$, ce qui montre

$$F(T'') \subseteq F(p_{\setminus\overline{q}}) \cup F(q_{\setminus\overline{p}}) \tag{37}$$

Par l'axiome (8) appliqué à T'' on a $\exists f \in F(T'') \cap F(p) \cap F(q^b)$ donc $f \in F(T'') \cap F(p) \cap F(q)$ et ainsi

$$F(T'') \cap (F(p_{\setminus\overline{q}}) - I\!F(p_{\setminus\overline{q}})) \neq \emptyset \text{ et } F(T'') \cap (F(q_{\setminus\overline{p}}) - I\!F(q_{\setminus\overline{p}})) \neq \emptyset \tag{38}$$

Deux cas sont maintenant possibles suivant que $I\!F_{\setminus q^b}(q)$ soit vide ou non :

cas A1 : si $I\!F_{\setminus q^b}(q) = \emptyset$ alors $SMAST(p,q) = SMAST(p,q^b) = |T''| = SuperMast(p_{\setminus \overline{q^b}}, q^b{}_{\setminus \overline{p}})$ et comme T'' vérifie toutes les conditions d'un super-arbre d'accord de $p_{\setminus \overline{q}}$ et $q_{\setminus \overline{p}}$ ((34), (36), (37), (38)) on a

$$SMAST(p,q) = |T''| \leq SuperMast(p_{\setminus \overline{q}}, q_{\setminus \overline{p}}) = |T|$$

par définition de $SuperMast$.

cas A2 : si $I\!F_{\setminus q^b}(q) \neq \emptyset$, soit $q^{i_1} \dots q^{i_{s'}}$ les sous-arbres fils de q autres que q^b et t.q. $I\!F(q^{i_1}) \neq \emptyset, \dots, I\!F(q^{i_{s'}}) \neq \emptyset$, et soit T' l'arbre dont la racine a $i_{s'}+1$ sous-arbres fils : $T'', q|I\!F(q^{i_1}), \dots, q|I\!F(q^{i_{s'}})|$. Pour tout $i_1 \leq i_j \leq i_{s'}$, on sait $I\!F(q^{i_j}) = I\!F(q^{i_j}{}_{\setminus \overline{p}})$, donc $T'|I\!F(q^{i_j}) = q_{\setminus \overline{p}}|I\!F(q^{i_j})$. On a donc construit un arbre T' t.q. :

- $T'|F(q_{\setminus \overline{p}}) \subseteq_h q_{\setminus \overline{p}}$ et $T'|F(p_{\setminus \overline{q}}) = T''|F(p_{\setminus \overline{q}}) \subseteq_h p_{\setminus \overline{q}}$ (notamment en raison de (34) et (36)),
- $F(T') = F(T'') \cup I\!F_{\setminus q^b}(q) \subseteq F(p_{\setminus \overline{q}}) \cup F(q_{\setminus \overline{p}})$ (notamment en raison de (37)),
- $F(T') \cap (F(p_{\setminus \overline{q}}) - I\!F(p_{\setminus \overline{q}})) \neq \emptyset$ et $F(T') \cap (F(q_{\setminus \overline{p}}) - I\!F(q_{\setminus \overline{p}})) \neq \emptyset$ (en raison de (38)).

Les trois points ci-dessus montrent que T' est un super-arbre d'accord de $p_{\setminus \overline{q}}$ et $q_{\setminus \overline{p}}$. De plus,

$$\begin{array}{rcll} |T'| = & |T''| & + & |I\!F(q^{i_1})| + \ldots + |I\!F(q^{i_{s'}})| \\ = & SMAST(p,q^b) & + & |I\!F_{\setminus q^b}(q)| \\ = & SMAST(p,q) & & \end{array}$$

Or T' étant un super-arbre d'accord de $p_{\setminus \overline{q}}$ et $q_{\setminus \overline{p}}$, on a

$$SMAST(p,q) = |T'| \leq SuperMast(p_{\setminus \overline{q}}, q_{\setminus \overline{p}}) = |T| \ .$$

($\overleftarrow{B}$) Cas où $SMAST(p,q) = SMAST(p^a,q) + |I\!F_{\setminus p^a}(p)|$, p^a étant sous-arbre fils de p

La preuve est identique à celle du cas ($\overleftarrow{A}$).

($\overleftarrow{C}$) Cas où $SMAST(p,q) = \widetilde{W}(p,q)$

Soit C_T un couplage de poids maximum dans $G(p,q)$. Si C_T possède des arêtes de poids nul (la proposition 14 et sa preuve indiquent que c'est possible) alors on les enlève. C_T a trois types d'arêtes :

- $(p^{a_1}, q^{b_1}), \dots, (p^{a_w}, q^{b_w})$ connectant un sous-arbre fils de p à un sous-arbre fils de q
- $(p^{a_{w+1}}, v'_{a_{w+1}}), \dots, (p^{a_{w+x}}, v'_{a_{w+x}})$ connectant un sous-arbre fils de p à un sommet virtuel
- $(q^{b_{w+1}}, v_{b_{w+1}}), \dots, (q^{b_{w+y}}, v_{b_{w+y}})$ connectant un sous-arbre fils de q à un sommet virtuel

Toute arête de type (p^a, q^b) a un poids $SMAST(p^a, q^b)$, toute arête de type (p^a, v'_a) a un poids $|I\!F(p^a)|$ et toute arête de type (q^b, v_b) a un poids $|I\!F(q^b)|$, et $\widetilde{W}(p,q) = w(C_T)$ est la somme de ces poids.

Pour toute arête de type (p^a, q^b), soit $T''_{ab} = SuperMast_t(p^a{}_{\setminus \overline{q^b}}, q^b{}_{\setminus \overline{p^a}})$. En procédant comme dans la preuve du cas (A) ci-dessus :

- de $T''_{ab}|F(p^a{}_{\setminus \overline{q^b}}) \subseteq_h p^a{}_{\setminus \overline{q^b}}$ et $T''_{ab}|F(q^b{}_{\setminus \overline{p^a}}) \subseteq_h q^b{}_{\setminus \overline{p^a}}$ on peut montrer

$$T''_{ab}|p_{\setminus \overline{q}} \subseteq_h p_{\setminus \overline{q}} \text{ et } T''_{ab}|F(q_{\setminus \overline{p}}) \subseteq_h q_{\setminus \overline{p}} \tag{39}$$

- de $F(T''_{ab}) \subseteq I\!F(p^a) \cup I\!F(q^b) \cup (F(p^a) \cap F(q^b))$ on déduit

$$F(T''_{ab}) \subseteq F(p_{\setminus \overline{q}}) \cup F(q_{\setminus \overline{p}}) \tag{40}$$

– de $F(T''_{ab}) \cap F(p^a{}_{\backslash\overline{q^b}}) \cap F(q^b{}_{\backslash\overline{p^a}}) \neq \emptyset$ on déduit

$$F(T''_{ab}) \cap F(p_{\backslash\overline{q}}) \cap F(q_{\backslash\overline{p}}) \neq \emptyset \tag{41}$$

Soit $T'_1 = SuperMast_t(p^{a_1}{}_{\backslash\overline{q^{b_1}}}, q^{b_1}{}_{\backslash\overline{p^{a_1}}}), \ldots, T'_w = SuperMast_t(p^{a_w}{}_{\backslash\overline{q^{b_w}}}, q^{b_w}{}_{\backslash\overline{p^{a_w}}})$. On construit l'arbre T' dont la racine a les $w+x+y$ sous-arbres fils suivants :

$$T'_1, \ldots, T'_w, p|\mathbb{F}(p^{a_{w+1}}), \ldots, p|\mathbb{F}(p^{a_{w+x}}), q|\mathbb{F}(q^{b_{w+1}}), \ldots, q|\mathbb{F}(q^{b_{w+y}}) \ .$$

Par construction, chaque arête de C_T correspond à un sous-arbre fils, et le poids de l'arête correspond à la taille de ce sous-arbre fils (par induction pour les T'_i et par construction pour les $p|\mathbb{F}(p^{a_{w+x}})$ et les $q|\mathbb{F}(q^{b_{w+1}})$). On a donc $|T'| = w(C_T) = \widetilde{W}(p,q) = SMAST(p,q)$.

En raison de la façon dont T' est construit est de (39), (40), (41) on déduit que T' est un super-arbre d'accord de $p_{\backslash\overline{q}}$ et $q_{\backslash\overline{p}}$, donc

$$SMAST(p,q) = |T'| \leq SuperMast(p_{\backslash\overline{q}}, q_{\backslash\overline{p}}) = |T| \ .$$

Dans tous les cas d'obtention de $SMAST(p,q)$ on a donc montré que

$$SMAST(p,q) \leq SuperMast(p_{\backslash\overline{q}}, q_{\backslash\overline{p}}) = |T|$$

ce qui complète la preuve. □

COROLLAIRE 23
Soient P et Q deux arbres enracinés, alors $SMAST(P,Q) = SuperMast(P,Q)$.

PREUVE :
Il suffit de noter que $F(P\backslash Q) = F(P)$ et $F(Q\backslash P) = F(Q)$, donc $P_{\backslash\overline{Q}} = P$ et $Q_{\backslash\overline{P}} = Q$, puis le théorème précédent permet de conclure. □

7.4 Complexité

En $O(n)$ on peut parcourir les deux arbres P, Q et pour chaque $p \in P$ resp. $q \in Q$, connaître $|\mathbb{F}(p)|$ resp. $|\mathbb{F}(q)|$.

Dans le calcul de $SMAST(p,q)$, pour tout couple (p,q) on a besoin de savoir si $p \cap q = \emptyset$ ou pas. On peut répondre en $O(1)$ à toute question de ce type après un prétraitement en $O(n^2)$.

Maintenant construire les bipartis pour tous les couples (p,q) ne coûte globalement que $O(n^2)$. En effet, soit p^a et q^b des sous-arbres de P, resp. Q, dont les pères respectifs sont p, resp. q :

- SOMMETS : un sous-arbre $p^a \in P$ (wlog $q^b \in Q$) ne sera sommet que dans les graphes bipartis impliquant p : il y en a $O(n)$. Donc tout sous-arbre donnera $O(n)$ sommets (non-virtuels et virtuels), ce qui borne par $O(n^2)$ le nombre de sommets de tous les graphes bipartis puisque le nombre de sous-arbres de P et de Q est borné par $O(n)$.
- ARÊTES : tout couple (p^a, q^b) ne donnera une arête (encore à condition que sa valeur soit positive) que dans un biparti, celui associé à (p,q). Comme le nombre de $p^a \in P$ et de $q^b \in Q$, sont bornés par $O(n)$, le nombre d'arêtes entre sommets non-virtuels de tous les bipartis est borné par $O(n^2)$. Les arêtes entre un sommet virtuel et son homologue non-virtuel représentant un sous-arbre $p^a \in P$ (wlog $q^b \in Q$), sont en même nombre que le nombre d'apparitions de sous-arbres dans les bipartis, donc en $O(n^2)$. Donc le nombre total d'arêtes considérées dans l'ensemble des bipartis construits pour le calcul de $SMAST(P,Q)$ est en $O(n^2)$.

Clairement maintenant, le coût du calcul de $SMAST(p,q)$ est dominé par le calcul d'un couplage de valeur maximum dans le graphe biparti $G(p,q)$. Dans ce graphe, $\sum_{e \in G} v(e) \in O(n^2)$ en raison des sous-arbres ayant des feuilles spécifiques, donc la complexité de l'algorithme de [34] pour trouver un couplage maximum est ici $O(n^{2.5})$ (contre $O(n^{1.5})$ pour le graphe du problème MAST). Comme on a $O(n^2)$ paires de sous-arbres (p,q), on en déduit que la complexité de l'algorithme ci-dessus pour calculer $SuperMast(\mathcal{T})$ est de $O(n^{4.5})$ dans le cas général.

Si le degré des noeuds dans les arbres sources est borné par une constante d, l'algorithme de Gabow et Tarjan [23] a une complexité en $O(d^{2.5} \log n)$, ce qui donne une complexité en $O(d^{2.5} n^2 \log n)$ pour résoudre le problème MAT.

Comme suggéré par [48], Si d est suffisamment petit, pour résoudre le problème du couplage maximum dans un biparti, on peut procéder par énumération, ce qui donne une complexité en $O(d! n^2)$ pour le problème MAT.

7.5 Cas des arbres non-enracinés

Notons qu'à l'inverse de l'approche de [48], il n'est pas possible d'obtenir un super-arbre d'accord maximum de deux arbres sources T_1, T_2 non-enracinés en examinant des paires de sous-arbres *complémentaires*.

Mais, comme pour le problème MAST [2], on peut procéder en essayant successivement tous les enracinements possibles de T_1 et T_2 *en une feuille commune* (pour être sûr que $T_1|F_{12}$ et $T_2|F_{12}$ seront enracinés en une même feuille), puis en greffant les feuilles de $\mathbb{F}(\mathcal{T})$ sur le plus grand sous-arbre enraciné d'accord maximum des enracinements de $T_1|F_{12}$ et $T_2|F_{12}$ (en utilisant l'algorithme de la section 6), enfin en désenracinant le super-arbre obtenu.

La complexité de l'algorithme résultant est $O(n^{5.5})$.

8 Un algorithme pour le problème général

Bien qu'une mesure d'accord d'une collection $\mathcal{T}$ de k arbres sources puisse être obtenue comme la taille moyenne d'un super-arbre d'accord maximum de tout couple d'arbres $T_i, T_j \in \mathcal{T}$, on peut souhaiter calculer le super-arbre d'accord exact de k arbres sources.

La section 5 a montré que le problème MAT général était $W[2]$-difficile. Nous donnons toutefois ici un algorithme (de complexité exponentielle) qui trouve une application en pratique dans le cas d'arbres compatibles ou dont l'incompatibilité n'est due qu'à un faible nombre de feuilles.

Les arbres sources sont d'abord traduits en ensembles G' de triplets et de fans les définissant chacun des arbres sources de façon unique par l'algorithme BREAKUP de [36]. Puis on applique l'algorithme ONETREE de [36] d'abord sur la collection G de triplets et de fans induits par les arbres sources, puis sur celle obtenue en éliminant une feuille (toutes les feuilles sont examinées tour à tour), puis sur celle obtenue en éliminant deux feuilles, et ainsi de suite jusqu'à ce que ONETREE renvoit un arbre non-nul.

Dans un tel cas, ONETREE renvoit un arbre T t.q. $\forall T_i \in \mathcal{T}, T|F(T_i) \subseteq_h T_i$. Autrement dit, T est un super-arbre d'accord de $\mathcal{T}$. D'autre part, l'algorithme ci-dessus garantit qu'il n'existe pas de super-arbre d'accord de $\mathcal{T}$ de plus grande taille, sinon il aurait été rencontré avant.

8.1 Complexité

L'algorithme BREAKUP renvoit en temps $O(nk)$ un nombre $O(nk)$ de triplets et de *fans* dans G où $n = |F(T)|$ est le nombre de feuilles dans les arbres sources et k le nombre d'arbres sources. L'algo-

Algorithme 3: Calcul d'un super-arbre d'accord maximum de k arbres

Données : une collection $\mathcal{T}$ d'arbres sources
Résultat : un arbre $T := SuperMast_t(\mathcal{T})$
$p \leftarrow 0$ et $G \leftarrow \emptyset$
pour chaque $T_i \in \mathcal{T}$ **faire**
  $G'_i \leftarrow$ `BreakUp(`T_i`)`
  $G \leftarrow G \cup G'_i$
tant que $\mathcal{T}$ *n'est pas compatible* **faire**
  pour chaque *ensemble* S *de* p *feuilles* **faire**
    $G' \leftarrow G|F(\mathcal{T}) - S$
    $T \leftarrow$ `OneTree(`G'`)`
    si $T \neq \emptyset$ **alors** renvoyer T
    $p \leftarrow p + 1$

rithme ONETREE est utilisé pour savoir si l'ensemble de triplets et de fans obtenus est compatible ou pas. Cet algorithme demande un temps d'exécution $N_{OT} = O(n(n+nk+bn))$ où b est la somme des carrés des nombres de feuilles dans les fans. Alternativement, on peut utiliser l'algorithme BUILD de [1], après avoir transformé chaque fan renvoyé par BREAKUP en $O(n^2)$ contraintes [36]. L'algorithme BUILD a alors en entrée $O(nk + n^3k)$ contraintes, ie fonctionne en temps $N_{OT} = O(n^6k^2 \log nk)$.

L'algorithme ci-dessus a donc une complexité en $O(n^{\frac{p(p+1)}{2}} N_{OT} + nk)$ où $p = n - |T|$, pour $T := SuperMast_t(\mathcal{T})$.

Notons que pour savoir s'il existe un super-arbre d'accord maximum ayant au moins $n-p$ feuilles pour p fixé, une variante de l'algorithme ci-dessus permet de répondre en temps $O(n^p N_{OT} + nk)$.

9 Conclusion et questions ouvertes

Nous avons défini MAT, une extension du problème MAST au contexte des super-arbres. Le problème MAST est un cas particulier de MAT, si bien que le problème MAT est NP-difficile pour trois arbres sources ou plus de degré *non-borné*.

Nous avons montré que MAT est plus difficile que MAST au sens où il est W[2]-difficile pour k arbres de degré *non-borné*, cas où le problème MAST devient polynomial [2, 19]. Le problème MAT reste NP-difficile si les arbres sources ont un nombre f borné de feuilles (pour $f \geq 3$), cas polynomial pour le problème MAST.

Pour le cas de *deux* arbres de degré non-borné, sur la base du problème MAST, nous obtenons un algorithme polynomial de complexité $O(n + N)$ où N est la complexité nécessaire pour le problème MAST sur deux arbres (actuellement $N = O(n^{1.5})$). Nous indiquons aussi une modification de l'algorithme produisant un super-arbre qui n'est plus strictement un super-arbre d'accord des arbres sources (c'est un super-arbre d'accord d'une collection d'arbres obtenus par contractions des arbres sources initiaux). Les multifourches de cet arbre indiquent explicitement les parties de la phylogénie estimée pour lesquelles il est nécessaire de disposer de plus de données. Ces multifourches sont dues à une absence d'information croisée dans les arbres sources.

Pour le cas général de k arbres sources nous avons montré qu'il n'existe pas d'algorithme polyunomial en n et p, le nombre minimum de feuilles qu'il faut enlever pour que les arbres sources soient en accord. Nous donnons un algorithme en $O(n^{p^2} N_{OT} + nk)$ où N_{OT} est la complexité de

l'algorithme ONETREE de [36].

Pour le cas de k arbres sources, une heuristique consiste à utiliser de façon répétée l'algorithme donné pour deux arbres sources. On obtient ainsi une estimation $\hat{T}$ du super-arbre d'accord maximum d'une collection $\mathcal{T} = \{T_1, \ldots, T_k\}$:

$$\hat{T} = SuperMast_t(T_k, SuperMast_t(...SuperMast_t(T_2, T_1)...))$$

Plusieurs questions restent ouvertes :

- Est-ce que le problème MAT reste polynomial dans le cas d'un nombre borné $k > 3$ arbres de degré non-borné ? Nous conjecturons qu'il existe un algorithme polynomial dans un tel cas.
- Dans le cas où l'on dispose de plus de deux arbres sources, le problème MAT est NP-difficile, même pour des arbres de degré borné. Le fait qu'il soit $\mathrm{W}[2]$-difficile laisse peu d'espoir sur l'existence d'un algorithme FPT pour ce problème. Nous donnons dans la section 8 un algorithme naïf de complexité exponentielle. Toutefois, il est sûrement possible d'obtenir un algorithme de moindre complexité.
- Récemment, [24] ont introduit le problème MCT (*Maximum Compatible Tree*), une variante du problème *MAST* qui est susceptible de conserver un plus grand ensemble des feuilles initiales et proposent un algorithme en $O(n^{kd}2^{2kd})$ dans le cas d'arbres enracinés. Même s'ils disent ne pas savoir si ce problème est polynomial ou NP-difficile dans le cas de deux arbres, il est sûrement possible d'en trouver une extension dans le cas des super-arbres en utilisant les idées données ici et d'obtenir un algorithme heuristique pour résoudre ce problème.

Références

[1] A. V. Aho, Y. Sagiv, T.G. Szymanski, and J.D. Ullman. Inferring a tree from lowest common ancestors with an application to the optimization of relational expressions. *SIAM Journal on Computing*, 10(3) :405–421, 1981.

[2] A. Amir and D. Keselman. Maximum agreement subtree in a set of evolutionary trees : metrics and efficient algorithm. *SIAM J. on Comp.*, 26(3) :1656–1669, 1997.

[3] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, and M. Protasi. Complexity and approximation. Springer-Verlag, November 1999.

[4] R. Bar-Yehuda and S. Even. A linear-time approximation algorithm for the weighted vertex cover problem. *J. Algorithms*, 2 :198–203, 1981.

[5] B.R. Baum. Combining trees as a way of combining data sets for phylogenetic inference, and the desirability of combining gene trees. *Taxon*, 41 :3–10, 1992.

[6] M. Bellare, S. Goldwasser, C. Lund, and A. Russeli. Efficient probabilistically checkable proofs and applications to approximations. In *Proceedings of the Twenty-Fifth Annual A.C.M. Symposium on Theory of Computing*, pages 294–304, 1993.

[7] O.R.P. Bininda-Edmonds and H.N. Bryant. Properties of matrix representation with parsimony analyses. *Syst. Biol.*, 47 :497–508, 1998.

[8] O.R.P. Bininda-Edmonds, J.L. Gittleman, and M.A. Steel. The (super)tree of life : procedures, problems, and prospects. *Ann. Rev. Ecol. Syst.*, 2002.

[9] O.R.P. Bininda-Edmonds and M.J. Sanderson. Assessment of the accuracy of matrix representation with parsimony analysis supertree construction. *Syst. Biol.*, 50(4) :565–579, 2001.

[10] D. Bryant. *Building trees, hunting for trees and comparing trees*. PhD thesis, University of Canterbury, Department of Math., 1997.

[11] D. Bryant, M. Fellows, V. Raman, and U. Stege. On the parametrized complexity of mast and 3-hitting set. manuscript, 1998.

[12] D. Bryant and M.A. Steel. Extension operations on sets of leaf-labelled trees. *Adv. Appl. Math.*, 16 :425–453, 1995.

[13] M. Cesati. Compendium of parameterized problems, 2001. Available online at `http ://bravo.ce.uniroma2.it/home/cesati/research/`.

[14] D. Chen, L. Diao, O. Eulenstein, and D. Fernandez-Baca. Supertrees by flipping. In *Comp. Combin. Conf. (COCOON '02)*, page (to appear), 2002.

[15] D. Chen, L. Diao, O. Eulenstein, and D. Fernandez-Baca. Flipping : a supertree construction method. *DIMACS Series in Disc. Math. and Theor. Comp. Sci.*, 61 :135–160, 2003.

[16] R. Cole, M. Farach, R. Hartigan, Przytycka T., and M. Thorup. An $O(n \log n)$ algorithm for the maximum agreement subtree problem for binary trees. *SIAM J. on Computing*, 30(5) :1385–1404, 2001.

[17] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. M.I.T. Press, Cambridge, Massachusetts, second edition, 2001.

[18] R.G. Downey, M.R. Fellows, and U. Stege. Computational tractability : The view from mars. *Bull. of the Europ. Assoc. for Theoret. Comp. Sci.*, 69 :73–97, 1999.

[19] M. Farach, T. Przytycka, and M. Thorup. Agreement of many bounded degree evolutionary trees. *Inf. Proc. Letters*, 55(6) :297–301, 1995.

[20] U. Feige. A threshold of $\ln n$ for approximating Set Cover. *Journal of the A.C.M.*, 45(4) :634–652, 1998.

[21] U. Feige, M. M. Halldórsson, and G. Kortsarz. Approximating the domatic number. In *Proceedings of the Thirty-Second Annual A.C.M. Symposium on Theory of Computing*, pages 134–143, 2000.

[22] R. Finden and A.D. Gordon. Obtaining common pruned trees. *J. of Classif.*, 2 :255–276, 1985.

[23] H.N. Gabow and R.E. Tarjan. Faster scaling algorithms for network problems. *SIAM J. Comput.*, 18(5) :1013–1036, 1989.

[24] G. Ganapathysaravanabavan and T. Warnow. Finding a maximum compatible tree for a bounded number of trees with bounded degree is solvable in polynomial time. In O. Gascuel and B.M.E. Moret, editors, *Proc. of the Workshop on Algorithms for Bioinformatics (WABI'01)*, volume 2149 of *LNCS*, pages 156–163, 2001.

[25] M.R. Garey and D.S Johnson. *Computers and Intractability : A Guide to the Theory of NP-Completeness*. Freeman, New-York, 1979.

[26] J. Gatesy, C. Matthee, R. DeSalle, and C. Hayashi. Resolution of a supertree/supermatrix paradox. *Syst. Biol.*, 51(4) :652–664, 2002.

[27] A.G. Gordon. Consensus supertrees : the synthesis of rooted trees containing overlapping sets of labelled leaves. *J. of Classif.*, 3 :335–346, 1986.

[28] A. Gupta and N. Nishimura. Finding largest subtrees and smallest supertrees. *Algorithmica*, 21(2) :183–210, 1998.

[29] D. Gusfield. Efficient algorithms for inferring evolutionary trees. *Networks*, 21 :19–28, 1991.

[30] D. Harel and R.E. Tarjan. Fast algorithms for finding nearest common ancestor. *Computer and System Science*, 13 :338–355, 1984.

[31] J. Hein, T. Jiang, L. Wang, and K. Zhang. On the complexity of comparing evolutionary trees. In *Proc of the 6th Ann. Symp. on Combin. Pattern Matching (CPM'95)*, volume 937 of *LNCS*. Springer-Verlag, 1995.

[32] D. S. Hochbaum. Approximation algorithms for the set covering and vertex cover problems. *SIAM J. Comp.*, 11 :555–556, 1982.

[33] D.S. Johnson. Approximation algorithms for combinatorial problems. *Journal of Computer and System Sciences*, 9(3) :256–278, 1974.

[34] M.Y. Kao, T.W. Lam, W.K. Sung, and H.F. Ting. A decomposition theorem for maximum weight bipartite matchings with applications to evolutionary trees. In *Proc. of the 8th Ann. Europ. Symp. Alg. (ESA)*, pages 438–449. Springer-Verlag, New York, NY, 1999.

[35] M.Y. Kao, T.W. Lam, W.K. Sung, and H.F. Ting. An even faster and more unifying algorithm for comparing trees via unbalanced bipartite matchings. *J. of Algo.*, 40 :212–233, 2001.

[36] M.P. Ng and N.C. Wormald. Reconstruction of rooted trees from subtrees. *Disc. Appl. Math.*, 69 :19–31, 1996.

[37] R. Niedermeier and P. Rossmanith. An efficient fixed parameter algorithm for 3-hitting set. *Journal of Discrete Algorithms*, 2(1), 2001.

[38] R. Niedermeier and P. Rossmanith. An efficient fixed parameter algorithm for 3-Hitting Set. *Journal of Discrete Algorithms*, 1 :89–102, 2003.

[39] R. Page. Modified mincut supertrees. In O. Gascuel and M.-F. Sagot, editors, *Proc. of the Workshop on Algorithms for Bioinformatics (WABI'02)*, LNCS, pages 538–551. Springer-Verlag, 2002.

[40] R. Przytycka. Sparse dynamic programming for maximum agreement subtree problem. In B. Mirkin, F.R. McMorris, F.S Roberts, and A. Rzhetsky, editors, *Mathematical Hierarchies and Biology*, DIMACS, pages 249–264. Providence, RI, 1997.

[41] A. Purvis. A modification to Baum and Ragan's method for combining phylogenetic trees. *Syst. Biol.*, 44 :251–255, 1995.

[42] M.A. Ragan. Matrix representation in reconstructing phylogenetic relationships among the eukaryots. *Biosystems*, 28 :47–55, 1992.

[43] F. Ronquist. Matrix representation of trees, redundancy, and weighting. *Syst. Biol.*, 45 :247–253, 1996.

[44] M.J. Sanderson, A. Purvis, and C. Henze. Phylogenetic supertrees : assembling the trees of life. *Trends. Ecol. Evol.*, 13 :105–109, 1998.

[45] C. Semple and M.A. Steel. A supertree method for rooted trees. *Disc. Appl. Math.*, 105 :147–158, 2000.

[46] M.A. Steel. The complexity of reconstructing trees from qualitative characters and subtree. *J. of Classif.*, 9 :91–116, 1992.

[47] M.A. Steel, A.W. Dress, and S. Böcker. Simple but fundamental limitations on supertree and consensus tree methods. *Syst. Biol.*, 49 :363–368, 2000.

[48] M.A. Steel and T. Warnow. Kaikoura tree theorems : Computing the maximum agreement subtree. *Information Processing Letters*, 48 :77–82, 1993.

[49] J.L. Thorley and M. Wilkinson. A view of supertrees methods. In *Bioconsensus, DIMACS*, volume 61, pages 185–194. Amer. Math. Soc. Pub., 2003.

[50] T. J. Warnow. Tree compatibility and inferring evolutionary history. *Journal of Algorithms*, 16 :388–407, 1994.

[51] M. Wilkinson, J. Thorley, D.T.J. Littlewood, and R.A. Bray. *Interrelationships of the Platyhelminthes*, chapter 27, Towards a phylogenetic supertree of Platyhelminthes. Taylor and Francis, London, 2001.