Construction et comparaison d'arbres Application à l'étude de l'Évolution

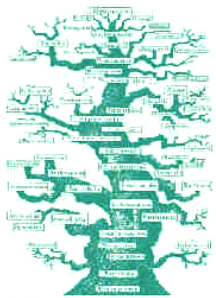
Vincent Berry

Equipe *Méthodes et Algorithmes pour la Bioinformatique*
8 décembre 2008
`http://www.lirmm.fr/~vberry`

L.I.R.M.M. (Université Montpellier II - C.N.R.S.)

Jury

Gilles Caraux	invité
Alain Denise	rapporteur
Olivier Gascuel	examinateur
Manolo Gouy	rapporteur
Alain Guénoche	examinateur
Jean-Claude König	examinateur
Mike Steel	rapporteur

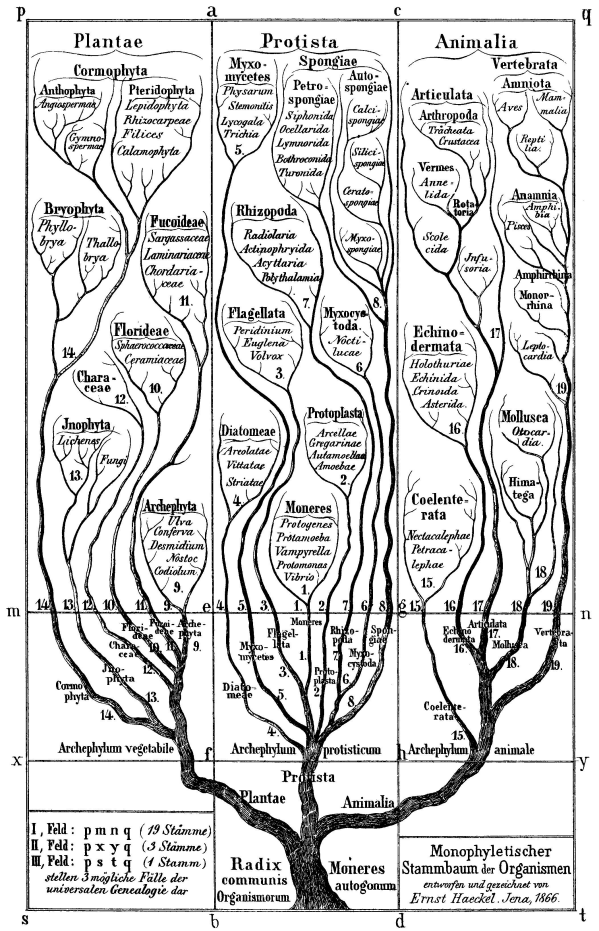


UM2 UNIVERSITÉ MONTPELLIER 2 SCIENCES ET TECHNIQUES

Laboratoire d'Informatique de Robotique et de Microélectronique de Montpellier LIRMM

Phylogénies = arbres étiquetés

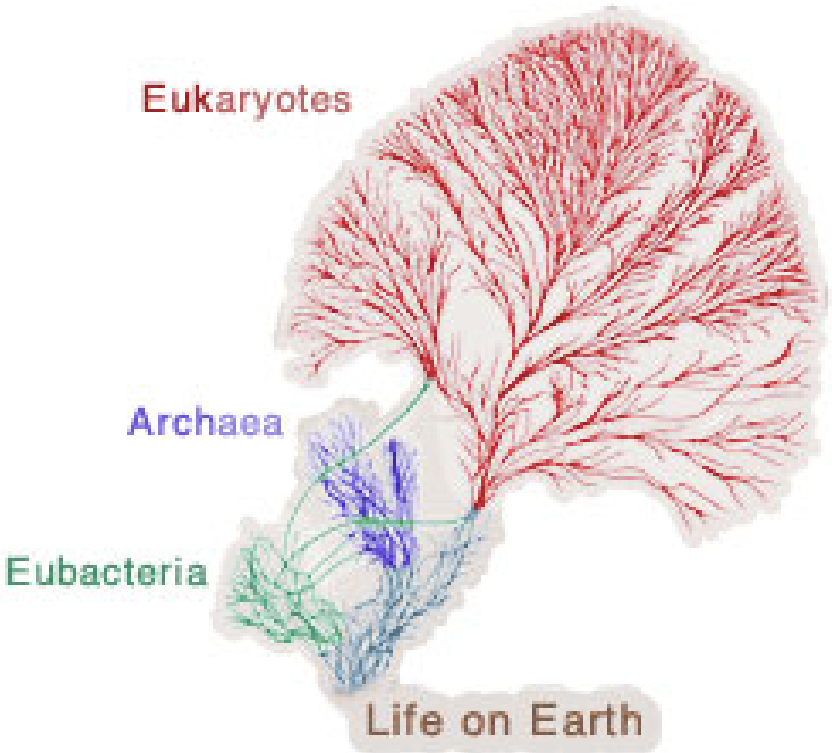
Représentation des apparitions et extinctions d'espèces vivantes suivant la théorie de l'Évolution

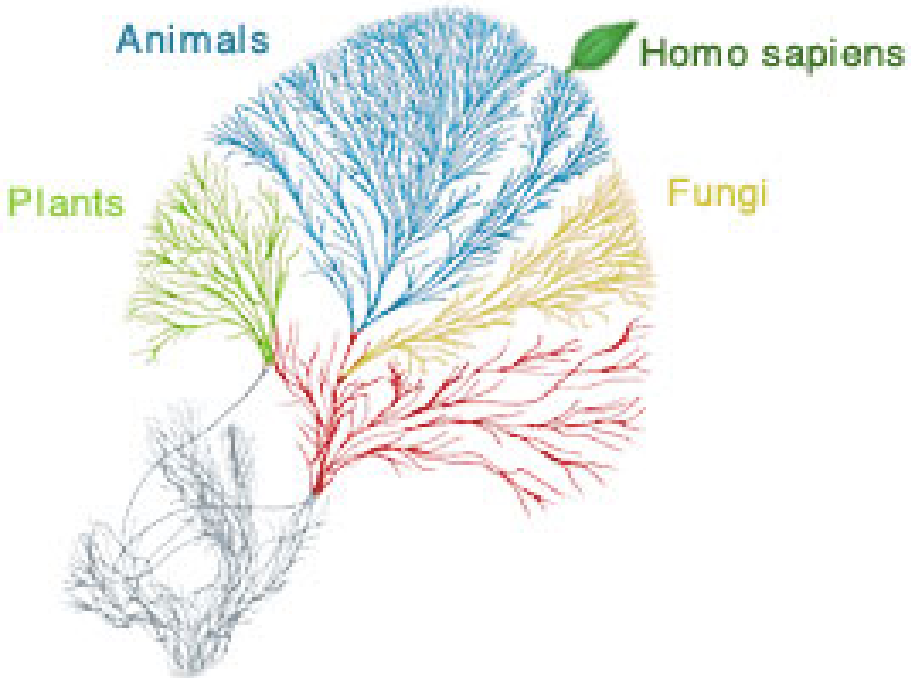


Arbre de Haeckel 1866

Phylogénies = arbres étiquetés

Représentation des apparitions et extinctions d'espèces vivantes suivant la théorie de l'Évolution





vision actuelle - www.tolweb.org

Enjeux

Reconstruction de l'arbre de la Vie : les relations évolutives entre espèces sont loin d'être toutes élucidées

- 64% des noeuds de la taxonomie NCBI ne sont pas encore résolus
- L'arbre du projet collaboratif *Tree Of Life* (320 biologistes, 21 pays) comporte encore de très nombreux noeuds irrésolus.

Enjeux

Reconstruction de l'arbre de la Vie : les relations évolutives entre espèces sont loin d'être toutes élucidées

- 64% des noeuds de la taxonomie NCBI ne sont pas encore résolus
- L'arbre du projet collaboratif *Tree Of Life* (320 biologistes, 21 pays) comporte encore de très nombreux noeuds irrésolus.

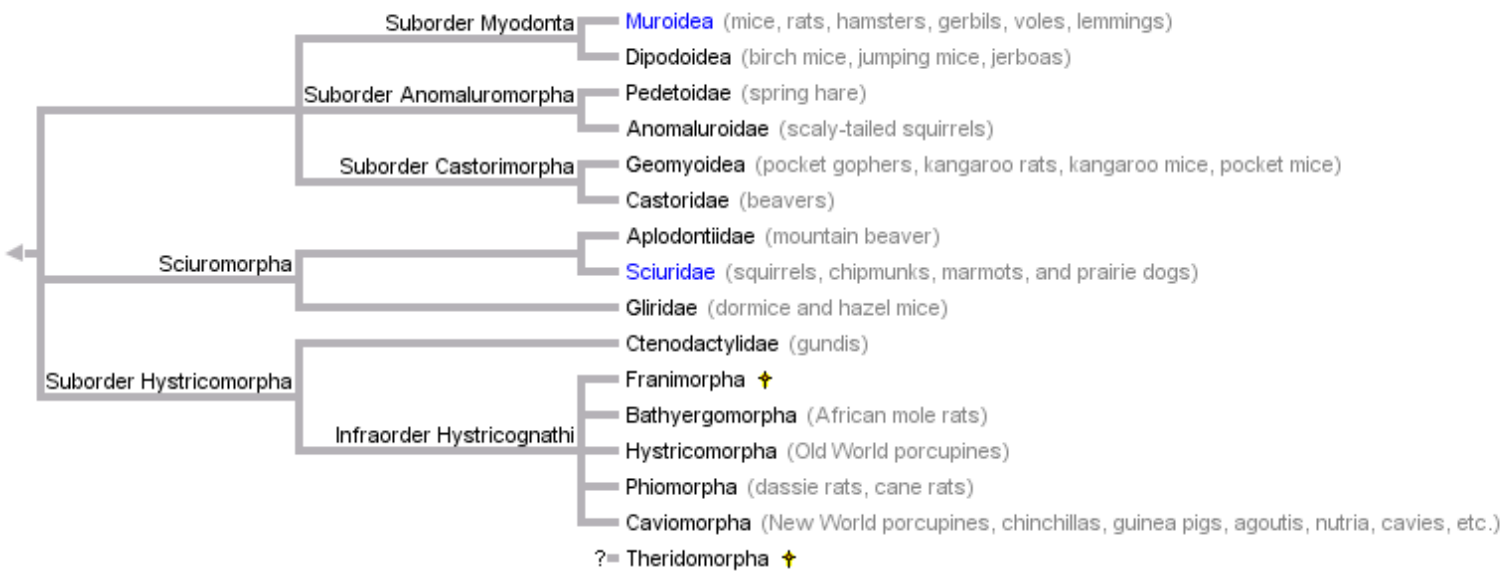
Vision de la phylogénie des rongeurs en 1984

67% des noeuds non résolus

Enjeux

Reconstruction de l'arbre de la Vie : les relations évolutives entre espèces sont loin d'être toutes élucidées

- 64% des noeuds de la taxonomie NCBI ne sont pas encore résolus
- L'arbre du projet collaboratif *Tree Of Life* (320 biologistes, 21 pays) comporte encore de très nombreux noeuds irrésolus.



Vision de la phylogénie des rongeurs en 2008

36% des noeuds non résolus

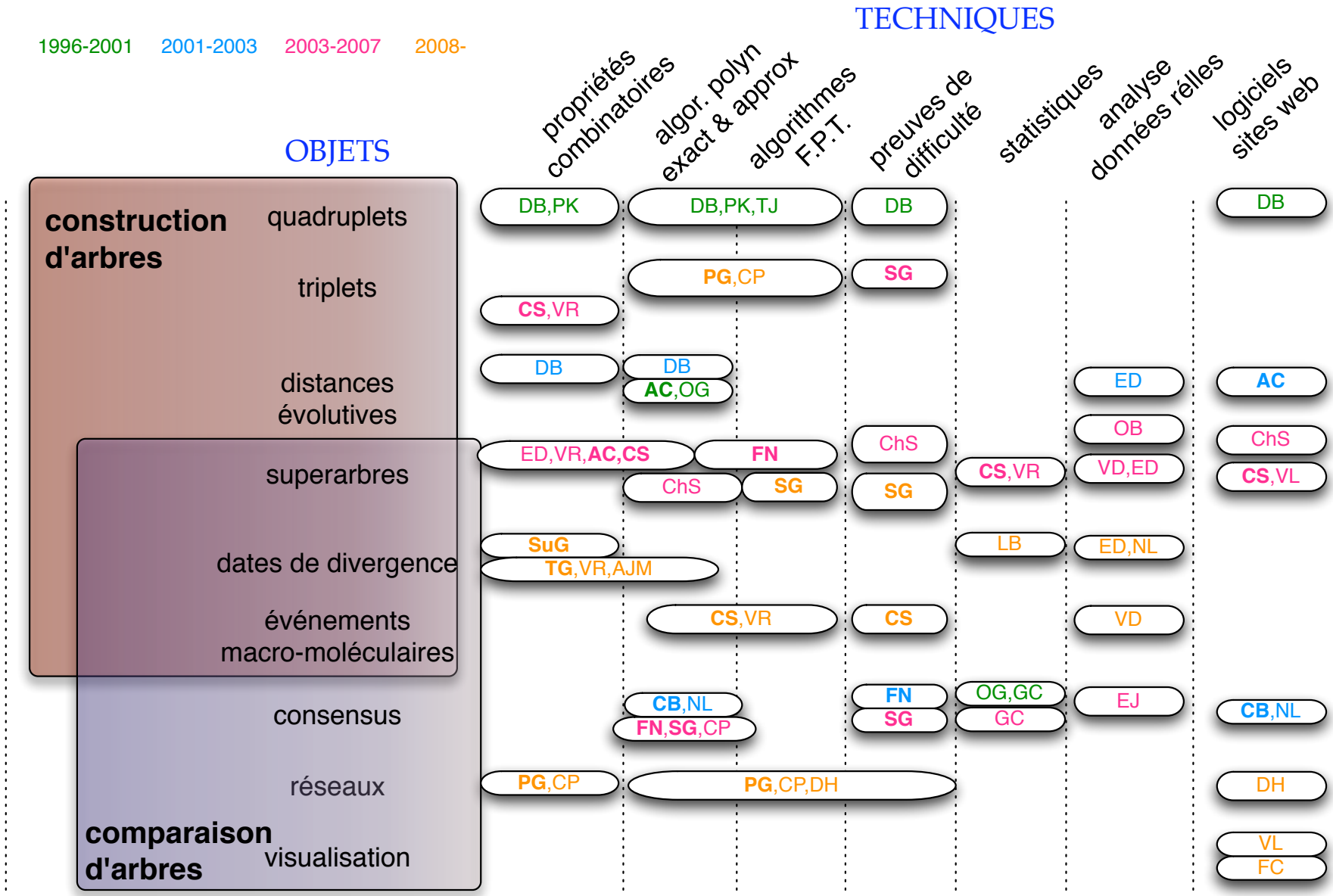
Enjeux

Pourquoi comparer des arbres ?

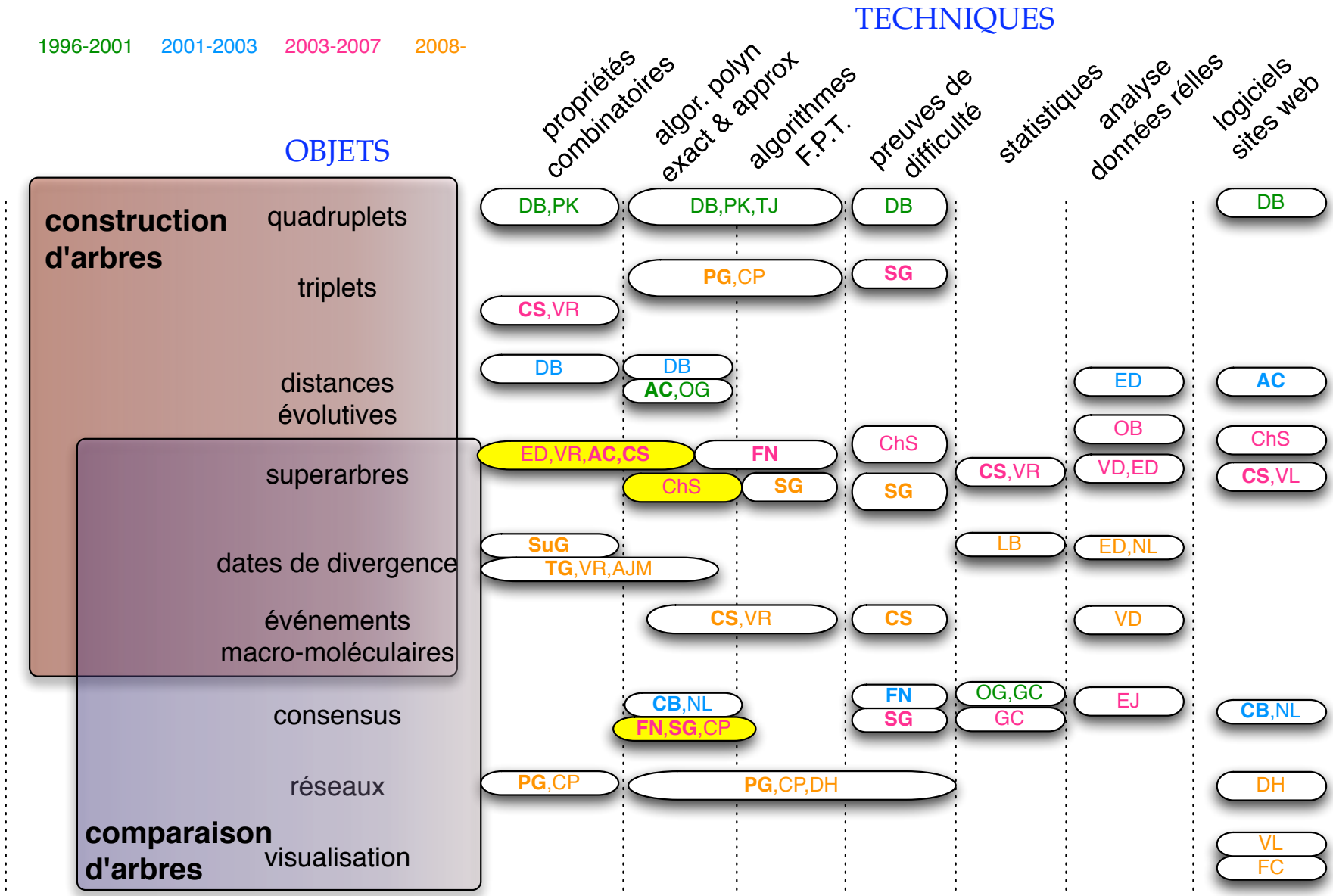
- pour identifier les zones d'accord ou de désaccord d'arbres de gènes, et plus précisément
- pour séparer le signal phylogénétique du bruit (imperfection des méthodes et/ou des données)
- pour identifier des phénomènes de co-spéciation (relations hôtes/parasites)
- pour identifier des événements macro-évolutifs (transferts de gènes)

1. Introduction au domaine de recherche
2. Directions de recherches
3. Comparaison d'arbres sur un même ensemble d'étiquettes
   - Algorithmes certifiants d'isomorphisme et de compatibilité
   - Sous-arbre d'Accord Maximum (MAST)
4. Construction d'arbre depuis des arbres sources ayant des ensembles d'étiquettes disjoints
   - Le problème de l'inclusion taxonomique
5. Construction de superarbres et comparaison d'arbres
   - Contexte
   - Propriétés combinatoires et pratiques intéressantes
   - La méthode PhySIC_IST
6. Projets de recherche

Directions de recherche



Directions de recherche



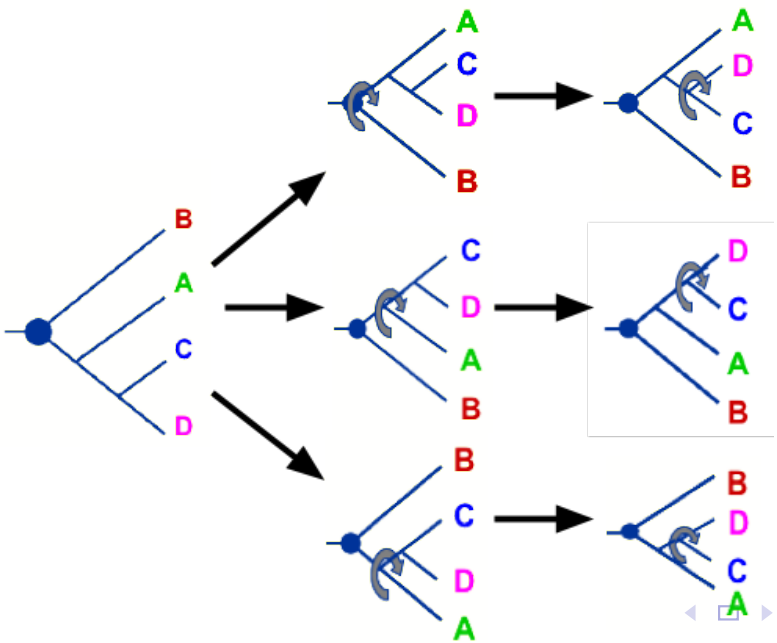
Plan

Objets considérés

Les arbres représentant l'évolution des espèces classiquement :

- sont enracinés (pas toujours vrai)
- ont leurs feuilles bijectivement associées à des étiquettes
- ont leurs sous-arbres non-ordonnés

Isomorphisme d'arbres :

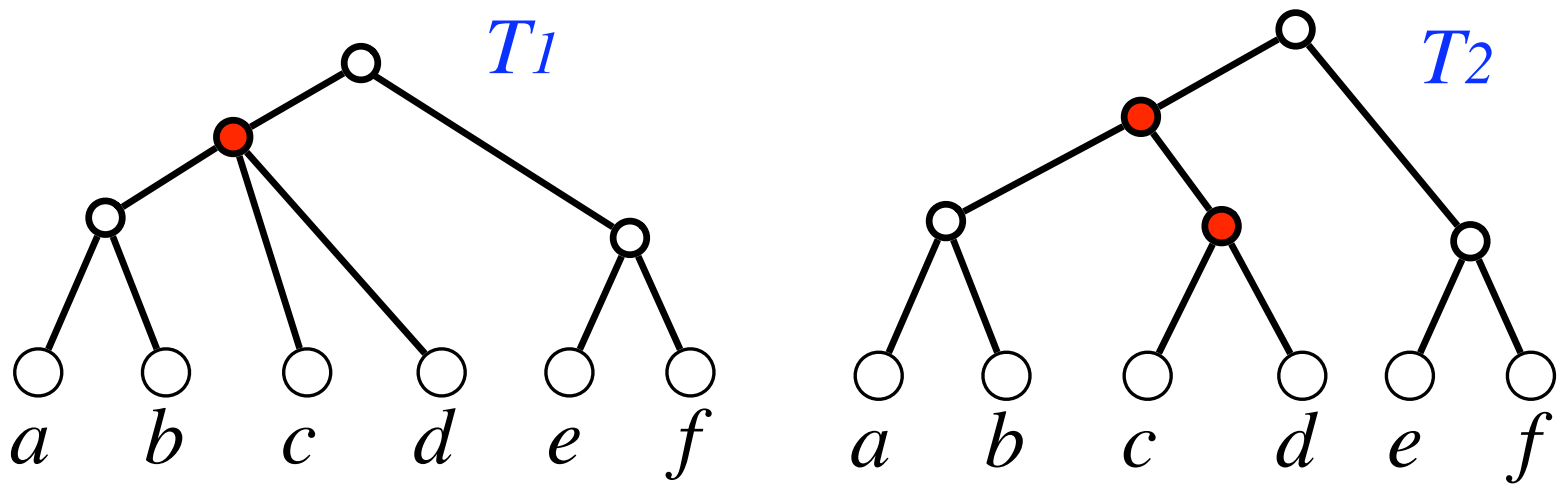


[Chateau 07]

Deux problèmes algorithmiques simples

Compatibilité d'arbres

- L'évolution est supposée se dérouler suivant un schéma arboré binaire.
- Les noeuds non-binaires sont considérés comme traduisant une incertitude. Ils sont nommés multifourches ou *soft polytomies*
- Deux arbres sont compatibles ssi il existe un arbre qui contient tous leurs clades.

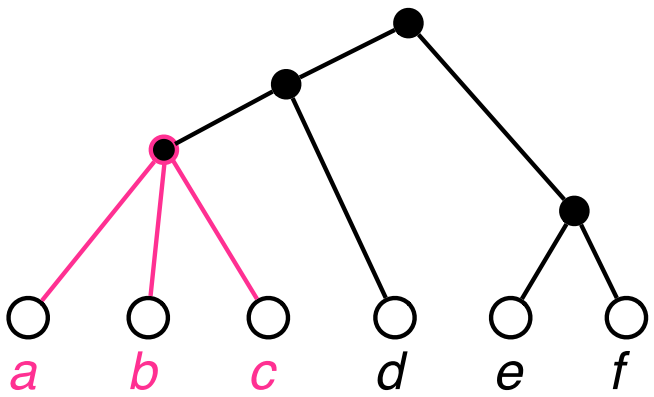


Triplets et fans

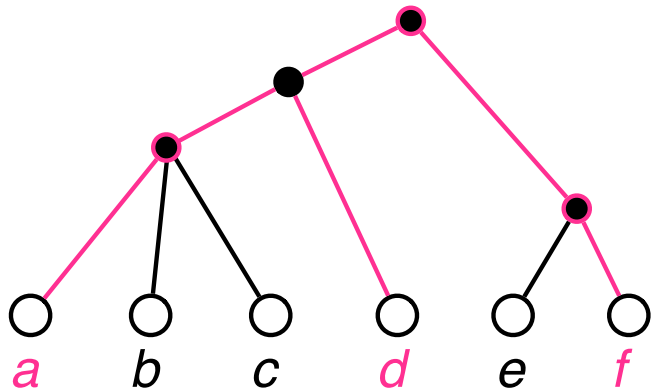
Des objets bien connus dans le domaine :

Définitions

Sur trois 3 éléments a, b, c de L, un arbre peut soit former des triplets enracinés (notés $ab|c$, $ac|b$ et $bc|a$), soit proposer un fan (noté $\{a, b, c\}$).



un fan $\{a,b,c\}$



un triplet $adlf$

Conflits

Définitions

Soit T_1, T_2 des arbres sur un ensemble L d'étiquettes.

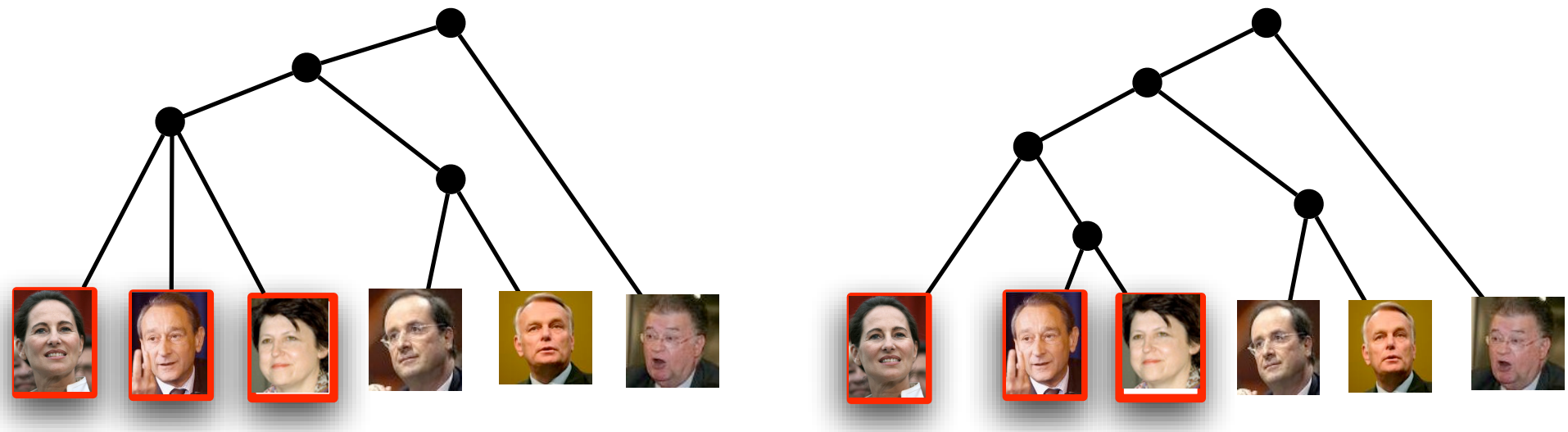
- Un conflit mou entre T_1 et T_2 (étiquetés par L) est un sous-ensemble a, b, c de L tel que
  $\{a, b, c\} \in f(T_1)$ et $ab|c \in tr(T_2)$ (ou réciproquement).
- Un conflit dur entre T_1 et T_2 est un sous-ensemble de trois éléments de L tel que
  $ab|c \in tr(T_1)$ et $ac|b \in tr(T_2)$.

Conflits

Définitions

Soit T_1, T_2 des arbres sur un ensemble L d'étiquettes.

- Un conflit mou entre T_1 et T_2 (étiquetés par L) est un sous-ensemble a, b, c de L tel que
  $\{a, b, c\} \in f(T_1)$ et $ab|c \in tr(T_2)$ (ou réciproquement).
- Un conflit dur entre T_1 et T_2 est un sous-ensemble de trois éléments de L tel que
  $ab|c \in tr(T_1)$ et $ac|b \in tr(T_2)$.



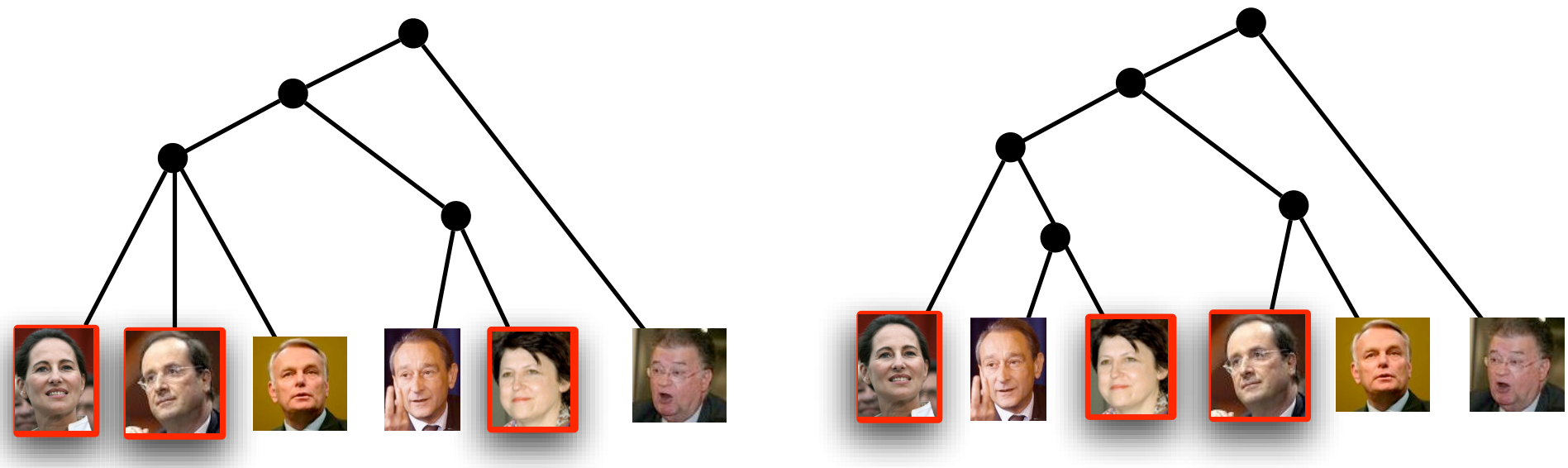
un conflit mou dans une famille bien connue

Conflits

Définitions

Soit T_1, T_2 des arbres sur un ensemble L d'étiquettes.

- Un conflit mou entre T_1 et T_2 (étiquetés par L) est un sous-ensemble a, b, c de L tel que
  $\{a, b, c\} \in f(T_1)$ et $ab|c \in tr(T_2)$ (ou réciproquement).
- Un conflit dur entre T_1 et T_2 est un sous-ensemble de trois éléments de L tel que
  $ab|c \in tr(T_1)$ et $ac|b \in tr(T_2)$.



un conflit dur dans une famille bien connue

Algorithmes de décision certifiants

Observations

- Deux arbres T_1 et T_2 sont isomorphes ssi ils n'ont pas de conflits durs ou mous.
- Deux arbres T et T' sont compatibles ssi ils n'ont pas de conflits dur.

Objectif

Un algorithme prenant en entrée deux arbres enracinés sur le même ensemble de feuilles et répondant OUI ssi les deux arbres sont isomorphes.

Certificat

Si l'algorithme répond NON, on veut qu'il nous produise la raison du non-isomorphisme, *i.e.* un conflit entre les deux arbres.

Résultats précédemment connus

Complexité connues pour deux arbres à n feuilles :

1. Un algorithme linéaire mais non-certifiant pour l'isomorphisme [Gusfield 81]
2. Un algorithme linéaire mais non-certifiant et procédant en plusieurs passes pour la compatibilité [Warnow 94]
3. Un algorithme certifiant en $O(n \log n)$ pour décider l'isomorphisme [Downey et al 99]
4. Un algorithme certifiant en $O(n^2)$ pour décider la compatibilité [Ganapathy et Warnow 01]

Question (back in 2003)

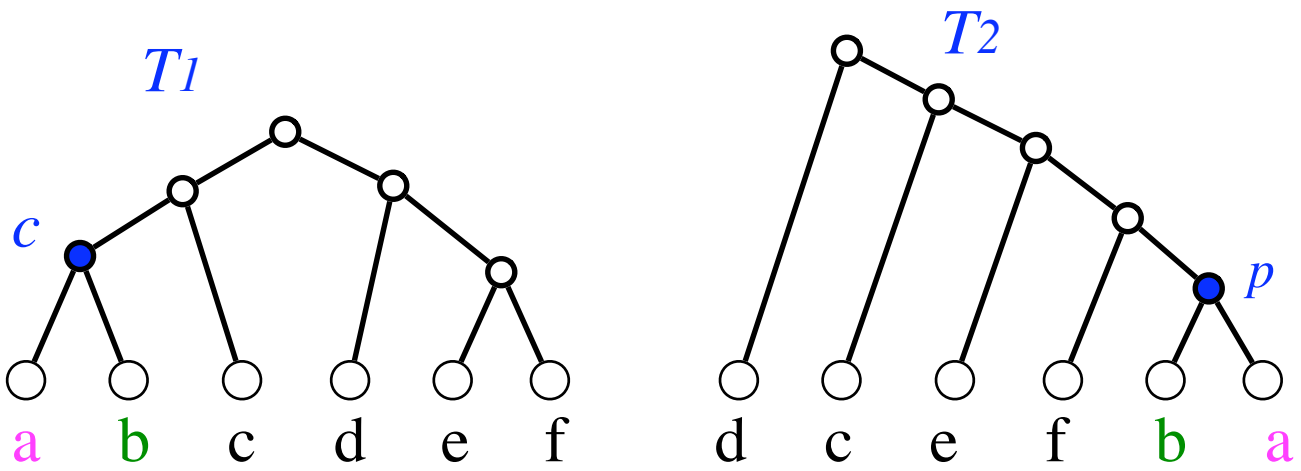
Existence d'un algorithme linéaire **et** certifiant, pour l'isomorphisme **et** pour la compatibilité ?

Algorithme certifiant linéaire d'*isomorphisme-ou-conflit*

Principe : parcours bottom-up en *grignotant* les cerises.

Propriété

Si on trouve deux cerises c et p identiques dans les deux arbres, alors on peut réduire les arbres en remplaçant les cerises par une nouvelle feuille.

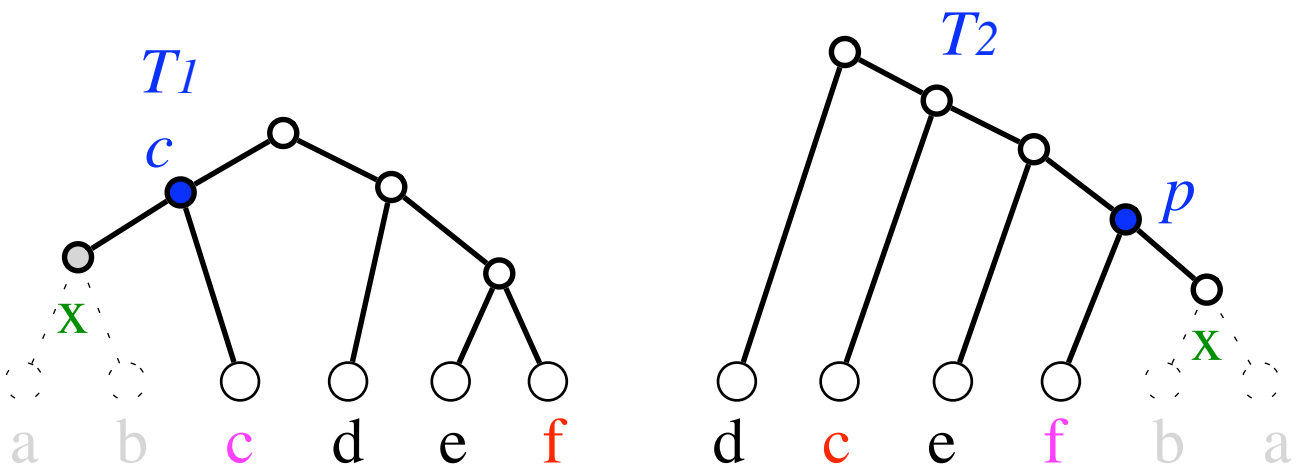


Algorithme certifiant linéaire d'*isomorphisme-ou-conflit*

Principe : parcours bottom-up en *grignotant* les cerises.

Propriété

Si $c = \{x, c\}$ *est une cerise et* $p = pere_{T_2}(x)$ *est tel que* $L(p) \neq \{x, c\}$ *alors alors on peut identifier facilement un conflit* $\{x, c, f : f \in L(p)\}$.

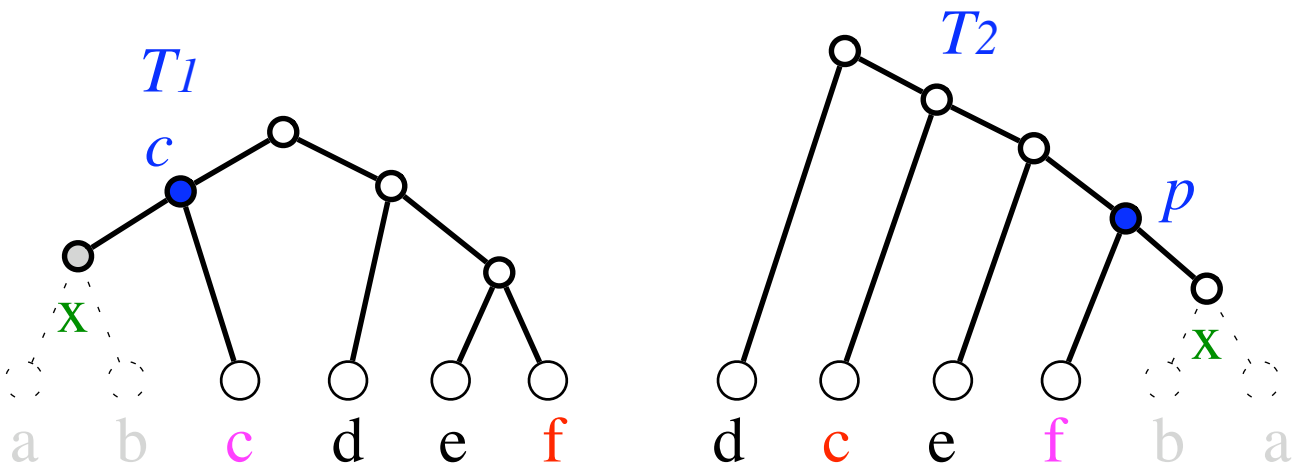


Algorithme certifiant linéaire d'*isomorphisme-ou-conflit*

Principe : parcours bottom-up en *grignotant* les cerises.

Propriété

Si $c = \{x, c\}$ est une cerise et $p = pere_{T_2}(x)$ est tel que $L(p) \neq \{x, c\}$ alors alors on peut identifier facilement un conflit $\{x, c, f : f \in L(p)\}$.



Complexité amortie en $O(n)$ pour deux arbres à n feuilles, en $O(kn)$ pour k arbres.

Extensions

Propriété

On peut garder une complexité linéaire dans différents cas :

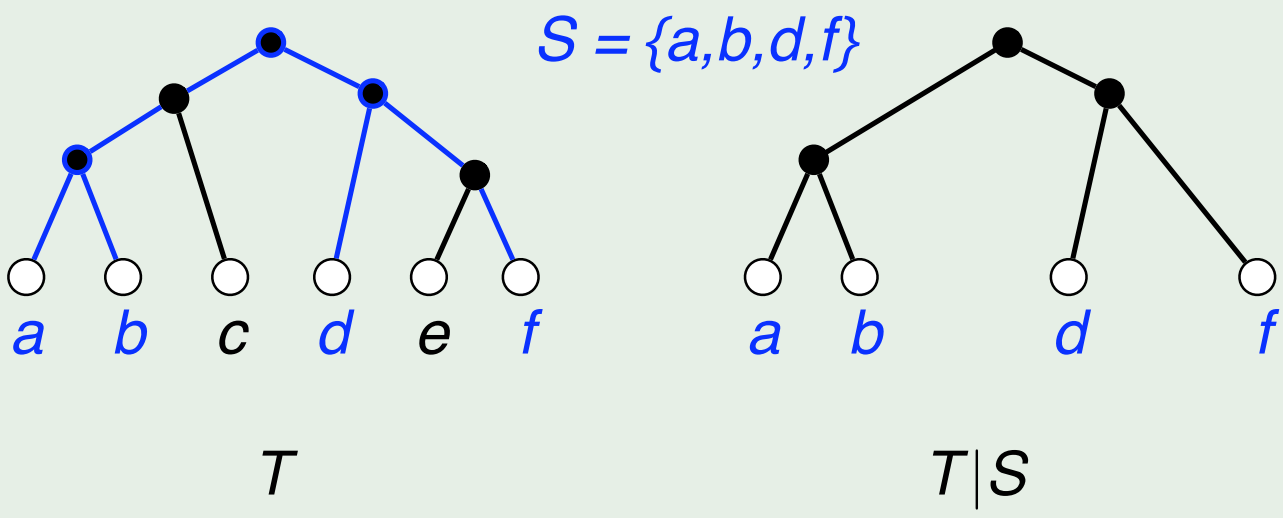
- arbres non-binaires et non-enracinés
- arbres dont les noeuds internes portent des étiquettes (taxonomies)
- arbres dont les fils d'un noeud sont ordonnées (codage de structure secondaire d'ARN)
- arbres pouvant contenir des étiquettes en plusieurs exemplaires (arbres de familles multigéniques) [SBR08]
- au cas de la compatibilité : réduire progressivement le nombre d'arbres en remplaçant deux arbres par leur raffinement commun minimum. [BN 04,BN 06]

Restriction d'un arbre à un ensemble de feuilles

Définition

Soit T un arbre et S un sous-ensemble de feuilles. La restriction de T à S, notée $T|S$, est le plus petit sous-arbre de T reliant les feuilles de S.

Exemple

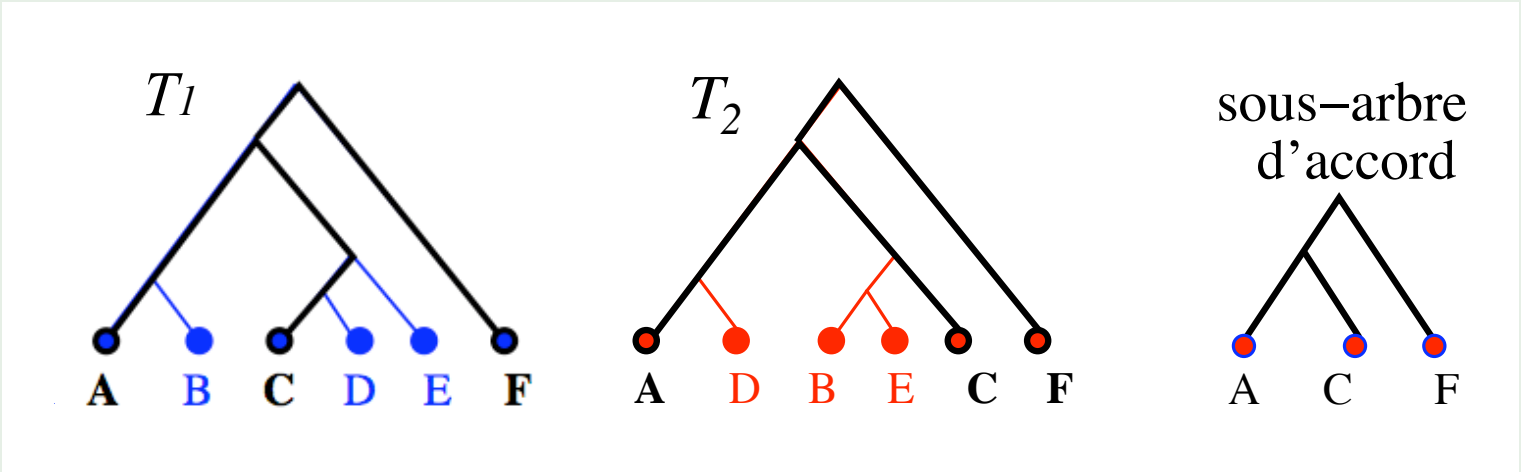


Définition

Soit $\mathcal{T}$ une collection d'arbres sur L.

- T est un sous-arbre d'accord de $\mathcal{T}$ ssi $L(T) \subseteq L$ et T est isomorphe à $T_i|L(T)$ pour tout arbre $T_i \in \mathcal{T}$
- T est un sous-arbre d'accord maximum (MAST) de $\mathcal{T}$ s'il possède un nombre maximum de feuilles.

Exemple

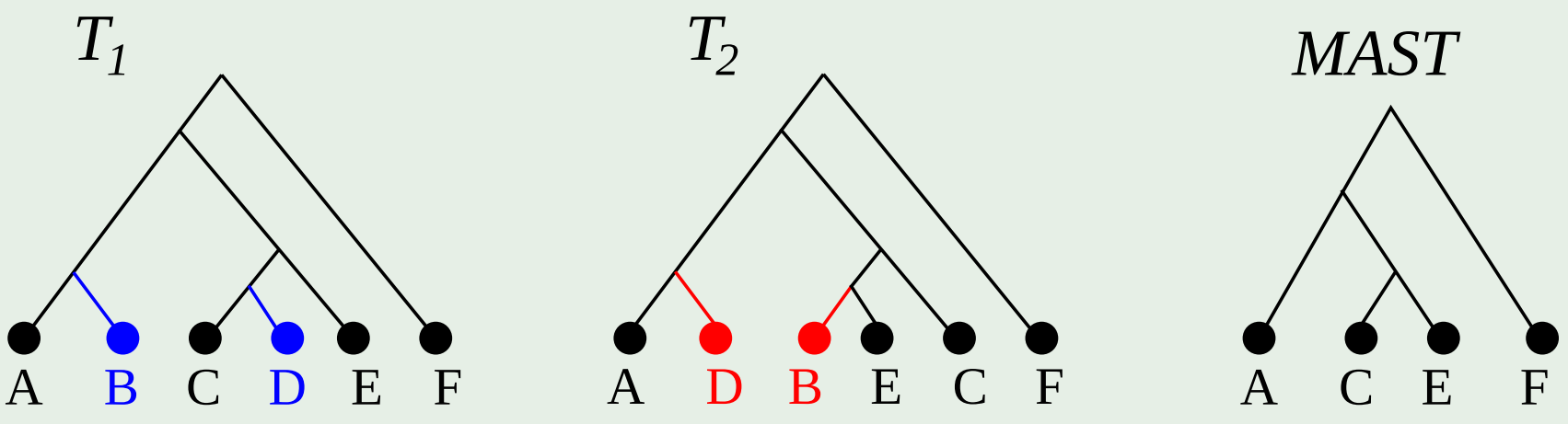


Définition

Soit $\mathcal{T}$ une collection d'arbres sur L.

- T est un sous-arbre d'accord de $\mathcal{T}$ ssi $L(T) \subseteq L$ et T est isomorphe à $T_i|L(T)$ pour tout arbre $T_i \in \mathcal{T}$
- T est un sous-arbre d'accord maximum (MAST) de $\mathcal{T}$ s'il possède un nombre maximum de feuilles.

Exemple



Résultats sur MAST

Etant donné une collection d'arbres $\mathcal{T}$ et soit

- n le nombre de feuilles,
- k le nombre d'arbres,
- d leur degré maximum,
- p le plus petit nombre de feuilles à enlever de $L(\mathcal{T})$ pour obtenir l'accord des arbres.

$k = 2$	$O(n^{1.5})$	[COLE et al 01, KAO et al 01]
$k > 2$	NP-difficile	[AMIR KESELMAN 97]
k qcq	3-approx en $O(kn)$	[BERRY et al 05]
FPT en d	$O(kn^3 + n^d)$	[FARACH 96, BRYANT 97]
FPT en p	$O(3^p kn)$	[BERRY NICOLAS 04]

(Un algorithme FPT a une complexité en $O(f(p) \times polynome(n))$)

Algorithme FPT pour MAST

Question :

étant donné une collection $\mathcal{T}$ d'arbres sur L, existe-t-il un MAST(T_1, T_2) obtenu en supprimant au plus p étiquettes de L ?

Rappel

Les arbres de $\mathcal{T}$ sont isomorphiques ssi ils n'ont pas de conflit dur ou mou sur trois feuilles

Corollaire

Si a, b, c est un conflit entre arbres de $\mathcal{T}$, alors

aucun MAST$(\mathcal{T})$ ne contient ces trois feuilles à la fois

(il faut enlever au moins une des trois feuilles pour obtenir l'isomorphisme, ie l'accord, des arbres de $\mathcal{T}$).

Algorithme FPT en p

Technique de la recherche bornée :

$MAST(\mathcal{T}, p)$

Résultat : Un MAST de $\mathcal{T}$ obtenu en enlevant au plus p feuilles ou $\emptyset$ si impossible

si *les arbres de* $\mathcal{T}$ *sont isomorphes* **alors**
  renvoyer n'importe quel $T \in \mathcal{T}$

sinon
  si $p = 0$ **alors** renvoyer $\emptyset$;
  $C \leftarrow$ IsomorphismeOuTrouverConflit($\mathcal{T}$);
  pour chaque *feuille* f *du conflit* C **faire**
    $T \leftarrow MAST(\mathcal{T} \mid (L(\mathcal{T}) - \{f\}), p - 1)$;
    si $T \neq \emptyset$ **alors** renvoyer T;
  renvoyer $\emptyset$ /* aucune solution viable */

Algorithme FPT en p

Technique de la recherche bornée :

$MAST(\mathcal{T}, p)$

Résultat : Un MAST de $\mathcal{T}$ obtenu en enlevant au plus p feuilles ou $\emptyset$ si impossible

si *les arbres de* $\mathcal{T}$ *sont isomorphes* **alors**
  renvoyer n'importe quel $T \in \mathcal{T}$

sinon
  si $p = 0$ **alors** renvoyer $\emptyset$;
  $C \leftarrow$ IsomorphismeOuTrouverConflit($\mathcal{T}$);
  pour chaque *feuille* f *du conflit* C **faire**
    $T \leftarrow MAST(\mathcal{T} \mid (L(\mathcal{T}) - \{f\}), p - 1)$;
    si $T \neq \emptyset$ **alors** renvoyer T;
  renvoyer $\emptyset$ /* aucune solution viable */

Algorithme FPT en p

Technique de la recherche bornée :

$MAST(\mathcal{T}, p)$

Résultat : Un MAST de $\mathcal{T}$ obtenu en enlevant au plus p feuilles ou $\emptyset$ si impossible

si *les arbres de* $\mathcal{T}$ *sont isomorphes* **alors**
  renvoyer n'importe quel $T \in \mathcal{T}$

sinon
  si $p = 0$ **alors** renvoyer $\emptyset$;
  $C \leftarrow$ IsomorphismeOuTrouverConflit($\mathcal{T}$);
  pour chaque *feuille* f *du conflit* C **faire**
    $T \leftarrow MAST(\mathcal{T} \mid (L(\mathcal{T}) - \{f\}), p - 1)$;
    si $T \neq \emptyset$ **alors** renvoyer T;
  renvoyer $\emptyset$ /* aucune solution viable */

Algorithme FPT en p

Technique de la recherche bornée :

$MAST(\mathcal{T}, p)$

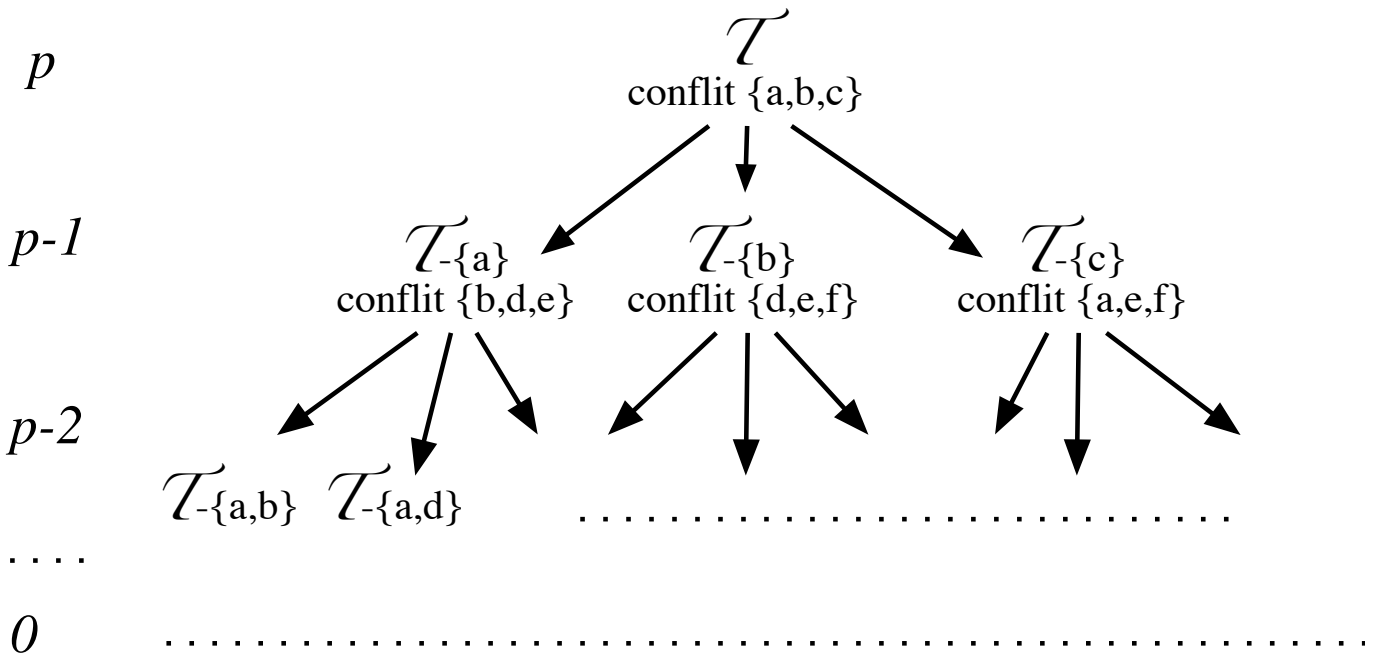
Résultat : Un MAST de $\mathcal{T}$ obtenu en enlevant au plus p feuilles ou $\emptyset$ si impossible

si *les arbres de* $\mathcal{T}$ *sont isomorphes* **alors**
  renvoyer n'importe quel $T \in \mathcal{T}$

sinon
  si $p = 0$ **alors** renvoyer $\emptyset$;
  $C \leftarrow$ IsomorphismeOuTrouverConflit($\mathcal{T}$);
  pour chaque *feuille* f *du conflit* C **faire**
    $T \leftarrow MAST(\mathcal{T} \mid (L(\mathcal{T}) - \{f\}), p - 1)$;
    si $T \neq \emptyset$ **alors** renvoyer T;
  renvoyer $\emptyset$ /* aucune solution viable */

Complexité

Arbre des appels récursifs :



- Chaque appel récursif peut engendrer 3 nouveaux appels.
- A chaque appel, la valeur de p diminue de 1, donc l'arbre des appels est de profondeur $O(p)$
- l'arbre des appels comporte $O(3^p)$ noeuds, donc autant d'exécutions de la fonction $MAST(\mathcal{T}, p)$.

Complexité

Coût de chaque exécution de la fonction $MAST(\mathcal{T}, p)$:

Restrictions = simples parcours d'arbres : $O(k)$

---

si *les arbres de* $\mathcal{T}$ *sont isomorphes* **alors**
  renvoyer n'importe quel $T \in \mathcal{T}$

sinon
  si $p = 0$ **alors** renvoyer $\emptyset$;
  $C \leftarrow$ IsomorphismeOuTrouverConflit($\mathcal{T}$);
  pour chaque *feuille* f *du conflit* C **faire**
    $T \leftarrow MAST(\mathcal{T} \mid (L(\mathcal{T}) - \{f\}), p - 1)$;
    si $T \neq \emptyset$ **alors** renvoyer T;
  renvoyer $\emptyset$

Chaque exécution coûte $O(kn)$ donc l'algorithme est au total en $O(3^p kn)$. Algorithme exponentiel utilisable en pratique ([DEVIENNE et al 07]).

Improved parametrized complexity of Maximum Agreement Subtree and Maximum Compatible Tree problems, V. Berry et F. Nicolas, IEEE/ACM Trans. on Comp. Biol. and Bioinf., 3(3), 289–302, 2006.

Complexité

Coût de chaque exécution de la fonction $MAST(\mathcal{T}, p)$:

k applications de l'alg. certifiant d'isomorphisme : $O(kn)$

---

si *les arbres de* $\mathcal{T}$ *sont isomorphes* **alors**
  renvoyer n'importe quel $T \in \mathcal{T}$

sinon
  si $p = 0$ **alors** renvoyer $\emptyset$;
  $C \leftarrow$ IsomorphismeOuTrouverConflit($\mathcal{T}$);
  pour chaque *feuille* f *du conflit* C **faire**
    $T \leftarrow MAST(\mathcal{T} \mid (L(\mathcal{T}) - \{f\}), p - 1)$;
    si $T \neq \emptyset$ **alors** renvoyer T;
  renvoyer $\emptyset$

Chaque exécution coûte $O(kn)$ donc l'algorithme est au total en $O(3^p kn)$. Algorithme exponentiel utilisable en pratique ([DEVIENNE et al 07]).

Improved parametrized complexity of Maximum Agreement Subtree and Maximum Compatible Tree problems, V. Berry et F. Nicolas, IEEE/ACM Trans. on Comp. Biol. and Bioinf., 3(3), 289–302, 2006.

Complexité

Coût de chaque exécution de la fonction $MAST(\mathcal{T}, p)$:

---

si *les arbres de* $\mathcal{T}$ *sont isomorphes* **alors**
  renvoyer n'importe quel $T \in \mathcal{T}$

sinon
  si $p = 0$ **alors** renvoyer $\emptyset$;
  $C \leftarrow$ IsomorphismeOuTrouverConflit($\mathcal{T}$);
  pour chaque *feuille* f *du conflit* C **faire**
    $T \leftarrow MAST(\mathcal{T} \mid (L(\mathcal{T}) - \{f\}), p - 1)$;
    si $T \neq \emptyset$ **alors** renvoyer T;
  renvoyer $\emptyset$

Chaque exécution coûte $O(kn)$ donc l'algorithme est au total en $O(3^p kn)$. Algorithme exponentiel utilisable en pratique ([DEVIENNE et al 07]).

Improved parametrized complexity of Maximum Agreement Subtree and Maximum Compatible Tree problems, V. Berry et F. Nicolas, IEEE/ACM Trans. on Comp. Biol. and Bioinf., 3(3), 289–302, 2006.

Extensions

- Le problème MAST sur k arbres peut être 3-approximé (par son complémentaire) en temps linéaire
- Le superarbre d'accord maximum (SMAST) de deux arbres peut être calculé en temps linéairement proportionnel au temps nécessaire pour calculer le MAST de deux arbres
- Le problème SMAST sur k arbres binaires peut être résolu en temps $O((6n)^k)$ ou en temps $O((2k)^p kn^2)$ (FPT)
- Divers résultats de difficulté.

– *Maximum Agreement Supertree, Berry et Nicolas, Journal of Discrete Algorithms, 2007.*

– *Fixed-Parameter Tractability of the Maximum Agreement Supertree Problem, Guillemot et Berry, IEEE/ACM Trans. Comp. Biol. and Bioinf., 2008.*

– *Linear time 3-approximation for the MAST problem, Berry, Guillemot, Nicolas et Paul, ACM Transaction on Algorithms (TALG), 2009.*

Plan

Le challenge de M. Sanderson

Défi (2001) :

> *Sanderson et le consortium Deeep Green ne réussissent pas à combiner les arbres de TreeBASE pour faire de l'inférence de superarbre. Ils mettent la communauté scientifique au défi.*

Deux problèmes :

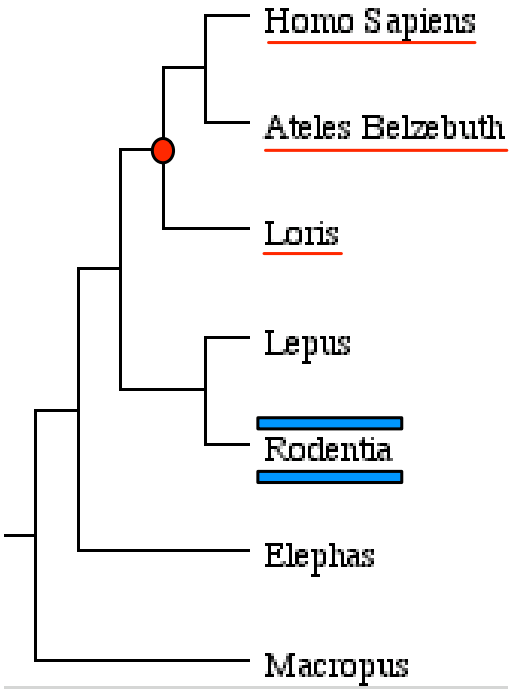
- l'inconsistance taxonomique :

  $$\textit{Rongeur} \neq \textit{Rongeur} \text{ et } \textit{Rongeur} = \textit{Souris}$$

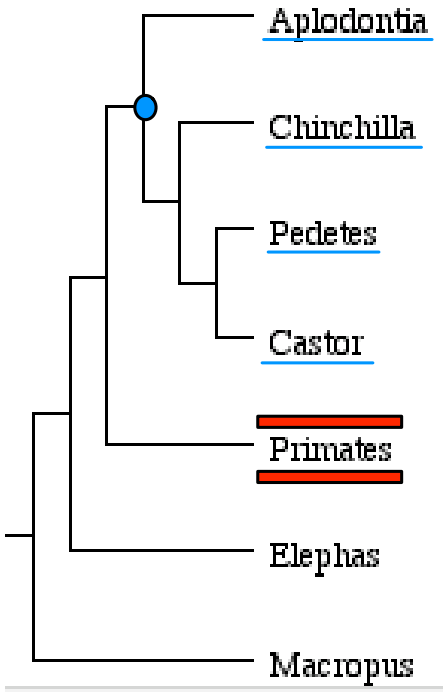
- l'inclusion taxononomique : $\textit{Castor} \subseteq \textit{Rongeur}$

non prise en compte de niveaux taxonomiques différents.

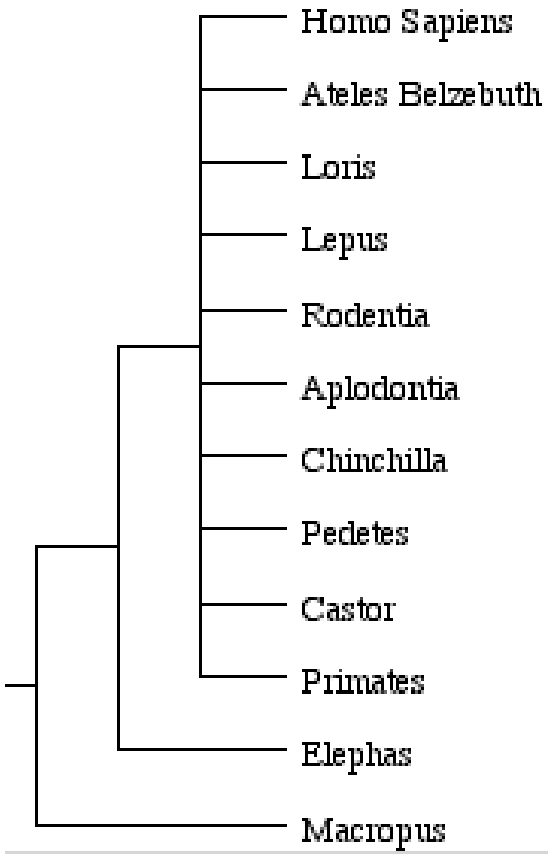
Inclusion taxonomique



arbre source 1



arbre source 2



superarbre MRP

$\Rightarrow$ Les inclusions taxonomiques nécessitent un traitement spécifique si l'on veut obtenir un résultat sensé

Une première solution

Conserver des étiquettes uniquement aux feuilles des arbres, en remplaçant chaque occurence d'un taxon de niveau supérieur par un taxon de plus bas niveau, bien choisi.

Dans l'exemple précédent ceci peut conduire à remplacer :

- le taxon *Rodentia* par le taxon *Castor*
- le taxon *Primates* par le taxon *Gorilla*

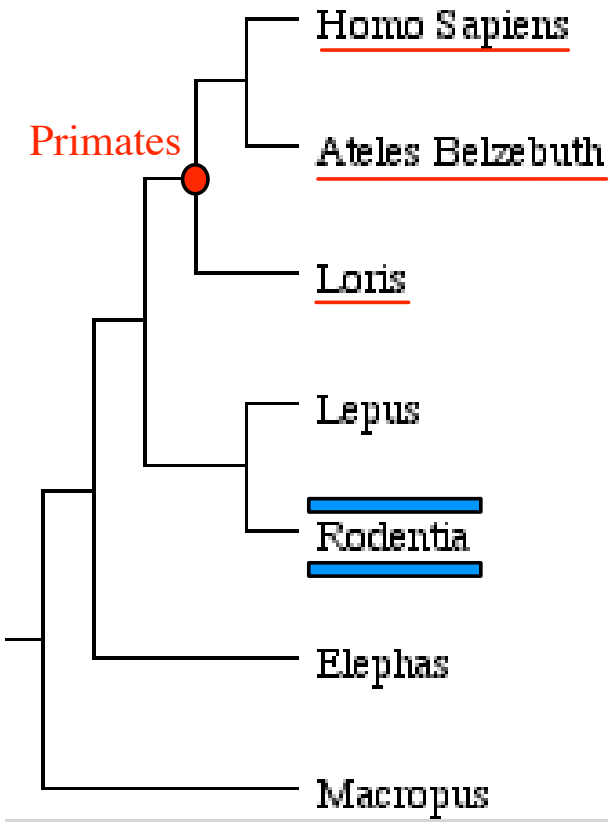
Problèmes :

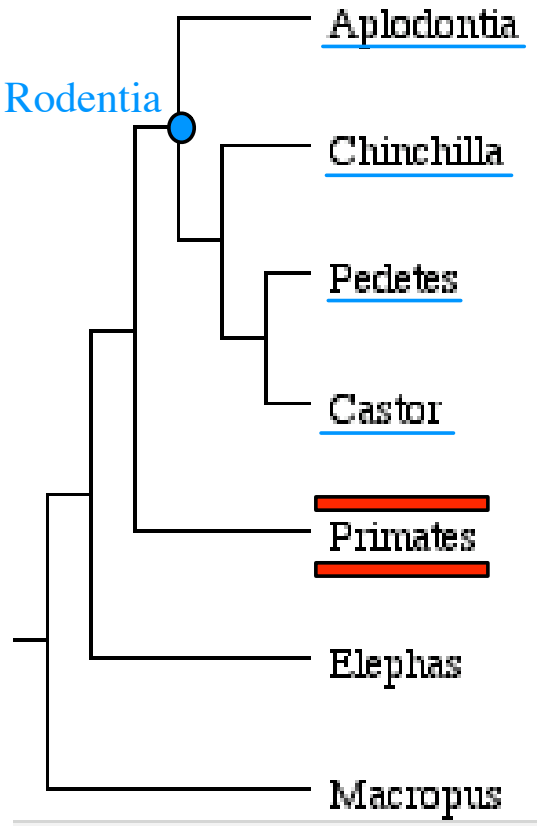
- Le choix est difficile.
  Le retour aux séquences est fastidieux est échoue quand une séquence consensus a été utilisée
- Le choix est important !
  Suivant le représentant choisi, on n'a pas forcément la même résolution ni un chevauchement suffisant entre arbres sources pour inférer un super-arbre informatif.

Une deuxième solution

Préciser les taxons aux noeuds internes : indique les parentés entre taxons quand il en existe une.

Exemple sur le cas des rongeurs et des primates :





Une deuxième solution

Avantage 1 : information facile à trouver :

- le biologiste qui construit une phylogénie ou un super-arbre dispose de cette connaissance.
- pour les phylogénies des BDs, depuis une liste de taxa, on peut retrouver de façon automatique cette information dans les taxonomies publiques
  (*Catalog of Life, Mammals of the world, NCBI*, etc)

Avantage 2 : pas de choix de taxons à effectuer : toutes les inclusions peuvent être précisées sans explosion combinatoire.

Avantage 3 : l'information croisée devient beaucoup plus importante. On perd moins de résolution.

Conclusion :

besoin d'une méthode de super-arbres acceptant plusieurs niveaux taxonomiques

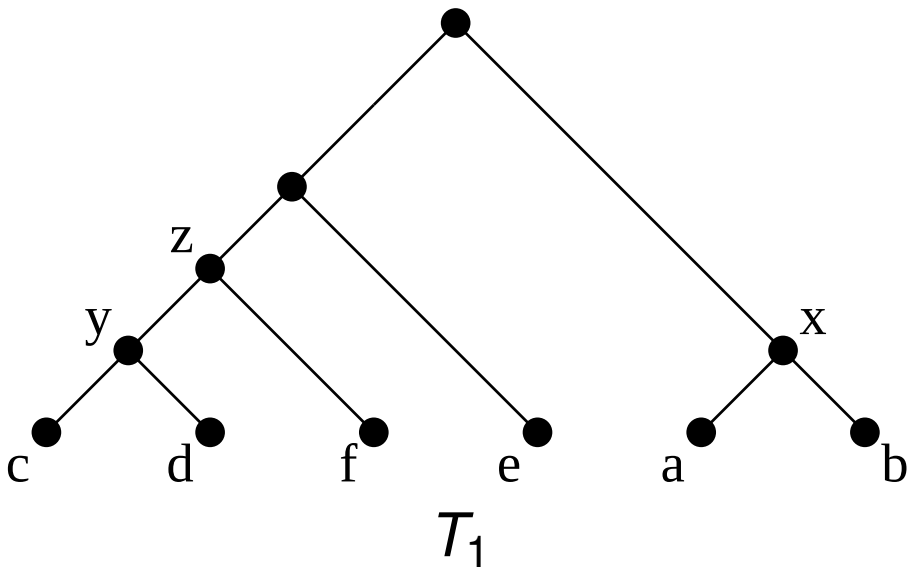
Compatibilité d'arbres semi-étiquetés

Définitions

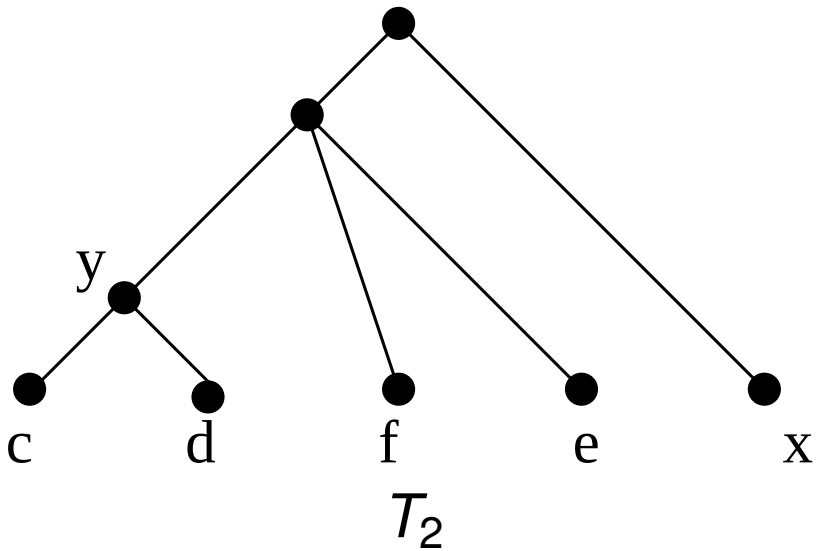
Un arbre semi-étiqueté est un arbre dont toutes les feuilles, **ainsi** que certains noeuds internes, sont étiquetés

Soit T_1, T_2 deux arbres semi-étiquetés, T_1 contient (displays) T_2 ssi

- T_1 restreint aux étiquettes de T_2 contient tous les clades de T_2.
- toutes les relations ancêtre/descendant observées dans T_2 entre étiquettes sont présentes dans T_1.



T_1

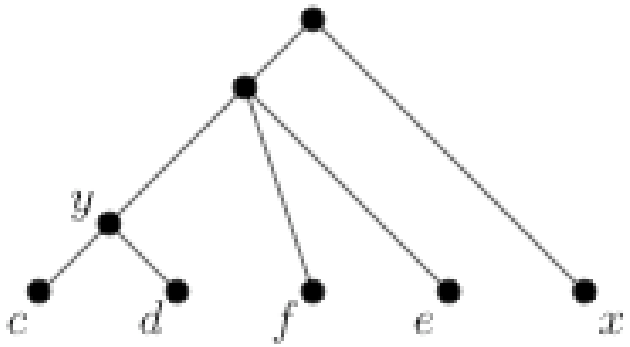


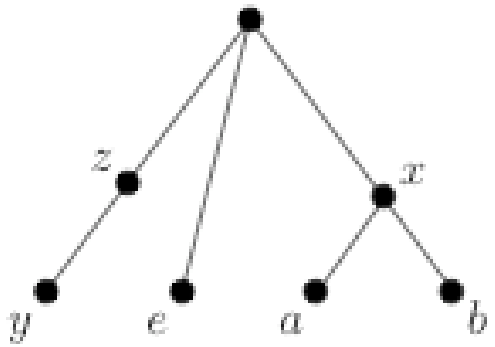
T_2

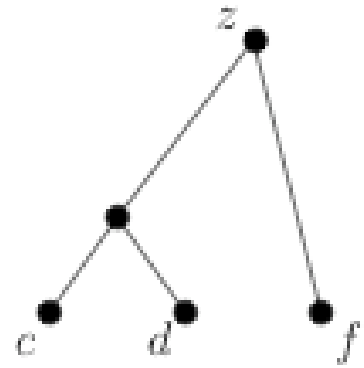
Compatibilité ancestrale

Définition

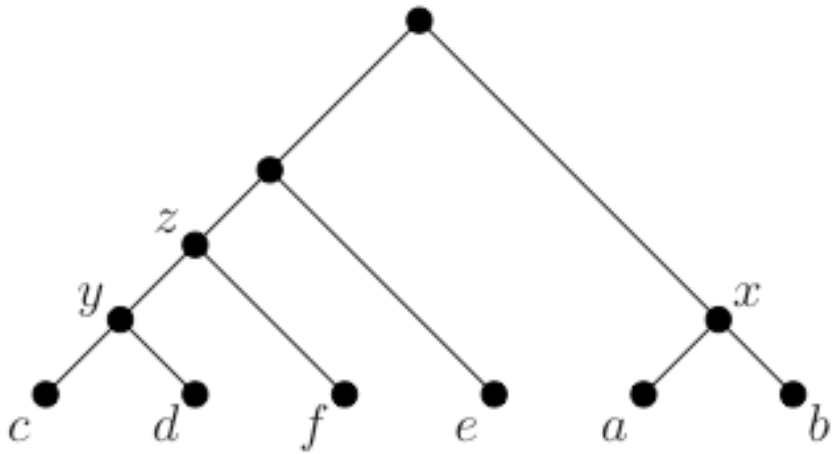
Une collection $\mathcal{T}$ d'arbres semi-étiquetés est ancestralement compatible ssi il existe un arbre T contenant chaque $T_i \in \mathcal{T}$.







Pour la collection $\mathcal{T}$ ci-dessus, l'arbre T suivant convient :



Inspirations

Question :

un algorithme décidant la compatibilité ancestrale d'une collection d'arbres semi-étiquetés

Arbres étiquetés aux feuilles uniquement

- [AHO et al 81] proposent (dans le domaine des bases de données) le célèbre algorithme BUILD basé sur un codage des arbres en triplets, de complexité $O(kn^3)$.
- [HENZINGER et al 96] améliorent la complexité à $O(kn\sqrt{n})$ mais en se limitant aux arbres binaires (codage économique en triplets).

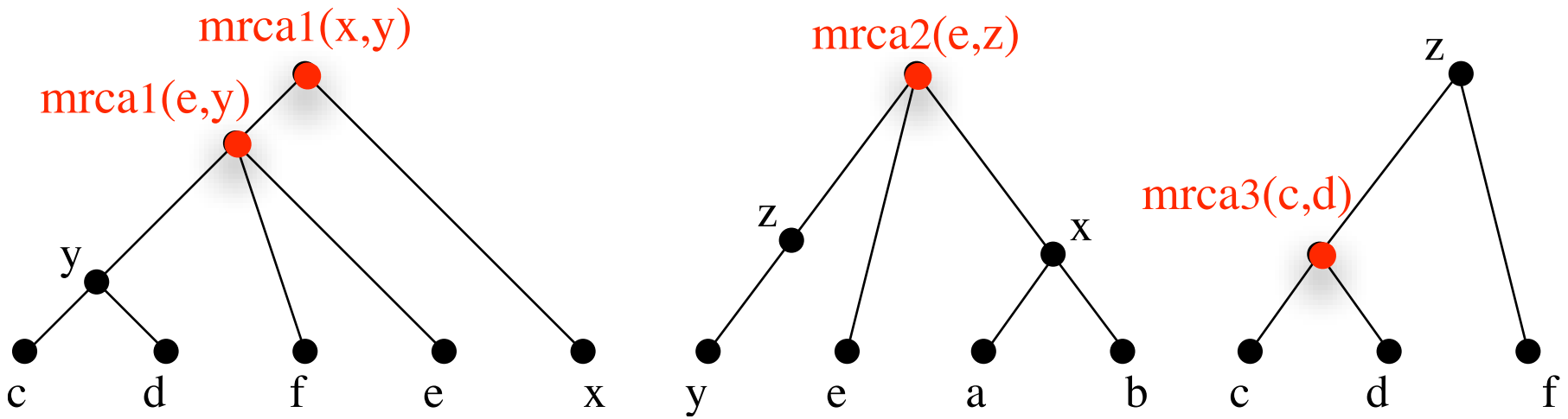
Arbres étiquetés aux noeuds et aux feuilles

- [DANIEL SEMPLE 04] définissent un autre graphe à décomposer, mais très dense : complexité en $O(k^2n^3)$.

L'algorithme *Ancestral Build**

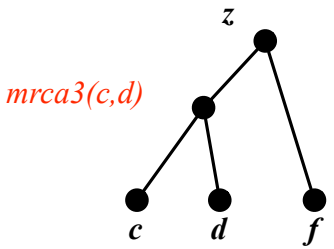
Prétraitement

Soit $\mathcal{T}$ une collection d'arbres semi-étiquetés, on commence par ajouter des étiquettes (différentes) à tous les noeuds qui n'en ont pas, de façon à les repérer.

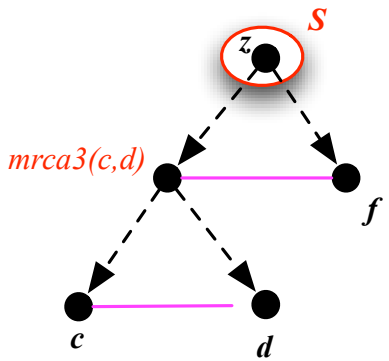


Principe de l'algorithme

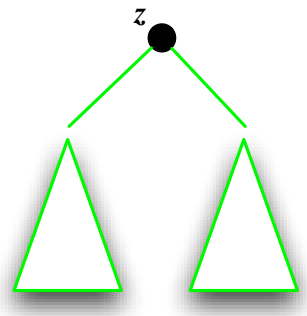
2 - On encode dans un graphe les relations d'ancestralité entre labels.



arbre source



graphe

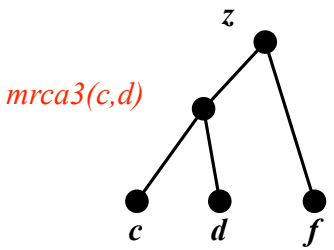


superarbre reconstruit

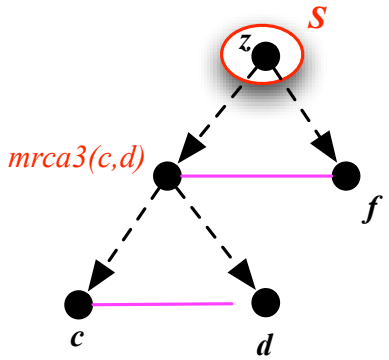
- on ajoute un **arc** de l'étiquette a vers b si a est le père de b dans un même arbre source $T_i \in \mathcal{T}$.
- on ajoute une arête entre a et b s'ils sont frère dans un même arbre source.

Principe de l'algorithme

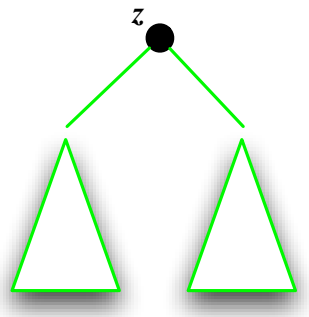
2 - On encode dans un graphe les relations d'ancestralité entre labels.



arbre source



graphe



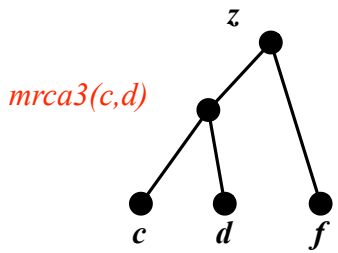
superarbre reconstruit

3- On décompose récursivement le graphe :

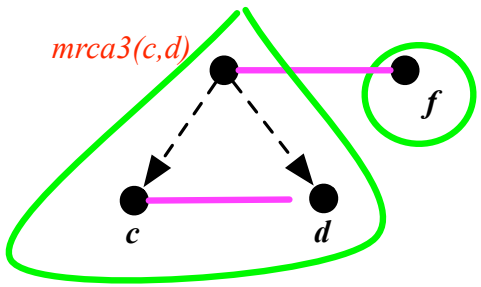
- les sommets n'ayant pas d'ancêtre ($\mathcal{S}_0$) sont la racine du superarbre reconstruit
- les sommets liés par des arêtes aux autres, n'ont pas le droit de faire partie de la racine
- une fois enlevés les sommets utilisés pour la racine, les arêtes entre arc composantes connexes peuvent être enlevées
- des appels récursifs sur les arc-composantes connexes donneront les sous-arbres du superarbre

Principe de l'algorithme

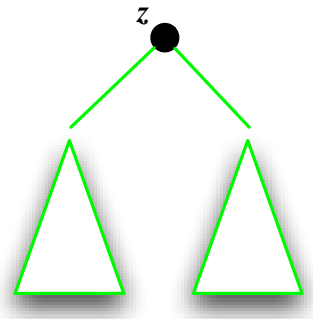
2 - On encode dans un graphe les relations d'ancestralité entre labels.



arbre source



graphe



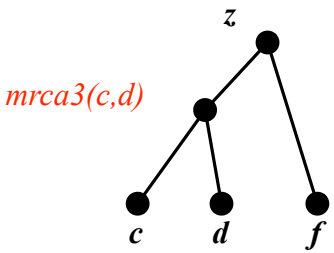
superarbre reconstruit

3- On décompose récursivement le graphe :

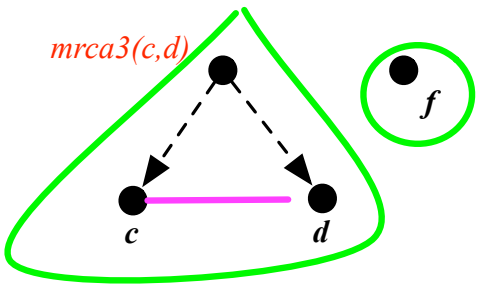
- les sommets n'ayant pas d'ancêtre ($\mathcal{S}_0$) sont la racine du superarbre reconstruit
- les sommets liés par des arêtes aux autres, n'ont pas le droit de faire partie de la racine
- une fois enlevés les sommets utilisés pour la racine, les arêtes entre arc composantes connexes peuvent être enlevées
- des appels récursifs sur les arc-composantes connexes donneront les sous-arbres du superarbre

Principe de l'algorithme

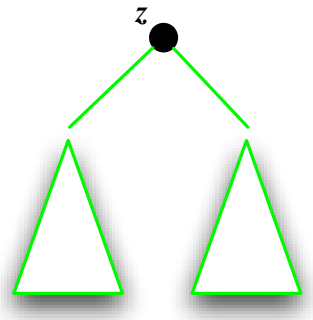
2 - On encode dans un graphe les relations d'ancestralité entre labels.



arbre source



graphe

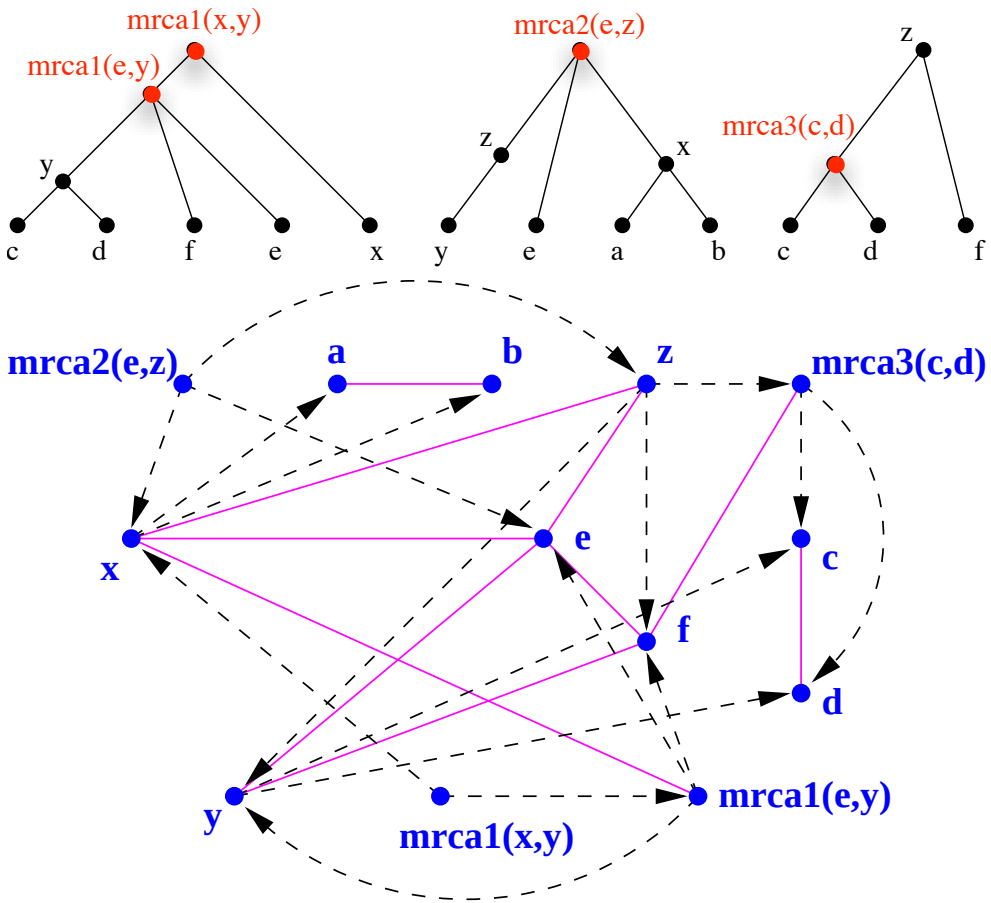


superarbre reconstruit

3- On décompose récursivement le graphe :

- les sommets n'ayant pas d'ancêtre ($\mathcal{S}_0$) sont la racine du superarbre reconstruit
- les sommets liés par des arêtes aux autres, n'ont pas le droit de faire partie de la racine
- une fois enlevés les sommets utilisés pour la racine, les arêtes entre arc composantes connexes peuvent être enlevées
- des appels récursifs sur les arc-composantes connexes donneront les sous-arbres du superarbre

Exemple sur l'ensemble des arbres sources



Fast Computation of Supertrees for Compatible Phylogenies with Nested Taxa, V. Berry et C. Semple, Systematic Biology, 55(2), U108–U126, 2006.

Décomposition du graphe $\mathcal{D}$

Algorithme :

1. Soit S_0 l'ensemble des noeuds de $\mathcal{D}$ qui n'ont aucune arête entrante et ne sont incidents à aucun arc.
2. Si S_0 est vide alors renvoyer "`incompatible`" (aucun arbre ne peut respecter les contraintes topologiques imposées par les arbres sources dans leur ensemble).
3. Si S_0 contient un seul noeud alors renvoyer l'arbre composé d'une feuille unique ayant l'étiquette de ce noeud.
4. Sinon,
   1. Supprimer les noeuds de S_0 (et leurs arcs incidents) de $\mathcal{D}$
   2. Identifier $S_1, \ldots, S_r$ les arcs-composantes connexes de $\mathcal{D}$.
   3. Supprimer toute arête dont les extrémités sont dans deux composantes S_i, S_j différentes
   4. Pour chaque S_i ($i \in 1..r$) faire un appel récursif sur S_i.
   5. Si un de ces appels renvoye "`incompatible`" alors renvoyer "`incompatible`".
   6. Sinon attacher les sous-arbres renvoyés par les appels récursifs à la racine et renvoyer l'arbre

Complexité

k arbres n feuilles m noeuds au total de degré maximum d

- *AncestralBuild** tourne en $O(\log^2(n)kd^2)$ en général et en $O(m\log^2(n))$ sur les arbres binaires, soit aussi bien que [Henzinger et al 96] mais pour des arbres plus généraux.
- En conclusion, la compatibilité de phylogénies enracinées peut être décidée en temps quasi-linéaire.

Application : une phylogénie des *Strepsirrhini*
(100 espèces, 10 niveaux taxonomiques)

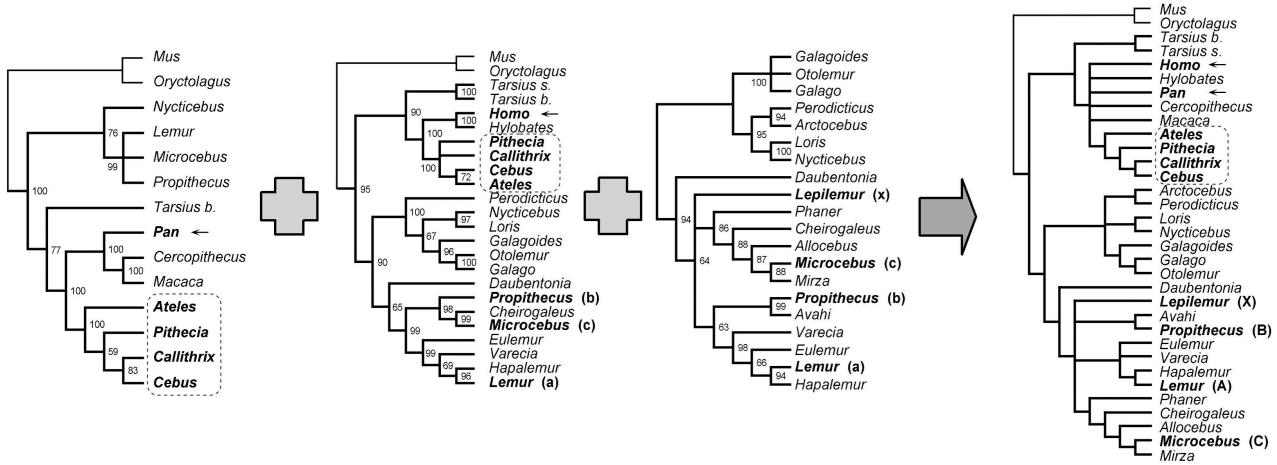
Implémentation
Algorithme implémenté dans *Splitstree v.4* [HUSON BRYANT 06]

Fast Computation of Supertrees for Compatible Phylogenies with Nested Taxa, V. Berry et C. Semple, Systematic Biology, 55(2), U108–U126, 2006.

Plan

Méthodes de superarbre

Données : ensemble de jeux de données → collections d'arbres



Intérêts des méthodes de superarbre :

- Combiner des données de nature hétérogène
- Obtenir une phylogénie depuis plusieurs arbres de gènes
- Préférables aux supermatrices quand trop de données manquantes
- Connaître les zones problématiques dans la phylogénie
- Mesurer le degré d'accord entre arbres d'une collection

Méthodes de Superarbres : ma démarche

Bibliographie (2001) :

- aucune méthode de superarbres ne fait l'unanimité.
- MRP, la plus utilisée, critiquée pour des soucis de biais d'échantillonnage et la proposition de clades contredisant tous les arbres sources

Ma démarche :

proposer de nouvelles méthodes de superarbres ayant

- *des propriétés combinatoires intéressantes*
- *des propriétés pratiques intéressantes*

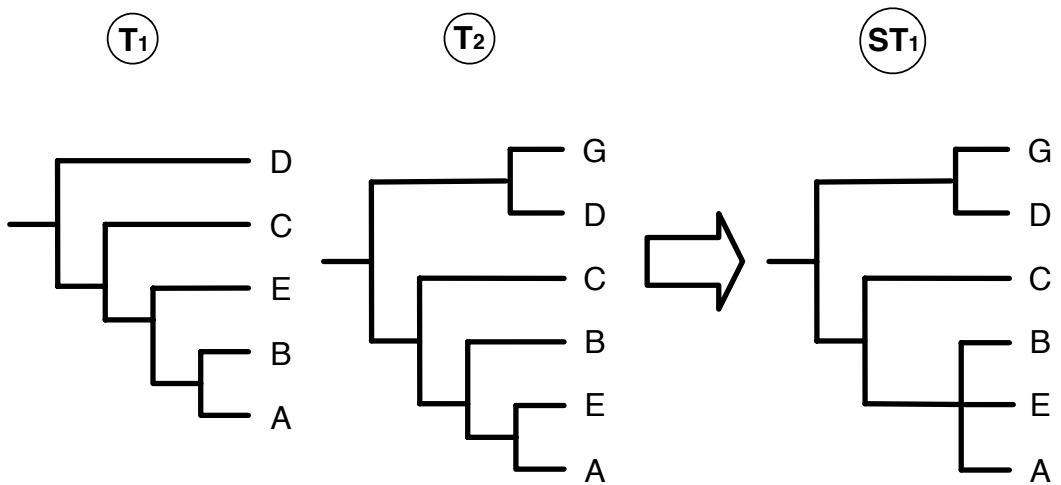
Inspirations :

- [Steel et al 02]
- [Wilkinson et al 01, 04]

Propriétés combinatoires intéressantes en contexte Veto

- Propriété de non-contradiction (PC) :

  Le superarbre proposé ne doit pas contenir de clade contredisant les relations des arbres sources :

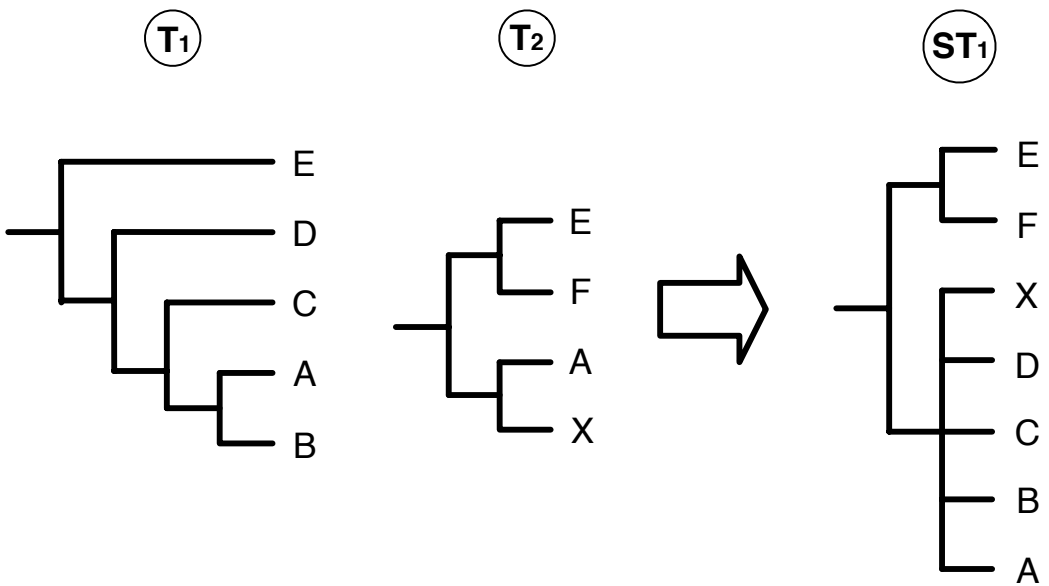


PhySIC : a Veto Supertree Method with Desirable Properties, V. Ranwez, V. Berry, A. Criscuolo, P.-H. Fabre, S. Guillemot, C. Scornavacca et E.J.P. Douzery, *Systematic Biology*, 56(5), 293-304, 2007.

Propriétés combinatoires intéressantes en contexte Veto

- Propriété d'induction (PI) :

  Le superarbre proposé doit uniquement contenir des clades présents dans les arbres sources ou induits collectivement par ces arbres

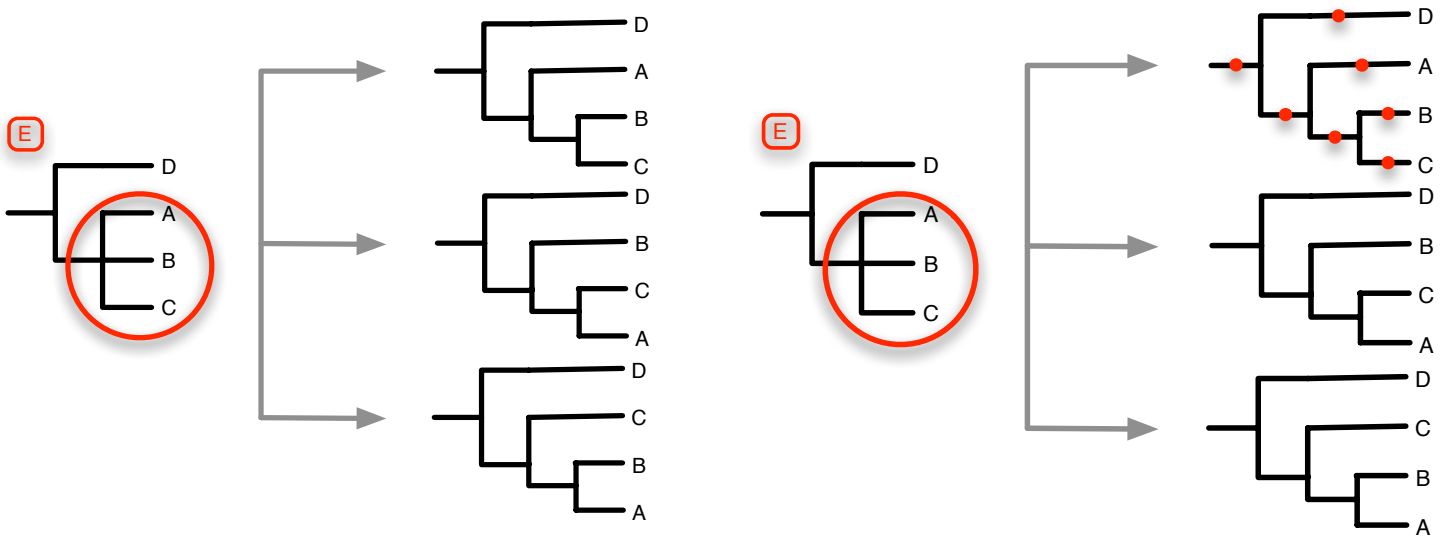


PhySIC : a Veto Supertree Method with Desirable Properties, V. Ranwez, V. Berry, A. Criscuolo, P.-H. Fabre, S. Guillemot, C. Scornavacca et E.J.P. Douzery, *Systematic Biology*, 56(5), 293-304, 2007.

Propriété pratique intéressante dans tout contexte

- Critère d'informativité (CIC) :
  - Essayer d'obtenir un superarbre aussi informatif que possible : degré de résolution & d'inclusion de taxa initaux.
  - On généralise le critère CIC proposé par [THORLEY ET AL 98] qui prend aussi en compte le fait que des taxa peuvent ne pas avoir été inclus dans un superarbre proposé :

$$CIC(T,n) = -\lg \frac{\text{nb de phyl. binaires complètes } \textit{contenant} \text{ le superarbre } T}{\text{nb de phylogénies existant sur } n \text{ taxa}}$$



Résultats combinatoires

En se basant sur une expression des propriétés PI et PC en termes de triplets des arbres sources et des superarbres candidats, on montre :

Théorème

- Etant donné un superarbre T pour une collection $\mathcal{T}$, on peut vérifier en temps polynomial s'il vérifie PI et PC.
- Il existe un algorithme polynomial inférant un superarbre vérifiant toujours PI et PC pour une collection $\mathcal{T}$ donnée.

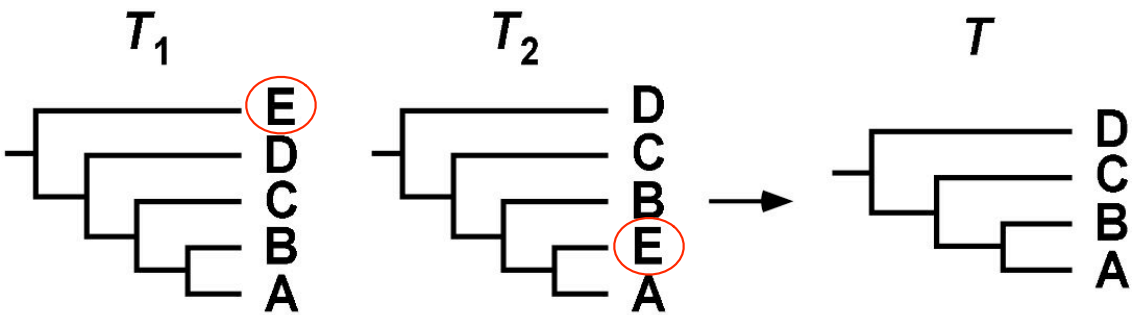
PhySIC : a Veto Supertree Method with Desirable Properties, V. Ranwez, V. Berry, A. Criscuolo, P.-H. Fabre, S. Guillemot, C. Scornavacca et E.J.P. Douzery, *Systematic Biology*, 56(5), 293-304, 2007.

La méthode *PhySIC_IST*

PhySIC_IST : *PHYlogenetic Signal with Induction and non-Contradiction Inserting a Subset of Taxa*

Cette méthode

- est basée sur un algorithme construisant un superarbre par insertion d'étiquettes (taxa) dans un squelette d'arbre.
- est une heuristique pour maximiser le *CIC* du superarbre produit
- renvoit un superarbre qui respecte exactement PC et PI. Pour cela,
  - elle propose un (petit) nombre de multifourches
  - elle exclue un petit nombre d'étiquettes



Evaluation pratique : 1 – Simulations

Un protocole de simulations standart dans le domaine [GASTESY ET AL 02, ...] montre que :

- *PhySIC_IST* produit des arbres presque aussi résolus que MRP (erreur de Type II)
- *PhySIC_IST* fait moins d'erreurs de Type I que MRP

Par ailleurs, *PhySIC_IST*

- propose toujours en temps polynomial un superarbre vérifiant les propriétés PI et PC
- étiquette les noeuds du superarbre produit indiquant la raison des irrésolutions (conflit ou manque de chevauchement)
- intègre un curseur statistique permettant d'aller d'une méthode de vote à une méthode de veto
- met en évidence les incongruences de chaque source en regard du reste de la collection

Evaluation pratique : 1 – Simulations

Un protocole de simulations standart dans le domaine [GASTESY ET AL 02, ...] montre que :

- *PhySIC_IST* produit des arbres presque aussi résolus que MRP (erreur de Type II)
- *PhySIC_IST* fait moins d'erreurs de Type I que MRP

Par ailleurs, *PhySIC_IST*

- propose toujours en temps polynomial un superarbre vérifiant les propriétés PI et PC
- étiquette les noeuds du superarbre produit indiquant la raison des irrésolutions (conflit ou manque de chevauchement)
- intègre un curseur statistique permettant d'aller d'une méthode de vote à une méthode de veto
- met en évidence les incongruences de chaque source en regard du reste de la collection

Evaluation pratique : 2 – données réelles

Applications :

- un superarbre couvrant plus de 95% des genres de mammifères
- un superarbre des animaux depuis 94 arbres de gènes couvrant 79 espèces
- Logiciel + interface web opérationnels sur la plateforme montpellieraine ATGC.

PhySIC_IST : cleaning source trees to infer more informative supertrees, C. Scornavacca, V. Berry, E.J.P. Douzery and V. Ranwez, *BMC Bioinformatics*, 2008.

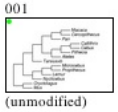
Corrections to the source trees suggested by PhySIC_IST

Display tree in:

- (•) Top view
- ( ) Bottom view

Click on a tree:

001

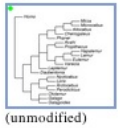


(unmodified)

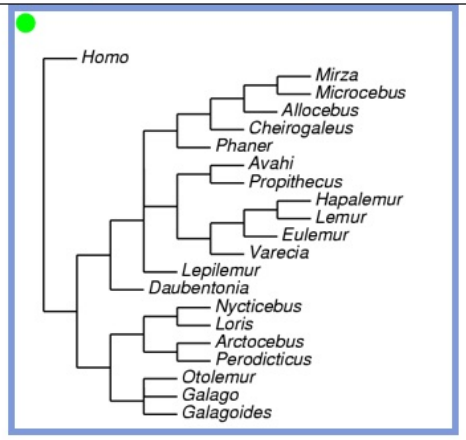
002

(unmodified)

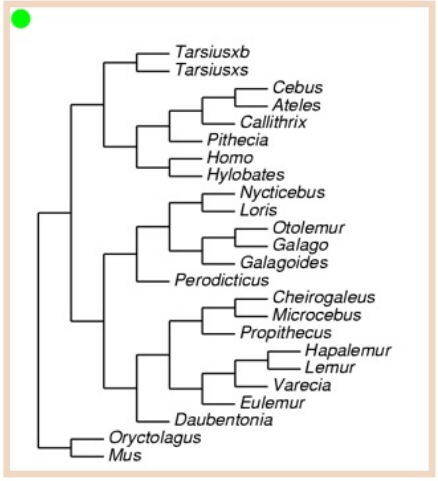
003



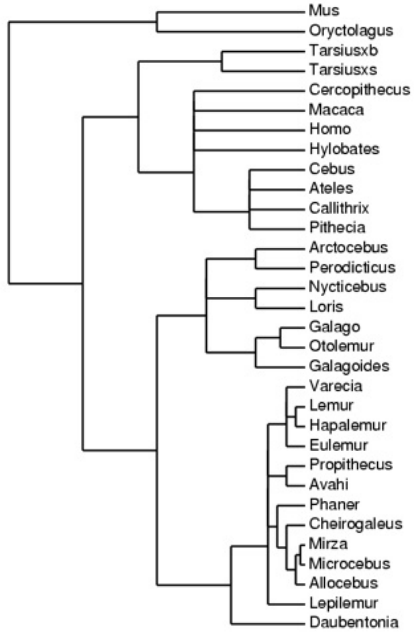
(unmodified)



Zoom on source tree #003



Zoom on source tree #002



Zoom on supertree

Correction threshold: 0.9

This page uses the **scriptree visualization tool.** Use **TreeDyn** for a better tree visualization.

Webmaster: **Vincent Lefort**

Plan

Projets de recherche

Collaborations en cours :

1. Méthodes de construction de réseaux phylogénétiques (DH,PG,CP,RR)
2. Méthode d'inférence de superarbre multiniveaux (ChS,OBE)
3. Méthode de consensus non-biaisée prenant en compte les dates de divergences (TG,VR,AJM)
4. PhyloExplorer - une interface web pour la gestion de collection d'arbres (VR,NC,SP)
5. Assemblage de superarbre depuis la base données TreeBASE (OBE)
6. Méthode de rééchantillonnage pour mesurer la congruence d'arbres (GC)
7. Variante de MRP prenant en compte les dates de divergences aux noeuds internes pour augmenter le chevauchement entre arbres sources (SG,LB)
8. Visualisation d'arbres (FC,RC) (VL,VR,FC,...)
9. ...

Projets à moyen terme

Projet ANR PhylAriane financé sur 3 ans :

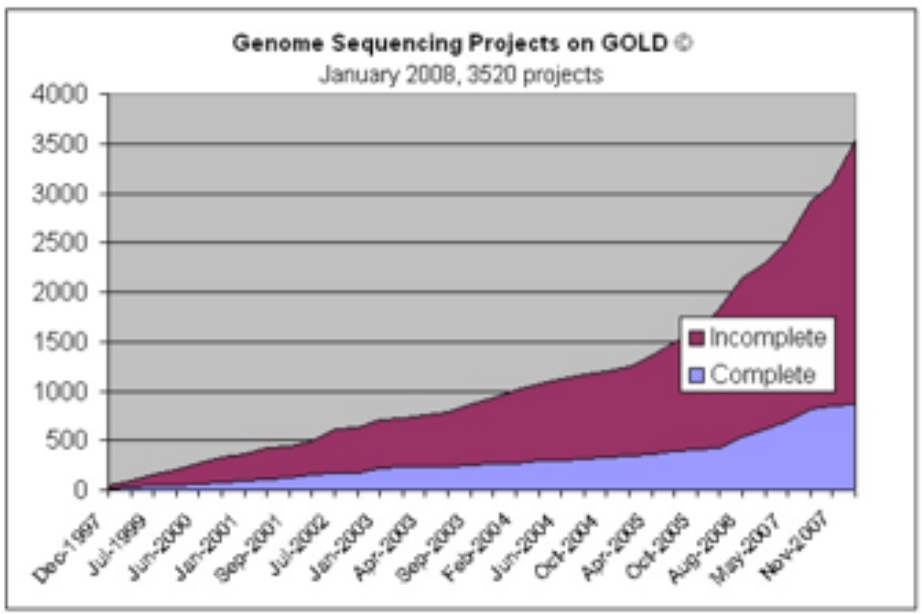
Phylogénomique : algorithmes et représentations intégrés pour l'analyse de l'évolution du vivant

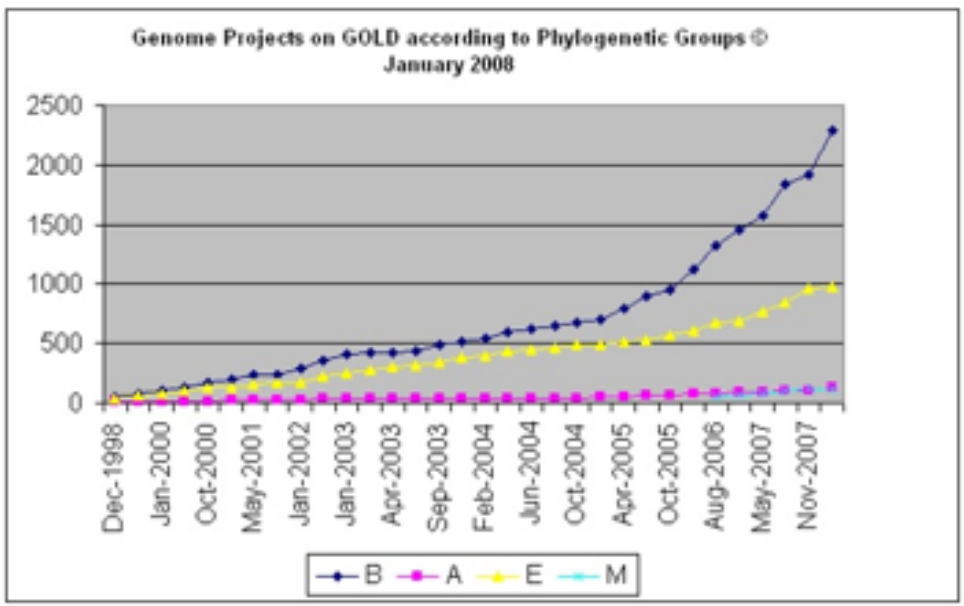
- Vincent Berry et al (LIRMM)
- Vincent Ranwez et al (ISEM)
- Vincent Daubin et al (LBBE)

Gain d'un ordre de grandeur des données

Genome OnLine Database (GOLD) :

- 881 génomes complets disponibles
- plus de 3000 en cours de séquençage dont 2000 génomes bactériens et 1000 eucaryotes





Contexte et enjeux

- 64% des noeuds de l'arbre du NCBI (600 000 espèces) sont non-résolus.
- 1,8 million d'espèces inventoriées, +0,01 million / an
- Pas de progrès significatif depuis les 70s sur les relations entre grands groupes de bactéries.
- Changements majeurs récents sur l'arrangement supposé des 8 grands groupes eucaryotes : suggère des imperfections dans ce que nous pensons être des eucaryotes [Baldauf 03].
- La connaissance de génomes *complets* permet de connaître de façon *exhaustive* le contenu en gènes des génomes : atout fondamental pour retracer les grands événements (duplications, pertes et transferts) et arbitrer entre les scénarios possibles.

Contexte et enjeux

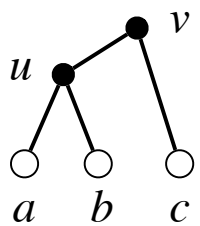
- 64% des noeuds de l'arbre du NCBI (600 000 espèces) sont non-résolus.
- 1,8 million d'espèces inventoriées, +0,01 million / an
- Pas de progrès significatif depuis les 70s sur les relations entre grands groupes de bactéries.
- Changements majeurs récents sur l'arrangement supposé des 8 grands groupes eucaryotes : suggère des imperfections dans ce que nous pensons être des eucaryotes [Baldauf 03].
- La connaissance de génomes *complets* permet de connaître de façon *exhaustive* le contenu en gènes des génomes : atout fondamental pour retracer les grands événements (duplications, pertes et transferts) et arbitrer entre les scénarios possibles.

Disparités entre arbres de gènes et des espèces

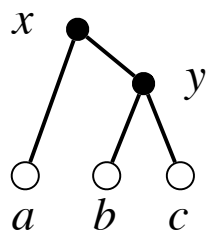
Constat

Les arbres de gènes ne traduisent pas fidèlement l'arbre des espèces : les phénomènes macro-moléculaires perturbent le signal d'héritage des gènes au cours du temps.

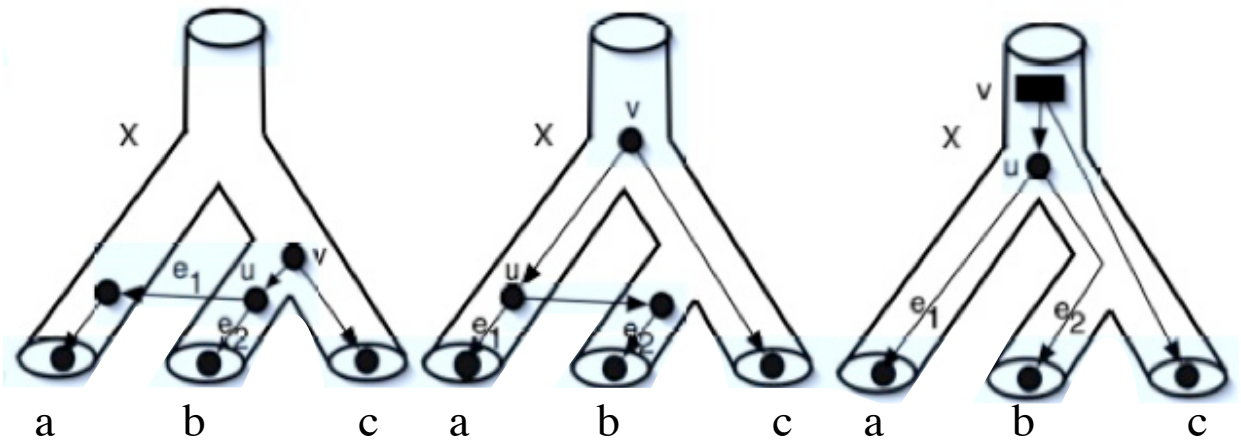
un arbre de gène



arbre des espèces supposé



Trois explications différentes à base de macro-événements

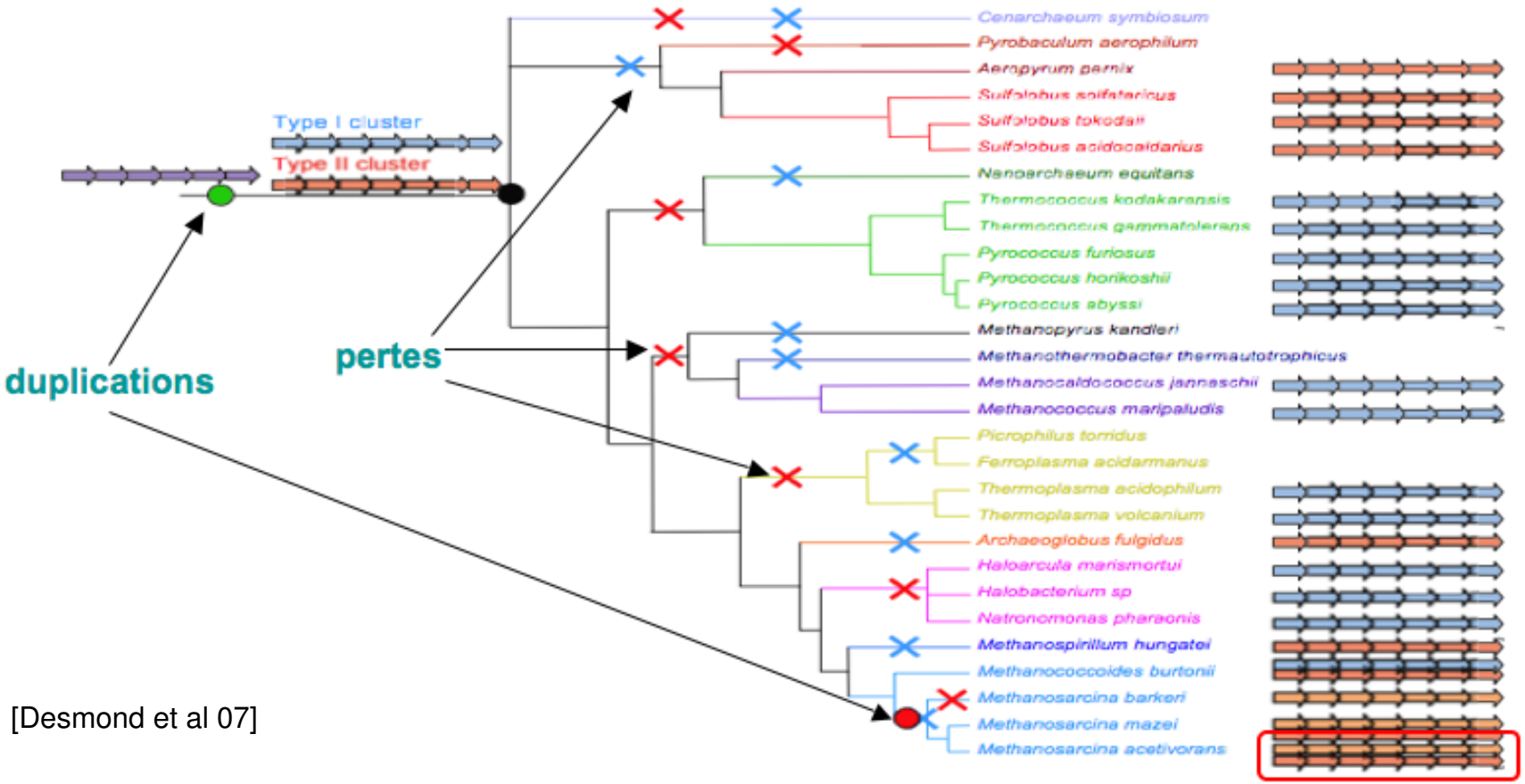


génération et transfert | transfert et perte | duplication et pertes

Un objectif pratique du projet

Familles multigéniques

Les phénomènes macro-moléculaires (duplications, pertes, transferts de gènes) peuvent amener un même gène à être présent en plusieurs copies chez certaines espèces.

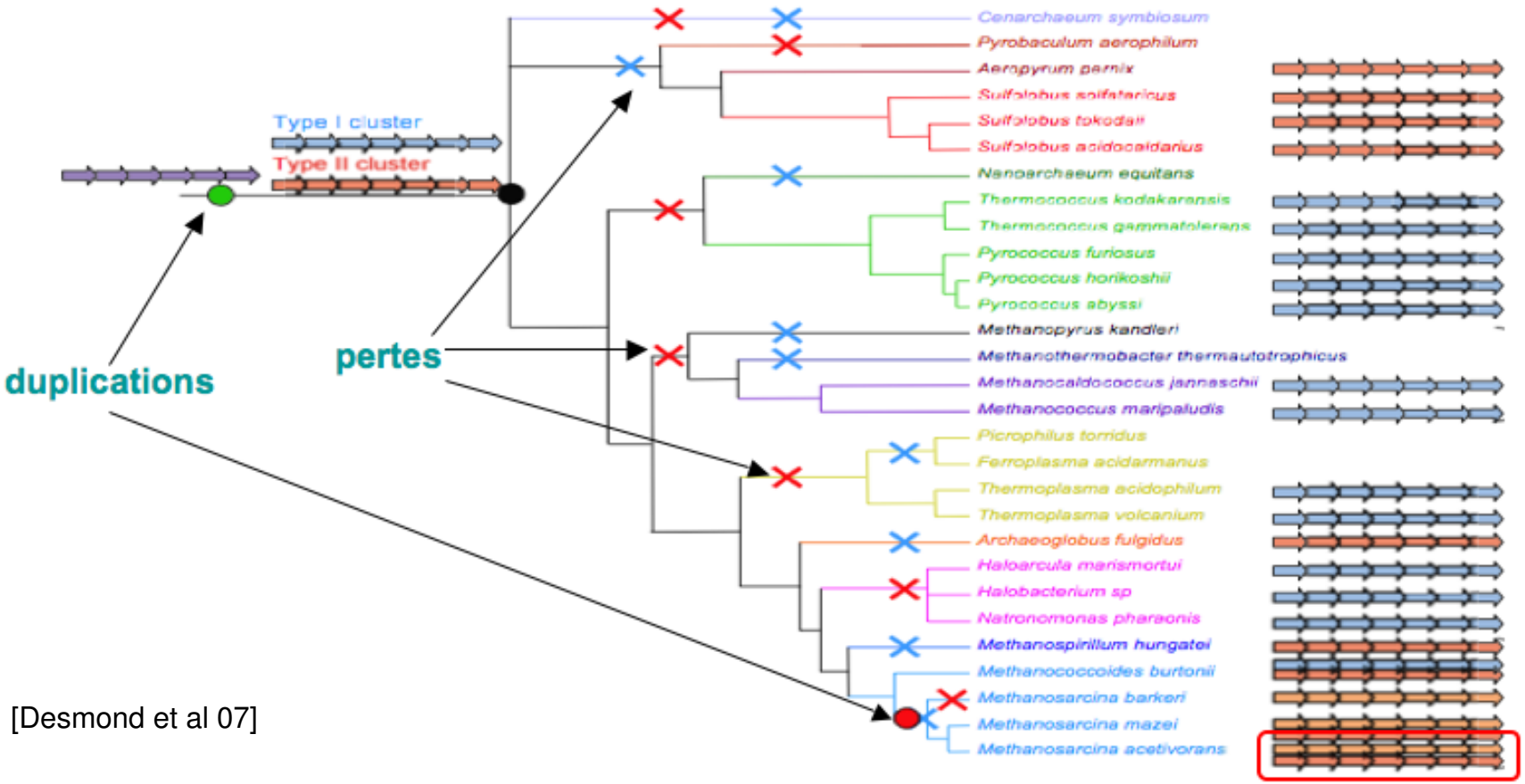


[Desmond et al 07]

Un objectif pratique du projet

Objectif

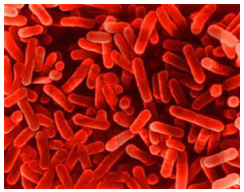
Par la comparaison des milliers d'arbres de gènes, dont ceux des familles multigéniques, construire l'arbre des espèces et inférer les macro-événements ayant affecté ces gènes.



[Desmond et al 07]

Objectif généraux du projet

Partir des séquences et des arbres de gènes pour retracer l'évolution des organismes anciens :



- Elucider les relations entre grands groupes de bactéries (transferts et duplications).

- Explorer l'histoire des duplications entre grands groupes de mammifères.

l'évolution d'une carrière

Merci à mes coauteurs :

.... et à vous pour votre votre attention ;-)