Histoires évolutives et autres comptes. Algorithmes et graphes pour la bioinformatique

Sèverine Bérard

Université de Montpellier

Habilitation à diriger les recherches

École doctorale n° 166 Information Structure Système

Mention informatique

Histoires évolutives et autres comptes

Algorithmes et graphes pour la bioinformatique

---

présentée par

Sèverine Bérard

H.D.R. soutenue à Montpellier le 4 novembre 2016 devant le jury composé de :

Mme Nadia El Mabrouk	Professeure, Université de Montréal, Canada	Rapportrice
Mme Hélène Touzet	DR CNRS, Université de Lille 1	Rapportrice
Mr Stéphane Vialette	DR CNRS, U. Paris-Est Marne-la-Vallée	Rapporteur
Mr Alain Guénoche	DR CNRS, Université de Marseille	Examinateur
Mr Vincent Ranwez	Professeur, Montpellier SupAgro	Examinateur
Mme Mireille Régnier	DR INRIA, École polytechnique	Présidente

Allez les filles !
Go girls !
Vamos chicas !
Gehen Mädchen !
. . .

Remerciements

Je tiens tout d'abord à remercier Nadia El Mabrouk, Hélène Touzet et Stéphane Vialette qui ont accepté de relire ce manuscrit. Merci pour ce travail et pour leur expertise. Je remercie tout aussi chaleureusement Alain Guénoche, Vincent Ranwez et Mireille Régnier d'avoir bien voulu participer à ce jury.

Le travail présenté dans ce manuscrit a été réalisé au travers de nombreuses collaborations et dans un environnement scientifique et humain de grande qualité. Ne pouvant citer et remercier toutes les personnes qui y ont contribué, de très près, ou de plus loin, je reproduis ici la dernière diapositive de ma présentation orale, vous montrant le portrait de certaines d'entre elles :

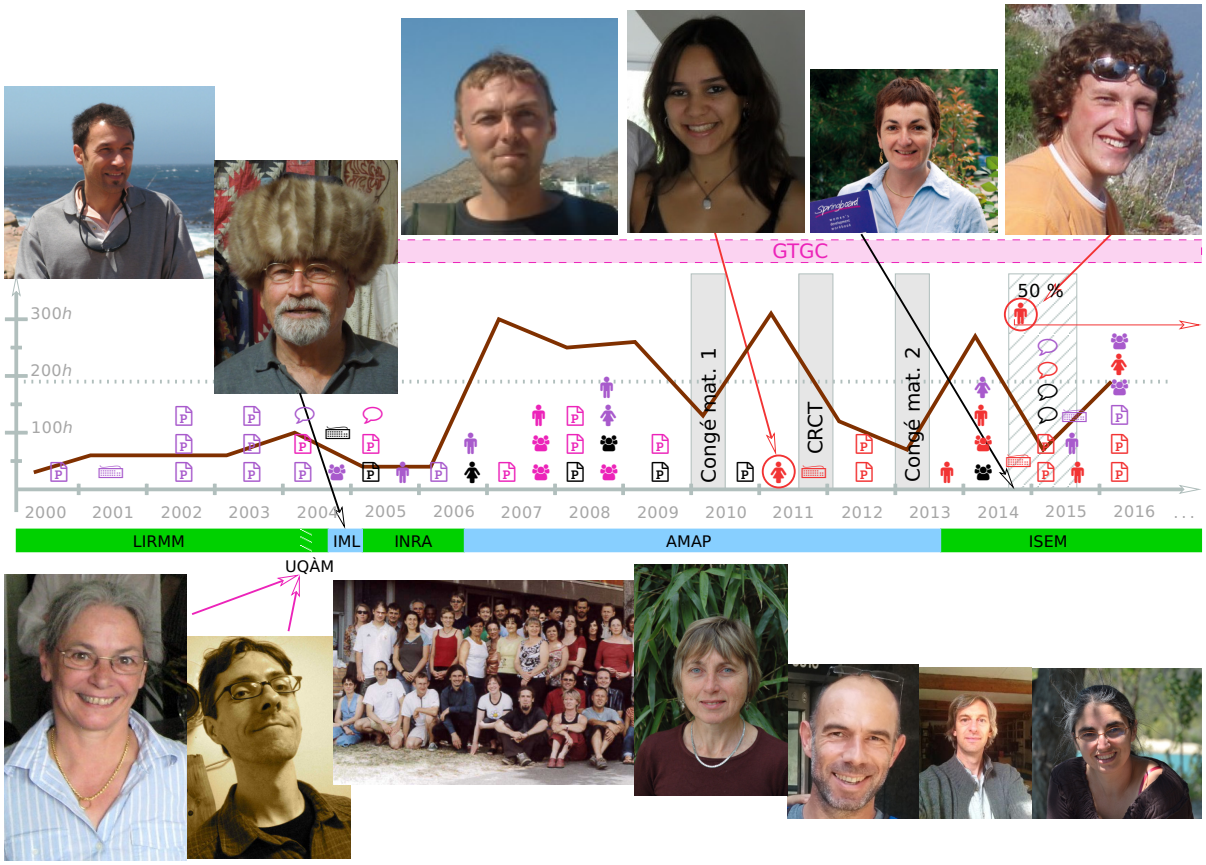

Table des matières

Table des figures

Avant propos

Alors le fil vert sur le bouton vert, le fil rouge sur le bouton rouge.

Colonel Blanchet (Robert Lamoureux)

« On a retrouvé la septième compagnie » 1975.

Ce mémoire d'habilitation à diriger les recherches (HDR) a été constitué en deux temps. Le premier, à l'automne 2015, durant lequel je me suis attachée à produire un CV détaillé (annexe A, page 41) regroupant tous les éléments factuels de ma carrière scientifique depuis la date de son démarrage que j'ai estimée au début des années 2000, lors de mon stage de recherche de DEA [1]. À la fin de ce CV, j'ai essayé de mettre en perspective ces éléments factuels de manière temporelle, géographique, thématique, humaine (au travers de mes collaborations), sans oublier le volet enseignement qui fait partie intégrante de mon métier d'enseignante-chercheuse. Ce CV peut constituer un bon point d'entrée pour aborder ce manuscrit.

Dans un second temps, février-juin 2016, j'ai rédigé la partie principale de ce mémoire (pages 3-40) pour laquelle j'ai opté pour la formule "HDR sur articles". C'est-à-dire que j'ai choisi de présenter de manière synthétique mes contributions principales et d'annexer à la fin de l'ouvrage les articles scientifiques correspondant à mes travaux les plus récents.

Le premier temps a été très serein et très agréable, me permettant de faire un bilan sur ma carrière et de réfléchir aux points communs de mes travaux antérieurs. Alors que le deuxième temps s'est fait au pas de course, se télescopant avec une période d'enseignement chargée, de nombreux stages d'étudiants à encadrer, des échéances pour des publications et la visite de Anne Bergeron pendant deux mois (bien que ravie de pouvoir travailler à nouveau avec elle). La rédaction de ce mémoire a toutefois été une expérience très enrichissante, que nous avons partagée avec Annie Chateau en nous "*coachant*" mutuellement, nous obligeant ainsi à respecter les délais. Si tout se passe bien, nous soutiendrons le même jour, objectif que l'on s'était fixé au départ.

Ayant choisi une version compacte de présentation de mes travaux, je ne reviens pas aux définitions fondamentales des deux disciplines dans lesquelles s'inscrivent mes recherches, à savoir l'informatique et la biologie, mais je donnerai les définitions des concepts peu communs abordés

1. équivalent du master 2 à l'heure actuelle.

dans les travaux présentés. Je ne détaille pas non plus tous les aspects des différents champs thématiques que j'aborde. Aussi, un lecteur ou une lectrice qui voudrait plus de détails sur les thèmes abordés pourra consulter les ouvrages de référence suivants :

- pour un(e) informaticien(ne) qui voudrait se plonger dans le monde de la biologie évolutive, je recommande le « *Fundamentals of molecular evolution* » de Graur et Li [49], dont je m'étais servi pour ma thèse ou, en français, le très complet « Biologie évolutive » [95] co-écrit par de nombreuses personnalités de l'ISE-M ;
- pour l'algorithmique de base, les graphes et la théorie de la complexité, le fameux « Introduction à l'algorithmique » de Cormen, Leiserson et Rivest [28] ;
- « *Algorithms on Strings, Trees, and Sequences* » de Dan Gusfield [50] pour tout ce qui concerne plus spécifiquement l'algorithmique des séquences et du texte ;
- ou aussi « *Algorithms on Strings* » de nos collègues français Maxime Crochemore, Christophe Hancart et Thierry Lecroq [30], dont une version française est accessible en ligne http://www-igm.univ-mlv.fr/~mac/CHL/CHL-2011.pdf ;
- pour tout ce qui concerne les problèmes combinatoires des réarrangements génomiques, il y a l'exhaustif « *Combinatorics of Genome Rearrangements* » [38] qui classe, à la manière d'un catalogue, les problèmes (formalisés en fonction des événements évolutifs considérés, de la prise en compte de l'orientation, etc.) et répertorie tous les résultats en terme de complexité et d'algorithmes connus à la date de parution.

Dans ce manuscrit, les références concernant des publications personnelles sont citées comme celle-ci : [BCP+16] et listées page 93. Les autres références, par exemple [100], sont listées page 95.

Une version électronique de ce mémoire est disponible à l'adresse suivante :
http://www.pages-perso-severine-berard.univ-montp2.fr/HDR/HDR_SBérard2016.pdf ;
accompagné du CV détaillé dans un fichier à part, permettant l'activation des liens internes :
http://www.pages-perso-severine-berard.univ-montp2.fr/HDR/CV_SBérard2016.pdf.

Enfin, c'est une évidence mais je tenais à le rappeler ici, tous les travaux présentés dans ce mémoire sont issus de collaborations. Collaborations avec mes encadrants lorsque j'étais en thèse ; avec mes collègues : doctorants, post-doctorants, enseignants-chercheurs et chercheurs ; ou avec mes étudiants. La recherche est comme beaucoup d'autres choses le fruit d'un travail collectif.

Il ne me reste plus qu'à vous souhaiter une agréable lecture.

Introduction

— *Des histoires, des histoires, des histoires, est-ce que je peux avoir une histoire s'il te plaît, tu peux me raconter une histoire ?*

— *Encore une histoire, vous avez été sages, vous êtes sûrs ? Bon d'accord, je vais vous raconter l'histoire du lac des cygnes sur une musique de Piotr Tchaïkovski. Vous êtes prêts ? Vous êtes bien installés ? Alors chut, plus de bruit, parce que l'histoire va commencer.*

« Il était une fois . . . Des histoires en musique », d'Élodie Fondacci

Émission radiophonique de Radio Classique.

J'ai toujours aimé raconter des histoires, en inventer même. Les histoires dont je parle dans ce mémoire sont d'un type particulier, ce sont des *histoires évolutives*. Ces histoires là essayent de retracer l'évolution biologique qui a mené à l'actuelle diversité des êtres vivants. On parle de *reconstruction d'histoire évolutive* et parfois de *reconstruction de génomes ancestraux*. Cela ressemble à un puzzle ou à une enquête. L'évolution a laissé des traces dans ce que l'on peut observer aujourd'hui, et elles constituent autant d'indices qui vont nous être utiles pour déchiffrer ces histoires évolutives et remonter dans le temps. Ensuite il faut aussi deviner ou imaginer le modèle évolutif qui a permis aux êtres vivants d'évoluer. Il est difficile d'appréhender toutes les traces laissées par l'évolution dans un même modèle. Je m'intéresse plus particulièrement aux traces laissées dans les *génomes* et leurs *séquences d'ADN*. Les récents progrès des techniques de séquençage offrent un accès quasi illimité à ces données. Reconstruire des histoires évolutives c'est aussi faire des hypothèses. Quels événements évolutifs ont eu lieu ? À quelle fréquence ? De manière uniforme dans le temps et l'espace ? On ne peut pas répondre directement à ces questions. On peut cependant développer des modèles et évaluer s'ils sont cohérents avec les données observées. Le modèle nous donne un cadre, sûrement en partie inexact, dans lequel chercher l'histoire évolutive. En fait, le modèle définit un ensemble, souvent très grand, d'histoires évolutives possibles. À nous de chercher laquelle est la vraie, ou du moins la plus réaliste de cet ensemble. Pour cela, on donne généralement des scores ou des coûts aux différentes possibilités, et il s'agit alors de déterminer la ou les histoires de meilleur score (ou de moindre coût). Vu la taille des données, ces phases d'exploration et de calcul nécessitent d'utiliser la puissance des ordinateurs. L'interaction avec l'ordinateur se fait aux travers d'*algorithmes* qui seront traduits

en programmes. C'est à l'aide de ces algorithmes que l'on encode les données du problème, et que l'on explore, de la manière la plus habile possible, l'espace des solutions. Les *graphes* sont une structure de données fréquemment utilisée en informatique et qui se prête bien à être le support de modélisations. Un graphe est une simple interconnexion de nœuds, représentés par des points, et d'arêtes, représentées par des traits entre ces points. Les graphes ont un pouvoir d'expression extraordinaire. Ils ont également la bonne propriété d'être des objets très bien étudiés en informatique et de disposer d'une panoplie de méthodes efficaces pour leur traitement. C'est dans ce contexte et avec ces boîtes à outils que j'ai cherché à déchiffrer des histoires évolutives.

Je vais vous raconter quatre *histoires* qui correspondent à mes principaux travaux de recherche. L'ordre de présentation s'est imposé (presque) naturellement, suivant à la fois l'ordre chronologique de ces travaux et la taille croissante des objets biologiques auxquels je me suis intéressée :

— *marqueurs génomiques*[2], sous forme de séquences – répétées en tandem (section 1.1) ;

— *ensembles de marqueurs génomiques*, sous forme de permutations (section 1.2) ;

— *ensembles de génomes entiers*, sous forme d'ensembles d'adjacences de gènes accompagnés de leurs arbres phylogénétiques (section 1.3), ou à nouveau sous forme de séquences pour nos derniers travaux sur les bactériophages (section 1.4).

Dans la première histoire sur les séquences répétées en tandem, on verra comment retrouver des suites d'amplifications, événements évolutifs les plus fréquents sur ces séquences, pour les aligner entre elles. L'alignement produit n'étant que le début d'une autre histoire que l'on ne contera pas ici. La deuxième histoire se situe au début du siècle, où l'on a vu arriver en masse les données de génomes complets permettant ainsi les premières études génomiques de grande ampleur. Sous deux modèles évolutifs différents, on racontera comment transformer un génome en un autre. La troisième histoire remonte dans le temps en s'intéressant aux ancêtres de ces génomes, il s'agira alors de reconstruire des histoires évolutives de tout un ensemble de génomes actuels et de trouver leurs ancêtres. Dans cette histoire, un rebondissement nous amènera également à utiliser nos résultats pour améliorer les assemblages de génomes actuels. Enfin, la quatrième histoire a pour sujet des êtres étranges : les bactériophages. Leur mode d'évolution particulier brouille les pistes et les outils d'alignement classiques n'arrivent pas à les aligner dans leur totalité malgré leur petite taille. En revenant à des alignements deux à deux exacts, nous expliquerons comment la méthode que nous avons mise au point permet de déchiffrer une partie de l'histoire évolutive d'un ensemble de bactériophages et par la même de les aligner.

Tous ces travaux font appel à de l'algorithmique de graphe et de texte. Nous avons créé ou utilisé des modèles évolutifs dans le cadre du *principe de parcimonie*, c'est-à-dire que l'on suppose que l'explication la plus économe en événements évolutifs est celle qui doit être retenue. Ce cadre de raisonnement nous amène donc à *compter* les événements ou à calculer la somme de leurs coûts. Nous avons appliqué nos solutions à des données biologiques conduisant à diverses applications. Inférer des histoires évolutives nous a permis d'aligner des séquences, de donner

2. Un marqueur génomique peut être un gène, un groupe de gènes ou tout autre zone d'intérêt localisée sur la séquence d'ADN.

accès à la structure de génomes ancestraux ou d'améliorer les assemblages de génomes dans les bases de données.

Une deuxième partie de ce mémoire est consacrée à l'exposé de mes projets de recherche actuels (juin 2016). Ces projets s'appuient sur les trois dernières publications données en annexes et correspondant aux deux dernières histoires racontées. Ils s'inscrivent dans les thématiques de la *coévolution* et de l'*alignement de séquences* avec des applications à l'assemblage de génomes pour le premier (section 2.1.1) et à l'aide à l'annotation pour le second (section 2.1.2). Je donne ensuite les liens entre ces deux thématiques ainsi que leurs convergences possibles (section 2.1.3). Puis je termine cette partie en donnant des perspectives plus générales (section 2.2).

Et comme toute histoire a une fin (en général) une partie conclusion viendra clôturer ce mémoire.

Contributions

Not everything that can be counted counts.
Not everything that counts can be counted.

William Bruce Cameron

« Informal Sociology, [...] » 1963.

Cette partie présente de manière synthétique une grande partie de mes travaux de recherche. Ces travaux s'organisent autour de 3 thèmes de recherche : *alignement*, *génomique comparative* et *coévolution*. Dans chacun de ces thèmes, pour déchiffrer des relations entre séquences ou entre espèces, nous sommes amenés à reconstruire des histoires évolutives. C'est sous cet angle que je présente dans ce qui suit les quatre problèmes principaux que j'ai abordés dans mes recherches. Comme mentionné dans l'introduction, l'ordre de présentation suit à la fois la chronologie des travaux mais aussi la taille croissante des objets biologiques auxquels je me suis intéressée. Le thème de l'alignement se retrouve ainsi en début et en fin de présentation. Les deux premières histoires (sections 1.1 et 1.2) sont plus détaillées que les deux suivantes (sections 1.3 et 1.4) qui introduisent les articles donnés en annexes du mémoire (annexes B, C et D à partir de la page 57). À la fin de chaque présentation, un encadré présente de manière sommaire les algorithmes et les graphes impliqués, car ils sont les outils majeurs que je manipule, ainsi que des mots-clés ; j'y évoque également les applications, les logiciels et les impacts de ces travaux.

Histoires de séquences : alignements

L'alignement de séquence

Côté informatique, nous nous plaçons ici clairement dans l'algorithmique du texte, où l'alignement de séquences est un problème très classique. Aligner deux séquences revient à minimiser le nombre (ou la somme des coûts) de trois opérations (ou événements évolutifs) affectant leurs caractères et permettant de passer d'une séquence à l'autre : substitution, insertion et délétion. Un exemple d'alignement est donné à la figure 1.1. Ce problème se généralise aisément à plusieurs séquences mais il existe alors plusieurs manières de calculer le coût de l'alignement, le score SP (*Sum of Pair*) étant le plus connu [69, 22]. Le problème d'aligner deux séquences est de

$$\begin{pmatrix} O & C & T & O & B & R & - & E \\ O & - & T & A & - & R & I & E \end{pmatrix}$$

FIGURE 1.1 – Exemple d'alignement des séquences $OCTOBRE$ et $OTARIE$ mettant en jeu quatre identités (O, T, R et E), une substitution (d'un O par A), deux délétions (C et B) et une insertion (I). Si l'on donne les coûts 1 aux insertions-délétions, 3 aux substitutions et 0 aux identités, le coût de cet alignement est de 6 ($= 0 + 1 + 0 + 3 + 1 + 0 + 1 + 0$).

complexité polynomiale (quadratique) et a été résolu dans les années 70 [71, 89]. Le problème de l'alignement multiple, c'est-à-dire aligner plusieurs séquences en même temps, est quant à lui NP-difficile [99].

Côté biologie, c'est un problème central depuis l'apparition des premières séquences génomiques (ADN, protéines) dans les années 60-70. Son but est de détecter les similarités entre séquences car on suppose que les séquences similaires ont une fonction semblable. Ces similarités peuvent également indiquer une origine évolutive commune, on parlera alors de séquences homologues. L'alignement de deux séquences est très utilisé pour la recherche de séquences proches dans les banques de données. L'alignement multiple est quant à lui, le point de départ de nombreuses analyses bioinformatiques, comme la construction d'arbres phylogénétiques ou la détection de domaines protéiques.

J'ai abordé le problème de l'alignement de séquences par deux fois (au moins) : lors de ma thèse, où il s'agissait d'aligner des séquences répétées en tandem, et plus récemment, pour aligner des génomes complets de bactériophages (*cf.* section 1.4). Ces deux problèmes ont en commun que le modèle évolutif utilisé (ou ensemble d'opérations élémentaires) différait du modèle classique (substitution, insertion et délétion).

Séquences répétées en tandem

Les séquences répétées en tandem constituent une classe de séquences biologiques dans laquelle on trouve les microsatellites et les minisatellites. Il s'agit de séquences constituées de copies d'un motif dont les occurrences, parfois très différentes, se situent côte à côte (*cf.* figure 1.2). Ces séquences sont considérées comme des marqueurs d'évolution récente, leur nombre de copies pouvant différer d'un humain à l'autre par exemple. Elles sont fréquemment utilisées en génétique des populations [3, 90] et ont même servi à soutenir l'hypothèse *Out-of-Africa* sur les origines de l'être humain [5]. Les séquences répétées en tandem sont également présentes dans un certain nombre de maladies, comme l'épilepsie ou la maladie de Creutzfeldt-Jakob [18].

L'évolution de ces séquences se fait au travers des 3 événements évolutifs élémentaires (substitution, insertion et délétion de nucléotides) mais aussi au travers de mécanismes qui créent, ou suppriment, une ou plusieurs copies du motif répété. Ce qui implique que deux séquences proches en termes d'événements évolutifs peuvent être différentes en termes de contenu de séquence et rend ainsi l'alignement classique peu enclin à identifier les séquences semblables.

Nous encodons les séquences répétées en tandem par la suite de leurs motifs, où chaque motif différent, *un variant*, est représenté par une lettre. Nous obtenons de nouvelles séquences

ACGT	*ACGT*	*ACGT*	*AGGT*	*AGGT*	*AGT*	*AGGT*	*ACGA*	*ACGA*	*AGGT*	*ACGT*
a	*a*	*a*	*b*	*b*	*c*	*b*	*d*	*d*	*b*	*a*

FIGURE 1.2 – Exemple de séquence répétée en tandem sur la première ligne constituée de 11 répétitions d'un motif variant autour de *ACGT*. Sur la deuxième ligne sa traduction en carte où chaque caractère représente un variant du motif répété.

de caractères et chaque opération de notre modèle évolutif agit exactement sur un caractère. Seul le sens sémantique des substitutions, insertions et délétions diffère puisqu'elles modifient maintenant un motif entier. Une séquence ré-encodée comme la suite de ses variants est appelée *une carte*. Un exemple de ré-encodage est donnée à la figure 1.2. La succession des variants dans la carte peut être directement obtenue par une technique spécifique appelée *Minisatellite Variant Repeat PCR (MVR-PCR)* [58]. On dispose donc de jeux de données directement codés sous ce format.

Nous avons abordé le problème de l'alignement de ce type de séquences en définissant un modèle évolutif particulier comprenant les trois opérations classiques ainsi que les opérations d'*amplification* et de *contraction*. L'amplification copie un facteur [1] de la séquence, c'est-à-dire un ou plusieurs caractère(s), et met le ou les exemplaire(s) du facteur copié à côté du facteur original. La contraction est l'événement inverse. Sous ce modèle, les opérations ne sont plus commutatives et l'ordre dans lequel on les applique est donc important. Nous avons résolu le problème de l'alignement de 2 séquences sous ce modèle pour les cas où les amplifications (resp. contractions) copient (resp. retirent) un motif à la fois [BER03]. On améliore légèrement la complexité et le modèle (coûts des événements variables) dans [BNB+06]. Un chapitre de ma thèse [Bé03] est consacrée au problème général où nous relaxons les contraintes sur les amplifications et contractions. Un exemple d'évolution de deux séquences répétées représentées sous forme de cartes et leur alignement est présenté à la figure 1.3.

Une partie très intéressante du problème de l'alignement de deux cartes de séquences répétées en tandem est aussi la plus difficile : il s'agit de reconstruire l'histoire évolutive d'un variant d'une carte qui s'aligne avec plusieurs variants dans l'autre carte, ce morceau de séquence étant appelé *arche*, en rouge dans la figure 1.3. Ce problème particulier avait d'ailleurs été l'objet de mon stage de DEA [Bé00]. Il s'agit de retrouver une suite d'amplifications (ou de contractions, c'est symétrique) et de substitutions de coût minimum, qui à partir d'un variant génère (ou compresse) l'arche toute entière. Dans le modèle évolutif des séquences répétées en tandem, les amplifications et contractions sont toujours de coût moins élevé que les autres opérations car ce sont les événements les plus fréquents dans cette classe de séquences. Lorsque l'on cherche à retrouver l'histoire évolutive qui a mené à la création d'une arche, on va donc chercher à maximiser le nombre d'amplifications par rapport aux nombres de substitutions. Pour générer l'arche *bbcbddb* de longueur 7 de la figure 1.3 à partir d'un variant *b*, on pourrait effectuer 6 amplifications de *b*, puis muter ces *b* en *c* ou *d* aux positions 3, 5 et 6, ce qui ferait appel à 3

1. Une séquence x est un facteur d'une séquence y s'il existe deux séquences u et v telles que $y = uxv$ [30].

Individu 1 (I_1)

Événement	Séquence				
	a	a	a	a	a
substitution	a	**e**	a	a	a
	1	2	3	4	5

Individu 2 (I_2)

Événement	Séquence										
	a	a	a	a	a						
substitution	a	a	a	**b**	a						
5*amplification	a	a	a	b	**b**	**b**	**b**	**b**	**b**	a	
substitution	a	a	a	b	b	**c**	b	b	b	a	
substitution	a	a	a	b	b	c	b	**d**	b	a	
amplification	a	a	a	b	b	c	b	d	**d**	b	a
	1	2	3	4	5	6	7	8	9	10	11

I_1 :	a	e	a	a	$-$	$-$	$-$	$-$	$-$	$-$	a
	|	[	|	[	\	(	\	(	\	\	|
I_2 :	a	a	a	b	b	c	b	d	d	b	a

FIGURE 1.3 – Exemple d'évolution d'un minisatellite chez 2 individus (en haut). Alignement de ces cartes (en bas). Dans la ligne du milieu, '|', ']', '\', '(' représentent respectivement une identité, une substitution, une amplification et une amplification suivie d'une substitution à la même position. On observe une arche dans la séquence I_2 : *bbcbddb* qui s'aligne avec un seul variant a dans la séquence I_1.

substitutions et donnerait un coût total de $6A + 3S$. Cependant, il existe une histoire moins coûteuse ($6A + \mathbf{2}S$) qui génère le d de la 6^e position de l'arche par une amplification du d précédent. En fait, chaque variant peut être généré à partir de l'amplification d'un variant identique mais ce n'est plus vrai lorsque les variants sont entremêlés les uns dans les autres. Pour une arche *baba*, le 2^e a ne peut pas avoir été généré à partir du premier a si le 2^e b a été généré à partir du premier b, ces deux amplifications sont *incompatibles*. Cela peut très bien se modéliser en terme de *chevauchement d'intervalles* dont les extrémités sont les paires de variants identiques comme on peut le voir à la figure 1.4 à gauche. Deux intervalles se chevauchent si leur intersection est non vide mais qu'aucun n'est inclus dans l'autre. Trouver un plus grand nombre d'amplifications compatibles revient alors à trouver un plus grand nombre d'intervalles non chevauchants, et cela peut se traduire en terme d'algorithmique de graphe par trouver un *stable maximum* dans le *graphe de chevauchements* de ces intervalles. Dans ce graphe de chevauchements, les sommets sont les intervalles et il y a une arête entre deux sommets si et seulement si les intervalles qu'ils représentent se chevauchent [45]. Un stable est un sous-graphe ne comportant aucune arête. Un exemple de graphe de chevauchements d'intervalles représentants des arches est donné à la figure 1.4 à droite.

Notre algorithme d'alignement de séquence répétées en tandem se déroule en deux étapes. La première constitue un pré-traitement des deux séquences, en calculant le coût de toutes les arches potentielles. Dans un deuxième temps, l'algorithme procède de manière classique en utilisant une matrice de programmation dynamique où les arches constituent des dépendances longues distances. D'où une complexité en temps en $O(n^3)$ au lieu de la classique complexité quadratique de l'alignement de séquences.

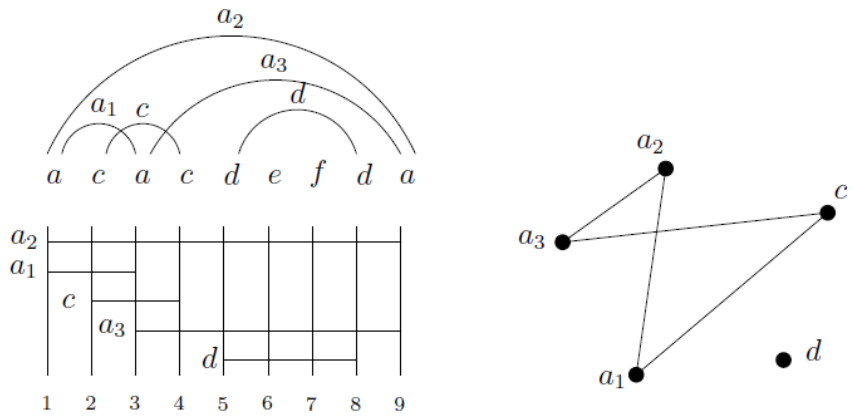

FIGURE 1.4 – Transformation du problème de l'ensemble maximal d'arches compatibles en problème de stable max dans un graphe de chevauchements. Chaque arche correspond à un intervalle, et les arches sont compatibles si les intervalles correspondant ne se chevauchent pas [a]. Le graphe de chevauchements est construit en associant à chaque sommet un intervalle et il y a une arête entre deux sommets ssi les intervalles se chevauchent. Ainsi, trouver un ensemble maximal d'arches compatibles revient à trouver un stable max dans le graphe.

a. En fait on applique un léger décalage aux intervalles se rencontrant à un même point : les intervalles qui ont la même origine ou la même extrémité sont considérés comme chevauchants (comme a_1 et a_2), ceux dont l'extrémité de l'un est l'origine de l'autre sont considérés comme non chevauchants (comme a_1 et a_3).

Nous avons implémenté notre algorithme dans un programme nommé MS_ALIGN. À l'aide de ce programme, nous avons étudié des données biologiques provenant du minisatellite humain MSY1 [59, 17]. Nous avons construit à partir de nos résultats des arbres phylogénétiques semblables à ceux obtenus grâce à d'autres marqueurs du chromosome Y indépendants de MSY1 [94]. Nos arbres offraient une meilleure résolution et permettaient de distinguer des populations à l'intérieur des haplogroupes [2] formés par les marqueurs du chromosome Y.

Alignement de séquences répétées en tandem

Dans ce problème, il faut déchiffrer les suites d'amplifications de motifs répétés dans chacune des deux séquences pour les aligner correctement. Il s'agit donc d'inférer les histoires évolutives de ces sous-séquences (arches) pour les inclure dans un processus plus classique d'alignement par programmation dynamique.

Algorithmique & Graphes Principalement des graphes d'intervalles et de chevauchements (= graphe de cordes). Les algorithmes produits font appel à de la programmation dynamique. La phase de pré-traitement pour retrouver l'histoire évolutive des arches est apparentée à la recherche d'un stable maximum dans un graphe de chevauchement, problème polynomial dans ce cas.

2. Un haplogroupe désigne une combinaison de marqueurs génétiques utilisé en biologie évolutive humaine.

Alignement de séquences répétées en tandem

Applications/Logiciel Nous avons réalisé un programme nommé MS_Align [a] qui implémente notre méthode d'alignement, de complexité polynomiale. Il s'agit du premier programme capable d'aligner des cartes de séquences répétées en tandem. Des études ultérieures ont utilisé MS_Align pour décrire, entre autres, l'histoire de la colonisation des souris [15] ou l'évolution d'une famille de protéines chez les plantes [81].

Publications/Impact Ces travaux ont donné lieu à plusieurs publications en plus de ma thèse, dont principalement [BER03] et [BNB+06], citées respectivement 49 fois et 17 fois [b]. Ils ont ouvert la voie à de nombreuses autres recherches, tant sur le plan algorithmique [84, 1, 76], que sur les applications [78, 72].

a. http://www.atgc-montpellier.fr/ms_align/
b. d'après http://scholar.google.fr/ en juin 2016.

Histoires de génomes : génomique comparative

La génomique comparative est un champ thématique extrêmement large. Beaucoup de sujets de recherche peuvent se revendiquer de la génomique comparative du moment qu'ils mettent en jeu des comparaisons. Il est généralement admis que l'on se place ici au niveau de génomes complets (ou autant qu'ils peuvent l'être) et que l'on cherche l'histoire évolutive qui les relient. Cette histoire, qui explique les relations entre génomes actuels, est donnée en termes de *réarrangements génomiques*, c'est-à-dire d'événements évolutifs de grande ampleur qui ont bouleversé l'organisation des génomes. Ces réarrangements sont définis par le modèle évolutif sous lequel on regarde le problème.

Les deux problèmes que j'ai abordés, décrits dans les sous-sections suivantes, et que je classe dans le domaine de la génomique comparative, ont pour but de trouver une histoire évolutive entre deux génomes. Dans ce contexte, on appelle cette histoire un *scénario évolutif*. Dans les deux problèmes, on se place sous un modèle évolutif assez simple, ne contenant qu'un seul type de réarrangement génomique, c'est-à-dire une seule opération : l'*inversion* dans un cas, le *DCJ* (*Double Cut and Join*) dans l'autre. Nous avons choisi l'inversion car c'est de loin le réarrangement génomique le plus fréquent dans les génomes [75, 43]. Quant au DCJ, comme on le verra dans la suite, il permet de modéliser plusieurs réarrangements différents. Une idée commune à ces travaux vient du fait que pour trouver un scénario entre deux génomes, on minimise un critère de coût. Cependant, pour un coût optimal, il y a très souvent plusieurs scénarios possibles. Pour pouvoir choisir parmi les scénarios optimaux un scénario pertinent du point de vue biologique, il faut ajouter un critère. L'idée est de privilégier les scénarios dans lesquels des éléments côte à côte à la fois dans le génome de départ et dans celui d'arrivée, restent côte à côte tout au long du scénario. Le critère que nous avons choisi est donc celui de la *conservation de structures combinatoires* dans les scénarios. Ces structures peuvent être

les adjacences ou les intervalles communs que l'on définit dans la suite. Dans les deux cas, les génomes sont représentés par des permutations. Cela veut dire que l'on définit des marqueurs génomiques se correspondant de manière unique dans chaque génome et qu'on numérote ces marqueurs ; dans cette configuration les marqueurs sont dits *uniques* et *universels*. Ainsi, si l'on considère n marqueurs, chaque génome peut-être représenté comme une permutation d'entiers de l'intervalle $[1..n]$. Si les marqueurs n'étaient pas uniques, un même entier apparaîtrait plusieurs fois ; s'ils n'étaient pas universels, des entiers apparaissant dans un génome ne se retrouveraient pas dans l'autre. Et pour encore simplifier la formalisation du problème, l'un des deux génomes est numéroté de manière à être la permutation identité $(1\,2\,3\,\ldots\,n)$ sans perte de généralité. Pour finir la formalisation, il ne reste plus qu'à définir en termes de permutations, l'effet d'un réarrangement génomique sur un génome.

Perfect Sorting By Reversals

Lorsque le modèle évolutif ne comprend pour seul réarrangement que l'inversion, *reversal* en anglais, le problème de trouver un scénario optimal entre deux génomes est connu sous le nom de "*Sorting By Reversals*". Ce problème est apparu pour la première fois dans un article de Sankoff [86]. Il s'agit de minimiser le nombre d'inversions permettant de transformer un génome en l'autre. On considère le génome de départ comme une permutation et le génome d'arrivée comme l'identité, c'est pourquoi on parle de *tri* par inversion. Une inversion agit sur un intervalle de la permutation en y "renversant" l'ordre : par exemple l'inversion de l'intervalle $(3\,4)$ dans la permutation $(1\,2\,3\,4\,5)$ donne $(1\,2\,4\,3\,5)$. Une inversion est souvent notée comme l'intervalle qu'elle retourne.

Il faut se rappeler que les éléments des permutations représentent des marqueurs génomiques. Ceux-ci sont codés dans les génomes sur la molécule d'ADN qui est double-brin, et sont lus dans un sens ou dans l'autre, selon le brin sur lequel ils se trouvent. Grossièrement, l'événement évolutif d'inversion, coupe la molécule d'ADN en deux endroits, retourne la portion entre les deux coupures et ressoude [49]. Si bien que les marqueurs, en plus de changer d'ordre, changent d'orientation.

Deux versions de notre problème de tri par inversion existent alors, celle ne considérant pas l'orientation des marqueurs, et une seconde qui l'encode en ajoutant un signe + ou - devant chaque nombre de la permutation selon l'orientation du marqueur qu'il représente [3]. Le premier problème a été montré NP-difficile par Caprara [21], alors que le second a été résolu par Hannenhalli et Pevzner [52]. Leur article, intitulé « Comment transformer les choux en navets » est fondateur de la théorie connue depuis sous le nom de "Théorie d'Hannenhalli et Pevzner". Ils introduisent en particulier le fameux *graphe de breakpoints* avec ses arêtes "du désir" et "de la réalité". Le graphe de breakpoints représente une permutation signée (*cf.* figure 1.5), ses sommets sont les extrémités de chaque marqueur (2 extrémités par marqueur [4], donc $2n$ sommets) et il possède deux types d'arêtes :

3. Le signe plus est généralement omis.

4. Les extrémités d'un marqueur représentent son début et sa fin dans le sens de sa lecture sur l'ADN, un élément positif de la permutation aura donc son début (d) à gauche et sa fin (f) à droite alors que ce sera le contraire pour un élément négatif.

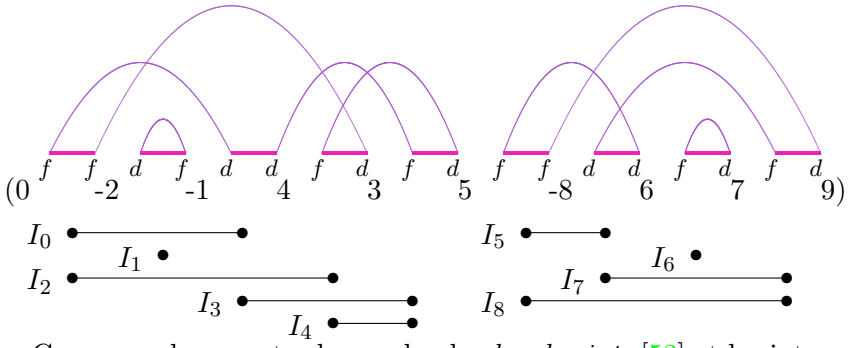

FIGURE 1.5 – Correspondance entre le graphe des *breakpoints* [52] et les intervalles élémentaires [10] pour la permutation P=(0 -2 -1 4 3 5 -8 6 7 9). Au-dessus de P est représenté le graphe des *breakpoints* avec ses arêtes du désir en traits fins et violets et ses arêtes de la réalité en traits épais et roses. Au-dessous, les 9 intervalles élémentaires, on remarque que ces intervalles et les arêtes du désir se correspondent.

— celles qui relient la fin d'un marqueur au début du suivant dans l'ordre de la permutation identité, représentant ce que l'on souhaite, ce sont les arêtes du désir ;

— celles qui relient la fin d'un marqueur au début du suivant dans l'ordre de la permutation que nous devons trier, ce sont les arêtes de la réalité.

La distance d'inversion peut se lire sur le graphe des breakpoints en fonction du nombre de cycles mais en ajoutant des valeurs correctives correspondant, sans entrer dans les détails, au nombre d'"obstacles" (*hurdles*) et au fait que la permutation soit une "forteresse" (*fortress*) ou pas.

Il faudra attendre encore quelques années pour avoir une présentation plus simple de cette théorie par Anne Bergeron dans [10], qui introduit les intervalles élémentaires (*cf.* figure 1.5). Le calcul de la distance ainsi que l'algorithme de tri se base alors sur les cycles formés par ces intervalles, leur graphe de chevauchement et un arbre représentant les composantes de la permutation [5]. Les paramètres mystérieux de la théorie d'Hannenhalli et Pevzner, comme les *hurdles* et les *fortresses* se lisent de manière claire sur ces éléments. C'est en ce sens que leur théorie a été simplifiée. Plusieurs algorithmes ont progressivement amélioré les résultats initiaux. Les meilleurs à ce jour permettent de calculer la *distance d'inversion* (notée d) en temps linéaire [6] et de trouver un scénario atteignant cette distance en temps $O(n^{\frac{3}{2}})$, en utilisant l'algorithme d'Éric Tannier et coauteurs [92] et une structure de donnée particulière, due à Han [51]. Une description détaillée de ces problèmes et d'autres problèmes relevant de la combinatoire des réarrangements génomiques, ainsi que leurs mises en perspective est disponible dans le livre de Guillaume Fertin et ses coauteurs [38].

On s'intéresse plus particulièrement à une variante du problème *Sorting By Reversals*, dans laquelle on cherche à discriminer parmi les scénarios optimaux, en nombre potentiellement exponentiel [12], ceux qui seraient plus probables au niveau biologique. Pour cela, on ajoute la

5. Les composantes ne sont pas présentées ici.

contrainte qu'un scénario d'inversions ne puisse pas "casser" des intervalles de marqueurs présents à la fois dans le génome de départ et d'arrivée, de tels intervalles sont appelés *intervalles communs* [97, 55]. Ces intervalles communs modélisent le fait qu'un groupe de gènes peuvent être réarrangés dans un génome tout en restant connectés. Ce groupe de gènes a également de grandes chances d'être présent dans l'ancêtre commun des deux génomes. La *distance parfaite* (notée d_p) est alors le nombre minimum d'inversions ne cassant pas d'intervalles communs nécessaires pour trier une permutation. Pour toute permutation P, un scénario parfait de longueur minimale existe mais n'est pas forcément parcimonieux parmi tous les scénarios possibles, d'où $d_p(P) \geq d(P)$. La distance parfaite peut aussi se calculer pour un sous-ensemble $\mathcal{F}$ d'intervalles communs, on parlera alors de *distance $\mathcal{F}$-parfaite*. Calculer cette distance, même pour des familles "simples" d'intervalles communs (comme les familles d'intervalles imbriqués), est un problème NP-difficile [39].

Scénarios commutants Dans un premier temps, nous nous sommes intéressés aux scénarios parfaits et parcimonieux à la fois, de tels scénarios n'existent pas pour toutes les instances. Nous nous sommes restreints à une sous-classe des scénarios parfaits, les *scénarios commutants*. Des intervalles communs de deux permutations sont dits *commutants* s'ils s'intersectent de manière triviale, c'est-à-dire s'ils sont disjoints ou que l'un est inclus dans l'autre. Deux intervalles sont soit commutants, soit *chevauchants*. Un scénario est dit commutant si toutes ses inversions commutent deux à deux.

Nous avons montré dans [BBC04] que tout scénario commutant parcimonieux conservait tous les intervalles communs des deux permutations dans toutes les étapes intermédiaires. Nous avons montré ultérieurement que tout scénario commutant est forcément parcimonieux [BBCP05]. On peut visualiser les relations entre ces différents types de scénarios sur le diagramme de Venn de la figure 1.6. L'intersection entre les scénarios parfaits et parcimonieux peut être vide, dans ce cas, les scénarios parfaits de longueur minimale ne sont pas parcimonieux et il n'existe pas de scénario commutant.

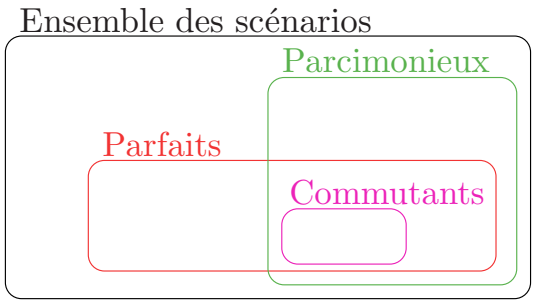

FIGURE 1.6 – Relations d'inclusion entre les différents types de scénarios. En noir, l'ensemble de tous les scénarios possibles pour deux génomes donnés ; en vert, l'ensemble des scénarios parcimonieux, c'est-à-dire ceux qui atteignent la distance d'inversion ; en rouge les scénarios parfaits, ceux qui ne cassent aucun intervalle commun ; en rose, les scénarios commutants, c'est-à-dire parfaits et dont toutes les inversions commutent.

Nous avons également caractérisé les permutations qui admettaient un tel scénario : ce sont les permutations dont le graphe de chevauchement des intervalles élémentaires (*cf.* [10]) est un arbre. Nous avons proposé un algorithme en temps linéaire permettant de décider si une permutation admettait un scénario commutant, et dans ce cas, l'algorithme effectue le tri. Ces travaux ont permis de déceler une erreur dans les données utilisées dans [47] pour

Souris =	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Rat =	-4	-3	-2	1	-13	-15	14	-16	8	9	10	-11	12	5	6	7

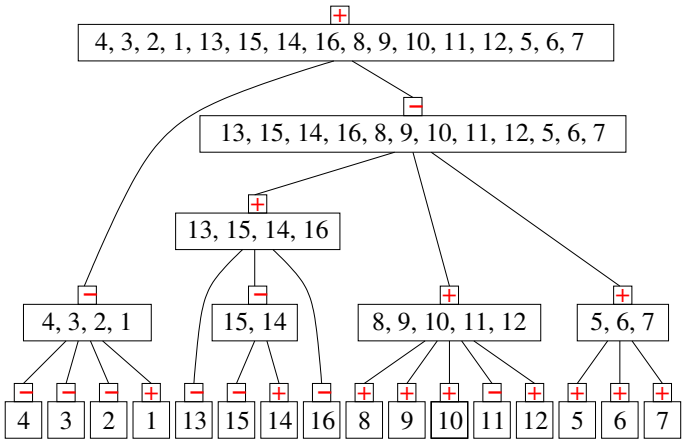

FIGURE 1.7 – Arbre des intervalles forts représentant la comparaison des chromosomes X du rat et de la souris. Cet arbre est *bien défini* car il n'a que des nœuds linéaires. L'ensemble des sommets qui ont un signe différent de celui de leur parent forme un scénario commutant (parfait et parcimonieux) qui transforme le chromosome X du rat en le chromosome X de la souris en 11 inversions : $(4,3,2,1)$, (1), $(13,15,14,16,8,9,10,11,12,5,6,7)$, $(13,15,14,16)$, (13), $(15,14)$, (14), (16), $(8,9,10,11,12)$, (11), $(5,6,7)$.

reconstruire des scénarios évolutifs entre l'humain, la souris et le rat, à l'occasion du séquençage de ce dernier[6].

Scénarios parfaits Dans un deuxième temps, nous sommes revenus au problème de la distance parfaite en étudiant les propriétés des familles d'intervalles communs. Le nombre d'intervalles communs d'une permutation[7] peut être quadratique. Cependant, on peut donner une "base" d'intervalles permettant de définir/générer tous les intervalles communs, cette base est constituée des *intervalles forts*, qui eux sont en nombre linéaire. Un intervalle commun est fort s'il commute avec tous les autres intervalles communs. Les singletons et la permutation entière sont des intervalles forts triviaux. De part leur définition, la relation d'inclusion des intervalles forts définit un arbre à n feuilles, les singletons, et dont la racine est la permutation entière. Cet arbre est appelé *arbre des intervalles forts* et peut-être relié à une structure de donnée largement utilisée dans la théorie de la *décomposition modulaire de graphes*, appelée *PQ-arbre* [33, 64, 11]. Sans rentrer dans le détail de leur calcul, l'arbre des intervalles forts possède deux types de nœuds :

— *linéaires*, si ses enfants sont "bien ordonnés" (représentés par des rectangles) ;

— *premiers*, sinon (représentés par des ellipses).

6. La rat a été le 3^e mammifère à avoir été séquencé, après l'homme en 2001 et la souris en 2002.

7. Par abus de langage, on parle des intervalles communs d'**une** permutation P mais ce sont les intervalles communs entre P et la permutation identité vers laquelle on souhaite effectuer le tri.

Humain =	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Rat=	-13	-4	5	-6	-12	-8	-7	2	1	-3	9	10	11	14	-15	16

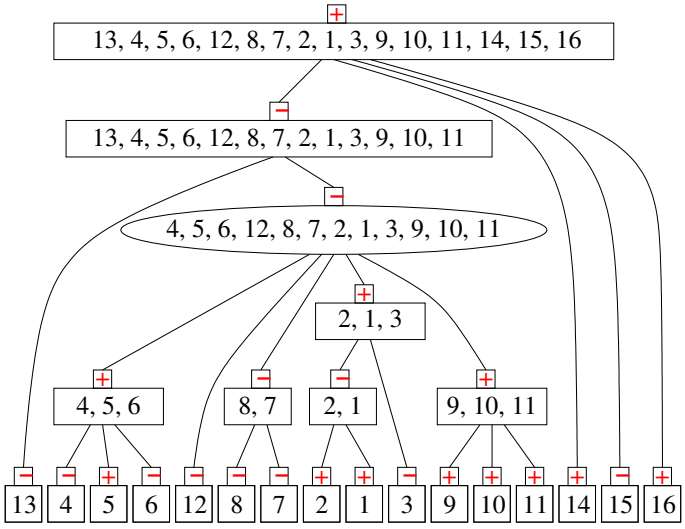

FIGURE 1.8 – Arbre des intervalles forts représentant la comparaison des chromosomes X de l'humain et du rat. Cet arbre est *non ambigu* car aucun nœud premier n'a de parent premier. Un scénario parfait de longueur minimale est obtenu en triant les 5 enfants du seul nœud premier : $(4, 5, 6)$, (12), $(8, 7)$, $(2, 1, 3)$ et $(9, 10, 11)$ en ordre décroissant en utilisant n'importe quel scénario parcimonieux qui trie sa permutation quotient (2 -5 -3 1 4), et ensuite en inversant les nœuds linéaires qui ont un parent linéaire de signe différent. La longueur de ce scénario est 13.

Nous utilisons une version signée des arbres d'intervalles forts, où le signe du nœud nous indique dans quel sens nous devrons le trier. Des exemples de tels arbres calculés sur des données réelles de l'humain, du rat et de la souris [47] sont représentés aux figures 1.7 et 1.8.

Nous avons montré dans [BBCP07] deux résultats importants reliant intervalles communs, arbres des intervalles forts et scénarios parfaits :

1. Un intervalle commun est soit un nœud de l'arbre des intervalles forts, soit l'union de fils *consécutifs* d'un nœud *linéaire* de l'arbre.
2. Un scénario est parfait si et seulement si chaque inversion qu'il contient est soit un nœud de l'arbre des intervalles forts, soit l'union de fils d'un nœud *premier* de l'arbre.

L'arbre des intervalles forts est donc un moyen compact de représenter les intervalles communs mais aussi le support sur lequel on a pu caractériser les scénarios parfaits de longueur minimale. Nous avons défini trois types d'arbres d'intervalles forts :

— *bien définis* quand ils ne contiennent aucun nœud premier (*cf.* figure 1.7) ;

— *non ambigus* quand aucun nœud premier n'a de parent premier (*cf.* figure 1.8) ;

— *ambigus* sinon, dans ce cas, il existe au moins une arête reliant deux nœuds premiers.

Pour les deux premières classes d'arbres, on est capable de donner un signe à tous les nœuds de l'arbre, pour la dernière en revanche, on ne peut pas décider à l'avance dans quel sens trier les

nœuds premiers fils d'un nœud premier. En fonction de cette caractérisation des arbres, nous donnons des algorithmes permettant de calculer des scénarios parfaits de longueur minimale :

Type d'arbre	**Complexité**	**Algorithme**
Bien défini	Linéaire	Inverser les nœuds ayant un signe différent de celui de leur parent, exemple à la fig. 1.7
Non ambigu	Sous-quadratique	Trier les nœuds premiers puis inverser les nœuds linéaires ayant un signe différent de celui de leur parent, exemple à la fig. 1.8
Ambigu	FPT	Algorithme de type "*brute force*" : essayer tous les signes possibles pour les nœuds non signés puis procéder comme ci-dessus

La classe des arbres bien définis correspond exactement aux permutations qui admettent un scénario commutant, permutations définies au paragraphe précédent. Ces arbres sont aussi appelés *co-arbres* dans la théorie de la décomposition modulaire de graphes [24]. De plus dans ce cas il existe un unique scénario parfait et parcimonieux (à l'ordre des inversions près). Pour la classe des arbres non ambigus, on peut trouver un scénario en temps $O(n^{\frac{3}{2}})$, en utilisant notre algorithme et les dernières améliorations du tri par inversion [92, 51]. Et pour la classe des arbres ambigus, nous avons montré que l'on pouvait obtenir un algorithme FPT (*Fixed Parameter Tractable*)[34]. Dans [BCP08] nous avons amélioré cette complexité en temps en faisant porter l'exponentielle par un paramètre plus petit que celui de [BBCP07] à savoir d, le degré premier maximal de l'arbre des intervalles forts (d est le nombre maximum d'enfants premiers d'un nœud premier). En tenant compte de la complexité actuelle pour le tri par inversion, on a donc pour la classe des arbres ambigus un algorithme qui fonctionne en temps $O(2^d n^{\frac{3}{2}})$ et en espace $O(n^2)$.

Perfect DCJ

Les inversions ne sont pas les seuls réarrangements génomiques qui perturbent l'ordre des gènes. On peut citer les transpositions, les translocations, les fusions et les fissions de chromosomes. Différents modèles incluant des sous-ensembles de ces réarrangements ont été utilisés pour calculer des scénarios et des distances évolutives. Pour des modèles, apparemment compliqués, comme celui contenant inversions, translocations, fusions et fissions, des algorithmes polynomiaux ont été trouvés par Hannenhalli et Pevzner [53]. La distance en terme de ces 4 réarrangements est d'ailleurs nommée d_{HP} en référence à ces auteurs. Dans d'autres modèles d'apparence plus simples, comme celui ne contenant que des transpositions, le problèmes du tri dans le cas non signé, introduit par Bafna et Pevzner [7, 8] est resté de complexité indéterminée pendant 15 ans. Il a été montré NP-difficile par Bulteau et coauteurs [20] en faisant une réduction au problème SAT. Le meilleur ratio d'approximation connu à ce jour est $1,375$, dû à Elias et Hartman [36].

Au milieu de la décennie précédente, l'introduction d'un nouveau type d'événement, appelé *DCJ* pour *Double Cut and Join* par Sophia Yancopoulos et coauteurs [100] a permis d'établir

	$\mathcal{F}$ **imbriquée**	$\mathcal{F}$ **faiblement séparable**	$\mathcal{F}$ **quelconque**
Inversions	NP-difficile [39]	Polynomial [BBCP07]	FPT [a][BCP08]
DCJ	Polynomial [BCC+09]	NP-difficile [BCC+09]	FPT [b] [BCC+09]

a. $O(2^d n^{\frac{3}{2}}\sqrt{\log(n)})$ où d est le degré premier maximal. Complexité qui passe à $O(2^d n^{\frac{3}{2}})$ grâce aux améliorations du tri par inversions.

b. $O(2^l n^2)$ où l est le nombre de nœuds linéaires d'un certain type.

FIGURE 1.9 – Résultats de complexité, sous les modèles évolutifs inversions ou DCJ, pour le problème du calcul d'un plus court scénario $\mathcal{F}$-parfait, c'est-à-dire ne cassant pas les intervalles de la famille $\mathcal{F}$, $\mathcal{F}$ étant un ensemble d'intervalles communs. Les algorithmes FPT sont utilisables sur les familles $\mathcal{F}$ particulières pour lesquelles le problème a été montré NP-difficile.

un cadre permettant de modéliser tous les événements évolutifs cités plus haut. En effet, chacun de ces événements peut s'exprimer en terme de "cassures" puis d'"unions" de segments d'ADN permettant de les modéliser comme des opérations DCJ. Il faut toutefois noter que le modèle DCJ autorise la création temporaire de chromosomes circulaires. Des algorithmes linéaires à la fois pour la distance DCJ et le tri par DCJ ont été trouvés par Anne Bergeron et coauteurs [13]. Comme pour les autres modèles évolutifs, il peut exister plusieurs scénarios optimaux en nombres d'opérations DCJ, nécessitant encore d'ajouter des critères pour discriminer entre eux.

Dans ce contexte, nous nous sommes intéressés dans [BCC+09] à définir et calculer ce que pourrait être une distance DCJ parfaite. Nous avons introduit la notion de scénarios DCJ parfaits, qui ne cassent pas les familles d'intervalles communs. Dû à la création de chromosomes circulaires temporaires, nous avons légèrement adapté la définition de conservation d'intervalles communs par le scénario de la manière suivante : un intervalle commun à deux génomes est conservé par une opération de DCJ si ses gènes sont répartis de manière contiguë dans au plus un segment chromosomique linéaire et un nombre non limité de chromosomes circulaires.

Nous nous intéressons au problème général où nous cherchons un scénario parfait de longueur minimale préservant un sous-ensemble $\mathcal{F}$ des intervalles communs. Nous nous sommes servis à nouveau de l'arbre des intervalles forts et nous avons caractérisé certaines familles d'intervalles communs. Les familles d'intervalles *imbriquées* sont des familles où tous les intervalles sont forts, c'est-à-dire qu'ils ne se chevauchent pas. Les familles d'intervalles *faiblement séparables* sont celles où tout intervalle est soit chevauchant, soit l'union de deux intervalles chevauchants. La première famille génère des arbres d'intervalles forts ne contenant que des nœuds premiers, c'est une sous-classe des arbres ambigus, alors que la deuxième famille génère des arbres d'intervalles forts non ambigus.

Le modèle DCJ a souvent donné des problèmes algorithmiques de complexité comparable à celui du modèle avec inversions uniquement. Pourtant, dans le cadre des scénarios parfaits, nous

avons montré que ces deux modèles avaient des comportements différents :

— si la famille d'intervalles communs est *imbriquée*, nous avons montré que trouver un scénario DCJ parfait de longueur minimum est polynomial en temps, tandis que c'est NP-difficile pour les inversions [39] ;

— si la famille est *faiblement séparable* nous avons montré que le problème DCJ est NP-difficile, tandis que cette version est polynomiale dans le cas des inversions [BBCP07] ;

— le problème général de calculer un scénario de longueur minimale qui préserve une famille $\mathcal{F}$ d'intervalles communs est NP-difficile sous les deux modèles.

La table de la figure 1.9 résume ces résultats. Dans [BCC+09], nous donnons un algorithme exact de complexité en temps $O(2^q n^2)$ où la partie exponentielle est bornée par le nombre q de nœuds linéaires de l'arbre des intervalles forts. Cet algorithme conduit directement a une complexité quadratique dans le cas des familles d'intervalles imbriqués pour lesquelles l'arbre ne contient que des nœuds premiers. Les différences de comportement entre le modèle DCJ et inversion peuvent se caractériser sur l'arbre des intervalles forts : la difficulté dans les tris parfaits par inversions vient des nœuds premiers fils d'un nœud premier alors que dans le tri parfait par DCJ, ce sont certains nœuds linéaires qui posent problème.

Recherche de scénarios de réarrangements parfaits entre 2 génomes

Dans ce problème, nous avons 2 génomes modélisés comme une suite de marqueurs génomiques (uniques et universels) et il s'agit d'inférer un scénario évolutif comprenant le moins d'événements possibles et ne cassant pas des intervalles communs entre les deux génomes. Nous avons étudié ce problème sous deux modèles évolutifs différents, inversions seulement ou DCJ seulement.

Algorithmique & Graphes Graphe des breakpoints, de chevauchements. Arbre des intervalles forts (similaires aux PQ-arbres). Intervalles communs, décomposition modulaire de graphe. Preuve de complexité et complexité paramétrique.

Applications/Logiciel Chromosome X des premiers mammifères séquencés (détection d'erreur dans l'assemblage du chromosome X de la souris [BBC04]).

Publications/Impact Ces travaux ont donné lieu aux publications [BBCP07], [BBC04], [BCP08] et [BCC+09] citées respectivement 53 fois, 47 fois, 13 fois et 11 fois [a]. Dans le cadre des scénarios parfaits, nous avons montré qu'en dépit de la similitude des deux opérations, les instances faciles pour les inversions sont les instances difficiles pour les DCJ, et *vice versa*.

a. d'après http://scholar.google.fr/ en juin 2016.

Histoires d'un ensemble de génomes et de leurs ancêtres : coévolution

Les travaux présentés dans cette section correspondent aux articles des annexes B et C respectivement page 57 et page 65.

Le terme *coévolution* est employé ici pour désigner l'évolution conjointe de marqueurs génomiques. Les êtres vivants évoluent par spéciations et les gènes portés par leurs génomes évoluent également mais d'une manière qui leur est propre : ils subissent les mêmes spéciations et sont affectés par d'autres événements comme des duplications et des pertes. Les gènes ne sont pas indépendants de leurs voisins car ils sont portés par une même molécule d'ADN et car leurs produits interagissent. Un événement de réarrangement génomique touche rarement un seul gène à la fois et les taux de substitutions de nucléotides par exemple, ne sont pas spécifiques à un seul gène mais bien à une zone de la molécule [46]. C'est pourquoi reconstruire l'histoire évolutive d'un gène sans tenir compte de celle de son voisin peut mener à des incohérences. À l'opposé, reconstruire l'histoire de tous les gènes en même temps conduit à une impasse côté temps calcul comme on va le voir de suite. Dans les travaux présentés dans cette section, nous avons fait le choix d'un compromis en reconstruisant l'histoire de paires de gènes voisins, autrement dit l'histoire d'*adjacences* de gènes.

Ces travaux sont une suite logique de ceux de la section précédente puisque nous nous intéressons maintenant à la reconstruction de génomes ancestraux. Auparavant, nous produisions des histoires évolutives entre deux génomes sous certaines contraintes. Le but est toujours de retrouver la véritable histoire évolutive entre ces deux génomes d'espèces contemporaines, appelons-les G_1 et G_2. La théorie de l'évolution nous indique que toutes les espèces actuelles descendent d'un même ancêtre commun et que l'on peut donc encoder leurs relations de parentés par un arbre phylogénique. Arbre où les feuilles sont les espèces actuelles et la racine notre dernier ancêtre commun, le fameux LUCA (pour *Last Universal Common Ancestor*). Chercher la véritable histoire évolutive de G_1 à G_2, c'est en fait chercher les histoires évolutives de A à G_1 et de A à G_2, avec A le plus proche ancêtre commun de G_1 et G_2. Pour cela, on peut introduire un troisième génome actuel G_3 comme *outgroup*, qui va venir fixer cet ancêtre commun le long du chemin de G_1 à G_2 (*cf.* figure 1.10). Et le problème de trouver un scénario consistant pour G_1 et G_2 revient alors à trouver un génome A qui minimise la somme des distances $d(G_i, A) \forall i \in [1..3]$, c'est le *problème de la médiane*, qui est notoirement NP-difficile pour les principaux modèles évolutifs classiquement considérés [93].

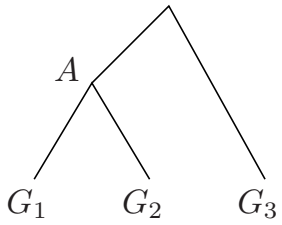

FIGURE 1.10 – Illustration du problème de la médiane : trouver un génome A qui minimise la somme des distances $d(G_i, A)$.

On s'intéresse de manière plus générale aux relations de parentés entre toutes les espèces

contemporaines et à la reconstruction de leurs ancêtres. Au vu du résultat que je viens d'énoncer, on se rend compte que modéliser des génomes comme des suites de marqueurs génomiques et chercher explicitement quels réarrangements se sont produits n'est pas une piste qui nous mènera à des algorithmes rapides. Nous avons donc envisagé une autre piste. Il s'agit de considérer les génomes comme des ensembles d'adjacences de gènes ou de marqueurs génomiques. Deux marqueurs sont adjacents s'ils sont présents côte à côte sur le génome. Un génome $G = (m_1, m_2, m_3, m_4, m_5)$ devient alors $G = \{m_1m_2, m_2m_3, m_3m_4, m_4m_5\}$. Notez qu'à partir de l'ensemble des adjacences on peut retrouver la structure du génome, c'est-à-dire l'ordre dans lequel y apparaissent ses marqueurs. Ces marqueurs ont des histoires évolutives encodées sous forme d'arbres phylogéniques et sont souvent bien étudiés par ailleurs [2]. Nous nous intéressons ici à l'histoire évolutive des adjacences de gènes que nous présentons sous forme d'*arbres phylogénétiques d'adjacences* [BGB+12a]. Ceux-ci nous donnent accès aux adjacences ancestrales nous indiquant quels gènes ancestraux sont voisins. En combinant ces informations on trouve un ordre partiel des gènes dans les génomes ancestraux.

DeCo : reconstruction de génomes ancestraux

Dans nos premiers travaux [BGB+12a] (annexe B, p. 57), nous avons modélisé l'évolution des adjacences de gènes en fonction de l'évolution des gènes. Notre modèle inclut les événements évolutifs affectant les espèces (spéciations), les gènes (duplications, pertes) et les adjacences (duplications, pertes, créations et cassures). Pour contourner la difficulté combinatoire, nous avons supposé que les adjacences évoluaient indépendamment les unes des autres dans un génome. En adoptant ce point de vue, on ne modélise plus directement les événements de réarrangements génomiques mais on les perçoit aux travers des remaniements de l'ordre des marqueurs dans les espèces.

Nous posons le problème en ces termes : trouver une histoire évolutive des adjacences, qui respecte les histoires évolutives des gènes constituant ces adjacences, et qui soit de coût minimum. Nous définissons le coût d'une histoire évolutive d'adjacences comme la somme des coûts des événements de création et de cassure d'adjacences. Ce parti pris sur l'indépendance des adjacences nous permet de travailler dans le cadre très agréable de la *programmation dynamique*. À l'aide d'une variante de l'algorithme de Fitch-Sankoff [41, 85], nous sommes capables de trouver une histoire évolutive de coût minimal en temps quadratique. Notre algorithme, implémenté dans le logiciel DeCo, pour Detection of Coevolution, est donc capable de travailler sur de grands jeux de données, tout en n'ayant pas de contrainte sur l'unicité et l'universalité [8] des marqueurs génomiques qu'il considère. Une fois les arbres d'adjacences inférés, il suffit de regarder pour chaque espèce ancestrale les adjacences prédites et ainsi, en les mettant bout à bout, reconstituer un ordre partiel des marqueurs ancestraux sur les génomes ancestraux.

Le point fort de notre méthode en est aussi son point faible puisqu'en considérant les adjacences de manière indépendante, on ne contraint plus les marqueurs à avoir seulement deux voisins. Les gènes ancestraux peuvent donc se retrouver adjacents à plus de 2 autres gènes, rendant ainsi les chromosomes ancestraux non linéaires (ce qui n'est biologiquement pas probable). Ce-

8. voir p. 13 pour un rappel de ces notions.

pendant, il existe des méthodes de linéarisation permettant d'inférer des arrangements de gènes linéaires pour les chromosomes ancestraux [67]. De plus, nous avons remarqué que meilleurs étaient les arbres de gènes en entrée, moins nous inférions de gènes ancestraux avec un nombre de voisins $\geqslant 3$. Ceci nous a poussés à nous intéresser à la qualité des jeux de données et conduits à examiner les bases de données publiques comme ENSEMBL [42]. Nous avons alors constaté que bon nombre de génomes présents étaient fragmentés, c'est-à-dire que dans ces génomes, certains gènes ont seulement 1, voire 0 voisin. Ce constat a donné lieu aux travaux de la section suivante.

ARt-DeCo : inférence d'adjacences actuelles

L'assemblage de génome est un problème crucial pour la génomique comparative et d'autres disciplines, car beaucoup d'études nécessitent d'avoir des génomes complètement séquencés en entrée. Les méthodes classiques de séquençage commencent par amplifier le génome cible pour en obtenir plusieurs exemplaires. Ensuite, ces exemplaires sont coupés en petits morceaux de façon aléatoire, c'est-à-dire que les coupures ne se font pas au même endroit sur les différents exemplaires. Enfin, ces petits morceaux de séquence sont traduits sur l'alphabet $\{A, C, G, T\}$. Il s'agit alors de retrouver, à l'aide des chevauchements de ces petites séquences, la séquence complète du génome initial. Or c'est un problème difficile, NP-difficile [9]. De bonnes heuristiques existent heureusement mais ne parviennent pas toujours à assembler la totalité des morceaux, laissant les génomes sous forme de "brouillons permanents". Ces assemblages fragmentés peuvent être dus à une couverture trop faible du génome au départ ou à certaines régions, difficiles à séquencer, comme les régions répétées [96].

Les améliorations récentes des technologies de séquençage (par exemple les protocoles de séquençage *long-read*), ainsi que les progrès dans les méthodes de traitement comme les assemblages hybrides [63, 4] et les méthodes de fermeture de gaps [74, 83, 37] permettent d'obtenir l'organisation complète des génomes d'origine microbienne [62] ; cependant, le problème de l'assemblage de génome reste difficile pour les grands génomes eucaryotes [80].

Les assemblages de génomes actuels fragmentés sont caractérisés par le fait que les chromosomes sont divisés en plusieurs *contigs* regroupés en *scaffolds*, dont l'ordre relatif et l'orientation ne sont pas connus. Cette information manquante sur l'ordre et l'orientation de ces scaffolds peut masquer les adjacences conservées et conduire à reconstruire un génome ancestral tout aussi fragmenté.

Les problèmes de reconstruction de génomes ancestraux et d'assemblage de génomes actuels peuvent être vus comme un seul et même problème consistant à ordonner des marqueurs génomiques anciens ou actuels. L'analogie algorithmique entre ces deux problèmes a été remarquée dans [65] en notant une similitude entre le graphe des *breakpoints*, utilisé pour la reconstruction de l'ordre des gènes dans le génome ancestral, et de graphe de *de Bruijn* [27], utilisé dans l'assemblage du génome. Cette observation a conduit à l'élaboration d'approches visant à la fois à améliorer l'assemblage des génomes actuels dans un cadre évolutif et à reconstruire l'organisation des génomes ancestraux.

9. Shortest Common Superstring [79].

C'est dans ce contexte que nous avons étendu DeCo, en développant la méthode ARt-DeCo : Assembly Recovery through DeCo [ABC+15a] (annexe C, p. 65). Dans DeCo, nous considérions comme sûres les adjacences de gènes données en entrée. Il était naturel, sachant que les génomes étaient fragmentés, de supposer que des adjacences réelles des génomes n'étaient pas contenues dans les jeux de données. Ainsi dans ARt-DeCo, nous calculons une probabilité qu'une adjacence non observée soit réelle, en fonction du niveau de fragmentation du génome et de son nombre de chromosomes attendus. En plus de prédire des adjacences ancestrales, ARt-DeCo prédit également des adjacences dans les génomes actuels, en se basant sur la probabilité calculée et sur le score de l'histoire évolutive, reflétant le contexte phylogénétique où se trouve l'espèce. Ce faisant, ARt-DeCo permet d'améliorer l'assemblage des génomes en réduisant la fragmentation. Bien qu'ARt-DeCo ne réduise pas complètement la fragmentation des génomes, nous avons constaté que les adjacences qu'il prédit sont très fiables. Nous avons pu constater cette fiabilité *via* des expériences *in silico* en masquant artificiellement des adjacences et en essayant de les retrouver. ARt-DeCo obtient des taux de spécificité supérieurs à 80 %, c'est-à-dire que plus de 80 % des adjacences prédites par ARt-DeCo appartenaient à l'ensemble des adjacences masquées (*cf.* [ABC+15a]). Nous avons également pu comparer nos prédictions à des cartes physiques [10] indépendantes et observer qu'il y avait peu de conflits (travaux en cours sur les génomes de type *Anopheles*).

Prédiction d'adjacences anciennes et actuelles

Dans ce problème, nous avons un ensemble de marqueurs génomiques pour lesquels nous connaissons l'histoire évolutive (encodée sous forme d'arbre phylogénétique *réconcilié* [a] avec l'arbre d'espèce) et un ensemble d'adjacences de ces marqueurs dans les génomes actuels. Nous cherchons à reconstruire une histoire évolutive des adjacences de coût minimal permettant d'inférer des adjacences ancestrales, et donc l'ordre des gènes ancestraux, ainsi que adjacences actuelles (pour ARt-DeCo), permettant de réduire la fragmentation des génomes des espèces actuelles.

Algorithme Programmation dynamique principalement, c'est la combinatoire qui est intéressante. *De la haute-couture entre arbres de gènes et arbres d'adjacences.*

Graphe Des arbres pour encoder toutes les histoires évolutives. Ainsi que des graphes, que l'on souhaite être des ensembles de chemins, pour encoder la structure des génomes (sommet = gènes ; arêtes=adjacences).

Applications/Logiciel Les logiciels DeCo [b] et ARt-DeCo implémentent directement les algorithmes décrits dans les articles [BGB+12a] [c] et [ABC+15a].

a. je n'ai pas du tout parlé de réconciliation, je renvoie les lecteurs et lectrices curieu(x/ses) à [57].
b. http://pbil.univ-lyon1.fr/software/DeCo/
c. cité 22 fois d'après http://scholar.google.fr/ en juin 2016.

10. Une carte physique donne les positions relatives de marqueurs génomiques sur un chromosome.

Prédiction d'adjacences anciennes et actuelles

DeCo a été la première méthode à mettre en œuvre le principe de coévolution en parcimonie et dans un modèle évolutif comprenant duplications et pertes de gènes. Suite à ce premier modèle, plusieurs extensions ont été proposées. DeCoLT [73] est une extension de DeCo qui permet de modéliser les transferts horizontaux de gènes entre les espèces, un événement évolutif fréquent chez les bactéries par exemple. Deux extensions probabilistes ont été récemment introduites : dans [88], le critère d'optimisation est un critère de maximum de vraisemblance, tandis que DeClone [23] met en oeuvre une approche probabiliste de la parcimonie en permettant d'échantillonner des scénarios d'évolution en fonction d'une distribution de probabilité de Boltzmann-Gibbs.

Histoires de génomes atypiques : alignement de génomes de bactériophages

Les travaux présentés dans cette section correspondent à l'article de l'annexe D page 79.

Les bactériophages

Les bactériophages, aussi nommés phages, sont des virus de bactéries. Leur découverte remonte au début du 20^e siècle. Ils sont présents partout et en très grande quantité. Les phages sont connus et utiles principalement pour deux raisons :

— d'une part, car ils ont le potentiel de tuer les bactéries, causes de nombreuses maladies infectieuses chez l'homme ainsi que chez les animaux et les plantes. Cette propriété a ouvert la voie à la "thérapie par les phages" ou *phagothérapie*, qui s'est développée dans la première moitié du 20^e siècle mais qui a été laissée de côté à partir des années 1940 avec la découverte de la pénicilline et d'autres antibiotiques. Actuellement, il y a un regain d'intérêt pour la phagothérapie étant donné que la résistance des bactéries aux antibiotiques est devenue une menace pour la santé publique ;

— d'autre part, ils ont participé à l'essor de la biologie moléculaire dans les années 1960, en tant que vecteurs de clonage de gènes. C'est grâce à eux que Frédéric Sanger a obtenu son 2^e prix Nobel de chimie pour avoir réussi à séquencer de l'ADN en utilisant un phage par une méthode qui porte aujourd'hui son nom. Et c'est aussi un phage qui a été le premier organisme à être séquencé grâce à cette technique en 1977 : le phage $\phi X174$ (5386 nucléotides).

Bien que le statut d'être vivant à part entière ne leur soit pas universellement reconnu du fait de leur nature de virus, les bactériophages évoluent. La théorie de la *recombinaison modulaire*, introduite pour la première fois par Botstein en 1980 [16], postule que les phages sont des assemblages de modules (on parle d'une structure en *mosaïque*) codant pour des fonctions biologiques,

et que deux modules codant la même fonction ne présentent pas forcément des séquences similaires. Un module peut contenir plusieurs gènes. On définit la recombinaison modulaire comme l'échange de modules codant une même fonction. Ce processus de recombinaison est une stratégie d'adaptation fréquemment observée chez les phages. En effet, la comparaison de génomes de phages révèle la présence d'alternances de séquences similaires et de séquences divergentes. Cette particularité d'organisation en mosaïque rend inadaptés les outils classiques d'alignement.

ALPHA : un aligneur de bactériophages

Dans l'article [BCP+16] (annexe D, p. 79), nous nous sommes intéressés à la comparaison simultanée de plusieurs génomes de bactériophages. Ceci afin de mettre en évidence leur organisation en modules et surtout de mettre leurs modules en correspondance, ce qui ouvre la voie au transfert d'annotations fonctionnelles, comme expliqué en fin de section. La difficulté vient du fait qu'un module peut présenter des séquences totalement différentes. Plus précisément, nous cherchons à aligner des génomes *colinéaires*. Nous travaillons sur des phages de la famille des *Siphoviridae*, la plus grande famille de phages caudés [98], qui présentent cette particularité. Des génomes sont dits colinéaires si les modules exécutant la même fonction y apparaissent dans le même ordre. L'alignement que nous produisons n'est plus seulement un alignement de nucléotides mais aussi un alignement de modules.

Notre méthode calcule tous les alignements exacts entre les paires de génomes de l'ensemble de phages que nous cherchons à aligner. Les morceaux identiques ainsi détectés vont servir de points d'ancrage entre génomes et ne doivent donc pas apparaître plusieurs fois sur un même génome. Pour ce faire, nous cherchons des alignements de longueur supérieure à un seuil, qui est l'unique paramètre de notre méthode. Ces alignements induisent une relation d'équivalence entre nucléotides. Nous groupons ces classes d'équivalence dans des nœuds, et les organisons selon la relation d'ordre des nucléotides dans leurs génomes respectifs. Ceci construit le *graphe d'alignement*, processus détaillé à la figure 1.11. La particularité de notre approche est de ne pas linéariser ce graphe et de travailler directement avec l'ordre partiel. Cela nous permet de déduire quelles parties de séquences dissemblables se correspondent. Un exemple d'un tel graphe sur des données réelles, produit par le logiciel qui implémente notre méthode, ALPHA, pour ALignment of PHAges, est donné à la figure 1.12.

La prochaine étape de notre travail est de permettre le *transfert d'annotations*. Nous parlons ici d'*annotation fonctionnelle* qui a pour but de déterminer la fonction des gènes. L'annotation structurale, qui permet de localiser les gènes et les autres éléments codants, est souvent déjà effectuée par des méthodes *ab initio* se basant sur la fréquence des nucléotides [25]. Les méthodes de transfert d'annotations fonctionnelles classiques comparent les séquences à annoter avec les séquences présentes dans les banques de données [54, 68]. Ces méthodes s'appuient sur le paradigme que des séquences similaires ont une structure et une fonction conservées. Cette démarche ne peut pas s'appliquer sur les génomes de phages qui peuvent présenter des séquences différentes codant pour une même fonction, ces séquences forment un module comme défini plus haut. Avec ALPHA, nous pouvons repérer ces modules comme des colonnes de l'alignement. On peut supposer qu'il y a 5 modules dans l'alignement présenté à la figure 1.12. Ces modules

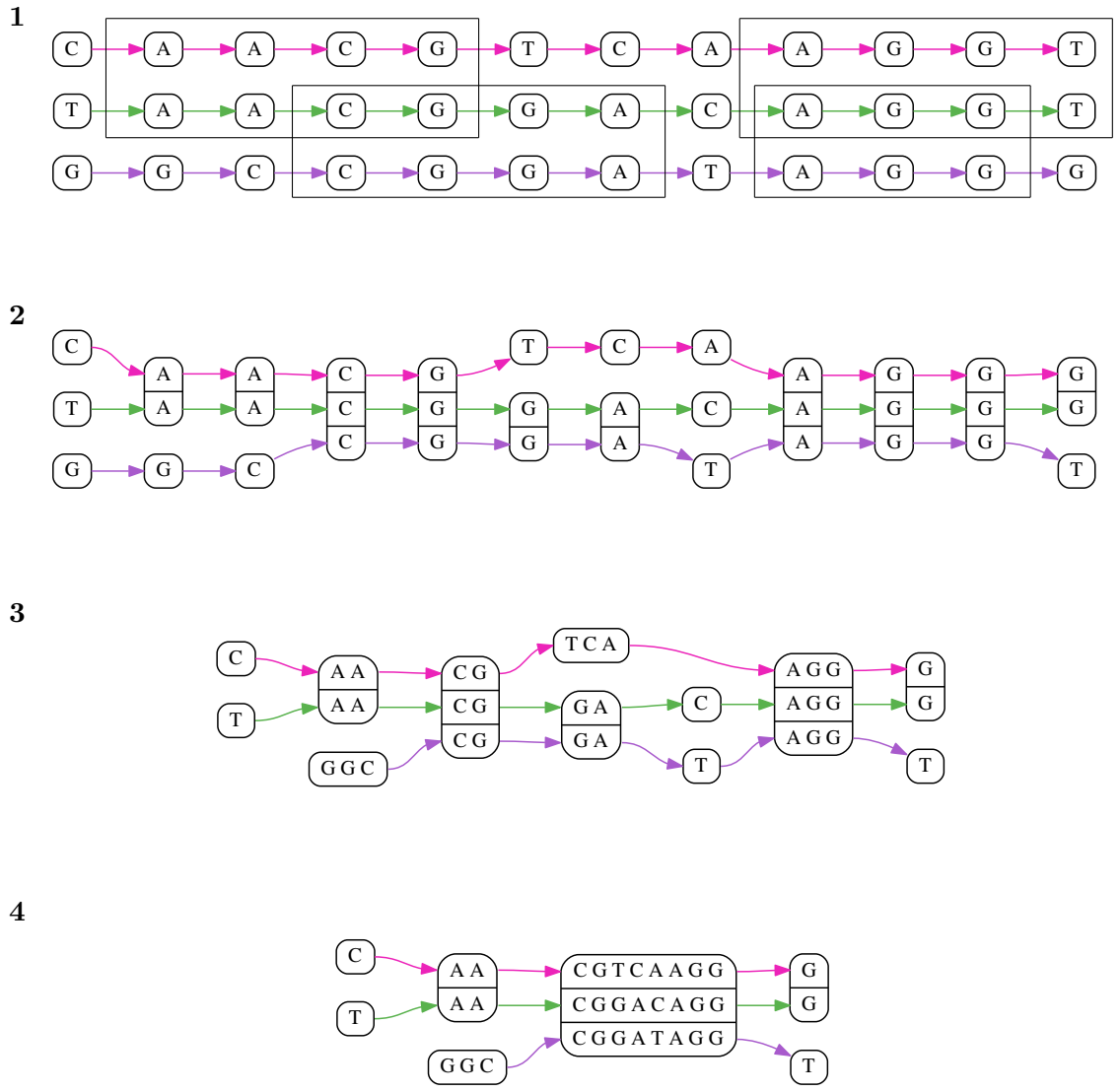

FIGURE 1.11 – Construction du graphe d'alignement de 3 génomes. Chaque génome est représenté comme la suite de ses nucléotides. L'étape **1** identifie tous les alignements locaux exacts de longueur supérieure à un seuil entre paires de séquences (encadrés en noir), ce qui définit une relation d'équivalence entre nucléotides ; l'étape **2** (*graphe des colonnes*) groupe les nucléotides équivalents en colonnes ; l'étape **3** (*graphe d'alignement*) fusionne les colonnes consécutives contenant le même ensemble de génomes ; l'étape **4** (*graphe d'alignement contracté*) agrège les nœuds situés entre deux nœuds portant sur un même ensemble de génomes, pourvu que tous les chemins aient la même longueur (en nombre de nucléotides). Cette dernière étape crée des nœuds correspondant à des alignements multiples sans gaps. Tout au long du processus, l'ordre des nœuds est préservé en suivant l'ordre des nucléotides sur leurs génomes respectifs. Il faut noter que le processus s'arrête à l'étape 2 si les génomes ne sont pas colinéaires, on peut le détecter car cela crée des cycles.

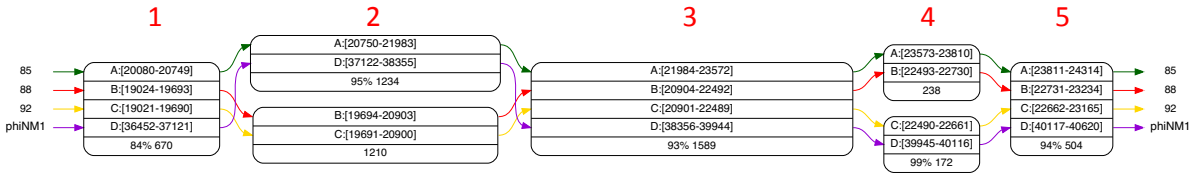

FIGURE 1.12 – Portion d'alignement par ALPHA de 4 bactériophages infectant la bactérie *S. aureus*, nommés 85, 88, 92 et phiNM1. Chaque génome est représenté par une couleur. Les nœuds de ce graphe groupent ensemble les morceaux de génomes similaires, les bornes étant indiquées entre crochets et le pourcentage de similarité étant indiqué dans la dernière ligne du nœud quand ce n'est pas 100 %. Cet alignement identifie clairement 5 modules et leurs variants.

sont constitués d'un seul variant (dont les séquences sont regroupées dans un même nœud), comme aux colonnes 1, 3 et 5 ; ou de deux variants (les séquences de chaque variant ont leur propre nœud), colonnes 2 et 4. Dans ce dernier cas, ces variants peuvent présenter entre eux des séquences très différentes bien que codant pour une même fonction, sans identité détectable ou avec moins de 13 % d'identité au niveau protéique par exemple [98, 60]. Pour chacune de ces 5 colonnes, on veut transférer les annotations fonctionnelles des gènes annotés, si celles-ci sont concordantes, aux gènes non annotés de la même colonne. Cependant, seul ALPHA peut repérer les modules des colonnes 2 et 4 et proposer un transfert d'annotations entre les deux variants qui s'y trouvent. Cette mise en correspondance des variants d'un même module par ALPHA pourrait grandement faciliter le transfert d'annotations dans les génomes de phages.

Alignement de génomes complets de bactériophages

Dans ce problème, il s'agit d'inférer les parties recombinées des génomes en utilisant des alignements exacts. Ce faisant nous pouvons proposer un alignement multiple des génomes de phages mettant en évidence leur structure en mosaïque et leur organisation en modules. L'identification de ces modules ouvre la voie au transfert d'annotations.

Algorithmique & Graphes Graphe d'alignement, DAG, ordre partiel.

Applications/Logiciel Nous avons réalisé le logiciel ALPHA [a] qui est le premier logiciel d'alignement multiple de génomes de bactériophages. Les applications directes d'ALPHA, outre l'étude des mécanismes évolutifs chez les phages, est la détection des modules génomiques de phages et le transfert d'annotation.

Publications/Impact Ces travaux ont donné lieu à la publication [BCP+16]. Ils font également l'objet d'une collaboration soutenue depuis 2014 entre Anne Bergeron (LaCIM, UQÀM, Montréal), Krister Swenson, Annie Chateau (LIRMM) et moi-même.

a. http://www.lirmm.fr/~swenson/alpha/alpha.htm.

Projet de recherche

Rien ne se perd, rien ne se crée, tout se transforme.

Antoine Lavoisier

« Traité élémentaire de chimie », 1789.

Je présente dans cette partie les projets de recherche liés aux deux thématiques de recherche que je poursuis actuellement, la *coévolution*, présentée dans la section 1.3, et l'*alignement de bactériophages*, présenté dans la section 1.4, et qui font l'objet de mes publications les plus récentes jointes en annexe. Je commence par détailler les pistes de travail en cours ou déjà planifiées (section 2.1). Dans chacun des deux thèmes, le but est de répondre à des questions biologiques qui s'inscrivent dans des problèmes plus généraux de "finition de génomes" (*genome finishing*[1]) et de compréhension de leur évolution, sujets d'actualité après l'ère du séquençage massif de génomes complets. Pour le premier thème, il s'agit de fournir des outils pour *améliorer l'assemblage des génomes* tout en reconstruisant leur évolution (section 2.1.1) et pour le second, de fournir des outils pour comprendre l'évolution des bactériophages et *aider au transfert d'annotations fonctionnelles* (section 2.1.2). J'expose également les convergences et les liens entre ces deux thématiques (section 2.1.3). Puis je conclue cette section en donnant des perspectives de recherche à plus long terme (section 2.2).

Histoires en cours

Coévolution et aide à l'assemblage

Dans cette partie, nous nous intéressons aux améliorations possibles de notre méthode ARt-DeCo qui permet l'inférence d'adjacences actuelles et ancestrales. C'est le travail de thèse de Yoann Anselmetti (ISE-M) que je co-encadre avec Éric Tannier (LBBE, Lyon) et Annie Chateau (LIRMM). Yoann terminera sa thèse à l'automne 2017. L'ANR ANCESTROME qui finance le fonctionnement sur ce thème se terminera également fin 2017. Ces recherches sont réalisées en collaboration avec Cédric Chauve (Simon Fraser University, Canada). Yoann a effectué un

1. De mon point de vue, le problème de la finition de génome consiste à obtenir les séquences d'ADN complètes de chaque chromosome ainsi que les annotations structurelles et fonctionnelles des éléments qu'ils portent.

stage de 6 mois à Vancouver chez Cédric au début de sa deuxième année de thèse (juillet-décembre 2015). Pendant ce stage, il a participé à un projet de séquençage de plusieurs génomes de moustiques du type *Anopheles*, dirigé par un consortium de chercheurs [70]. Yoann sera d'ailleurs coauteur de la publication d'une 2^e version de leurs assemblages (prévue fin 2016).

Nos travaux sur l'aide à l'assemblage commencent à être connus et nous avons plusieurs demandes de collaborations autour de jeux de données qui pourraient être intéressants à étudier, d'une part pour valider notre méthode sur de nouveaux génomes et d'autre part car cela soulève de nouvelles questions méthodologiques. Plusieurs travaux sont en cours ou planifiés sur cette thématique d'aide à l'assemblage mais aussi plus généralement sur notre méthode d'inférence d'histoires évolutives, je les expose brièvement ci-dessous :

1) Intégration de données de séquençage Jusqu'ici, nous considérions comme sûres les adjacences présentes dans les banques de données et attribuions une probabilité d'existence à celles non observées selon le niveau de fragmentation de l'espèce. En intégrant des données de séquençage, nous disposons d'informations supplémentaires sur l'existence de certaines adjacences. En effet, les données de séquençage de type *paired-end*[2] permettent de donner une estimation de distance entre deux marqueurs génomiques. Plus précisément, ce sont des méthodes comme BESST [82] qui infèrent à partir de ces données *paired-end* des *liens de scaffolding*. Nous avons intégré ces liens de scaffolding, traduits en scores, à ARt-DeCo, permettant ainsi de prendre en compte des données physiques issues du séquençage dans l'inférence d'adjacences. La version du programme ainsi modifiée se nomme AD-seq. C'est avec cette version que nous avons étudié le jeu de données des 18 *Anopheles*. Nous commençons à rédiger une publication sur l'étude de ce jeu de données, incluant les avancées méthodologiques et des discussions biologiques sur les chromosomes X des *Anopheles*. En particulier, les gènes portés par les chromosomes X soutiennent une phylogénie alternative de ces moustiques [44]. Notre méthode peut apporter des éléments de réponses en reconstruisant l'histoire évolutive des gènes portés par ces chromosomes X ainsi qu'en améliorant l'assemblage des génomes et donc le positionnement chromosomique des gènes. Nous prévoyons une soumission à l'automne 2016.

2) Visualisation des résultats et comparaison avec une carte physique Nous pouvons représenter la structure des chromosomes par des graphes où chaque marqueur génomique est un sommet et il y a une arête entre deux sommets si les marqueurs associés sont adjacents. Un ensemble de n chromosomes est alors représenté par n graphes linéaires. Avec ce type de graphes, nous pouvons représenter les données connues et considérées comme fiables, comme celles des cartes physiques, et y ajouter les prédictions avec des scores de confiance, comme ceux que l'on peut obtenir avec DeClone [23] (*cf.* figure 2.1). Un problème est alors de mesurer la compatibilité de nos prédictions avec les données considérées comme fiables. On peut traduire cela en problème de graphe où l'on cherche à minimiser le nombre d'arêtes à retirer pour que les ordres partiels induits par la carte et les prédictions soient compatibles. C'est le sujet de Lisa de Mattéo, stagiaire de M1 Informatique, que j'encadre actuellement. Le problème se complexifie

2. Ce type de séquençage permet de séquencer simultanément deux fragments distants sur la molécule d'ADN.

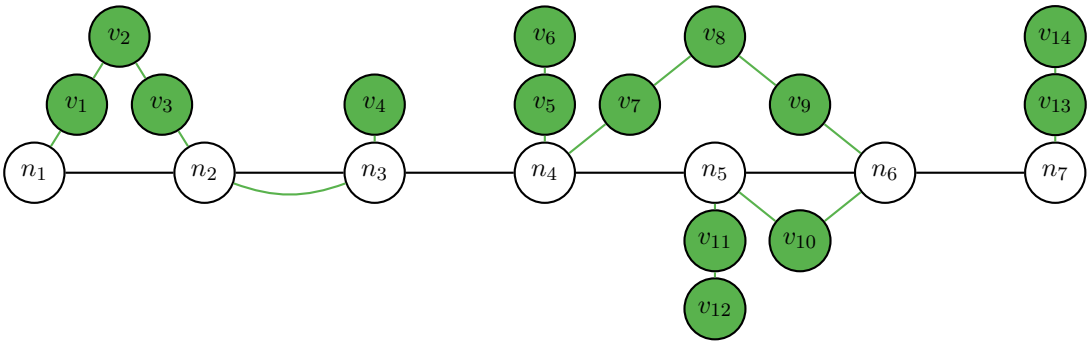

FIGURE 2.1 – Compatibilité de cartes et de prédictions. Dans ce graphe, les sommets représentent des marqueurs génomiques et les arêtes des relations d'ordre entre marqueurs. Les nœuds et arêtes noirs sont les informations issues d'une carte physique indiquant les positions relatives des marqueurs sur le chromosome. Ces cartes sont souvent peu denses en nombre de marqueurs. Sur ce graphe, on ajoute nos prédictions d'adjacences (ici linéaires et sans score de confiance) sous forme d'arêtes vertes entre sommets déjà présents dans la carte ou non. Ces arêtes vertes indiquent des adjacences "strictes" entre marqueurs [a]. Sur cet exemple, on voit deux arêtes entre les sommets n_2 et n_3, ce qui signifie que la carte et les prédictions sont en accord parfait. Entre les marqueurs n_1 et n_2, la carte et les prédictions sont compatibles car on peut ordonner les marqueurs en respectant les ordres partiels induits : $(n_1, v_1, v_2, v_3, n_2)$. Par contre, entre les sommets n_4 et n_6 il y a une incompatibilité des informations de la carte (chemin noir (n_4, n_5, n_6)) et de celles des prédictions (chemin vert $(n_4, v_7, v_8, v_9, n_6)$). Pour résoudre ce conflit, on cherche le nombre minimum d'arêtes vertes à casser. Une solution ici est de casser l'arête $\{n_4, v_7\}$, on peut alors ordonner tous les marqueurs : $(n_1, v_1, v_2, v_3, n_2, n_3, v_4, v_6, v_5, n_4, v_{12}, v_{11}, n_5, v_{10}, n_6, v_9, v_8, v_7, v_{14}, v_{13}, n_7)$.

a. ce qui correspond à la notion de *couverture* dans les relations d'ordre.

lorsque nos prédictions ne sont pas linéaires (*cf.* paragraphe suivant). Le but est de déterminer de manière simple le niveau de conflit entre nos prédictions et l'ordre partiel donné par la carte. On pourra aussi comparer les méthodes de prédictions d'adjacences entre elles, ce qui n'est pas évident à l'heure actuelle, où l'on compare seulement le nombre d'adjacences proposées mais pas leur exactitude.

3) Linéarisation Comme évoqué dans la section 1.3, notre méthode considère les adjacences comme indépendantes et peut donc proposer un nombre de voisins supérieur à deux. Cela entraîne des *conflits synténiques*, c'est-à-dire des zones où la linéarité des chromosomes n'est pas respectée. Dans le cadre de projets de séquençage, comme celui des *Anopheles*, il nous faut produire des prédictions linéaires. Pour cela, nous pouvons utiliser directement des méthodes existantes de linéarisation comme [67], cependant ces méthodes sont soit des heuristiques, soit autorisent des chromosomes circulaires. Nous voulons réfléchir à un moyen de linéariser nos prédictions en prenant en compte toutes les informations dont on peut disposer ainsi que celles spécifiques à notre méthode, comme les scores de DeClone ou des liens de scaffolding, ou les propriétés structurelles du graphe qui les représente.

4) Modification du modèle Pour traiter les génomes actuels avec nos méthodes (DeCo et ARt-DeCo), nous éliminons les marqueurs génétiques pour lesquels l'ordre relatif n'est pas clair. Il arrive en effet assez fréquemment que des gènes se chevauchent, par exemple c'est le cas pour plus de 13 % des gènes chez l'humain et la souris [66, 87]. Cet élimination entraîne à mon sens une perte d'information. Nous envisageons d'étudier un modèle dans lequel au lieu de prendre en entrée un ensemble de marqueurs réduits mais ordonnés totalement, nous prenons en entrée un ensemble plus grand de marqueurs ordonnés seulement partiellement par endroit. Il faudra évaluer si nous pouvons toujours fonctionner avec le modèle évolutif des adjacences que nous avons élaboré et surtout si le cadre de la programmation dynamique tient toujours. Une autre modification à apporter à notre modèle, *a priori* plus directe à mettre en œuvre, est d'autoriser la remise en question de certaines adjacences. On pourrait imaginer un système de poids donnant plus ou moins de confiance aux adjacences ou alors une remise en cause plus systématique, au vu du voisinage par exemple. Cette remise en question des adjacences devrait rapidement être intégrée à AD-seq.

5) Validation sur données réelles et simulées Nous avons déjà testé la validité de nos approches (DeCo et ARt-DeCo) sur des jeux de données où nous avions simulé de la fragmentation ainsi que sur des données réelles. Cependant, la possibilité de confronter nos prédictions à différents types de jeux de données est toujours très intéressante. Pour tester notre dernière amélioration, AD-seq, prenant en compte des données de séquençage, nous allons créer un jeu de données simulant un génome mal assemblé. Pour ce faire, nous allons prendre les données de séquençage d'un génome bien assemblé, sous la forme d'un ensemble de *reads*, retirer aléatoirement différents pourcentages de ces *reads*, faire appel à un logiciel d'assemblage pour reconstituer un génome en imitant ce qui se fait avec les données réelles et appliquer AD-seq pour évaluer ses performances en se comparant au génome initial. Nous avons également été sollicités par des collègues sur différents types de jeux de données réelles. Chacun de ces jeux de données, en plus d'être des outils de validation d'AD-seq, nous permet de contribuer à l'exploration des questions biologiques qui y sont associées :

— Thomas Faraut (INRA, Toulouse) nous donne accès à une carte physique du génome du canard qu'il a produite et qui n'a pas été utilisée pour l'assemblage disponible dans les banques de données. Nous sommes en train de confronter nos prédictions à cette carte et soutenons, comme lui, la présence de scaffolds chimériques ;

— Benoit Nabholz (ISE-M) serait intéressé pour qu'on analyse avec AD-seq un jeu de données de 48 oiseaux [101] au sein duquel il étudie un groupe de passereaux. Un des intérêts est qu'il a la possibilité de refaire du séquençage ciblé et par exemple d'obtenir des *long reads* PACBIO [80] que l'on pourrait utiliser pour valider nos prédictions, ces reads couvrent en effet de grandes portions de génomes ;

— Sylvain Glémin (ISE-M) nous a également contactés pour voir si on pouvait détecter des problèmes d'ordre dans l'assemblage du blé, ce qui nous pousse à implémenter la remise en question d'adjacences ;

— Enfin, étant dans un laboratoire de biologie, nous avons accès à certains outils de biologie

moléculaire et nous envisageons de valider nos adjacences également par des expériences couplées de PCR (*Polymerase Chain Reaction*[3]) et séquençage. Pour cela, nous sommes en train de déterminer un ensemble d'adjacences potentielles qui répondent à plusieurs critères : appartenir à des espèces pour lesquelles nous avons accès aux tissus, avoir des bons scores de support par DeClone, se trouver en dehors d'une zone répétée qui pourrait perturber notre PCR et enfin être à une distance estimée inférieure à la capacité de recouvrement de l'expérience biologique. Nous envisageons de mener ces expériences à l'ISE-M cet automne.

6) Publication logicielle Une publication courte (2 pages) est en cours de préparation sur la réunion de toutes les variantes de DeCo (*cf.* encadré p. 25) au sein d'un même logiciel avec tous les coauteurs de ces variantes (soumission envisagée fin 2016).

Des idées plus prospectives concernent également ces travaux. À l'heure actuelle, nous considérons les gènes comme unités évolutives. On peut cependant penser que l'unité évolutive de base n'est pas le gène mais quelque chose de plus petit. De nombreux travaux s'intéressent à l'évolution des domaines protéiques qui sont des composants des protéines et donc des gènes qui les encodent [61, 9, 19]. Nous envisageons d'adapter notre méthode sur ces unités plus petites pour lesquelles ont connaît les familles et les relations [40]. Une autre piste serait de découper les gènes en exons, et de prendre ces exons comme unités évolutives. Il faudrait alors déterminer quels exons sont homologues et retracer leurs histoires évolutives pour pouvoir utiliser les principes de nos méthodes.

Toutes ces perspectives ont pour but d'améliorer notre méthode d'inférence d'histoires évolutives qui prédit de nouvelles adjacences fiables dans les génomes actuels et en réduit ainsi la fragmentation. Sur les 69 espèces de la base de donnée Ensembl[4] [42], seul le ver *Caenorhabditis elegans* a autant de scaffolds qu'attendus (7 scaffolds en l'occurrence pour 6 chromosomes plus l'ADN mitochondrial). Les assemblages de la plupart des autres espèces possèdent entre 100 et 10 000 scaffolds pour quelques dizaines de chromosomes attendus. Cela montre la réelle nécessité de travailler sur l'amélioration des assemblages d'espèces actuelles.

Il y a encore beaucoup de pistes méthodologiques à creuser sur ce sujet. Notamment en ce qui concerne la caractérisation du problème de *scaffolding et reconstruction de l'histoire évolutive simultanément*. Je compte poser sur ce thème un sujet de M2 l'an prochain (2016-17) ainsi qu'un sujet de thèse pour la rentrée 2017. Ce sujet de thèse concernerait l'algorithmique de graphe et commencerait par l'analyse des points **2**, **3** et **4** cités ci-dessus. Au vu de l'ampleur que prend cette thématique en terme de charge de travail pour l'analyse des jeux de données, je souhaite confier cette mission à un ingénieur de recherche qui pourrait s'occuper de traiter les nouveaux jeux de données sur lesquels nous sommes sollicités. Enfin, le dépôt d'un projet de recherche plus conséquent autour de cette thématique (et l'obtention de son financement) me permettrait de fonctionner plus aisément (missions, stages d'étudiants, visites entre collaborateurs, ...) et pourrait prendre le relais d'ANCESTROME.

3. C'est une technique d'amplification d'ADN basée sur des amorces permettant de cibler la zone d'intérêt.
4. Release 84, mars 2016.

Alignement de bactériophages et aide à l'annotation

Dans cette partie, nous nous intéressons aux génomes mosaïques des bactériophages. Ce travail s'effectue en collaboration avec Anne Bergeron (UQÀM, Canada), Annie Chateau et Krister Swenson (LIRMM). Nous avons mis au point un outil qui permet d'aligner ces génomes. En particulier, ALPHA est capable de mettre en correspondance des modules de bactériophages colinéaires codant pour la même fonction mais présentant des séquences très différentes. C'est le premier outil d'alignement de génomes complets de bactériophages et nous nous attachons à le faire connaître aux biologistes impliqués dans l'annotation des bactériophages. En pratique, les zones codant pour des protéines sont connues mais non annotées fonctionnellement ("*hypothetical protein*", "*gpX*" ou encore d'autres variantes). Pour un module groupant plusieurs génomes, dont une partie contient des gènes annotés fonctionnellement et l'autre partie contient des gènes sans annotation, on pourrait *transférer* des annotations. Pour cela nous considérons plusieurs niveaux :

Algorithme Nous travaillons actuellement sur la mise au point d'un algorithme d'aide au transfert d'annotation couplé à ALPHA. Cet algorithme proposera, à partir de l'ensemble des annotations fonctionnelles connues, une annotation (plus ou moins précise) à transférer aux autres gènes de fonction inconnue. Cet outil travaillera au niveau des nœuds du graphe d'alignement et permettra, quand c'est possible, un transfert d'annotations automatique. Pour cela nous réfléchissons à des notions de similarités d'annotations, à l'aide d'*ontologies*, travail que j'avais déjà abordé lors de mon post-doctorat à l'IML [BTH05, HBT09]. Cet algorithme pourra aussi valider ou invalider des modules où toutes les annotations sont connues. Ce thème a été l'objet d'un TER de M1 Bioinformatique qui a commencé à dégrossir le travail mais beaucoup de choses restent encore à préciser.

Interface graphique Nous travaillons sur l'ergonomie d'ALPHA au travers de son interface graphique pour permettre une interaction avec l'utilisateur. Notre logiciel d'alignement présente le résultat de son calcul de manière figée. Or on sait que lorsque l'on doit annoter finement un génome, on peut être amené à retoucher les bornes des zones proposées (au vu des annotations par exemple), c'est-à-dire couper des nœuds ou les fusionner. En plus de l'aspect technique, cela demande des aménagements des propriétés des nœuds du graphe pour que les inférences restent cohérentes. Ce thème a fait l'objet d'un TER de L3 Informatique. Les étudiants ont apporté de nouvelles fonctionnalités à l'interface, comme la possibilité de déplacer les nœuds, de les fusionner et de les couper. Cela rend l'interface beaucoup plus ergonomique mais d'autres améliorations sont encore nécessaires, en particulier l'intégration de connexions avec d'autres outils.

Modèle pour les génomes non-colinéaires Nous réfléchissons également à la détection, localisation et prise en charge de modules qui auraient subi des réarrangements génomiques perturbant leur ordre. La difficulté est double ici. Elle vient d'une part des modèles évolutifs pouvant intégrer des réarrangements comme les transpositions, et on a vu dans la partie géno-

mique comparative (*cf.* section 1.2) que ces événements étaient sources de complexité. D'autre part, l'identification même des marqueurs à mettre en correspondance est difficile puisque les modules peuvent ne pas se ressembler en séquences. Aligner plusieurs génomes (bien choisis) en même temps est une piste pour pallier ce problème. En effet, chaque génome d'une même famille apporte des informations légèrement différentes qui, combinées entre elles, peuvent permettre la détection de ces modules. Cependant il faut être prudent car augmenter le nombre de génomes complique aussi la combinatoire. Une fois les modules détectés, il faudra encore réfléchir à leur traitement et à la présentation des résultats dans une interface graphique pratique pour l'annotateur. Au final, notre outil pourrait ressembler au logiciel MAUVE [31, 32] dans le sens où il permettrait d'aligner plusieurs génomes en tenant compte des réarrangements génomiques. Cependant, à la différence de ce dernier, nous ne cherchons pas à aligner à tout prix et sommes plus contraints par les spécificités des génomes de bactériophages. Ceux-ci ont l'avantage d'être petits (en nombre de fonctions) et compacts (en longueur). Pour travailler sur ce sujet, nous souhaitons inviter Anne pour un séjour de deux mois début 2017. Nous avons déposé cette demande lors d'un appel à projet spécifique de l'université de Montpellier qui pourra financer partiellement cette visite.

Plusieurs pistes d'améliorations théoriques sont également à l'étude. Par exemple, ALPHA pourrait permettre la définition d'une *distance entre séquences de phages*. Cette distance pourrait être calculée sur la base du graphe d'alignement entre deux phages alignés ou entre un phage et un alignement de plusieurs phages. Elle serait utilisée pour *identifier les phages d'origine inconnue*, en comparant le nouveau phage à plusieurs alignements de phages de familles connues pour trouver de quelle famille il est la plus proche. Cette distance pourrait aussi permettre d'améliorer les classifications existantes en étant intégrée au processus d'inférence phylogénétique.

Cette thématique de recherche est encore assez prospective et de nombreuses pistes sont possibles. La finalité plus générale est la *compréhension de l'évolution des bactériophages*. Je pense qu'un étudiant en post-doctorat pourrait être à l'interface entre le côté méthodologique et l'aspect biologie évolutive de l'étude des bactériophages. Ce post-doc pourrait être moteur dans l'étude de l'évolution des phages, leur alignement n'étant qu'une première étape (*cf.* section 2.2).

Convergence et lien entre ces deux thématiques

Les deux projets de recherche présentés ci-dessus ont pour caractéristique commune, en plus d'utiliser des graphes et des algorithmes, de revenir à un "bas niveau" de données : données issues des expériences de séquençages pour le premier, correspondances exactes entre nucléotides pour le second. En effet, pour résoudre les problèmes d'alignement ou de comparaison de séquences biologiques, on a longtemps travaillé avec des données de "haut niveau" (séquences assemblées et annotées) sans en questionner leur fiabilité, ou en s'accommodant du bruit qu'elles contiennent. Par exemple, la notion de synténie en génomique comparative est venu pallier les erreurs de séquençage qui pouvaient amener à inférer des micro-réarrangements. Les données utilisées, bien que pas toujours exactes ni complètes, ont permis de commencer à comprendre différents mécanismes et de donner de premiers résultats. Par exemple, nos travaux sur les séquences répétées en tandem ont permis de résoudre des relations évolutives entre différentes populations

bien que l'on puisse douter que nous disposions du nombre exact de répétitions des différents variants dans les séquences de chaque individu. Pour aller plus loin, nous avons maintenant besoin de données exactes et complètes. On cherche donc, pour une meilleure fiabilité, à maîtriser le processus de bout en bout : des données brutes, en suivant leur différentes transformations jusqu'à la formalisation et la résolution de la question scientifique que l'on aborde. Pour cela il faut intégrer plusieurs niveaux de données, de connaissances et de méthodes, ce qui est tout à fait stimulant.

L'autre point commun de ces deux projets est le souci d'apporter des solutions concrètes à des problèmes biologiques, assemblage ou annotation ; mais aussi de produire des logiciels applicables en pratique, simples d'utilisation, et fiables car reposant sur de solides bases théoriques. En particulier, dans les deux cas, nous nous sommes attachés à réduire au maximum le nombre de paramètres de nos méthodes. En effet, un nombre trop important de paramètres empêche une compréhension fine du modèle et constitue souvent un frein à l'utilisation du logiciel car on ne sait pas quelles valeurs de paramètres choisir.

Ces deux projets de recherche sont aussi liés par le fait que l'on peut en inverser les finalités : on peut envisager de faire du transfert d'annotations avec ARt-DeCo et de l'aide à l'assemblage avec ALPHA.

La première idée est en cours de test au travers du stage de master 2 de Frédéric Bigey, chercheur INRA en formation continu dans notre master de Bioinformatique. Frédéric travaille sur les levures et dispose de génomes de certaines souches de levures qui ne sont pas très bien assemblées ni très bien annotées. Nous lui avons proposé d'étudier son jeu de données avec notre logiciel ARt-DeCo qui en plus de l'aider à finaliser l'assemblage pourra aussi aider à l'annotation. En effet, ARt-DeCo groupe dans un premier temps les adjacences pouvant avoir une origine évolutive commune en classes. Ce classement s'effectue sur la base de la position des gènes des extrémités des adjacences dans les arbres phylogénétiques des familles de ces gènes. Puis ARt-DeCo effectue son calcul de programmation dynamique et propose des adjacences ancestrales et actuelles, dont on peut mesurer la cohérence en fonction de certains critères comme le nombre moyen de voisins par exemple (attendu à 2). Nous envisageons de discriminer les annotations de gènes, au travers de leur placement dans des familles de gènes, et donc dans les phylogénies de celles-ci, au vu des résultats d'ARt-DeCo. On pourra ainsi avoir de premières indications sur les familles de gènes susceptibles d'accueillir au mieux un gène candidat à l'annotation. Il restera bien sûr ensuite à transférer une annotation précise, ce qui ne sera pas évident dans tous les cas.

La deuxième idée semble assez facile à implémenter. En effet, supposons qu'on ait des génomes de bactériophages fragmentés, c'est-à-dire en plusieurs morceaux. On pourrait utiliser ALPHA sur un génome fragmenté en incluant dans l'analyse d'autres génomes de bactériophages colinéaires non fragmentés (de la même famille ou supposés proches). Cela aura pour effet de placer les morceaux du génome fragmenté le long de l'ordre partiel induit par les génomes non fragmentés, et ainsi ordonner les morceaux du génome fragmenté.

Histoires futures

J'ai évoqué jusqu'ici des perspectives de recherche déjà plus ou moins avancées. Les travaux que j'ai présentés sur la coévolution et l'alignement de bactériophages offrent plusieurs pistes de poursuite à plus long terme. L'une d'entre elles concerne l'*étude de l'évolution des bactériophages*. Cette question scientifique provient d'un premier travail de Anne Bergeron et Krister Swenson [91] où ils essayaient de déchiffrer l'histoire évolutive des phages en retrouvant les différents événements de recombinaison modulaire qui s'étaient produits. Cependant, pour appliquer leurs algorithmes à grande échelle ils avaient besoin d'une méthode automatique permettant de détecter les modules des phages. L'outil ALPHA que nous avons récemment mis au point permet désormais de faire cela pour les phages colinéaires. Cet outil, qui permettra de compléter les annotations fonctionnelles des phages, pourrait également nous aider à en comprendre les mécanismes évolutifs. Cette question de la compréhension de l'évolution des phages peut ouvrir la voie à leur contrôle, et à leur "éducation" par des techniques spécifiques, dans le but de les utiliser pour lutter contre les bactéries multirésistantes aux antibiotiques (BMR). D'après un récent rapport de l'INSERM : *"En raison de leur fréquence élevée, de la gravité des infections dont elles sont responsables et de leur capacité à se diffuser, les bactéries multirésistantes font l'objet d'un programme de surveillance et de prévention depuis le milieu des années 1990. L'Institut de veille sanitaire [...] coordonne notamment depuis 2002, une surveillance nationale de [certaines BMR] dans les établissements de santé"*[5]. L'étude « Burden BMR » a estimé le nombre de décès attribués aux infections par des bactéries multirésistantes à 12 500 en France pour l'année 2012 [26]. De plus, la recherche de nouveaux antibiotiques n'avance pas au même rythme que se développent de nouvelles résistances. Ceci est dû au fait que les principaux groupes pharmaceutiques ont déserté ce type de recherche jugée trop peu rentable. Pour exemple, on compte 16 nouveaux antibiotiques sur la période 1983-1987, et seulement 2 entre 2008-2012 [29]. Cela rend nécessaire l'étude d'approches alternatives, telles que les bactériophages qui représentent une solution potentielle pour lutter contre ces bactéries résistantes dans le cadre de leur application en phagothérapie [48]. Une étude récente [35] a obtenu de très bons résultats pour lutter contre une pneumonie due à une BMR chez les souris et sera sûrement suivi d'essais cliniques en phase I/II[6] sur l'homme. Cependant, de nombreux freins existent encore pour envisager un usage généralisé en thérapie humaine car aucune étude n'a encore été publiée sur l'innocuité des phages, pourtant généralement admise [56]. En revanche, l'utilisation des phages pour décontaminer des surfaces, notamment en milieu hospitalier pour lutter contre certaines infections nosocomiales, mérite une attention particulière. Michael Hochberg, Oliver Kaltz et Marie Vasse, des chercheurs de l'équipe « Évolution expérimentale des communautés » de l'ISE-M, ont montré dans [14] comment l'évolution des bactériophages peut augmenter leur capacité d'infection par rapport aux génotypes ancestraux. Ils envisagent en effet l'utilisation de bactériophages artificiellement évolués ("éduqués") par une technique de transferts successifs [77], pour les rendre pré-adaptés à une souche donnée, particulièrement résistante. L'objectif de cet entraînement est double : d'une part augmenter le spectre de génotypes des bactéries pouvant être infectées, et

5. Source : http://www.inserm.fr/.
6. Ce n'est qu'après la phase III que l'on peut commercialiser un médicament en France.

d'autre part augmenter significativement l'impact du traitement sur les souches spécifiquement pathogènes ou résistantes aux antibiotiques. Cette étude lance une piste très intéressante sur la façon dont les phages pourraient être utilisés comme un moyen efficace de lutte contre les infections bactériennes ou pour la désinfection des surfaces, avec un risque réduit de résistance bactérienne par rapport aux méthodes conventionnelles. Une connaissance approfondie des mécanismes fondamentaux d'interaction et de coévolution phages-bactéries pourrait permettre de mettre au point des méthodes efficaces. Cette perspective de recherche me paraît ambitieuse et passionnante. En plus de mes collaborations actuelles, j'envisage dans ce cadre de travailler avec l'équipe « Évolution expérimentale des communautés » de mon laboratoire, avec laquelle nous avons déjà des contacts. Il serait nécessaire d'intégrer plusieurs travaux de différents domaines de recherche, à la fois appliqués et théoriques. La modélisation des interactions phages-bactéries et de leur coévolution sera une étape cruciale. Nos travaux sur la coévolution des gènes pourraient être utilisés dans ce contexte. Comprendre l'évolution des phages est un vrai challenge et les enjeux en termes de santé publique sont extrêmement motivants.

Conclusion

Écrire est un métier . . . qui s'apprend en écrivant.

Simone de Beauvoir

« La Force de l'âge » 1960.

Ce mémoire d'habilitation à diriger les recherches présente les travaux que j'ai réalisés depuis le début de ma carrière scientifique. Tous les éléments factuels sont regroupés dans le CV détaillé présenté en annexe A, page 41. Mes principaux thèmes de recherche : *alignement*, *génomique comparative* et *coévolution*, sont exposés mais non détaillés dans la première partie de ce mémoire (pages 3-28). Cette partie a pour but de mettre en évidence le lien entre ces thématiques en exhibant le fil conducteur, la *reconstruction d'histoires évolutives*. Mes différents travaux de recherche ont abordé cette reconstruction sous diverses facettes :

- avec des éléments biologiques différents (séquences d'ADN, ensembles de marqueurs génomiques ou génomes complets) ;
- modélisés par des objets variés (textes, permutations, ensembles, arbres, graphes) ;
- sous plusieurs modèles évolutifs (classique, avec amplifications et contractions, inversions, DCJ, recombinaison modulaire) ;
- et pour de multiples finalités (alignement, scénario évolutif, reconstruction de génomes ancestraux, aide à l'assemblage, aide à l'annotation).

J'ai ensuite présenté l'articulation de ces thématiques avec mon projet de recherche qui s'inscrit dans les thèmes de finition de génomes et de compréhension de leur évolution (partie 2, p. 29-38). Il s'agit d'une part, *via* l'aide à l'assemblage et à l'annotation, d'obtenir des données biologiques de haute qualité et d'autre part, de les utiliser pour reconstruire et comprendre des histoires évolutives. Ce projet s'appuie sur les trois derniers articles que nous avons publiés dans des revues internationales avec comité de lecture (annexes B, C et D, à partir de la page 57). Ces trois articles exposent de manière plus détaillée mes travaux concernant la reconstruction d'histoires évolutives de plusieurs génomes complets, sous l'angle de la coévolution pour les deux premiers ou sous l'angle de l'alignement de séquences pour le troisième.

Mes travaux de recherche actuels et la plupart des précédents ont toujours chercher à retracer l'histoire évolutive d'objets biologiques en minimisant le coût d'une telle histoire. Souvent nous

avons créé le modèle évolutif (séquences répétées ou adjacences). Souvent nous avons compté, énuméré, joué avec la combinatoire des objets créés. Souvent nous nous sommes baladés dans des graphes. Souvent nous avons produit des algorithmes exacts. Je n'imagine pas changer radicalement de direction dans le futur. La poursuite de mes deux thématiques de recherche actuelles avec l'idée d'intégrer les données des couches les plus basses aux plus hautes me semble pertinente. Plusieurs problèmes algorithmiques se posent, et y répondre en temps polynomial sera le défi.

CV détaillé

Sèverine Bérard | Maîtresse de conférences en informatique

04 67 14 32 61 • Severine.Berard@umontpellier.fr
www.pages-perso-severine-berard.univ-montp2.fr

Identité

Nom: Bérard
Âge: 39 ans
Prénom: Sèverine
Nationalité: Française
Adresse professionnelle: ISE-M – Institut des Sciences de l'Évolution, UMR 5554 - Université de Montpellier, Place Eugène Bataillon - C.C. 064, 34095 Montpellier cedex 05 - France

Formation et expériences professionnelles

Formation

Thèse de doctorat en Informatique **2003**
Mention Très honorable *Université Montpellier 2*

DEA d'informatique **2000**
Mention Très bien, rang 1/24 *Université Montpellier 2*

Maîtrise d'informatique **1999**
Mention Bien *Université Montpellier 2*

Licence d'informatique **1998**
Mention Bien *Université Montpellier 2*

DEUG MIAS : Mathématiques, Informatique, Application aux Sciences **1997**
Mention Passable *Université Montpellier 2*

Baccalauréat, Série C **1994**
Mention Bien *Académie de Montpellier*

Thèse de doctorat en informatique

Titre: *Comparaison de séquences répétées en tandem et application à la génétique*

Date et lieu de soutenance: 5 décembre 2003 au LIRMM (Montpellier)

Jury:

Marie-France Sagot	Directrice de Recherche, Inria Rhône-Alpes	*Rapporteur*
Alain Denise	Professeur, LRI, Université Paris-Sud	*Rapporteur*
Gilles Vergnaud	Institut de Génétique et Microbiologie, U. Paris-Sud	*Rapporteur*
Michel Habib	Professeur, Université Montpellier 2	*Président*
Olivier Gascuel	Directeur de Recherche, CNRS, LIRMM	*Directeur de thèse*
Éric Rivals	Chargé de Recherche, CNRS, LIRMM	*Co-directeur*
Maxime Crochemore	Professeur, Université de Marne-la-Vallée	*Invité*

Qualifications universitaires

2004 - 2008 : 27^{e} section (informatique) et 67^{e} section (biologie des populations et écologie)

Position actuelle

Maîtresse de Conférences — **Montpellier**
Université de Montpellier, faculté des sciences, département informatique — *Depuis sept. 2013*
Laboratoire de recherche ISE-M, Institut des Sciences de l'Évolution de Montpellier

Expériences professionnelles

Maîtresse de Conférences – 7 ans — **Montpellier**
Université Montpellier 2, faculté des sciences, département informatique — *2006 - 2013*
UMR AMAP, botAnique et bioinforMatique de l'Architecture des Plantes

Chargée de recherche INRA – 18 mois — **Toulouse**
Département de Mathématiques et Informatique Appliquées (MIA) à l'INRA de Toulouse — *2005 - 2006*
Unité BIA, Biométrie et Intelligence Artificielle

Post-doctorat CNRS – 6 mois — **Marseille**
Institut de Mathématiques de Luminy (IML), équipe d'Alain Guénoche — *2004 - 2005*
Titre : Méthodes de prédiction fonctionnelle dans le graphe d'interactions protéines-protéines

ATER en Informatique – 11 mois — **Montpellier**
Université Montpellier 2 — *2003 - 2004*
LIRMM, Laboratoire d'Informatique de Robotique et de Micro-électronique de Montpellier

Stage post-doctoral – 6 semaines — **Montréal - Canada**
UQÀM, LaCIM, équipe d'Anne Bergeron — *2004*
Titre : Réarrangements génomiques

Allocataire-monitrice – 36 mois — **Montpellier**
Université Montpellier 2, Thèse en Informatique — *2000 - 2003*
LIRMM, Laboratoire d'Informatique de Robotique et de Micro-électronique de Montpellier

Congé pour Recherches ou Conversions Thématiques

1 semestre de CRCT — **Université Montpellier 2**
Demandé et obtenu au niveau local — *août 2011 - janvier 2012*
Ces six mois de décharge d'enseignement m'ont permis d'encadrer le stage de Coralie Gallien [E6] jusqu'à son terme, fin septembre 2011, de travailler sur la publication issue de ce stage [P3], ainsi que de programmer le logiciel associé DeCo [L3].

Implications dans les conseils et autres instances administratives

Élue du département scientifique B3STE de l'université de Montpellier : depuis mars 2016. Les départements scientifiques sont des composantes de l'université chargées de la coordination de la recherche dans un domaine thématique donné et du renforcement du lien recherche-formation.

Élue à la commission recherche du conseil académique de l'Université de Montpellier : depuis janvier 2015. La commission recherche est le lieu de discussion de la politique scientifique de l'établissement. Nous sommes amenés à proposer et évaluer des appels à projets pour des financements "internes" (post-doctorats, manifestations scientifiques, projet de recherche, équipements, ...). La commission gère aussi les carrières des enseignants-chercheurs (CRCT, primes, décharges, ...). Fréquence des réunions : 2-4 demi-journées par mois.

Nommée à la commission formation de l'UM2 : 2014 (et de l'UM depuis avril 2016). La commission formation gère le plan de formation, les règles de prise en charge financière, la charte de formation et examine les demandes de formations individuelles, VAE, bilans de compétences et congés de formation professionnelle. 3-4 réunions/an.

Élue au conseil de service de l'unité BIA : juin 2005 - sept. 2006. Ce conseil se réunit mensuellement et a pour vocation de régler les affaires courantes de l'unité.

Nommée à la commission informatique de l'unité BIA : sept. 2005 - sept. 2006. Cette commission se réunit moins régulièrement et sert d'interface entre les utilisateurs et le service informatique.

Nommée à la commission bibliothèque du LIRMM : 2001 - 2003. Représentante des doctorants à la commission bibliothèque du LIRMM : participation à l'élaboration de la politique scientifique et à la mise en place d'une base de donnée pour la gestion du fond documentaire. Fréquence des réunions : 1 fois/mois en moyenne.

Activités pédagogiques

2000-2006 Enseignements vacataires ∼ **350 h**

- Vacataire à l'Université Montpellier 2 (UM2), 2000 (lorsque j'étais étudiante en DEA)
- Monitrice à l'UM2, 2000 – 2003
- ATER à l'UM2, 2003 – 2004
- Vacataire à l'Université de Toulouse III et dans les écoles d'ingénieurs INSA-T et ENSA-M, 2005 – 2006

Langages de programmation : Scheme, C++
DEUG 1^{re} *et* 2^{e} *année, IUP* 1^{re} *année* ∼ **180 *h***
Algorithmique, programmation, logique du 1^{er} ordre

Système & Réseaux
IUP 2^{e} *année* ∼ **150 *h***
Linux, compilation, fichiers, processus ; fonctionnement des réseaux, couches ISO, protocoles, routage

Bioinformatique
DESS BioInfo, DEA Informatique, Doctorants, INSA, ENSA-M ∼ **30 *h***
Introduction à la bioinformatique, comparaison de séquences

2006-2016 Enseignements UM2/UM ∼ **2000 h**

- 10 années universitaires
- 2 semestres de congés maternité, 1 semestre de CRCT, 1 année à temps partiel 50 %
- soit 16 semestres d'enseignement + heures complémentaires

Mathématiques discrètes
Licence 1, Cours/TD ∼ **390 *h***
Ensembles, relations binaires, fonctions applications, raisonnements, graphes

Encadrements stages, projets tutorés, TER, doctoriales et référente de groupe de L1
Licence 1, Licence 3, Master 1, Master 2, doctorants ∼ **390 *h***
Suivi d'étudiants sur un projet, un stage, une formation ou une année

Réseaux
Licence 3, Master 2, Cours/TD/TP ∼ **330 *h***
Fonctionnement des réseaux, couches ISO, protocoles, routage, programmation réseau, modèle Client-Serveur

Initiation à l'algorithmique, algorithmique du texte
Licence 1, Licence 3, Master 1, Cours/TD/TP ∼ **250 *h***
Conditionnelles, tableaux, boucles ; algorithmes classiques de recherche de motifs

Concepts de base en informatique et C2I
Licence 1, Cours/TP ∼ **190 *h***
Découverte de l'environnement numérique, système de fichiers, ligne de commande, bureautique, web

Bioinformatique
Master 2, Cours/TD/TP ∼ **175 *h***
Analyse de séquences, introduction à la bioinformatique et algorithmique pour la bioinformatique avancée

Logique
Licence 2, TD/TP ∼ **160 *h***
Logique des propositions, calcul formel, formes normales, méthode de résolution

Divers
Tous niveaux, TD/TP/référentiel ∼ **100 *h***
Du QT, des TP de graphes, des remplacements et 20h de référentiel pour la responsabilité du parcours Bioinformatique

Responsabilités dans les modules et parcours

Responsable du parcours *Bioinformatique*
Master pluridisciplinaire Intégration de Compétences (IC) *2008 - 2012*
Responsabilité en 2008-2009 puis co-responsable les 3 années suivantes avec Anne-Muriel Chifolleau. Gestion des candidatures, gestion des emplois du temps, coordination des enseignements et de l'équipe enseignante, encadrement de stages et de TER, prise en charge de modules "orphelins". Le master pluridisciplinaire IC est devenu *Sciences et numérique pour la santé* et le parcours *Bioinformatique* est devenu BCD : http://www.lirmm.fr/BCD/.

Responsable du module *Analyse de séquences*
Master 2, parcours Bioinformatique *2006 - 2011*
Mise en place de cette UE : création du contenu et des supports de cours, TD et TP.
http://www.lirmm.fr/∼berard/FMIN367/.

Co-responsable du module *Bioinformatique avancée*
Master 2, Informatique et Bioinformatique *depuis 2006*
Cette UE est constituée d'une suite de conférences de chercheurs en bioinformatique. Organisation de cette UE chaque année universitaire : définition du contenu, choix des intervenants, gestion de l'emploi du temps.

Co-responsable du module *Concepts de base en informatique*
Licence 1 *2007 - 2010*
Organisation de cette **UE à très gros effectif** durant 3 années. Refonte du contenu des TP et mise en place d'un site web dédié, mise en place de fichiers partagés pour la gestion des emplois du temps des enseignants et la collecte des notes de contrôle continu (en moyenne plus de 1600 étudiants inscrits et plus de 60 intervenants).

Responsable du module *Mathématiques discrètes*
Licence 1 *depuis 2010*
Enseignements dans cette UE de L1 (nommée ensuite *Fondements de l'informatique* puis *De la combinatoire aux graphes*) depuis 2006, responsable à partir de 2010. Prise en charge du cours et remise à jour du contenu pédagogique (∼ 250 étudiants).

Création et co-responsabilité des modules *Algorithmique du texte*
Licence 3 et Master 1 *depuis 2012*
Avec Annie Chateau nous avons créé et mis en place les deux UE d'algorithmique du texte au niveau L3 et M1. Ces deux UE complémentaires abordent des définitions de bases (mots, facteurs, bords, périodes, ...), des algorithmes classiques d'alignement de séquences, de recherche de motifs (exacts, approchés, multiples) mais aussi des structures de données comme la table et l'arbre de suffixes.

Création de supports pédagogiques

- **Mathématiques discrètes** Cours/TD/Examens ; module de 50h.
- **Algorithmique du texte** Cours/TD/TP/Examens, *ex nihilo* ; module de 50h.
- **Analyse de séquences** Cours/TD/TP/Examens, *ex nihilo* ; module de 25h.
- **Bioinformatique** Une dizaine de cours thématiques (∼ 2*h*) *ex nihilo* et des examens.
- **Concepts de base en informatique** TP et contrôles continus ; module de 25h.
- **Réseaux M2 IC** Cours/TP/Exam en Java *ex nihilo* ; module de 25h.
- **Maquette LMD3-LMD4** J'ai également contribué à l'élaboration de plusieurs maquettes, notamment pour le parcours Bioinformatique du master pluridisciplinaire (*STIC-Santé* puis *Sciences et Numérique pour la Santé*).

Activités liées à la recherche

Dans tout ce qui suit, le code couleur des références correspond aux thématiques de recherche détaillées p. 12 : alignement, génomique comparative et coévolution.

Logiciels

- **[L1] ALPHA** (ALignment of PHAges) 2015

Ce logiciel permet d'aligner des génomes de bactériophages. Il peut les comparer à différentes échelles, mettre en évidence les insertions et les recombinaisons, ainsi que les variants de la plupart des modules. Programmé en Python ; contribution à l'élaboration de l'algorithme et à quelques lignes de code ; ALPHA a été programmé à 99 % par Krister Swenson. http://www.lirmm.fr/∼ swenson/alpha/alpha.htm.

- **[L2] ARt-DeCo** (Assembly Recovery through DeCo) 2014

Ce logiciel basé sur DeCo permet, en plus de reconstruire un ordre de gènes pour des génomes ancestraux, de proposer de nouvelles adjacences entre gènes de génomes actuels et de compléter ainsi les assemblages actuels. Programmé en C++/Bio++ ; contribution au code : 20 %. Disponible sur demande.

- **[L3] DeCo** (Detection of Co-evolution) 2011

DeCo est un logiciel permettant de reconstruire l'évolution des relations entre gènes. Programmé en C++/Bio++ ; contribution au code : 100 %. Disponible sur http://pbil.univ-lyon1.fr/software/DeCo/.

- **[L4] SimCT** (Similarity Clustering Tree) 2005

SimCT est un logiciel générique de visualisation de relations entre entités biologiques basées sur les ontologies. Contribution : mise au point de l'algorithme ; SimCT a été codé par Laurent Tichit et Carl Herrmann. Disponible sur http://tagc.univ-mrs.fr/SimCT.

- **[L5] MS_Align** (Mini-Satellite Alignment) 2001

MS_Align est un logiciel d'alignement de séquences répétées en tandem. Programmé en C++ ; contribution au code : 100 %. Disponible sur http://www.atgc-montpellier.fr/ms_align/.

Relectures d'articles

Pour des journaux: Bioinformatics (2005) ; Transactions on Computational Biology and Bioinformatics (2006, 2008, 2012) ; Theoretical Computer Science (2008) ; Discrete Mathematics, Algorithms and Applications (2010) ; Information Processing Letter (2010) ; Journal of Computationnal Biology (2011) ; South African Computer Journal (2014).

Pour des conférences: MICAI'05 ; JOBIM'09 (comité de programme) ; RECOMB-CG'10 (comité de programme) ; WABI'14.

Pour un livre: Mathematics of Evolution and Phylogeny. Éditeur O. Gascuel. Oxford University Press, 2005.

Jury de thèse

Joseph Lucas — **Université Pierre et Marie Curie**

Étude de l'évolution de l'ordre des gènes de vertébrés par simulation — *2016*

Sous la direction de Hugues Roest Crollius, Directeur de recherche (Institut de Biologie de l'École Normale Supérieure de Paris). Rapporteurs : David Sankoff et Nicolas Lartillot. Membres du jury : Alessandra Carbone, Sèverine Bérard et Claire Lemaitre. Thèse soutenue le 17 mai 2016.

Expertise

Referee for a Full Proposal application to the Marsden Fund — **New-Zealand**

Panel: Mathematical and Information Sciences — *2005*

The Marsden Fund has been set up by the New Zealand Government to fund excellent fundamental research in a wide range of fields in the sciences, engineering, the social sciences and the humanities. The Fund is administered by the Royal Society of New Zealand.

Organisation de conférences

- Ces conférences ont regroupé de 30 à un peu plus de 100 personnes.

GTGC'16 : à Lyon, le 1^{er} juillet 2016 (journée satellite de JOBIM'16). https://gtgc2016.sciencesconf.org/

GTGC'12 : à Lille, les 13 et 14 décembre 2012 (journée satellite de JOBIM'12)

GTGC'10 : à Montpellier, « Annotation des génomes et Génomique Comparée » (satellite de JOBIM'10)

GTGC'09 : à Nantes, le 8 juin 2009 (satellite de JOBIM'09 et des Journées Scientifiques de Nantes 2009)

ALPHY-GTGC'09 : à Montpellier, les 26 et 27 janvier 2009

GTGC'08 : à Lille, le 3 juillet (satellite de JOBIM'08)

GTGC'07 : à Marseille, le 9 juillet (satellite de JOBIM'07)

GTGC'06 : à Nantes, les 12 et 13 octobre 2006

ALPHY-GTGC'06 : à Lyon, les 23 et 24 janvier 2006

SeqBio'03 : à Montpellier, « Algorithmes et Séquences» et « Indexation de texte et découverte de motifs ».

Animation scientifique

Création et animation du Groupe de Travail francophone en Génomique Comparative (GTGC)

Audience nationale *2005 - 2016*

Organisation régulière de rencontres thématiques (en moyenne une fois par an) : http://lbbe.univ-lyon1.fr/projets/GTGC/ J'ai initié la mise en place de ce groupe de travail et d'une liste de diffusion associée. Éric Tannier, puis dans un 2^e temps Jean-Stéphane Varré, Guillaume Fertin et Hugues Roest-Crollius se sont joints au comité d'organisation. Le but de ce groupe est de faire se rencontrer les chercheurs francophones (bioinformaticiens, biologistes, informaticiens, mathématiciens, ...) travaillant autour des thèmes de la génomique comparative. Ce projet est né au mois de juin 2005 et a bien intéressé la communauté. La liste de diffusion a rapidement recueilli une petite centaine de personnes provenant de différents laboratoires et de l'étranger (Canada, États-Unis et Singapour). Nous avons depuis 10 ans organisé différentes manifestations et réunions réunissant en moyenne une cinquantaine de personnes. La banalisation des données de génomes complets et l'utilisation croissante de la génomique comparative dans les études scientifiques ces dernières années ont réduit l'intérêt d'un groupe de travail spécifique à ce sujet. Nous envisagions donc de clôturer cette aventure scientifique par une dernière réunion, qui se tiendra en juillet prochain. Mais l'intérêt que semble susciter cette journée et l'entrée de nouveaux collègues dans le comité d'organisation changeront peut-être cette décision.

Animation du champs transversal MIA

Laboratoire AMAP *2006-2013*

AMAP est une unité mixte CIRAD/INRA/IRD et université, composée de chercheurs en botanique et de modélisateurs. J'ai animé en tant que co-responsable, le champs thématique transversal Mathématiques et Informatique Appliquées. J'ai organisé principalement des réunions, des séminaires ainsi que des cours à destination d'autres chercheurs. J'ai mis en place une liste de diffusion ainsi qu'un site web permettant à la communauté math-info du laboratoire de partager et d'échanger. Cette animation m'a pris en moyenne 1 à 2 jours par mois.

Organisation du séminaire des doctorants du LIRMM

Laboratoire LIRMM *2004*

Mise en place et gestion du séminaire qui se déroulait une fois par semaine, création et maintenance d'une page web dédiée. Temps consacré : 1h/semaine en moyenne de janvier à juin 2004

Financements obtenus

ANCESTROME

Appel à projet « Bioinformatique » du programme « Investissements d'avenir » de l'ANR *2012-2017*

ANCESTROME est piloté par le LBBE à Lyon, et implique l'ISE-M à Montpellier, le laboratoire DYOGEN à l'ENS Paris, le LCB à Marseille et le laboratoire BMGE de l'institut Pasteur. http://ancestrome.univ-lyon1.fr/
Financement du salaire de thèse de Yoann Anselmetti [E1] et de ses frais de fonctionnement sur la période.

Chercheuse invitée

LabEx NUMEV *2013-2014*

2 AAP en collaboration avec Annie Chateau et Krister Swenson pour faire venir Anne Bergeron en 2013 et 2014, pour un montant de 2*6000€.

Animation de la recherche

Réseaux méthodologiques de MIA (INRA) *2006*

2000 € m'ont été alloués pour financer les premières activités du groupe de travail GTGC.

Communications scientifiques

Articles dans des revues internationales avec comité de lecture

- [P1] **2016** Sèverine Bérard, Annie Chateau, Nicolas Pompidor, Paul Guertin, Anne Bergeron and Krister M. Swenson. **Aligning the unalignable: bacteriophage whole genome alignments**. BMC Bioinformatics. 17:30. doi:10.1186/s12859-015-0869-5
- [P2] **2015** Yoann Anselmetti, Vincent Berry, Cedric Chauve, Annie Chateau, Éric Tannier et Sèverine Bérard. **Ancestral gene synteny reconstruction improves extant species scaffolding**. BMC Genomics 16(Suppl 10):S11 doi:10.1186/1471-2164-16-S10-S11
- [P3] **2012** Sèverine Bérard, Coralie Gallien, Bastien Boussau, Gergely J. Szöllősi, Vincent Daubin and Eric Tannier. **Evolution of gene neighborhoods within reconciled phylogenies**. Bioinformatics (2012) 28 (18): i382-i388. doi: 10.1093/bioinformatics/bts374
- [P4] **2010** Cédric Gaucherel, Sèverine Bérard et François Munoz. **Equation or algorithm: differences and choosing between them**, Acta Biotheoretica. 59(1): 67-79
- [P5] **2009** Sèverine Bérard, Annie Chateau, Cedric Chauve, Christophe Paul and Éric Tannier. **Computation of Perfect DCJ Rearrangement Scenarios with Linear and Circular Chromosomes**, Journal of Computational biology, volume 16, number 10, pages 1287-1309, Mary-Ann Liebert Inc. publishers
- [P6] **2009** Carl Herrmann, Sèverine Bérard, Laurent Tichit **SimCT: a generic tool to visualize ontology-based relationships for biological objects**, Bioinformatics 25(23): 3197-3198
- [P7] **2008** Sèverine Bérard, Cedric Chauve and Christophe Paul. **A more efficient algorithm for perfect sorting by reversals**, Information Processing Letters, volume 106, issue 3, 30 April 2008, pages 90-95
- [P8] **2007** Sèverine Bérard, Anne Bergeron, Cedric Chauve and Christophe Paul. **Perfect sorting by reversal is not always difficult**, IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB), vol. 4, nunber 1, pages 4-16
- [P9] **2006** Sèverine Bérard, François Nicolas, Jérôme Buard, Olivier Gascuel et Éric Rivals. **A fast and specific alignment method for minisatellite maps**, Evolutionary Bioinformatics Online (EBO) 2006:2 327-344
- [P10] **2003** Sèverine Bérard, Éric Rivals. **Comparison of Minisatellites**, Journal of Computational biology, 10(3-4), pages 357-72, Mary-Ann Liebert Inc. publishers

Articles dans des conférences internationales avec comité de lecture

- [P11] **2015** Yoann Anselmetti, Vincent Berry, Cedric Chauve, Annie Chateau, Éric Tannier et Sèverine Bérard. **Ancestral gene synteny reconstruction improves extant species scaffolding**, RECOMB-CG, 4-7 October 2015, Frankfurt, Germany. Publié dans [P2]
- [P12] **2012** Sèverine Bérard, Coralie Gallien, Bastien Boussau, Gergely J. Szöllősi, Vincent Daubin and Eric Tannier. **Evolution of gene neighborhoods within reconciled phylogenies**, 11th European Conference on Computational Biology (ECCB'12), 9-12 September 2012, Basel, Switzerland. Publié dans [P3]
- [P13] **2008** Sèverine Bérard, Annie Chateau, Cedric Chauve, Christophe Paul and Éric Tannier. **Perfect DCJ rearrangement**, 6th RECOMB Comparative Genomics Satellite Workshop, p158-169, 13-15 October 2008, Paris, France
- [P14] **2008** Cédric Gaucherel, Sèverine Bérard et François Munoz. **Equation against Algorithm: Which differences and which one to choose?**, European conference on Computing And Philosophy E-CAP 2008, 16-18 june 2008, Montpellier, France
- [P15] **2005** Sèverine Bérard, Anne Bergeron, Cedric Chauve and Christophe Paul. **Perfect sorting by reversal is not always difficult (extended abstract)**, 5th Workshop on Algorithms in Bioinformatics (WABI'05), 3-6 October 2005, Palma de Majorque
- [P16] **2004** Sèverine Bérard, Anne Bergeron and Cedric Chauve. **Conservation of Combinatorial Structures in Evolution Scenarios**, 2nd RECOMB Comparative Genomics Satellite Workshop, 16-19 October 2004, University of Bologna, Italy
- [P17] **2004** Éric Rivals, Sèverine Bérard. **Alignement de séquences avec opérations non-commutatives**, Quatrièmes journées francophones de recherche opérationnelle (FRANCORO IV), pages 16, Fribourg, Suisse

- [P18] **2002** Sèverine Bérard, Éric Rivals. **Comparison of Minisatellites**, Sixth Annual International Conference on Computational Biology (RECOMB), pages 67-76, Washington, DC, USA, ACM press

Chapitre d'ouvrage

- [P19] **2016** Yoann Anselmetti, Nina Luhmann, Sèverine Bérard, Eric Tannier, Cedric Chauve. **Comparative Methods for Reconstructing Ancient Genomes Organization**, dans J. Setubal et J. Stoye (eds). Methods in Molecular Biology, Comparative Genomics, Springer series, *soumis le 08/04/16, accepté.*

Articles dans des conférences nationales avec comité de lecture

- [P20] **2016** Yoann Anselmetti, Vincent Berry, Cedric Chauve, Annie Chateau, Éric Tannier et Sèverine Bérard. **Comment la reconstruction de génomes ancestraux peut aider à l'assemblage de génomes actuels**, Journées Ouvertes Biologie Informatique Mathématiques (JOBIM'16) à Lyon, *soumis le 15/04/16, accepté.*
- [P21] **2005** Sèverine Bérard, Laurent Tichit and Carl Herrmann. **ClusterInspector, a tool to visualize ontology based relationships**, Actes des Journées Ouvertes Biologie Informatique Mathématiques (JOBIM'05), pages 447-457, Lyon
- [P22] **2003** Sèverine Bérard, Éric Rivals. **Comparaison de séquences avec amplifications et contractions**, 5e congrès de la société Française de Recherche Opérationnelle et d'Aide à la Décision (ROADEF), pages 169-170, Avignon
- [P23] **2002** Sèverine Bérard, Éric Rivals. **Comparaison de minisatellites**, Actes de la 24e réunion annuelle du Groupe de Génétique et Biologie des Populations (PPD), page 128, Montpellier
- [P24] **2002** Sèverine Bérard, Éric Rivals. **Comparaison de minisatellites**, Actes des Journées Ouvertes Biologie Informatique Mathématiques (JOBIM'02), pages 261-262, Saint-Malo

Académique

- [P25] **2003** Sèverine Bérard. **Thèse de doctorat : Comparaison de séquences répétées en tandem et application à la génétique**, Université Montpellier 2, LIRMM
- [P26] **2000** Sèverine Bérard. **Rapport de DEA : Reconstruction d'histoire de répétitions en tandem**, Tuteur de stage : Éric Rivals, Université Montpellier 2, LIRMM

Communication orale dans un colloque avec comité de sélection

- [O1] **2015** Sèverine Bérard. « Aligning the unalignable: bacteriophage whole genome alignments », Colloque *Interactions Phages-Bactéries : du fondamental à l'appliqué*, 19-20 novembre 2015, Montpellier. Présentation de la publication [P1]

Communications orales sur invitations

- [O2] **2015** « Algorithmique pour l'évolution des interactions géniques : `DeCo` et `ARt-DeCo` ». Séminaire Mathématiques, Évolution, Biologie (MEB). Marseille, 25 novembre 2015
- [O3] **2015** « Bioinformatique et évolution ». Conférence de culture scientifique dans le cadre d'un stage MathC2+ organisé par l'IREM et l'académie de Montpellier
- [O4] **2015** « Histoire de la bioinformatique ». Journée Femmes et Sciences // Donner des Elles à l'UM, 26 mars 2015
- [O5] **2005** « Réarrangements génomiques : le problème Sorting By Reversals ». École de jeunes chercheurs en algorithmique et calcul formel. Cours de 2h. Responsable : Valérie Berthé. Effectif : ~ 50. Montpellier, du 4 au 8 Avril 2005. (École thématique CNRS)
- [O6] **2004** « Comparaison de séquences répétées en tandem et application à un minisatellite humain grâce au logiciel MS_Align ». Journées Bioinformatiques du CINES (Centre Informatique National de l'Enseignement Supérieur). Montpellier, 19-21 octobre 2004

Posters

- [Po1] **2015** Aligning the unalignable : bacteriophage multiple alignments using partial order. Sèverine Bérard, Annie Chateau, Nicolas Pompidor, Paul Guertin, Anne Bergeron et Krister Swenson. 4^e journée scientifique du LabEx NUMEV, 2 juillet 2015 à SupAgro Montpellier
- [Po2] **2015** Ancestral gene synteny reconstruction improves extant species scaffolding. Yoann Anselmetti, Vincent Berry, Cedric Chauve, Annie Chateau, Éric Tannier et Sèverine Bérard. Mathematical and Computational Evolutionary Biology (MCEB), 21-25 juin 2015
- [Po3] **2013** Algorithmique pour l'évolution des interactions géniques. Pierre-Antoine Jean, Vincent Berry, Sèverine Bérard, Annie Chateau et Éric Tannier. JOBIM'13, du 1er au 4 juillet à Toulouse

Encadrements

Étudiant en thèse

Yoann Anselmetti [E1]

École doctorale E2M2 (Lyon), doctorat effectué à l'ISE-M *2014 - 2017*

Assemblage et évolution de la structure de génomes anciens et actuels. Encadrants : Sèverine Bérard (50 %), Éric Tannier (40 %) et Annie Chateau (10 %). A donné lieu aux publications [P2], [Po2], [P19] et [P20].

Comité de suivi de thèse

Alix Armero Villanueva

École doctorale SIBAGHE (Montpellier), doctorat effectué à l'UMR AGAP (Agropolis/CIRAD) *2014 - 2017*

Travail de thèse sur la génomique comparative du cocotier
Sous la direction de Dominique This et Stéphanie Sibidé-Bocs

Magali Semeiria

École doctorale E2M2 (Lyon), doctorat effectué au LBBE *2012 - 2015*

Travail de thèse sur le développement d'un modèle probabiliste pour l'évolution des relations entre gènes
Sous la direction de Éric Tannier et Laurent Guéguen

Paul-Camilo Zalamea

École doctorale SIBAGHE (Montpellier), doctorat effectué à AMAP *2006 - 2009*

Cecropia growth pattern periodicity : could be a Neotropical genus be a good biological clock to estimate the age of disturbed areas ?
Sous la direction de Daniel Barthélémy et Patrick Heuret

Stages M2 recherche

Frédéric Bigey [E2]

M2 BCD / Formation continue, stage bibliographique + stage de 3 mois (avril-juin) *2015 - 2016*

Intégration de méthodes de reconstruction d'adjacences de gènes pour l'assemblage de génomes de levures. Co-encadré avec Annie Chateau. Position actuelle : chercheur INRA (en formation continue)

Amal Zine El Aabidine [E3]

M2 BCD, stage effectué au LIRMM *2013 - 2014*

Reconstruction de l'histoire évolutive des recombinaisons modulaires chez les bactériophages. Co-encadrée avec Annie Chateau et Krister Swenson. Position actuelle : ingénieur de recherche CDD CNRS

Yoann Anselmetti [E4]

M2 BCD, stage effectué à l'ISE-M. Étudiant que je co-encadre actuellement en thèse. *2013 - 2014*

Reconstruction de l'histoire évolutive des génomes : réconciliation et reconstruction des relations ancestrales. Co-encadré avec Vincent Berry et Annie Chateau. Position actuelle : doctorat oct. 2014 $\longrightarrow$ 2017

Pierre-Antoine Jean [E5]

M2 BCD, stage effectué au LIRMM. A donné lieu à la publication [Po3] *2012 - 2013*

Algorithmique pour l'évolution des interactions géniques. Co-encadré avec Vincent Berry, Annie Chateau et Éric Tannier. Position actuelle : doctorat oct. 2014 $\longrightarrow$ 2017

Coralie Gallien [E6]

M2 Informatique, stage effectué à AMAP. A donné lieu à la publication [P3] *2011 - 2012*

Détection de coévolution de gènes. Co-encadrée avec Éric Tannier. Position actuelle : développeur Backend, La Valériane, entreprise de e-santé à Montpellier

Benjamin Severac **[E7]**
M2 Informatique, stage effectué au LIRMM *2007 - 2008*
Comparaison de génomes et distance de transformation. Co-encadré avec Éric Rivals. Position actuelle : créateur de Kreaseed SARL, développeur

Anne Dievart **[E8]**
M2 IC / Formation continue, stage effectué au CIRAD *2007 - 2008*
Évolution des récepteurs riches en leucines chez les plantes supérieures. Co-encadrée avec Christophe Perrin. Position actuelle : chercheur CIRAD

Marc Bussière **[E9]**
M2 ENSEEIHT, stage effectué dans l'unité BIA *2004 - 2005*
Améliorations et extensions du logiciel d'alignement de cartes de séquences répétées en tandem MS_Align. Position actuelle : ingénieur d'études et développement chez Akka Technologies (logiciels informatiques)

Autres encadrements en lien avec la recherche

Lisa de Mattéo **[E10]**
M1 Informatique, Stage dété (2 mois) *2015 - 2016*
Algorithmique de graphe pour la détection de conflits entre prédictions et cartes génétiques.

Julien Chopelet et Thierry Bonnabaud la Bruyère **[E11]**
M1 BCD, Travaux Encadrés de Recherche (UM) *2015 - 2016*
Transfert d'annotation dans les génomes de bactériophages. Co-encadré avec Anne Bergeron, Annie Chateau et Krister Swenson

Ivan Brunet-Manquat, Mattéo Coquilhat, Faustine Durand, Mickael Hamouma **[E12]**
L3 Informatique, Travaux Encadrés de Recherche (UM) *2015 - 2016*
Interface graphique pour le logiciel ALPHA [L1]. Co-encadré avec Anne Bergeron, Annie Chateau et Krister Swenson

Nicolas Pompidor **[E13]**
L2 CMI Informatique, stage CMI 6 semaines (LIRMM). A donné lieu à la publication [P1] *2014 - 2015*
Alignement des génomes de bactériophages. Co-encadré avec Anne Bergeron, Annie Chateau et Krister Swenson

Roch Quilichini-Dardie, Helima Yakoubi, Tue Minh Duc Ngo, Hang Ji **[E14]**
M1 IPS, Travaux Encadrés de Recherche (UM) *2013 - 2014*
Restructuration d'un programme de reconstruction phylogénétique DeBlock. Co-encadrés avec Annie Chateau

Anthime Cottin, Rémy Peru, Ridjali Ridjali **[E15]**
M1 Informatique, Travaux Encadrés de Recherche (AMAP) *2013 - 2014*
Diagnostic architectural nomade : développement informatique d'un outil d'identification de l'état des arbres à partir de la méthode ARCHI. Co-encadrés avec Sylvie Sabatier, Samuel Dufour-Kowalski, Pierre Bonnet, Yves Caraglio, Christophe Drénou

Luc-André Abric, Olivier Sallenave, Anne Tireau **[E16]**
M1 Informatique, Travaux Encadrés de Recherche (AMAP) *2007 - 2008*
Outils de manipulation et de configuration de représentation générique de plante : GreenLab User Interface. Co-encadrés avec Jean-Francois Barczi

Sébastien Harispe, Pol Kennel, Marilyne Summo, Zahra Vafa **[E17]**
M1 IC Bioinformatique, projet tutoré (UM2) *2007 - 2008*
Application to Blot Synteny. Développement d'un visualisateur de synténies. Co-encadrés avec Annie Chateau, Isabelle Mougenot et Vincent Ranwez

Julien Wollbrett **[E18]**
M1 IPS Bioinformatique, stage de 3 mois (UM2) *2006 - 2007*
Codage d'un algorithme pour le calcul de la distance d'inversion

Nicolas Canicatti, Jean-Philippe Villemin, Julien Wollbrett **[E19]**
M1 IPS Bioinformatique, projet tutoré (UM2) *2006 - 2007*
Création d'une base de données de comparaison de génomes entièrement séquencés

Groupe de 12 étudiants **[E20]**
L3 Informatique, stage (UM2) *2006 - 2007*
Interface graphique pour la manipulation de permutations signées

Paul Delcassé **[E21]**
IUP Systèmes Intelligents M1, U. Toulouse III, stage de fin d'année (BIA) *2005 - 2006*
Interface web de traduction de séquences répétées en tandem en cartes. Projet DNA2Map

Laura Guillon **[E22]**
IUP Systèmes Intelligents M1, U. Toulouse III, stage de fin d'année (BIA) *2005 - 2006*
Prédiction d'ARN non codant dans un contexte d'analyse comparative. Co-encadrée avec Christine Gaspin

Alexandre Baranov, Marion Groschene, Marie-Pierre Labeyrie, Patrice Oms **[E23]**
IUP GMI 2^{e} année, stage d'initiation à la recherche (UM2) *2003 - 2004*
Alignement de séquences répétées en tandem avec coûts de mutation variables

Déroulement de la carrière

Dans cette dernière partie de mon CV détaillé, je reprends les données factuelles qui précèdent pour les mettre en perspective. Considérant le point 0 de mes activités d'enseignement et de recherche à l'année 2000, date de mon stage de DEA, de mes premiers enseignements à l'université et de mon entrée en doctorat. Cela fait, à ce jour de rédaction, avril 2016, environ 16 années de carrière. La figure 1 page 14 reprend visuellement ces différents éléments.

Dans l'espace et dans le temps

J'ai commencé ma carrière dans l'année 2000. Année lors de laquelle j'ai effectué mon stage de DEA sous la direction d'Éric Rivals au LIRMM à Montpellier. C'est aussi cette année que j'ai commencé ma thèse, toujours au LIRMM et encadrée par Éric Rivals. J'ai soutenue en décembre 2003 alors que j'étais "demi-ATER". Étant libre d'enseignements à la fin du printemps 2004, j'ai pu effectuer un séjour post-doctoral de 6 semaines au LaCIM à l'Université du Québec À Montréal (UQÀM). J'avais choisi d'aller travailler avec Anne Bergeron dont les thématiques de recherche en génomique comparative m'intéressaient particulièrement. J'y ai rencontré Cédric Chauve. Ce fut un tournant dans ma carrière scientifique et le début de belles collaborations. En 2004-2005, cela a été la période de recherche de postes. J'ai d'abord obtenu un post-doctorat CNRS sous la direction d'Alain Guénoche à l'Institut de Mathématiques de Luminy (IML) à Marseille qui a été écourté suite à l'obtention d'un poste de chargée de recherche dans le département Mathématiques et Informatique Appliquées (MIA) de l'INRA à Toulouse. J'ai commencé par finaliser mes travaux précédents : un dernier papier issu de ma thèse [P9], celui de mon post-doc [C21] et celui de ma collaboration avec Anne et Cédric [C16]. J'ai aussi commencé une nouvelle thématique de recherche sur l'ARN. Cependant, je voulais pour des raisons personnelles revenir sur Montpellier. Apprenant qu'il ne serait pas possible d'obtenir une mutation au sein de l'INRA, j'ai choisi de candidater sur des postes de maître de conférences à Montpellier. J'ai ainsi obtenu un poste à l'UMR de botAnique et bioinforMatique de l'Architecture des Plantes (AMAP). J'y suis arrivée dans un contexte particulier puisque l'équipe d'informaticiens qui avait demandé ce poste était en partance mais le poste, lui, restait attaché à AMAP. J'ai donc essayé de m'intégrer aux thématiques du laboratoire mais sans réel succès. C'est en 2011, avec l'encadrement du stage de master 2 de Coralie Gallien [E6] sur des thématiques plus proches de mes travaux précédents que ma carrière scientifique redémarre. Cherchant à encadrer des étudiants en thèse, je me suis sentie limitée par mon appartenance à AMAP dont les thématiques de recherche ne correspondaient pas aux miennes. J'ai donc demandée mon intégration à l'Institut des Sciences de l'Évolution de Montpellier (ISE-M) au printemps 2014, où je poursuis mes recherches depuis le 1er septembre 2014.
Bien qu'étant restée géographiquement proche de Montpellier, j'ai appartenu à 6 laboratoires différents en 16 années : LIRMM (Montpellier), LaCIM (Montréal, Canada), BIA INRA (Toulouse), IML (Marseille), AMAP (Montpellier) et ISE-M (Montpellier).

Thématiquement en recherche

J'ai commencé ma carrière en m'intéressant à l'*alignement de séquences génétiques*, en l'occurrence des séquences d'ADN répétées en tandem. Très vite je me suis passionnée pour la *génomique comparative* et cela a correspondu à la période où l'on commençait à séquencer des génomes entiers. Lors de mon post-doctorat et de mon entrée à l'INRA, j'ai commencé à aborder les thèmes d'*interactions protéines-protéines* [P21, L4, P6] et de *repliement d'ARN* [E22]. Arrivée à AMAP, je me suis intéressée à la *croissance des plantes* [E16, E15] et avec des collègues nous avons réfléchi à des *problématiques de modélisation* [P14, P4]. J'ai ensuite abordé par la comparaison d'arborescences et la programmation dynamique la thématique de la *coévolution*, que l'on peut voir aussi comme de la phylogénie comparée. Et plus récemment, je me suis à nouveau intéressée à l'alignement de séquences mais d'un autre type que celles de ma thèse : des génomes complets de bactériophages.

Je résume rapidement ci-dessous mes trois thématiques de recherche principales :

1. **Alignement de séquences** : Ce problème, très classique en bioinformatique, consiste à trouver la meilleure correspondance entre deux (ou plusieurs) séquences. Pour 2 séquences, des solutions exactes ([Needleman & Wunsch, 1970], [Smith & Waterman, 1981]) ou des heuristiques efficaces (BLAST [Altschul *et al.*, 1990]) existent pour le modèle évolutif comprenant insertions, délétions et substitutions de caractères. Dans ma thèse je me suis intéressée à ce problème sous un modèle évolutif comprenant 2 événements supplémentaires : l'amplification en tandem et la contraction en tandem. Ce modèle évolutif correspond aux séquences répétées en tandem. Ces séquences sont constituées de motifs adjacents et constituent une classe de séquences génétiques dont font partie microsatellites et minisatellites. Le problème est difficile car les opérations ne sont pas commutatives. Notre solution fait appel à de la programmation dynamique et à de l'algorithmique de graphe. J'ai réalisé un programme (MS_Align [L5]) qui implémente cette méthode. Il s'agit du premier programme capable d'aligner des cartes de minisatellites. À l'aide de ce programme, nous avons étudié des données biologiques provenant du minisatellite humain MSY1. [P9-10, P17-18, P22-26, O6, E7, E9, E21, E23].

   Plus tard, je me suis à nouveau intéressée à l'alignement de séquences. Ce coup-ci, il s'agissait d'aligner des séquences de génomes de bactériophages qui sont des virus de bactéries. Le modèle évolutif de ces bactériophages inclut un événement de réarrangement génomique, appelé recombinaison modulaire. Quand on compare ces génomes, ils présentent une alternance de séquences similaires et de séquences divergentes, ce qui rend inopérants les outils classiques. Nous avons mis au point une première méthode automatique d'alignement des génomes de bactériophages colinéaires, basée sur des ancres exactes ordonnées selon un ordre partiel. [L1, P1, O1, Po1, E3, E11-E12, E13].

2. **Génomique comparative** : C'est un vaste domaine de recherche qui a connu un grand essor lors de l'arrivée des premiers génomes complets d'eucaryotes au début des années 2000. Un des problèmes de cette thématique est de trouver des scénarios évolutifs entre 2 espèces ou le long d'un arbre phylogénétique. On se place ici à une échelle plus macroscopique que pour l'alignement et l'intérêt est de détecter des réarrangements génomiques. Pour cela, les espèces sont modélisées comme des suites de gènes et il s'agit de trouver un scénario permettant de passer de l'une à l'autre en minimisant le nombre ou le coût des événements évolutifs impliqués dans ce scénario. Le problème est polynomial ou NP-complet selon le modèle évolutif que l'on considère. Dans le cas des problèmes polynomiaux, il y a généralement plusieurs scénarios différents qui atteignent le coût minimum. Il faut alors avoir un critère supplémentaire de choix pour trouver le meilleur d'entre eux. Je me suis principalement intéressée à la conservation de structures dans les scénarios évolutifs, permettant de discriminer parmi les scénarios optimaux ceux qui étaient plus pertinents biologiquement. Les modèles évolutifs que j'ai considérés contiennent soit des inversions seulement, soit des DCJ (*Double-Cut and Join*). [P5, P7-8, P13, P15-16, O5, E17-20].

3. **Coévolution** : Les espèces évoluent suivant une phylogénie qui retrace leurs relations de parenté. Les gènes portés par ces espèces évoluent aussi, selon des histoires qui leur sont propres, et interagissent. Ces relations entre gènes évoluent, elles aussi, mais les méthodes qui retracent leur histoire ne sont pas encore aussi développées que les méthodes de reconstruction phylogénétique de gènes ou d'espèces. Pour une espèce donnée, nous connaissons les positions de ses gènes le long de ses chromosomes, ce qui définit des adjacences entre gènes. Nous avis mis au point une méthode algorithmique permettant de détecter et décrire les coévolutions possibles entre gènes au regard de leurs adjacences. Ceci dans le but d'inférer l'organisation des génomes ancestraux. Nous avons défini un modèle évolutif pour ces adjacences et développé un algorithme retraçant leur histoire évolutive (nommé DeCo). [L3, P3, P12, P19, E4-6, E14, E10, Po3].

   Nous nous sommes ensuite intéressés de manière plus générale à l'évolution de la structure des génomes dans un contexte phylogénétique. Nous voulions intégrer l'assemblage des séquences avec les méthodes d'inférence des événements évolutifs. Nous avons utilisé `DeCo` pour proposer des nouvelles adjacences entre gènes de génomes actuels. Cette nouvelle méthode, nommée ARt-DeCo, permet ainsi d'améliorer les assemblages en réduisant leur fragmentation. [L2, P2, P11, P19-20, E1-2, E10, O2, Po2]

Au travers de toutes ces thématiques, mes recherches s'appuient principalement sur l'algorithmique du texte, les graphes et la programmation dynamique dans le but de produire des algorithmes efficaces.

Au travers de collaborations

Tout au long de ma carrière j'ai interagi avec de nombreuses personnes. Je cite ici mes collaborations principales, c'est-à-dire avec les chercheurs et chercheuses avec qui j'ai plusieurs publications et encadrements d'étudiants :

- **Éric Rivals** *(2000-2006).* C'est avec Éric que j'ai fait mon stage de recherche de DEA sur la reconstruction d'histoire de répétitions en tandem [P26]. J'ai poursuivi en thèse sous son encadrement. Lors de ma thèse, j'ai mis au point une méthode d'alignement de séquences répétées en tandem [L5, P10, P17-18, P22-26]. Nous avons valorisé un dernier papier de thèse un peu plus tard [P9].
- **Anne Bergeron et Cédric Chauve** *(2004-2009 et à nouveau depuis ∼ 2014).* J'ai rencontré Anne et Cédric lors de mon séjour post-doctoral à Montréal en 2004. J'ai travaillé avec eux sur la thématique de la génomique comparative [P5, P7-8, P13, P15-16]. Nos chemins se sont à nouveau croisés, au travers de l'alignement de bactériophages pour Anne [P1, E3, E11-E13] et de la coévolution pour Cédric [P2, P11, P19-20]. Cédric a reçu pour 6 mois Yoann, l'étudiant en thèse que j'encadre, en séjour doctoral à Vancouver au 2^e semestre 2015.
- **Christophe Paul** *(2005-2009).* Ma collaboration avec Christophe est incluse dans celle avec Cédric et Anne sur la génomique comparative [P5, P7-8, P13, P15].
- **Éric Tannier** *(depuis 2005).* Je collabore avec Éric depuis la mise en place du groupe de travail GTGC. Nous avons commencé à publier ensemble en 2008 sur le problème des DCJ en génomique comparative [P5, P13]. Notre collaboration actuelle se centre autour de la coévolution [P2-3, P11-12, P19-20, E1, E4-6].
- **Annie Chateau** *(depuis 2008).* J'ai connu Annie lors de mon stage post-doctoral chez Anne Bergeron. Nous avons été recrutées la même année à l'université de Montpellier en 2006. Nous avons commencé à publier ensemble en 2008 sur le problème des DCJ en génomique comparative [P5, P13]. Nous avons encadré ensemble beaucoup d'étudiants [E1-5, E11-14, E17]. Notre collaboration actuelle se centre à la fois autour de la coévolution [P2, P11, P20] et de l'alignement de bactériophages [P1].

Sans oublier l'enseignement

J'ai commencé à enseigner très tôt dans ma carrière, dès mon DEA. J'ai toujours aimé enseigner et être en contact avec les étudiants. J'ai tenu à être monitrice lors de ma thèse. J'ai cherché à enseigner lorsque j'étais en post-doctorat ou chargée de recherche à l'INRA. Et je n'ai pas hésité à candidater sur des postes de maître de conférences qui offrent l'opportunité d'allier recherche et enseignements. Voici ci-dessous les caractéristiques de mes activités d'enseignement :

Diversité Tout d'abord, j'enseigne dans des modules très variés, allant des aspects les plus pratiques (réseaux, programmation) aux aspects les plus théoriques (mathématiques discrètes, algorithme du texte). J'enseigne également à tous les niveaux du L1 au M2.

Pluridisciplinarité Je suis impliquée dans des enseignements à l'interface entre la biologie et l'informatique, notamment au niveau du parcours Bioinformatique mais aussi dans le module plutôt orienté recherche du master Informatique. Je fais partie de l'équipe pédagogique du parcours Bioinformatique (BCD) et donc impliquée dans la gestion de l'offre de formation.

Fort investissement en Licence 1 J'ai effectué beaucoup d'enseignements dans des UE de L1 (Initiation à l'algorithmique : FLIN101 - 3 ans, Concepts de base en informatique : FLIN102 - 4 ans et Mathématiques discrètes : FLIN201 - depuis 10 ans), et j'en ai assuré la responsabilité (FLIN102 - 3 ans et FLIN201 - depuis 6 ans). Je suis aussi référente de groupes de L1 (3 ans). Cela consiste à aiguiller les étudiants lors de leurs premiers pas à l'université et ce tout au long de leur première année.

Transition lycée-université En plus de mon rôle de référente de groupe de L1, je m'intéresse également à la transition lycée-université. J'ai participé au groupe de travail sur ce sujet à l'IREM. J'accueille des lycéens dans mes cours et TD *via* l'opération « lycéens à l'université ». Et je suis ravie de faire de la vulgarisation scientifique quand l'occasion m'en est donnée comme lors d'un stage MathC2+ où j'ai pu expliquer la biologie évolutive et la bioinformatique à des lycéens de $1^re S$ [O3].

Créativité Enfin, enseigner c'est aussi créer des supports de cours/TD/TP et des sujets d'examens. Tout au long de ma carrière j'ai créé plusieurs supports *ex nihilo*. J'ai aussi adapté et fait évoluer plusieurs cours que j'ai repris.

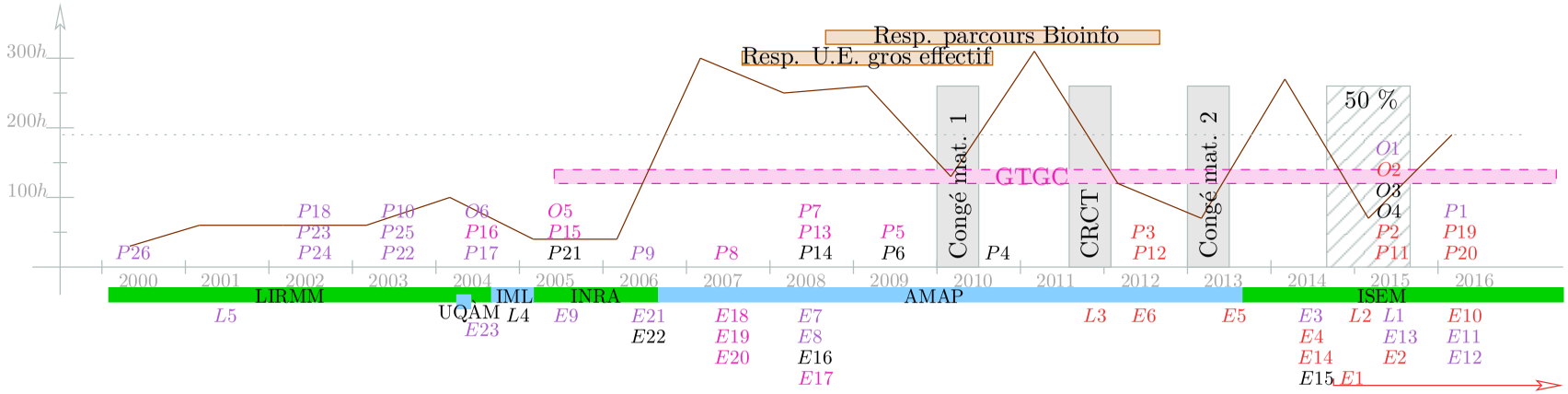

Figure 1 : Vue synthétique de la carrière. En abscisse, les années, en ordonnée le nombre d'heures d'enseignement effectuées, la ligne pointillée correspondant au service normal d'un enseignant-chercheur. La courbe marron représente les heures d'enseignement effectuées. Les barres verticales représentent les modifications de mon temps de travail. La ligne horizontale bicolore indique mon appartenance aux différents laboratoires. Les publications, P, les communications orales, O, les logiciels, L, ainsi que les encadrements d'étudiants, E, sont représentés à leur année respective et colorés selon leur appartenance à mes thématiques de recherche (violet pour l'*alignement*, rose pour la *génomique comparative*, rouge pour la *co-évolution* ou en noir si en dehors des thématiques principales).

Article DeCo

BIOINFORMATICS

Vol. 28 ECCB 2012, pages i382–i388
doi:10.1093/bioinformatics/bts374

Evolution of gene neighborhoods within reconciled phylogenies

Sèverine Bérard[1,2,*], Coralie Gallien[1], Bastien Boussau[3,4], Gergely J. Szöllősi[3], Vincent Daubin[3] and Eric Tannier[3,5,*]

[1]Univ Montpellier2, UMR AMAP, Montpellier F-34000, [2]LIRMM, CNRS, Univ Montpellier2, Montpellier F-34392, France, [3]LBBE, UMR CNRS 5558, Université de Lyon 1, Villeurbanne F-69622, France, [4]Department of Integrative Biology, UC Berkeley 4163A Valley Life Sciences Bldg Berkeley, CA 94720-3140, USA and [5]INRIA Rhône-Alpes, Montbonnot F-38322, France

ABSTRACT

Motivation: Most models of genome evolution integrating gene duplications, losses and chromosomal rearrangements are computationally intractable, even when comparing only two genomes. This prevents large-scale studies that consider different types of genome structural variations.

Results: We define an 'adjacency phylogenetic tree' that describes the evolution of an adjacency, a neighborhood relation between two genes, by speciation, duplication or loss of one or both genes, and rearrangement. We describe an algorithm that, given a species tree and a set of gene trees where the leaves are connected by adjacencies, computes an adjacency forest that minimizes the number of gains and breakages of adjacencies (caused by rearrangements) and runs in polynomial time. We use this algorithm to reconstruct contiguous regions of mammalian and plant ancestral genomes in a few minutes for a dozen species and several thousand genes. We show that this method yields reduced conflict between ancestral adjacencies. We detect duplications involving several genes and compare the different modes of evolution between phyla and among lineages.

Availability: C++ implementation using BIO++ package, available upon request to Sèverine Bérard.

Contact: Severine.Berard@cirad.frorEric.Tannier@inria.fr

Supplementary information: Supplementary material is available at *Bioinformatics* online.

1 INTRODUCTION

A phylogenetic tree describes the kin relationships between a set of homologous objects. Non-homologous objects may have other types of relationships, such as interactions, functional relationships, co-expression or neighborhood between genes. Studying the pattern of descent of these relationships can be used to define homology between them, reconstruct ancestral relationships and build phylogenetic trees.

The evolution of gene proximity or interaction has been the subject of numerous recent studies. It is for example a way to assess co-evolution between genes, even if often co-evolution is detected by searching for similarities in gene trees, but without modeling explicitly the relation that make the genes co-evolve (Rodionov *et al.*, 2011; Tuller *et al.*, 2010).

Closer to our study, (Pinney *et al.*, 2007) and (Dutkowski and Tiuryn, 2009) or (Ma *et al.*, 2008) propose methods to reconstruct ancestral protein–protein interactions or gene neighborhoods based on a model of evolution allowing gene duplications. They, however, assume that the chronology of duplications is known, which often is not the case. (Patro *et al.*, 2011) define a general problem of network evolution without this assumption and give a heuristic solution for the comparison of two species. Our model considers the more specific problem of gene neighborhoods on chromosomes, but generalizes (Patro *et al.*, 2011)'s method in that it handles an arbitrary number of species and provides an exact solution to a less constrained problem.

Several methods are aimed at building ancestral chromosomes (which can be seen as relationships between genes). Most of these methods, however, ignore duplications and losses and are limited to gene families which have exactly one representative in each studied species (Alekseyev and Pevzner, 2009; Chauve and Tannier, 2008; Chauve *et al.*, 2010; Ma *et al.*, 2006; Ouangraoua *et al.*, 2011). The number of such gene families becomes smaller and smaller as the number of species grows. Some methods take as input gene trees allowing duplications and losses (Lajoie *et al.*, 2010; Muffato *et al.*, 2010) but do not model these events and treat them as noise that is removed for the construction of chromosomes by traveling salesman-like optimization methods. (Chauve *et al.*, 2010), (Ouangraoua *et al.*, 2011) or (Zheng and Sankoff, 2011) model duplications only in the context of whole genome duplications.

Here, we propose a method that takes a species tree and a set of gene trees as inputs, and models the gain and breakage of gene adjacencies along a pair of trees, taking duplications and losses into account. We consider two genes to be 'adjacent' if they are on the same chromosome in the same genome and no other gene is located between the two. We give an exact polynomial algorithm which minimizes the number of gains and breakages of adjacencies, or more generally, the gain/breakage cost of an evolutionary scenario for gene adjacencies. The result consists of sets of 'adjacency trees', which are phylogenetic trees describing the evolution of a family of homologous adjacencies (adjacencies that share a common ancestor and derived from it).

We assume that adjacencies evolve independently from each other, so we do not model the rearrangement explicitly (inversions, translocations etc.), but model their effect on adjacencies, which thus can undergo gains and breakages.

Doing this, we solve a problem that fits in the methodological program started by (Sankoff and El-Mabrouk, 2000), which mixes rearrangements and reconciliations of phylogenetic trees (a reconciliation is an annotation of gene tree nodes by duplication or speciation events, according to a species tree).

Algorithmically, the dynamic programming principle we use generalizes the Sankoff–Fitch (Fitch, 1971; Sankoff, 1975)

*To whom correspondence should be addressed.

parsimony algorithms on binary alphabets. Indeed, when there are no duplications nor losses in gene trees, adjacencies may be described by a binary character (presence or absence in a genome) evolving along the species tree, as in (Tang and Wang, 2005) or (Feijao and Meidanis, 2011). In our case, there is also a binary character (presence or absence of an adjacency), but evolving along pairs of reconciled gene trees.

The description of the method requires that we formally introduce the three kinds of trees we handle (species, gene and adjacency trees), as well as the definition of the optimization problem we propose (Section 2). We detail some properties of the solutions, pointing at the possible advantages and drawbacks of this approach in Section 3. Then in Section 4, we describe the algorithm which solves the problem. Proofs are provided in the Supplementary Material.

We implemented the algorithm and applied it to mammalian and plant genomes. We show that compared with other methods not explicitly modelling evolutionary events, we have more precise and less erroneous views at ancestral genome organization (Section 5). In addition, we are able to detect segmental duplications including several genes, and to visualize how much the modes of evolution are different according to the considered clades or lineages.

2 MODEL

All the trees in this article have one or more vertices, they are rooted and have maximum degree 3. A tree T induces a partial order on its nodes, where descendants are lower than ancestors.

For a tree T, $V(T)$ denotes its vertex set and $L(T)$ is leaf set (vertices with no descendants, $L(T) \subseteq V(T)$). For a node N of T, $T(N)$ is the subtree of T rooted at N. $P(N)$ is the parent of N (it is defined only if N is not the root). $L(N)$ is the set of leaves of $T(N)$.

We consider all trees to be annotated, which means here that each node N of a tree is labeled by

- A type of event $E(N)$.
- A species $S(N)$.

The events $E(N)$ are taken from $\{Extant, Spec, GDup, GLos, ADup, ALos, Break\}$. These are abbreviations for 'Extant', 'Speciation', 'Gene duplication', 'Gene loss', 'Adjacency duplication', 'Adjacency loss' and 'Adjacency breakage'. Together with the 'Adjacency Gain' (abbreviated *Gain*, which never labels the nodes of the trees as there is exactly one gain per adjacency tree), they are all the evolutionary events we consider.[1] Note that *ALos* means the loss of an adjacency due to the concurrent loss of the two involved genes, while *Break* means the loss of an adjacency due to a rearrangement. In the objective function we only take *Gain* and *Break* into account, so both are given a cost $C(Gain)$ and $C(Break)$.

All trees depend on a set of extant *genomes*, which are disjoint sets of genes plus binary relations on these sets of genes called 'adjacencies'. The two genes of an adjacency are called its 'extremities'. There are three types of trees (illustrated in Fig. 1), which have the following properties.

(1) A species tree T_S describes the diversification of species. It is binary, and verifies $E(N) = Spec$ for all internal node N and $E(N) = Extant$ for all leaves N. All $S(N)$ are distinct species and if N is an internal node, $S(N)$ defines an ancestral species.

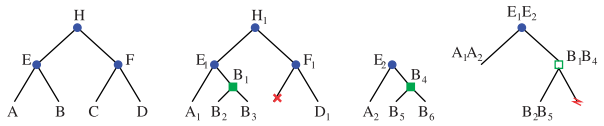

Fig. 1. Examples of a species tree (left), two gene trees (middle) and an adjacency tree (right). Blue dots are speciation nodes. Leaves are extant (species, genes, adjacencies), except the one labeled by a red cross (gene loss) or a red flash (breakage). Green squares are (gene or adjacency) duplication nodes. Gene labels refer to the species they belong to. Every node of the adjacency tree is labeled by a couple of nodes from gene trees

(2) A gene tree T_G describes the evolution of a family of homologous genes along a species tree T_S. All gene trees here are 'reconciled' with the 'LCA (Last Common Ancestor) reconciliation' (Goodman *et al.*, 1979) where all gene losses are represented by leaves, which means every node N verifies:

- If N is a leaf, then $E(N) \in \{Extant, GLos\}$, and if N is an internal node, then $E(N) \in \{Spec, GDup\}$.
- If $E(N) = Extant$, then there is a gene $G(N)$ that belongs to $S(N)$ and all such genes are distinct.
- If $E(N) = GDup$ then the children $N1$ and $N2$ of N are such that $S(N1) = S(N2) = S(N)$
- If $E(N) = Spec$ then the children $N1$ and $N2$ of N are such that there are two edges $AA1$ and $AA2$ of T_S such that $P(A1) = P(A2) = A$ and $S(N) = S(A)$, $S(A1) = S(N1)$, and $S(A2) = S(N2)$.
- Let $\mathcal{L}$ be the set of leaves of $T_G(N)$; Let $S(\mathcal{L})$ be the set of all extant species which are descendants of some $S(l)$, $l \in \mathcal{L}$; Let now N_S be the lowest node in T_S such that $S(\mathcal{L}) \subseteq \cup_{l \in L(N_S)} S(l)$; Then, $S(N) = S(N_S)$.

(3) An adjacency tree T_A describes the descent pattern of adjacencies. As adjacencies are pairs of genes, they follow the evolution of genes: if an adjacency AB descents from an adjacency CD, then A descents from C and B from D. So adjacency trees are defined given a set of reconciliated gene trees $\mathcal{T}_G$ and have to follow their LCA reconciliations. Formally, every node N of an adjacency tree verifies

- If N is a leaf, then $E(N) \in \{Extant, GLos, ALos, Break\}$ and if N is an internal node, then $E(N) \in \{Spec, GDup, ADup\}$
- If $E(N) \neq Break$, then there is a couple $A(N) = XY$ of gene tree nodes X and Y (possibly from two different gene trees) such that $S(N) = S(X) = S(Y)$.
- If $E(N) = Extant$, then $G(X)G(Y)$ is an adjacency.
- If $E(N) = GLos$, then $E(X) = GLos$ or $E(Y) = GLos$ (and not both).
- If $E(N) = ALos$, then $E(X) = E(Y) = GLos$.
- If $E(N) = Spec$, then $E(X) = E(Y) = E(N)$. In addition, N has two children $N1$ and $N2$ and either $E(N1) = Break$ (respectively, $E(N2) = Break$)) or $A(N1)$ (respectively, $A(N2)$) is a couple of children of X and Y.

[1] Even if 'Extant' is not an evolutionary event, it is included because it annotates some tree leaves.

- If $E(N)=ADup$, then $E(X)=E(Y)=GDup$. In addition, N has two children $N1$ and $N2$ either $E(N1)=Break$ (respectively, $E(N2)=Break$)) or $A(N1)$ (respectively, $A(N2)$) is a couple of children of X and Y.
- If $E(N)=GDup$, then $E(X)=GDup$ or $E(Y)=GDup$ (suppose it is Y). In addition, N has only one child $N1$ and either $E(N1)=Break$ or $A(N1)$ is a couple of genes composed of X and one child of Y.

An 'adjacency forest' is a set of adjacency trees, such that for two nodes $N1$ and $N2$ in this forest, $A(N1)\neq A(N2)$, and such that for each adjacency A from any species, there exists a leaf L in the forest, which verifies $A(L)=A$.

The cost of an adjacency tree T_A, is

$$C(T_A)=|\{v\in V(T_A), E(v)=Break\}|*C(Break)+Gain(T_A),$$

where $Gain(T_A)$ is computed in this way: if the root R of T_A is such that $A(R)=XY$ and either

- $P(X)=P(Y)$ or
- X is the root of a gene tree, and either Y is also a root, or $S(P(Y))\neq S(Y)$

then $Gain(T_A)=0$, else $Gain(T_A)=C(Gain)$. The cases where $Gain(T_A)=0$ are those arising from tandem duplications or those where the adjacency can have been gained earlier in the evolution.

The cost of an adjacency forest is the sum of the costs of all adjacency trees.

The problem we address is to take as input a species tree, a set of gene trees and a set of extant adjacencies, and to compute an adjacency forest of minimum cost. We give a polynomial algorithm which gives one optimal solution.

3 PROPERTIES

3.1 The cost of a duplication or loss event

The optimization focuses only on breaks and gains of adjacencies. The dynamic programming technique we use does not allow to count duplication and loss events in the objective function. This is because we make the hypothesis of independent evolution of couples of genes, and as long as one gene has its own events and belongs to several couples, this independence is broken.

Nevertheless, duplication events have an importance for the solutions. The duplication of an adjacency has the same cost as the independent duplication of two genes, but the events can still be discriminated because the two do not have the same effect: the independent duplications propagate only one adjacency, and the joint duplication propagates two. It is thus possible to catch the places where a joint duplication is advantageous in terms of gains and breaks.

3.2 The linearity of genomes

In extant genomes, one gene can participate in at most two adjacencies. We have not required this property in the input of the program because it is not used, and in this way we could easily adapt the problem to other kinds of relationships. The drawback of this is that there is no need that in ancestral genomes, genes participate to at most two adjacencies.

(Feijao and Meidanis, 2011) prove that in a duplication-free framework, where Fitch's algorithm is applied on the presence and absence of adjacencies, choosing the absence whenever there is a choice to make ensures that the resulting genomes are linear. When there are duplications, it is not necessarily the case and it can be seen in the data, where some conflicts remain. But as we will see in the last section, the amount of conflict is reduced compared with other kinds of algorithms, and can be used to assess the quality of the gene trees, as well as the quality of the model.

3.3 The chronology of duplications

No chronology of duplications is required in the input as in (Pinney *et al.*, 2007), (Dutkowski and Tiuryn, 2009) or (Ma *et al.*, 2008). But a chronology can be derived from the output. Indeed an adjacency duplication means that two genes are duplicated together, while two nodes of an adjacency tree such that one is the descendant of the other and both are gene duplication events define a directed relation between the two duplications, even if they are not comparable from the gene trees (not in the same tree or not comparable in one tree). But this relation is not necessarily an order relation. There are examples where temporal relationships defined by adjacency trees contradict the partial order of the nodes of one gene tree: see such an example in Figure S1 in Supplementary Material. (Patro *et al.*, 2011) proscribe this kind of conflict and propose a heuristic principle to get rid of it when it happens.

3.4 Tandem duplications

Tandem duplications are special types of duplications, where the two duplicates are adjacent. Here, tandem duplications are not modeled explicitly as a different event from ordinary duplications. However, tandem duplications of one gene are indirectly taken into account: they cost zero (as the gain of an adjacency between two children of a duplication node is costless; see Section 2), while a non-tandem duplication of one gene can cost one breakage plus two gains when one duplicate is inserted between two other genes.

3.5 The orientations of the genes

It is possible to take the orientation of the genes into account by duplicating each gene into two gene extremities and define adjacencies as relations between gene extremities instead of genes (the extremities of an adjacency are gene extremities in that case). The current implementation can be used in this way, it is just a matter of formatting the input. In this case, one gene extremity is supposed to participate in only one adjacency, and tandem duplications are not handled anymore, because there are no duplications of only one gene extremity.

4 ALGORITHM

4.1 Restriction to two gene trees

We first restrict the problem to the comparison of two gene trees, without loss of generality. To do this, the extant adjacencies are clustered according to the following relation between two distinct adjacencies AB and CD:

1). A and C are in the same gene tree, noted $G1$, as well as B and D, in a tree noted $G2$ (the roles of A and B and of C and D may be exchanged).

2a). if $G1 \neq G2$, then there are two nodes $N1 \in G1, N2 \in G2$ such that $S(N1) = S(N2)$ and A and C are descendants of $N1$, while B and D are descendants of $N2$.

2b). if $G1 = G2$, then the lowest common ancestor of A and B is the same node as the lowest common ancestor of C and D (it is necessarily a duplication node).

This relation between adjacencies satisfying all conditions is an equivalence relation (reflexive, symmetric and transitive). Equivalence classes are treated independently. This is justified by the following lemma, whose proof stands in the Supplementary Material.

LEMMA 1. *If there is a tree of adjacencies which contains adjacencies AB and CD, then AB and CD are in the same equivalence class.*

In other words, if AB and CD are not in the same class they cannot be homologous. The converse is not true however. Solutions for one class may consist of several adjacency trees.

This clustering allows to divide the problem into equivalence classes, which concern one or two gene trees. If in an equivalence class, adjacencies have extremities in the same tree, by definition of the classes, there is a common ancestor to all pairs of extremities of adjacencies. By removing this vertex, we get two trees rooted at its children, and all adjacencies have one extremity in each of these two trees.

So we may restrict ourselves to this case where we have exactly two gene trees and all adjacencies are between these two trees. Moreover, we may suppose that each tree is rooted at the lowest common ancestor of all genes involved in adjacencies of the chosen class, because we may simply consider the subtree rooted at this vertex. This yields that the two roots are necessarily assigned to the same species.

4.2 Recurrence formulas

Formally, we have two gene trees T_G^1 and T_G^2, extant adjacencies have one extremity in each tree, and if R^1 and R^2 are the respective roots of T_G^1 and T_G^2, then $S(R^1) = S(R^2)$.

For a pair of nodes $(v^1, v^2) \in V(T_G^1) \times V(T_G^2)$ such that $S(v^1) = S(v^2)$, we compute two values, $c_1(v^1, v^2)$ and $c_0(v^1, v^2)$ by recurrence formulas described in the sequel. Remark that we only consider pairs of nodes annotated with the same species because an adjacency is always linking genes from the same genome. We prove that these numbers have the following properties (proofs are in the Supplementary Material, Appendix 2).

THEOREM 1.
- *$c_1(v^1, v^2)$ is the minimum cost of an adjacency forest F for the adjacencies between two gene trees $T_G^1(v^1)$ and $T_G^2(v^2)$, such that there is a node N in F with $A(N) = v^1 v^2$.*
- *$c_0(v^1, v^2)$ is the minimum cost of an adjacency forest F for the adjacencies between two gene trees $T_G^1(v^1)$ and $T_G^2(v^2)$, such that there is no node N in F with $A(N) = v^1 v^2$.*

In consequence, the minimum cost of an adjacency forest will be given by computing the minimum between $c_1(R^1, R^2)$ and $c_0(R^1, R^2)$.

The recurrence for the computation of $c_1(v^1, v^2)$ and $c_0(v^1, v^2)$ follows a case analysis, according to the type of event associated to v^1 and v^2. The roles of v^1 and v^2 are symmetrical. We note $ca(v)$ and $cb(v)$ the two children of a node v.

Case 1. $E(v^1) = Extant$ and $E(v^2) = Extant$.
If $v^1 v^2$ is an adjacency then $c_1(v^1, v^2) = 0$ and $c_0(v^1, v^2) = \infty$; else $c_1(v^1, v^2) = \infty$ and $c_0(v^1, v^2) = 0$.

Case 2. $E(v^1) = GLos$ and $E(v^2) \neq GLos$.
In this case $c_1(v^1, v^2) = 0$ and $c_0(v^1, v^2) = 0$.

Case 3. $E(v^1) = GLos$ and $E(v^2) = GLos$.
In this case $c_1(v^1, v^2) = 0$ and $c_0(v^1, v^2) = 0$. This case has to be distinguished from the previous one for the backtracking procedure described in the following subsection.

Case 4. $E(v^1) \in \{Extant, Spec\}$ and $E(v^2) = GDup$.

$$c_1(v^1, v^2) = \min \begin{cases} c_1(v^1, ca(v^2)) + c_0(v^1, cb(v^2)) \\ c_0(v^1, ca(v^2)) + c_1(v^1, cb(v^2)) \\ c_1(v^1, ca(v^2)) + c_1(v^1, cb(v^2)) + C(Gain) \\ c_0(v^1, ca(v^2)) + c_0(v^1, cb(v^2)) + C(Break) \end{cases}$$

$$c_0(v^1, v^2) = \min \begin{cases} c_0(v^1, ca(v^2)) + c_0(v^1, cb(v^2)) \\ c_0(v^1, ca(v^2)) + c_1(v^1, cb(v^2)) + C(Gain) \\ c_1(v^1, ca(v^2)) + c_0(v^1, cb(v^2)) + C(Gain) \\ c_1(v^1, ca(v^2)) + c_1(v^1, cb(v^2)) + 2 * C(Gain) \end{cases}$$

Case 5. $E(v^1) = Spec$ and $E(v^2) = Spec$.
Suppose without loss of generality that $S(ca(v^1)) = S(ca(v^2))$ and $S(cb(v^1)) = S(cb(v^2))$.

$$c_1(v^1, v^2) = \min \begin{cases} c_1(ca(v^1), ca(v^2)) + c_1(cb(v^1), cb(v^2)) \\ c_1(ca(v^1), ca(v^2)) + c_0(cb(v^1), cb(v^2)) + C(Break) \\ c_0(ca(v^1), ca(v^2)) + c_1(cb(v^1), cb(v^2)) + C(Break) \\ c_0(ca(v^1), ca(v^2)) + c_0(cb(v^1), cb(v^2)) + 2 * C(Break) \end{cases}$$

$$c_0(v^1, v^2) = \min \begin{cases} c_0(ca(v^1), ca(v^2)) + c_0(cb(v^1), cb(v^2)) \\ c_1(ca(v^1), ca(v^2)) + c_0(cb(v^1), cb(v^2)) + C(Gain) \\ c_0(ca(v^1), ca(v^2)) + c_1(cb(v^1), cb(v^2)) + C(Gain) \\ c_1(ca(v^1), ca(v^2)) + c_1(cb(v^1), cb(v^2)) + 2 * C(Gain) \end{cases}$$

Case 6. $E(v^1) = GDup$ and $E(v^2) = GDup$.
In this case $c_1(v^1, v^2) = min(D1, D2, D12)$ where
$D1$ is the cost in the case the v^1 duplication comes first,
$D2$ is the cost in the case the v^2 duplication comes first,
$D12$ is the cost in the case where the v^1 and v^2 duplications are simultaneous.

$$\mathbf{D1} = \min \begin{cases} c_1(ca(v^1), v^2) + c_0(cb(v^1), v^2) \\ c_0(ca(v^1), v^2) + c_1(cb(v^1), v^2) \\ c_1(ca(v^1), v^2) + c_1(cb(v^1), v^2) + C(Gain) \\ c_0(ca(v^1), v^2) + c_0(cb(v^1), v^2) + C(Break) \end{cases}$$

$$\mathbf{D2} = \min \begin{cases} c_1(v^1, ca(v^2)) + c_0(v^1, cb(v^2)) \\ c_0(v^1, ca(v^2)) + c_1(v^1, cb(v^2)) \\ c_1(v^1, ca(v^2)) + c_1(v^1, cb(v^2)) + C(Gain) \\ c_0(v^1, ca(v^2)) + c_0(v^1, cb(v^2)) + C(Break) \end{cases}$$

$\mathbf{D12} = \min$ (over all 16 following cases)

$$
\begin{cases}
(1)\ c_1(ca(v^1),ca(v^2)) + c_1(cb(v^1),cb(v^2)) + c_0(ca(v^1),cb(v^2)) + c_0(cb(v^1),ca(v^2)) \\
(2)\ c_1(ca(v^1),ca(v^2)) + c_1(cb(v^1),cb(v^2)) + c_0(ca(v^1),cb(v^2)) + c_1(cb(v^1),ca(v^2)) + C(Gain) \\
(3)\ c_1(ca(v^1),ca(v^2)) + c_1(cb(v^1),cb(v^2)) + c_1(ca(v^1),cb(v^2)) + c_0(cb(v^1),ca(v^2)) + C(Gain) \\
(4)\ c_1(ca(v^1),ca(v^2)) + c_1(cb(v^1),cb(v^2)) + c_1(ca(v^1),cb(v^2)) + c_1(cb(v^1),ca(v^2)) + 2*C(Gain) \\
(5)\ c_1(ca(v^1),ca(v^2)) + c_0(cb(v^1),cb(v^2)) + c_0(ca(v^1),cb(v^2)) + c_0(cb(v^1),ca(v^2)) + C(Break) \\
(6)\ c_1(ca(v^1),ca(v^2)) + c_0(cb(v^1),cb(v^2)) + c_0(ca(v^1),cb(v^2)) + c_1(cb(v^1),ca(v^2)) + C(Gain) + C(Break) \\
(7)\ c_1(ca(v^1),ca(v^2)) + c_0(cb(v^1),cb(v^2)) + c_1(ca(v^1),cb(v^2)) + c_0(cb(v^1),ca(v^2)) + C(Gain) + C(Break) \\
(8)\ c_0(ca(v^1),ca(v^2)) + c_1(cb(v^1),cb(v^2)) + c_0(ca(v^1),cb(v^2)) + c_0(cb(v^1),ca(v^2)) + C(Break) \\
(9)\ c_0(ca(v^1),ca(v^2)) + c_1(cb(v^1),cb(v^2)) + c_0(ca(v^1),cb(v^2)) + c_1(cb(v^1),ca(v^2)) + C(Gain) + C(Break) \\
(10)\ c_0(ca(v^1),ca(v^2)) + c_1(cb(v^1),cb(v^2)) + c_1(ca(v^1),cb(v^2)) + c_0(cb(v^1),ca(v^2)) + C(Gain) + C(Break) \\
(11)\ c_0(ca(v^1),ca(v^2)) + c_0(cb(v^1),cb(v^2)) + c_1(ca(v^1),cb(v^2)) + c_1(cb(v^1),ca(v^2)) \\
(12)\ c_0(ca(v^1),ca(v^2)) + c_1(cb(v^1),cb(v^2)) + c_1(ca(v^1),cb(v^2)) + c_1(cb(v^1),ca(v^2)) + C(Gain) \\
(13)\ c_1(ca(v^1),ca(v^2)) + c_0(cb(v^1),cb(v^2)) + c_1(ca(v^1),cb(v^2)) + c_1(cb(v^1),ca(v^2)) + C(Gain) \\
(14)\ c_0(ca(v^1),ca(v^2)) + c_0(cb(v^1),cb(v^2)) + c_1(ca(v^1),cb(v^2)) + c_0(cb(v^1),ca(v^2)) + C(Break) \\
(15)\ c_0(ca(v^1),ca(v^2)) + c_0(cb(v^1),cb(v^2)) + c_0(ca(v^1),cb(v^2)) + c_1(cb(v^1),ca(v^2)) + C(Break) \\
(16)\ c_0(ca(v^1),ca(v^2)) + c_0(cb(v^1),cb(v^2)) + c_0(ca(v^1),cb(v^2)) + c_0(cb(v^1),ca(v^2)) + 2*C(Break)
\end{cases}
$$

$$
c_0(v^1,v^2) = \min
\begin{cases}
\text{Cases where the } v^1 \text{ duplication comes first} \\
c_0(ca(v^1),v^2)+c_0(cb(v^1),v^2) \\
c_0(ca(v^1),v^2)+c_1(cb(v^1),v^2)+C(Gain) \\
c_1(ca(v^1),v^2)+c_0(cb(v^1),v^2)+C(Gain) \\
c_1(ca(v^1),v^2)+c_1(cb(v^1),v^2)+2*C(Gain) \\
\text{Cases where the } v^2 \text{ duplication comes first} \\
c_0(v^1,ca(v^2))+c_0(v^1,cb(v^2)) \\
c_0(v^1,ca(v^2))+c_1(v^1,cb(v^2))+C(Gain) \\
c_1(v^1,ca(v^2))+c_0(v^1,cb(v^2))+C(Gain) \\
c_1(v^1,ca(v^2))+c_1(v^1,cb(v^2))+2*C(Gain)
\end{cases}
$$

We do not examine the case $E(v^1)=Extant$ and $E(v^2)=Spec$ because in this case $S(v^1)\neq S(v^2)$.

The algorithm implements these recurrence formulas in an iterative way following a dynamic programming technique, by computing the costs in a post-order traversal of the couples of tree nodes.

4.3 Backtracking

The recurrence formulas allow the construction of a cost matrix which rows are the nodes of the first gene tree, and columns are the nodes of the second gene tree. The nodes of the adjacency forest are constructed while backtracking on the cost matrix starting at $\min(c_1(R^1,R^2),c_0(R^1,R^2))$. The backtracking procedure classically follows each cost on the chosen line in the recurrence formulas, creating adjacency trees from root to leaves. A node N with $A(N)=v^1v^2$ is created each time $c_1(v^1,v^2)$ is chosen. The event labeling this node depends on the events labeling v^1 and v^2 : *Extant* for Case 1, *GLos* for Case 2, *ALos* for Case 3, *GDup* for Cases 4 and 6. (D1&D2), *Spec* for Case 5 and *ADup* for Case 6 (D12). A node N with $E(N)=Break$ is created each time there is a $C(Break)$ in the chosen formula.

Edges between the nodes follow the pattern of descent between adjacencies:

- *Break* nodes are leaves, and their parent are the nodes constructed in the formula where $C(Break)$ occurs;
- In Cases 4 and 6 (D1&D2), there is an edge between v^1v^2 and one of $v^1ca(v^2)$, $v^1cb(v^2)$, $ca(v^1)v^2$, $cb(v^1)v^2$, if c_1 is chosen for either of them.
- In Cases 5 and 6 (D12), there is an edge between v^1v^2 and one or two of $ca(v^1)ca(v^2)$, $cb(v^1)cb(v^2)$, $cb(v^1)ca(v^2)$, $ca(v^1)cb(v^2)$ if c_1 is chosen for either of them (there can be arbitrary choices for equivalent solutions).

Recurrence formulas imply that the backtracking procedure does not create twice the same node: each formula computes the cost for v^1v^2 between pairs of nodes where at least one is a descendant of v^1 or v^2.

An example of an algorithm input and output is drawn on Figure 2.

4.4 Complexity

The algorithm takes as input a dataset composed by a species tree, several gene trees and a list of adjacencies. It first computes the equivalence classes of adjacencies. Then for each class it constructs two subtrees to compute c_0 and c_1 costs on their roots and applies the backtracking procedure. The algorithm outputs the adjacency forest resulting from the union of all adjacency forests built on each class.

Let n be the number of gene trees and k be the maximum size of a tree. The algorithm runs in $O(n^2\times k^2)$. Indeed, the maximum number of adjacency equivalence classes is bounded by $O(n^2)$, while for each equivalence class, every couple of node is examined with a constant-time case analysis.

In practice, the number of equivalence classes is much closer to n than to n^2 and trees are small compared with the total number of genes. For all datasets we tested, including dozens of species and thousand of genes, the execution time was under 10 min.

5 IMPLEMENTATION AND APPLICATION

We implemented the algorithm using the Bio++ platform (Dutheil *et al.*, 2006). The program, named *DeCo* (Detection of Co-evolution or DeCoration of trees), takes as input a species tree, a set of genes along with the species they are in, a set of adjacencies and a set of gene trees.

We tested it on four datasets, with costs $C(Gain)=C(Break)=1$. The first and second datasets are based on 5039 gene trees from the Ensembl database (release 57) restricted to mammalian assembled genomes (11 species).[2] The first set of trees are those provided in this

[2]The parsimony framework practically makes it necessary to work with only assembled genomes, since we would count to many breaks for unassembled

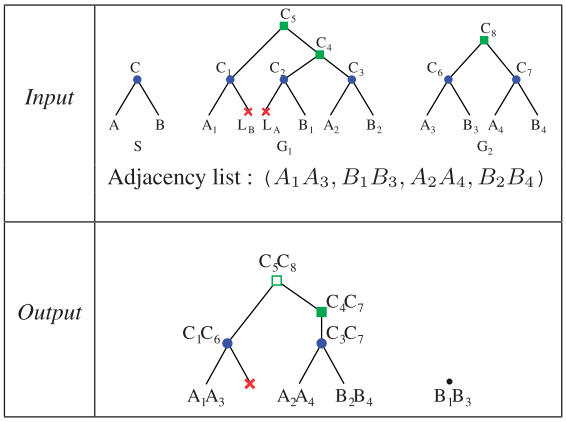

Fig. 2. Example of the application of the algorithm on two genes trees, G_1 and G_2, a species tree S and an adjacency list shown on the line *Input*. The costs are $C(Gain)=C(Break)=1$. All the costs $c_b(E_i,E_j)$ are computed for $b\in 0,1$, $E\in A,B,C$, $i,j\in[1..8]$, with E_i in G_1 and E_j in G_2. As a result $c_0(C_5,C_8)=2$ while $c_1(C_5,C_8)=1$. Therefore, the adjacency forest on the line *Output* contains C_5C_8. The left tree has cost 0 while the right one costs $C(Gain)=1$ for the gain of the adjacency B_1B_3

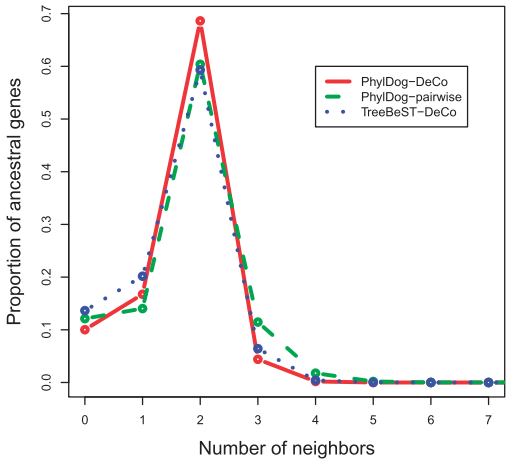

Fig. 3. Proportion of genes having k neighbors, function of k. Red plain line is obtained with DeCo and PhylDog trees. Green dashed line is obtained with PhylDog trees and the pairwise alternative. Blue dotted line is obtained with TreeBeST trees and DeCo

database, made according to the TreeBeST pipeline (Vilella *et al.*, 2009). The second consists of the trees reconstructed by the PhylDog method (Boussau *et al.*, 2012), with an explicit model of duplication and losses of trees. Both sets of trees were reconciled according to the LCA method (Goodman *et al.*, 1979), which gives gene trees with the properties written in Section 2).

Then, we computed ancestral adjacencies according to the method described here, and compared with the 'pairwise alternative', an implementation of the principles used by (Chauve and Tannier, 2008), (Muffato *et al.*, 2010), (Bertrand *et al.*, 2010) or (Boussau *et al.*, 2012), in which adjacencies are constructed by comparing couples of species (the method is described in the Supplementary Material, Appendix 3) instead of all genomes together.

We computed the degree of each ancestral gene, that is, the number of adjacencies which has it as an extremity. As shown in Figure 3, most ancestral genes have degree 2, which means the signal of linearity of the ancestral genomes is recovered. We can observe the gain obtained by using PhylDog trees instead of TreeBeST trees (red plain versus blue dotted line), and the gain obtained by using DeCo instead of the pairwise alternative (red plain versus green dashed line). These two gains are nearly equivalent, showing that to get better ancestral genomes, we need good trees as well as good adjacency inference algorithms. Better trees tend to give a better estimate of the ancestral gene content, minimizing the degree 0 (probably wrong) genes, while the adjacency inference algorithm may minimize the number of genes with degree >2: convergent evolution can yield false ancestral adjacencies, which add to the two true ones. Convergent evolution is impossible to handle in a pairwise method.

The third and fourth datasets are constructed from the Ensembl (release 65) and EnsemblPlant (release 12) databases, restricted to some assembled mammalian (11 species, 19 217 gene trees with an average of 10 genes) and angiosperm (9 genomes, 35 182 gene trees, with an average of 9 genes) species. We chose these two clades for a phylogenetic comparison because the estiated divergence times are similar, and there are approximatively as many assembled genomes in both. We compared the number of segmental duplications involving more than one gene in these two datasets. In Figure 4, phylogenetic trees of mammals and angiosperms are drawn, in which branch length is the number of pairs of genes duplicated together over the total number of ancestral genes found in the same branch. We find that on average branch lengths are more than three times longer in plants, indicating genome architectures rapidly evolving compared with slow mammalian ones. Angiosperm genomes have been shaped by several whole genome duplications: at the basis of monocots, a triplication at the basis of dicots, plus one event on the Maize and Poplar lineages, and two on the Arabidopsis one. These events probably create a long branch in Poplar, or Glycine, but are not always visible (*e.g.* in *Arabidopsis*) due to differentiated losses which blurred the synteny signal. The difference in branch length can partly be due to whole genome duplications. But measuring the average size of the duplicated segments by computing

$$\frac{\#GDup}{\#GDup-\#Adup}.$$

we found no significant difference between the two phyla ($=1.08$ on average among all branches for both), indicating that the changes in genome architectures following a whole genome duplication are not fully accessible to this method. The long branch at the basis of eutheria would deserve more studies to know to which extend it is artifactual and due to the quality of gene trees.

6 PERSPECTIVES

The algorithm can easily be extended to handle other relations than adjacencies (interactions, regulations, co-expression or any functional relation which can evolve by gain or breakage like adjacencies). It can be seen as even more adapted to less constrained relations (without a linear organization). Indeed, if a gene is lost,

ones, preventing the reconstruction of some ancestral adjacencies. It is possible to envisage branch specific costs, where unassembled genomes would have low breakage cost, and then could be used in a dataset.

Fig. 4. Angiosperm and mammalian phylogenies, where branch lengths are proportional to the number of adjacency duplications normalized by the number of genes. The scale is indicated at the bottom left of the two figures

no adjacency is automatically and freely gained between its two neighbors in this model. But the computation time should be higher for other relations, as the possible number of relations is a quadratic function of the number of genes, while the number of adjacencies only grows linearly.

Possible extensions can be to include transfers (Doyon *et al.*, 2011), incomplete lineage sorting or gene conversion (Rasmussen and Kellis, 2012) to the possible events. And also to allow other types of reconciliations than the LCA one. More flexible cost functions for duplications may also be desirable, but in this case the independent evolution between adjacencies is lost, and the use of dynamic programming does not seem generalizable.

ACKNOWLEDGEMENT

The authors thank Cedric Chauve and Vincent Ranwez for useful discussions.

Funding: Ancestrome project, ANR-10-BINF-01 (to S.B., B.B., G.S., V.D. and E.T.). Univ Montpellier2 and INRIA (to C.G.). Postdoctoral fellowship from the Human Frontier Science Program and the CNRS (to B.B.)

Conflict of Interest: None declared.

REFERENCES

Alekseyev,M.A. and Pevzner, P.A. (2009) Breakpoint graphs and ancestral genome reconstructions. *Genome Res.*, **19**, 943–957.

Bertrand,D. *et al*. (2010) Reconstruction of ancestral genome subject to whole genome duplication, speciation, rearrangement and loss. In *Proceedings of WABI'10, Algorithms in Bioinformatics*, Lecture Notes in Bioinformatics, Springer, Berlin Heidelberg, pp 78–89.

Boussau,B. *et al*. (2012) Genome-scale coestimation of species and gene trees, in revision.

Chauve,C. and Tannier,E. (2008) A methodological framework for the reconstruction of contiguous regions of ancestral genomes and its application to mammalian genomes. *PLoS Comput. Biol.*, **4**, e1000234.

Chauve,C. *et al*. (2010) Yeast ancestral genome reconstructions: the possibilities of computational methods II. *J. Comput. Biol.*, **17**, 1097–1112.

Doyon,J.-P. *et al*. (2011) Models, algorithms and programs for phylogeny reconciliation. *Brief. Bioinform.*, **12**, 392–400.

Dutheil,J. *et al*. (2006) Bio++: a set of c++ libraries for sequence analysis, phylogenetics, molecular evolution and population genetics. *BMC Bioinformatics*, **7**, 188.

Dutkowski,J. and Tiuryn,J. (2009) Phylogeny-guided interaction mapping in seven eukaryotes. *BMC Bioinformatics*, **10**, 393.

Feijao,P. and Meidanis,J. (2011) SCJ: a breakpoint-like distance that simplifies several rearrangement problems. *IEEE/ACM Trans. Comp. Biol. Bioinf.*, **8**, 1318–1329.

Fitch,W. M. (1971) Toward defining the course of evolution: minimum change for a specified tree topology. *Sys. Zool.*, **20**, 406–416.

Goodman,M. *et al*. (1979) Fitting the gene lineage into its species lineage, a parsimony strategy illustrated by cladograms constructed from globin sequences. *Syst. Zool.*, **28**, 132–163.

Lajoie,M. *et al*. (2010) Inferring the evolutionary history of gene clusters from phylogenetic and gene order data. *Mol. Biol. Evol.*, **27**, 761–772.

Ma,J. *et al*. (2006) Reconstructing contiguous regions of an ancestral genome. *Genome Res.*, **16**, 1557–1565.

Ma,J. *et al*. (2008) Dupcar: reconstructing contiguous ancestral regions with duplications. *J. Comput. Biol.*, **15**, 1007–1027.

Muffato,M. *et al*. (2010) Genomicus: a database and a browser to study gene synteny in modern and ancestral genomes. *Bioinformatics*, **26**, 1119–1121.

Ouangraoua,A. *et al*. (2011) Reconstructing the architecture of the ancestral amniote genome. *Bioinformatics*, **27**, 2664–2671.

Patro,R. *et al*. (2011) Parsimonious reconstruction of network evolution. In *Proceedings of WABI'11*, Vol. LNBI 6833, p. 237249.

Pinney,J.W. *et al*. (2007). Reconstruction of ancestral protein interaction networks for the bzip transcription factors. *Proc. Natl. Aca. Sa. USA*, **104**, 20449–20453.

Rasmussen,M.D. and Kellis,M. (2012) Unified modeling of gene duplication, loss, and coalescence using a locus tree. *Genome Res.*, **22**, 755–765.

Rodionov,A. *et al*. (2011) A new, fast algorithm for detecting protein coevolution using maximum compatible cliques. *Algorithms Mol. Biol.*, **6**, 17.

Sankoff,D. (1975) Minimal mutation trees of sequences. *SIAM J. Appl. Math.*, **28**, 35.

Sankoff,D. and El-Mabrouk,N. (2000) Duplication, rearrangement and reconciliation. In Sankoff, D. and Nadeau, J.H. (eds), *Comparative Genomics: Empirical and Analytical Approaches to Gene Order Dynamics, Map alignment and the Evolution of Gene Families*, Vol. 1 of *Computational Biology*. Kluwer Academic Press, Kluwer Academic publishers Dordrecht/Boston/London, pp 537–550.

Tang,J. and Wang,L. (2005) Improving genome rearrangement phylogeny using sequence-style parsimony. In *Proceedings 5th IEEE Conference on Bioinformatics and Bioengineering (BIBE 2005)*, pp. 137–144.

Tuller,T. *et al*. (2010) Reconstructing ancestral gene content by coevolution. *Genome Res.*, **20**, 122–132.

Vilella,A.J. *et al*. (2009) Ensemblcompara genetrees: Complete, duplication-aware phylogenetic trees in vertebrates. *Genome Res.*, **19**, 327–335.

Zheng,C. and Sankoff,D. (2011) On the pathgroups approach to rapid small phylogeny. *BMC Bioinformatics*, **12 (Suppl. 1)**, S4.

Article ARt-DeCo

Anselmetti et al. BMC Genomics 2015, 16(Suppl 10):S11
http://www.biomedcentral.com/1471-2164/16/S10/S11

BMC Genomics

RESEARCH Open Access

Ancestral gene synteny reconstruction improves extant species scaffolding

Yoann Anselmetti[1,3], Vincent Berry[2], Cedric Chauve[5], Annie Chateau[2], Eric Tannier[3,4], Sèverine Bérard[1,2*]

From 13th Annual Research in Computational Molecular Biology (RECOMB) Satellite Workshop on Comparative Genomics
Frankfurt, Germany. 4-7 October 2015

Abstract

We exploit the methodological similarity between ancestral genome reconstruction and extant genome scaffolding. We present a method, called ARt-DeCo that constructs neighborhood relationships between genes or contigs, in both ancestral and extant genomes, in a phylogenetic context. It is able to handle dozens of complete genomes, including genes with complex histories, by using gene phylogenies reconciled with a species tree, that is, annotated with speciation, duplication and loss events. Reconstructed ancestral or extant synteny comes with a support computed from an exhaustive exploration of the solution space. We compare our method with a previously published one that follows the same goal on a small number of genomes with universal unicopy genes. Then we test it on the whole Ensembl database, by proposing partial ancestral genome structures, as well as a more complete scaffolding for many partially assembled genomes on 69 eukaryote species. We carefully analyze a couple of extant adjacencies proposed by our method, and show that they are indeed real links in the extant genomes, that were missing in the current assembly. On a reduced data set of 39 eutherian mammals, we estimate the precision and sensitivity of ARt-DeCo by simulating a fragmentation in some well assembled genomes, and measure how many adjacencies are recovered. We find a very high precision, while the sensitivity depends on the quality of the data and on the proximity of closely related genomes.

Introduction

Knowledge of genome organization (gene content and order) and of its dynamics is an important question in several fields such as cancer genomics [1-3], to understand gene interactions involved in a common molecular pathway [4], or evolutionary biology, for example to establish a species phylogeny by comparative analysis of gene orders [5].

On one side, studying genome organization evolution, and in particular proposing gene orders for ancestral genomes, requires well assembled extant genomes, while, on the other side, the assembly of extant genomes can in return benefit from evolutionary studies. This calls for integrative methods for the joint scaffolding of extant genomes and reconstruction of ancestral genomes.

The reconstruction of ancestral genome organization is a classical computational biology problem [6], for which various methods have been developed [7-13]. The rapid accumulation of new genome sequences provides the opportunity to integrate a large number of genomes to reconstruct their structural evolution. However, a significant proportion of these genomes is incompletely assembled and remains at the state of contigs (permanent draft) as illustrated by statistics on GOLD [14]. To improve assemblies, methods known as scaffolding were developed to order contigs into scaffolds. Most scaffolding methods use either a reference genome, or the information provided by paired-end reads, or both [15-22]. We refer to Hunt et al. [23] for a detailed comparative analysis of recent scaffolding methods.

In recent developments, scaffolding methods taking into account multiple reference genomes and their

* Correspondence: severine.berard@umontpellier.fr
[1]Institut des Sciences de l'Évolution de Montpellier (ISE-M), Place Eugène Bataillon, Montpellier, 34095, France
Full list of author information is available at the end of the article

phylogeny have been proposed [24-27]. It suggests a methodological link with ancestral genome reconstructions: if ancestral genes are considered as contigs, scaffolding extant or ancestral contigs in a phylogenetic context differs only in the location of the considered genome within the phylogeny (leaf or internal node). This link has been observed [28] and exploited to develop a method for combining scaffolding and ancestral genomes reconstruction [29]. However, the latter, due to the computational complexity of the ancestral genome reconstruction problem, is currently limited to a few genomes and to single-copy universal genes.

We propose to overcome this computational complexity by considering independent ancestral gene neighborhood reconstructions [12] instead of whole genomes, which allows to scale up to dozens of whole genomes and to use as input data genes with complex histories. We develop a method that scaffolds ancestral and extant genomes at the same time. The algorithm improves over previous methods of scaffolding by the full integration of the inference of evolutionary events within a phylogenetic context.

The principle of our method is imported from DeCo [12]. So we call it ARt-DeCo for Assembly Recovery through DeCo. DeCo is an algorithm for ancestral synteny reconstruction. It is a dynamic programming scheme on pairs of reconciled gene tree, generalizing the classic dynamic programming scheme for parsimonious ancestral character reconstructions along a tree. It computes a parsimonious set of ancestral gene neighborhoods, the cost being computed as the weighted sum of gains and losses of such neighborhoods, due to genome rearrangements. In addition to DeCo, ARt-DeCo considers a linkage probability for each couple of genes in extant genomes, that is included in the cost function in order to be able to propose gene neighborhoods in extant as well as in ancestral genomes.

We implemented ARt-DeCo and tested it on several data sets. First we reproduced the experiment of [29] on 7 tetrapod genomes limited to universal unicopy genes, with comparable accuracy. Then we used all genes from 69 eukaryotic genomes from the Ensembl database [30]. The program runs in about 18 h and is able to propose ancestral genome structures and thousands of extant scaffolding linkages. We examine in details one of them, chosen randomly on the panda genome, and show why it seems reasonable to propose it as an actual scaffolding adjacency. Then on a reduced data set of 39 whole mammalian genomes, we tested the precision and sensitivity of the scaffolding performed by ARt-DeCo by simulating artificial fragmentation of the human or horse genomes, removing up to 75% of the known gene neighborhoods of these well assembled genomes, and comparing the removed adjacencies with the ones proposed by ARt-DeCo. We measure a >95% precision, while sensitivity, as expected, depends on the quality of the data and on the presence of closely related extant genomes. This denotes the domain of efficiency of our method: a vast majority of proposed adjacencies can be considered with confidence, but the final resulting scaffolding is still incomplete.

Ancestral and extant adjacencies

We describe the ARt-DeCo algorithm for the joint reconstruction of ancestral genomes and scaffolding of extant genomes. An overview is depicted in Figure 1.

Input

The input of the method is

- A species tree with all considered genomes and their descent pattern; We suppose the number of chromosomes of each extant genome is known.
- A set of genes for all considered genomes, clustered into homologous families; for each family a rooted gene tree depicts the descent pattern of the genes.
- A set of *adjacencies, i.e.*, pairwise relations between neighboring genes AB on extant chromosomes. Genes A and B are called the *extremities* of the adjacency AB. We consider as neighbors two genes that are not separated by another gene present in the dataset, but a relaxed definition can be used with no impact on the method itself.

Internal nodes of the species tree are labeled with ancestral species (we always consider ancestral species at the moment of a speciation) and leaves are labeled with extant species. Gene trees are reconciled with the species tree: all ancestral genes are labeled by the ancestral species they belong to, so the input yields a gene content for all ancestral species. Genes and species are partially ordered by the descent relation, so we may speak of a last, or lowest, or most recent common ancestor. Here, as in [12], we use a reconciliation minimizing the number of duplications and losses of genes.

A module of ARt-DeCo is able to produce a suitable input from the raw Ensembl Compara [30] gene tree files and a species tree if needed. Once the input is given, two preliminaries are necessary: partitioning extant adjacencies and computing an a priori adjacency probability for each extant species. They are detailed in the two following subsections.

A partition of extant adjacencies

The goal of this step is, without loss of generality, to reduce the analysis of the whole data set to the independent analysis of pairs of gene trees and adjacencies, each

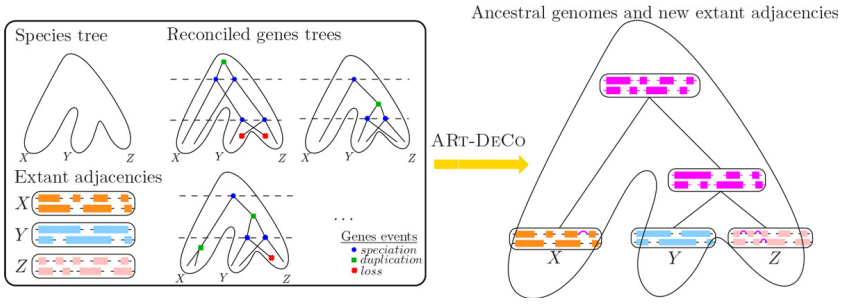

Figure 1 Input and output of the ARt-DeCo method. The left box shows the input of ARt-DeCo: a species tree (here on extant species X, Y and Z), the adjacencies in the genome of extant species (each colored block represents a contig, that is, a linear arrangement of genes, linked by adjacencies) and the reconciled genes trees for their genes. The output of ARt-DeCo is shown on the right-hand side in magenta color: the method computes both new adjacencies for extant species and contigs for the ancestral species.

having an extremity in each of the gene trees. Moreover, we want that the roots of the two gene trees correspond to ancestral genes mapping to the same ancestral species.

The partition is realized thanks to a necessary condition for two adjacencies to share a common ancestor. Two adjacencies A_1B_1 and A_2B_2, for genes A_1, A_2, B_1, B_2, may have a common ancestor AB only if A_1 and A_2 (respectively B_1 and B_2) are in the same gene family, so have a common ancestor A (respectively B), and such that A and B belong to the same species. In other words, the ancestral adjacency has the possibility to exist only when the genes of this adjacency are present in a same ancestral species.

It is easy to check that this relation is an equivalence relation, which then partitions adjacencies into equivalence classes. Each equivalence class C can be represented by two ancestral genes: they are the most ancient distinct A and B genes involved in the two-by-two comparisons of adjacencies A_1B_1 and A_2B_2 in this class. Necessarily every adjacency in this class has a gene which is a descendant of A, and another which is a descendant of B. A and B are in the same species (ancestral or extant), and cannot be the descendant one of another.

For a node N of a gene tree T, $T(N)$ is the subtree of T rooted at N. Consider the two disjoint subtrees $T(A)$ and $T(B)$. All adjacencies in the equivalence class C have one extremity which is a leaf of $T(A)$ and the other which is a leaf of $T(B)$. So each equivalence class may be treated independently from the other, and the input can be restricted, without loss of generality, to $T(A)$ and $T(B)$.

An *a priori* probability for all adjacencies

Given two extant genes v_1 and v_2 from the same extant genome G, we give an *a priori* probability that there is an adjacency between v_1 and v_2. If the genome G is perfectly assembled, then this probability is given by the input, that is, it is 1 if there is an adjacency in the input and 0 otherwise. But if the genome G is not perfectly assembled, then this probability depends on the quality of the assembly. It will allow the program to propose more adjacencies in extant genomes that are more fragmented.

We note n the number of contigs in an extant genome (which is the number of genes minus the number of adjacencies if all contigs are linear arrangements of genes), and p the number of chromosomes. We always have $n \geq p > 0$. All contigs are assumed to have two distinct extremities.

We call a *solution* of the scaffolding problem a set of $n - p$ adjacencies between the extremities of contigs, which forms p chromosomes from the n contigs. The number of different solutions for given n and p is denoted by $f(n, p)$.

Let v_1 and v_2 be extremities of two different contigs; the *a priori* probability $P(v_1 \sim v_2)$ of v_1 and v_2 to be adjacent if they are not seen adjacent in the data and $n > p$ is:

$$P(v_1 \sim v_2) = \rho(v_1, v_2) \times \frac{f(n-1, p)}{f(n, p)}$$

where ρ is a correction function which is equal to 4 if v_1 and v_2 are the only genes in their contigs, 2 if one of v_1 v_2 is the only gene in its contig but not the other, and 1 otherwise. If $n = p$, we have $P(v_1 \sim v_2) = 0$ if the adjacency v_1v_2 is not in the data, and $P(v_1 \sim v_2) = 1$ otherwise.

For the computation of $P(v_1 \sim v_2)$ we use the following formula for $f(n, p)$.

Lemma 1 *For each* $n \geq 1$ *and* $p \in \mathbb{N}^*$, *we have:*

$$f(n,p) = \frac{n!}{p!} 2^{n-p} \binom{n-1}{p-1}.$$

Proof First remark that the formula $f(n, p)$ can be extended to the case where $p > n$ and to the case where $n \geq 1$ and $p = 0$, by setting its value to 0 in those cases (there is no possible way to transform n contigs into p chromosomes). In those cases, the equality is still true, since $\binom{n-1}{p-1}$ is then equal to 0. Thus, in all what follows, we use this extension of definition when needed.

We proceed now by induction on $n \geq 1$.

Base case: $n = 1$, it is straightforward since $f(1,1) = 1 = \frac{1!}{1!}2^0\binom{0}{0}$, and for $p > 1$, we have $f(1,p) = 0 = \frac{1!}{p!}2^{1-p}\binom{n-1}{p-1}$, since the binomial coefficient is equal to 0 in this case.

Induction: we suppose that for each $k \leq n$, for each $p \in \mathbb{N}^*$, we have:

$$f(k,p) = \frac{k!}{p!}2^{k-p}\binom{k-1}{p-1}.$$

We consider $f(n+1, p)$, for a fixed $p \in \mathbb{N}^*$. We sum over all possibilities for one specific chromosome to be composed of x contigs. This gives the recurrence formula:

$$f(n+1,p) = \frac{1}{p}\sum_{x=1}^{n+1-(p-1)}\left(2^{x-1}\frac{(n+1)!}{(n+1-x)!}f(n+1-x,p-1)\right).$$

Where $\frac{1}{p}$ is used to avoid couting the same solution several times. $2^{x-1}\frac{(n+1)!}{(n+1-x)!}$ can be written $2^{x-1}x!\binom{n+1}{x}.x!$ representing the number of possibilities to sort x contigs, 2^{x-1} allows to take into account contig orientations and $\binom{n+1}{x}$ represents the number of possibilities to pull x contigs of $n+1$.

By induction hypothesis, we have:

$$f(n+1,p) = \frac{1}{p}\sum_{x=1}^{n+1-(p-1)}\left(2^{x-1}\frac{(n+1)!}{(n+1-x)!}\times\frac{(n+1-x)!}{(p-1)!}2^{n+1-x-(p-1)}\binom{n+1-x-1}{p-2}\right)$$

which simplifies into:

$$f(n+1,p) = \frac{(n+1)!}{p!}2^{n+1-p}\sum_{x=1}^{n+2-p}\binom{n-x}{p-2}.$$

We change the variable in the sum, let $h = n\text{-}x$. Then, we have:

$$f(n+1,p) = \frac{(n+1)!}{p!}2^{n+1-p}\sum_{h=p-2}^{n-1}\binom{h}{p-2}.$$

By the Hockey-stick's identity, namely for all $n, r \in \mathbb{N}, n > r, \sum_{i=r}^{n}\binom{i}{r} = \binom{n+1}{r+1}$, we finally obtain:

$$f(n+1,p) = \frac{(n+1)!}{p!}2^{n+1-p}\binom{n}{p-1},$$

which concludes the proof.

The expression of $f(n, p)$ leads to the following simple expression for P:

$$P(v_1 \sim v_2) = \rho(v_1, v_2) \times \frac{n-p}{2n(n-1)}.$$

We define $p(S) = \frac{n-p}{2n(n-1)}$ the part of this formula that does not depend on v_1 and v_2, as an assembly fragmentation measure for genome S.

A Dynamic programming scheme

We largely refer to DeCo [12] for a full description of the dynamic programming scheme, and only describe the overall principle and the differences we introduce. Adjacencies are constructed between ancestral genes (equivalently internal gene tree nodes), and propagate along gene trees. For two nodes v_1 and v_2 defining genes belonging to the same (ancestral or extant) species, we define a *solution* as a descent pattern of ancestral and extant adjacencies explaining the input extant adjacencies that have an extremity in $T(v_1)$ and another in $T(v_2)$. So a solution is a set of ancestral adjacencies and descent relations linking ancestral and extant adjacencies. The cost of a solution is the cumulative cost of gains and breakages of adjacencies (due to rearrangements) in the descent pattern, according to an individual cost for gains (*Gain*) and breakages (*Br*).

More precisely we define two costs $c_0(v_1, v_2)$ (respectively $c_1(v_1, v_2)$), which are the minimum cost previously mentioned, given that there is an (respectively there is no) adjacency between v_1 and v_2 in a solution. All c_0 and c_1, for every couple v_1 and v_2, can be computed by the dynamic programming scheme described in [12]. ARt-DeCo and DeCo have the same time complexity, that is $O(g^2 \times k^2)$ where g is the number of gene trees in the input and k be the maximum size of a tree.

The main difference is that in [12] extant genomes were supposed to be perfectly assembled and in particular, if v_1 and v_2 are extant genes (or equivalently gene tree leaves, which corresponds to Case 1 in [12]), then DeCo would use the following scoring rules:

$c_0(v_1, v_2) = \infty$ and $c_1(v_1, v_2) = 0$ if v_1v_2 is an adjacency in the data, otherwise $c_0(v_1, v_2) = 0$ and $c_1(v_1, v_2) = \infty$.

Here we modify these rules (it is the only case different from the dynamic programming equations given in [12] and Additional file 1) and propose instead that

$$c_1(v_1, v_2) = -\log_b(P(v_1 \sim v_2)) \;\&\; c_0(v_1, v_2) = -\log_b(1 - P(v_1 \sim v_2))$$

These formulas define a cost system which is consistent with the previous one: when $n = p$ (perfectly assembled genomes) it gives the same result. When it is not the case, the costs are between 0 and ∞ as the probabilities go from 0 to 1.

We left the basis of the logarithm as a variable b. Giving a value to b determines a sensitivity for finding new adjacencies. It can be dependent on the genome S hosting v_1 and v_2. We choose the basis so that $c_1(v_1, v_2) < c_0(v_1, v_2) + Br$ where Br is the cost of an adjacency breakage. Thus an adjacency is systematically proposed when it is inferred in the closest ancestor of S. The adjacency is obviously not always true in that case, because a rearrangement can have broken it in S. But it is a necessary condition to be able to propose any adjacency. If a genome is highly fragmented, proposing such an adjacency is more likely to lead to a true scaffolding adjacency than to cancel an evolutionary rearrangement. The relation $c_1(v_1, v_2) < c_0(v_1, v_2) + Br$ yields:

$$b > \left(\frac{1 - p(S)}{p(S)}\right)^{1/Br}$$

where $p(S)$ represents the fragmentation of the genomes hosting v_1 and v_2, defined in the previous section. Preliminary experiments show that there is indeed a phase change in the number of inferred adjacencies when b reaches the right hand side of the above equation (Figure 2). Above this value, the number of inferred adjacencies is mainly constant, while it changes radically for smaller values. In following experiments we then fixed b to 1.05 times the right hand side of the above equation, in order to be sure to be on the plateau following the phase change.

Exploration of the solution space

The dynamic programming scheme of DeCo allows the quantitative exploration of the whole solution space. This has been developed, in the DeCo model, in [31], where it was shown how to explore all solutions (*i.e.* evolutionary histories for adjacencies) under a Boltzmann probability distribution defined as follows: for a given instance (pair of gene trees and set of extant adjacencies) with solution space S, the parsimony score of an adjacency history h is denoted by $s(h)$, and the Boltzmann probability of h is defined as $e^{-s(h)/kT} / \sum_{g \in S} e^{-S(g)/kT}$. Here kT is a constant that can be used to skew the probability distribution: when kT is small, parsimonious histories dominate the distribution, while a large kT leads to a more uniform distribution over the whole solution space.

This allows to associate to a feature of a solution (here an ancestral adjacency) a support defined as the ratio between the sum of the probabilities of the solutions that contain this feature and the sum of the probabilities of all solutions. This approach has been implemented in the DeClone software [31]. We integrated this possibility to ARt-DeCo and thus associate a support to both extant and ancestral adjacencies. Computations were run with a value of the kT constant equal to 0.1 to ensure that the Boltzmann distribution is highly dominated by optimal and slightly sub-optimal solutions. This value was chosen after preliminary tests on a subset of instances that showed that scenarios sampled with this value of kT were in very large majority optimal scenarios.

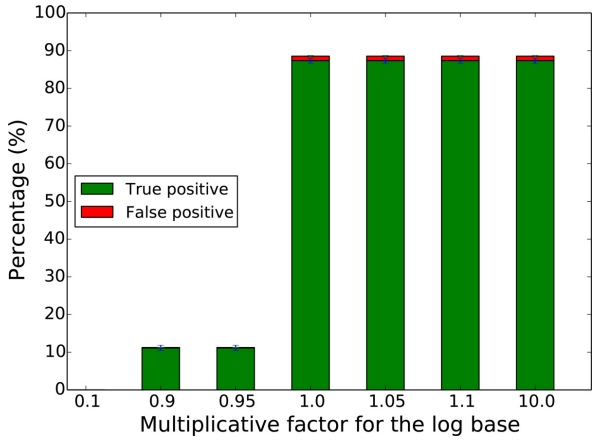

Figure 2 Determination of a good value for base log b. We simulated 550 fissions on a data set of 7 tetrapod species (see Section Results) and evaluate the ability of ARt-DeCo to recover broken adjacencies by the simulated fissions for different values for the base log b. On the x axis is the multiplicative factor of $\left\lceil \left(\frac{1 - p(S)}{p(S)}\right)^{\frac{1}{Br}} \right\rceil$, where $Br = 1$. As we can see on the graph, there is a phase change at 1.0, meaning that from this value a good number of adjacencies can be proposed. Increasing the multiplicative factor does not qualitatively change the results. This experiment was repeated for different numbers of simulated fissions (see Additional file 2) and different species trees, and in all cases results exhibited the same profile.

Results

We tested ARt-DeCo on three data sets. The first one is composed of 7 tetrapod species with only universal unicopy genes, and aims at comparing our method with the method of Aganezov *et al.* [29]; on this data set, we obtain comparable results. Then we ran ARt-DeCo on the complete Ensembl Compara [30] database, including 69 eukaryotic species and 20,279 gene families with arbitrary numbers of duplications and losses. This shows that

ARt-DeCo scales up and can process large data sets of whole genomes; for this data set we examine carefully one scaffolding adjacency proposed by ARt-DeCo in the poorly assembled panda genome and provide evidence it is likely a true scaffolding adjacency. The third data set we consider is the restriction to the 39 eutherian mammals genomes of the previous data set. The computational efficiency of ARt-DeCo allows to reproduce the computation many times with simulated missing adjacencies, and replicates to obtain empirical error bars on the measures. We performed all experiments with fixed costs for adjacency gains (*Gain*) and breakages (*Br*), respectively set to 3 and 1. There are several reasons for this discrepancy: first the actual number of adjacencies is very low compared to the space of possible adjacencies, which makes more probable to break a particular one $\left(p = \frac{1}{\#adj}\right)$ than to gain a particular one $\left(p = \frac{2}{\#genes \times (\#genes - 1)}\right)$: there is a huge unobserved space of possible solutions that should affect the costs; second it has been remarked that good statistical estimates of genomic distances when genomes are coded by the presence or absence of adjacencies are obtained with a state of possible adjacencies 3 times larger than the number of adjacencies [32].

Seven tetrapods - comparison with the method of Aganezov *et al.* [29]

By querying Biomart [33], we produced a data set similar to the one described in Aganezov *et al.* [29]: it consists in 8,818 universal unicopy gene families from Human, Chimp, Macaque, Mouse, Rat, Dog and Chicken. The latter was not present in the data set of Aganezov *et al.* [29], and we included it here because of a fundamental difference between the two methods: our method works with rooted phylogenies whereas Aganezov *et al.* [29] is not sensitive to the position of the root. This means that our method cannot scaffold an outgroup species, simply because, for any adjacency absent from the outgroup, it is more parsimonious to assume it is gained in all ingroup species. So we just added a distant outgroup to scaffold the 6 species used in [29].

We produced different sets of randomly fragmented genomes by considering $n = 50$ to $n = 1050$ random artificial breaks (or "fissions") in genomes, sticking to the described experiments in [29]. This means we simply removed n random adjacencies per genome from the data set. For each n, we replicated the experiment 30 times.

For each replicate with n random artificial adjacency breaks, let TP be the number of removed adjacencies that ARt-DeCo retrieves and FP be the number adjacencies not in the removed ones but proposed by ARt-DeCo.

We measured, following [29], approximations of the sensitivity and precision:

$$'True\ positive' = \frac{TP}{n + FP}$$

$$'False\ positive' = \frac{FP}{n + FP}$$

Aganezov *et al.* report that "*True positive*" takes values between 75% and 87%, and "*False positive*" takes values from 0.5% to 9%, varying in function of n. We report similar values for all our experiments (see Table 1).

Thus, on small data sets and discarding gene families with complex histories, we obtain similar performance. The next experiments illustrate that the contribution of our method is then to be able to process much larger and much more complex data sets.

69 eukaryotes - a proof of scaling up

We ran ARt-DeCo on the full Ensembl Compara database (1,222,543 protein coding extant genes in 69 extant species) in about 18 h. The input contains 1,023,492 adjacencies in the extant genomes, showing that many genomes assemblies are highly fragmented, from 11 chromosomes for the perfectly assembled opossum genome to 12,704 contigs for the wallaby genome, an order of magnitude comparable to the number of genes. In Figure 3, the black bars show the proportion of genes with 0, 1 or 2 syntenic neighbors in the extant input genomes. Around 30% of genes have at most one neighbor, while we would expect less than 1% for perfectly assembled genomes.

ARt-DeCo predicts 36,445 new extant adjacencies. As shown in Figure 3 (red bars) there is a significant increase in extant genes with 2 syntenic neighbors, as in

Table 1 Statistics on adjacencies recover by ARt-DeCo on 7 tetrapods dataset with different number of simulated breaks

#Breaks (n)	50	150	250	350	450	550
TP	283	829	1364	1895	2418	2922
FP	14	16.5	21	24.5	32	40
True positive	88.64%	**89.98%**	89.38%	89.00%	88.32%	87.35%
False positive	4.40%	1.78%	1.39%	**1.15%**	1.18%	1.19%
#Breaks (n)	650	750	850	950	1050	
TP	3431	3917	4398	4875	5338	
FP	46	57	63	73.566	83	
True positive	86.84%	85.87%	85.12%	84.38%	83.58%	
False positive	1.17%	1.25%	1.22%	1.27%	1.30%	

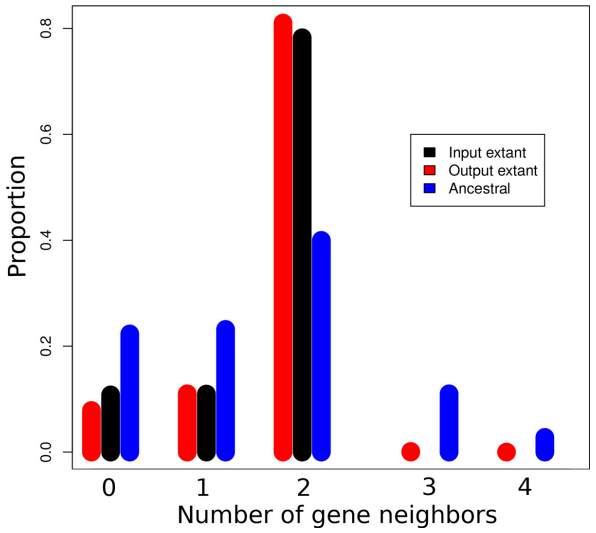

Figure 3 Number of syntenic neighbors of extant and ancestral genes. Distribution of the proportion of genes with a given number of neighbors in extant and ancestral genomes before and after adjacency prediction for the data set on 69 eukaryotes.

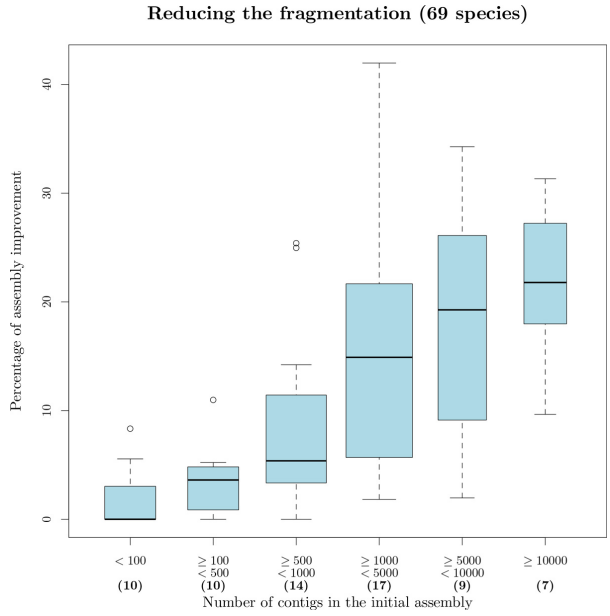

Figure 4 Percentage of improvement of genome assemblies, according to their initial fragmentation. Statistics are obtained for the 69 eukaryotes dataset, excluding genomes that are already well assembled (bold figures between parenthesis indicate cardinalities of classes).

a *bona fide* scaffolding, at the expense of a very small number of genes with more than two neighbors, corresponding to syntenic conflicts. Complementary computations show that more than 99.6% contigs in extant species remain linear (two genes having degree 1 and others degree 2), in spite of the large number of contig connections inferred in some species (*e.g.*, the number of contigs goes from 2,599 down to 1,864 for *Ailuropoda melanoleuca* or from 11,528 down to 7,930 for *Tarsius syrichta*). Figure 4 shows the percentage of improvement given by the method relatively to the initial data. Precisely, this percentage is obtained by computing $\frac{C_I - C_N}{C_I - p}$ on extant species which are not completely assembled, where C_I, resp. C_N and p are the number of contigs in the initial genome, resp. the number of contigs after adjacency inference by ARt-DeCo and the expected number of chromosomes. The figure shows that the more fragmented is the initial genome, the better ARt-DeCo improves it.

Figure 5 shows the average degree of non-linearity of extant species with at least one non-linear contig, representing 43 of 69 species, computed only on non-linear contigs. Degree of non-linearity (D_{nl}) correspond to supplementary degree of genes that are not consistent with a linear conformation and computed as follow:

$$D_{nl} = \sum_{x=1}^{n} (d_x - 2) + (m - 2)$$

n = *N umber of genes with degree* >2
d_x = *N umber of degree of gene x*
m = *N umber of genes with degree* = 1

On 43 species with non-linear contigs, 23 have non-linear contigs with only one extra branch. For the 20 remaining species, contigs are more branchy and few are circular.

We also reconstruct 1,547,546 ancestral adjacencies on 3,245,572 ancestral genes. As previously noted [34], errors in gene trees artificially inflate the number of ancestral genes computed with gene tree/species tree reconciliations. Nevertheless, the pattern of ancestral gene neighborhood shows mainly ancestral genes with 0, 1 or 2 neighbors, and some conflicts rapidly decreasing (Figure 3, blue bars). More than 92% inferred contigs in ancestral genomes are linear. Figure 6 presents the density histogram of average degree of non-linearity for ancestral species on inferred contigs that are not linear. As we can see most of the species have an average of degree non-linearity less than 20 meaning that in average non-linear ancestral contigs have a degree of non-linearity less than 20. Moreover, more than 50% of ancestral species have a degree of non-linearity less than 6 indicating that most of non-linear ancestral contigs are weakly branchy. However a large number of ancestral species have contigs strongly branchy and circular and need additional processes to obtain linear contigs. It is likely that better ancestral and extant genomes would result from better input gene trees.

We analyze in details one predicted extant adjacency, in order to understand why it is present in the output

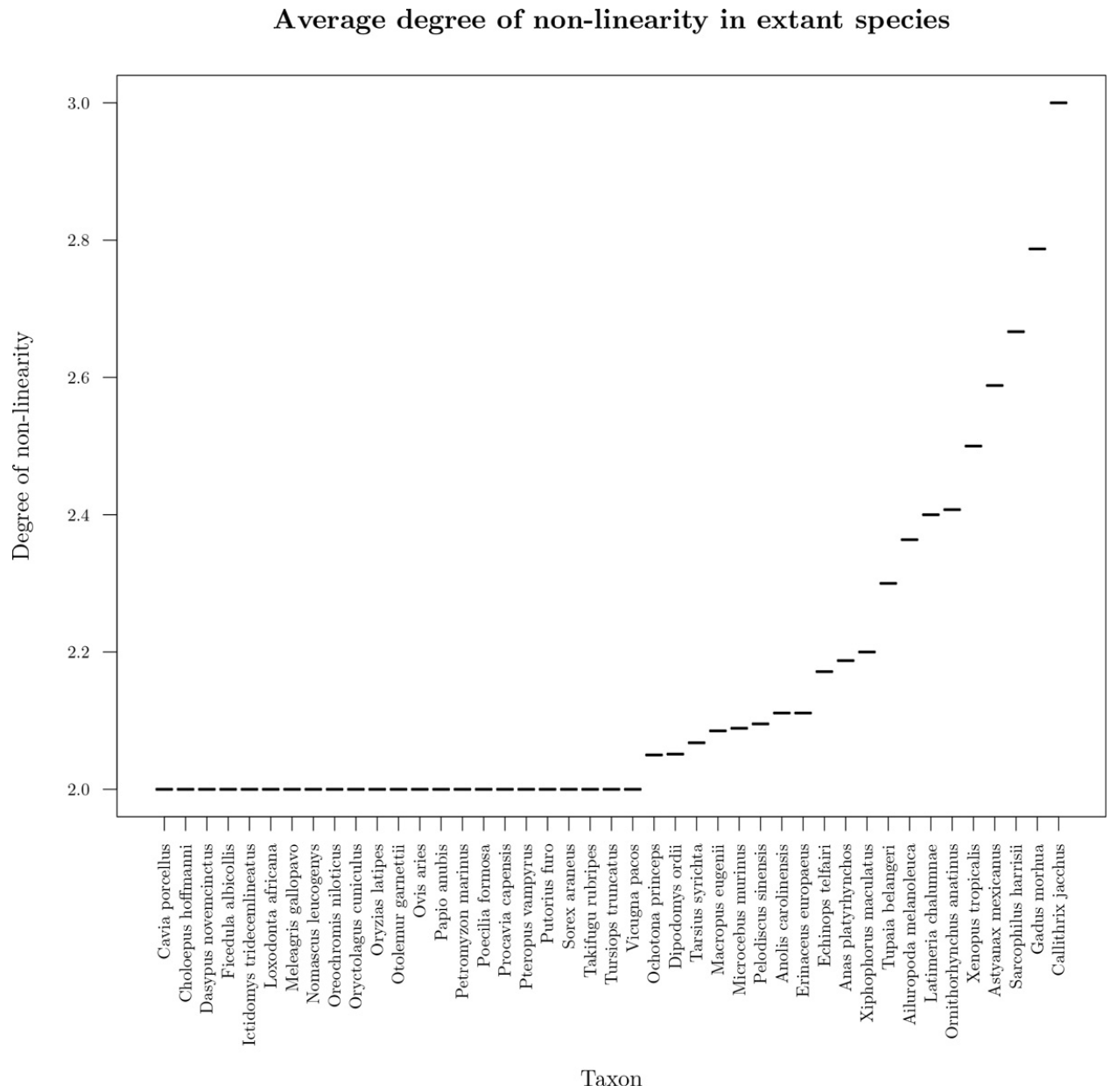

Figure 5 Distribution of average degree of non-linearity on non-linear contigs by extant species. On this graph, only species with at least one non-linear contig are shown, representing 43 of the 69 species. 23 of these species have an average degree of non-linearity of 2, meaning that their non-linear contigs contain an extra branch (one gene of degree 3 and one extra gene with degree 1).

of ARt-DeCo and not in the input. The adjacency we chose is randomly taken from the predicted ones between a contig and a chromosome in the panda genome (*Ailuropoda melanoleuca*). For this adjacency between genes RCSD1 and CREG1, we analyze gene neighborhoods around homologous adjacencies in others species. On Figure 7, we represent the species tree with information on evolutionary events that occurred along species tree (adjacency loss, duplication and gain, and gene loss and duplication) and adjacency status with color code on species name (red for species without RCSD1-CREG1 adjacency, blue for species with RCSD1-CREG1 adjacency in Ensembl database and green for species for which ARt-DeCo infers an adjacency between RCSD1 and CREG1 while not present in Ensembl). To illustrate the validity of new adjacencies inferred by ARt-DeCo we analyze the gene neighborhood around RCSD1-CREG1 adjacency. As, we can see on Figure 7, we observe that gene order and content is the same between cat (*Felis catus*), human (*Homo*

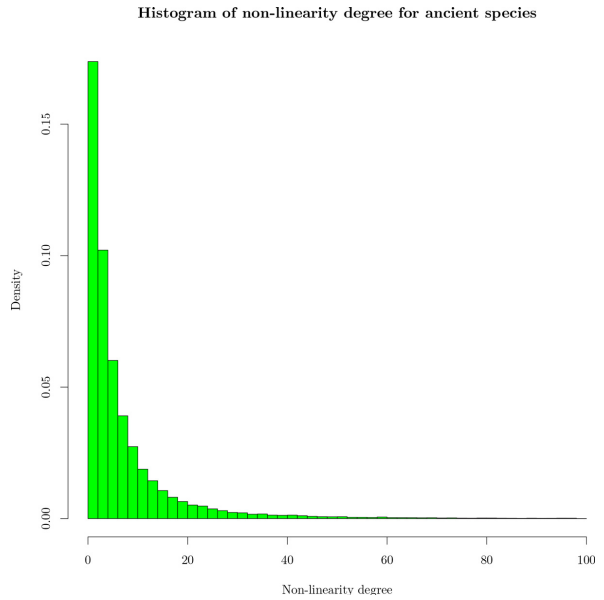

Figure 6 Density histogram of average degree of non-linearity for ancestral species on non-linear contigs. Most of the ancestral species have an average degree of non-linearity of 20 meaning that in average contigs of ancestral species reconstructed by ARt-DeCo have a degree of non-linearity less than 20. This figure shows that a large number are non-linear and additional operations are necessary to obtain linear contigs.

sapiens) and panda except for ADCY10 gene (in black) that is not present in panda genome. The RCSD1-CREG1 adjacency is confirmed by adjacency support of >99.96%, according to the exploration of the solution space. Due to the high fragmentation of panda genome and previous information it is reasonable to think that this adjacency is true. We observe the same results for the kangaroo rat (*Dipodomys ordii*) genome with gene content similarity with close species, high adjacency support (>99.99%) and high genome fragmentation (9,720 contigs).

This analysis also allowed to see a possible lack of data in Ensembl. As can be seen on the mouse lemur (*Microcebus murinus*) genome, there is no adjacency between CREG1 and RCSD1 because no RCSD1 gene has been annotated in Ensembl for this species. However, the gene content and order around CREG1 is very similar to that of close genomes (*e.g.*, human). Moreover, Ensembl contains an incomplete DNA sequence for the equivalent position of CD247 and RCSD1 genes in mouse lemur. This implies that the genes CD247 and RCSD1 could be present in mouse lemur but are not annotated.

39 mammals - validity

We switched to a smaller dataset to measure the validity of the method, because the computing time don't allow too many replicates in the entire database. We selected all protein coding gene families from the 39 eutherian mammal genomes stored in the Ensembl database [30].

ARt-DeCo proposes 1,056,418 ancestral adjacencies and 22,675 new adjacencies in extant genomes. A proportion of 95% of these adjacencies have a >0.9 support, meaning that they are present in over 90% of parsimonious solutions, computed as described in [31] for a kT value equal to 0.1 (chosen to ensure that the probability distribution over the solution space is highly dominated by optimal solutions).

Figure 8A shows the shape of extant genomes through the number or cumulative support of adjacencies incident to one gene. The distribution for all genes is plotted for extant genomes in the input and in the output, taking support into account or not. The figure shows that the genomes scaffolded with ARt-DeCo host more genes having exactly two neighbors (highest peak in the figure, the input is in black and output is in red). Peaks are integer numbers: unweighted measures have all their values integer while weighted measures still have peaks at integers. Complementary computations show that more than 99.7% contigs in extant species are indeed linear.

Figure 8B is the analog of Figure 8A but for ancestral genomes: blue for the number of neighbors in the version of ARt-DeCo without support (only one solution is given), and pink for the version of ARt-DeCo with support. The several peaks of the graph illustrate that ancestral genomes are not in the shape of disjoint paths, as we would expect it from linear genomes. This was already remarked in [12], and is likely due to errors in gene trees in the Ensembl database [35,36]. Additional computations show that ancestral species indeed contain a larger proportion of nonlinear contigs: only 91.2% contigs are linear for those species, among which contigs hosting only one gene are more represented than in extant genomes. Thus, a small part of the inferred adjacencies are incorrect, leading to some artificially tree-like or cyclic contigs.

The bars with supports are more dispersed, as expected, because they take their values from non integer numbers. It puts the conflicts into perspective: when a gene has more than two neighbors, usually one adjacency is less supported.

We also performed experiments with artificial adjacency breaks as in the 7 tetrapods experiment. We removed from 1 to 75% input adjacencies from the human genome, and then from the horse genome. We chose the human and horse genome because of their phylogenetic position: one has many closely related genomes, while the other is rather distant from its closest neighbor inside the placentals. This allows us to measure the effect of the presence of closely related genomes in

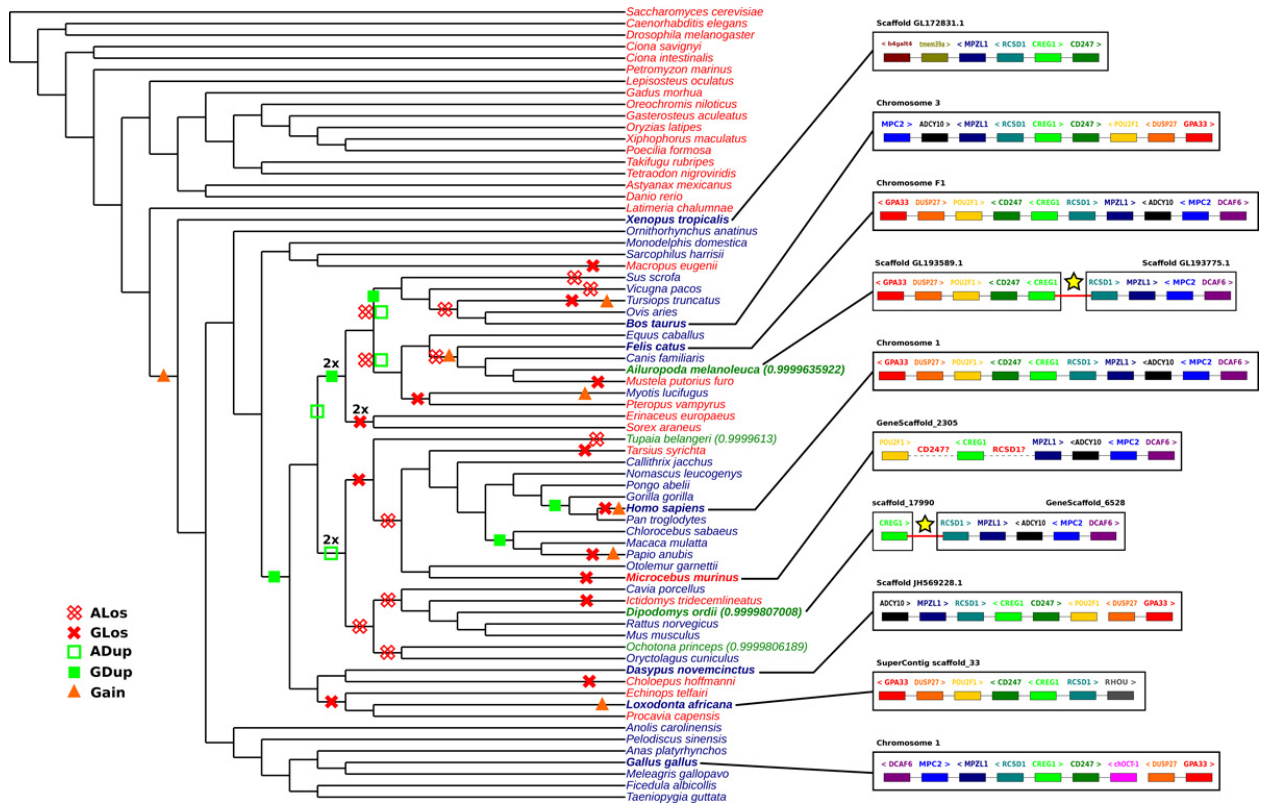

Figure 7 Evolutionary history of the adjacency between RCSD1 (turquoise) and CREG1 (light green). ARt-DeCo infers the creation of this adjacency at the root of Amniots. We integrated the different evolutionary events concerning the RCSD1-CREG1 adjacency along the species tree. Empty red crosses represents an adjacency losses (*i.e.*, cases where both adjacent genes are lost at the same time); each full red cross represents a gene loss (only one of the genes is lost); each empty green square indicates an adjacency duplication (places where the two adjacent genes are duplicated together); a full green square indicates a gene duplication and an orange triangle represents an adjacency gain. Color code for species name gives information on adjacency status. In red species, RCSD1 and CREG1 genes are not adjacent, while blue species host the RCSD1-CREG1 adjacency as described by Ensembl, and green species have the RCSD1-CREG1 adjacency inferred by ARt-DeCo (though it is absent from Ensembl). For green species, the adjacency support is indicated. For some species, representing most of the clades in Ensembl, we illustrate the gene content around the RCSD1-CREG1 adjacency, which illustrates strong similarities in the genomes of blue and green species.

the given phylogeny. The two situations are very different because of the bias in taxonomic sampling around human. The presence of very close relatives in the data set makes the problem much easier for the human genome.

Indeed, as shown on Figure 9, the sensitivity (measured by the "*T rue positive*" rate as in the previous section, to keep a coherence and comparability with [29]) of the method is around 40% for the human genome, and 5% for the horse genome. The precision is high in all cases, decreasing with the number of broken adjacencies but the number of "*F alse positives*" stays quite low.

The complexity of the data is a real issue here. While in a prepared, filtered data set of 7 tetrapods the sensitivity was above 80% in all cases, here with all genes from 39 genomes including duplications and losses, it is much lower in all cases.

From all data sets, we observe that the precision of ARt-DeCo is always high, while the sensitivity varies according to the conditions. So we can see the method as a rather sure predictor of a small number of scaffolding linkages, without the pretension to reconstruct fully assembled genomes.

Discussion and Conclusion

Ancestral gene order reconstruction, when ancestral genes are given, can be seen as a scaffolding problem. Indeed ancestral genes may be seen as contigs, and finding an order between contigs is a similar problem in both extant and ancestral genomes. If this similarity had already been remarked and exploited in some way [24,26,37], a fully integrated approach has only recently been achieved by Aganezov *et al.* [29], with a method which was limited to universal unicopy genes and a small number of genomes. Extending DeCo [12], a software aimed at reconstructing ancestral genomes and scaling up to dozens of genomes with possibly complex histories,

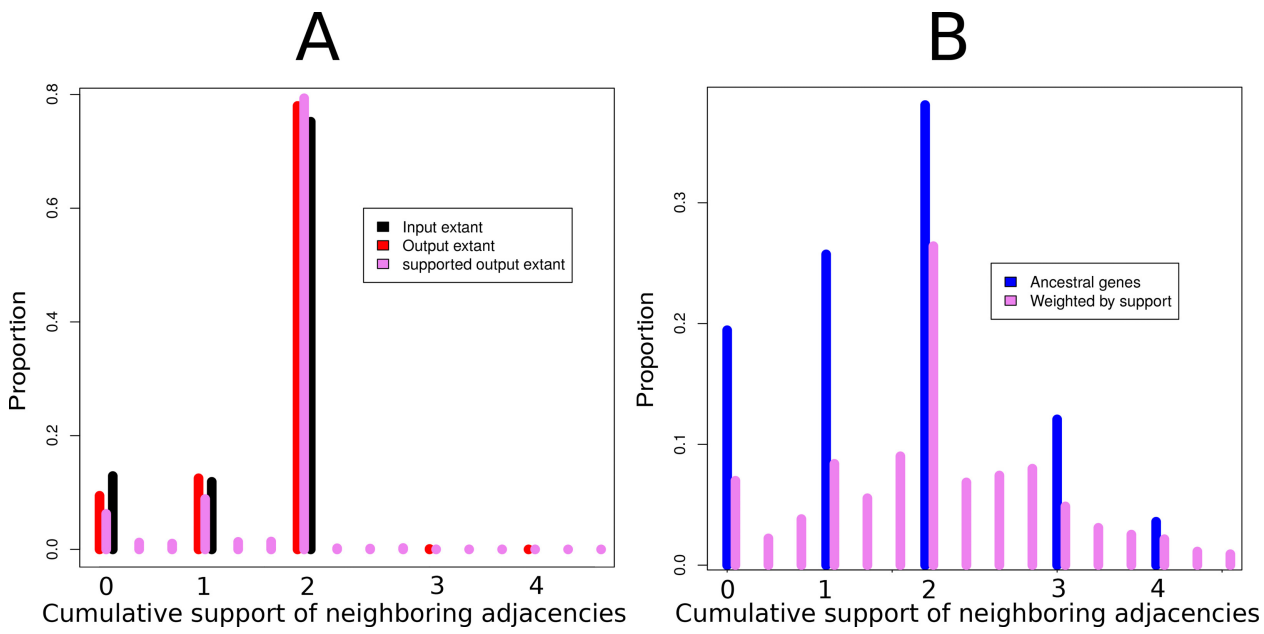

Figure 8 Weighted neighborhoods of extant (A) and ancestral genes (B). Distribution of the proportion of extant and ancestral genes with a given neighborhood weight in extant genomes before and after adjacency prediction, with or without support, for the data set on 39 mammals. The neighborhood weight of a gene is the sum of the supports of all adjacencies involving this gene. Continuous values are binned by intervals.

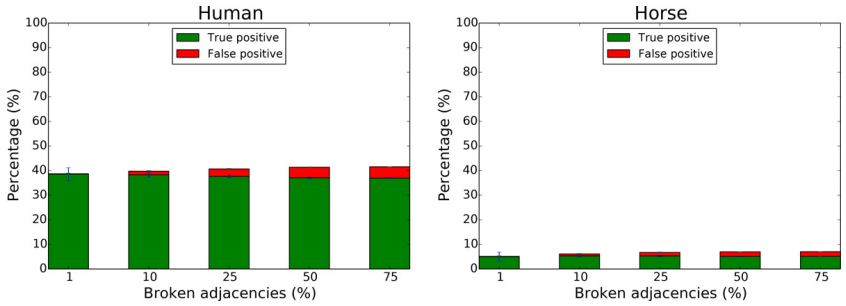

Figure 9 Capacity of ARt-DeCo to recover adjacencies after simulated breaks on human and horse genome.

we implement the additional possibility of scaffolding extant genomes in the same process, by handling equally ancestral and ancient genomes, with known and unknown parts in genome structures.

We demonstrate the efficiency of this approach on several eukaryote genomes data sets. It runs fast enough, proposes many additional supported adjacencies in extant genomes, and from several investigations we think we can state that such links are very likely to exist in reality. We are able to detect the less likely ones by assigning a support on ancestral and extant adjacencies by the same principle.

The main computational difference with the approach of Aganezov *et al*. [29] is that adjacencies are supposed to evolve independently. It has several notable consequences. The first one is the running time, because we switch from an NP-complete to a polynomial problem, and we are able to handle a large number of whole genomes. The second one is the shape of ancestral genomes. While methods modeling rearrangements [29] end up with *bona fide* genome structures, as linear arrangement of genes, our adjacency sets can be conflictual, both in ancestral and extant genomes. This means a gene can have more than two neighbors, unlike in real genomes. Whereas this can be seen as a serious drawback because genomes are not realistic, we would like to argue that it has several advantages, in addition to the running time. Indeed, the amount of conflicts can be a measure of uncertainty of the methods

and data. It has been remarked many times that data sets, and in particular gene trees, are far from perfect. But better gene trees produce ancestral genomes with less conflicts [34]. Conflicts can point at problems that don't necessarily concern the method itself, but give an overview of the quality of the data. This overview is lost if we force the data to fit in a linear structure. But if a linear ancestral genome is really needed, linearization techniques exist [38], even if we would argue for linearization techniques that also put into question the input data.

Some limitations would be still to overcome. For example we don't handle the orientation of the genes. This would be desirable to have a finer account of ancestral and extant genomes, and to have a better fit between the *a priori* probability of an adjacency (computed in an oriented mode) and the reconstructed adjacencies. It would not be conceptually much complicated because adjacencies can be considered between gene extremities instead of between genes. But it would result in a loss of sensitivity because inversions of a single gene, which seem to be frequent, would fall into a rearrangement signal, increasing the probability to lose the traces of neighborhoods. We leave this open for a future work.

Another perspective is to be able to question extant adjacencies given in the input. In our framework they have probability 1, but a scaffolding is not necessarily only giving an order to the contigs. It can be inserting a contig inside another, or cutting a chimeric contig because a better arrangement can be proposed. Assembly errors are often numerous, not only because of a lack of information, but also because of false information [39]. It could be done by re-assigning an *a priori* probability to each extant adjacency, and not only to the ones outside the contigs. Finally, following the idea introduced in RACA [26], it could be interesting to pair the predictions of ARt-DeCo with sequence information such as mate-pairs or even physical or optical maps in order to integrate both evolutionary signal and sequencing data.

Additional material

Additional File 1: Figure 10 ARt-DeCo dynamic programming scheme. Recurrence formulas used in ARt-DeCo to reconstruct adjacencies in ancestral and extant genomes.

Additional File 2: Figure 11 Effect of number of adjacencies breaks on adjacencies recover in function of base log value. On the left, graph represents the number of adjacencies recovery in function of multiplicative factor for the log base for 50 simulated breaks in each species of 7 tetrapods dataset. On the right, the same graph but with 1050 simulated breaks for each species. As we can see the histogram profile is quietly similar between these two experiments and the one with 550 simulated breaks (see Figure 2). In conclusion, Number of adjacencies breaks didn't impact the optimal value for the log base. (Values are available in Table 1.

Competing interests
The authors declare that they have no competing interests.

Authors' contributions
YA, AC, CC, VB, ET and SB conceived the method and the tests. YA and SB implemented ARt-DeCo and YA tested it on all data sets. CC assigned the support scores on all retrieved adjacencies. YA, AC, CC, VB, ET and SB wrote the paper.

Acknowledgements
This work and its publication is funded by the Agence Nationale pour la Recherche, Ancestrome project ANR-10-BINF-01-01.
ARt-DeCo analyses benefited from the Montpellier Bioinformatics Biodiversity platform services. We thank Yann Ponty for technical help for patching DeClone to integrate the scaffolding and the exploration of the solution space.
This article has been published as part of *BMC Genomics* Volume 16 Supplement 10, 2015: Proceedings of the 13th Annual Research in Computational Molecular Biology (RECOMB) Satellite Workshop on Comparative Genomics: Genomics. The full contents of the supplement are available online at http://www.biomedcentral.com/bmcgenomics/supplements/16/S10.

Authors' details
[1]Institut des Sciences de l'Évolution de Montpellier (ISE-M), Place Eugène Bataillon, Montpellier, 34095, France. [2]Institut de Biologie Computationnelle (IBC), Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier (LIRMM), Université Montpellier - CNRS, 161 rue Ada, Montpellier, 34090, France. [3]Laboratoire de Biométrie et Biologie Évolutive, LBBE, UMR CNRS 5558, University of Lyon 1, 43 boulevard du 11 novembre 1918, 69622 Villeurbanne, France. [4]Institut National de Recherche en Informatique et en Automatique (INRIA) Grenoble Rhône-Alpes, 655 avenue de l'Europe, 38330 Montbonnot, France. [5]Department of Mathematics, Simon Fraser University, 8888 University Drive, Burnaby, V5A 1S6, Canada.

Published: 2 October 2015

References

1. Raphael BJ, Volik S, Collins C, Pevzner PA: **Reconstructing tumor genome architectures.** *Bioinformatics* 2003, **19(Suppl. 2)**.
2. Fischer A, Vázquez-García I, Illingworth CJR, Mustonen V: **High-definition reconstruction of clonal composition in cancer.** *Cell Reports* 2014, **7(5)**:1740-1752.
3. McPherson A, Roth A, Ha G, Shah SP, Chauve C, Sahinalp SC: **Joint inference of genome structure and content in heterogeneous tumor samples.** *Research in Computational Molecular Biology Lecture Notes in Computer Science* 2015, **9029**:256-258.
4. Hurst L, Pál C, Lercher M: **The evolutionary dynamics of eukaryotic gene order.** *Nat Rev Genet* 2004, **5(4)**:299-310.
5. Swenson K, Arndt W, Tang J, Moret B: **Phylogenetic reconstruction from complete gene orders of whole genomes.** *Proceedings of the 6th Asia Pacific Bioinformatics Conference* 2008, 241-250.
6. Sankoff D: **Mechanisms of genome evolution: models and inference.** *Bulletin of the International Statistical Institute* 1989, **47**:461-475.
7. Ma J, Ratan A, Raney BJ, Suh BB, Zhang L, Miller W, Haussler D: **DUPCAR: Reconstructing Contiguous Ancestral Regions with Duplications.** *Journal of Computational Biology* 2008, **15(8)**:1007-1027.
8. Chauve C, Tannier E: **A methodological framework for the reconstruction of contiguous regions of ancestral genomes and its application to mammalian genomes.** *PLoS Computational Biology* 2008, **4(11)**:1000234.
9. Alekseyev MA, Pevzner PA: **Breakpoint graphs and ancestral genome reconstructions.** *Genome Research* 2009, **19(5)**:943-957.
10. Ma J: **A probabilistic framework for inferring ancestral genomic orders.** *IEEE International Conference on Bioinformatics and Biomedicine, BIBM* 2010, 179-184.
11. Zheng C, Sankoff D: **On the PATHGROUPS approach to rapid small phylogeny.** *BMC Bioinformatics* 2011, **12(Suppl. 1)**:4.
12. Bérard S, Gallien C, Boussau B, Szöllősi GJ, Daubin V, Tannier E: **Evolution of gene neighborhoods within reconciled phylogenies.** *Bioinformatics* 2012, **28(18)**:382-388.

13. Hu F, Lin Y, Tang J: **MLGO: phylogeny reconstruction and ancestral inference from gene-order data.** *BMC Bioinformatics* 2014, **15**:354-359.
14. Reddy TBK, Thomas AD, Stamatis D, Bertsch J, Isbandi M, Jansson J, Mallajosyula J, Pagani I, Lobos EA, Kyrpides NC: **The Genomes OnLine Database (GOLD) v.5: a metadata management system based on a four level (meta)genome project classification.** *Nucleic Acids Research* 2014, **43(D1)**:1099-1106[https://gold.jgi-psf.org/distribution].
15. Simpson JT, Wong K, Jackman SD, Schein JE, Jones SJM, Birol I: **ABySS: A parallel assembler for short read sequence data.** *Genome Research* 2009, **19(6)**:1117-1123.
16. Koren S, Treangen TJ, Pop M: **Bambus 2: Scaffolding metagenomes.** *Bioinformatics* 2011, **27(21)**:2964-2971.
17. Salmela L, Mäkinen V, Välimäki N, Ylinen J, Ukkonen E: **Fast scaffolding with small independent mixed integer programs.** *Bioinformatics* 2011, **27**:3259-3265.
18. Gao S, Sung WK, Nagarajan N: **Opera : Reconstructing optimal genomic scaffolds with high-throughput paired-end sequences.** *Journal of Computational Biology* 2011, **18(11)**:1681-1691.
19. Boetzer M, Henkel CV, Jansen HJ, Butler D, Pirovano W: **Scaffolding pre-assembled contigs using SSPACE.** *Bioinformatics* 2011, **27(4)**:578-579.
20. Gritsenko AA, Nijkamp JF, Reinders MJT, de Ridder D: **GRASS: A generic algorithm for scaffolding next-generation sequencing assemblies.** *Bioinformatics* 2012, **28(11)**:1429-1437.
21. Simpson JT, Durbin R: **Efficient de novo assembly of large genomes using compressed data structures.** *Genome Research* 2012, **22(3)**:549-556.
22. Luo R, Liu B, Xie Y, Li Z, Huang W, Yuan J, He G, Chen Y, Pan Q, Liu Y, Tang J, Wu G, Zhang H, Shi Y, Liu Y, Yu C, Wang B, Lu Y, Han C, Cheung DW, Yiu SM, Peng S, Xiaoqian Z, Liu G, Liao X, Li Y, Yang H, Wang J, Lam TW, Wang J: **SOAPdenovo2: an empirically improved memory-efficient short-read de novo assembler.** *GigaScience* 2012, **1(1)**:18.
23. Hunt M, Newbold C, Berriman M, Otto TD: **A comprehensive evaluation of assembly scaffolding tools.** *Genome Biology* 2014, **15(3)**:42.
24. Husemann P, Stoye J: **Phylogenetic comparative assembly.** *Algorithms for Molecular Biology* 2010, **5(1)**:3-14.
25. Rajaraman A, Tannier E, Chauve C: **FPSAC: Fast Phylogenetic Scaffolding of Ancient Contigs.** *Bioinformatics* 2013, **29(23)**:2987-2994.
26. Kim J, Larkin DM, Cai Q, Asan Zhang Y, Ge RL, Auvil L, Capitanu B, Zhang G, Lewin HA, Ma J: **Reference-assisted chromosome assembly. Proceedings of the National Academy of Sciences (PNAS).** 2013, **110(5)**:1785-1790.
27. Kolmogorov M, Raney B, Paten B, Pham S: **Ragout - A reference-assisted assembly tool for bacterial genomes.** *Bioinformatics* 2014, **30(12)**:302-309.
28. Lin Y, Nurk S, Pevzner PA: **What is the difference between the breakpoint graph and the de Bruijn graph?** *BMC Genomics* 2014, **15(Suppl. 6)**:6.
29. Aganezov S, Sitdykovaa N, Alekseyev MA, AGCConsortium: **Scaffold assembly based on genome rearrangement analysis.** *Computational Biology and Chemistry* 2015, **57**:46-53.
30. Cunningham F, Amode MR, Barrell D, Beal K, Billis K, Brent S, Carvalho-Silva D, Clapham P, Coates G, Fitzgerald S, Gil L, Girón CG, Gordon L, Hourlier T, Hunt SE, Janacek SH, Johnson N, Juettemann T, Kähäri AK, Keenan S, Martin FJ, Maurel T, McLaren W, Murphy DN, Nag R, Overduin B, Parker A, Patricio M, Perry E, Pignatelli M, Riat HS, Sheppard D, Taylor K, Thormann A, Vullo A, Wilder SP, Zadissa A, Aken BL, Birney E, Harrow J, Kinsella R, Muffato M, Ruffier M, Searle SMJ, Spudich G, Trevanion SJ, Yates A, Zerbino DR, Flicek P: **Ensembl 2015.** *Nucleic Acids Research* 2015, **43**:662-669.
31. Chauve C, Ponty Y, Zanetti JPP: **Evolution of genes neighborhood within reconciled phylogenies: an ensemble approach.** *To appear in BMC Bioinformatics* 2015 [http://biorxiv.org/content/early/2015/09/08/026310].
32. Biller P, Guéguen L, Tannier E: **Moments of genomes evolution by Double Cut-and-Join.** *BMC Bioinformatics* 2015, **16**.
33. Kasprzyk A: **BioMart: Driving a paradigm change in biological data management.** *Database* 2011, **2011**:049.
34. Boussau B, Szöllősi GJ, Duret L, Gouy M, Daubin V: **Genome-scale coestimation of species and gene trees.** *Genome Research* 2013, **23**:323-330.
35. Nouhati E, Semeria M, Lafond M, Seguin J, Boussau B, Guéguen L, El-Mabrouk N, Tannier E: **Efficient gene tree correction guided by species and synteny evolution.** 2015 [https://hal.archives-ouvertes.fr/hal-01162963].
36. Rajaraman A, Chauve C, Ponty Y: **Assessing the robustness of parsimonious predictions for gene neighborhoods from reconciled phylogenies.** *Lecture Notes in Computer Science* 2015, **9096**:260-271.
37. Luhmann N, Chauve C, Stoye J, Wittler R: **Scaffolding of ancient contigs and ancestral reconstruction in a phylogenetic framework.** *Proceedings of Brazilian Symposium on Bioinformatics Lecture Notes in Computer Science* 2014, **8826**:135-143.
38. Maňuch J, Patterson M, Wittler R, Chauve C, Tannier E: **Linearization of ancestral multichromosomal genomes.** *BMC Bioinformatics* 2012, **13(Suppl. 19)**:11.
39. Denton JF, Lugo-Martinez J, Tucker AE, Schrider DR, Warren WC, Hahn MW: **Extensive error in the number of genes inferred from draft genome assemblies.** *PLoS Computational Biology* 2014, **10(2)**:1003998.

doi:10.1186/1471-2164-16-S10-S11
Cite this article as: Anselmetti *et al.*: Ancestral gene synteny reconstruction improves extant species scaffolding. *BMC Genomics* 2015 **16**(Suppl 10):S11.

Article ALPHA

Bérard *et al. BMC Bioinformatics* (2016) 17:30
DOI 10.1186/s12859-015-0869-5

BMC Bioinformatics

METHODOLOGY ARTICLE Open Access

Aligning the unalignable: bacteriophage whole genome alignments

Sèverine Bérard[1,2], Annie Chateau[2,3], Nicolas Pompidor[2], Paul Guertin[4,5], Anne Bergeron[4] and Krister M. Swenson[2,3*]

Abstract

Background: In recent years, many studies focused on the description and comparison of large sets of related bacteriophage genomes. Due to the peculiar mosaic structure of these genomes, few informative approaches for comparing whole genomes exist: dot plots diagrams give a mostly qualitative assessment of the similarity/dissimilarity between two or more genomes, and clustering techniques are used to classify genomes. Multiple alignments are conspicuously absent from this scene. Indeed, whole genome aligners interpret lack of similarity between sequences as an indication of rearrangements, insertions, or losses. This behavior makes them ill-prepared to align bacteriophage genomes, where even closely related strains can accomplish the same biological function with highly dissimilar sequences.

Results: In this paper, we propose a multiple alignment strategy that exploits functional collinearity shared by related strains of bacteriophages, and uses partial orders to capture mosaicism of sets of genomes. As classical alignments do, the computed alignments can be used to predict that genes have the same biological function, even in the absence of detectable similarity. The Alpha aligner implements these ideas in visual interactive displays, and is used to compute several examples of alignments of *Staphylococcus aureus* and *Mycobacterium* bacteriophages, involving up to 29 genomes. Using these datasets, we prove that Alpha alignments are at least as good as those computed by standard aligners. Comparison with the progressiveMauve aligner – which implements a partial order strategy, but whose alignments are linearized – shows a greatly improved interactive graphic display, while avoiding misalignments.

Conclusions: Multiple alignments of whole bacteriophage genomes work, and will become an important conceptual and visual tool in comparative genomics of sets of related strains.
A python implementation of Alpha, along with installation instructions for Ubuntu and OSX, is available on bitbucket (https://bitbucket.org/thekswenson/alpha).

Keywords: Bacteriophages, Whole genome alignments, Comparative genomics, Partial orders

Background

The most abundant, and probably the most diverse, biological entities are bacteriophages, the viruses that infect bacteria. Helped by recent advances in sequencing, comparative studies [1–3] use dozens – even hundreds – of genomes from bacteriophages that infect single or related bacteria species. Using dot plots and clustering techniques, these studies produce meaningful clusters that share significant similarity (> 50 %).

In order to explore the relations between bacteriophages in the same cluster, it seems natural to turn to whole genome multiple alignments. One of the main features of whole genome aligners is that they take into account genome rearrangement events that scramble the order of large segments of chromosomes (see [4] for a review). All of these approaches are based on the principle that sequence similarity is a good predictor of functional similarity. And it is, most of the time. One notable exception are bacteriophage genomes, in which similar biological function may be encoded by dissimilar sequences – sequences with no detectable similarity, either as nucleotide sequences, or as amino acid sequences – encoding different protein folds, rendering

*Correspondence: swenson@lirmm.fr
[2]LIRMM, CNRS - Univ. Montpellier, 161 rue Ada, 34392 Montpellier, France
[3]IBC Institut de Biologie Computationnelle, Montpellier, France
Full list of author information is available at the end of the article

traditional multiple sequence aligners mostly useless [5–7].

In this paper, our goal is to construct biologically meaningful multiple alignments of whole bacteriophage genomes from the *Siphoviridae*, the largest family of *tailed* bacteriophages [8]. In order to achieve this, we exploit unique structural properties of these genomes. The main one is that even loosely related tailed bacteriophages are often *functionally collinear*, meaning that different functionalities mostly follow the same order on the genomic sequence, which is, up to a circular permutation: lysogeny, DNA assembly, head morphology and DNA packaging, tail assembly, and lysis [9]. The second one is that genome size is constrained by the fact that it must fit into a capsid whose shape, thus volume, is geometrically determined by a handful of genes. This size constraint implies that segment duplication is a rare event within a genome, and when it occurs, duplicated sequences are short. Last, but not least, bacteriophage genomes are characterized by an *"unusually high degree of horizontal genetic exchange in their evolution"* [10], resulting in large sequences – up to thousands of base pairs – that are exact or almost exact matches between different strains.

As an example of this last feature, Fig. 1 compares two segments of *S. aureus* bacteriophages 88 and 92 (see Table 1 for all accession numbers of bacteriophages discussed in the paper). The figure is composed of 5 columns, three of which corresponding to exact matches of length 259, 35 and 49, and the other two columns containing large segments occupying the same position but without any 'detectable' similarity, meaning neither the nucleotide nor the translated sequences produce any significant BLAST hit.

The mosaic patterns exhibited by the comparison of two bacteriophages, as illustrated in Fig. 1, have given rise to the *modular* theory of phage genome organization [5], which postulates that biological functions are grouped into modules whose order is mostly conserved along the genomic sequence. Each module has *variants* that perform the same function, possibly encoded by dissimilar sequences. The fact that two variants are *aligned* in the pairwise comparison may allow the transfer of functional annotation between two collinear phages.

While it is easy to do a pairwise comparison of genomes, upgrading the comparison to multiple genomes is not simple. The standard approaches used by multiple sequence aligners often start by identifying *anchors*, that are similar segments of significant length shared by all genomes, and then align the sequences between the anchors. Anchors exist in sets of related bacteriophages, but may be very short or very far apart, and the sequences between the anchors may fail to align properly.

The theory behind our framework is based on *partial order alignment graphs* [4, 11–13] which were initially developed for standard multiple sequence alignments. Most applications of these graphs require, in the last phase of the alignment, a linearization of the graph. However, an interesting suggestion appearing in [13] is to skip this last step and work directly with the partial order: this is exactly what is needed for bacteriophage genomes, but apparently the approach has yet to be applied in this context.

Among the multiple sequence aligners, progressiveMauve [14] is one of the few that recognizes the need to identify 'local' anchors, that are shared by a subset of the target genomes. Unfortunately, Mauve alignments are linearized, blurring the combinatorial properties of the partial order.

Our evolutionary model includes typical mutations characterizing sequence evolution such as substitutions and indels, gains and losses of functions, and recombinations. For sets of genomes that have evolved under these conditions, we will show that it is possible to identify functionally related sequences even when they lack similarity. We also detect large rearrangements events that contradict the functional collinearity hypothesis, such as gene transpositions, and duplications.

In this paper, we report the conception and implementation of Alpha (**Al**ignments of bacterio**pha**ge genomes), the first aligner specifically designed for whole bacteriophage genomes. With the help of partial order structures, Alpha captures the unique mosaic structure of bacteriophage genomes, and provides an interactive graphical interface with the generated multiple alignment. We also give a detailed comparison between Alpha and progressiveMauve [14] alignments.

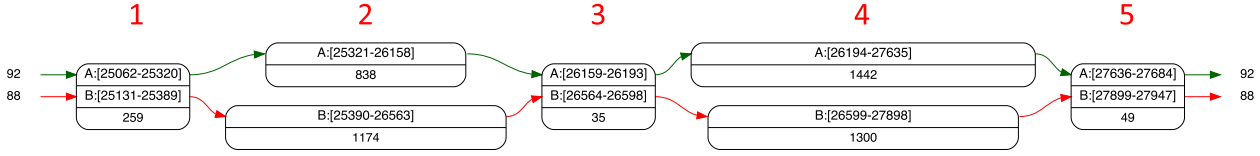

Fig. 1 Local alignment of *S. aureus* bacteriophages 88 and 92. The first, third and fifth columns are exact matches of length 259, 35 and 49. The sequences in the second and fourth columns do not have detectable similarity, but, given their similar position along the genome, putatively encode similar biological functions. Indeed, the fourth column encodes two variants of the *integrase* gene

Table 1 Accession numbers and datasets

Name	Accession	Start	End	Dataset
U2	AY500152	75	45955	Myco6
Alvin	KP027205	75	45163	Myco6
DD5	EU744252	75	46027	Myco6
BillKnuckles	JN699000	75	45715	Myco6
Perseus	JN572689	75	47564	Myco6
Dreamboat	JN660814	75	44313	Myco6
U2	AY500152	1	33000	Myco29
Doom	JN153085	1	33000	Myco29
Alvin	KP027205	1	33000	Myco29
BXB1	AF271693	1	33000	Myco29
Solon	EU826470	1	33000	Myco29
Bethlehem	AY500153	1	33000	Myco29
DD5	EU744252	1	33000	Myco29
Pinto	KJ690250	1	33000	Myco29
BillKnuckles	JN699000	1	33000	Myco29
KBG	EU744248	1	33000	Myco29
Lesedi	JF937100	1	33000	Myco29
Museum	JF937103	1	33000	Myco29
Violet	JN687951	1	33000	Myco29
Kugel	JN699016	1	33000	Myco29
MrGordo	JN020140	1	33000	Myco29
KSSJEB	JF937110	1	33000	Myco29
Switzer	JF937108	1	33000	Myco29
Perseus	JN572689	1	33000	Myco29
Dreamboat	JN660814	1	33000	Myco29
Seabiscuit	KJ194585	1	33000	Myco29
Trouble	KF024724	1	33000	Myco29
BPBiebs31	JF957057	1	33000	Myco29
Wheeler	KF416340	1	33000	Myco29
Graduation	KF560331	1	33000	Myco29
JC27	JF937099	1	33000	Myco29
Thor	KP027204	1	33000	Myco29
Aeneas	JQ809703	1	33000	Myco29
SarFire	KF024726	1	33000	Myco29
SkiPole	GU247132	1	33000	Myco29
phiETA3	NC_008799	1	43282	Staph4
phiNM1	NC_008583	1	43128	Staph4
phiNM2	DQ530360	1	43145	Staph4
B236	KP893290	1	43228	Staph4
85	AY954953	1	44283	Staph6
88	AY954966	1	43231	Staph6
92	AY954967	1	42431	Staph6
29	AY954964	1	42802	Staph6
187	AY954950	1	39620	Staph6
53	AY954952	1	43883	Staph6

Methods

Partial order alignment graphs and functional collinearity

This section draws heavily on multiple Whole Genome Alignment (WGA) tools and definitions. However, since we have the goal of identifying functional analogs, some classical notions of this field will have a somewhat different meaning: in these cases, we try to be precise and to underline the differences.

A *match* M between two genome sequences G and H will be denoted $M = \{G[s..t], H[u..v]\}$, where s and u are the start positions of the match in the respective genomes, and $t - s = v - u$; it asserts the equalities $G[s] = H[u], \ldots, G[t] = H[v]$.

We next formalize the notion of homologous positions, defining what constitutes a 'column' of a multiple alignment, along the lines of J. Kececioglu's original paper [11].

Let $\mathcal{M}$ be a set of matches on genomes $\mathcal{G} = \{G_1, G_2, \ldots, G_n\}$ and let p_{ij} denote position j of genome G_i. Two positions p_{ij} and p_{kl} are *equivalent* if there is a match in $\mathcal{M}$ that asserts that $G_i[j] = G_k[l]$, or if there is a sequence of matches that links $G_i[j]$ to $G_k[l]$ through a sequence of such assertions. The equivalence class of p_{ij} is denoted $[\![p_{ij}]\!]$, and we will refer to it as a *column*. The *support* of a column $sp[\![p_{ij}]\!]$ is the set of genomes that have a position in the column $[\![p_{ij}]\!]$.

Note that there is no requirement that a column span the complete set $\mathcal{G}$, and it usually does not in bacteriophage genomes. The notion of support generalizes the notion of *anchor* that is used in WGA, which corresponds to columns whose support is equal to $\mathcal{G}$.

The *column graph* is obtained by linking the different columns according to their order in each genome. Formally, it is the directed (multi)graph whose vertices are the columns $[\![p_{ij}]\!]$, and there are edges from $[\![p_{ij}]\!]$ to $[\![p_{kl}]\!]$ for all genomes G_g that have positions p_{gs} in $[\![p_{ij}]\!]$ and $p_{g(s+1)}$ in $[\![p_{kl}]\!]$. An example of the construction is given in Fig. 2, part (B).

Definition 1. *Let $\mathcal{M}$ be a set of matches between genomes $\mathcal{G} = \{G_1, G_2, \ldots, G_n\}$. The set $\mathcal{G}$ is* functionally collinear with respect to *$\mathcal{M}$ if its column graph is acyclic.*

In most WGA systems, collinearity is meant to represent sequences that 'align well'. Classical collinearity is broken by pairs of dissimilar sequences, and is an indication of gene rearrangements, gains or losses, in most higher organisms. Functional collinearity, on the other hand, excludes rearrangements, but allows dissimilar sequences to be compared. Interestingly, both concepts have the same formal definition, but the use of the column graph is quite different from one application to the other.

A necessary condition for a column graph to be functionally collinear is that each column have at most one

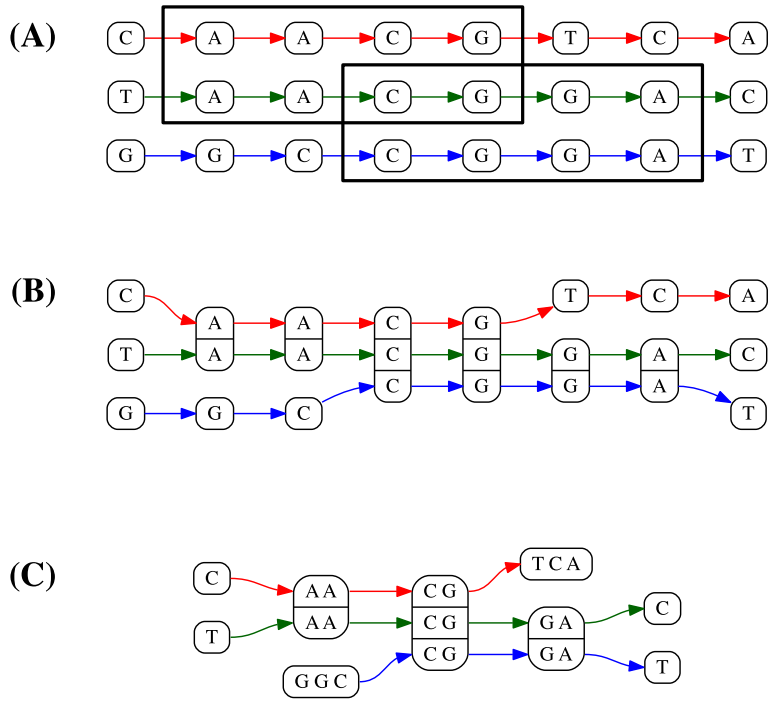

Fig. 2 The column graph and the alignment graph. **a** A set of two exact matches on three genomes; the total length of matches is 8. **b** The matches define columns for which all positions contain the same nucleotide. The column graph is obtained by adding edges corresponding to consecutive positions in the genomes. **c** Consecutive columns with the same support are merged to form the alignment graph, whose vertices are exact alignments

position in a given genome, since the presence of two or more positions would generate a cycle. We have therefore the following definition:

Definition 2. *A set $\mathcal{M}$ of matches is said to be valid if, for each column $[\![p_{ij}]\!]$, both $[\![p_{ij}]\!]$ and $sp[\![p_{ij}]\!]$ have the same number of elements.*

The most likely sources of invalid sets of matches are duplicated segments, and tandem repeats with a variable number of repeats, which may cause overlapping matches. Most WGA deal with these invalid matches by eliminating matches that are not unique in a genome, and by trimming overlapping matches that result from tandem repeats. Given the simple data structure that we use for the alignment graph – equivalence classes – we eliminate duplicated segments, and trim overlapping matches by post-processing the columns: any column that has more than one position for a genome is split into singletons.

Allowing the length of matches to be very small may cause misalignments, and may also create cycles in the column graph, due to random small transpositions. It is therefore wise to have a minimal length m for matches. In the Results section, we will discuss how the value of m is set, and how it can be changed.

When the column graph has no cycles, we use a condensed version, called the *alignment graph*, in which consecutive vertices with the same support are merged, as in Fig. 2, part (C).

Construction of the alignment graph

Given a set of genomes $\mathcal{G} = \{G_1, G_2, \ldots, G_n\}$ we use GenomeTools [15], a suffix-array based program, to get a set of exact matches of size at least m. The parameter m is normally chosen as the minimum length such that there is no cycle in the graph. It is set by default at 15, but can be changed to any desired value. Large values of m are sometimes useful in initial explorations of a set of bacteriophages and small values can be used to refine alignments.

The main steps of the algorithm underlying Alpha are described in Algorithm 1. It takes as input a set of matches of minimum length m, constructs the column graph, and the alignment graph when possible. After initializing the graph, it uses the Union-Find algorithm [16] to construct the columns. This step runs in $\mathcal{O}(\alpha(M_\ell)M_\ell)$, where M_ℓ is the total length of matches, and α is the inverse of the Ackermann function [17].

Algorithm 1 Construction of the column and alignment graphs

```
for each genome G do                                          ▷ All columns initialized as singletons
    for each position p in G do
        ⟦p⟧ ← {p}
    end for
end for
for each match M of 𝓜 do                                      ▷ Construction of the columns
    for each assertion G_i[j] = G_k[l] of M do
        Merge classes ⟦p_ij⟧ and ⟦p_kl⟧
    end for
end for
for each class ⟦p_ij⟧ do                                      ▷ Invalid columns are split
    if |⟦p_ij⟧| ≠ |sp⟦p_ij⟧| then
        Split class ⟦p_ij⟧ into singletons
    end if
end for
for each genome G do                                          ▷ Add edges
    for each pair of consecutive positions p_j and p_(j+1) in G do
        Put a directed edge from ⟦p_j⟧ to ⟦p_(j+1)⟧
    end for
end for
if the graph has no cycles then                               ▷ Construct the alignment graph
    Merge consecutive columns with the same support
end if
```

The third step visits all classes, determining whether they are valid. For each class, this determination can be made in $\mathcal{O}(n)$ steps, where n is the number of genomes, yielding a complexity of $\mathcal{O}(nN_\ell)$ for splitting all the non-valid classes, where N_ℓ is the total length of genomes. Note that at most N_ℓ new classes can be created by the splitting process.

In the fourth step, edges between consecutive columns are added to the graph, and it is checked for the presence of cycles that would indicate non-collinearity. This can be done in time proportional to N_ℓ. Assuming the genomes are collinear, consecutive columns with the same support are then merged to give the alignment graph.

The total complexity is thus $\mathcal{O}(\alpha(M_\ell)M_\ell + nN_\ell)$. We will see in the Results section that the quantities M_ℓ and nN_ℓ vary greatly depending on the data.

From the alignment graph to gapless alignments

The vertices of an alignment graph are exact alignments, since all rows are equal. Regions that do not contain exact matches of length at least m remain unaligned. In order to increase the compactness of whole genome alignments, we *contract* the vertices further with the following requirement:

Definition 3. *The pair of vertices* (U, V) *is* contractible *if* U *precedes* V *in the partial order,* U *and* V *have the same support, the support of every vertex between* U *and* V *is included in* $sp(V)$*, and the lengths, in base pairs, of all segments that span from* U *to* V *are the same.*

A *contracted vertex* is obtained by merging into a single vertex all vertices between a pair of contractible vertices U and V, including U and V. Contracted vertices are gapless alignments, since all sequences in them have the same length. Without upper bounds to the length of contracted vertices, there is a small chance that some of these alignments have no biological foundation, since insertions and deletions could conspire in producing sequences with the same length, but low similarity. As we will see in the Results section, bacteriophage genomes are well behaved in this regard.

Figure 3 shows an example of a contracted vertex together with its expanded graph. All vertices display the coordinates of aligned segments, except for narrow ones, together with the common length of these segments. In addition, a contracted vertex also displays the *percentage of identity*, which is the percentage of columns that have the same conserved nucleotide in each genome of its support. Since the contracted version of the alignment graph turned out to be the most useful in practice (see the Results section), we will refer to it as the alignment graph, and use the term 'expanded alignment graph' when the full version is required.

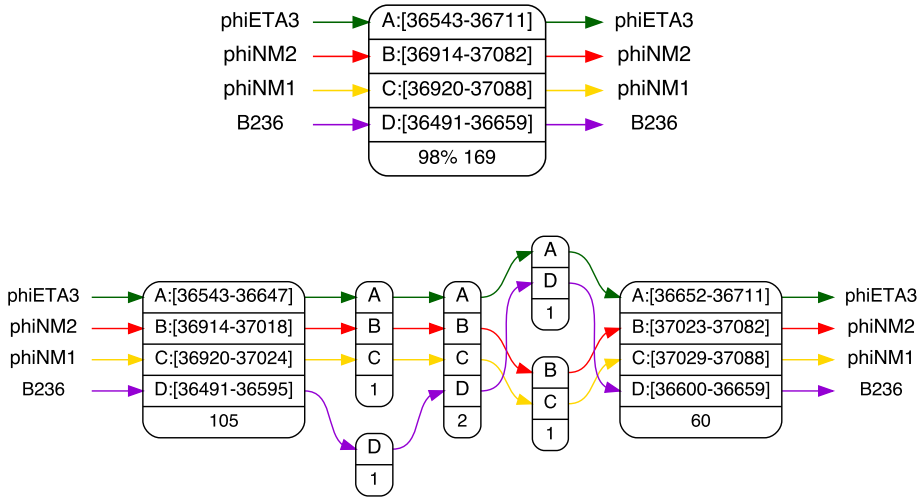

Fig. 3 Contracted vertices. A example of a contracted vertex (top) and its expanded graph (bottom) in the alignment of four *S. aureus* bacteriophages. The length of the vertex is 169 nucleotides, and the percentage of identity is 98 %. In the expanded graph, there are two single nucleotide mutations that account for the 98 % identity

Results and discussion

In this section, we first present examples of multiple alignments and what can be deduced from them. We next assess the validity of the gapless alignments inferred by Alpha by submitting the aligned sequences to three standard aligners. Finally, we compare the results of Alpha and progressiveMauve [14] on whole genome alignments.

Alpha is an interactive tool that allows the manipulation and the visualization of whole genome alignment graphs with hundreds of vertices, involving dozens of species: we can only hope to give a glimpse of the possibilities. Alpha's graph layouts are powered by the open source graph visualization software Graphviz [18].

The alignment graph

The Alpha aligner, in its most basic mode, takes as input a file containing genomes in Fasta format, and produces an *alignment graph* when the genomes have no large rearranged segments. The vertices of the alignment graph are gapless multiple alignments inferred by Alpha. The vertices contain the positions of the aligned segments in each genome. They also display the length of the alignment and the percentage of columns that have the same nucleotide in each genome – for readability, vertices smaller than 20 bp may be masked. Alignments are connected by color-coded arrows, one color for each genome.

Figure 4 shows an example of a local alignment of four *S. aureus* bacteriophages, in which two deletions are easily identified. Dotted arrows replace vertices that span less than 20 bp, implying that phage B236 lacks a group of functionalities that spans over a thousand bp, and phage phiETA3, one that spans over 500 bp.

Bacteriophage genomes have been sequenced for decades, and since they can adopt different linear and circular configurations in their complex life cycle, there is no universal consensus as to *where the sequence begins*. Functional collinearity is biologically defined up to a circular permutation of the sequences, but bacteriophage

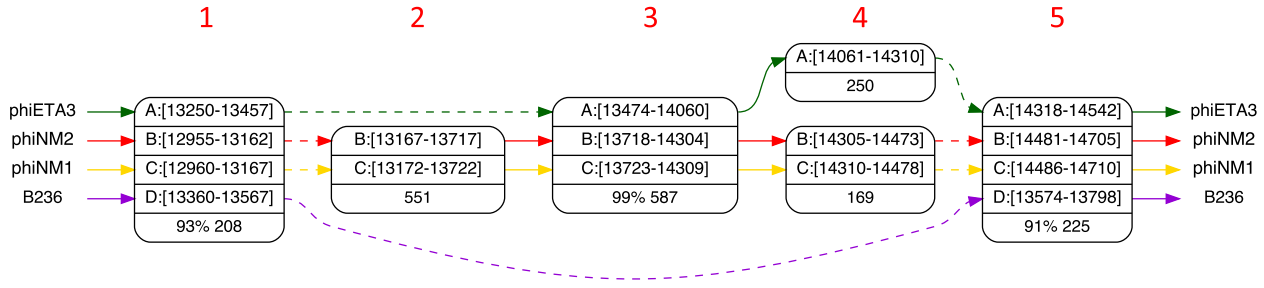

Fig. 4 A local alignment of four *S. aureus* bacteriophages. In column 3, three phages are in a 587 bp gapless alignment, with 99 % identical columns. A major deletion in phage B236 spans columns 2, 3, and 4, and the corresponding arrow is dotted to reflect the fact that some basepairs are not shown, 6 bp in this case

sequences in databases are linear. If a set of bacteriophage genomes was obtained from different projects, or different laboratories, it may be necessary to synchronize them using a simple procedure that looks for the largest similar sequence shared by all genomes, and sets the beginning there. We call this process *normalization*, and Alpha checks its input to decide whether normalization is needed.

Figure 5 shows a group of *S. aureus* bacteriophages whose sequences have been normalized: positions of bacteriophage phiNM1 are offset by more than 16 500 bp with respect to the other genomes, while maintaining collinearity with them. In this figure the second and fourth columns identify two pairs of large sequences. These pairs share similar *loci* within the alignment but they lack detectable similarity, and they are predicted to be variants of the same module. However, the second column splits sequences {A, D} from {B, C}, while the fourth splits sequences {A, B} from {C, D}, illustrating the very peculiar behavior of horizontal transfer in bacteriophages.

The anchor view

Anchors are alignments that span the whole set of n genomes under study. They are maximal exact matches, in the sense of Hohl et al. [19], but they can be as short as 1 base pair. Each anchor is constructed as the intersection of at least $n - 1$ exact pairwise alignments of length at least m, thus the reliability of anchors increases with both the number of genomes, and the parameter m. The ordered set of anchors forms the *backbone* of a set of bacteriophage genomes and captures their *common core*, as defined in Mosaic [20].

The anchor view of Alpha presents a sequence of anchors for the whole genomes, or for selected regions: it is the normal starting point to explore a set of genomes. Figure 6 shows an example of an anchor view for a set of 29 mycobacteriophages computed with $m = 175$. There are not many anchors for such a large value of m, but they are very well supported because each anchor is defined by at least 29 exact matches whose length is at least 175 bp. Since anchors are articulation points of the alignment graphs – removing them disconnects the graph – each group of sequences spanning from one anchor to the next can be explored separately, as we show in Fig. 7.

Once a pair of anchors is selected in the anchor view, an alignment spanning from one to the other is generated. It is possible to ask the aligner to align the sequences between them, and a new value of m is automatically computed, generally much smaller – or set to any desired value. In Fig. 7, for example, the value of m is 15.

Assessing the validity of alpha gapless alignments

In order to assess the gapless alignments produced by Alpha, we ran it on three datasets. Table 2 presents the principal characteristics of these datasets such as N_ℓ, the total length of genomes, the value of m used in the computation, and M_ℓ, the total length of matches.

Dataset Staph6 contains 6 complete *S. aureus* bacteriophage genomes: 29, 53, 85, 88, 92 and 187. Dataset Myco29 contains 29 mycobacteriophage genomes from cluster A1 of the Actinobacteriophage Database (http://phagesdb.org). Myco6 contains bacteriophage genomes {U2, Alvin, DD5, BillKnuckles, Perseus, Dreamboat}, a subset of Myco29 chosen for diversity within cluster A1. Some of the mycobacteriophage genomes were trimmed in order to automatically run the experiments with small values of m; transpositions at the end of the sequences cause Alpha to increase the value of m (see Table 1 for details).

Alpha computed 491 gapless alignments of 2 or more sequences on these datasets. All 491 alignments were re-aligned using Clustal Omega [21], T-Coffee [22] and Muscle [23], using the default values for DNA alignments, and we report the number of alignments that include gaps. For each dataset, Table 3 gives the number of vertices in the expanded and contracted alignment graphs, the number of vertices containing at least two sequences, the number of contracted vertices, and the number of gapped alignments obtained by the three aligners.

All four aligners agree on 479 of the $491 = 78 + 154 + 259$ gapless alignments proposed by Alpha. The remaining 12 were contested by one or more of the aligners, for a total of 18 gapped alignments. They cover 8 different regions of the genomes since, in four cases, aligners proposed gapped alignments for the same region in both the Myco6 and Myco29 datasets.

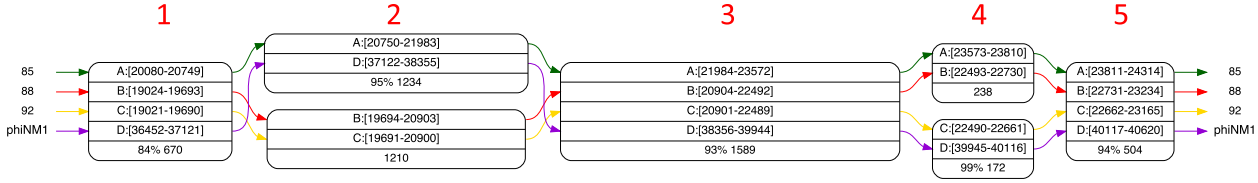

Fig. 5 Modules and variants. Local alignment of four *S. aureus* bacteriophages clearly showing modules and variants. Notice that sequences with the same variant in the second column are switched in the fourth

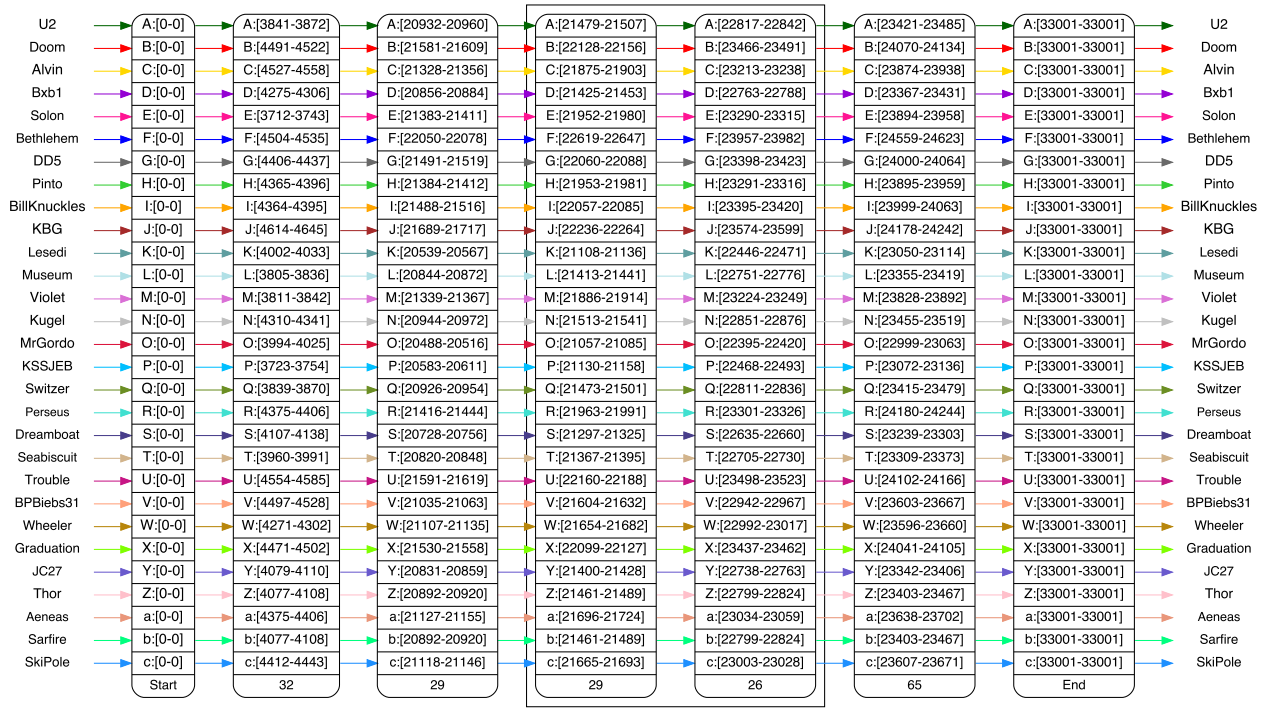

Fig. 6 The anchor view. The alignment of 29 *Mycobacterium* bacteriophage genomes is a challenge to visual approaches. Here we use a large value of $m = 175$ to display a sequence of anchors, or backbone, in a single screen. The user can then zoom quickly to a specific region, and detect, as Fig. 7 shows, interesting subsets of genomes. Boxes group pairs of anchors bounding a gapless alignment

Judging whether nucleotide alignments are in fact gapless is a delicate task. When proteins sequences were available, we used the gapless protein alignments to reject the corresponding gapped nucleotide alignment. This was done by using tblastx with sequences that do not share the same gap patterns – all the gapped alignments showed only two different gap patterns – and by confirming annotations using blastx. Using this method, we could rule out 12 of the 18 alternative gapped alignments, as Table 4 shows, for Regions 1, 2, 3, 4, and 8. All sequences and alignments are available in the Additional file 1.

For Region 5, the three aligners proposed three different gapped alignments, when aligning the 6 genomes of dataset Myco6, but they all switched to gapless alignments when aligning the same region in the 29 genomes of dataset Myco29, which contains Myco6. Using the principle that alignments with more sequences should be more accurate, those three gapped alignments were also ruled out.

Of the remaining three gapped alignments, one was proposed by Muscle, while Alpha, Clustal Omega and T-Coffee proposed gapless alignments (Region 7). Given the overall poor performance of Muscle, we chose the rule of the majority. Finally, for Region 6, there was a tie between the four aligners, and there was no decision.

As the results of Table 4 clearly indicate, Alpha gapless alignments were confirmed in all cases except for the tie in Region 6, where Alpha and Clustal Omega – the two best gapless aligners – predict a gapless alignment against T-Coffee and Muscle, who have the most confirmed misalignments. This is a welcomed and surprising result, since the only parameter of Alpha is m, the minimal length of exact matches: it does not rely on thresholds, and does not maximize any score. Instead, it relies on the transitive properties of the equality relation to provide reliable anchors, and on the unique constraint imposed on bacteriophages who, before traveling, must pack their whole genome in a small suitcase.

Comparison with Mauve alignments

In the preceding section, we showed that Alpha alignments are sound, in the sense that they predict biological meaningful similarities. The next question is to evaluate to what extent Alpha captures all meaningful similarities.

In order to do this, we compared Alpha with progressiveMauve [14], since it is one of the few aligners that explicitly computes the partial order underlying collinear blocks, storing this information in the `.backbone` file generated during an alignment. As Fig. 8 shows, Mauve alignments are displayed in a linear way, using colors to show which subsets of segments are in an alignment. With

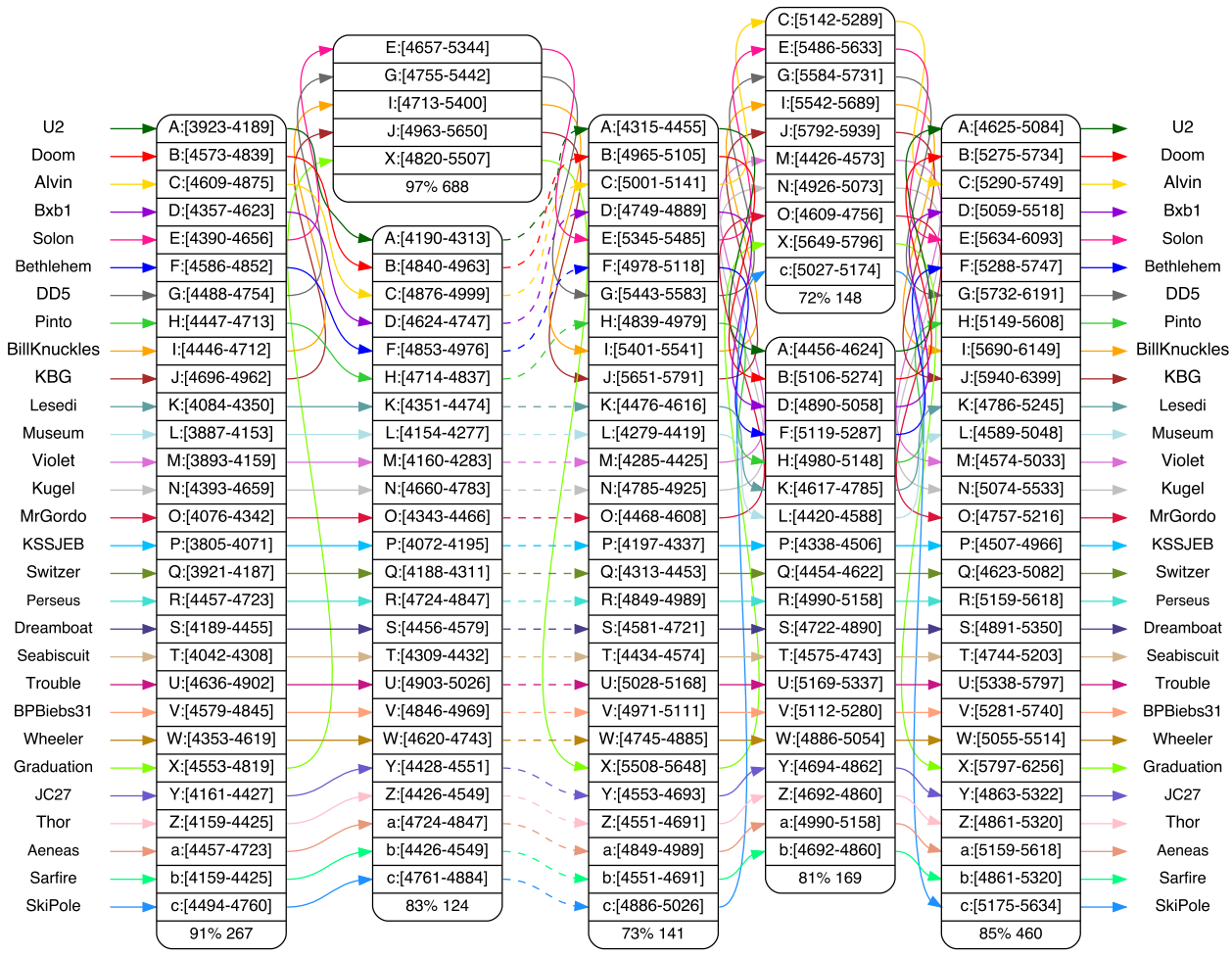

Fig. 7 Zooming from the anchor view. Local alignment of a set of 29 *Mycobacterium* bacteriophages, obtained by zooming in from the anchor view of Fig. 6. Working with a large number of phages allows us to quickly identify subsets of phages that have distinct patterns, such as phages A, C and E (U2, Alvin and Solon) in this example

more than a handful of genomes, such as the Myco29 dataset, this type of visualization quickly becomes impractical (see Additional file 2: Figure S1 that shows the Mauve alignment of the region spanned in Fig. 7).

Comparing partial order alignments stemming from two different aligners turned out to be a daunting task: an alignment proposed by one aligner can be broken into several alignments by the other, since alignments may involve different subsets of the input genomes. The most practical approach was to compare only alignments that included all input genomes from both aligners.

We used a dataset of four *S. aureus* phages {phiETA3, phiNM2, phiNM1, B236}, with default parameters for progressiveMauve, and interactively setting m between

Table 2 Parameters of the three datasets. Column n is the number of genomes; N_ℓ is the total length of genomes; m is the minimal match length; M_ℓ is the total length of matches

Name	Hosts	n	N_ℓ	m	M_ℓ
Staph6	*S. aureus*	6	256 250	36	171 295
Myco6	*Mycobacterium*	6	274 292	25	407 536
Myco29	*Mycobacterium*	29	957 000	31	8 634 944

Table 3 Statistics on the alignments of the three datasets

Dataset:	Staph6	Myco6	Myco29
Total number of vertices before contraction	1565	5327	10897
Total number of vertices after contraction	323	729	2324
Vertices aligning at least two sequences	202	425	964
Contracted vertices	78	154	259
Clustal Omega gapped alignments	0	1	0
T-Coffee gapped alignments	1	2	2
Muscle gapped alignments	2	5	5

Table 4 Validating gapless alignments: each column contains the number of predicted gapped alignments by each aligner. The number of mis-alignments is given in the last lines

Region's start and end positions	Alpha	Clustal O	T-Coffee	Muscle	Gapless	Method
1. U2[1869-2931]	0	0	0	2	Yes	tblastx
2. U2[6932-7965]	0	0	0	2	Yes	tblastx
3. U2[10982-11518]	0	0	2	2	Yes	tblastx
4. U2[11521-12378]	0	0	0	2	Yes	tblastx
5. U2[17171-18327]	0	1	1	1	Yes	switch
6. U2[76-263]	0	0	1	1	?	tie
7. 85[23847-24292]	0	0	0	1	Yes	majority
8. 85[35337-35536]	0	0	1	1	Yes	tblastx
Error(s) if Alpha is right for Region 6	0	1	5	12		
Error(s) if Alpha is wrong for Region 6	1	2	4	11		

13 and 21 for Alpha. All alignments from both aligners were identified by their phiETA3 start and end positions, and further refined by whether they were identified by Mauve only, Alpha only, or both. This produced 70 blocks, available in the Additional file 3, covering 13706 (31.7 %) of the 43282 bp of the phiETA3 genome. Of the 70 blocks, 42 were quite short, from 1 to 96 bp, while the length of the remaining 28 ranged from 123 to 3864 bp.

All blocks predicted by both aligners were considered as valid alignments. Furthermore, short blocks (less than 100 bp) that were predicted by only one aligner – mostly by Mauve – were considered as valid: they averaged 24

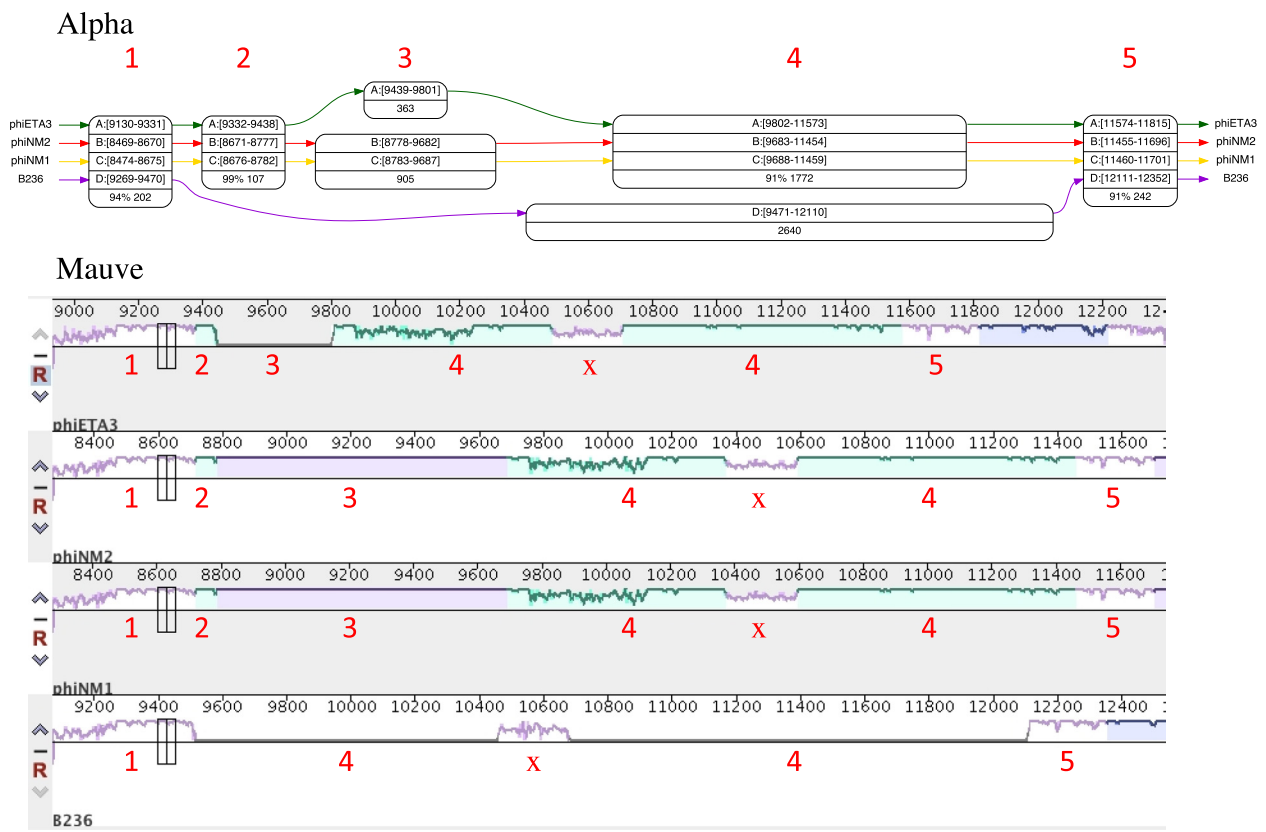

Fig. 8 Visual comparison. Alpha and Mauve alignments of the same region of the Staph4 dataset. Numbers in red were added to show the correspondence between Alpha alignments and Mauve color blocks. All Alpha alignments are found in the Mauve alignment, but Mauve apparently detects an alignment of the four genomes that is missed by Alpha: the block labeled 'x'. We show in the text that block x is a misalignment

bp. Three of the remaining large blocks – two predicted only by Mauve, and one only by Alpha – were validated using protein alignments. Figure 9 shows, in the various green-blue-shaded rectangles, the repartition of valid alignments, measured in total bp length, between common and exclusive alignments for Mauve and Alpha. These results are further partitioned by discriminating whether a block is an alignment, or an extension of an alignment.

Seven alignments predicted only by Mauve, totaling 2803 bp, were suspicious. They correspond to the yellow and red rectangles of Fig. 9. Three of them had non-significant tblastx results, despite the fact that at least one of the two sequences was annotated as coding. The remaining four alignments had a high number of gaps, contradicting the amino acid alignments of the corresponding homolog coding sequences. An example of each of these cases is detailed below, and the evidence for all others is detailed in Additional file 3.

Example of the non-homolog case. This is the block labeled 'x' in Fig. 8. Mauve aligns phage phiETA, [10484-10705], with phage B236, [10459-10678]. The nucleotide alignment has 8 gaps of lengths 1, 2, 3, 1, 3, 2, 2, and 1, over 222 bp. The best tblastx hit between the two sequences has length only 9 aa, with 4/9 identities, and with an e-value of 0.5. In this region, phage B236 is annotated for protein AKC04696:49-118, while phage phiETA3 is annotated for two proteins: YP_001004350:238-286 followed by YP_001004350:1-20. Thus there are two annotated coding regions with no detectable similarity, and trying to align them using the given nucleotide alignment would introduce numerous frameshifts.

Example of the homolog case (contradictory alignments). Mauve aligns phage phiETA, 8704-9129, with phage phiNM2, 8046-8468. The nucleotide alignment has 7 gaps, of lengths 2, 2, 1, 1, 3, 4, and 4. The corresponding blastp 142 aa alignment of annotated proteins YP_001004347:1-142 and ABF73110:1-141 has 58/142(41 %) identities, 92/142(64 %) positives and 1 gap. This is a rather good alignment of two distant homologs, but the Mauve alignment does not reflect the conservation of the two proteins.

When one excludes Mauve misalignments – that is, considering the green and blue rectangles of Fig. 9 – both aligners generally agree on the topology of the alignment graphs. As could be predicted, due to Mauve's seed and extend strategy, it has wider alignments compared to Alpha, and this accounts for the 778 bp extensions that Alpha could not predict. On the other hand, the 8 bp extensions that Alpha alone predicted, in four different blocks, were in fact verified to be exact matches that Mauve somehow missed. The two aligners also failed to detect a few small alignments, with a little advantage to Alpha: Mauve detected 294 bp that Alpha did not, while Alpha detected 358 bp that Mauve did not.

The high rate of Mauve misalignments on bacteriophage genomes is problematic. More than 20 % of Mauve column alignments on this dataset are either random, or belong to nucleotide alignments whose gaps contradicts the corresponding annotated proteins alignments. It should be noted, however, that the Mauve algorithm uses Muscle to produce alignments between its own anchors, and the way Muscle introduces gaps in nucleotide alignments does not seem to be appropriate for bacteriophage genomes, as we already saw in the preceding section on validating Alpha gapless alignments.

Overall, Alpha's alignment strategy captures the essential features of bacteriophage genomes. Regions that are similar are detected, while more dissimilar regions are not aggressively aligned. Being a conservative aligner, Alpha relies on functional collinearity to predict distant

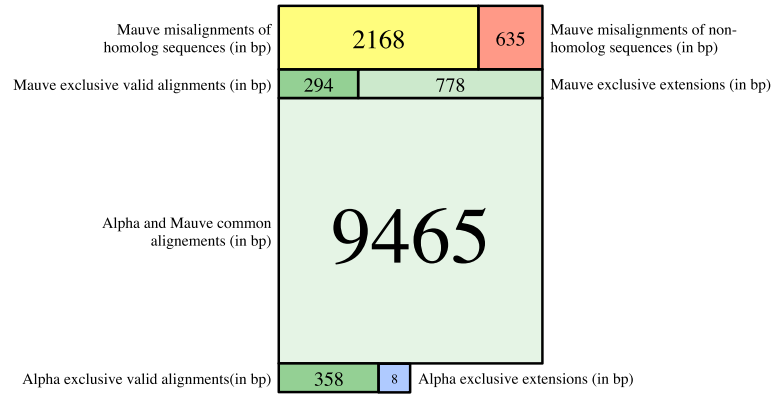

Fig. 9 Alignments comparison. Diagram representing the repartition of 13706 bp of 70 block alignments by Alpha or Mauve. The top 5 rectangles are from alignments produced by Mauve, and the bottom 3 are from Alpha alignments. Green and blue rectangles represent valid alignments, while yellow and red are misalignments

homologs that should be aligned using amino acids translations, such as the variants in columns 2 and 4 of Fig. 5.

Conclusion

Due to their peculiar mosaic structure where similar functions do not correspond to similar sequences, bacteriophage genomes are not well-suited to traditional whole genome alignment techniques. On the other hand, they exhibit features that can be leveraged to obtain alignments, most notably functional collinearity, a low duplication rate, and the presence of long shared sequences.

In this paper, we presented a mathematical model based on partial order graphs for performing multiple alignment of bacteriophage whole genomes, along with algorithms to operate on the model. Relying exclusively on the equality relation, the model is almost parameter free, greatly reducing the need to calibrate the aligner, yet delivers biologically meaningful results. The model has been implemented in the form of an interactive aligner that can perform multiple alignments of dozens of genomes and present the result in an attractive format.

We also showed that Alpha, used on bacteriophage genomes, produces biologically meaningful alignments, while avoiding the high rate of misalignments of complex heuristics such as progressiveMauve.

Our model supposes that all genomes under consideration are functionally collinear. This is often the case, but not always. Our program can detect when this condition is not satisfied – indicating the presence of rearrangements – but does not perform the alignment in such a case. A short-term goal is to extend our mathematical model and aligner to deal with rearrangements.

Some bacteriophage genomes present in the online databases are well annotated, other are less so. Another goal is to extend the aligner in order to perform automated transfer of annotations using the generated alignments.

Finally, while we focussed this study on Siphoviridae, we plan to test Alpha on more general viruse families in which horizontal transfer is widespread, and for which the collinearity property may hold.

The code, along with installation instructions for Ubuntu and OSX, is available on bitbucket (https://bitbucket.org/thekswenson/alpha).

Additional files

Additional file 1: Regions in which gapless alignments predicted by Alpha are contested by Clustal Omega, T-Coffee or Muscle. (ZIP 32 kb)

Additional file 2: Mauve alignment corresponding to the Alpha alignment of Fig. 7. (JPG 962 kb)

Additional file 3: Detailed comparison of Alpha and Mauve alignments for the four *S. aureus* phages phiETA3, phiNM2, phiNM1 and B236. (XLSX 83 kb)

Competing interests
The authors declare that they have no competing interests.

Authors' contributions
SB, AC, PG, AB and KS conceived the study and developed the model. KS wrote the code, helped by NP and SB. SB, AC, NP, PG and AB performed and analyzed the experiments. AB drafted the manuscript, helped by PG, SB and KS. All authors read and approved the final manuscript.

Acknowledgements
AB is partially supported by Canada NSERC Grant number 05729-2014. KS is partially supported by VIROGENESIS (EU H2020-PHC-32-2014 #634650). The team acknowledges the financial support of Labex NUMEV (ANR-10-LABX-20).

Author details
[1] ISEM, CNRS - Univ. Montpellier, Montpellier, France. [2] LIRMM, CNRS - Univ. Montpellier, 161 rue Ada, 34392 Montpellier, France. [3] IBC Institut de Biologie Computationnelle, Montpellier, France. [4] LaCIM, Université du Québec à Montréal, Montréal, Canada. [5] Département de mathématiques, Collège André-Grasset, Montréal, Canada.

Received: 6 October 2015 Accepted: 22 December 2015
Published online: 13 January 2016

References

1. Hatfull GF, Jacobs-Sera D, Lawrence JG, Pope WH, Russell DA, Ko CC, et al. Comparative genomic analysis of 60 Mycobacteriophage genomes: genome clustering, gene acquisition, and gene size. J Mol Biol. 2010;397(1):119–43.
2. Grose JH, Jensen GL, Burnett SH, Breakwell DP. Genomic comparison of 93 Bacillus phages reveals 12 clusters, 14 singletons and remarkable diversity. BMC Genomics. 2014;15:855.
3. Grose JH, Casjens SR. Understanding the enormous diversity of bacteriophages: the tailed phages that infect the bacterial family Enterobacteriaceae. Virology. 2014;468–470:421–43.
4. Kehr B, Trappe K, Holtgrewe M, Reinert K. Genome alignment with graph data structures: a comparison. BMC Bioinforma. 2014;15:99.
5. Botstein D. A theory of modular evolution for bacteriophages. Ann N Y Acad Sci. 1980;354:484–90.
6. Hatfull GF, Cresawn SG, Hendrix RW. Comparative genomics of the mycobacteriophages: insights into bacteriophage evolution. Res Microbiol. 2008;159:332–9.
7. Kahankova J, Pantucek R, Goerke C, Ruzickova V, Holochova P, Doskar J. Multilocus PCR typing strategy for differentiation of Staphylococcus aureus siphoviruses reflecting their modular genome structure. Environ Microbiol. 2010;12(9):2527–538.
8. Veesler D, Cambillau C. A common evolutionary origin for tailed-bacteriophage functional modules and bacterial machineries. Microbiol Mol Biol Rev. 2011;75(3):423–33.
9. Casjens SR. Comparative genomics and evolution of the tailed-bacteriophages. Curr Opin Microbiol. 2005;8(4):451–8.
10. Hatfull GF. Bacteriophage genomics. Curr Opin Microbiol. 2008;11:447–53.
11. Kececioglu J. The maximum weight trace problem in multiple sequence alignment. CPM'93 proceedings, LNCS 684. Berlin: Springer-Verlag; 1993, pp. 106–119.
12. Morgenstern B, Dress A, Werner T. Multiple DNA and protein sequence alignment based on segment-to-segment comparison. Proc Natl Acad Sci USA. 1996;93(22):12098–12103.
13. Lee C, Grasso C, Sharlow MF. Multiple sequence alignment using partial order graphs. Bioinformatics. 2002;18(3):452–64.
14. Darling AE, Mau B, Perna NT. progressiveMauve: multiple genome alignment with gene gain, loss and rearrangement. PLoS ONE. 2010;5(6):11147.
15. Gremme G, Steinbiss S, Kurtz S. GenomeTools: a comprehensive software library for efficient processing of structured genome annotations. IEEE/ACM Trans Comput Biol Bioinform. 2013;10(3):645–56.
16. Galler BA, Fisher MJ. An Improved Equivalence Algorithm. Commun ACM. 1964;7(5):301–3.
17. Tarjan RE. Efficiency of a Good But Not Linear Set Union Algorithm. J ACM. 1975;22(2):215–25.

18. Gansner ER, North SC. An open graph visualization system and its applications to software engineering. Softw Pract Experience. 2000;30(11):1203–1233.
19. Hohl M, Kurtz S, Ohlebusch E. Efficient multiple genome alignment. Bioinformatics. 2002;18(Suppl 1):S312–S320.
20. Chiapello H, Bourgait I, Sourivong F, Heuclin G, Gendrault-Jacquemard A, Petit MA, et al. Systematic determination of the mosaic structure of bacterial genomes: species backbone versus strain-specific loops. BMC Bioinforma. 2005;6:171.
21. Sievers F, Wilm A, Dineen D, Gibson TJ, Karplus K, Li W, et al. Fast, scalable generation of high-quality protein multiple sequence alignments using Clustal Omega. Mol Syst Biol. 2011;7:539.
22. Notredame C, Higgins DG, Heringa J. T-Coffee: A novel method for fast and accurate multiple sequence alignment. J Mol Biol. 2000;302(1): 205–17.
23. Edgar RC. MUSCLE: a multiple sequence alignment method with reduced time and space complexity. BMC Bioinforma. 2004;5:113.

Publications personnelles

[ABC+15a] Yoann Anselmetti, Vincent Berry, Cedric Chauve, Annie Chateau, Eric Tannier et Sèverine Bérard, « Ancestral gene synteny reconstruction improves extant species scaffolding », *BMC Genomics*, vol. 16(Suppl 10), n° S11, 2015. Réf. [P2] du CV détaillé.

[ABC+15b] Yoann Anselmetti, Vincent Berry, Cedric Chauve, Annie Chateau, Eric Tannier et Sèverine Bérard, « Ancestral gene synteny reconstruction improves extant species scaffolding », *in RECOMB-CG, Frankfurt, Germany*, 2015. Réf. [P11] du CV détaillé.

[ABC+16] Yoann Anselmetti, Vincent Berry, Cedric Chauve, Annie Chateau, Eric Tannier et Sèverine Bérard, « Comment la reconstruction de génomes ancestraux peut aider à l'assemblage de génomes actuels », *in Actes des Journées Ouvertes Biologie Informatique Mathématiques (JOBIM'16) accepté*, 2016. Réf. [P24] du CV détaillé.

[ALB+te] Yoann Anselmetti, Nina Luhmann, Sèverine Bérard, Eric Tannier et Cedric Chauve, « Comparative methods for reconstructing ancient genomes organization », *in Methods in Molecular Biology, Comparative Genomics*, 2016 *accepté*. Réf. [P19] du CV détaillé.

[BBC04] Sèverine Bérard, Anne Bergeron et Cedric Chauve, « Conservation of combinatorial structures in evolution scenarios », *in 2nd RECOMB Comparative Genomics Satellite Workshop, University of Bologna, Italy*, 2004. Réf. [P16] du CV détaillé.

[BBCP05] Sèverine Bérard, Anne Bergeron, Cedric Chauve et Christophe Paul, « Perfect sorting by reversal is not always difficult (extended abstract) », *in 5th Workshop on Algorithms in Bioinformatics (WABI'05), Palma de Majorque*, 2005. Réf. [P15] du CV détaillé.

[BBCP07] Sèverine Bérard, Anne Bergeron, Cedric Chauve et Christophe Paul, « Perfect sorting by reversal is not always difficult », *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 4, n° 1, p. 4–16, 2007. themeGCRéf. [P8] du CV détaillé.

[BCC+08] Sèverine Bérard, Annie Chateau, Cedric Chauve, Christophe Paul et Eric Tannier, « Perfect DCJ rearrangement », *in 6th RECOMB Comparative Genomics Satellite Workshop, Paris, France*, p. p158–169, 2008. Réf. [P13] du CV détaillé.

[BCC+09] Sèverine Bérard, Annie Chateau, Cedric Chauve, Christophe Paul et Eric Tannier, « Computation of perfect DCJ rearrangement scenarios with linear and circular chromosomes », *Journal of Computational Biology*, vol. 16, n° 10, p. 1287–1309, 2009. Réf. [P5] du CV détaillé.

[BCP08] Sèverine Bérard, Cedric Chauve et Christophe Paul, « A more efficient algorithm for perfect sorting by reversals », *Information Processing Letters*, vol. 106, n° 3, p. 90–95, 2008. Réf. [P7] du CV détaillé.

[BCP+16] Sèverine Bérard, Annie Chateau, Nicolas Pompidor, Paul Guertin, Anne Bergeron et Krister M. Swenson, « Aligning the unalignable : bacteriophage whole genome alignments », *BMC Bioinformatics*, vol. 17, n° 1, p. 1–13, 2016. Réf. [P1] du CV détaillé.

[BER03] Sèverine Bérard et Éric Rivals, « Comparison of minisatellites », *Journal of Computational Biology*, vol. 10, n° 3-4, p. 357–72, 2003. Réf. [P10] du CV détaillé.

[BGB+12a] Sèverine Bérard, Coralie Gallien, Bastien Boussau, Gergely J. Szöllősi, Vincent Daubin et Eric Tannier, « Evolution of gene neighborhoods within reconciled phylogenies », *Bioinformatics*, vol. 28, n° 18, p. i382–i388, 2012. Réf. [P3] du CV détaillé.

[BGB+12b] Sèverine Bérard, Coralie Gallien, Bastien Boussau, Gergely J. Szöllősi, Vincent Daubin et Eric Tannier, « Evolution of gene neighborhoods within reconciled phylogenies », *in 11th European Conference on Computational Biology (ECCB'12), Basel, Switzerland*, 2012. Réf. [P12] du CV détaillé.

[BNB+06] Sèverine Bérard, François Nicolas, Jérôme Buard, Olivier Gascuel et Éric Rivals, « A fast and specific alignment method for minisatellite maps », *Evolutionary Bioinformatics Online*, vol. 2, p. 327–344, 2006. Réf. [P9] du CV détaillé.

[BR02a] Sèverine Bérard et Eric Rivals, « Comparaison de minisatellites », *in Actes de la 24ème réunion annuelle du Groupe de Génétique et Biologie des Populations*, p. 128, 2002. Réf. [P22] du CV détaillé.

[BR02b] Sèverine Bérard et Eric Rivals, « Comparaison de minisatellites », *in Actes des Journées Ouvertes Biologie Informatique Mathématiques (JOBIM'02)*, p. 261–262, 2002. Réf. [P23] du CV détaillé.

[BR02c] Sèverine Bérard et Eric Rivals, « Comparison of Minisatellites », *in Proc. of the Sixth Annual International Conference on Computational Molecular Biology (RECOMB)*, p. 67–76, 2002. Réf. [P18] du CV détaillé.

[BR03] Sèverine Bérard et Eric Rivals, « Comparaison de séquences avec amplifications et contractions », *in 5ème congrès de la société Française de Recherche Opérationnelle et d'Aide à la Décision*, p. 169–170, 2003. Réf. [P21] du CV détaillé,.

[BTH05] Sèverine Bérard, Laurent Tichit et Carl Herrmann, « Clusterinspector, a tool to visualize ontology based relationships », *in Actes des Journées Ouvertes Biologie Informatique Mathématiques (JOBIM'05)*, p. 447–457, 2005. Réf. [P20].

[Bé00] Sèverine Bérard, « Rapport de DEA : Reconstruction d'histoire de répétitions en tandem », Mémoire de DEA, Université Montpellier II, LIRMM, 2000. Réf. [P26] du CV détaillé.

[Bé03] Sèverine Bérard, *Comparaison de séquences répétées en tandem et application à la génétique*. Thèse de doctorat, Université Montpellier II, LIRMM, 2003. Réf. [P25] du CV détaillé.

[GBM08] Cédric Gaucherel, Sèverine Bérard et François Munoz, « Equation against algorithm : Which differences and which one to choose ? », *in European conference on Computing And Philosophy E-CAP*, 2008. Réf. [P14].

[GBM10] Cédric Gaucherel, Sèverine Bérard et François Munoz, « Equation or algorithm : differences and choosing between them », *Acta Biotheoretica*, vol. 59, n° 1, p. 67–79, 2010. Réf. [P4].

[HBT09] Carl Herrmann, Sèverine Bérard et Laurent Tichit, « SimCT : a generic tool to visualize ontology-based relationships for biological objects », *Bioinformatics*, vol. 25, n° 23, 2009. Réf. [P6].

[RB04] Eric Rivals et Sèverine Bérard, « Alignement de séquences avec opérations non-commutatives », *in Quatrièmes journées francophones de recherche opérationnelle (FRANCORO IV), Fribourg, Suisse*, p. 16, 2004. Réf. [P17] du CV détaillé.

* *

*

Rappel du code couleur correspondant aux thématiques de recherche : alignement, génomique comparative et coévolution.

Bibliographie

[1] Mohamed I. Abouelhoda, Robert Giegerich, Behshad Behzadi et Jean-Marc Steyaert : Alignment of minisatellite maps based on run-length encoding scheme. *Journal of Bioinformatics and Computational Biology*, 7(02):287–308, 2009.

[2] Raja H. Ali, Sayyed A. Muhammad et Lars Arvestad : GenFamClust : an accurate, synteny-aware and reliable homology inference algorithm. *BMC Evolutionary Biology*, 16(1): 1–19, 2016.

[3] Santos Alonso et John A.L. Armour : MS205 minisatellite diversity in Basques : evidence for a pre-Neolithic component. *Genome Research*, 8:1289–1298, 1998.

[4] Dmitry Antipov, Anton Korobeynikov, Jeffrey S. McLean et Pavel A. Pevzner : hybridSPAdes : an algorithm for hybrid assembly of short and long reads. *Bioinformatics*, 32(7):1009–1015, 2016.

[5] John A.L. Armour *et al.* : Minisatellite diversity supports a recent African origin for modern humans. *Nature Genetics*, 13(2):154–60, 1996.

[6] David A. Bader, Bernard M. E. Moret et Mi Yan : A Linear-Time Algorithm for Computing Inversion Distance between Signed Permutations with an Experimental Study. *Journal of Computational Biology*, 8(5):483–491, 2001.

[7] Vineet Bafna et Pavel A. Pevzner : Sorting Permutations by Transpositions. *In Proceedings of the Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, 22-24 January 1995. San Francisco, California.*, pages 614–623, 1995.

[8] Vineet Bafna et Pavel A. Pevzner : Sorting by Transpositions. *SIAM Journal on Discrete Mathematics*, 11(2):224–240, 1998.

[9] Christoph P. Bagowski, Wouter Bruins et Aartjan J.W. te Velthuis : The Nature of Protein Domain Evolution : Shaping the Interaction Network. *Current Genomics*, 11(5): 368–376, 2010.

[10] Anne Bergeron : A very elementary presentation of the Hannenhalli-Pevzner theory. *In Combinatorial Pattern Matching, 12th Annual Symposium, CPM 2001 Jerusalem, Israel, July 1-4, 2001 Proceedings*, pages 106–117, 2001.

[11] Anne Bergeron, Cédric Chauve, Fabien de Montgolfier et Mathieu Raffinot : Computing common intervals of K permutations, with applications to modular decomposition of graphs. *SIAM Journal on Discrete Mathematics*, 22:1022–1039, 2008.

[12] Anne Bergeron, Cedric Chauve, Tzvika Hartman et Karine St-Onge : On the properties of sequences of reversals that sort a signed permutation. *In* J. Nicolas et C. Thermes, éditeurs : *Actes des Journées Ouvertes Biologie Informatique Mathématiques (JOBIM'02)*, pages 99–108, Saint-Malo, France, 10-12 June, 2002.

[13] Anne Bergeron, Julia Mixtacki et Jens Stoye : A Unifying View of Genome Rearrangements. *In Algorithms in Bioinformatics, 6th International Workshop, WABI 2006, Zurich, Switzerland, September 11-13, 2006, Proceedings*, pages 163–173, 2006.

[14] Alex Betts, Marie Vasse, Oliver Kaltz et Michael E. Hochberg : Back to the future : evolving bacteriophages to increase their effectiveness against the pathogen Pseudomonas aeruginosa PAO1. *Evolutionary Applications*, 6(7):1054–1063, 2013.

[15] François Bonhomme, Eric Rivals, Annie Orth, Gemma R. Grant, Alec J. Jeffreys et Philippe R.J. Bois : Species-wide distribution of highly polymorphic minisatellite markers suggests past and present genetic exchanges among house mouse subspecies. *Genome Biology*, 8(R80):337–48.

[16] David Botstein : A theory of modular evolution for bacteriophages. *Annals of the New York Academy of Sciences*, 354:484–490, 1980.

[17] Nordine Bouzekri, Paul G. Taylor, Michael F. Hammer et Mark A. Jobling : Novel mutation processes in the evolution of a haploid minisatellite, MSY1 : array homogenization without homogenization. *Human Molecular Genetics*, 7(4):655–9, 1998.

[18] Jérôme Buard et Alex J. Jeffreys : Big, bad minisatellites. *Nature Genetics*, 15:327–328, 1997.

[19] Marija Buljan et Alex Bateman : The evolution of protein domain families. *Biochemical Society Transactions*, 37(4):751–755, 2009.

[20] Laurent Bulteau, Guillaume Fertin et Irena Rusu : Sorting by Transpositions is Difficult. *CoRR*, abs/1011.1157, 2010.

[21] Alberto Caprara : Sorting by reversals is difficult. *In Proceedings of the First Annual International Conference on Research in Computational Molecular Biology (RECOMB), January 20-23, Santa Fe, NM, USA.*, pages 75–83, 1997.

[22] Humberto Carrillo et David Lipman : The Multiple Sequence Alignment Problem in Biology. *SIAM Journal on Applied Mathematics*, 48(5):1073–1082, 1988.

[23] Cédric Chauve, Yann Ponty et João Paulo Pereira Zanetti : Evolution of genes neighborhood within reconciled phylogenies : An ensemble approach. *In Advances in Bioinformatics and Computational Biology - 9th Brazilian Symposium on Bioinformatics, BSB 2014, Belo Horizonte, Brazil, October 28-30, 2014, Proceedings*, pages 49–56, 2014.

[24] Michel Chein, Michel Habib et Marie-Catherine Maurer : Partitive hypergraphs. *Discrete Mathematics*, 37(1):35–50, 1981.

[25] Dylan Chivian, Timothy Robertson, Richard Bonneau et David Baker : *Ab Initio Methods*, pages 547–557. John Wiley & Sons, Inc., 2005.

[26] Mélanie Colomb-Cotinat, Jessica Lacoste, Bruno Coignard, Sophie Vaux, Christian Brun-Buisson et Vincent Jarlier : Morbidité et mortalité des infections à bactéries multi-résistantes aux antibiotiques en france en 2012. Étude Burden BMR, rapport, Institut de veille sanitaire, Saint-Maurice, Juin 2015. http:/www.invs.sante.fr.

[27] Phillip E.C. Compeau, Pavel A. Pevzner et Glenn Tesler : How to apply de Bruijn graphs to genome assembly. *Nature Biotechnology*, 29:987–991, 2011.

[28] Thomas Cormen, Charles Leiserson et Ronald Rivest : *Introduction à l'algorithmique*. Practical Approach. DUNOD, 1994.

[29] Damien Coulomb : Résistances aux antibiotiques, la course contre la montre. *In Science & Santé n° 12*, pages 4–5, janvier/février 2013.

[30] Maxime Crochemore, Christophe Hancart et Thierry Lecroq : *Algorithms on strings*. Cambridge University Press, 2007.

[31] Aaron C.E. Darling, Bob Mau, Frederick R. Blattner et Nicole T. Perna : Mauve : multiple alignment of conserved genomic sequence with rearrangements. *Genome Research*, 14(7):1394–1403, Jul 2004.

[32] Aaron C.E. Darling, Bob Mau et Nicole T. Perna : progressiveMauve : multiple genome alignment with gene gain, loss and rearrangement. *PLoS ONE*, 5(6):e11147, 2010.

[33] Fabien de Montgolfier : *Décomposition modulaire des graphes. Théorie, extensions et algorithmes*. Thèse de doctorat, Université Montpellier II, 2003.

[34] Rodney G. Downey et Michael R. Fellows : *Parameterized Complexity*. Monographs in Computer Science. Springer, 1999.

[35] Nicolas Dufour, Laurent Debarbieux, Mélanie Fromentin et Jean-Damien Ricard : Treatment of Highly Virulent Extraintestinal Pathogenic Escherichia coli Pneumonia With Bacteriophage. *Critical Care Medicine*, 43(6), 2015.

[36] Isaac Elias et Tzvika Hartman : A 1.375-Approximation Algorithm for Sorting by Transpositions. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, 3(4):369–379, 2006.

[37] Adam C. English, Stephen Richards, Yi Han, Min Wang, Vanesa Vee, Jiaxin Qu, Xiang Qin, Donna M. Muzny, Jeffrey G. Reid, Kim C. Worley et Richard A. Gibbs : Mind the gap : upgrading genomes with Pacific Biosciences RS long-read sequencing technology. *PloS ONE*, 7:e47768, 2012.

[38] Guillaume Fertin, Anthony Labarre, Irena Rusu, Eric Tannier et Stéphane Vialette : *Combinatorics of Genome Rearrangements*. Computational molecular biology. MIT Press, 2009.

[39] Martin Figeac et Jean-Stéphane Varré : Sorting by Reversals with Common Intervals. *In Algorithms in Bioinformatics, 4th International Workshop, WABI 2004, Bergen, Norway, September 17-21, 2004, Proceedings*, pages 26–37, 2004.

[40] Robert D. Finn *et al.* : The Pfam protein families database : towards a more sustainable future. *Nucleic Acids Research*, 44(Database-Issue):279–285, 2016.

[41] Walter M. Fitch : Toward defining the course of evolution : minimum change for a specified tree topology. *Systematic Zoology*, 20 (4):406–416, 1971.

[42] Paul Flicek *et al.* : Ensembl 2014. *Nucleic Acids Research*, 42(D1):D749–D755, 2014.

[43] Margarita Flores *et al.* : Recurrent DNA inversion rearrangements in the human genome. *Proceedings of the National Academy of Sciences*, 104(15):6099–6106, 2007.

[44] Michael C. Fontaine *et al.* : Extensive introgression in a malaria vector species complex revealed by phylogenomics. *Science*, 347(6217), 2015.

[45] Delbert R. Fulkerson et Oliver A. Gross : Incidence matrices and interval graphs. *Pacific Journal of Mathematics*, 15:835–855, 1965.

[46] Brandon S. Gaut et Bruce S. Weir : Detecting substitution-rate heterogeneity among regions of a nucleotide sequence. *Molecular Biology and Evolution*, 11(4):620–629, 1994.

[47] Richard A. Gibbs, George M. Weinstock *et al.* : Genome sequence of the Brown Norway rat yields insights into mammalian evolution. *Nature*, 428:493–521, 2004.

[48] Zhabiz Golkar, Omar Bagasra et Donald Pace : Bacteriophage therapy : a potential solution for the antibiotic resistance crisis. *The Journal of Infection in Developing Countries*, 8(02), 2014.

[49] Dan Graur et Wen-Hsiung Li : *Fundamentals of molecular evolution*. Sinauer, Sunderland, Massachusetts, 2^{e} édition, 2000.

[50] Dan Gusfield : *Algorithms on Strings, Trees, and Sequences - Computer Science and Computational Biology*. Cambridge University Press, 1997.

[51] Yijie Han : Improving the Efficiency of Sorting by Reversals. *In Proceedings of the 2006 International Conference on Bioinformatics & Computational Biology, BIOCOMP'06, Las Vegas, Nevada, USA, June 26-29, 2006*, pages 406–409, 2006.

[52] Sridhar Hannenhalli et Pavel A. Pevzner : Transforming Cabbage into Turnip (Polynomial Algorithm for Sorting Signed Permutations by Reversals). *In Proceedings of the 27th ACM Symposium on Theory of Computing*, pages 178–189. ACM Press, 1995.

[53] Sridhar Hannenhalli et Pavel A. Pevzner : Transforming Men into Mice (Polynomial Algorithm for Genomic Distance Problem). *In 36th Annual Symposium on Foundations of Computer Science, Milwaukee, Wisconsin, 23-25 October 1995*, pages 581–592, 1995.

[54] Jennifer L. Harrow *et al.* : The Vertebrate Genome Annotation browser 10 years on. *Nucleic Acids Research*, 42(D1):D771–D779, 2014.

[55] Steffen Heber et Jens Stoye : Finding All Common Intervals of k Permutations. *In Combinatorial Pattern Matching, 12th Annual Symposium, CPM 2001 Jerusalem, Israel, July 1-4, 2001 Proceedings*, pages 207–218, 2001.

[56] Alexandra Henein : What are the limitations on the wider therapeutic use of phage ? *Bacteriophage*, 3(2):e24872, 2013. PMID : 24228220.

[57] Edwin Jacox, Cedric Chauve, Gergely J. Szöllősi, Yann Ponty et Celine Scornavacca : ecceTERA : Comprehensive gene tree-species tree reconciliation using parsimony. *Bioinformatics*, 2016.

[58] Alec J. Jeffreys, Annette MacLeod, Keiji Tamaki, David L. Neil et Darren G. Monckton : Minisatellite repeat coding as a digital approach to DNA typing. *Nature*, 354:204–209, 1991.

[59] Mark A. Jobling, Nordine Bouzekri et Paul G. Taylor : Hypervariable digital DNA codes for human paternal lineages : MVR-PCR at the Y-specific minisatellite, MSY1 (DYF155S1). *Human Molecular Genetics*, 7:643–653, 1998.

[60] Shuji Kanamaru : Structural similarity of tailed phages and pathogenic bacterial secretion systems. *Proceedings of the National Academy of Sciences of the United States of America*, 106(11):4067–4068, 2009.

[61] Anna R. Kersting, Eshchar Mizrachi, Erich Bornberg-Bauer et Alexander A. Myburg : Protein domain evolution is associated with reproductive diversification and adaptive radiation in the genus Eucalyptus. *New Phytologist*, 206(4):1328–1336, 2015.

[62] Sergey Koren et Adam M. Phillippy : One chromosome, one contig : complete microbial genomes from long-read sequencing and assembly. *Current opinion in microbiology*, 23: 110–120, 2015.

[63] Sergey Koren, Michael C. Schatz, Brian P. Walenz, Jeffrey Martin, Jason T. Howard, Ganeshkumar Ganapathy, Zhong Wang, David A. Rasko, W. Richard McCombie, Erich D. Jarvis et Adam M. Phillippy : Hybrid error correction and de novo assembly of single-molecule sequencing reads. *Nature Biotechnology*, 30:693–700, 2012.

[64] Gad M. Landau, Laxmi Parida et Oren Weimann : Using PQ Trees for Comparative Genomics. *In Combinatorial Pattern Matching, 16th Annual Symposium, CPM 2005, Jeju Island, Korea, June 19-22, 2005, Proceedings*, pages 128–143, 2005.

[65] Yu Lin, Sergey Nurk et Pavel A Pevzner : What is the difference between the breakpoint graph and the de Bruijn graph ? *BMC Genomics*, 15(Suppl 6):S6, 2012.

[66] Izabela Makalowska, Chiao-Feng Lin et Wojciech Makalowski : Overlapping genes in vertebrate genomes. *Computational Biology and Chemistry*, 29(1):1 – 12, 2005.

[67] Ján Manuch, Murray Patterson, Roland Wittler, Cédric Chauve et Eric Tannier : Linearization of ancestral multichromosomal genomes. *BMC Bioinformatics*, 13(S-19):S11, 2012.

[68] Claudine Médigue et Ivan Moszer : Annotation, comparison and databases for hundreds of bacterial genomes. *Research in Microbiology*, 158(10):724–736, décembre 2007.

[69] Michio Murata, Jane S. Richardson et Joel L. Sussman : Simultaneous comparison of three protein sequences. *Proceedings of the National Academy of Sciences*, 82(10):3073–3077, 1985.

[70] Daniel E. Neafsey, Robert M. Waterhouse *et al.* : Highly evolvable malaria vectors : The genomes of 16 Anopheles mosquitoes. *Science*, 2014.

[71] Saul B. Needleman et Christian D. Wunsch : A General Method Applicable to the Search for Similarities in the Amino Acid Sequence of Two Proteins. *Journal of Molecular Biology*, 48:443–53, 1970.

[72] Christian M. Parobek, Jeffrey A. Bailey, Nicholas J. Hathaway, Duong Socheat, William O. Rogers et Jonathan J. Juliano : Differing patterns of selection and geospatial genetic diversity within two leading Plasmodium vivax candidate vaccine antigens. *PLoS Neglected Tropical Diseases*, 8(4):e2796, 2014.

[73] Murray Patterson, Gergely J. Szöllosi, Vincent Daubin et Eric Tannier : Lateral gene transfer, rearrangement, reconciliation. *BMC Bioinformatics*, 14(S-15):S4, 2013.

[74] Daniel Paulino, René L. Warren, Benjamin P. Vandervalk, Anthony Raymond, Shaun D. Jackman et Inanç Birol : Sealer : a scalable gap-closing application for finishing draft genomes. *BMC Bioinformatics*, 16:230, 2015.

[75] Pavel A. Pevzner et Glenn Tesler : Genome Rearrangements in Mammalian Evolution : Lessons From Human and Mouse Genomes. *Genome Research*, 13(1):37–45, 2003.

[76] Tamar Pinhas, Dekel Tsur, Shay Zakov et Michal Ziv-Ukelson : Edit distance with duplications and contractions revisited. *In Combinatorial Pattern Matching*, pages 441–454. Springer, 2011.

[77] Jean-Paul Pirnay *et al.* : Introducing yesterday's phage therapy in today's medicine. *Future Virology*, 7(4):379–390, 2012.

[78] Kevin M. Potter, Valerie D. Hipkins, Mary F. Mahalovich et Robert E. Means : Mitochondrial DNA haplotype distribution patterns in Pinus ponderosa (Pinaceae) : Range-wide evolutionary history and implications for conservation. *American Journal of Botany*, 100(8):1562–1579, 2013.

[79] Kari-Jouko Räihä et Esko Ukkonen : The Shortest Common Supersequence Problem over Binary Alphabet is NP-Complete. *Theoretical Computer Science*, 16:187–198, 1981.

[80] Anthony Rhoads et Kin Fai Au : PacBio Sequencing and Its Applications. *Genomics, proteomics & bioinformatics*, 13:278–289, 2015.

[81] Eric Rivals, Clémence Bruyere, Claire Toffano-Nioche et Alain Lecharny : Formation of the arabidopsis pentatricopeptide repeat family. *Plant Physiology*, 141(3):825–839, 2006.

[82] Kristoffer Sahlin, Francesco Vezzi, Björn Nystedt, Joakim Lundeberg et Lars Arvestad : BESST - efficient scaffolding of large fragmented assemblies. *BMC Bioinformatics*, 15:281, 2014.

[83] Leena Salmela, Kristoffer Sahlin, Veli Mäkinen et Alexandru I. Tomescu : *Research in Computational Molecular Biology : 19th Annual International Conference, RECOMB 2015, Warsaw, Poland, April 12-15, 2015, Proceedings*, chapitre Gap Filling as Exact Path Length Problem, pages 281–292. Springer International Publishing, Cham, 2015.

[84] Michael Sammeth et Jens Stoye : Comparing tandem repeats with duplications and excisions of variable degree. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, 3(4):395–407, 2006.

[85] David Sankoff : Minimal mutation trees of sequences. *SIAM Journal on Applied Mathematics*, 28:35, 1975.

[86] David Sankoff : Edit Distances for Genome Comparisons Based on Non-Local Operations. *In Combinatorial Pattern Matching, Third Annual Symposium, CPM 92, Tucson, Arizona, USA, April 29 - May 1, 1992, Proceedings*, pages 121–135, 1992.

[87] Chaitanya R. Sanna, Wen-Hsiung Li et Liqing Zhang : Overlapping genes in the human and mouse genomes. *BMC Genomics*, 9(1):1–11, 2008.

[88] Magali Semeria, Eric Tannier et Laurent Guéguen : Probabilistic modeling of the evolution of gene synteny within reconciled phylogenies. *BMC Bioinformatics*, 16(14):1–11, 2015.

[89] Temple F. Smith et Michael S. Waterman : Identification of common molecular subsequences. *Journal of Molecular Biology*, 147:195–197, 1981.

[90] John D. Stead et Alec J. Jeffreys : Structural analysis of insulin minisatellite alleles reveals unusually large differences in diversity between Africans and non-Africans. *American Journal of Human Genetics*, 71:1273–1284, 2002.

[91] Krister M. Swenson, Paul Guertin, Hugo Deschênes et Anne Bergeron : Reconstructing the modular recombination history of Staphylococcus aureus phages. *BMC Bioinformatics*, 14(15):1–9, 2013.

[92] Eric Tannier, Anne Bergeron et Marie-France Sagot : Advances on sorting by reversals. *Discrete Applied Mathematics*, 155(6-7):881–888, 2007.

[93] Eric Tannier, Chunfang Zheng et David Sankoff : Multichromosomal Genome Median and Halving Problems. *In Algorithms in Bioinformatics, 8th International Workshop, WABI 2008, Karlsruhe, Germany, September 15-19, 2008. Proceedings*, pages 1–13, 2008.

[94] The Y Chromosome Consortium : A Nomenclature System for the Tree of Human Y-Chromosomal Binary Haplogroups. *Genome Research*, 12:339–48, 2002.

[95] Frédéric Thomas, Michel Raymond et Thierry Lefevre : *Biologie évolutive*. De Boeck, 2^{e} édition, 2016.

[96] Todd J. Treangen et Steven L. Salzberg : Repetitive DNA and next-generation sequencing : computational challenges and solutions. *Nature Reviews Genetics*, 13:36–46, 2012.

[97] Takeaki Uno et Mutsunori Yagiura : Fast Algorithms to Enumerate All Common Intervals of Two Permutations. *Algorithmica*, 26(2):290–309, 2000.

[98] David Veesler et Christian Cambillau : A Common Evolutionary Origin for Tailed-Bacteriophage Functional Modules and Bacterial Machineries. *Microbiology and Molecular Biology Reviews*, 75(3):423–433, 2011.

[99] Lusheng Wang et Tao Jiang : On the Complexity of Multiple Sequence Alignment. *Journal of Computationnal Biology*, 1(4):337–48, 1994.

[100] Sofia Yancopoulos, Oliver Attie et Richard Friedberg : Efficient sorting of genomic permutations by translocation, inversion and block interchange. *Bioinformatics*, 21(16):3340–3346, 2005.

[101] Guojie Zhang, Erich D. Jarvis et M. Thomas P. Gilbert : A flock of genomes. *Science*, 346(6215):1308–1309, 2014.

Résumé

Résumé en français...

Abstract

Abstract in english...

Numéro d'ordre : 00000