Genomics of Gene Gain and Gene Loss in Eukaryotes

Dissertation
in fulfilment of the requirements for the degree
Doctor rerum naturalium
of the Faculty of Mathematics and Natural Sciences
at Kiel University
submitted by

Robert Bakarić
Kiel, 2016.

First referee:	Prof. Dr. rer. nat. Diethard Tautz
Second referee:	Prof. Dr. rer. nat. Tal Dagan
Date of the oral examination:	05.12.2016
Approved for publication:	05.12.2016

Signed: Prof. Dr. Wolfgang J. Duschl (Dean)

To my loving family.

Acknowledgements

I owe my gratitude to Prof. Dr. Tomislav Domazet-Lošo and Prof. Dr. Diethard Tautz for the opportunity and support they shown in bringing this project to its end. I thank them for their patience and complete independence in conducting it in my own way and at my own pace, for being open to new ideas, and especially for their support when required the most.

I thank Prof. Dr. John Baines and Prof. Dr. Arne Traulsen for being part of my Thesis Committee during my time in Plön.

I am also grateful to Dr. Martin Sebastijan Šestak, Dr. Rafik Tarek Neme Garrido and Dr. Miguel Baltazar-Soares for their insightful discussions, suggestions, critics regarding the thesis and in general for helping me with various difficulties I came across during my studies. Moreover, a large parts of this thesis would not have been possible without rigorous critiques and suggestions made by Prof. Dr. Kristian Vlahoviček and Dr. Mirjana Domazet-Lošo.

I thank the International Max Planck Research School for Evolutionary Biology, especially Dr. Kerstin Mehnert, for her involvement and support in crucial moments during my studies.

I thank the University of Zagreb and grup for bioinformatics led by Prof. Dr. Kristian Vlahoviček for providing the essential computational resources required in this project.

I thank all my friends and colleagues which I haven't mentioned here but have directly and indirectly contributed to seeing this project reach its end.

I would like to thank my parents Marijan and Vesna and my brother Kristijan for their neverending encouragement and confidence they shown during this entire endeavour I embarked upon many years ago.

Zusammenfassung

Im Laufe der Evolution sind Arten zunehmend höherer Komplexität entstanden, sowohl auf biologischer (organischer) als auch genomischer Ebene. Ob es sich dabei um einen inhärenten Trend handelt ist in den letzten Jahrzehnten immer wieder in Frage gestellt worden, u.a. durch die Arbeiten von Stephen J. Gould und Eugene Koonin, die darauf hinweisen dass die gegenwärtige Evidenz nicht ausreicht um solche Trends klar zu belegen. Um irgendwelche der vorgeschlagenen komplexen Trends auf der genomischen Ebene zu wiederlegen oder zu bestätigen und somit deren potentielle Auswirkung auf eine höhere organisatorische Ebene zu implizieren, ist es notwendig eine hohe Anzahl von ancestralen Genomen in verschiedenen Evolutionslinien zu rekonstruieren. Eine solche Rekonstruktion erfordert die Bestimmung des Gewinns oder des Verlusts von Genen und Gen-Familien, die quer durch die verschiedenen taxonomischen Gruppen vorkommen. Solche Berechnungen basieren auf rechnerischen "jede-gegen-jede" Sequenzvergleichen. Doch selbst schnelle und gut etablierte heuristische Algorithmen wie BLAST kommen damit an die Grenzen der computertechnischen Möglichkeiten. Um dieses Problem zu lösen, wird in dieser Arbeit eine Lösungsmöglichkeit vorgeschlagen. Diese beruht auf dem Konzept einer Vorfiltrierung auf der Basis von Sequenzidentität, mittels eines hochdimensionierten index-basierten Suchalgorithmus, der tausendmal schneller als BLAST ist. Diese Lösung wurde in dem Computerprogramm *QPhyloStrat* implementiert und damit wurde eine Analyse zum Gewinn und Verlust von Genfamilien in 383 Eukaryotischen Linien durchgeführt. Die darauf basierende Rekonstruktion zeigt eine über alle Linien konsistentes glockenförmiges Muster an Veränderung in genomischer Komplexität, mit einer periodisch beginnenden Komplexität während des Protozoikums, gefolgt von Verlusten im Phanerozoikum. Es scheint auch eine generelle in-

verse Beziehung zwischen Gewinn und Verlust von Genfamilien zu bestehen. Neben diesen generellen Trends gibt es Evolutionsperioden mit besonders hohen Genfamilien Gewinn und Verlust Raten, die mit den bekannten evolutionären Transitionen korrelieren.

Summary

Evolution is often perceived as a process driving species toward greater complexity at both biological (organismal) and genomic level. However, this concept has repeatedly been challenged over the years through writings of authors like Stephen J. Gould and Eugene V. Koonin, rendering the current evidence inadequate for any strong, trend-like (progressive in particular) claims supporting the competing views. The current state of this problem is an agreement that despite the diversity of individual case-study evidence, it is still impossible to make any unequivocal conclusion without a sufficiently accurate evolutionary reconstruction of ancestral genomes across numerous evolutionary lineages. Such reconstruction would provide information regarding the change in the number of genes as a function of time and serve as an adequate proxy for monitoring genomic and consequently organismal complexity patterns. The reconstruction consists of a detailed mapping of gain and loss of genes and gene families over a large number of taxonomically diverse groups. In terms of computational difficulty, this task is seen as exceptionally hard, even in the case when a fast and a well-established heuristic sequence similarity search algorithm like BLAST is used. To address this problem I propose a novel, sequence identity based pre-filtering solution for homology detection, utilizing high dimensional index based similarity search algorithm, thousand times faster than BLAST. I implement this solution in a gene gain computation tool I call *QphyloStrat* and conduct the analysis by mapping the gene family gain and loss events across 383 Eukaryote lineages. The resulting reconstruction reveals a consistent, across all investigated lineages, bell-shaped pattern of change in genomic complexity, with complexity periodically increasing throughout Proterozoic eon, followed by a more systematic decrease prevailing the Phanerozoic. Moreover, a global inverse relationship between gain and loss of

gene families appears to be a general rule. Aside from these global trends, some evolutionary periods exhibit specific profiles with exceptionally high gene family gain or loss rates mostly associated to known key evolutionary transition events.

Contents

List of Figures

List of Tables

Nomenclature

Set Theory Symbols

$\mathbb{R}$	A set of real numbers
$\mathbb{N}$	A set of natural numbers
$\mathbb{Z}(\mathbb{Z}^+)$	A set of integers (positive integers)
Σ	Alphabet: A finite set of characters (letters, symbols or digits) intended to be used in string operations
Ω	A finite set of strings
Q	A finite set of query(input) strings
S	A finite set of subject(database) strings
A	A finite set of species
G	A finite set of genes (genome)
P	A finite set of nodes within T
bc	A set of nodes subset of P
C	A finite set of genes members of the same gene family (subset of S)
ps	A set of genes traced to a given point of origin (phylostrata cluster, a subset of S)
PS	A set of phylostrata clusters ($ps \subset PS$)
$\mathscr{K}$	A set of k-mers (k size substrings)

String Symbols

ω	A String: finite sequence of characters (letters, symbols or digits) from an alphabet (Σ)
q	Query (input) string
s	Subject (database) string
κ	A k-mer $\kappa \in \mathscr{K}$

Other Symbols

a	A species $a \in A$
p	A node $p \in P$
$\mathfrak{I}$	Gene family influx rate
$\mathfrak{C}$	Genome complexity
K	Kolmongorov complexity
ψ	Environment
c	Occurrence count
pgi	phylogeny gene identifier
ti	taxonomy identifier
pi	phylostratigraphy identifier

Functions

d_e	Edit distance
d_k	k-mer distance
d_{kt}	k-mer type distance
w	Edit operation

O Upper bound on runtime performance

JaccScore Jaccard score

e Expectation value

Acronyms / Abbreviations

AA Amino Acid

DNA Deoxyribonucleic Acid

RNA Ribonucleic Acid

GFGE Gene Family Gain Event

GFLE Gene Family Loss Event

GCGR Genome Complexity Growth Rate

LUCA Last Universal Common Ancestor

LECA Last Eukaryotic Common Ancestor

MRCA Most Recent Common Ancestor

BLAST Basic Local Alignment Search Tool

RMQ Range Minimum Query

ET Euler Tour

HD High Dimensionality

XNN X Nearest Neighbours

HSP High Scoring Pair

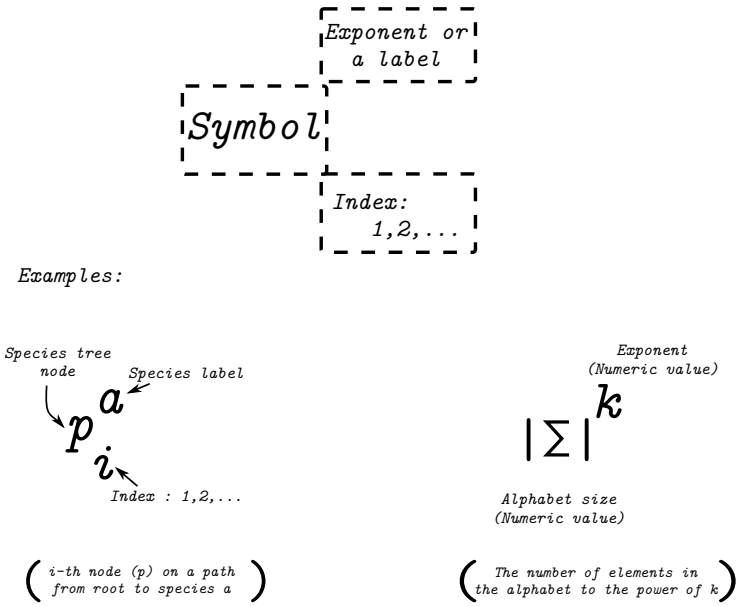


Fig. 1 An example of indexing and labelling conventions.

Chapter 1

Introduction

Genes are essential units of heritable information. They express information stored in genomes in the form of RNA molecules and proteins, which are the basic elements of the phenotype (Bromham, 2016). Gene content substantially differs between organisms, and at the start of the genome era, it was obvious that some genes are broadly present across all domains of life while others could only be found in narrow phylogenetic lineages (Dujon, 1996; Tautz and Domazet-Lošo, 2011). With an aim to understand the evolutionary dynamics of genomes, the gain and loss of genes has been the focus of comparative genomics for a long time (Albalat and Canestro, 2016; Koonin, 2009; Nei and Hughes, 1992; Nei and Rooney, 2005; Tautz and Domazet-Lošo, 2011).

Initially, it was recognized that gene duplications play an important role in the formation of gene families (Ohno, 1970) and the mechanisms of the sequence similarity maintenance within a gene family was discussed in terms of divergence and concerted evolution

(Brown and Sugimoto, 1974; Nei and Rooney, 2005). However, it was soon realized that concerted evolution cannot be a leading mechanism for maintaining similarity among gene family members (Nei and Hughes, 1992; Nei and Rooney, 2005). Instead, a birth-and-death model of protein family evolution was proposed which assumes that new genes are created by gene duplication followed by some duplicates being preserved in the genome for a long time due to purifying selection, whereas others are occasionally lost by deletion or inactivation (Nei and Rooney, 2005). However, in its original form the birth-and-death model only applies to the evolutionary dynamics of already existing genes and families; i.e. it does not consider mechanisms that lead to the formation of completely new gene families.

The first genome sequencing project revealed that genomes harbour many unknown genes that could be found only in specific phylogenetic clades (Dujon, 1996). Such orphan genes were studied from the perspective of evolutionary sequence rates (Domazet-Lošo and Tautz, 2003), sequence properties (Domazet-Lošo and Tautz, 2003; Fischer and Eisenberg, 1999) and their functional roles (Khalturin et al., 2008, 2009). The results of these studies revealed that a majority of orphan genes produce functional proteins that are often involved in accessory functions and specific ecological adaptations (Domazet-Lošo and Tautz, 2003; Khalturin et al., 2009), but in some cases could also be essential (Chen et al., 2010). This is supported by the finding that some orphan genes have evolutionary rates comparable to slow evolving genes that are broadly distributed on the tree of life (Domazet-Lošo and Tautz, 2003). Accordingly a model of orphan gene evolution was proposed which assumes that an orphan originates by duplication from an existing gene followed by fast divergence due to a new adaptation, which leads to a significant shift in the protein sequence space (Domazet-Lošo and Tautz, 2003). This model of orphan gene evolution was later generalized in the form of the punctuated protein family evolution which assumes that processes akin to orphan gene formation generate founder genes at any phylogenetic depth and lead to the constant influx of novel genes through evolutionary time (Domazet-Lošo et al., 2007).

Another possible mechanisms for the formation of novel genes is de novo emergence from a previously non-coding sequence (Carvunis et al., 2012; Heinen et al., 2009; Li et al., 2010; Neme and Tautz, 2014). For a long time, de novo emergence was considered to be

very unlikely (Tautz, 2014) and had therefore initially not been seriously considered as a model of origin of orphan genes (Domazet-Lošo and Tautz, 2003). However, it is now clear that de novo gene birth is in fact another important process that can lead to the formation of the orphan genes and novel genes in general (Carvunis et al., 2012; Neme and Tautz, 2013, 2014; Tautz and Domazet-Lošo, 2011). For instance, studies in yeast revealed that de novo genes emerge via protogenes – sequences with gene properties like stable expression or translation but without a well-established function (Carvunis et al., 2012). However, in practice it is very hard to distinguish if novel genes (orphan gene) are formed through the process of duplication and divergence beyond sequence recognition or through the process of de novo evolution from non-genic sequences (Schlötterer, 2015). In fact, it is possible to detect de novo evolution only by comparing closely related genomes at the DNA level (Cai et al., 2008; Neme and Tautz, 2014; Tautz and Domazet-Lošo, 2011). Therefore, it was proposed that both classes of genes – those originated from non-genic sequences and those that formed by duplication and divergence beyond sequence recognition – could be called de novo genes (Schlötterer, 2015).

It has been repeatedly noted that de novo gene emergence is particularly high in very recent evolutionary periods (Palmieri et al., 2014; Tautz and Domazet-Lošo, 2011; Wissler et al., 2013). This effect was studied in Drosophila and it was found that de novo genes are both rapidly gained and lost with higher chances to be eliminated from the genome in relatively short amount of time after their acquisition (Palmieri et al., 2014). This suggested that organisms continuously possess a pool of de novo genes as a source of variability for adaptive needs in ever changing environments (Neme and Tautz, 2016; Palmieri et al., 2014).

According to the most recent view on the evolutionary life cycle of genes in genomes (Neme and Tautz, 2014) de novo genes are acquired through gene "birth" from protogenes – a stochastically generated construct in non-coding sequences that lack a proper function but has gene-like features such as stable expression or translation (Carvunis et al., 2012). Further accumulation of mutations in protogenes may obstruct their expression and/or translation bringing them back to the initial non-coding random state. On the other hand, if a newly formed protogene construct affects an organism's fitness, it will become visible to

selection. If the protogene provides an advantage to the organisms positive selection could drive its fixation in the population. At this stage de novo gene is formed and it can become a source for other genes through duplication processes, horizontal gene transfer, fusion with other genes and similar mechanisms (Ivancevic et al., 2013; Kaessmann, 2010; Lynch and Conery, 2000; Mitelman et al., 2007; Zhou et al., 2008).

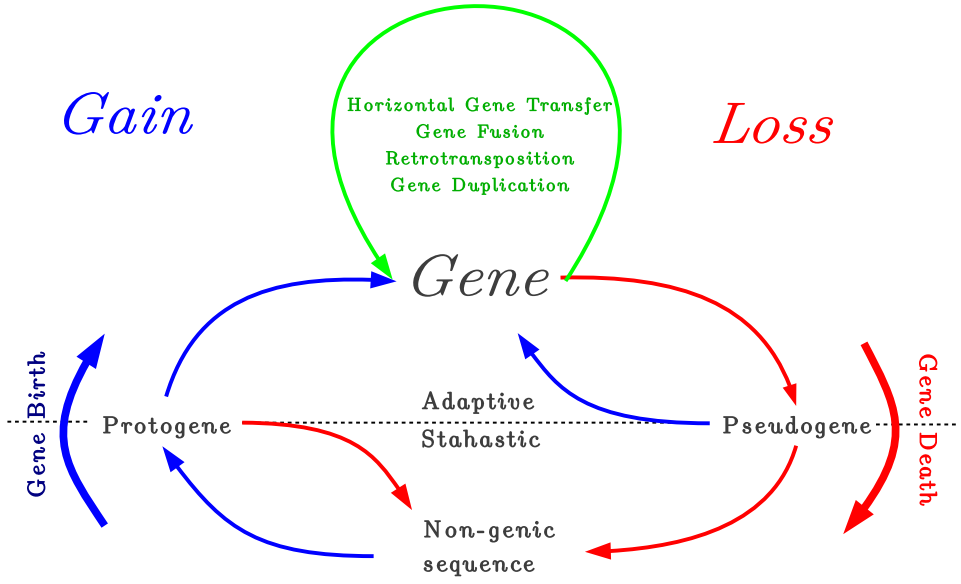


Fig. 1.1 Birth-and-death model of evolution. (Illustration created according to Neme and Tautz (2014)) Blue arrows indicate transitions by which new genes or their precursors (proto-genes and pseudogenes) are "born". Red arrows indicate transitions by which genes are lost, i.e. they represent processes that alter the underlying sequence such that the sequence is lost from the functional space of the genome. The green arrow stands for alternative additional gain mechanisms, which increase the gene repertoire through the existing ones. The proto-gene concept was proposed by Carvunis et al. (2012) as an intermediate structure, a precursor that has some gene-like properties (i.e. stable expression or translation), but is still not a proper functional unit. For both pseudogenes and proto-genes, the loss of selective pressure can lead to gene loss whereas the opposite may lead to gene gain.

Once stably incorporated, genes can eventually be lost from the genome if their sequence does not provide any more selective advantage to the organism (Neme and Tautz, 2014), for instance, if the environment changes. These genes accumulate mutations that could result in loss of expression and coding potential, a process known as pseudogenizations, or could be lost by a more invasive mechanism such as an excision via unequal crossover or transposable elements (Albalat and Canestro, 2016). Whichever the case, this model posits that a gene loss generally balances a gene gain and that both processes are equally important for the evolutionary integrity of genomes. However, although the model recognizes transition from the stochastic to the adaptive phase during gene birth, it does not state which forces shape the sequence during the gene death phase (Neme and Tautz, 2014).

A loss of gene could be fixed in the population through genetic drift or alternatively, if the gene loss for some reason increases fitness, through positive selection (Wolf and Koonin, 2013). Initially, more attention was given to the neutral mechanisms of gene loss (Wagner, 2008), however, recent studies show that adaptive loss is common in bacteria (D'Souza et al., 2014). For instance, the study conducted by Koskiniemi et al. (2012) shows that 25% of genes in *Salmonella enterica* increase fitness when deleted under one or several growth conditions. Similarly, a meta-analysis of bacterial genome-wide fitness data for 200 bacterial species across 144 different conditions shows extreme abundance of adaptive null mutations with particular mutations sometimes being adaptive in more than 10 conditions (Hottes et al., 2013). These results coupled with other similar studies (D'Souza et al., 2014; Morris, 2015; Morris et al., 2012) suggest the important role of selection in loss and reductive genome evolution.

Adaptive gene loss has also been reported in many eukaryotic species (Greenberg et al., 2003; Hoballah et al., 2007; Zufall and Rausher, 2004) mostly focusing on individual genes. For instance, in plants the loss of AN2 gene leads to the appearance of white flowers in *Petunia axillaris* which has been suggested to be the adaptation for pollination by nocturnal hawk moths (Hoballah et al., 2007). Another study shows that the loss of SRC gene in *A. thaliana* eliminates the self-incompatibility during fertilization thus allowing colonization of remote ecological niches like oceanic islands (Baker's rule) (Greenberg et al., 2003). Several

studies conducted on the opsin family, a group of light-sensitive proteins found in photoreceptor cells of the retina, across 23 vertebrates, reported adaptive gene loss as a response to changes in the environment (Davies, 2007; Davies et al., 2007). Moreover, the evolutionary origin of humans is thought to be associated to the adaptive loss of myosin heavy chain 16 (MYH16) (Stedman et al., 2004) and CMP-N-acetylneuraminic acid hydroxylaze (CMAH) (Chou et al., 2002) genes. It has been postulated that the loss of MYH16 gene has been one of the key factors that led to the increase in cranial capacity and brain size at the origin of humans (Stedman et al., 2004), while the loss of CMAH, which is still present in other primates, is related to pathogen resistance (Chou et al., 2002).

Although individual examples of adaptive gene loss are often reported, it is not clear how frequent they are in multicellular eukaryotes. Due to generally smaller population sizes of multicellular eukaryotes it is usually held that the evolution of gene loss is primarily driven by neutral processes (Albalat and Canestro, 2016). This view is supported by the finding that rates of gene loss and molecular evolution are correlated in five vertebrate and five insect species (Wyder et al., 2007). However, in order to carry out any tests regarding evolutionary mechanisms associated to gene loss, it is necessary to reconstruct ancestral genomes, which includes careful mapping of gene gain and loss to specific points on the phylogeny. Current studies are designed to look for gene loss cover around a dozen genomes, focusing mostly on the patterns within the Phanerozoic period (Kortschak et al., 2003; Kusserow et al., 2005; Ptitsyn and Moroz, 2012; Putnam et al., 2007; Sakarya et al., 2008; Technau et al., 2005). For example it was found that that the ancient metazoan genome was much more complex than previously thought and that gene loss was common in animal lineages (Putnam et al., 2007). However, currently no extensive analysis of gene loss across a wide range of eukaryotic species exists.

After a novel gene is stably incorporated in the genome repeated duplication process could generate a family of genes (paralogues) in the genome (Gabaldón and Koonin, 2013; Nei and Rooney, 2005). Similar selective pressures can act on the family members often, thus maintaining their sequence similarity through long time periods. In the context of phylogeny, speciation is another process that expands the size of gene families, members of which are

defined as orthologus (Gabaldón and Koonin, 2013). Regardless of the gene family size within a genome, or in the phylogeny, the family is initiated by a novel gene with unique sequence (Domazet-Lošo et al., 2007; Nei et al., 1997; Nei and Hughes, 1992; Nei and Rooney, 2005; Tautz and Domazet-Lošo, 2011). The novel genes that initiate gene families are described as founder genes and their origin was initially linked to the formation of orphan genes (Domazet-Lošo et al., 2007; Domazet-Lošo and Tautz, 2003) and later when it was recognized that non-coding sequences are also the source of sequence novelty, to the creation of de novo genes (Carvunis et al., 2012; Neme and Tautz, 2014; Tautz and Domazet-Lošo, 2011).

Gene families seeded by founder genes and expanded in the genomes could enter contraction phase trough gene loss depending on the ecological changes. (Prachumwat and Li, 2008; Sharpton et al., 2009). The contraction of gene family size could ultimately lead to its extinction in the genome. This is reverse to the formation of founder genes and it is an important process that influences the overall sequence diversity in the genomes. For instance, it is suggested that loss of the entire PYHIN family in bats is connected to flight-induced adaptation (Ahn et al., 2016). Although the loss of an entire gene family is occasionally reported (Guo, 2013), it seems to be a wide-spread phenomenon (Demuth et al., 2006; Hahn et al., 2007b), that directly affects the genome information content, but until recently it received only little attention. Generally, the loss of an entire gene family might not be a mere sum of effects of individual gene losses but could affect fitness in its own right. This is supported by the finding that bacteria tend to keep diversity of protein families while sacrificing paralogs during reductive evolution (Mendonca et al., 2011). However, whether the loss of entire gene family will have considerable deleterious effects on the long run is highly dependent on the environment (Albalat and Canestro, 2016; Morris, 2015).

The relationship between the loss of a trait and the environment, regardless of the underlying loss being a single gene or the entire family, has been investigated on several occasions (Ellers et al., 2012; Morris, 2015; Morris et al., 2012) with the increasing evidence supporting the "compensated trait loss" conjecture (Ellers et al., 2012). Moreover, this gave rise to a newly proposed theory of reductive evolution driven by selection against costly leaky

functions called the Black Queen hypothesis (Morris, 2015; Morris et al., 2012). In short, traits that are labelled as "Black Queen" are those that are neither purely private nor public (Morris, 2015), and according to the hypothesis, communities possessing such traits, are in the race to the bottom as their members sequentially lose the features that are available to them through the environment. As a result, the fitness of these members increases since the price for maintaining these costly functions is reduced to zero (Morris et al., 2012). Many of the evidence in favour of this hypothesis have been reviewed in Morris (2015), confirming the existence of Black Queen traits and their role in the evolution of microbial communities, however, similar claims cannot be made for multicellular eukaryotes. Although compensated trait loss in animals and plants has been recorded on several occasions (Ellers et al., 2012; Payne and Loomis, 2006) and gene dispensability (as an indicator) is reported in *C. elegans*, (Kamath et al., 2003; Sonnichsen et al., 2005) *D. melanogaster* (Dietzl et al., 2007), *M. muscullus* (White et al., 2013) and *H.sapiens* cancer cells (Osorio, 2015), relatively long generation times and small population sizes of these species present a formidable barrier for proving, or at least providing strong evidence for the Black Queen hypothesis to plays an important role in the evolution of multicellular eukaryotes. Therefore, novel bioinformatic solutions allowing for evolutionary analysis that span further back in time (such as ancestral gene content reconstruction strategies) will prove to be an invaluable factor in uncovering the effects the Black Queen has on the evolution of multicellular species, as well as provide further confirmation for the role it plays in microbial communities.

Ancestral gene family (gene) content reconstruction is a well established procedure in molecular evolution probably as old as the field itself (Fitch, 1970). Over the years many strategies have been proposed ; e.g., see Boussau et al. (2004); Kortschak et al. (2003); Kunin et al. (2005a); Kusserow et al. (2005); Ouzounis et al. (2006); Ptitsyn and Moroz (2012); Putnam et al. (2007); Technau et al. (2005). Most of these are based on a simple principle of tracing the origin of a gene family to its most recent point in evolution (the common ancestor of all species involved), but not necessarily the oldest one and therefore, usually focusing on a single ancestral period. The number of involved species in such reconstructions varies between a couple, to a dozen of species involved, which probably can be explained as a

consequence of computational cost associated with such calculations. One example of such comparative analysis showed that eumetazoan ancestral genome contained at least 7,766 gene families (Putnam et al., 2007), inferred from a comparison involving six genomes, sea anemone and 5 bilaterian ones. In another study 7,350 gene families were reported to be present in the most recent common ancestor (MRCA) of two *Neopterygii* and five *Sarcopterygii* species (Blomme et al., 2006), while 9,990 gene families were associated to a mammalian ancestor (Demuth et al., 2006).

From a systematic point of view as members of their respective families, genes are further divided into those that encode protein sequences and those that do not (rRNA coding genes). Ergo families are accordingly subdivided by analogy into gene and protein families (Daugherty et al., 2012). Protein families, more often than gene families are used when functional features of newly identified genes are to be identified, since the information within amino acid sequences prove to be more often conserved between distantly related family members than within nucleotide encoded ones. This conservation of information is a primary factor in determining relatedness since it is the strictest indicator of homology and therefore the clearest hallmark of common ancestry (Altschul et al., 1990; Dayhoff, 1976; Pearson and Sierk, 2005). This property was utilized by an ancestral protein family reconstruction analysis conducted in 2005, which involved 200,000 families across 165 species (12 eukaryotes and 153 prokaryotes) (Kunin et al., 2005a). The study provided the first global insight into both vertical and lateral "flow" of genetic material among the analysed lineages from which propensity for genes exchange across the tree of life was calculated. The result obtained for the first time revealed a magnitude of lateral transfer that proposed the replacement of the tree of life hypothesis with the network of life hypothesis (Kunin et al., 2005a). At the time the study was considered as one of the most sophisticated approaches for ancestral protein family reconstruction analysis utilizing a novel GeneTrace algorithm (Kunin and Ouzounis, 2003). Today, with more than 40,000 bacterial genomes (proteomes) available as of May, 2016 and thousands of eukaryotic pending (www.ensembl.org), the strategy implemented in the algorithm faces the same problem regarding computational demands as every other solution mentioned above.

As stated, the information preserved with in a protein reflects the relatedness, since it is the strictest indicator of homology. Homology has been a contentious topic and a source of vigorous debates for decades (Brigandt, 2002, 2003; Butler and Saidel, 2000; De Beer, 1971; Hillis, 1994; Kleisner, 2007). Today, its evolutionary perspective is considered to be the most informative (Brigandt, 2002), thus in the absence of a pre-specified reference point, homology in principle can be defined as the sameness with respect to its ancestral point of reference ((Kleisner, 2007): definition modified accordingly). This interpretation of homology is sometimes referred to as transformational (De Beer, 1971; Donoghue, 1992), since for any two given features (ancestral and descendant), to be defined as homologous, there has to exist a sequence of intermediate ones on a path from some specified ancestor to its descendant (Brigandt, 2002). Thus transformational homology aims to explain the theoretical goal of evolutionary biology – evolution by adaptation (Brigandt, 2002), and is therefore central to understanding ancestral reconstruction strategies. Ancestral genome (protein) reconstruction strategies require homology to be interpreted on a sequence level therefore further narrowing down the definition to its molecular interpretation. Therefore, homology as such refers to a concept in which two features are considered homologous if there exists a certain level of similarity, for instance, similarity between nucleotide or amino acid residues (Pearson, 2013).

Computing sequence similarity is one of the first, and foremost informative step in any analysis that involves sequence data. Thus, according to the evolutionary interpretation of homology, significant similarities between sequences can be interpreted as evidence of common ancestry, where the term "significant" refers to the similarity values higher than the one expected by chance alone (Pearson, 2013). Currently several different search algorithms have been applied for computing sequence similarity, e.g. BLAST (Altschul et al., 1990), FASTA (Lipman and Pearson, 1985), PARALIGN (Rognes, 2001), PSI-BLAST (Altschul et al., 1997), etc., all of which produce statistical estimates ensuring that the detected similarity is correlated with protein structure. A structure is more associated to function than a sequence (Rost, 1999). The explicit connection between expectation value (e-value) of aligned sequences and structural similarities of their protein products through strong correlation be-

tween e-values smaller than 10^{-3} and protein structural conformations was demonstrated by Rost (1999).

"Expectation" (commonly known as the e-value) is a statistical value reflecting the expected number of times a computed alignment score would occur by chance alone after thousands or millions of search queries performed (the number of cycles is usually defined by the size of the database) (Pearson, 2013). It depends on several parameters, one of which is the underlying distribution of alignment scores. It has been observed that aligning unrelated sequences produces score values that are indistinguishable from those computed when random sequences are aligned. This information has been utilized for modelling the extreme value distribution (Gumbel, 2013) upon which statistical significance to the observed e-value is determined (Karlin and Altschul, 1990). Therefore, by parsimony, the observed statistically significant sequence similarity, through homology, implies common ancestry.

Ancestral gene (protein) family reconstruction strategies utilise Dollo's parsimony model (Farris, 1977). The model considers only the information regarding presence and absence of families in each leaf of a given species tree. Moreover, it allows multiple family loss events to take place, with only one gain event occurring, thus accordingly inference of the origin of a given family is by definition traced to MRCA of all extent species containing that family (at least one family member). Traditionally, family computation depends on sequence clustering (Walsh and Stephan, 2001). Currently a variety of clustering strategies have been proposed spanning from those utilizing naïve exponential algorithms (clique, vertex cover (Garey and Johnson, 1979)), to those implementing different heuristic solutions (reviewed in (Schaeffer, 2007) some based on k-mer frequency strategies (Hauser et al., 2013; Li and Godzik, 2006) and others either on local (Edgar, 2010) or global (Sakarya et al., 2008) alignments. Given no prior information regarding phylogeny, identification starts by an all-vs-all sequence similarity search in which homologs across all queried species representatives are identified (Librado et al., 2012; Ptitsyn and Moroz, 2012; Sakarya et al., 2008; Walsh and Stephan, 2001). Once the family is calculated, using phylogeny information, MRCA of that family is established and losses are detected by the absence of a family member in a given extent lineage. However, such approach is computationally expensive since adding the number

of genomes to the calculation quadratically increases the runtime preformance (all-vs-all), thus limiting the ancestral genome reconstruction to only a "hand-full" of genomes.

Dollo's parsimony is also central for phylostratigraphy (Domazet-Lošo et al., 2007). This method is used to calculate the origin of each gene from a given query species genome (genome used in the analysis) by locating the most distantly related species in which a homolog to that gene can be identified. Furthermore, since homologs are genes belonging to the same family (as previously explained), the absolute origin of that particular family can be traced down to a specific lineage splitting event. Homology computation in phylostratigraphy is carried out using BLAST with a significance threshold (e-value) set to 10^{-3}. Genes from different or same family that have been traced to the same point of origin, are clustered into so-called "age groups", referred to as phylostrata, where "age" (phylostrata) is defined with respect to the underlying query species phylogeny (ergo the method is a tree based method). As an example consider the computation process depicted in Figure 1.2.

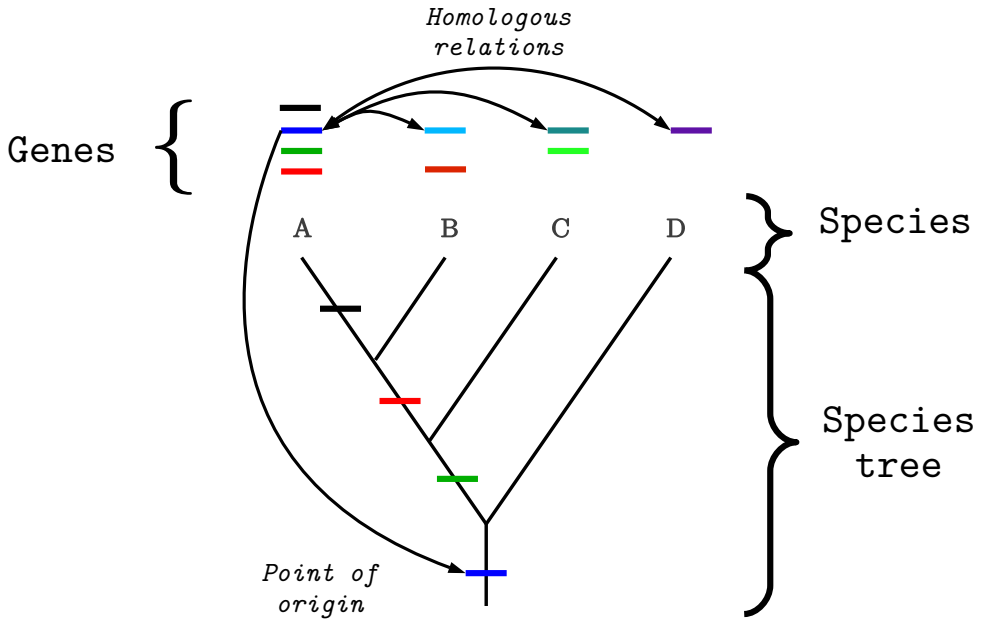


Fig. 1.2 Computing the emergence point of genetic information (gene gain) present in query species (A). Computation is carried out by detecting a homolog to a query species gene in the most distantly related species. Once a homologous relationship has been established the origin (gain) is traced to the most recent ancestor of those two species lineages (for blue gene that is the most recent common ancestor of lineages A and D).

The first step in the computation is to define the evolutionary relationships between species. In order to do that *a priori* phylogeny information is required. According to the example, given a query species A and a set of subject species (B, C, D) the computation begins by selecting a single gene from A and locating the most distantly related genome in which a homolog to that particular query species gene exists. This information implies the presence of a homolog of both descendant species in their common ancestor, thus assigning a gene from query species A to a period before the speciation of corresponding lineages took place, tracing its origin to a specific point in the evolutionary history of that lineage. By repeating the process for each gene in the genome the obtained final result is a set of subsets of genes traced to different points of origin. The set is referred to as "phylostratigraphy map" (Domazet-Lošo et al., 2007). Note that such gene family gain event (GFGE) represents a type of novelty in genome space and thus reflects the adaptive forces acting on the genome at that point in the evolution. In case protein coding genes are utilized (as done in practice) by analogy, the formation event is accordingly termed the protein family gain event. Historically, phylolstratigraphy posed a novel, computational approach for long term evolutionary analysis conducted on genomes. It was the first method able to cope with large amounts of data in a relatively short amount of time (computationally "cheap") tracing the origin of a given gene (protein) to its earliest point in evolution thus separating itself form other MRCA strategies described in the text above. However, the initial design of the method focused primarily on tracing the origin of acquired novelties, thus only depicting the evolutionary changes from a perspective of one focal species. For comparative analysis the phylostratigraphic approach heavily relies on the repetitive process of gene age computation performed for each genome separately. Moreover, phylostratigraphy relies on Dollo's parsimony (one gain and many losses) the method can only be seen as "lower bound" approximation on family emergence and thus needs to be treated as such.

Alternatively, a more powerful strategy for age computation utilizes Wagner's parsimony (Swofford and Maddison, 1987). Under this model, which allows for multiple gains and losses, the number of genes associated to a family is taken into account. By incorporating the information regarding family contractions and expansions through weight assignment the

approach is able to correct possible age miscalculations in the process (Capra et al., 2013). However, increase in accuracy comes with a price, rendering computation based on Wagner's parsimony much more difficult to perform and therefore excluding any large scale studies, such as those carried out by phylostratigraphy, to take place.

Reconciliation strategies (Doyon et al., 2011; Nakhleh et al., 2009) provide a third approach to gain-loss computation and ancestral family reconstruction. These strategies also require a species tree. However, in addition to it, they rely on the information obtained from a gene tree (reconstructed from sequences of the investigated gene family). Using these two datasets, reconciliation captures the emerging inconsistencies between trees and utilizes it to derive the most parsimonious scenario best describing the evolution of ancestral genes and species (Doyon et al., 2011). Reconciliation strategies are also more computationally expensive than those utilizing Dollo's parsimony (Capra et al., 2013). Therefore, conceptual tractability and computational efficiency makes the methods utilizing Dollo's parsimony more attractive than the rest.

In principle, all gain-loss and reconciliation strategies boil down to gene age inference utilizing some form of parsimony criterion (Capra et al., 2013). As opposed to it, alternative probabilistic models for inferring gain and loss of genes (families) exist (Akerborg et al., 2009; Csuros, 2010; De Bie et al., 2006; Gorecki et al., 2011; Rasmussen and Kellis, 2012). These strategies tend to make inferences more reliable in case when a large number of investigated events are considered (Capra et al., 2013). However, they come with a price, a significant increase in computational runtime, orders of magnitude higher than the aforementioned parsimony and reconciliation strategies and therefore are usually restricted to global pattern predictions (Hahn et al., 2007a).

Reconstruction of ancestral genome (proteome) content is a keys step toward understanding genome structure and function. The information acquired in the process can further be utilized for testing various hypotheses regarding evolutionary patterns such as the effect of Black Queen on animals plants and fungi (Morris, 2015) and genome complexity patterns (McShea, 2015; McShea and Hordijk, 2013; Wolf and Koonin, 2013), One of the central tenets of evolutionary biology addresses the pattern and trend associated with biolog-

ical complexity and its prevailing evolutionary mode (Adami, 2002; Adami and Cerf, 2000; Adamowicz et al., 2008; Bonner, 1988; Doolittle, 2012; Fitch and Ayala, 1995; Gould, 1997; Gould and Eldredge, 1977; Koonin, 2005, 2011; Lynch and Conery, 2003; McShea, 2015; McShea and Hordijk, 2013; McShea, 1996, 2000, 2001, 2002; Simon, 1969; Simpson, 1944; Wolf and Koonin, 2013) General discussion on the topic revolves around quantifying biological complexity (Adami, 2002; Adami and Cerf, 2000; Koonin, 2005, 2011) and whether the complexity has increased (Lynch and Conery, 2003; Simpson, 1944), barely changed (Gould, 1997; Gould and Eldredge, 1977) or even exhibits a distinct pattern (McShea and Hordijk, 2013; Wolf and Koonin, 2013) in the course of species evolutionary history. Quantifying complexity proved to be notoriously difficult when applied to biological systems (McShea and Hordijk, 2013; Szathmáry et al., 2001; Wolf and Koonin, 2013). Many attempts have been made to apply solutions and concepts from mathematics and computer science by reducing it to computational complexity. However, such an approach rendered the embryonic development as a computational solution consisting of a finite number of developmental steps (Szathmáry et al., 2001), which at best might be considered as loosely defined approximations, in a sense that they do not reflect what one might intuitively perceive as complex in biology. Biological systems are adaptive which means that when a system goes through a transition, the elements of that system change with it and the change is not additive thus directly ruling out any partial differential analysis (mathematical approach based on the assumption on additivity) one might use for describing a given system (Holland, 1996). Nevertheless, various different proxies have been utilized for quantifying biological complexity such as counting the number of cell types (Valentine et al., 1994), organizational levels (McShea, 2001), nucleotides, genes and transcription factors (Szathmáry et al., 2001), somehow always failing to accurately reflect what is perceived as complex in biology.

Unlike biological (organismal) complexity, genomic complexity is perceived as a property that is much easier to quantify due to distinct individual units it consists of (genes, gene families). With the increasing number of genomes becoming available, the information required for detailed reconstruction of ancestral genome content in terms of genes and gene families, and thus patterns associated to genomic complexity as species evolved, converts

from currently speculative (philosophical) discussion on the topic, into an evidence supported research. In a paper published by Wolf and Koonin (2013), a novel model of long-term genome evolution called biphasic model was proposed. According to this model, genome complexity has been assumed to periodically change in the course of evolution through periods of short erratic bursts in which complexity increases, followed by long stretches of genomic complexity reduction. Main proxy for quantifying genome complexity was suggested to be the number of genes conserved at a given evolutionary distance (Wolf and Koonin, 2013). Using genes as proxies (under the assumption that a gene is considered as a unit of information (Bromham, 2016)) certainly has its footing in Shannon's information theory (Shannon and Weaver, 1949). However, in case of ancestral gene content reconstruction through homologous sequence identification, each ancestral state is characterized by a set of homologous genes, rather than a single one thus in such cases using genes as proxies might lead to a redundant, artificially increased complexity values.

Currently no comprehensive study comparable to that conducted by Kunin et al. (2005a) involving a large number of eukaryotic lineages exists, given the high computational cost associated with such analysis. Thus the primary goal of this thesis is to reconstruct the gene family content of ancestral species across large number of eukaryotes, 383 to be more precise, utilizing family gain and loss calculations. To alleviate the aforementioned computation problem, it was necessary to develop an entire set of new theoretical solutions and technical strategies. These strategies were then applied to collected datasets of sequence information in order to investigate macroevolutionary patterns of changes in genome complexity across various evolutionary lineages. Moreover, to tackle the problem of redundancy associated with genes, when used as proxies for genome complexity calculations, I introduce a novel measure of ancestral genome complexity, based on the number of gene (protein) families conserved at a given evolutionary distance. I define the measure from automata-theoretic point of view and establish its footing in Shannon's information theory. By using families as proxies, I further investigate macroevolutionary patterns associated with changes in ancestral genome complexities, revealing a new and so far undocumented and un-conjectured global bipartite mode of evolution associated with increase in genomic complexity dominating a

better part of Proterozoic eon, followed by a decrease observed throughout Phanerozoic eon. Aside from this global trend, some evolutionary periods exhibit specific profiles with exceptionally high family gain or loss rates. In a more detailed analysis I show that these periods correspond to known key evolutionary transition periods.

Finally, I discuss my results as well as benefits and limitations of tools and algorithms developed in this study and give possible directions for future work on the subject.

Chapter 2

Methods

The chapter is divided into several main sections as follows:

1. The input data is introduced in the first section of this chapter and it includes a description of phylogeny re-construction process followed by a description of a flat file sequence database, custom made to meet the requirements posed by the computational challenges in this work.

2. Algorithms and computational strategies based on novel theoretical results applied for gene family gain/loss event calculations (gain and loss of gene families) are described in the second section of this chapter.

3. The third and fourth sections deal with genome complexity calculations and statistical data analysis relevant for final evaluation of generated results.

4. The final fifth section summarizes the entire computational process, describing the entire procedure staring from input data introduced in the first, to complexity estimations and statistical calculations described in the fourth section.

2.1 Input Data

2.1.1 Species Phylogeny

The tree of life (phylogeny) has long served as a useful structure for describing evolutionary relations between life-forms. It is a hierarchical tree-like structure in which each leaf-node represents one of the currently existing species, whereas inter-nodes (branching points) refer to their common ancestors. It is important to note that tree-like representations, such as the one used in defining phylogeny relations, imply a type of vertical inheritance, (transfer) of information form parents to their offspring and thus legitimizing any ancestral reconstruction analysis based on any heritable information (e.g. genetic material) contained in current species.

Defining phylogenetic relations between species (organisms) has been for a long time based mainly on a set of morphological characters (anatomical structures, physiological processes). As such some of the relations, especially those where analogous characters were used for classification purposes, have been incorrectly defined, placing different organisms into shared groups among which no direct common ancestry has ever existed. This all changed with the availability of molecular data like small subunit ribosomal RNA (SSU rRNA) molecule (Sogin et al., 1986). Using it for classification purposes, initiated the first major revision of phylogenetic relations among species. The molecule was used in various comparative studies in which the difference in sequence composition was used as a primary "character" in classification. As a result, a completely new set of relations, especially among single cell and basal multicellular eukaryotes, had been (re)defined (Taylor, 1978). However, needless to say such an approach relies upon several assumptions like steady mutation rate and high sequence conservation connected to rRNA molecules that gave rise to molecular clock hypothesis (Bromham and Penny, 2003; Kumar, 2005), which still is a source of a debate when it comes to inferring relationships among distantly related species. Therefore, with the reduction in sequencing cost, the rRNA data today is frequently accompanied with the information from protein coding genes and large datasets including whole proteome surveys. Together this information led to the second major revisions of the phylogeny relations,

based upon which the phylogeny reconstruction presented in this section has been created.

The primary challenge here is to modify established relations (focusing only on eukaryote species) taken from NCBI Taxonomy database according to most recent literature evidence. Moreover, the species in question are reduced to only those with the available genome sequence information and thus only a non-redundant set of internal branching nodes has been considered in the reconstruction process, such that inter-nodes which lack the available species genome in at least two of their descendent lineages have been removed together with all their subsequent nodes. To illustrate this consider the following situation. Let A (species genome available) and B (species genome not available) be two species and a common ancestor C, descendent of an older ancestral point D from which another descendent lineage F has branches out (as illustrated in Figure 2.1), note that the genome sequence for species F is available. In such case the ancestral point C together with species node B is eliminated from a tree, connecting nodes A and D together, making F a sister node to A. This process has been applied to each existing case.

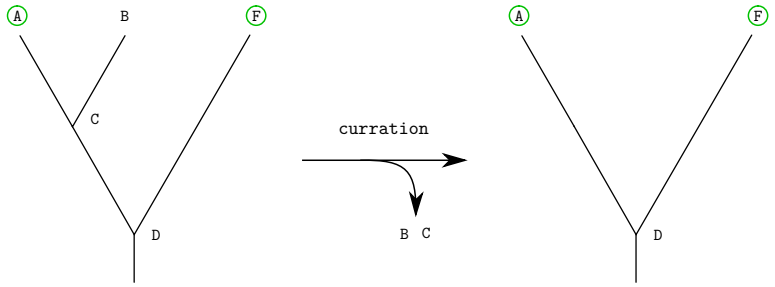
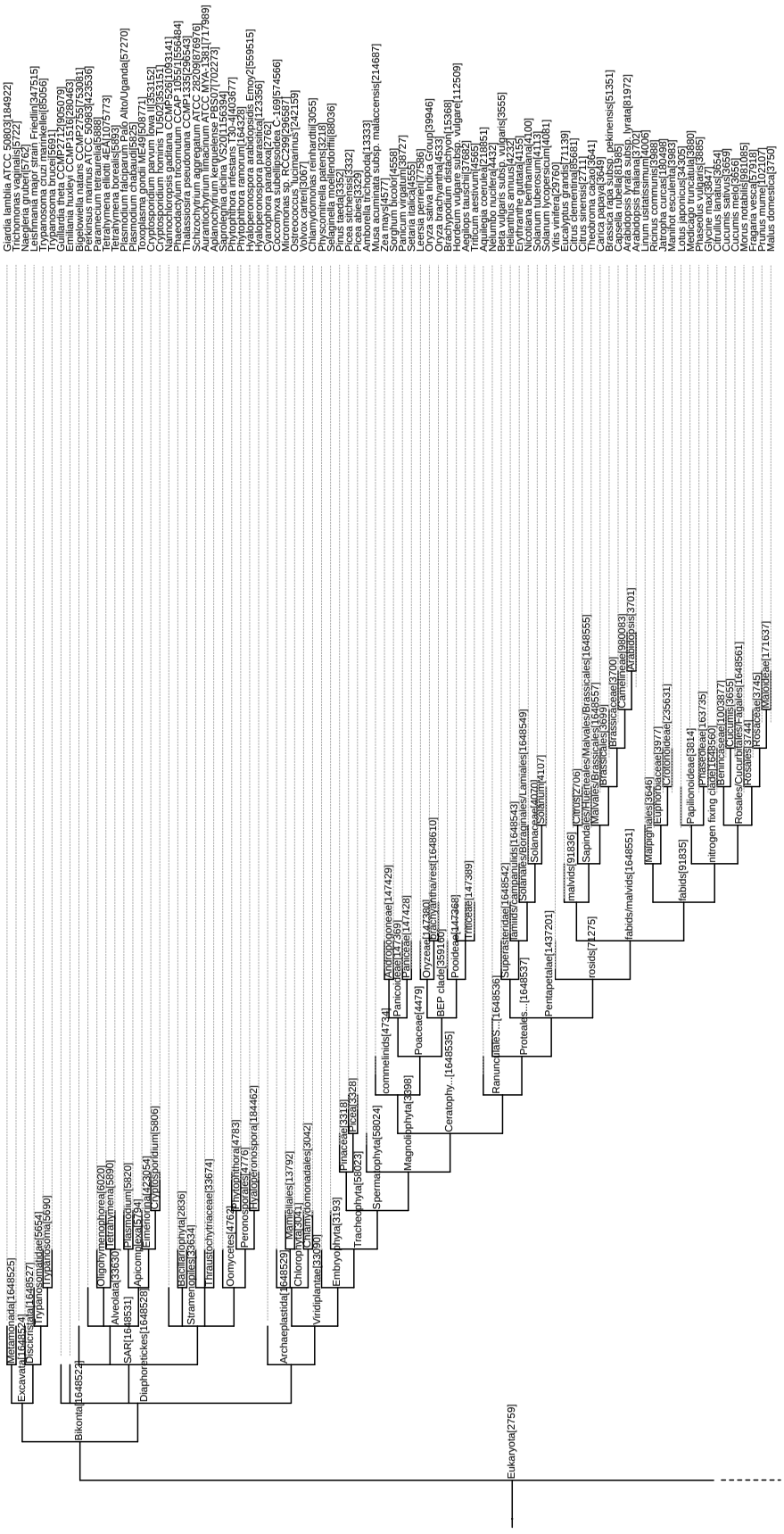


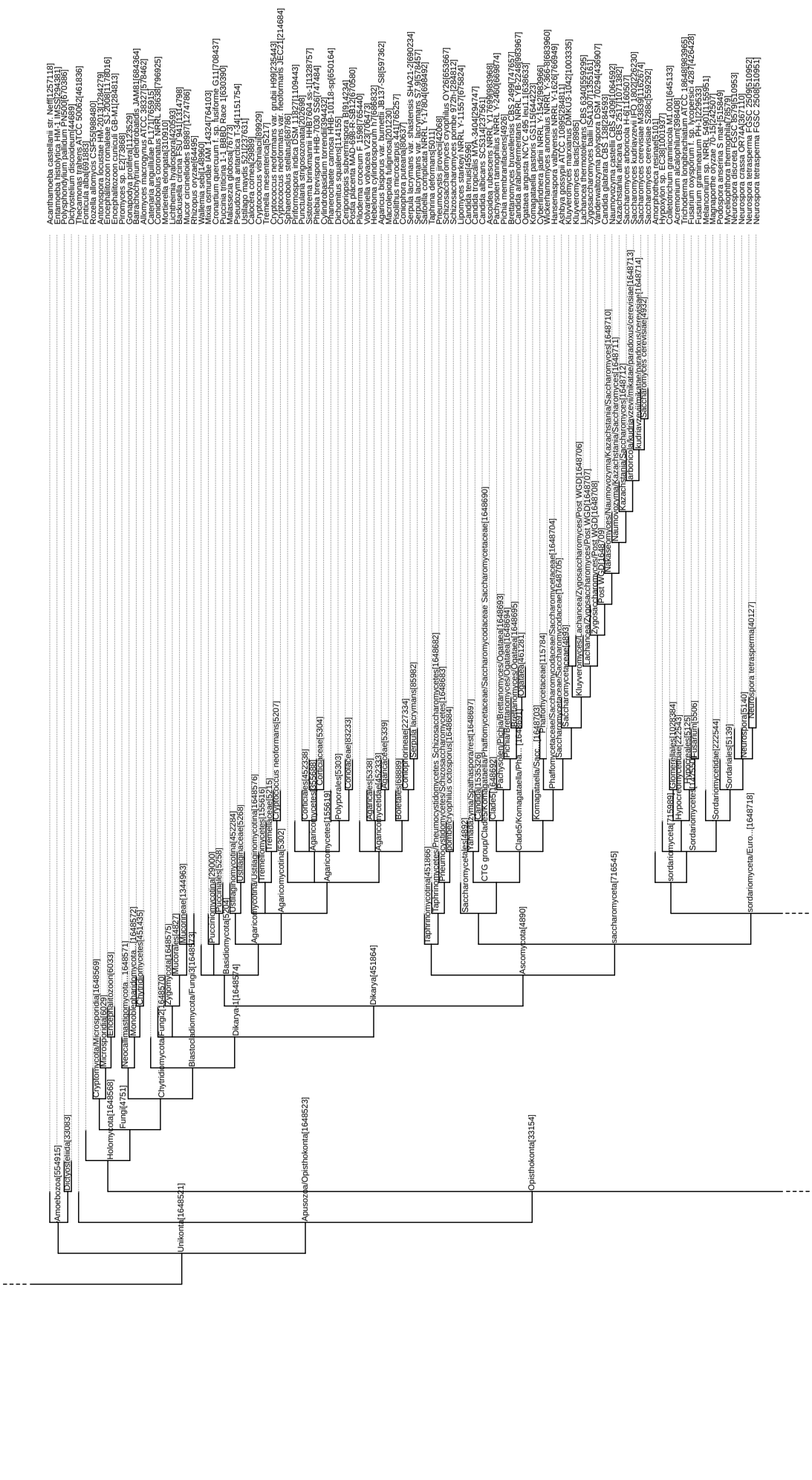
Fig. 2.1 The process of eliminating redundant ancestral and existing species nodes from species phylogeny tree. Green circles label current species with available sequenced genome information. If at least two descendent lineages from an ancestral point (inter-node) do not have available genome sequence information, the ancestral node and all species node lacking that information are eliminated form the tree.

As a final result, the generated species tree consists of 734 nodes, of which 383 are leaf-nodes, representing current species and 351 are internal nodes defining their evolutionary relations. Figure 2.2 depicts the result of this phylogeny reconstruction process, with references supporting the illustrated modifications summarized in Table 2.1.

Table 2.1 Summary of literature references supporting the phylogeny relations depicted in Figure 2.2.

Group	Literature reference
Amebozoa	Fiore-Donno et al. (2010)
Arthropoda	Giribet et al. (2012)
Birds	Jarvis et al. (2014); Jetz et al. (2012)
Chordata	Delsuc et al. (2006, 2008)
Ctenophora	Moroz et al. (2014)
	Wiegmann et al. (2011)
Diptera	Clark et al. (2007); Singh et al. (2009)
	Seetharam et al. (2013)
	Adl et al. (2012); Burki (2014)
Eukaryotes	Cavalier-Smith (2009); Ruiz-Trillo et al. (2007)
	Adl et al. (2005); Brown et al. (2012)
Fishes	Near et al. (2012)
	Kurtzman and Robnett (2013)
	Haag et al. (2014); Leonard and Richards (2012)
Fungi	Ebersberger et al. (2012); James et al. (2013)
	Capella-Gutiérrez et al. (2012)
	Gazis et al. (2012); J.W et al. (2006)
Fungi/Opithokonta	Torruella et al. (2012)
Hominidae	Bradley (2008); Wood and Richmond (2000)
Hymenoptera	Johnson et al. (2013); Ward (2014)
Insects	Misof et al. (2014); Peters et al. (2014)
	Sadd et al. (2015); Trautwein et al. (2012)
Mammals	O'Leary et al. (2013); Reis et al. (2012)
	Meredith et al. (2011); Song et al. (2012)
Metazoa	Dunn et al. (2014); Nosenko et al. (2013)
	Edgecombe et al. (2011)
Opisthokonta	Papsa et al. (2013); Torruella et al. (2012)
	Brown et al. (2009)
	Cavalier-Smith and Chao (2010)
Plants	Geering et al. (2014)
	Soltis et al. (2011)
	Chase and Reveal (2009)
	Burki et al. (2012); Moore et al. (2010)
	Refulio-Rodriguez and Olmstead (2014)
	Albert et al. (2013)
Porifera	Worheide et al. (2012)
Spiralia	Struck et al. (2014)


Giardia lamblia ATCC 50803[184922]
Trichomonas vaginalis[5722]
Naegleria gruberi[5762]
Leishmania major strain Friedlin[347515]
Trypanosoma cruzi marinkellei[85056]
Trypanosoma brucei[5691]
Guillardia theta CCMP2712[905079]
Emiliania huxleyi CCMP1516[280463]
Bigelowiella natans CCMP2755[753081]
Perkinsus marinus ATCC 50983[423536]
Paramecium tetraurelia[5888]
Tetrahymena elliotti 4EA[1075773]
Tetrahymena borealis[5893]
Plasmodium falciparum 3D7[36329] AltoVuganda[57270]
Plasmodium chabaudi[5825]
Toxoplasma gondii ME49[508771]
Cryptosporidium parvum Iowa II[353152]
Chromera velia CCMP2878[1169540]
Nannochloropsis gaditana CCMP526[1093141]
Phaeodactylum tricornutum CCAP 1055/1[556484]
Thalassiosira pseudonana CCMP1335[35128]
Aureococcus anophagefferens[44056]
Aplanochytrium kerguelense PBS07[702273]
Schizochytrium aggregatum[84654]
Phytophthora infestans T30-4[403677]
Phytophthora ramorum[164328]
Hyaloperonospora arabidopsidis[112509] Emoy2[559515]
Pythium ultimum DAOM BR144[431595]
Cyanophora paradoxa[2762]
Cocomyxa subellipsoidea C-169[574566]
Micromonas commoda[296587]
Ostreococcus 'lucimarinus'[242159]
Volvox carteri[3067]
Chlamydomonas reinhardtii[3055]
Physcomitrella patens[3218]
Selaginella moellendorffii[88036]
Pinus taeda[3352]
Picea glauca[3332]
Picea abies[3329]
Amborella trichopoda[13333]
Musa acuminata subsp. malaccensis[214687]
Sorghum bicolor[4558]
Panicum virgatum[38727]
Zea mays[4577]
Leersia perrieri[77586]
Oryza sativa Indica Group[39946]
Oryza brachyantha[4533]
Brachypodium distachyon[15368]
Hordeum vulgare subsp. vulgare[112509]
Aegilops tauschii[37682]
Triticum aestivum[4565]
Ananas comosus[4615]
Nelumbo nucifera[4432]
Beta vulgaris subsp. vulgaris[3555]
Helianthus annuus[4232]
Erythranthe guttata[4155]
Nicotiana benthamiana[4100]
Solanum tuberosum[4113]
Solanum lycopersicum[4081]
Vitis vinifera[29760]
Eucalyptus grandis[71139]
Populus trichocarpa[3694]
Citrus sinensis[27110]
Theobroma cacao[3641]
Carica papaya[3649]
Gossypium raimondii[29730] gekinensis[51351]
Capsella rubella[81985]
Arabidopsis lyrata subsp. lyrata[81972]
Arabidopsis thaliana[3702]
Linum usitatissimum[4006]
Ricinus communis[3988]
Jatropha curcas[180498]
Manihot esculenta[3983]
Lotus japonicus[34305]
Medicago truncatula[3880]
Glycine max[3847] [3885]
Cicer arietinum[3827]
Cucumis sativus[3659]
Morus notabilis[981085]
Prunus mume[102607]
Malus domestica[3750]
Eukaryota[2759]
Bikonta[1648522]
Diaphoretickes[1648528]
SAR[1648531]
Alveolata[33630]
Oligohymenophorea[66029]
Tetrahymena[5909]
Plasmodium[5820]
Apicomplexa[5794]
Cryptosporidium[5806]
Stramenopiles[33634]
Bacillariophyta[2836]
Chlorophyta[3041]
Mamiellales[13792]
Chlamydomonadales[3042]
Viridiplantae[33090]
Embryophyta[3193]
Tracheophyta[58023]
Spermatophyta[58024]
Pinaceae[3318]
Picea[3328]
Magnoliophyta[3398]
commelinids[4734]
Poaceae[4479]
Oryzeae[147389]
BEP clade[35910]
Pooideae[147389]
Triticeae[147389]
Ceratophy...[1648535]
Ranunculales...[1648536]
Proteales...[1648537]
Pentapetalae[1437201]
rosids[71275]
malvids[91836]
Malvales[27706]
Brassicales[1648555]
Brassicaceae[3699]
Camelineae[980083]
Arabidopsis[3701]
fabids[91835]
Malpighiales[3646]
Euphorbiaceae[3977]
Crotonoideae[235631]
Papilionoideae[3814]
nitrogen fixing clade[1228566]
Fabales[3803]
Cucurbitales[3744]
Rosales[3745]
Rosaceae[3745]
Malodeae[171637]


Acanthamoeba castellanii str. Neff[1257118]
Entamoeba histolytica HM-1:IMSS[294381]
Entamoeba dispar SAW760[370386]
Dictyostelium discoideum[44689]
Thecamonas trahens ATCC 50062[461836]
Fonticula alba[691883]
Rozella allomycis CSF55[988480]
Antonospora locustae HM-2013[1284279]
Encephalitozoon cuniculi GB-M1[1178016]
Encephalitozoon cuniculi GB-M1[284813]
Piromyces sp. E2[73868]
Gonapodya prolifera[1123529]
Batrachochytrium dendrobatidis JAM81[684364]
Allomyces macrogynus ATCC 38327[578462]
Catenaria anguillulae PL171[765815]
Capsaspora owczarzaki ATCC 30864[192566]
Mortierella elongata[310910]
Lichtheimia hyalospora[4206593]
Backusella circina FSU 941[1447798]
Mucor circinelloides B8987[1274786]
Rhizopus oryzae[64495]
Malassezia sp.[76770]
Malassezia globosa CBS 7966[1324][764103]
Cronartium quercuum f. sp. fusiforme G11[708437]
Puccinia triticina 1-1 BBBD Race 1[630390]
Puccinia graminis f. sp. tritici[56615]
Pseudozyma antarctica T-34[1151754]
Ustilago maydis 521[237631]
Cryptococcus gattii[36913]
Cryptococcus vishniacii[99929]
Tremella mesenterica[52217]
Cryptococcus neoformans var. grubii H99[235443]
Cryptococcus neoformans var. neoformans JEC21[214684]
Schizosaccharomyces pombe[4896]
Sphaerobolus stellatus[68786]
Trametes versicolor FP-101664 ss-1[1328757]
Sistotrema brinkmannii HHB7604 ss-1[1328757]
Phlebia brevispora HHB-7030 ss-8[747484]
Phanerochaete carnosa HHB-10118-sp[650164]
Dichomitus squalens[114155]
Punctularia strigosa HHB-11173 SS5[914234]
Postia placenta MAD-698-R-SB12[670580]
Piloderma croceum F 1598[765440]
Hebeloma cylindrosporum h7[686832]
Agaricus bisporus var. burnettii JB137-S8[597362]
Macrolepiota fuliginosa[2017260]
Coniophora puteana RWD-64-598 SS2[741257]
Coniophora puteana[80637]
Serpula lacrymans var. shastensis SH421-2[690234]
Serpula lacrymans var. lacrymans S7.9[578457]
Saitoella complicata NRRL Y-17804[698492]
Taphrina deformans[59311]
Taphrina deformans PYCC 5710[1258608]
Schizosaccharomyces cryophilus OY26[6536667]
Schizosaccharomyces pombe 972h-[284812]
Candida tenuis[45596]
Candida tropicalis MYA-3404[294747]
Candida albicans SC5314[237561]
Candida albicans WO-1[1173999983968]
Pachysolen tannophilus NRRL Y-2460[669874]
Pichia membranifaciens[49426]
Candida arabinofermentans NRRL YB-2248[983967]
Ogataea angusta NCYC 495 leu1.1[638633]
Cyberlindnera jadinii NRRL Y-1542[983966]
Wickerhamomyces anomalus NRRL Y-366-8[683960]
Komagataella pastoris GS115[644223]
Ashbya gossypii ATCC 10895[284811]
Kluyveromyces marxianus DMKU3-1042[1003335]
Kluyveromyces lactis[28985]
Candida glabrata CBS 138[1001937]
Zygosaccharomyces bailii ISA1307[1355161]
Vanderwaltozyma polyspora DSM 70294[436907]
Torulaspora delbrueckii[45914]
Naumovozyma castellii CBS 4309[1064592]
Kazachstania africana CBS 2517[1071382]
Kazachstania naganishii CBS 8797[1071383]
Saccharomyces kudriavzevii IFO 1802[226230]
Saccharomyces cerevisiae M3839[1162674]
Saccharomyces cerevisiae S288c[559292]
Amorphotheca resinae[5101]
Hypoxylon sp. EC38[1001937]
Colletotrichum graminicola M1.001[645133]
Aspergillus niger[5061]
Trichoderma longibrachiatum ATCC 18648[983965]
Fusarium oxysporum f. sp. lycopersici 4287[426428]
Neurospora crassa OR74A[367110]
Melanconium sp. NRRL 54901[1115951]
Magnaporthe oryzae 70-15[242507]
Mycosphaerella graminicola[336722]
Mycelophthora thermophila[78579]
Neurospora discreta FGSC 8579[5110953]
Neurospora crassa OR74A[367110]
Neurospora tetrasperma FGSC 2509[5110952]
Neurospora tetrasperma FGSC 2508[5110951]

Herpotrichiellaceae[43219]
Eurotiomycetes[147545]
Eurotiales[5042]
Aspergillaceae[1131492]
Penicillium[5073]
Eurotiomycetidae[451871]
Onygenales[33183]
mitosporic Onygenales[34383]
Coccidioides[5500]
Eurotiomycetes/Xylono...[1648719]
Dothideomycetidae[451867]
Capnodiales[134362]
Mycosphaerellaceae[93133]
Dothideomycetes[147541]
Botryosphaeriales[451869]
Botryosphaeriaceae[45131]
Pleosporales[92860]
Pleosporineae[715340]
Pleosporaceae[28556]
Ichthyophonida[198625]
Holozoa[1648563]
Choanoflagellida[28009]
Filozoa[1648564]
Ctenophora[10197]
Metazoa/Choanoflagellida[1648565]
Haplosclerida[6049]
Porifera[6040]
Homoscleromorpha/Calcerea[1648587]
Leucosolenida[86011]
Metazoa[33208]
Hexacorallia[6102]
Cnidaria[6073]
Acropora[6127]
Medusozoa[1648589]
Hydra[6083]
vulgaris/oligactis group[1648605]
Mollusca[6447]
Lophotrochozoa[1206795]
Annelida[6340]
Spiralia[1648590]
Platyhelminthes[6157]
Opisthorchiidae[6196]
Digenea[6179]
Schistosoma[6181]
Eumetazoa[6072]
Tylenchina/Panagrolaimoidea[1648593]
Tylenchida[6300]
Nematoda[6231]
Protostomia[33317]
Spirurina[1648592]
Onchocercidae[6296]
Loa/Dirofilaria/Onchocerca[1648637]
Chromadorea[119089]
Strongylida[6308]
Ancylostomatidae[33278]
Rhabditina[1648581]
Rhabditida[6236]
Rhabditoidea[55879]
Caenorhabditis[6237]
Caenorhabditis supergroup[1648594]
Elegans group[1648595]
brenneri/remanei/briggsae[1648597]
remanei/briggsae[1648598]
Ecdysozoa[1206794]
Arthropoda[6656]
Mandibulata[197563]
Pancrustacea[197562]
Neoptera[33340]
Hymenoptera[7399]
Apocrita[7400]
Apoidea[34735]
Apis[7459]
Phthiraptera/Endopterygota[1648626]
Aculeata[7434]
Formicidae[36668]
Dolichoderinae/Formicinae/Myrmicinae[1648629]
Formicinae/Myrmicinae[1648630]
Myrmicinae[34695]
Endopterygota[33392]
Cnidaria/Bilateria[1648588]
Obtectomera[104431]
Nymphalidae[33415]
Coleo.../Amphi...[1648627]
Amphi.../Diptera[1648628]
Anopheles[7164]
Culicidae[7157]
Culicinae[43817]
Diptera[7147]
Cyclorrhapha[480117]
grimshawi group[48302]
repleta group/virilis group[1648631]
Drosophila[7215]
Sophophora[32341]
pseudoobscura subgroup[32358]
melanogaster group/obscura group[1648632]
melanogaster group[32346]
erecta/yakuba[1648633]
melanogaster subgroup[32351]
melanogaster/simulans/sechellia[1648634]
simulans/sechellia[1648635]
Bilateria[33213]
Cladophialophora psammophila CBS 110553[1182543]
Coniosporium apollinis CBS 100218[1168221]
Talaromyces stipitatus ATCC 10500[441959]
Aspergillus acidus[929029]
Penicillium chrysogenum[5076]
Penicillium brevicompactum[5074]
Blastomyces dermatitidis ATCC 26199[447095]
Uncinocarpus reesii 1704[336963]
Paracoccidioides sp. 'lutzii' Pb01[502779]
Coccidioides posadasii str. Silveira[443226]
Coccidioides immitis RS[246410]
Myriangium duriaei CBS 260.36[1168546]
Piedraia hortae var. hortae[147573]
Dothistroma septosporum NZE10[675120]
Sphaerulina populicola[215467]
Aplosporella prunicola CBS 121167[1176127]
Neofusicoccum parvum UCRNP2[1287680]
Macrophomina phaseolina MS6[1126212]
Botryosphaeria dothidea[55169]
Lophiostoma macrostomum[372055]
Phaeosphaeria nodorum SN15[321614]
Bipolaris oryzae ATCC 44560[930090]
Pyrenophora teres f. teres 0-1[861557]
Alternaria brassicicola[29001]
Sphaeroforma arctica JP610[667725]
Creolimax fragrantissima[470921]
Capsaspora owczarzaki ATCC 30864[595528]
Salpingoeca rosetta[946362]
Monosiga brevicollis[81824]
Pleurobrachia bachei[34499]
Mnemiopsis leidyi[27923]
Amphimedon queenslandica[400682]
Ephydatia muelleri[6052]
Oscarella carmela[386100]
Leucosolenia complicata[433461]
Sycon ciliatum[27933]
Trichoplax adhaerens[10228]
Nematostella vectensis[45351]
Acropora digitifera[70779]
Acropora millepora[45264]
Aurelia aurita[6145]
Hydra viridissima[6082]
Hydra oligactis[6088]
Hydra vulgaris[6087]
Lottia gigantea[225164]
Crassostrea gigas[29159]
Capitella teleta[283909]
Helobdella robusta[6412]
Echinococcus multilocularis[6211]
Clonorchis sinensis[79923]
Opisthorchis viverrini[6198]
Schistosoma mansoni[6183]
Schistosoma japonicum[6182]
Trichinella spiralis[6334]
Strongyloides ratti[34506]
Bursaphelenchus xylophilus[6326]
Meloidogyne incognita[6306]
Ascaris suum[6253]
Wuchereria bancrofti[6293]
Loa loa[7209]
Onchocerca volvulus[6282]
Pristionchus pacificus[54126]
Oesophagostomum dentatum[61180]
Ancylostoma ceylanicum[53326]
Necator americanus[51031]
Heterorhabditis bacteriophora[37862]
Caenorhabditis angaria[860376]
Caenorhabditis japonica[281687]
Caenorhabditis elegans[6239]
Caenorhabditis brenneri[135651]
Caenorhabditis remanei[31234]
Caenorhabditis briggsae[6238]
Tetranychus urticae[32264]
Strigamia maritima[126957]
Daphnia pulex[6669]
Acyrthosiphon pisum[7029]
Pediculus humanus corporis[121224]
Orussus abietinus[222816]
Ceratosolen solmsi marchali[326594]
Microplitis demolitor[69319]
Megachile rotundata[143995]
Bombus terrestris[30195]
Apis florea[7463]
Apis mellifera[7460]
Harpegnathos saltator[610380]
Linepithema humile[83485]
Camponotus floridanus[104421]
Wasmannia auropunctata[64793]
Atta cephalotes[12957]
Tribolium castaneum[7070]
Bombyx mori[7091]
Heliconius melpomene[34740]
Danaus plexippus[13037]
Anopheles darlingi[43151]
Anopheles gambiae[7165]
Culex quinquefasciatus[7176]
Aedes aegypti[7159]
Megaselia scalaris[36166]
Drosophila grimshawi[7222]
Drosophila virilis[7244]
Drosophila mojavensis[7230]
Drosophila willistoni[7260]
Drosophila pseudoobscura pseudoobscura[46245]
Drosophila persimilis[7234]
Drosophila ananassae[7217]
Drosophila yakuba[7245]
Drosophila erecta[7220]
Drosophila melanogaster[7227]
Drosophila simulans[7240]
Drosophila sechellia[7238]

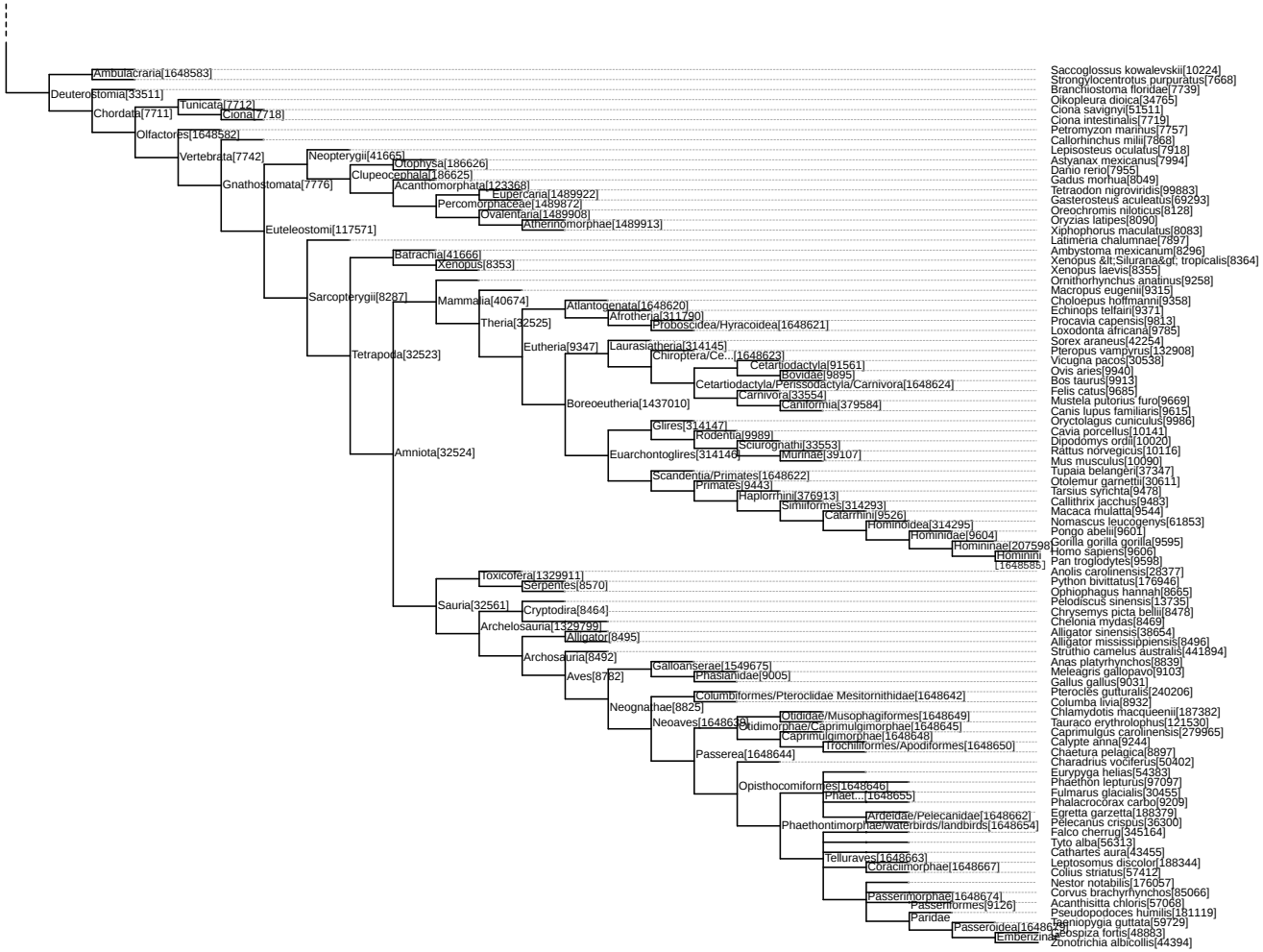


Fig. 2.2 Species tree used in the ancestral genome content reconstruction procedure. The tree contains 383 eukaryote species names of which all are listed on the far right side of the figure. Each taxon is labelled by a unique taxonomy identifier, the list of which can be found in the Appendix A.

2.1.2 Species Genome Information

383 eukaryote genomes mentioned in the previous subsection were collected from various online repositories including NCBI (Pruitt et al., 2012), Ensembl (Flicek et al., 2014), JGI (Nordberg et al., 2014), Compagen (Hemmrich and Bosch, 2008) etc. (the entire list of all repositories and genome sequences retrieved from them, can be found in Table A.3). Although the general practice implies submitting the sequenced and/or annotated genome information to one of the freely available repositories prior to any publication (Benson et al., 2007; Flicek et al., 2014; Nordberg et al., 2014), a substantial fraction of data still remains available from local servers only. This is not a problem if standardized conventions are applied in hosting the data, however, often this is not the case. The nature of data itself, standard management protocols and trailing data structures, simply do not allow many standards to be applied, thus making the data retrieval a rather challenging task.

Therefore, the main objective here was to collect as much data as possible in order to reach a minimum genome density required (at least one genome available in the branching lineage at each ancestral state) for reconstructing ancestral genome content across different evolutionary time periods. It is important to note that the collected sequences are amino-acid and thus a collection corresponding to an individual species represents its proteome information. The reason why amino-acid sequences were used instead of nucleotide ones representing genes, is due to higher sensitivity and sequence conservation required in homology (through sequence similarity) computation. However, when working with proteomes it is important to note that several protein sequences may refer to the same gene since different splicing variants can give rise to different amino-acid translations. Therefore, where the required information is available, each gene was represented by an amino-acid sequence corresponding to the longest splicing variant. This way each amino-acid sequence contains the highest number of domains and is the "best" representative of the underlying gene, making the resulting collection of sequences a set of protein coding genes and thus adequately termed genomes. Therefore, from 774 collected eukaryote genomes 383 were selected as the most representative. In all cases selection process was not carried out with any type of annotation quality criteria in mind. The assumption was that all genomes thus the identified genes were

Table 2.2 *db_200514* database content

	Bacteria	Archaea	Eukarya	Total
Number of sequences	31,402,812	646,350	13,407,071	45,456,233
Number of taxonomy units (species)	8,973	239	774	9,986
Size in GB	15.4	0.3	6.7	22.4

properly annotated and miss-annotations were rare. Missing annotations on the other hand were not to present a serious problem (as shown by the bootstrap analysis in figure 3.25). Lack of an unannotated gene just passes the origin point calculation of that gene family to another candidate hence compensating the lack of annotation.

In addition to those 383 eukaryote genomes, 8,973 Bacterial and 239 Archaeal genomes were included in order to be able to trace the origin of genes emerging before the appearance of eukaryotes. Both Bacterial and Archaeal genomes were first stripped of all redundancy using CD-HIT program (Li and Godzik, 2006). CD-HIT is a fast protein clustering tool designed for joining similar sequences together. It uses a greedy incremental single-linkage clustering strategy based on a similarity computation which is calculated by counting the number of identical k-mers (k-sized sequence fragments) shared between a pair of sequences. Briefly, the computation begins by sorting the sequences in order of decreasing length. The longest sequence then becomes the representative of the first cluster and each remaining consecutive sequence is then compared to it. If the similarity based on k-mer identity turns out to be above a given threshold, the sequence is assigned to a cluster. Otherwise, a new cluster is defined with that sequence selected as a new representative.

Using this program, all 9,212 prokaryote genomes were clustered based on 95% sequence identity threshold value reducing the overall size of the data by ≈ 3.5 times. Representatives were then grouped back together into individual subgroups so that all strains and species of the same genus were joined into artificial so called meta-genomes. This was done in order to reduce uninformative redundancies between sequences which in turn accelerated all subsequent computational tasks and increased accuracy of homology computation.

Though clustered together, eukaryote and prokaryote genomes sum-up to a substantial

fraction of sequence data and therefore managing it represents a new type of a problem. Usually in such cases for storage and retrieval, the data is uploaded into a relational database. However, in this particular case (as mentioned above) storing the data in one of the "relational solutions" would lead to an impractically overhead since the available tools for searching the data are general and do not posses a particular search strategy required in this situation. Therefore I created an alternative, simple, flat-file database indexing and formatting manager available as part of `PhyloToolKit-XXX` package, described its utility in the following section.

2.1.3 Database Structure and Content

In order to manage the collected sequence data and provide a simple interface for accessing and querying the information, I created a software tool called `MakePhyloDb`. It is a program which indexes each sequences in the flat file database by assigning a unique set of identifiers each containing:

pgi phylogeny gene identifier - a unique 64-bit integer sufficient for representing numbers up to 19 digits

ti taxonomy identifier - a 32-bit integer unique for each collection of sequences from the the same species

pi phylostratigraphy identifier - 32-bit integer specifying the phylostrat identifier to which a gene has already been assigned to (the number is used as a short-cut in the stratification process - explained in the next section)

All three identifiers are placed in the header of the *fasta* formatted sequence entry (Figure 2.3). The formatter preserves the initial fasta header information placing it after the index, thus allowing the original metadata to be easily accessed if necessary.

The described index structure serves as an adapter that can be further used for any additional custom made key sorting or sequence retrieving procedure and thus presents a flexible system used by a set of programs within the `PhyloToolKit-XXX` package.

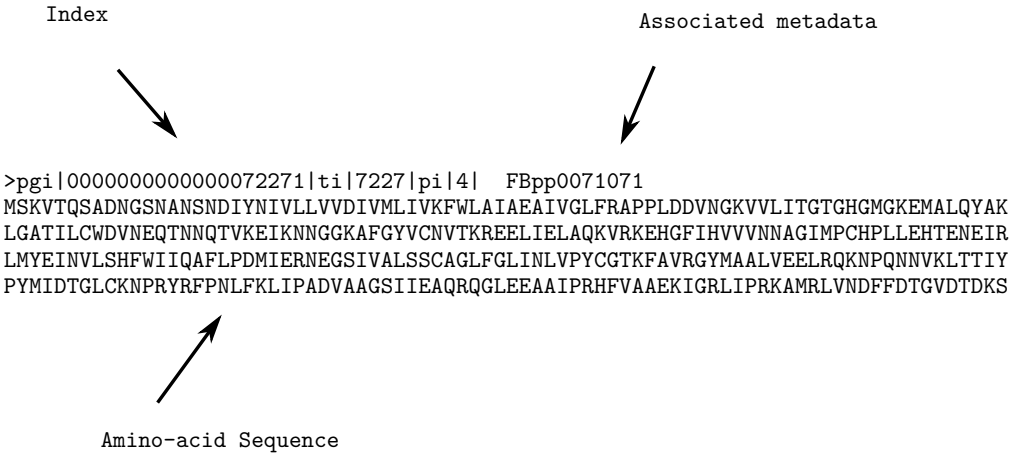


Fig. 2.3 An example of a database entry format and associated identifiers. A simple fasta format is used as the primary key for storing the sequence information.

Aside from sequence information, once formatted, the database contains two additional association files, one containing elementary statistics regarding the data itself and the other various formatting information. Both files are simple ASCII encoded text files and open to additional editing if required.

As far as flexibility with respect to adding and/or removing information to and from the database goes, the procedure is as simple as copying a file into a folder and deleting it. Such manipulations are possible due to the fact that the database is essentially a simple file folder (directory) subjected to any manipulation that can be applied to a typical set of files in a given directory. A detailed description of the database and its management tools can be found in Appendixes A and B.

2.2 Algorithms and Computation Strategies

In this section I present new theoretical results and algorithms for fast ancestral genome content reconstruction. The section is divided into several subsections starting with the rough description of a new algorithm for fast sequence similarity filtering strategy based on a new distance measure I call "the $k-$mer type distance" and a new similarity function I refer to "the JaccScore". The rigorous derivation of these measures and functions can be found in Appendix 1, thus the inquisitive reader is encouraged to refer to the appendix for a more

exhaustive elaboration and these matters.

Following this, I briefly describe new techniques for gene gain calculation (utilized by **QPhyloStrat** algorithm), result of which is further processed by the gene family gain (**PhyloClust** algorithm) and gene family loss computation strategies (**PhLoG** algorithm). All steps are required for ancestral family reconstruction, result of which is investigated and analysed in Chapter 3.

Before jumping directly into subsequent sections, at this point the reader is encouraged to go through a short introduction to mathematical definitions and concepts such as *strings, complexities* and *distances* covered in Appendix A, in order to better understand the underlying strategies. However, going through it is not essential thus materials are only offered as additional summaries for the reader.

2.2.1 HD-index Based Similarity Search Strategy

Similarity searching on many-string collection databases is a central problem in bioinformatics and computational biology: e.g Albá and Castresana (2007); Altschul et al. (1990, 1997); Capra et al. (2012); Edgar (2010), etc., where new, low cost technologies like high-throughput next-generation sequencing, are rapidly increasing the amount of available information (Benson et al., 2007; Flicek et al., 2014). Therefore, a faster, more efficient search strategies are required in order to cope with the growing problem.

The key idea behind the algorithm proposed in this section is based on the observation:

Observation 1 *Let k be a size of a substring, then two strings are more similar to each other if they share more k-sized substrings (k−mers) (Ukkonen, 1992)*

However, unlike in a conventional approach suggested by Ukkonen (1992), where variation in k-mer frequencies is used as a measure of distance, the strategy I propose here is based on computing the number of shared *k-mer types* (κ), with **same or similar in value** k-mer frequencies. Therefore, all features associated with k-mer frequency-based similarity computations are also valid for my k-mer type similarity search strategy. Moreover, I show in section A.3.1.2 that k-mer type distance (d_{kt}), upon which the entire computation is based,

is in fact a lower bound on Ukkonen's k-mer distance and thus less accurate, however, it allows a much faster end-computation.

General outline of the strategy starts by segmenting strings into a set of decreasing overlapping substrings until a k size substring is reached (as described in Section A.3.1.1). This is done for all query and subject strings. Once, k size substrings are computed, based on their occurrence counts, HD-index is created for all subject strings. This index exploits the *perfect hash* function in order to allow for a constant time data retrieval to take place. On the other hand, the k-mer information associated to query strings is utilized for locating a range of indexes within the HD-index data structure (described in Section A.3.2). Given that retrieval is a constant time operation (Section A.3.1.1), search process is proportional to the number of query strings, thus rendering the entire process linear. The number of intersections between computed ranges are then utilized for computing the $k-$mer type distance , followed by its conversion (in order to normalize the distance) into JaccScore values (Section A.3.1.2).

Once the computation has been completed the top scoring database candidate strings (having the highest JaccScore values) are associated to each query string. This is a restricted version of a problem called "the top-X similar string searching problem" (Zezula et al., 2006) defined as:

Problem 1 *Let Q be a set of input strings and S be a set of database strings and let X be a number. Then, for each input sting find ($q \in Q$) find the X number of the closest (most similar) strings from the database ($s \in S$).*

A number of solutions has been proposed in order to efficiently solve this problem (Yang et al., 2010), however, none of which is based on the heuristic approach utilised here (Section A.3.2).

By association each query string with a fixed number of subject strings the search process for each query has been decreases from the entire database to a small collection of strings with the highest chance for containing a significant local similarity region. It is important to note that similarity here is inferred from the identity of k-mer types restricted to a specific segment within a given query-subject pair. Therefore, even at this point it should be clear

that deep homologies are expected to be overlooked by any search strategy utilizing this principle. For a more technical description of the strategy please refer to Appendix A.

2.2.2 Gene Gain Computation

Unlike in a classic phylostratigraphy approach (Domazet-Lošo et al., 2007) where gene gain computation is carried out from leaf to root (in a given species tree), here I present a novel method that starts from root and progresses toward a given lief. That is, the computation is carried out by traversing a given species tree in a top-down manner. Utilizing this approach, first encounter of a homolog within a branching clade ensures that no earlier point of origin is possible to detect and eliminates the gene from all subsequent search queries since its point of origin has been located. Another heuristic that increases the computation speed relies on predetermining the origin points of genes within branching clades. If this information is available then the homology detection only needs to be directed toward a subset of genes emerging before or at the origin of a given branching clade but not latter. The reason is straight forward. If a gene has emerged after the lineage splitting event (origin of the branch) it by definition had no homolog within and lineage branching-off before that event. Note that this logic heavily relies on Dollo's parsimony model and therefore ignores any scenario under which a parallel gain (convergence) could have occurred.

Once the subset has been identified HD-index based similarity search strategy (introduced in the previous section) is applied retrieving X number of the most similar sequences to a given query. As explained earlier, using this strategy only a constant fraction of sequences with the highest potential for containing a HSP needs to be checked. A typical approach here would be to preform the check procedure using a BLAST search, but instead, in order to speed up the process, I developed a new, optimized BLAST search strategy I call **QuickBlast** which is able to preform the computation up to X times faster (X being the number of subject sequences associated to each query) than a regular BLAST, preforming a more sensitive homology search each time it passes through a collection (further information on **QuickBlast** can be found in Section A.3.4).

Figure 2.4 summarizes the general strategy (heuristic) for gene gain computation I call

QPhyloStrat algorithm. For a more technical description of the strategy (**QPhyloStrat** algorithm) and further details please refer to Sections A.3.3 and A.3.4 in Appendix 1.

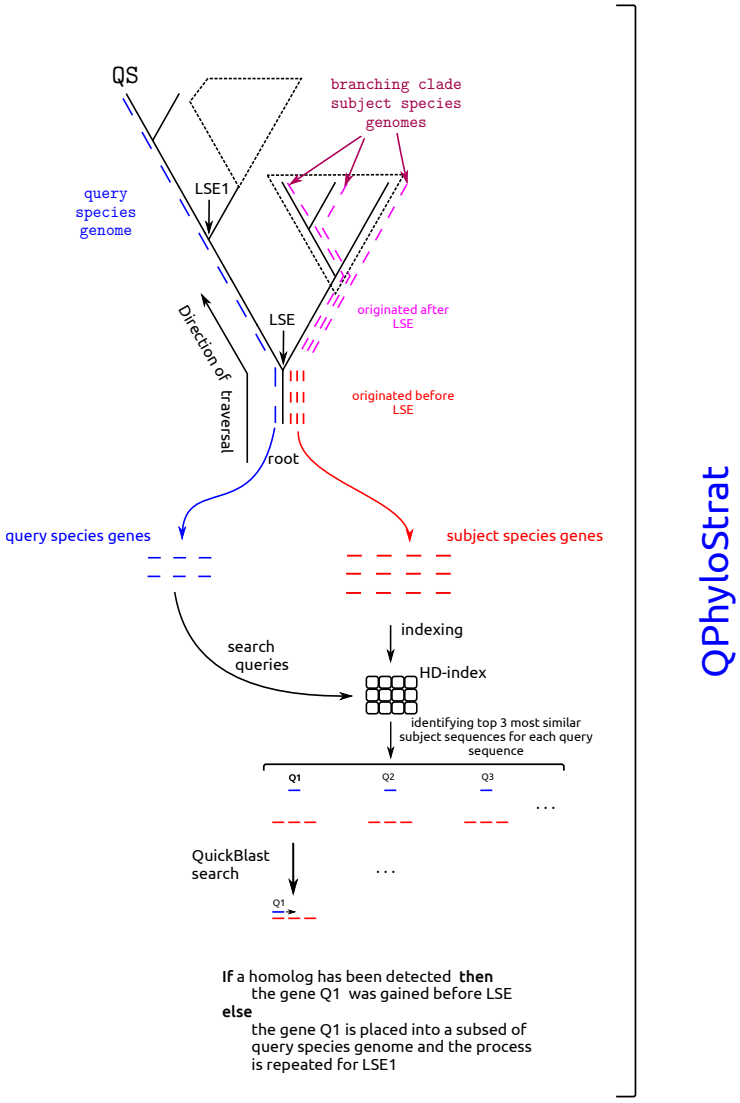


Fig. 2.4 High level description of QPhyloStrat computation process. The computation utilises the basic divide&conquer strategy which divides a genome into a series of subsets each containing a set of genes with higher chances for a homologous pair to be detected and thus the point of origin accordingly assigned to a given lineage splitting event (LSE).

2.2.3 Computing Gene Family Gain Events

A founder gene represents a significant shift in sequence space thus a basis of new gene lineage and/or an entire gene family. Gene family gain event (GFGE) is a point in the evolutionary history marked by the emergence of a founder gene from which the entire family has derived (Domazet-Lošo et al., 2007). Therefore computation of GFGE's is equivalent to computing the number of gene families deriving from the same evolutionary period. In order to obtain the information about the number of gene families it is necessary to identify genes which belong to the same family. A typical approach to gene family computation is carried out by clustering genes according to their relative sequence similarities. There are many different clustering approaches (Hauser et al., 2013; Li et al., 2003; Li and Godzik, 2006; Nepusz et al., 2010) that can be applied. However, since the entire range of computational solutions presented up to this point were treated as a "worst case scenarios", e.g. from selecting the lowest significance threshold (*e-value* $\leq 10^{-3}$) to age computation strategy based on Dollo's parsimony (earliest point of emergence), it is only natural to remain consistent, and propose the clustering strategy founded on same principles. Therefore, since founders represent primary genes from which all current ones have derived from, establishing their quantities at a given time period is essentially the process of calculating the least number (most parsimonious scenario) of hypothetical ancestral genes to which current existing ones can be reduced to. Figure 2.5 illustrates the underlying model.

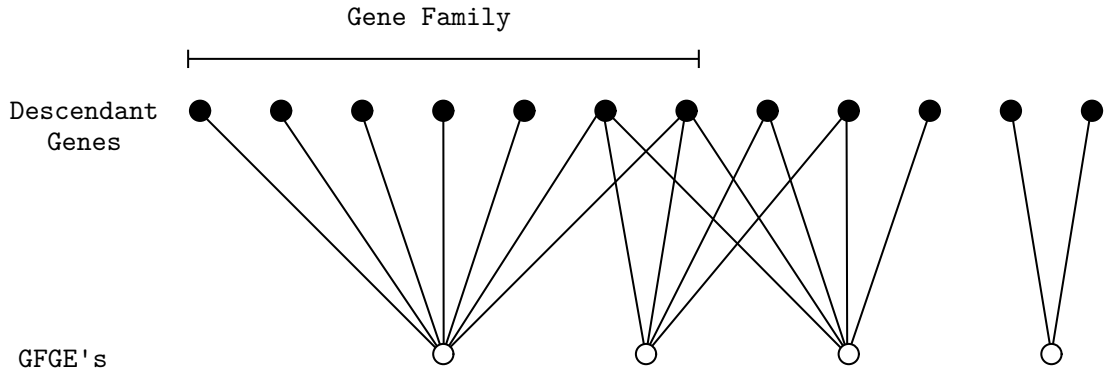


Fig. 2.5 The model of gene family formation. Each gene (filled circle) in current species has derived from one or more GFGEs (empty circle) emerging at some point in the evolutionary history.

According to model illustrated in figure 2.5, no singletons (single lineages) are allowed. This follows from the fact that each gene traced to a specific point of origin, is placed there based on the fact that a homologous pair exists, thus the same holds for its homologous pair, and therefore a minimum number of family members is two. This is also the underlying reason for computing the GFGE and not founders. Moreover, since here GFGEs are in the focus of the computation, singletons (which are usually produced using alternative clustering schemes (Hauser et al., 2013; Li and Godzik, 2006)) by definition have to be excluded unless they are associated to the youngest phylostrata in which case they could represent founders of new families.

The clustering algorithm I call **PhyloClust** based on the model presented in Figure 2.5 for computing GFGEs is described in Section A.3.5 of the Appendix A.

2.2.4 Computing Gene Family Loss Events

Gene family loss event (GFLE) is a point in species evolutionary history characterized by the process of elimination in which the last member of a gene family, emerged at some earlier point in time, is removed from genome of either current living species or an ancestral one (descendants of which do not contain a single member of the extinct family). Computation strategy for detecting such events described in this section is essentially a search process in which current species genomes are queried for descendent members of a given family. If a member could not be located species are then grouped together and their common ancestral point (if such exists) calculated. This point is then labelled as an exit point (GFLE) of that gene family for each descendent species lineage.

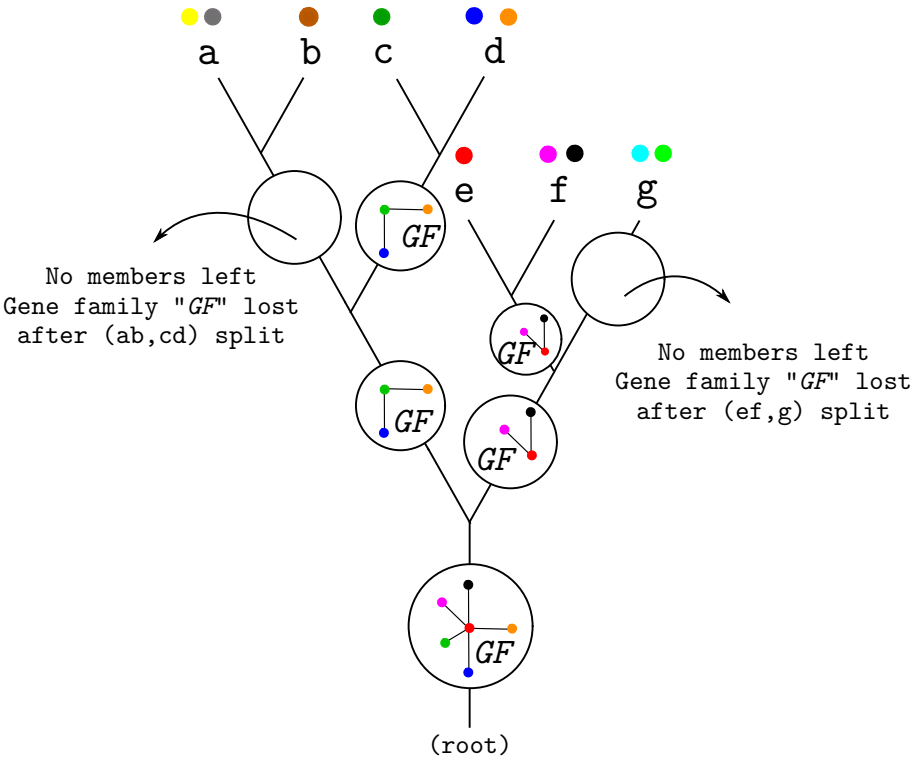


Fig. 2.6 The model of gene family extinction. Each gene (filled circle) in current species has derived from one or more founder genes (empty circle) emerging at some point in the evolutionary history.

As en example consider the situation illustrated in Figure 2.6. Let *GF* be a family and its gain event (GFGE) traced to the basis of the underlying hypothetical species tree (as illustrated). In order to compute the GFLE of that family it is necessary to locate all vertices within the tree (ancestral nodes) after which none of genomes of current living species contain a gene member of that family (*GF*). In Figure 2.6 there are two such positions. Given that $\{a,b..g\}$ is a set of species and coloured circles represent genes in their corresponding genomes, gene family *GF* contains no genes from either species "a" or "b" therefore after the evolutionary split of lineages "ab" and "cd" family *GF* was preserved in lineage "cd" and not "ab". That is, after the split of two lineages the gene family *GF* was eliminated from, and was not present in a common ancestor of species "a" and "b". The same can be said for split between lineage "g" and "ef". Species "g" contains two genes (light blue and light green) but since they are not members of family *GF* and since "g" does not contain any other gene which can be considered as a *GF* member, then the family *GF* was eliminated from the

genome of species "g". Therefore, GFLE of family *GF* in species lineage "g" has occurred after the split of ancestral lineage "ef" and lineage "g".

It is important to note that unlike GFGE which can only occur once in the course of evolution, the GFLE can happen a number of times across different species lineages.

The entire computation process is utilized by the **PhLoG** algorithm described in Section A.3.6.

2.3 Quantifying Genome Complexity

In the introductory chapter of this dissertation an entire section has been devoted discussing biological complexity. By the year 2001 more than 31 complexity measures (Lloyd, 2001) have been proposed in scientific literature and it is likely that many more have been added in the last 15 years. This number can be seen as a direct consequence of the lack of an unambiguous definition of complexity. Therefore, it is essential that one is precise in what is meant by the term complexity, if the results which rely on it are to be properly understood.

Here I argue in favour of physical complexity (Adami, 2002; Adami and Cerf, 2000) when utilized for quantifying biological complexity and use it as a basis for deriving a measure for genome complexity similar to that proposed by Wolf and Koonin (2013).

Initial principle upon which this measure is based is well established in Shannon's information theory stating that the complexity ($\mathfrak{C}$) is proportional to the amount of information a system has. Given that the information is calculated as a reduction in entropy, by assumption that all elements of a system are volatile, the total entropy can than be expressed as the number of such volatile elements. On the other hand, fixed or semi-volatile elements are expected to decrease the entropy of a system, hence:

$$I = H_{\max}(\text{system}) - H(\text{system}|\text{environment})$$

And therefore the complexity is proportional to the amount of information a system contains:

$$\mathfrak{C} \approx I$$

In Section A.4 I make a formal connection between the amount of information a genome contains and the number of gene families, thus deriving a simple measure of genome complexity according to which the amount of information a genome has is proportional to the number of gene families it contains and therefore genome complexity equals to:

$$\mathfrak{C} \approx |GF| \tag{2.1}$$

In case of the ancestral genomes, in the above expression $|GF|$ refers to the number of gene families conserved at a given evolutionary distance.

2.3.1 Gene Family Turnover Rate

In order to calculate gene family turnover rate for each phylostrata labelled as $p \in P$, (where P is a set of nodes in a given species tree), the number of GFGEs associated to a particular p is divided by the number of GFLEs belonging to the same group. Therefore, the gene family turnover rate, for a given phylostrata is defined as:

$$\mathfrak{I}_p = \log\left(\frac{|GFGE_p|}{|GFLE_p|}\right) \tag{2.2}$$

The logarithmic function in the above equation is used to tame the numbers and reflect the direction preserving the relative magnitude of the change (if the number of GFGEs is greater than GFLEs, $\mathfrak{I}_p > 0$ and if the number of GFGEs is smaller than GFLEs, $\mathfrak{I}_p < 0$)

2.3.2 Genome Complexities Growth Rate

Growth rate is the rate at which the number of elements in a given population increases in a given time period, expressed as a fraction of the initial population. In case of genome complexity the growth rate here refers to the change in complexity within a given time period. Thus a genomic complexity growth rate (GCGR) between two adjacent species (ancestral and/or existing) labelled according to species tree nodes p_i^a and p_{i-1}^a for $p^a \in P^a$, $a \in A$ and $i \in [1..|P^a|]$ I defined as:

$$GCGR_{p_i^a} = \left(\frac{\mathfrak{C}_{p_i^a} - \mathfrak{C}_{p_{i-1}^a}}{\mathfrak{C}_{p_{i-1}^a}} \right) \tag{2.3}$$

Note that the time parameter has been eliminated from the Eq. 2.3 since usually it is assumed that time frames in which measurements take place are evenly spaced. However in cases when this is not true I use an alternative expression:

$$GCGR_{p_i^a} = \left(\frac{\mathfrak{C}_{p_i^a} - \mathfrak{C}_{p_{i-1}^a}}{\mathfrak{C}_{p_{i-1}^a} \times \Delta T} \right) \tag{2.4}$$

A positive growth rate indicates that the genomic complexity is increasing, while a negative growth rate indicates that the genomic complexity is decreasing. Moreover, in section 3.2.2 I use an equivalent approach for computing the gene family loss rate. There I compute the loss rate by dividing the number of lost gene families in a given phylostrata by the number of families present in the previous phylostrata and I do this for each set of families gained at a given ancestral point.

2.4 Computing the Expected Number of Gene Family Gain Events

The expected number of gene family gain events (*E(|GFGE|)*) has been computed by counting the number of significant similarity matches between BLAST hits after all-against-all blast computation (e-value $\leq 10^{-3}$) has been carried out on a set of genes traced to a given origin point ($ps \in PS$). Equation 2.5 summarizes the computation of the expected number of GFGEs:

$$E(|GFGE|) = \sum_{i=1}^{|ps|} \frac{1}{\Delta(ps_i)} \tag{2.5}$$

where $ps \in PS$ is a set of genes in phylostrata ps, ps_i is a gene from that set for $i \in [1..|ps|]$ and $\Delta(ps_i)$ the number of BLAST hits each ps_i makes. The lowest value *E(|GFGE|)* can have is one, denoting that all genes in a given phylostrata are related, whereas its maximum value ($\max(E(|GFGE|)) = |ps|$) refers to the situation in which all genes in a given phylostrata are in fact GFGEs.

2.5 Statistical Analysis

Statistical analyses applied to various results throughout this thesis can be divided into:

- **Correlation and regression analysis:** Pearson's correlation (Pearson, 1895)
- **Testing statistical significance:** Chi-square test (Pearson, 1900), Fisher's exact test (Fisher, 1922), Hypergeometric test (Grant and Ewens, 2001)
- **Distribution fitting:** QQ-analysis (Wilk, 1968)
- **Multiple testing corrections:** Bonferroni correction (Grant and Ewens, 2001), False discovery rate (Benjamini and Hochberg, 1995)

It is important to note that in all calculations connected to correlation and regression analyses, coefficients and corresponding probabilities were computed by removing outliers

(5^{th} and 95^{th} percentile - black points). Therefore, the obtained values only reflect statistical features associated to "core" data (red points).

Hypergometric probability density function describes the probability of obtaining the x objects of type A within a given sample of size X, such that in the entire population of size Y there are only $y \geq x$ number of objects of that type (A). To set it apart form a classic binomial probability density function, the probability described by hypergometric distribution assumes sampling without replacement. For example consider a finite population of Y objects some of which (y) are of type A and $Y - y$ are of type B. If one takes randomly X objects from the population, the hypergometric probability density function gives a probability that x of those X objects are of type A. The function is summarized in equation 2.6:

$$P(x;Y,y,X) = \frac{\binom{y}{x}\binom{Y-y}{X-x}}{\binom{Y}{X}} \tag{2.6}$$

for $max(0, X+y-Y) \leq x \leq min(X,y)$. The function is particular important in phylostratigraphy analyses where it is used to determine the significance of the obtained pattern produced by mapping characters onto the stratified genomes (Domazet-Lošo et al., 2007). Here the function is used to determine the significance of the obtained stratified gene distribution with respect its background noise caused by non-uniformity (phylogeny-wise) within genome sequencing projects undertaken in the last two decades (some lineages contain a large number of sequenced genomes whereas others are virtually neglected). In a test for over-representation of successes in the sample, the hypergeometric p-value is calculated as the probability of randomly drawing x or more type A objects from the population Y in X total draws. In a test for under-representation, the p-value is the probability of randomly drawing x or less type A objects from the population Y in X total draws.

QQ-analysis is a graphical method for comparing two probability distributions by comparing their quantiles. Quantile is a sub-sample of data-points such that each quantile contains an equal number of elements. As an example in case of the probability distributions, quantile can refer to: terciles (each containing a third of data points), quartiles (each contain-

ing a quarter of data points), quintiles (each containing a fifth of data points), ..., percentiles (each containing a hundredth of data points), etc. QQ-analysis includes plotting quantiles from two distributions against each other. If the two are similar in the resulting plot, the points will approximately lie on $f(x) = x$ line. If they are linearly related they will still exhibit a linear pattern but will not be on $f(x) = x$ line. In both cases such result clearly indicated a good fit between two distributions compared.

2.6 Bootstrap

Bootstrapping is an alternative strategy to traditional statistical data analysis based on *a priori* assumption of an underlying probability distribution. That is, when testing a hypothesis in a complex experimental environment for which little or no information regarding the expected output is available, it is reasonable to assume that the obtained values ought to be either uniformly ("Laplace principle of insufficient reasoning" (Dupont, 1977)) or normally distributed. However, this is only an assumption with no pre-requisition restricting the possible alternatives. Bootstrapping bypasses this problem by letting the data distribution to "define" itself.

The procedure works as long as the experiment produces more than a few independent and identically distributed outcomes (which can be verified by a simple QQ-analysis). In calculations preformed in Chapter 3, the bootstrapping procedure begins by sampling (with replacements) a smaller fraction of outcomes obtained in the experiment, followed by a recalculation of distribution parameters. This process is repeated many times over (usually 100 times) in order to produce a result from which further estimates can be derived. For further material on the subject please refer to Efron and Tibshirani (1993) and Sprent (1998).

In case of genome complexity computation, bootstrapping has been preformed by randomly sampling (with replacement) lineages used in the complexity computation and recalculating the outcome (genome complexity at a given ancestral point).

2.7 Quality assessment

The measure of quality can be defined as a degree of relatedness of an observed outcome to its expected (reference) value. It typically includes two types of measurements: accuracy and precision. In this section I describe these measures and introduce three additional ones all used (or mentioned) in evaluating results obtained and presented throughout this thesis.

Let us assume that there exists a sample space that contains a set of all possible outcomes of a random experiment and let an event be a subset of that space. Then, given an experiment with two events of possible outcomes (observed and expected) the sample space (with respect to outcome classes) can be divided into four different categories as illustrated in Figure 2.7:

TP (True positives) - The observed outcomes that were expected as a result of the experiment.

TN (True negatives) - Outcomes not expected and not observed in the experimental results.

FP (False positives) - Outcomes occurring in the experiment but not expected.

FN (False negatives) - Outcomes not occurring in the experiment but expected to occur.

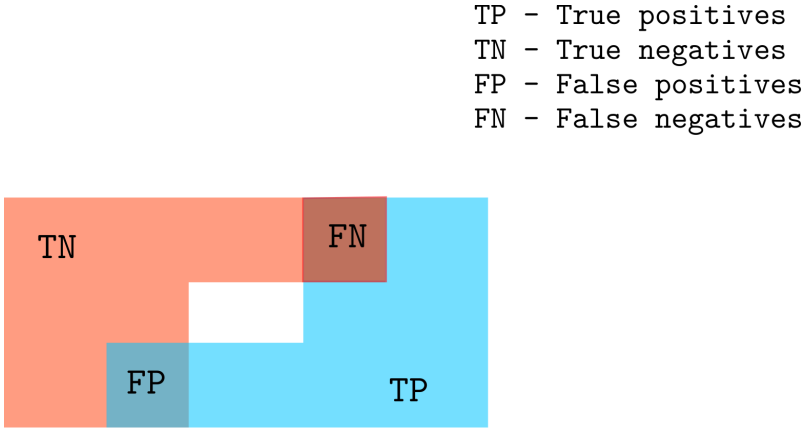


Fig. 2.7 Sample space illustration for an experiment with two obtainable events. The sample space is made up of all possible outcomes of an experiment and an event is a subset of that space. For an experiment with two overlapping events there are four classes of outcomes: TP -true positives (outcomes in both expected and observed classes), TN - true negatives (outcomes not in expected nor observed class but within a sample space) , FP - false positives (outcomes within the observed classes but not expected) and FN - false negatives (outcomes in expected class but not in observed)

Using this classification it is further possible to define:

Accuracy Accuracy is a degree of closeness of an observed outcome to its (expected) reference value. In an experiment with two types of outcomes possible (positive and negative) the accuracy is defined as a fraction of correctly obtained outcomes, where correctness is defined as:

$$\text{Accuracy} = \frac{|\text{TP}| + |\text{TN}|}{|\text{TP}| + |\text{FP}| + |\text{FN}| + |\text{TN}|} \tag{2.7}$$

Jaccard index Jaccard index is a modification of accuracy measure in which negative outcomes, due to the experimental set-up, are not obtainable. Index is defined as:

$$\text{Jaccard index} = \frac{|\text{TP}|}{|\text{TP}| + |\text{FP}| + |\text{FN}|} \tag{2.8}$$

Precision Unlike accuracy precision reflects the repeatability (reproducibility) of the measurement. Regardless of its distance form the reference, if the obtained results exhibit low variance they are labelled as highly accurate. The formal definition is summarized in equation 2.9:

$$\text{Precision} = \frac{|\text{TP}|}{|\text{TP}| + |\text{FP}|} \tag{2.9}$$

Sensitivity and Specificity Sensitivity reflects the proportion of positive outcomes that are correctly identified. As such it measures the probability of obtaining an expected outcome:

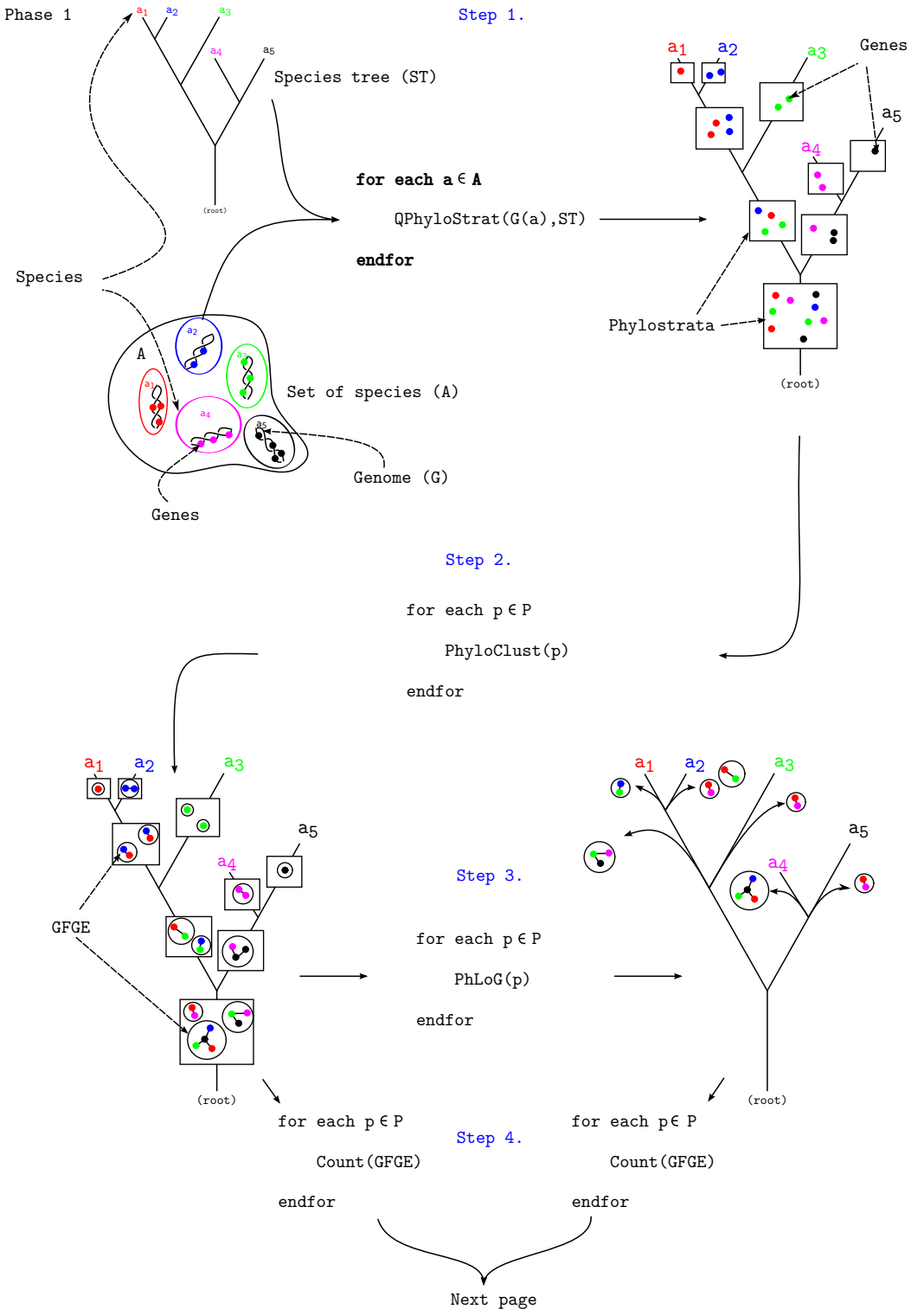
$$\text{Sensitivity} = \frac{|\text{TP}|}{|\text{TP}| + |\text{FN}|} \tag{2.10}$$

whereas specificity is its opposite:

$$\text{Specificity} = \frac{|\text{TN}|}{|\text{FP}| + |\text{TN}|} \tag{2.11}$$

measuring the fraction of negative, correctly identified outcomes.

2.8 Computational Workflow



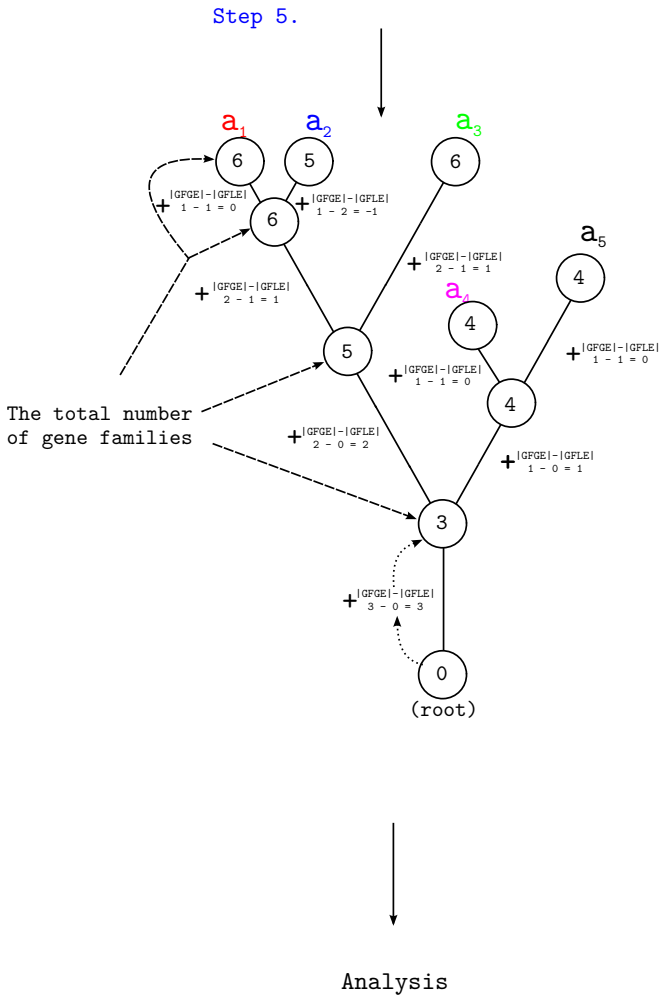


Fig. 2.8 Computational workflow: Ancestral gene family content reconstruction.

In this section the entire workflow based on the above described tools implemented in `PhyloToolKit` (available from https://github.com/RobertBakaric) underlying GFGE and GFLE computation, together with additional statistical analysis is presented. The entire process is divided in two phases; genome reconstruction phase followed by data analysis in which speed and quality of the obtained results is evaluated.

Reconstruction process begins by dividing genes into their respective age clusters (phylostrata) by computing gene gain events for each of 383 mentioned genomes. The overall computation procedure is illustrated in Figure 2.8 and consists of 5 steps.

Step 1: Calculate the age of genes (**QPhyloStrat** algorithm). Computation includes information regarding phylogenetic relations between species and their corresponding genomes.

Step 2: Genes traced to the same evolutionary period (phylostrata) are clustered into families using **PhyloClust** algorithm.

Before moving to Step 3, an additional correction analysis termed "BLAST backtracking correction" is applied. The procedure identifies cluster representatives within each phylostrata and compares them (using BLAST) to each other. Since each sequence in the analysis is a cluster representative to which each sequence of that cluster has a statistically significant similarity score (*e-value* of 10^{-3} or lower), *e-value* threshold used in "BLAST backtracking correction" process is set to 10^{-15} (this threshold value, according to triangular inequality principle, asserts those cluster members with pairwise matches based on the same HSP region to be "clusterable"). Once all matches have been located, the entire families (singletons included) are backtracked to phylostrata occupied by the oldest member and merged to its cluster. As a result a lot of singletons (produced by asymmetric stratification as a result a non-commutative property of the *e-value*) are eliminated and all ambiguities tide to family gain events, resolved.

Step 3: Calculated GFGEs are passed to **PhLoG** tool for GFLE computation.

Step 4: Counting the number of GFGEs and GFLEs within each evolutionary period.

Step 5: Ancestral gene family content reconstruction. Computation is carried by summing all GFGEs and GFLEs on a path from root to a given ancestral, or current species node.

Chapter 3

Results and Discussion

The chapter is divided in two main sections: **Performance Evaluation** and **Patterns and Trends Associated with Gene Family Gain/Loss Events in Species Evolution**. The first section investigates algorithmic solutions and heuristic strategies presented in Chapter 2. It includes computational runtime (speed) measurements followed by a quality assessment analysis. In the second part, the proposed strategies are applied to real data sets, containing over 9,000 genomes, in order to investigate patterns associated to gene family gain and loss events across 383 eukaryote lineages.

3.1 Performance Evaluation

Performance evaluation constitutes of two types of measurements, a runtime analysis and a quality assessment. Runtime analysis is a simple measurement in which the computational speed (time in seconds) of the underlying algorithm is measured as a function of its (increasing in size) input parameters. Such measurements are usually conducted in cases when a proposed computational strategy is too complex for a classic asymptotic analysis to take place (or it requires an experimental conformation). Quality assessment on the other hand, provides a simple framework for estimating how good the obtained result is. In cases where an exact solution is not computable (usually extremely long runtimes being the un-

derlying cause), alternative strategies like heuristic methods and approximative calculations are applied. These solutions often trade their accuracy and/or precision for speed. Therefore, quality assessment of the obtained result in such cases is an essential factor, required for evaluating whether a result is sufficiently reliable to be used in any downstream analysis and/or interpretation. As mentioned, quality control includes two types of measurements: accuracy and precision. Here the focus is primarily on precision, since the entire range of calculations done throughout the thesis revolves around fitting within some pre-specified lower bound framework, that is, results inside a given boundary, regardless of their location within it. Moreover, precision can only be computed if a referent value exists (experimental result labelled as "expected"), therefore, in cases where this type of value is not available, the analysis has not been preformed and results note reported.

3.1.1 HD-index Based Similarity Search Algorithm

The speed of HD-index based similarity search was compared to the BLAST program on a set of simulated sequences each 1000 AA residues long and having one HSP segment with at least 10 neighbouring sequence. A HSP segment is a substring of a given string characterized by high similarity to another string or its substring (Zhang, 2003). In this analysis each segment was characterized by having at least 25% identity and 10% coverage on its matching pair. An array of such clusters spanning between 10 and 10^5 sequences per set, with the above stated properties, was created and used in the analysis, such that the same set was used both as an input ($|Q|$) and a database ($|S|$).

The measurements were preformed on a single Intel(R) 2.7 GHz processor with 4 MB of cache memory. As a result, approximately quadratic increase in runtime (with respect to the input size) associated to BLAST computation can be observed in Figure 3.1 (each time the input is increased by a factor of 10 the runtime increases by a factor of $\approx 10^2$).

On the other hand near to linear runtime performance is associated to HD-index based similarity search strategy (increasing the input by a factor of 10 increases the runtime by ≈ 10 times). The result clearly indicates the speed dominance of HD-index based search algorithm over BLAST.

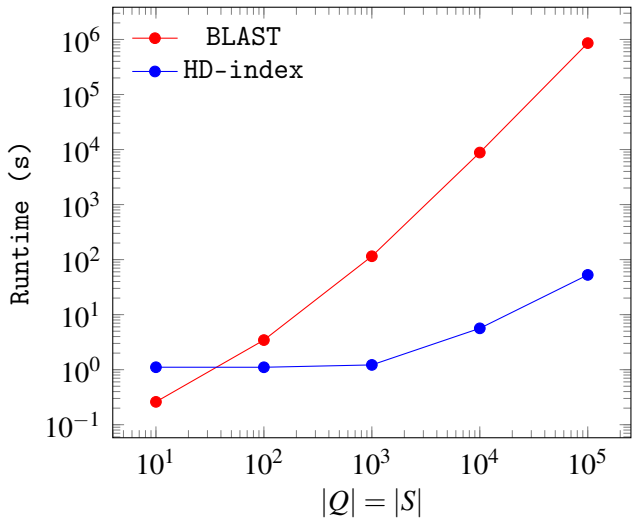


Fig. 3.1 Runtime comparison between HD-index similarity search algorithm and BLAST. Runtime performance was measured with respect to the total number of sequences within a given query($|Q|$) and subject($|S|$) set. Each query and subject set contained between 10 and 10^5 simulated sequences, each 1000 AA residues long.

To estimate the quality of HD-index based search strategy, the fraction of correctly identified sequences, within a given set containing one HSP segment per sequence, was computed. This time the set was generated by simulating 10,000 random sequences each 1,000 AA in length. The center of each sequence, 300 AAs long, was then copied nine times and mutated until only 5% of sequence identity with respect to the original segment was preserved (thus each pair representing a HSP with 5% preserved sequence identity). Random AAs were then added from both sides until the total length reached 1000. The process was repeated generating sets with 15, 20, 25, 35, 45, 55, 75 , 95 and 100 percent of preserved sequence identity between HSPs. Moreover, the density of identical AAs within each HSP segment, of each percent identity cluster, was negatively skewed, log-normally distributed so that on its right end (toward C-terminus), each HSP segment had a streak of at least $\approx 5-10$ identical, high scoring, consecutive AAs thus ensuring a seeding locations (3 identical AA) for both methods. The search process was then executed using both BLAST and HD-index based similarity search strategy, with the original 10,000 AA sequences as a query set and the rest divided into databases according to the above mentioned percent of preserved identity. Obtained results are summarized in Figure 3.2.

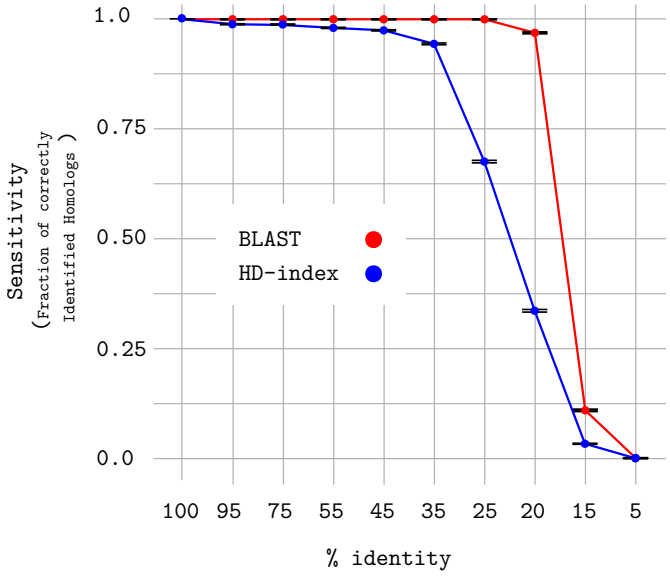


Fig. 3.2 Assessing the quality of HD-index based similarity search queries. Search quality (blue line/circles) was estimated by counting the number correctly identified homologous sequences at different percent identity thresholds (associated to HSP segments within them). Red line (circles) summarize the results of a comparative analysis repeated by using the BLAST program. In both cases the quality assessment included 10000 sequences, deviations associated to each measurement are depicted with horizontal lines below and above given value, reflecting its 95% confidence interval.

Quality-wise, while comparable when percent identity is high, BLAST clearly outperforms the HD-index based similarity search strategy when that percentage drops below 35%.

Next, since BLAST tool uses the *e-value* to separate significant matches from those identified by chance alone, the distribution of the expectation values was plotted (box-plots) as a function of percent identity category (Figure 3.3). Surprisingly, given a "percent category", e-values within it, ranges form $\pm 5-7$ orders of magnitude and therefore according to this result the two converge (or at least have a maximum comparability) when BLAST cutoff e-value is somewhere between $\approx 10^{-20}$ and $\approx 10^{-40}$.

According to the these results, any similarity detected by HD-index based search algorithm can certainly be characterized as evidence of homology, but, in comparison to BLAST, a larger fraction of true homologs is expected to be overlooked and not detected. In return this can affect the gain analysis (when applying phylostratigraphy approach to gene gain

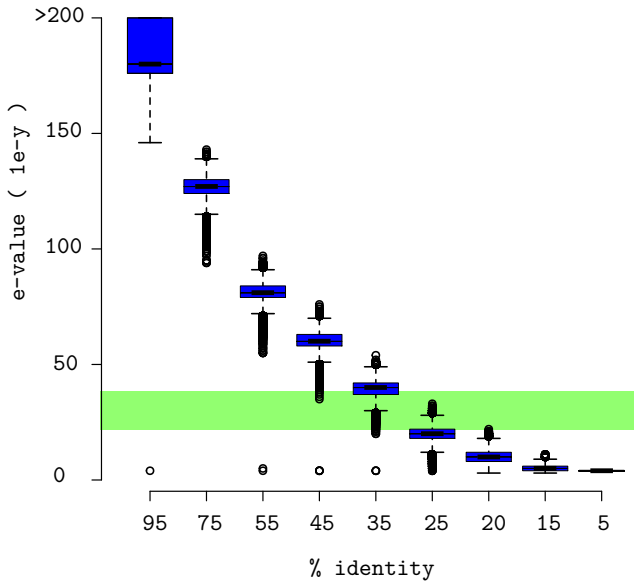


Fig. 3.3 The distribution of reported BLAST e-values associated to a given percent identity threshold category used in quality assessment analysis, results of which can be found in figure 3.2. The green stripe separates the comparing strategies according to reported quality estimates with respect to the calculated *e-value* threshold.

computation) by miscalculating the point of origin in cases when the *e-value* threshold is set to $10^{(-40 \text{ to } -20)}$ or higher (usually the default is 10^{-3}). The direction of such miscalculation will most certainly always be toward a more recent points. However, the opposite is not excluded since, as shown in Figure 3.2, the errors associated to BLAST homology detection are real and may miss an existing homolog when sequence identity is below (or at) 20% (category in which statistically significant matches still exist - Figure 3.3).

3.1.2 Gene Gain Computation - QPhyloStrat Algorithm

To test the runtime performance of the gene gain computation strategy utilized by *QPhyloStrat* algorithm, a set of 10 subsets of randomly selected protein sequences from the *D. melanogaster* genome (13,933 protein coding sequences - $|Q|$) and a separate one from *db_200514* database (from which only 10 genomes were randomly extracted - $|S|$), were selected such that each subset included $\lfloor |Q|/10 \rfloor$ ($\lfloor |S|/10 \rfloor$) more sequences than the previous one. All tests were preformed on a single Intel(R) 2.7 GHz processor with 4 MB of

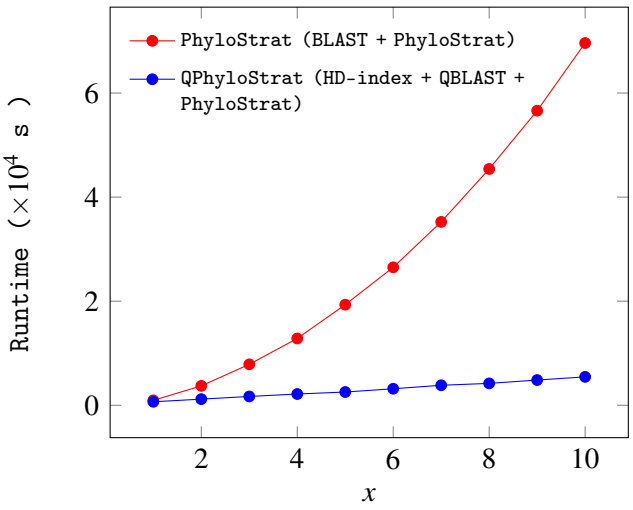


Fig. 3.4 Runtime comparison between *PhyloStrat* and *QPhyloStrat* gene age computation strategies. Runtime was measured with respect to the number of query and subject sequences such that the value on the x-axis represent the input sizes defined as: $x \times (\lfloor |Q|/10 \rfloor)$ and $x \times (\lfloor |S|/10 \rfloor)$, where $|Q|$ is the number of genes in the query file and $|S|$ the number of genes in the database.

cache memory and Ubuntu 12.04 LTS as the underlying operating system.

From the obtained runtime measurements presented in Figure 3.4, a clear linear increase in runtime performance of *QPhyloStrat* with respect to its input size is evident. On the other hand, runtime behaviour associated *PhyloStrat*[1] exhibits, what can be described as a polynomial increase in runtime performance as a function of the input size. From measurements reported in the previous analysis (Figure 3.1), the underlying cause for such runtime behaviour is most certainly the BLAST algorithm. Although, a set of heuristic solutions have been applied to increase its performance, essentially BLAST is still an alignment based tool with the multiplicative runtime behaviour that converges to quadratic, as query and subject data inputs become equal in size (which is an exact description of the obtained result presented in Figure 3.4). The *QPhyloStrat* gene gain computation strategy has a clear runtime dominance over a BLAST based pipeline which allows an investigator to conduct a required

[1] *PhyloStrat* is a linear time ($O(|\text{Blast Output}|)$) solution (a part of the `PhyloToolkit-XXX` package) utilizing a more efficient stratification algorithm for computing gene gain events than the one proposed by (Domazet-Lošo et al., 2007). Moreover, since the algorithm directly relies on BLAST, time measurements will by default include BLAST computation in order to be comparable with *QPhyloStrat*

calculation using a large number of species (genomes), that until now presented a serious technical obstacle preventing any such type of analysis to take place.

Unlike in the previous quality assessment, results obtained from *PhyloStrat* were taken as reference points in estimating the quality of those produced by the *QPhyloStrat* algorithm. Using the entire *db_200514* as a database and the *D. melanogaster* genome as an input, each gene in the *D. melanogaster* genome was traced back to its point of origin (Figure 3.5). The resulting distribution (number of genes traced to different evolutionary periods - phylostrata) was then evaluated in two ways. First, a simple gene assortment across different phylostrata was evaluated by computing Jaccard index as defined in Eq. 2.8 (TP - being genes traced to the same point of origin by both methods, FP - being genes traced to one point of origin by *QPhyloStrat* and to some more recent point by *PhyloStrat*, FN - being genes traced to one point of origin by *PhyloStrat* and to some more recent point by *QPhyloStrat*). Though the obtained index values ranged from 0.01 to 0.66, the average value rounded up to 0.13, indicating very low assortment quality (with respect to BLAST based computation) of *QPhyloStrat* results.

In case of (character) mapping analysis (assigning features to stratified genes), this result clearly demonstrates an overall incomparability between results obtained by *PhyloStrat* and *QPhyloStrat* calculation.

On the other hand, the obtained distributions (regardless of the underlying assortment) upon visual inspection are very similar. Therefore, the second type of quality assessment was preformed to establish whether the two (from a quantitative point of view) are comparable or not.

First, to eliminate the risk that the obtained results are sampling artefacts caused by the underlying phylogenetic clustering of genes within a database, *D. melanogaster* gene frequencies associated to each phylostrata were compared to gene frequencies associated to each branching clade by applying a simple hypergeometric test (Grant and Ewens, 2001). Furthermore, to account for multiple testing analysis (establishing the probability of the null hypothesis (*D. melanogaster* gene distribution is the same as database gene distribu-

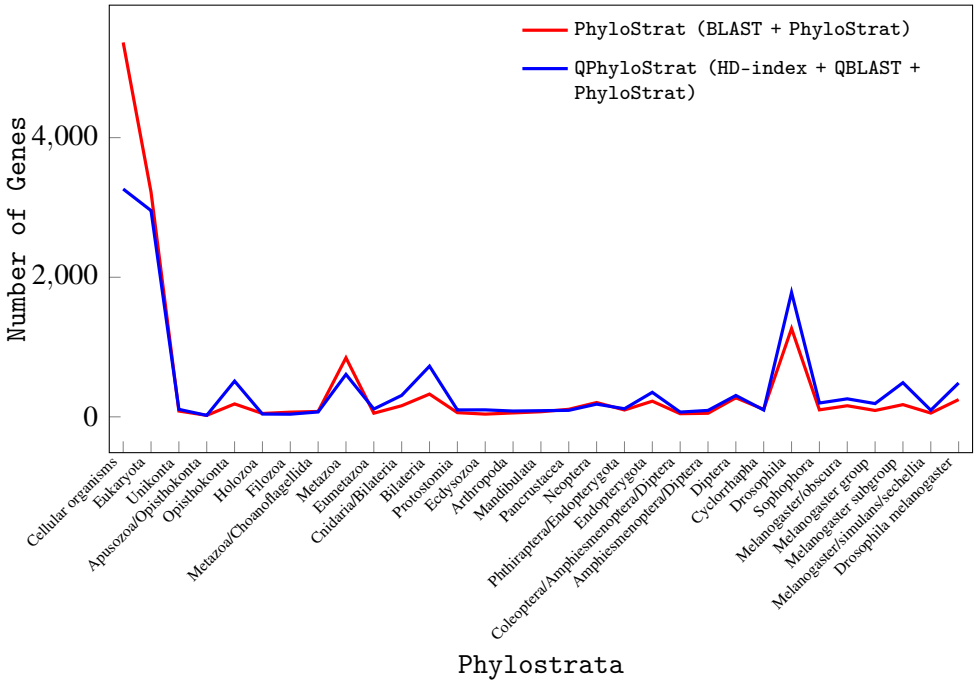


Fig. 3.5 Quality assessment of *QPhyloStrat* gene gain computation strategy. Red (*PhyloStrat*) and blue (*QPhyloStrat*) lines represent the distribution of 13,933 *D. melanogaster* genes traced to their respective points of origin.

tion defined by the underlying phylogenetic tree topology) in each phylostrata), trailing corrections (FDR correction (Benjamini, 2010; Benjamini and Hochberg, 1995) and Bonferonni correction (Dunn, 1961)), were applied. The obtained result, summarized in Tables A.2 and A.3 (Appendix A), clearly demonstrate how both cases cannot be labelled as "random/coincidental" and therefore rendering the result not to be an artefact of the underlying gene distribution.

Next, the trace-back computation process (based on which the origin points were calculated) was analysed by looking at percent identity values of stratified genes. Figure 3.6A and B summarize the obtained results.

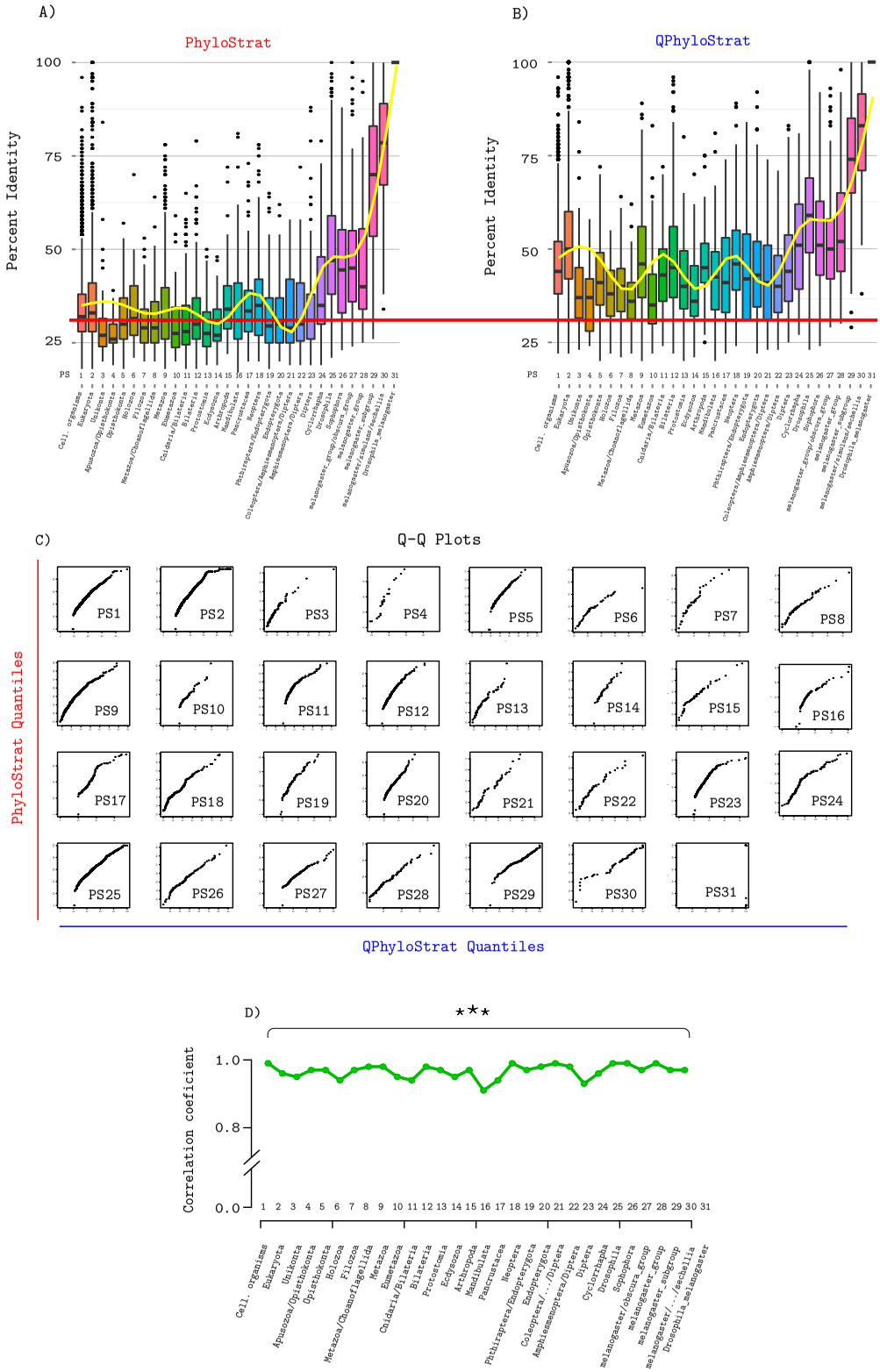


Fig. 3.6 Quality assessment of gene gain computation. Figures A and B summarize the distribution of percent identity match values of aligned sequences traced back to their respective points of origin (phylostrata) using *PhyloStrat* (A) and *QPhyloStrat* (B) computational strategy. The yellow line represents smoothed distribution means. Comparing the figures, a clear systematic shift toward higher percent identity values between matched homologous genes, generated by the *QPhyloStrat* algorithm is evident. C) QQ-analysis (goodness of the fit). Pairwise distribution quantile comparison shows individual distributions in A and B are most likely of the same type. Further supporting this hypothesis are significant high correlation coefficients associated to each comparison, plotted in figure D.

Clearly by applying *QPhyloStrat* computation strategy, genes traced to their respective points of origin were located there based on their mutual percent identity values generally higher than those computed by BLAST (higher by $\approx 10\%$). Moreover, to show that this is a systematic shift, each distribution (in each phylostrata) was further subjected to the additional QQ-analysis, (Figure 3.6C) revealing no significant discrepancy between compared strategies (further supported by highly significant correlation coefficients associated to each individual result (Figure 3.6D)).

Lastly, total gene counts per phylostrata cluster were compared by measuring the linear relation between the opposing strategies. Once again (Figure 3.7), highly significant relationship supported by a high correlation coefficient (0.95) confirms the initially conjectured similarity postulated upon visual inspection of Figure 3.5.

Though marginal (based upon the above results), it is still evident that certain discrepancies exist. As noted in the previous section these discrepancies ought to be associated to

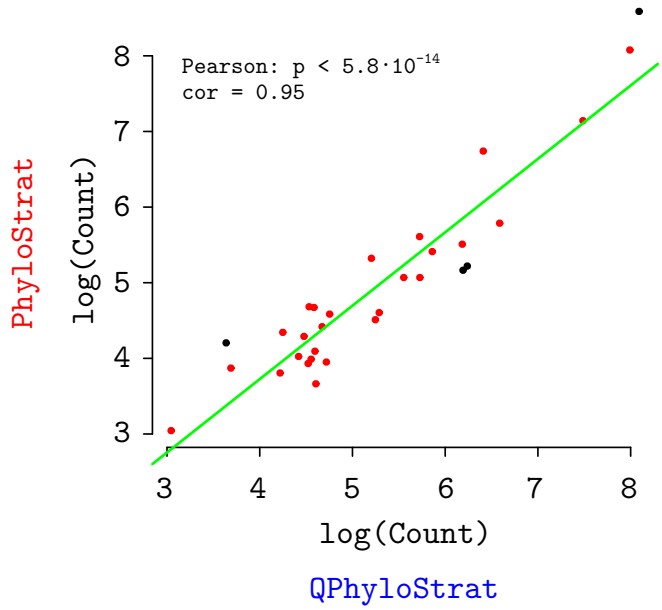


Fig. 3.7 Comparing gene gain distributions of *D. melanogaster* genes obtained by *PhyloStrat* and *QPhyloStrat*. Correlation between the two (green line) was computed by removing data points from upper and lower quantiles (the 5th and the 95th percentile - black points). High and significant Pearson's correlation coefficient indicates a good distribution overlap between the compared computation strategies.

both over- and under-estimation (with respect to BLAST based calculations) caused by non-positional identity based stratification utilized through HD-index search strategy within the *QPhyloStrat* algorithm and homologs not detected by BLAST search at low percent identity thresholds. Therefore, an additional distribution analysis was conducted. Much like in the previous case, where total counts were directly compared, in this analysis origin points of individual genes were overlapped, such that, if a gene was traced to the same point of origin by both comparing methods the number of true positives (blue squares on a matrix diagonal in Figure 3.8) was incremented by one. In the same way the number of false positive and negative cases was handled, however incrementing the value indexed by a location of a misplaced gene. For example if a gene was traced by *PhyloStrat* to phylostrata X and to phylostrata Y ($X \neq Y$) by *QPhyloStrat* than a value indexed by a pair (X,Y) was incremented by one and vice versa for the opposite scenario. Figure 3.8 summarizes the result of such analysis. Blue squares on a diagonal indicate genes traced to the same point of origin by both methods while red squares refer to false positives and negatives (intensity of the colour is proportional to the number of corresponding cases). As expected from all of the above reported results, due to the strong non-positional identity based stratification utilized by a HD-index implemented in *QPhyloStrat*, the observed discrepancies are mostly caused by false negatives (genes to which homologs were missed by *QPhyloStrat* but can be found earlier in time by *PhyloStrat*). Though this is now a clear indicator of sacrificed methods' sensitivity, it needs to be pointed out that the observed shifts are dominantly restricted to pre-Cambrian periods, more specifically to gravitational phylostrata (attractors) like: Opisthokonta, Metazoa and Bilateria. Misplacements from pre- to post-Cambrian periods are rare and in-fact 1,590 genes out of 7,192 represents a statistically significant under-represented fraction of shifts (Hypergeometric p-value = 0.00). Given that the background effect (the effect of the underlying database) was previously excluded as a potential source underpinning the attractors, the most obvious remaining explanation is convergence between a positional similarity utilized in *PhyloStrat* approach and a non-positional identity applied by *QPhyloStrat*. However, the reason underlying the proposed convergence, exactly at the above identified periods, stays an open problem, requiring further investigation.

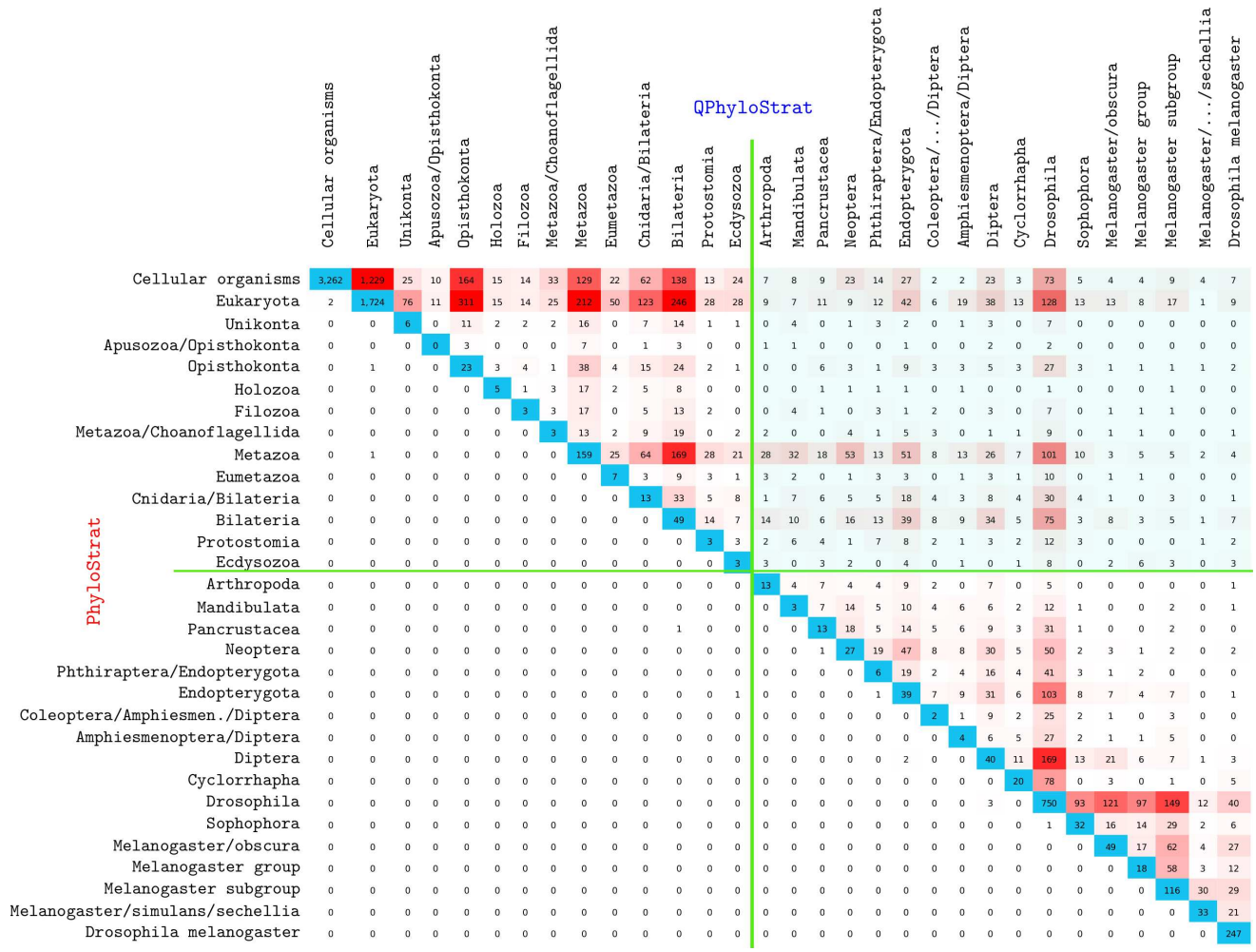


Fig. 3.8 Comparing results obtained by *PhyloStrat* and *QPhyloStrat* computation strategy. Values in blue squares reflect the number of genes traced to the same point of origin by both methods (BLAST based *PhyloStrat* and *QPhyloStrat*). Values in upper right field of the heat matrix reflect the number of genes traced to an origin point labelled on top of the map by *QPhyloStrat* algorithm, but traced to a alternative period specified on the left side of the map by *PhyloStrat* and vice versa for the lower left field of the matrix. Intensity of the red colour is proportional to the value of the number in the square. Dominating white squares in the lower left side of the matrix indicated that BLAST based *PhyloStrat* strategy rarely assigns a gene to its earlier point of origin in case when a more distant homolog exists. The same cannot be said for *QPhyloStrat* where a substantial fraction of genes (red squares) is assigned to a more recent point of origin, while at the same time having a homolog present in a more distant section of the database (section containing genes from more distantly related species). Columns in the upper right section of the matrix dominated by red squares represent the attractors (origin points toward which genes gravitate when *QPhyloStrat* method is applied). Green lines approximate the Cambrian period. Fraction of misplaced from pre- to post-Cambrian period, genes (pale blue block) is significantly lower than what could be expected by taking into account the entire population of misplaced genes (p= 0.00, Hypergeometric test)

Based on the above results, the established *PhyloStrat* pipeline should be preferred over *QPhyloStrat* algorithm in cases when runtime is not an essential factor in the analysis. However, when circumstances require fast, distribution oriented results, for example when the distribution is the subject of the analysis that includes a wide range of lineages (species), sufficient quality results can be obtained using *QPhyloStrat*.

3.1.3 Computing Gene Family Gain Events - PhyloClust Algorithm

The runtime performance of *PhyloClust* algorithm is bounded by the number of pairwise comparisons it needs to make in order to identify sequence pairs with homologous domains. Therefore, its theoretic tight boundary is proportional to the square of the total number of input sequences (genes). To test this hypothesis, a set of random sequences divided into clusters of 10 sequences each, was generated. Each sequence within a cluster contained a 300 AA region homologous to its cluster representative (75-100% identity) and each cluster contained form 50 to 400 sequences, thus ranging from 500 to 4,000 input sequences in total. Sequences were then randomly shuffled and submitted to *PhyloClust* for runtime analysis. The time measurements were preformed on a single Intel(R) 3.1 GHz processor with 4 MB of cache memory and Ubuntu 12.04 being the underlying operating system. The obtained measurements can be found in Figure 3.9.

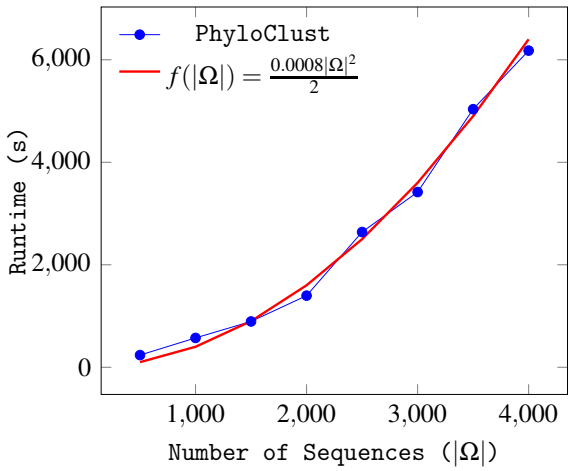


Fig. 3.9 *PhyloClust* runtime analysis. Runtime was measured with respect to the total number of sequences ($|\Omega|$) included in the analysis.

As expected a quadratic ($O(|\Omega|^2)$) runtime performance can be associated to the *PhyloClust* algorithm, thus increasing the difficulty of computation as the number of input sequences ($|\Omega|$) gets larger.

The aim of the *PhyloClust* algorithm is to cluster genes into families. Given a point of origin and a set of genes traced to that point, *PhyloClust* assigns genes into distinct families. The number of those families traced to a given time period reflects the number of gene family gain events (GFGEs), that is the number of families emerged through speciation (orthologs) or duplication (paralogs) in case of orphan genes.

Testing quality of the obtained results is based on a comparative analysis in which the number of computed gene family gain events is compared to the one expected according to the number homologs identified within a given set. Equation 2.5 describes the calculation. The obtained result can be found in Figure 3.10 with associated numerical values reported in Table 3.1.

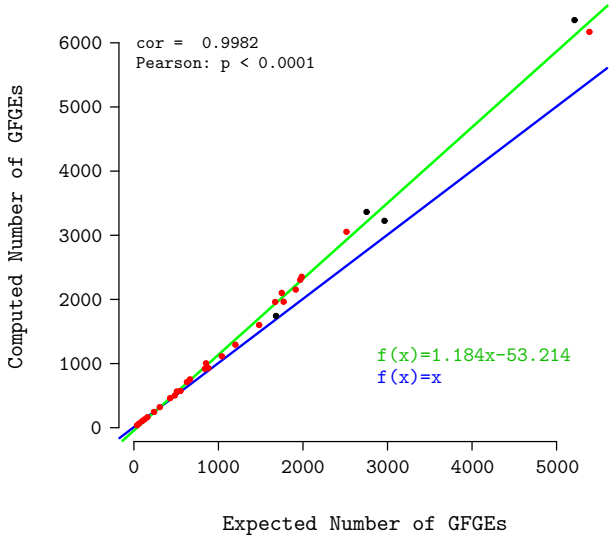


Fig. 3.10 Comparing the number of gene family gain events (GFGEs) obtained as an estimate (Eq.2.5) from the number of hits reported by BLAST (x-axis) and the number of GFGEs computed using *PhyloClust* algorithm (y-axis). Correlation between the two (green line) was computed by excluding data points the upper and lower quantiles (the 5th and 95th percentile - black points). The result indicates a strong significant correlation between the exacted and computed GFGEs. Moreover, a slight overestimation of calculated GFGEs can be observed as their numbers increase (deviation from the blue line: $f(x) = x$)

Table 3.1 List of phylostrata groups randomly selected for correlation analysis done in Figure 3.10.

Taxonomy Identifier	Number of GFGEs		Taxon
	Expected	Computed	
10197	1671	1960	Ctenophora
115784	238	246	Phaffomycetaceae
117571	5212	6354	Euteleostomi
119089	5388	6170	Chromadorea
134362	127	130	Capnodiales
1648574	2514	3054	Dikarya/Entomophthoromycota/Glomeromycota/Zygomycota
1648624	99	105	Cetartiodactyla/Perissodactyla/Carnivora
1648682	37	37	Taphrinomycetes/Pneumocystidomycetes-Schizosaccharomycetes
207598	159	164	Homininae
314293	428	461	Simiiformes
32525	663	754	Theria
32561	1040	1114	Sauria
33083	853	1004	Dictyosteliida
33511	2752	3364	Deuterostomia
4827	1200	1294	Mucorales
4890	1914	2153	Ascomycota
5073	484	503	Penicillium
5204	1481	1601	Basidiomycota
5215	550	573	Tremellaceae
5302	1771	1964	Agaricomycotina
5506	880	930	Fusarium
6049	508	566	Haplosclerida
6073	840	918	Cnidaria
6656	627	711	Arthropoda
716545	1967	2305	Saccharomyceta
7711	1749	2102	Chordata
7712	61	66	Tunicata
7718	1681	1742	Ciona
86011	2964	3225	Leucosolenida
9347	1985	2353	Eutheria
9604	306	321	Hominidae

Comparative analysis in Figure 3.10 reveals a strong (significant) correlation between the expected and the computed number of GFGEs, thus indicating a high precision of the applied clustering strategy (*PhyloClust* algorithm) when compared to calculated estimations. However, it is clear that the accuracy droops as the number of GFGEs per phylostrata group increases (green line deviates from the blue one as the number of GFGEs rises), thus empha-

sising the increasing error, with the increasing number of GFGEs.

3.1.4 Computing Gene Family Loss Events - PhLoG Algorithm

PhLoG is a simple linear time algorithm for computing the gene family loss events (GFLEs) based on the result produced by *PhyloClust*. In order to analyse its runtime behaviour a set of computed GFGE clusters on a path from Unikonta ancestral node to *D. melanogaster* species were sequentially added to the the program each time measuring how much it takes for the *PhLoG* to complete the task. The analysis was preformed on a machine with a single Intel(R) 2.7 GHz processor and 4 MB of cache memory (Ubuntu 12.04 LTS as the underlying operating system). Figure 3.11 summarizes the obtained results.

Clearly, obtained results support the conjectured linear runtime behaviour of *PhLoG* algorithm.

As far as the quality of the computed results is concerned, *PhLoG* processes the GFGEs produced by the *PhyloClust* algorithm, thus the quality of the obtained results is directly dependent on the quality of *PhyloClust* computation strategy and its outcome.

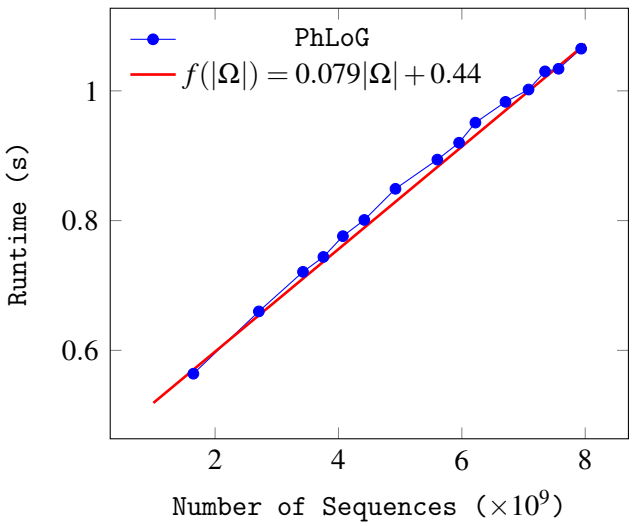


Fig. 3.11 *PhLoG* runtime analysis. Runtime was measured with respect to the total number of input sequences within the included GFGEs.

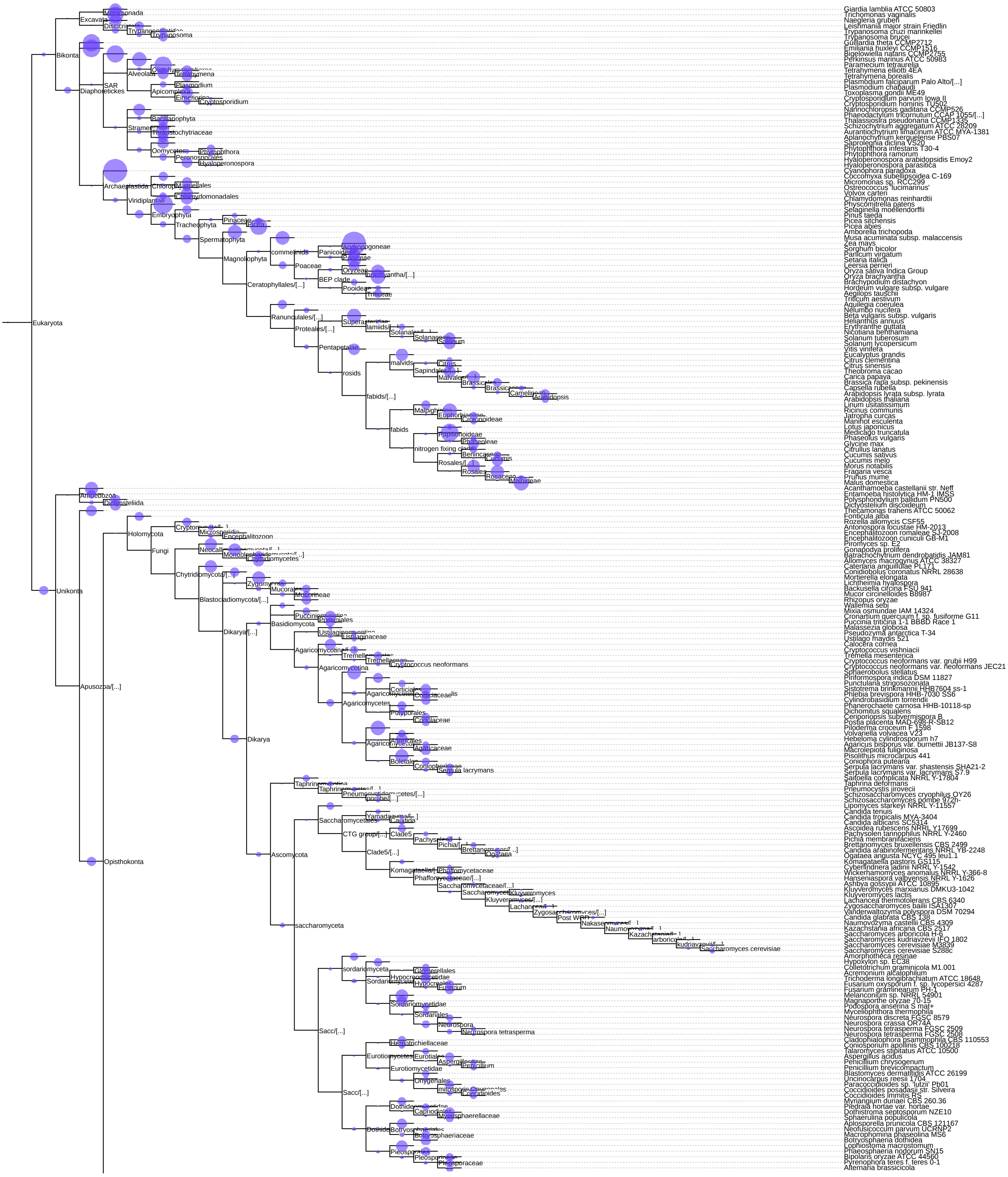
3.2 Patterns and Trends Associated to Gain and Loss of Gene Families

Over the last two decades, advancements in sequencing technologies, rise in processing capacity, ever increasing efficiency of computational strategies (software solutions), all pushed the frontier of research involving evolution of gene families. Gene families are associations based on mutual similarity between genes (in both sequence and function), descendant from a common ancestor (Walsh and Stephan, 2001). This common ancestry, as discussed in Chapters 1 and 2, certifies families as information caring structures. To be able to recognize, decipher and understand what this information is and what it implies, it is first necessary to have a complete and accurate estimate regarding the number of family gain and loss events across all available phylogenetic lineages. Thus, the first part of this section copes with gene family gain events and patterns regarding their preservation among the extent species. Following it are the results obtained by calculations involving gene family loss events, analysed within a comparative framework revealing their connection to the aforementioned gene family gain events.

In the second part, the information carried by gene families is the subject of investigation. Here, families are selected as proxies reflecting the amount of genome information carried through evolutionary time. Setting gene families as information holders rather than "raw" DNA or even individual genes, unilaterally eliminates any information redundancy (individual family members (orthologs) carry the same or similar information and therefore do not quantitatively contribute to the overall genome complexity) that can be seen as a measurement problem. Moreover, it increases the stability and information robustness to withstand various long term selective pressures (individual family members can be eliminated rather quickly but to eliminate entire families takes a lot more time Hughes and Friedman (2004)).

In the third part of this section, the rate of genome complexity change (as a measure of mutual information (Adami and Cerf, 2000; Yeung, 2006), with its foundations in Shannon's theory of information (Shannon and Weaver, 1949)) was placed under investigation.

3.2.1 Patterns Associated to Gene Family Gain Events



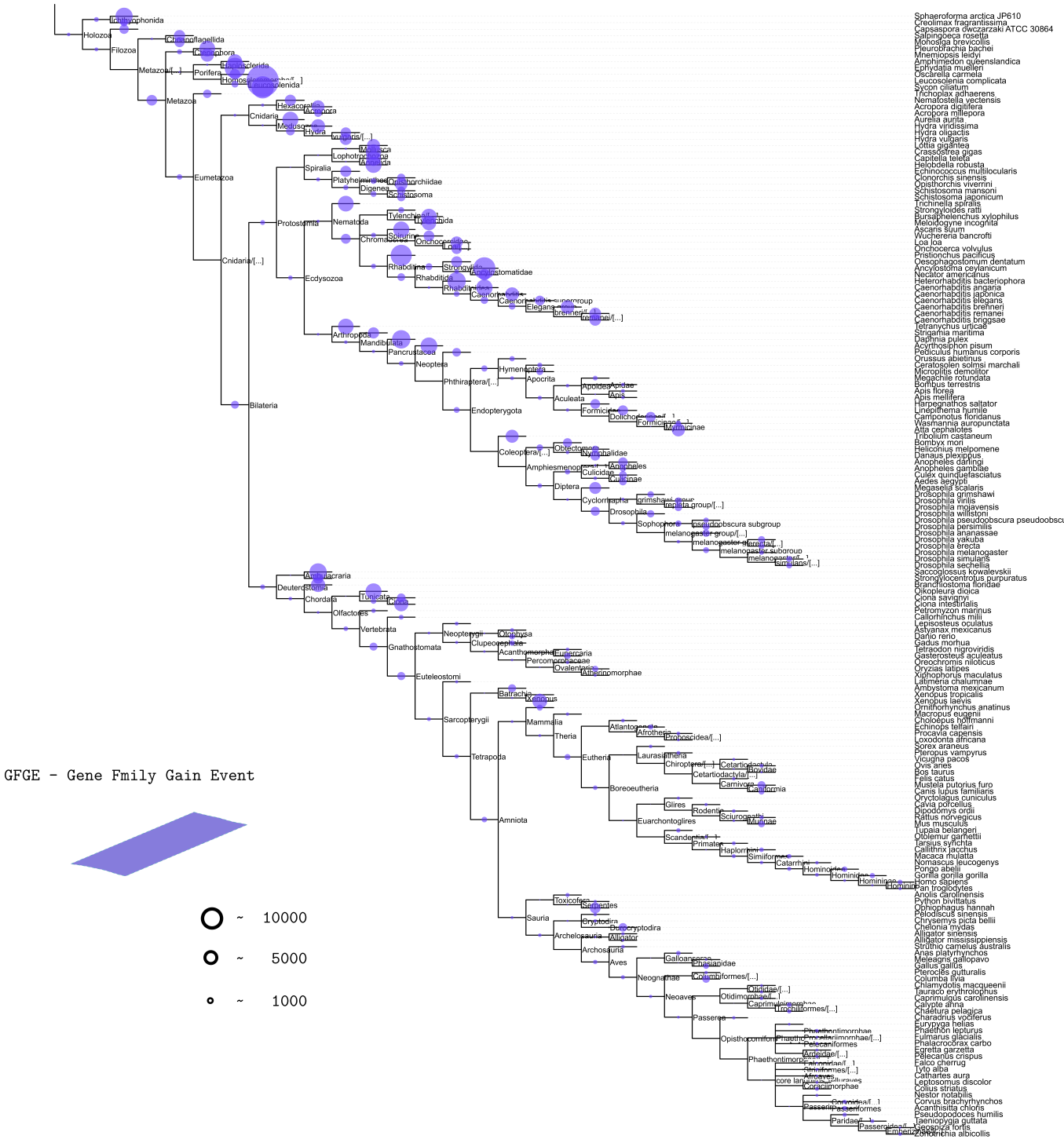


Fig. 3.12 The estimated number of gene family gain events across 383 species lineages. Size of each circle is proportional to the number of events.

Estimating the total number of gene families ($|GF|$) across a wide range of phylogenetic lineages (Figure 2.2), commences by computing the number of gene family gain events (GFGEs), as described in section 2.2.2. Each GFGE is then associated to a specific age group (phylostrata) thus enabling total number of GFGEs within a given phylogenetic period to be calculated. The result is summarized in Figure 3.12. The number of GFGEs at each ancestral node is depicted with a blue circle, sizes of which are proportional to the obtained number. It is necessary to point out that circles reflecting the number of GFGEs associated to the oldest two ancestral nodes (first cellular organisms and Eukaryote) are not reported due to technical restrictions underpinning calculations.

From the above figure the apparent difference between the number of GFGEs associated to internal nodes and those within leafs (current species nodes) is evident. Although this observation is not a surprising one and has its support in similar, previously published (related) studies (Tautz and Domazet-Lošo, 2011), reflecting upon it at this point would be considered "far-fetched" at best given that many other alternative explanations can justify the observed pattern. As an example, technical artefacts such as poorly resolved underlying phylogeny (causing the artificial rise of GFGEs in leaf nodes), complete loss of families in related taxa (Foret et al., 2010), gene annotation quality, etc. However, it should be emphasised that the clustering effect, previously labelled as an important factor in computing gene families (Kunin et al., 2005b), is systematically reduced here to a minimum and thus is expected to affect the total GFGE quantities the least[2] and therefore least contributes to the observed result.

Next, analysing the distribution of GFGEs within each individual lineage of 6 species (starting with Bikonta and Unikonta internal nodes and ending at: *D. melanogaster, H. sapiens, D. rerio, A. thaliana, S. cerevisiae*) revealed following observations:

a) The number of gene family gain events either decreases with evolutionary time (as a function of phylostrata form Unikonta/Bikonta towards the extent species) or can be described as a sinusoid-like wave (Figure 3.13).

[2] *PhyloClust*, clustering criteria is reduced to significance of similarity (*e-value* cut-off)

b) The average family size ranges between 2 members per every and/or every second family with the exception being those having the origin point traced back to Eumetazoa (3 members per family on average) (Figures: 3.14B, A.7B, 3.15B, A.8B, A.9B)

c) Gene family preservation (presence of family members in extent species) increases with evolutionary time (Figures: 3.14C, 3.15C, A.7C, A.9C)

d) The regime under which the observed increase occurs is not consistent throughout species evolutionary history. It exhibits a clear shift in its mode in periods right about, and after a well established major cladogeneses (Figures: 3.14D, 3.15D, A.7D, A.9D)

In the following paragraphs each of the above stated observations is discussed individually. The first observation addresses a general trend associated to a GFGE change across evolutionary time. When the total number of GFGEs accruing at a given evolutionary period is plotted as a function of time (phylostrata) a surprising pattern is observed. By fitting the average values using a local polynomial regression function, the resulting curve (in most cases) seams to be well approximated using a sinusoid wave function thus indicating an intrinsic oscillation associated to acquisition of genetic information (GFGEs) with the negative half cycle reaching its maximum within recent age groups. Though not in contrast with a classic perception according to which more evolved organisms (complex organisms) tend to have more information (since the net information increase still exists), it is surprising to see that its relative amount is decreasing. However, this again changes within the last couple of age groups ($\approx$ form family level to current species).

The second observation involves the average gene family size. As evident from figures 3.14B, A.7B, 3.15B, A.8B, A.9B the average family size with respect to the number of family members spans between 2 representatives per each and/or every second family with the exception (as stated above) being families emerged at Eumetazoa ancestral age group. Though the average gene family size at this point is approximately twice the average, when combined with the information about family preservation coefficient, families appearing within the Eumetazoan ancestor are not well preserved among extent species.

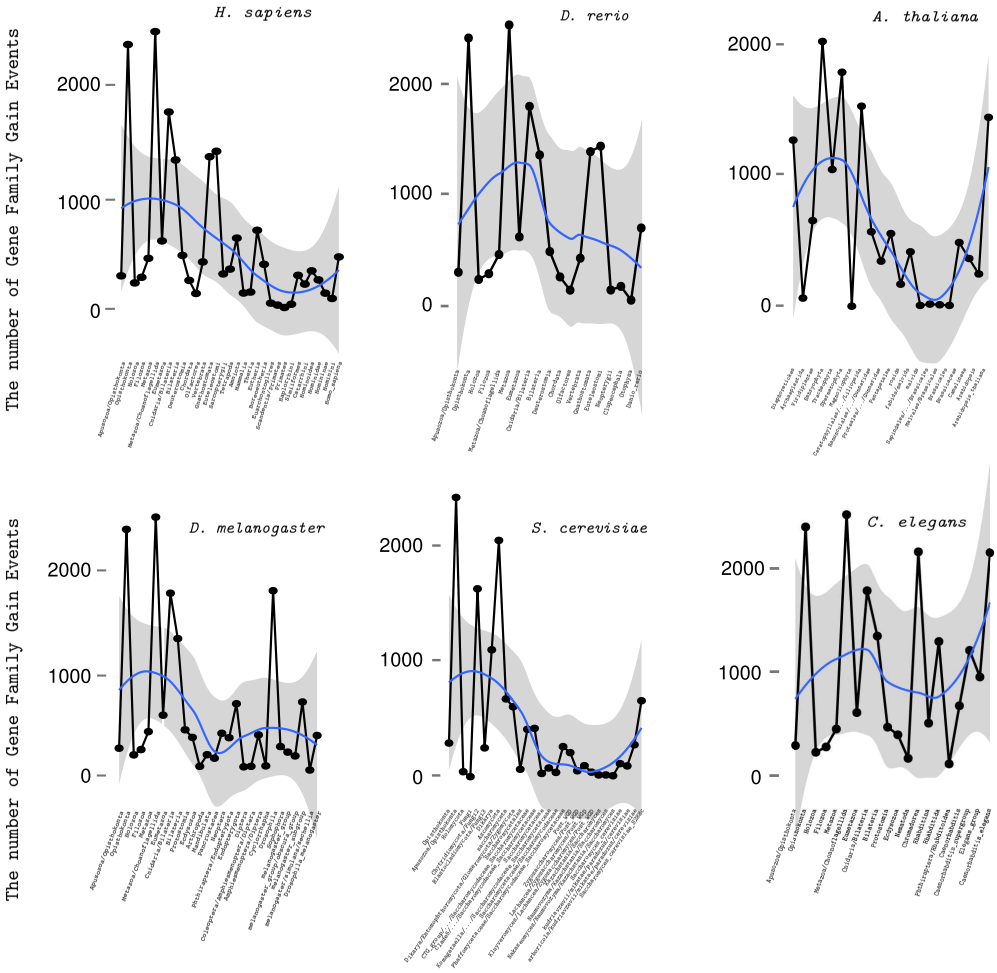


Fig. 3.13 The number of GFGEs within each phylostrata across six different lineages. Blue lines (local polynomial regression fitting curves) depicts a general trend regarding the pattern of change in the number of GFGEs on a path from evolutionary older to their current species groups. I all cases an evident decrease in the number of GFGEs can be observed.

Thus implying particular set of features associated to those families, features most probably associated to a well designed complex (higher number of genes involved in maintenance) processes specific for different lineages (low preservation). Since the focus of this study is on general, cross level (phylogenetic levels - phylostrata) regularities, the particular features associated to the Eumetazoan ancestor and/or any other ancestor will remain at this point an open problem for future investigation.

Next the pattern of gene family preservation rate as a function of time (expressed in terms of phylostrata) is investigated. Here, the gene family preservation rate is expressed

as the average fraction of species within which families formed at a given evolutionary period have been preserved (Figures: 3.14C and D, A.7C and D, 3.15C and D, A.8C and D, A.9C and D). Though, one might expect the pattern to change (perhaps in a linear way) as a function of time (the older the family is, the more relevance within the genome it holds; hence is preserved in more extent species), the results reveal a completely different trend. The preservation rate not only increases with time (the opposite of the above "older-means-more-preserved" conjecture) but exhibits a non-linear, two-phase progression pattern with a distinct "break point" (a shift in the mode) in all cases. In Deuterostomic and Proteostomic lineages this point appears to be correlated with the emergence of Vertebrates and Arthropods (right around the Cambrian period) and in plants is associated with the appearance of Rosids (Rosids include a quarter of all Magnoliophyta thus also represents a point corresponding to a key transition event in plants).

To show that pre- and post- preservation trends are significantly different and are not a product of an equivalent regime (therefore addressing the third observation), the ARIMA(0,1,0) (random walk) model, trained on preservation values up to a "break point", was used to make a projection with 95% confidence region (light gray area and 80% confidence region dark gray area), within which the expected preservation rates of younger gene families were likely to appear (Figures: 3.14D, A.7D, 3.15D, A.8D, A.9D)). The analysis clearly shows that $\approx 60\%$ of families has higher preservation rate values than those predicted by the model (95% level of confidence) and only $\approx 20\%$ of the computed values fall within the 80% confidence region (Table 3.2 summarizes the calculated fractions). Thus supporting the existence of a "break point" and a shift in gene family preservation regime that can be either explained as a cause or a consequence of the aforementioned cladogeneses. Distinguishing between the two is left for future investigation.

Another interesting result derived from analysing preservation regimes between families (Figures: 3.14, A.7, 3.15, A.8, A.9), is the impact it has on parsimony based, single (focal) species, long-term evolutionary reconstruction strategies such as phylostratigraphy (Domazet-Lošo et al., 2007), EvolMap (Sakarya et al., 2008), ProteinHistorian (Capra et al., 2012), etc.

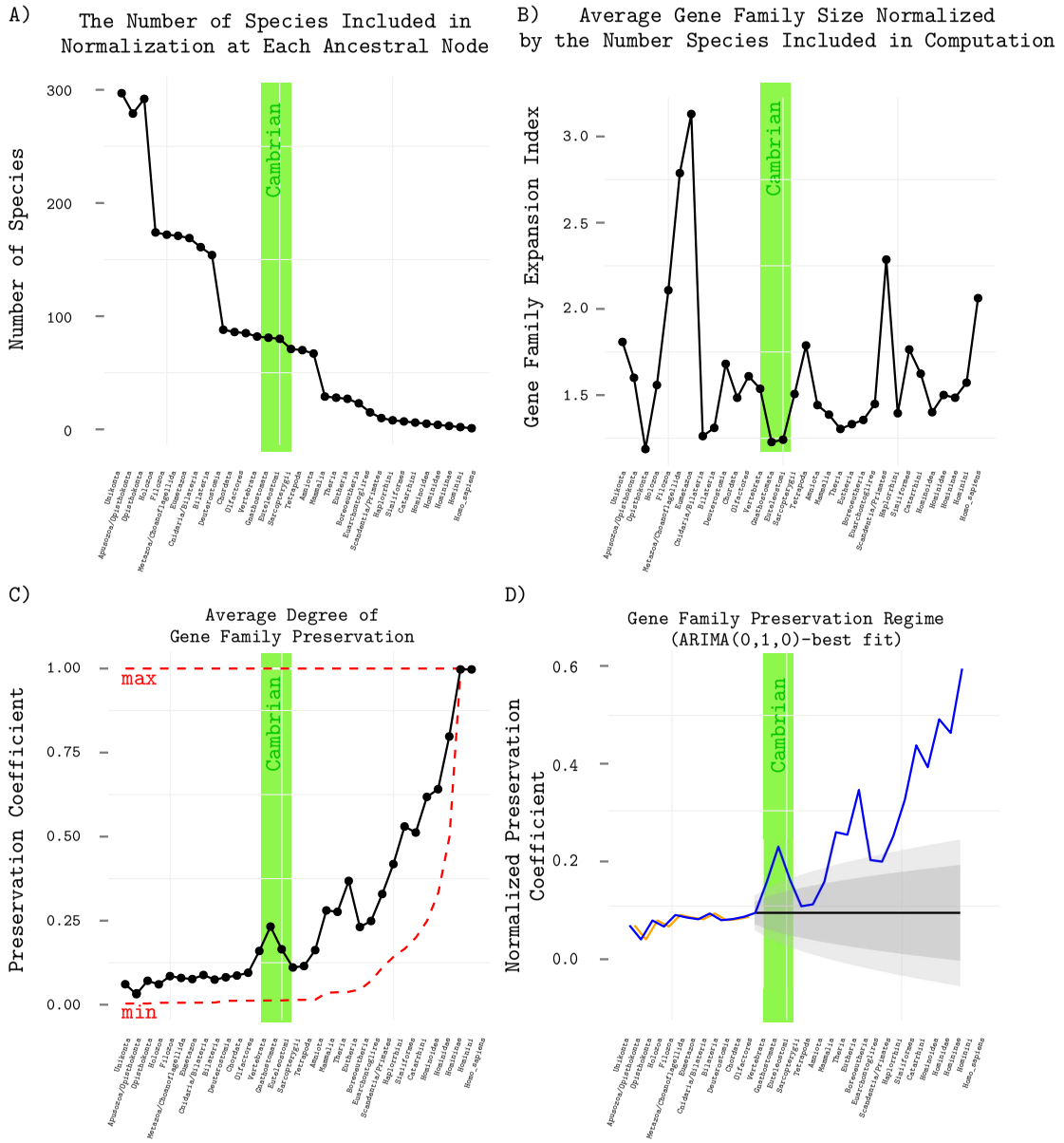


Fig. 3.14 Statistical analysis of GFGEs from Unikonta to *H. sapiens*. A) The number of species included in GFGE computation (normalization). B) The estimated average number of members within a gene family. C) The degree of gene family preservation plotted as a black dotted line. The preservation coefficient is calculated as the average number of species within which a family has been preserved divided by the total number of species associated to GFGEs with their origin traced to a given phylostrata. Red dashed lines mark the borders (minimum and maximum preservation rates each family in a given phylostrata can have). The blue line depicts a normalized average preservation with respect to calculated borders (dashed red lines). D) Gene family preservation regime analysis. Based on the average gene family preservation levels form Unikonta to Vertebrata, ARIMA(0,1,0) model was used to estimate 95% (light gray area) and 80% (dark gray area) confidence region within which preservation levels were expected to occur, if the same underlying evolutionary regime continued to influence the levels of gene family preservation. The orange line represents the ARIMA(0,1,0) data fit and the blue line, computed gene family preservation levels.

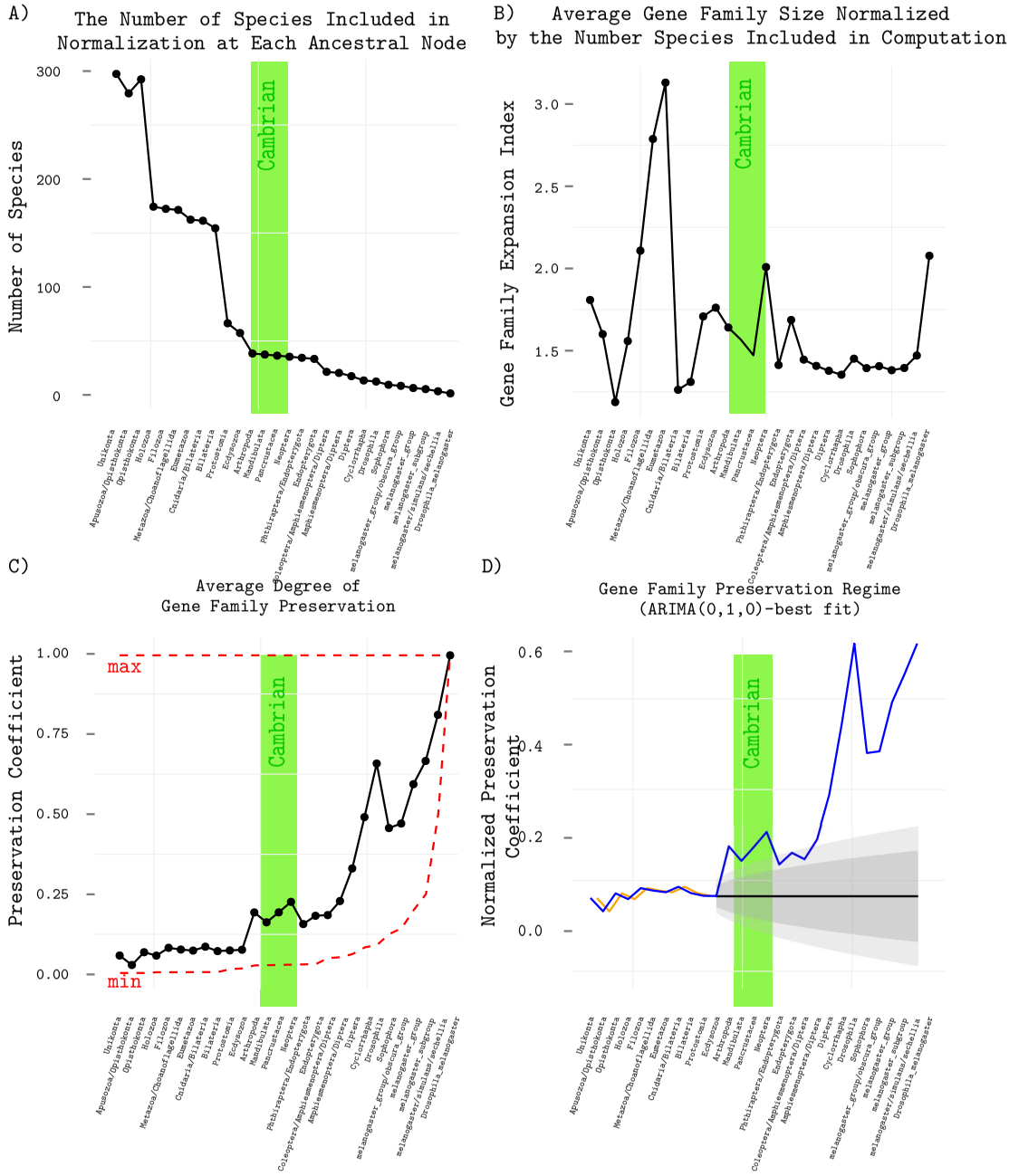


Fig. 3.15 Statistical analysis of GFGEs from Unikonta to *D. melanogaster*. A) The number of species included in GFGE computation (normalization). B) The estimated average number of members within a gene family. C) The degree of gene family preservation plotted as a black dotted line. The preservation coefficient is calculated as the average number of species within which a family has been preserved divided by the total number of species associated to GFGEs with their origin traced to a given phylostrata. Red dashed lines mark the borders (minimum and maximum preservation rates each family in a given phylostrata can have). The blue line depicts a normalized average preservation with respect to calculated borders (dashed red lines). D) Gene family preservation regime analysis. Based on the average gene family preservation levels form Unikonta to Arthropoda, ARIMA(0,1,0) model was used to estimate 95% (light gray area) and 80% (dark gray area) confidence region within which preservation levels were expected to occur, if the same underlying evolutionary regime continued to influence the levels of gene family preservation. The orange line represents the ARIMA(0,1,0) data fit and the blue line, computed gene family preservation levels.

Table 3.2 The average number of families with preservation coefficients higher than those expected. Second and third column contain the fraction of families outside the expectation areas corresponding to 95 and 80 percent confidence region.

Species name	The average number of families (%)	
	95% (conf.)	80% (conf.)
Homo sapiens	68	84
Danio rerio	67	83
Drosophila melanogaster	76	94
Saccharomyces cerevisiae	33	72
Arabidopsis thaliana	43	43
Total:	58	79

What follows from C and D plots in Figures: 3.14, 3.15 and those in the Appendix 1 (A.7, A.8, A.9) is that on average only 5-15% of gene families present in current species genomes with the early point of origin, are preserved. That is, given a *H. sapiens* genome, families within it traced to any phylostrata from Unikonta to Olfactores, represent only a fraction of gene families once present in the organism living at those time periods. Moreover, the information obtained by applying parsimony based comparative reconstruction strategies (like the phylostratigraphy approach), represents only a fraction of information associated to those evolutionary periods.

What is more surprising is the stability of the fraction (percentages) itself. Regardless of the length (time between two adjacent ancestral nodes), fractions do not change significantly, thus postulating at least two scenarios under which the observed values could be obtained:

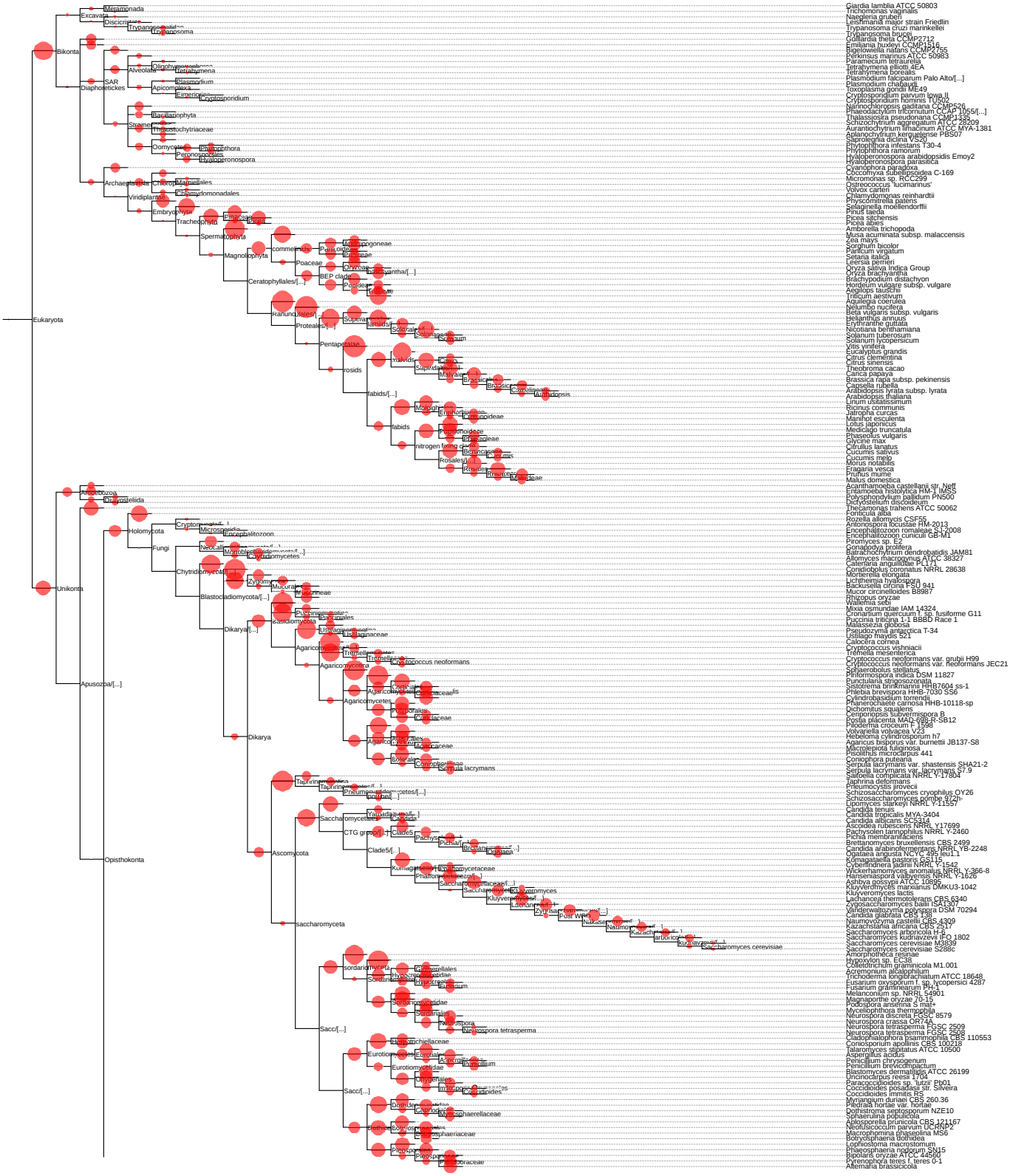
a) High cladogeneses and low preservation rates associated to early stages of evolution

b) Increased loss of ancient families in recent lineages (lineages including ancestral nodes emerged after their respective "break points").

Proving the first scenario is quite difficult since the number of current species dating to that time period is small in comparison to those appearing after the cladogeneses periods (like Cambrian). Moreover, this is also the main obstacle in calculating the inter-family preservation rates, thus rendering the proposed scenario (a), an open subject for future investigation.

On the other hand, to validate (or refute) the contribution of the proposed second conjecture, at each ancestral node, the number of gene family loss events (GFLEs) needs to be calculated. Therefore, in the next section I present results associated to GFLEs and their implications regarding the alternative scenario (b).

3.2.2 Patterns Associated to Gene Family Loss Events



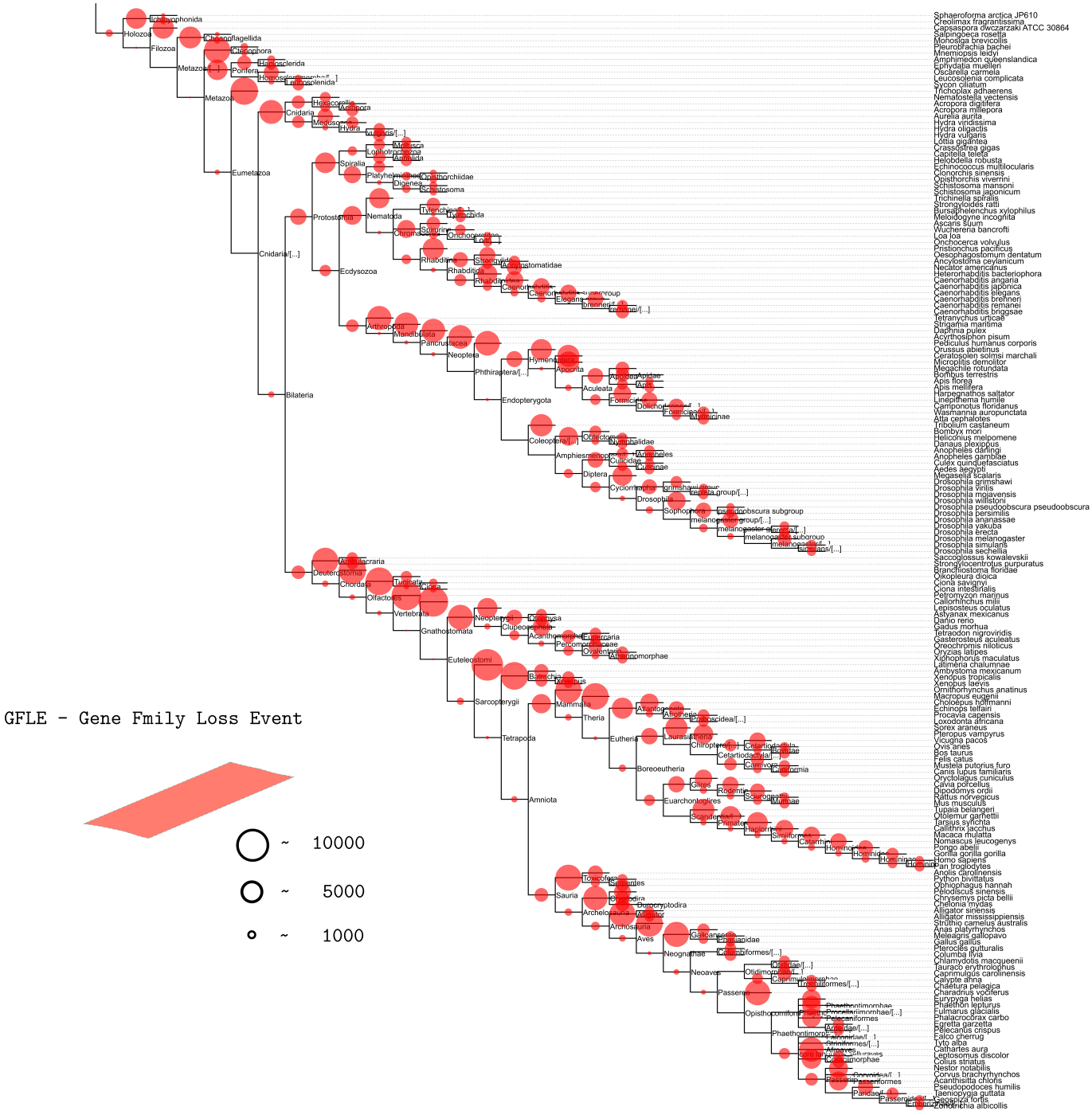


Fig. 3.16 The estimated number of gene family loss events across 383 evolutionary lineages. Size of each circle is proportional to the number of events.

Gene family loss event (GFLE) calculations were carried out as described in Chapter 2. Essentially the strategy behind the calculation can be summarized as a process of locating

the lineage splitting event after which one or more branching clades ($n-1$ clades) does not contain a member of a family emerged prior to that event. In Figure 3.16 the overall distribution of GFLEs, across large number of different phylogenetic lineages can be seen. As with GFGEs (Fig. 3.12), the size of each circle represents the number of loss events traced to a given evolutionary period.

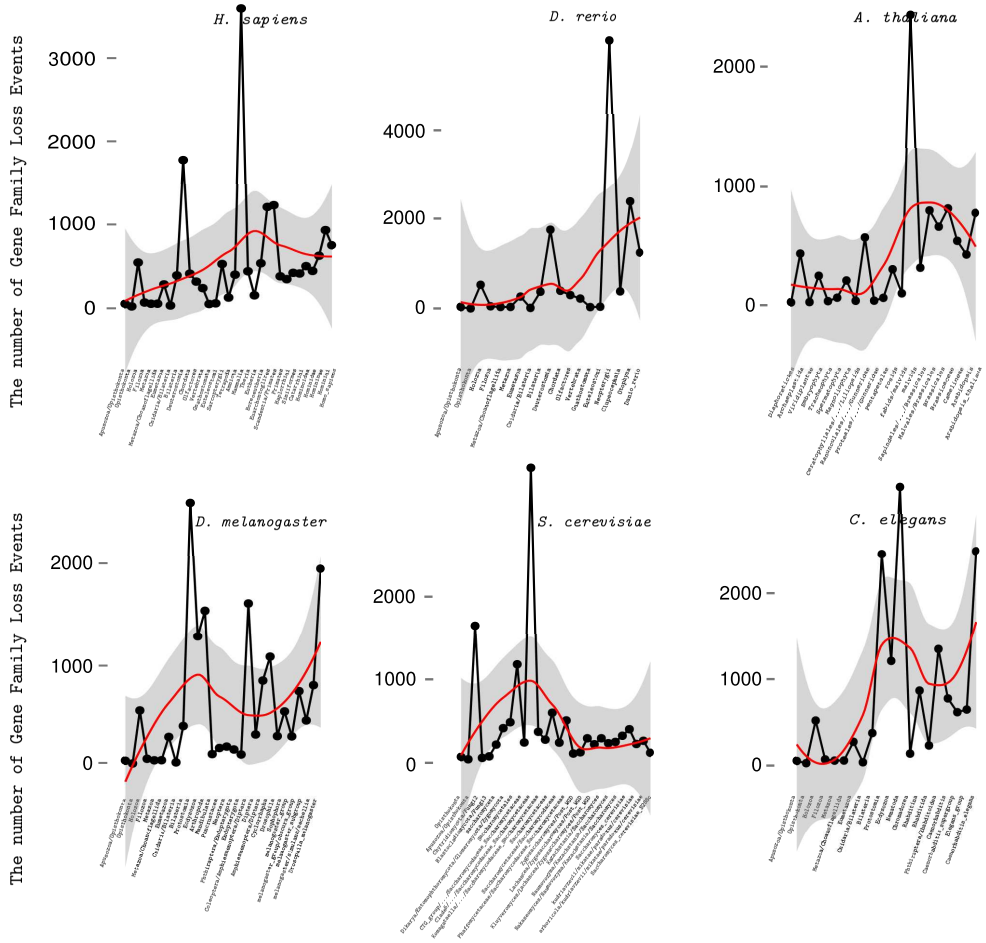


Fig. 3.17 The number of GFLEs within each phylostrata across five different lineages. Red lines (local polynomial regression fitting curves) depict a general trend regarding a pattern of change in the number of GFLEs on a path from evolutionary older to their current species groups. In five out of six lineages an increase in the number of GFLEs is evident.

First, in order to see a general pattern of loss (as it progresses with evolutionary time), the total number of loss events (GFLEs) is plotted as a function of phylostrata (Figure 3.17). Again, a non-parametric regression estimate (local polynomial regression fitting) was applied to the calculated GFLEs in order to expose a general pattern of loss. Clearly, the resulting

trend has an opposite mode to that of GFGE. In GFGEs the tendency regarding newly formed families declines as time moves forward while here the number of loss events increases. This appears to be a trend in five out of six investigated cases (*S. cerevisiae* being the exception).

Though evident, the observed rise in the number of family loss events can only be used to validate the stated observation (b) if, and only if, this increase systematically affects both older (prior to their respective "break points") and those more recently emerged families, since any alternative would be in direct contradiction to the result reported in the previous section. Nevertheless to verify and eliminate any potential alternative causes, the distribution of loss events as a function of its gain period was further analysed using a set of heat matrices and frequency plots (Figures 3.18A-D and 3.19A-D). From Figures 3.18A(B) and 3.19A(B), follows that in the course of evolution of both *H. sapiens* and *D. melanogaster*, there are so-called "hotspots", time periods in which the number of extinct families is quantitatively higher than in others. In *H. sapiens* these periods include those ending with the emergence of Holozoa, Deuterostomia and Mammalia most of which emerged at Unikonta, Opisthokonta, Metazoa, Bilateria, Gnathostomata and Euteleostomi. A similar pattern is evident in *D. melanogaster*.

What is even more intriguing is the fact that these great reductions in family counts are associated to evolutionary transitions currently labelled as those in which interaction with the environment increases. For example the appearance of Protostomic and Deuterostomic animals marks the emergence of a through gut (Martin-Duran et al., 2012) that can be seen as a mechanism alleviating the processing of nutrients from the environment. Up to that point on a path form Cnidarian/Bilaterian ancestor to true Bilateral animals, the gut was a simple, single-end cavity in which no specialization of digestion tract was possible. Once a through gut emerged (combined with higher mobility) higher intake of nutrients from the environment was possible which possibly rendered some pathways (and thus gene families) unnecessary, subjecting them to loss. However, this particular scenario is somewhat difficult to prove since the absence of information from key taxa, still prevents the reconstruction of the ancestral developmental mode.

H. sapiens

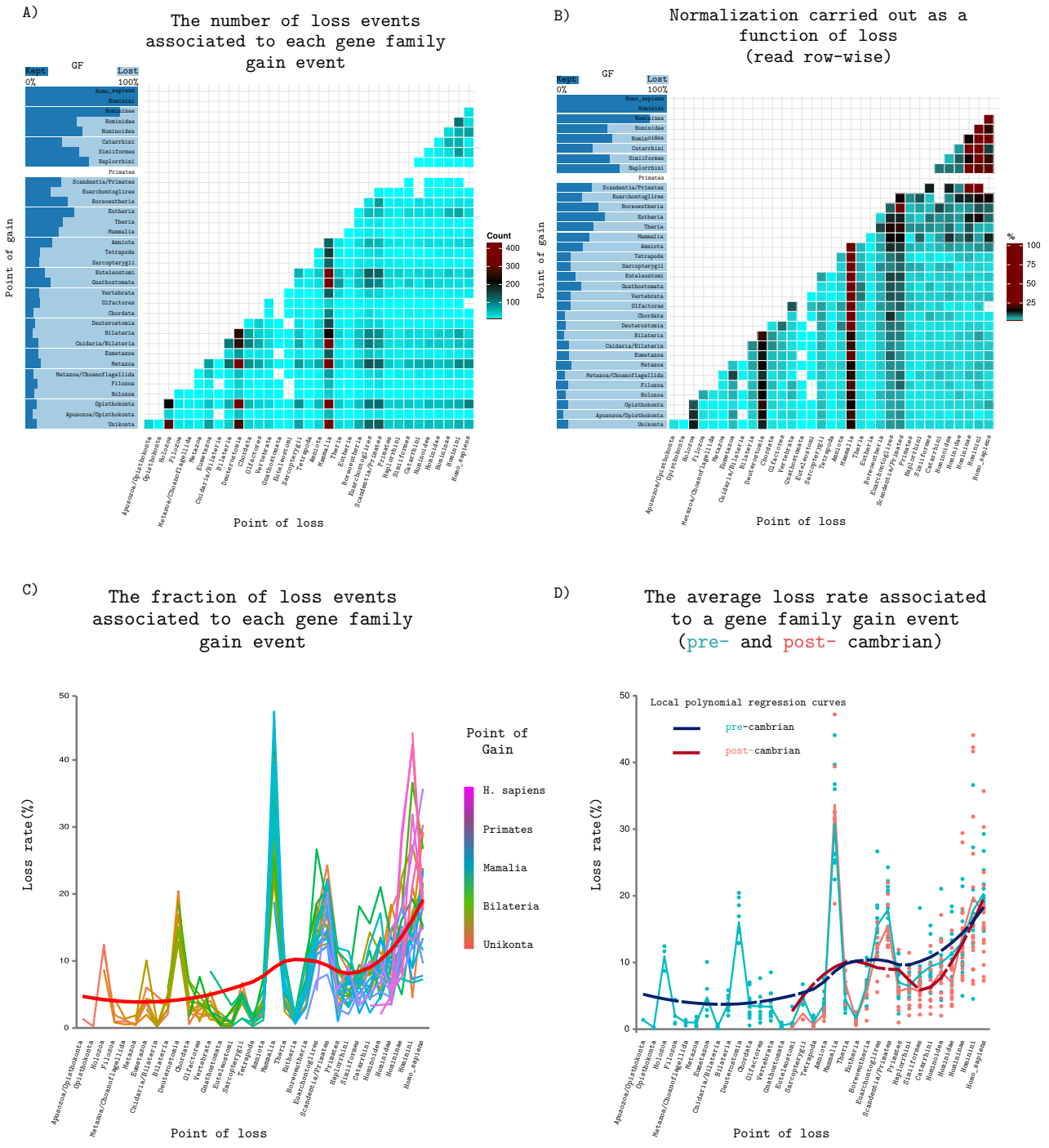


Fig. 3.18 The distribution of gene family gain events of the identified loss events within *H. sapiens* lineage associated to a given phylostrata group. (A) The number of loss events associated to each gene family gain event. (B) The number of loss events associated to each family gain event normalized by the overall number of extinct families associated to a given evolutionary period. (C) Gene family loss rates emerging at specific evolutionary periods. The red line (local polynomial regression fitting curve) depicts a general trend regarding the loss rate. Noteworthy is a clear increase in rate of extinct families. (D) Gene family loss rates emerging before and after the Cambrian period. In both cases the rate increases as a function of time.

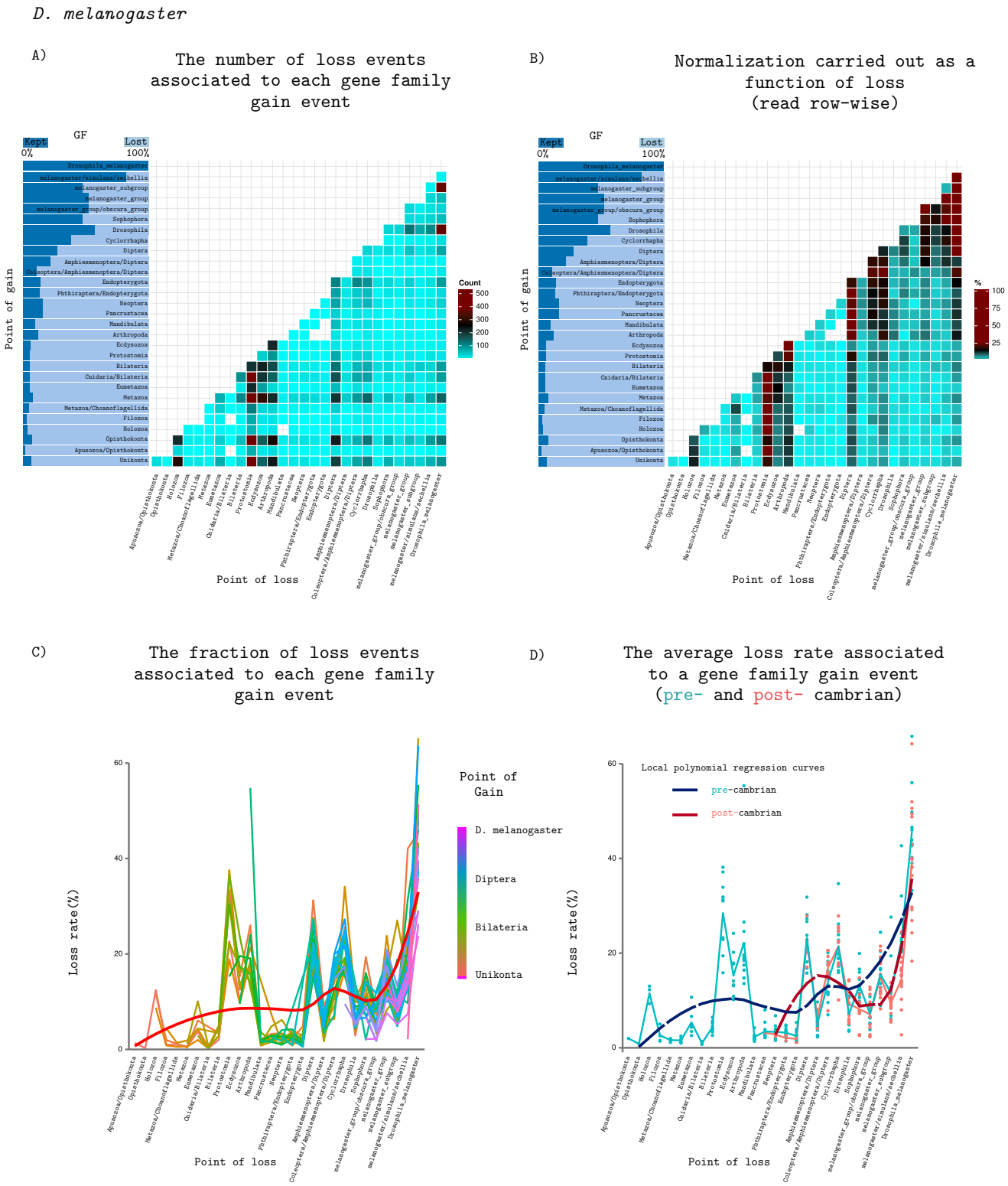
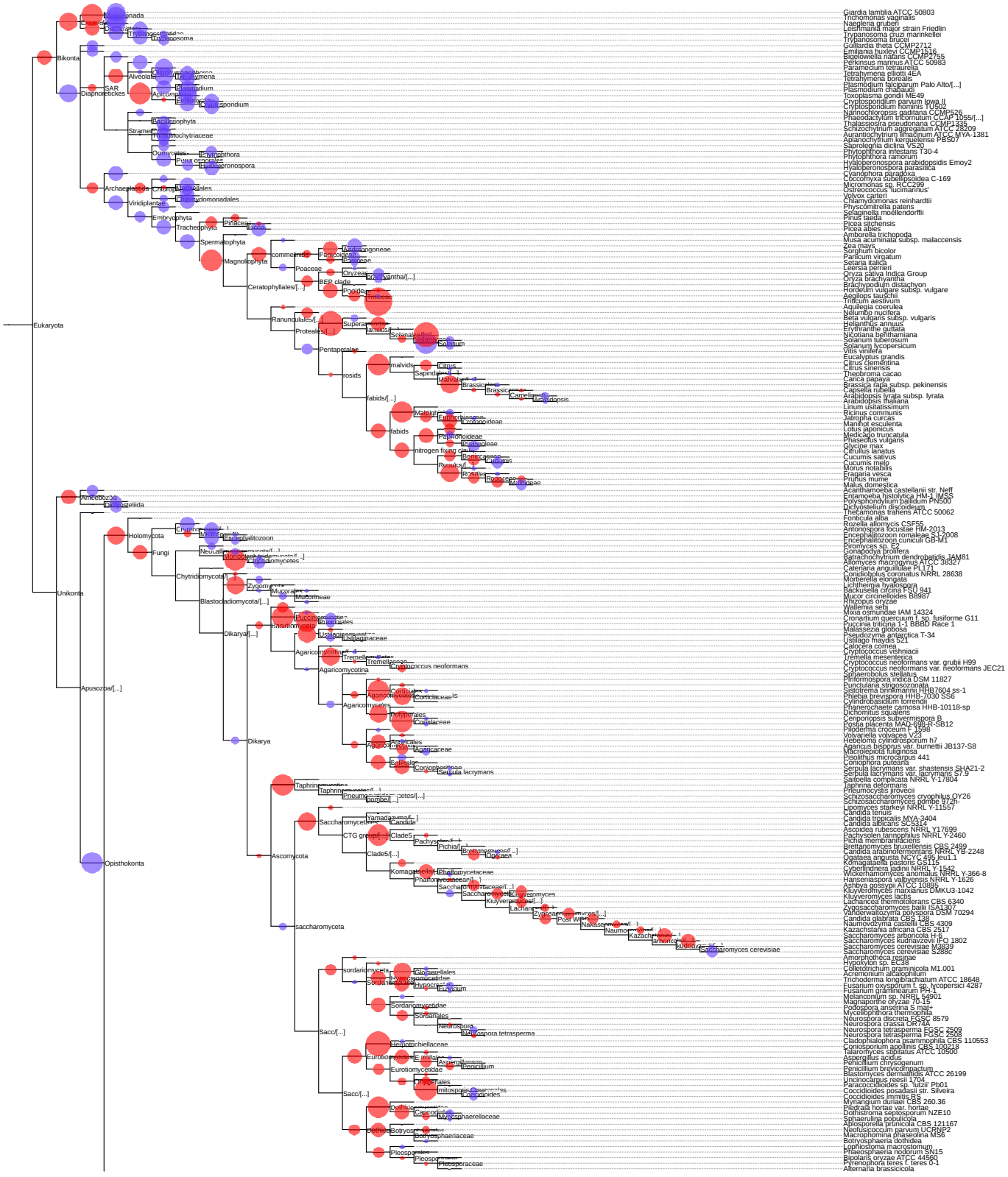


Fig. 3.19 The distribution of gene family gain events of the identified loss events within *D. melanogaster* lineage associated to a given phylostrata group. (A) The number of loss events associated to each gene family gain event. (B) The number of loss events associated to each family gain event normalized by the overlall number of extinct families associated to a given evolutionary period. (C) Gene family loss rates emerging at specific evolutionary periods. The red line (local polynomial regression fitting curve) depicts a general trend regarding the loss rate. Noteworthy is a clear increase in rate of extinct families. (D) Gene family loss rates emerging before and after the Cambrian period. In both cases the rate increases as a function of time.

Furthermore, when the number of families that were kept and lost within a given period are compared (blue horizontal bar-plots in Figures 3.18A(B) and 3.19A(B) it becomes clear that up until the Vertebrates (Arthropods) the fraction of families kept is comparably low to the ones lost (with fraction of loss being $\approx 90\%$), thus supporting the previously reported result regarding GFGE preservation rates. Moreover, after the mentioned periods almost linear reduction in the fraction of lost families can be observed (followed by an increase in relative number of families preserved). To further investigate patterns associated to gene family loss rates, the fraction of families lost at each phylostrata was divided by the number of families present up the that point. The computation was repeated with respect to each gain period. The obtained results summarized in Figures 3.18C and 3.19C (red line) both indicate a clear rise in gene family loss rates with time.

To exclude any potential biases caused by difference in loss rates associated to young and old families, the calculations were repeated separating families into those emerging before and after the respective time period (emergence of Vertebrata and Arthropoda). Figures 3.18D and 3.19D summarize the obtained result. Regardless of the age group (young/old), both patterns demonstrate a clear general increase in loss rate, thus confirming the conjecture stated in the summary of the previous section.

3.2.3 Gene Family Turnover Rate



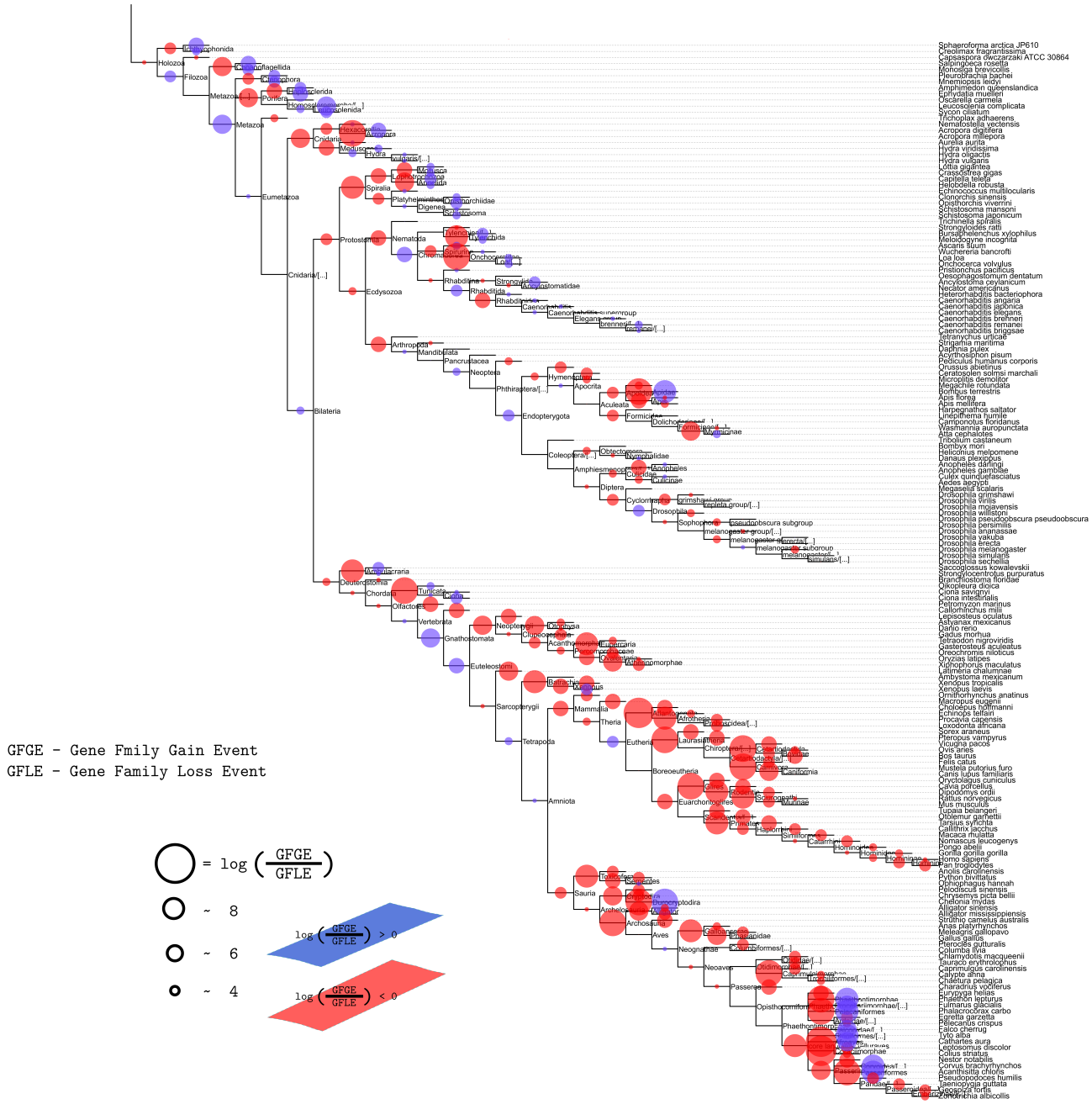


Fig. 3.20 Comparing the number of GFGEs and GFLEs across different ancestral and current species nodes. Size of circles are proportional to the number of events. Blue circles indicate the higher number of gene family gain events over gene family loss events while red circles are associated with the opposite trend. Size of the circles are proportional to $\log(\frac{|GFGE|}{|GFLE|})$ values.

By estimating the number of both, family gain and loss events across a wide range of phylogenetic lineages, it is further possible, as explained in Chapters 1 and 2, to investigate patterns

and trends associated to their turnover rates. As stated in section 2.3.1 the task can be carried out by calculating the ratio between the number of GFGEs and GFLEs at each evolutionary period within a given lineage. Given the resulting ratio is larger than 1, the number of newly formed families exceeds the extinct ones and vice versa. Figure 3.20 summarizes the obtained results. A simple visual inspection reveals that none of the extent lineages is unilaterally dominated by either increase or decrease in the number of gene families, therefore, directly implying a dynamic, alternating pattern of change attached to species gene family evolution. To confirm this observation, six out of 383 lineages have been extracted and their individual turnover rates calculated (Figure 3.21).

The first and most obvious pattern that reveals itself by examining those lineages is the erratic exchange between GFGE and GFLE dominance. The second pattern is the evident tendency toward increasing the number of gene family loss events as the time moves forward, while early evolutionary periods are dominated by an increase in gene family gain events.

By mapping down known key evolutionary features onto each of the plots in Figure 3.21, connections between relative increase in GFGEs (GFLEs) and the emergences of reported novelties becomes evident. Though it is not surprising to see a connection between an emerging novelty and the increase in genome information content (as reported by Domazet-Lošo et al. (2007)), higher gene family loss rate in periods associated with the appearance of mammals and "through gut" organisms like proto- and deuterostomic bilaterials is intriguing at least. In the previous section an observation was made regarding high number of extinct families associated to those two periods, however at that point no claims could be made regarding its relative abundance. Through this analysis it becomes evident that in comparison to the number of gain, loss events are dominating those two periods. The question remains whether these losses are cause or a consequence of the emergent phylogenetic orders. As a logical explanation, a scenario under which new features gained at those evolutionary epochs increased the interaction with the environment, enabling organisms to get their essential nutrients from the environment rather than synthesizing them themselves, many molecular pathways and thus gene families became redundant and therefore extinct (streamlining (Dufresne et al., 2005; Giovannoni et al., 2005a)).

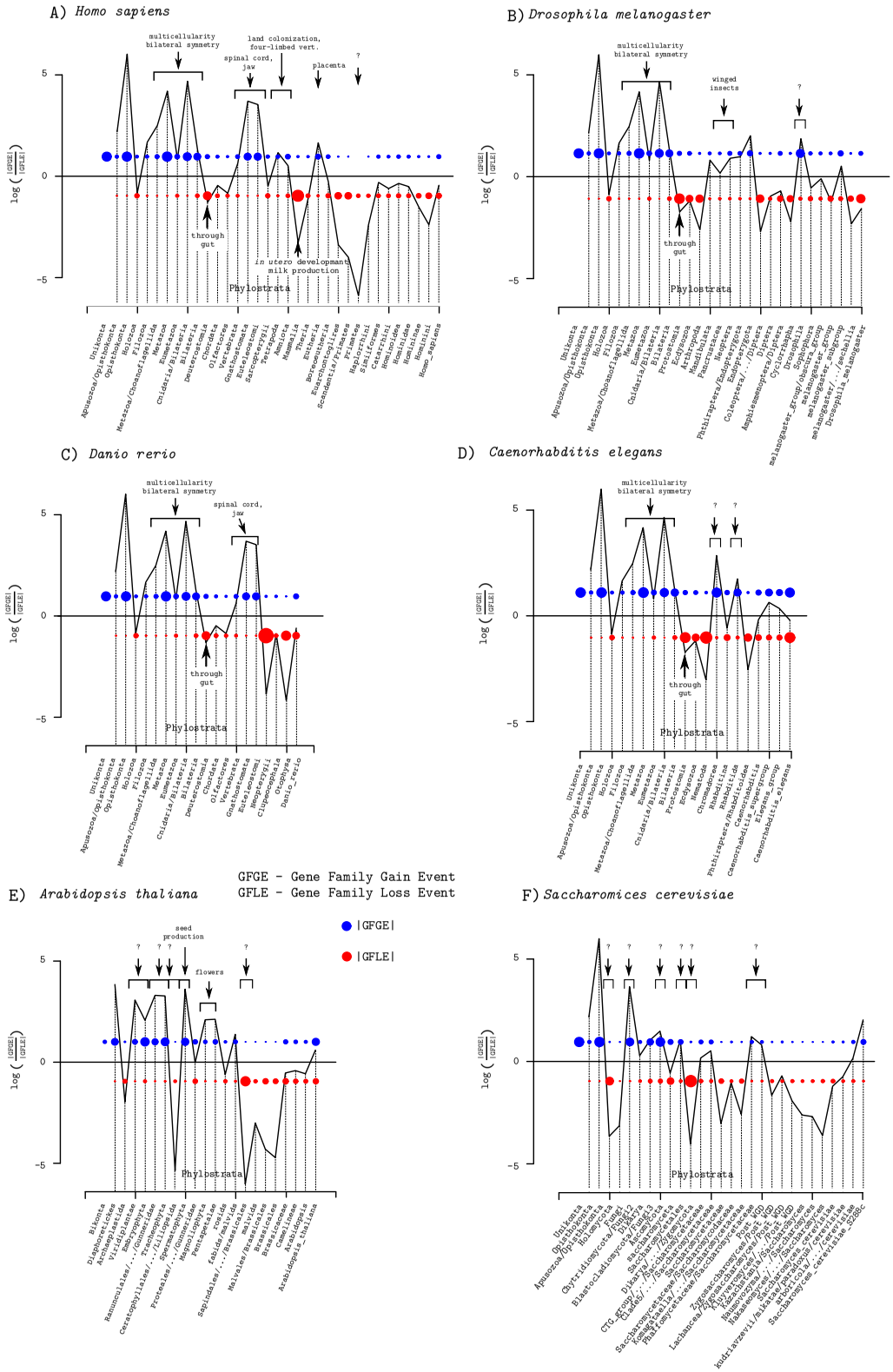


Fig. 3.21 Comparing the number of gene family gain and loss events within a given species lineage : (A) *H. sapiens*, B) *D. melanogaster*, C) *D. rerio*, D) *C. elegans* E) *A. thaliana* and F) *S. cerevisiae*). Size of each circle is proportional to the number of events. Blue circles indicate the number of gene family gain events while the red ones reflect the number of gene family loss events. Intriguing connection between the observed GFGEs and known evolutionary innovations is labelled above each plot in cases where the information was available. Note that innovations are not always accompanied by increase in the genome information content (GFGEs)

Aside form the evident oscillations in GFGE/GFLE ratios and their connection to different evolutionary periods, Figure 3.21 uncovers another interesting link. By comparing GFGEs and GFLEs individually (per phylostrata) across evolutionary time, regularity emerges; increase in the number of family gain events (Figure 3.21, blue circles) is followed by a decrease in the number of family loss events (Figure 3.21, red circles).

To prove this, the number of gain and loss events from six different lineages were compared and their counts correlated (Figure 3.22). The resulting pattern, supported by a significant negative correlation between family gain and loss events confirms the above stated observation: when GFGEs are high, GFLEs are low and vice versa. From this result the mechanism(s) underlying this link can only be interpreted as a pure guess therefore the cause for this relationship is left for future investigation.

Needless to say that all of the above results are in accord with those obtained separately for both family gain and loss events. However, what this type of analysis reveals (and cannot be derived from previous results) is the insight into mutual dependency and relation between GFGEs and GFLEs, which appear to be antagonistically connected.

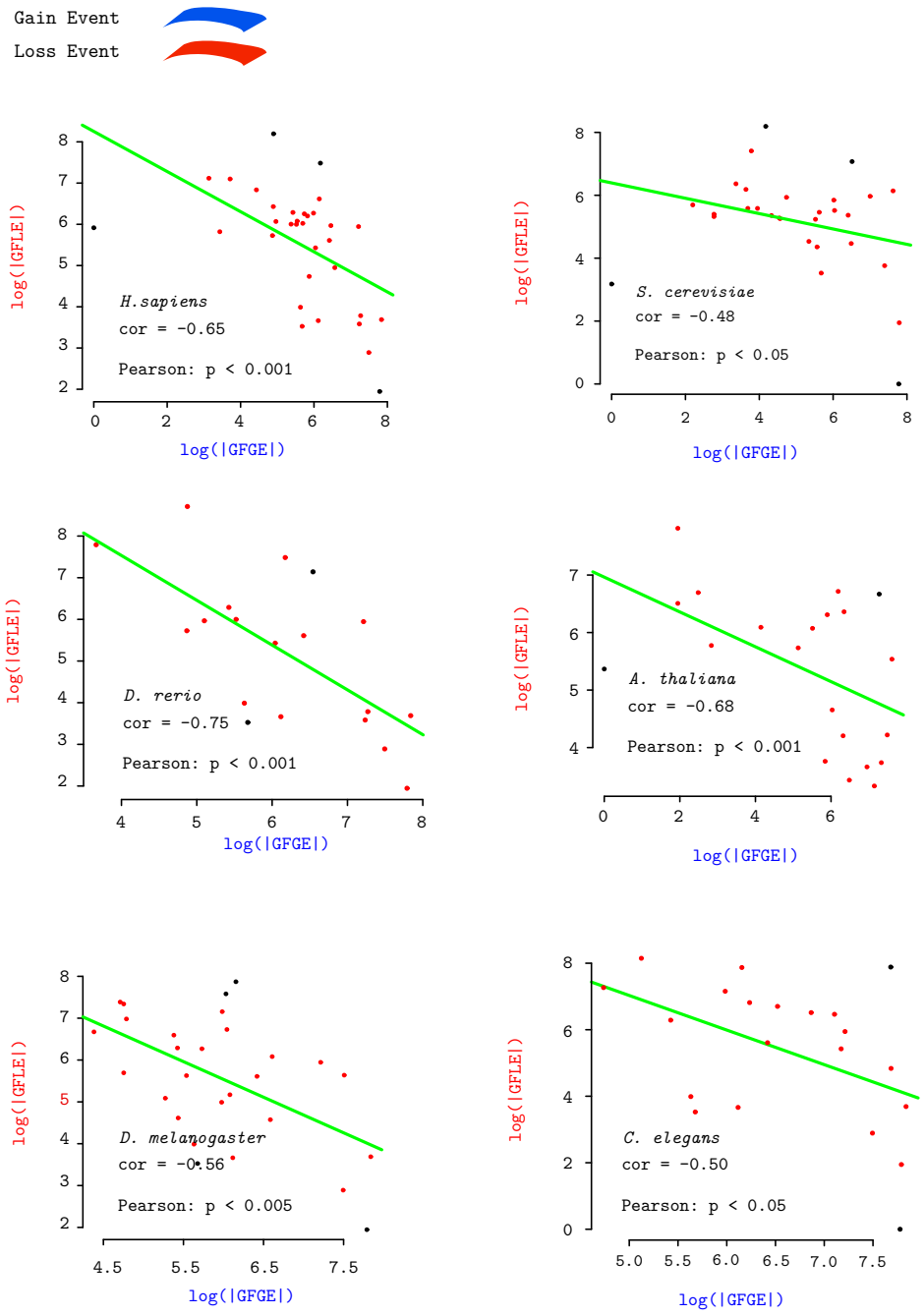


Fig. 3.22 Analysing the relation between the number of gene family gain and loss events within four different lineages. Correlation between the two (green line) was computed by removing points within the 5th and 95th percentile of the data (black points). In all six cases a significant negative correlation between gain and loss has been detected, implying the existence of an underlying antagonistic process connecting the two.

3.2.4 Reconstructing Ancestral Gene Family Content and Complexity Patterns

As described in Section 2.8, by knowing the number of gain and loss events associated to each evolutionary time period, it is further possible to quantify the ancestral gene family content. However, unlike in the analyses conducted so far, where each evolutionary epoch was studied independently, ancestral gene family content reconstruction is a "vertical" (cross time) analysis in which the estimated number of families, at a given epoch, depends on the estimations made prior to it. In general terms, this means that any produced pattern is sensitive to its initial conditions such that a small change in a prior estimate (bad estimate), in one such deterministic non-linear system, can affect and drastically change its end result. In order to reduce this effect to its minimum and at the same time eliminate any potential phylogeny related branching biases affecting the general pattern associated to gene family reconstruction estimates, ancestral gene family content reconstruction analysis was done first for *D. melanogaster* lineage by including up to two species in each branching clade. Since fairly low number of species were included in this reconstruction, the analysis was repeated by applying both *PhyloStrat* and *QPhyloStrat* as the initial gene gain computation strategies. That way one additional factor could be included in estimating the robustness of the reconstruction process. The obtained results are summarized in Figure 3.23.

First let us consider the effects of *PhyloStrat* and *QPhyloStrat* on the final result (shape of the distribution). In both cases the arching pattern ($\cap$) is evident. As time moves forward, the number of gene families within a given ancestral genome increases until it reaches $\approx$ Bilateria (Fig. 3.23A). At that point the evolution appears to be dominated by reduction, a decrease in the number of families associated to ancestral genomes. Moreover, the pattern associated to competing gene gain computation strategies (*QPhyloStrat* underestimates the age of genes, that is in some cases *QPhyloStrat* fails to identify a more distant homolog (Section 3.1.2)), is present in the overall reconstruction analysis (higher number of gene families are associated to more recent evolutionary periods and lower numbers to those earlier ones when compared to the *PhyloStrat* based pipeline). However, when subjected to QQ and asso-

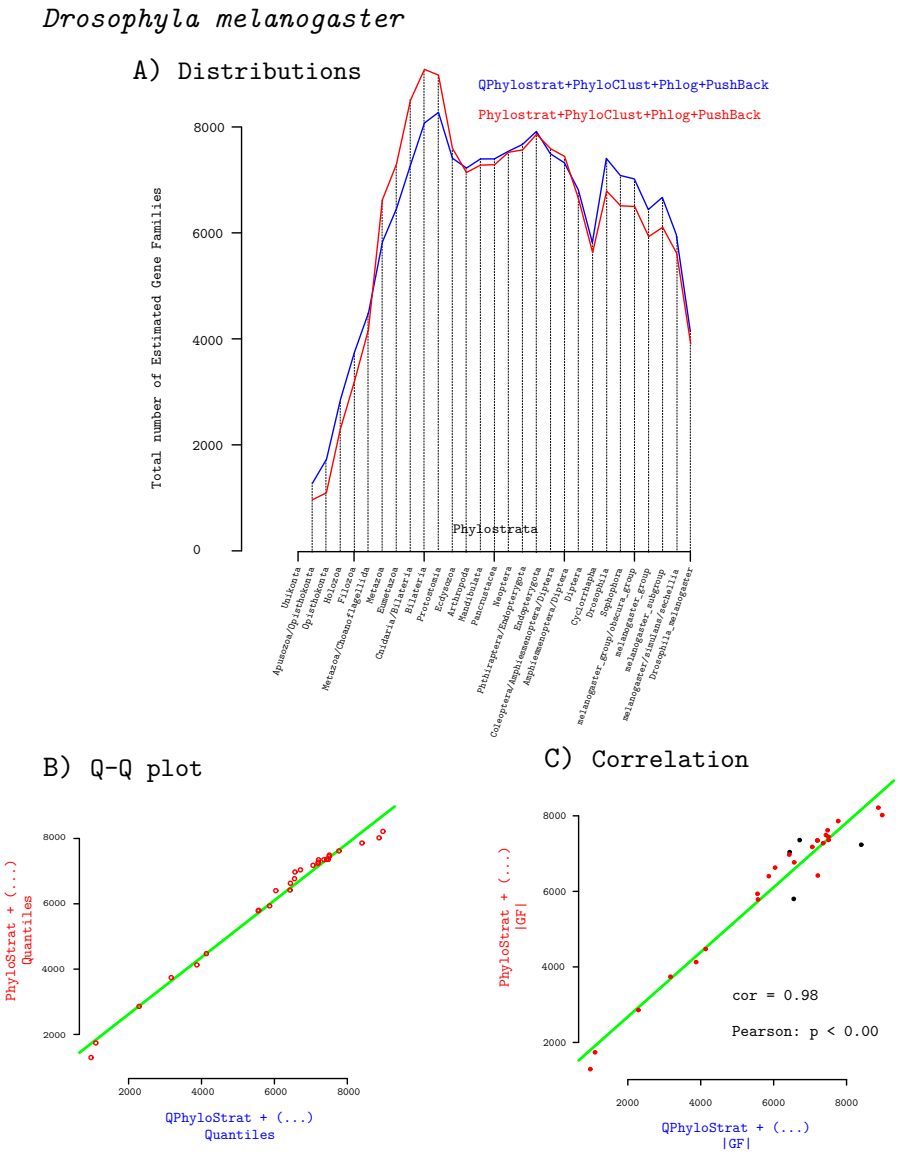
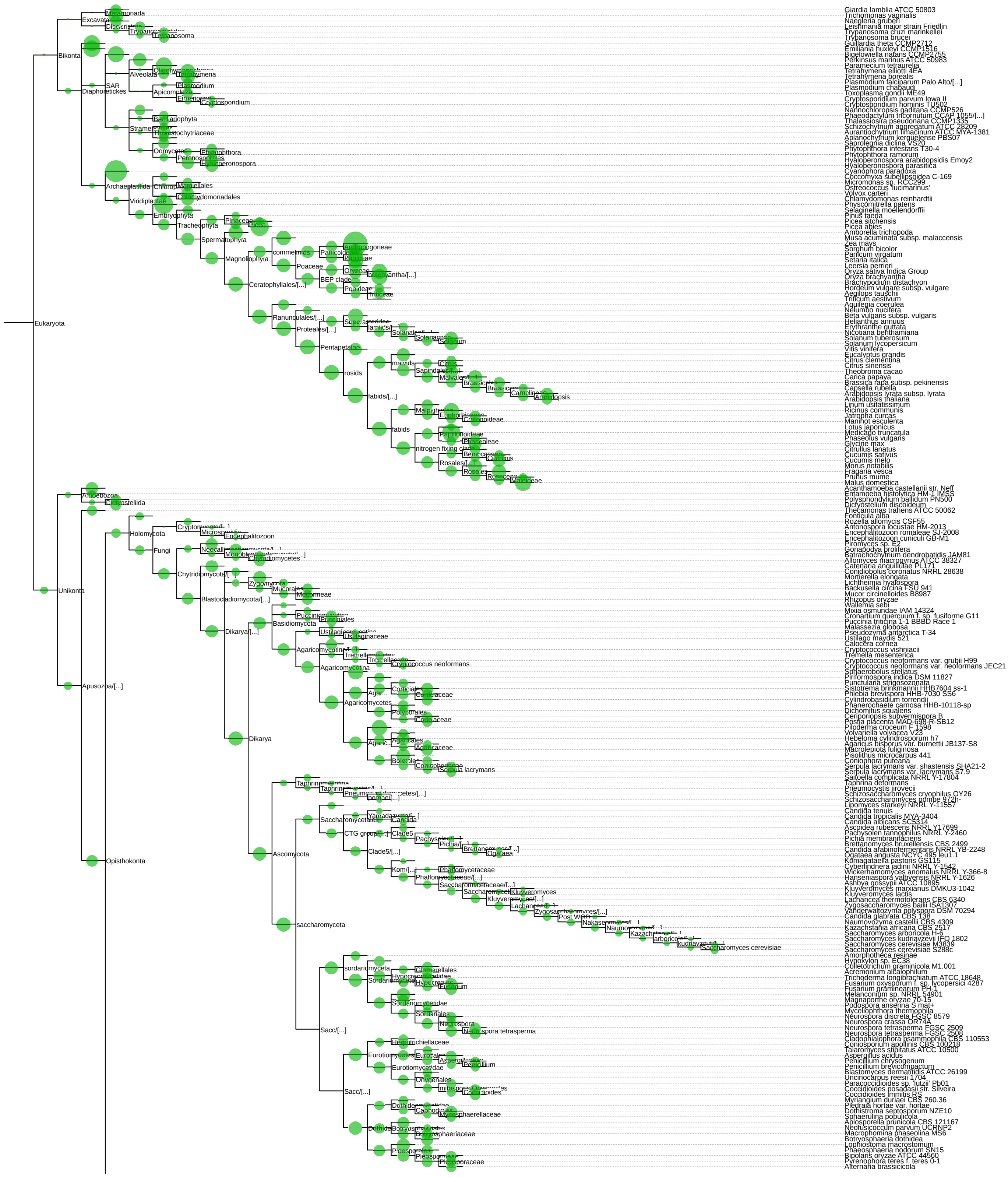


Fig. 3.23 The distribution of the total number of gene families in *D. melanogaster* lineage. At each node the number of families was estimated by including up to two species (plus *D. melanogaster*) in the calculation.

ciated correlation analysis the two show highly significant correlation regarding gene family numbers and the shape of compared distributions (Figures 3.23B and C). This result clearly demonstrated the preservation of the obtained pattern regardless of the initial gene gain computation approach. What's more, a classic BLAST-based gene gain calculation (*PhyloStrat*) accentuates the arching pattern,


Eukaryota
Bikonta
Unikonta
Excavata
Metamonada
Discicristata
Trypanosoma
Diaphoretickes
SAR
Alveolata
Tetrahymena
Plasmodium
Apicomplexa
Cryptosporidium
Bacillariophyta
Thraustochytriaceae
Oomycetes
Phytophthora
Peronosporales
Hyaloperonospora
Archaeplastida
Viridiplantae
Mamiellales
Chlamydomonadales
Embryophyta
Tracheophyta
Pinaceae
Picea
Spermatophyta
Magnoliophyta
commelinids
Poaceae
Andropogoneae
Paniceae
Oryzeae
BEP clade
brachyantha/[...]
Pooideae
Triticeae
Ceratophyllales/[...]
Ranunculales/[...]
Proteales/[...]
Pentapetalae
lamiids/[...]
Solanaceae
Solanum
rosids
malvids
Citrus
Sapindales/[...]
Brassicaceae
Arabidopsis
fabids/[...]
fabids
Crotonoideae
Papilionoideae
Phaseoleae
nitrogen fixing clade
Cucumis
Rosales/[...]
Rosales
Maloideae
Amoebozoa
Dictyosteliida
Apusozoa/[...]
Opisthokonta
Holomycota
Fungi
Encephalitozoon
Chytridiomycetes
Chytridiomycota/[...]
Blastocladiomycota/[...]
Mucorineae
Dikarya/[...]
Dikarya
Basidiomycota
Pucciniales
Ustilaginaceae
Agaricomycotina
Cryptococcus neoformans
Agaricomycetes
Corticiaceae
Polyporales
Coriolaceae
Agaricales
Agaricaceae
Boletales
Serpula lacrymans
Ascomycota
Saccharomycetales
Candida
CTG group/[...]
Clade5
Clade5/[...]
Ogataea
Kom/[...]
Phaffomycetaceae
Phaffomycetaceae/[...]
Kluyveromyces
Kluyveromyces/[...]
Zygosaccharomyces/[...]
Post WGD
Saccharomyces cerevisiae
saccharomyceta
sordariomyceta
Hypocreomycetidae
Glomerellales
Fusarium
Sordariomycetidae
Sordariales
Neurospora
Neurospora tetrasperma
Sacc/[...]
Herpotrichiellaceae
Eurotiomycetes
Eurotiomycetidae
Penicillium
Onygenales
Coccidioides
Mycosphaerellaceae
Botryosphaeriaceae
Pleosporales
Pleosporaceae
Giardia lamblia ATCC 50803
Trichomonas vaginalis
Naegleria gruberi
Leishmania major strain Friedlin
Trypanosoma cruzi marinkellei
Trypanosoma brucei
Guillardia theta CCMP2712
Emiliania huxleyi CCMP1516
Bigelowiella natans CCMP2755
Perkinsus marinus ATCC 50983
Paramecium tetraurelia
Tetrahymena elliotti 4EA
Tetrahymena borealis
Plasmodium falciparum Palo Alto/[...]
Plasmodium chabaudi
Toxoplasma gondii ME49
Cryptosporidium parvum Iowa II
Cryptosporidium hominis TU502
Nannochloropsis gaditana CCMP526
Phaeodactylum tricornutum CCAP 1055/[...]
Thalassiosira pseudonana CCMP1335
Schizochytrium aggregatum ATCC 28209
Aurantiochytrium limacinum ATCC MYA-1381
Aplanochytrium kerguelense PBS07
Saprolegnia diclina VS20
Phytophthora infestans T30-4
Phytophthora ramorum
Hyaloperonospora arabidopsidis Emoy2
Hyaloperonospora parasitica
Cyanophora paradoxa
Coccomyxa subellipsoidea C-169
Micromonas sp. RCC299
Ostreococcus 'lucimarinus'
Volvox carteri
Chlamydomonas reinhardtii
Physcomitrella patens
Selaginella moellendorffii
Pinus taeda
Picea sitchensis
Picea abies
Amborella trichopoda
Musa acuminata subsp. malaccensis
Zea mays
Sorghum bicolor
Panicum virgatum
Setaria italica
Leersia perrieri
Oryza sativa Indica Group
Oryza brachyantha
Brachypodium distachyon
Hordeum vulgare subsp. vulgare
Aegilops tauschii
Triticum aestivum
Aquilegia coerulea
Nelumbo nucifera
Beta vulgaris subsp. vulgaris
Helianthus annuus
Erythranthe guttata
Nicotiana benthamiana
Solanum tuberosum
Solanum lycopersicum
Vitis vinifera
Eucalyptus grandis
Citrus clementina
Citrus sinensis
Theobroma cacao
Carica papaya
Brassica rapa subsp. pekinensis
Capsella rubella
Arabidopsis lyrata subsp. lyrata
Arabidopsis thaliana
Linum usitatissimum
Ricinus communis
Jatropha curcas
Manihot esculenta
Lotus japonicus
Medicago truncatula
Phaseolus vulgaris
Glycine max
Citrullus lanatus
Cucumis sativus
Cucumis melo
Morus notabilis
Fragaria vesca
Prunus mume
Malus domestica
Acanthamoeba castellanii str. Neff
Entamoeba histolytica HM-1 IMSS
Polysphondylium pallidum PN500
Dictyostelium discoideum
Thecamonas trahens ATCC 50062
Fonticula alba
Rozella allomycis CSF55
Antonospora locustae HM-2013
Encephalitozoon romaleae SJ-2008
Encephalitozoon cuniculi GB-M1
Piromyces sp. E2
Gonapodya prolifera
Batrachochytrium dendrobatidis JAM81
Allomyces macrogynus ATCC 38327
Catenaria anguillulae PL171
Conidiobolus coronatus NRRL 28638
Mortierella elongata
Lichtheimia hyalospora
Backusella circina FSU 941
Mucor circinelloides B8987
Rhizopus oryzae
Wallemia sebi
Mixia osmundae IAM 14324
Cronartium quercuum f. sp. fusiforme G11
Puccinia triticina 1-1 BBBD Race 1
Malassezia globosa
Pseudozyma antarctica T-34
Ustilago maydis 521
Calocera cornea
Cryptococcus vishniacii
Tremella mesenterica
Cryptococcus neoformans var. grubii H99
Cryptococcus neoformans var. neoformans JEC21
Sphaerobolus stellatus
Piriformospora indica DSM 11827
Punctularia strigosozonata
Sistotrema brinkmannii HHB7604 ss-1
Phlebia brevispora HHB-7030 SS6
Cylindrobasidium torrendii
Phanerochaete carnosa HHB-10118-sp
Dichomitus squalens
Ceriporiopsis subvermispora B
Postia placenta MAD-698-R-SB12
Piloderma croceum F 1598
Volvariella volvacea V23
Hebeloma cylindrosporum h7
Agaricus bisporus var. burnettii JB137-S8
Macrolepiota fuliginosa
Pisolithus microcarpus 441
Coniophora puteana
Serpula lacrymans var. shastensis SHA21-2
Serpula lacrymans var. lacrymans S7.9
Saitoella complicata NRRL Y-17804
Taphrina deformans
Pneumocystis jirovecii
Schizosaccharomyces cryophilus OY26
Schizosaccharomyces pombe 972h-
Lipomyces starkeyi NRRL Y-11557
Candida tenuis
Candida tropicalis MYA-3404
Candida albicans SC5314
Ascoidea rubescens NRRL Y17699
Pachysolen tannophilus NRRL Y-2460
Pichia membranifaciens
Brettanomyces bruxellensis CBS 2499
Candida arabinofermentans NRRL YB-2248
Ogataea angusta NCYC 495 leu1.1
Komagataella pastoris GS115
Cyberlindnera jadinii NRRL Y-1542
Wickerhamomyces anomalus NRRL Y-366-8
Hanseniaspora valbyensis NRRL Y-1626
Ashbya gossypii ATCC 10895
Kluyveromyces marxianus DMKU3-1042
Kluyveromyces lactis
Lachancea thermotolerans CBS 6340
Zygosaccharomyces bailii ISA1307
Vanderwaltozyma polyspora DSM 70294
Candida glabrata CBS 138
Naumovozyma castellii CBS 4309
Kazachstania africana CBS 2517
Saccharomyces arboricola H-6
Saccharomyces kudriavzevii IFO 1802
Saccharomyces cerevisiae M3839
Saccharomyces cerevisiae S288c
Amorphotheca resinae
Hypoxylon sp. EC38
Colletotrichum graminicola M1.001
Acremonium alcalophilum
Trichoderma longibrachiatum ATCC 18648
Fusarium oxysporum f. sp. lycopersici 4287
Fusarium graminearum PH-1
Melanconium sp. NRRL 54901
Magnaporthe oryzae 70-15
Podospora anserina S mat+
Myceliophthora thermophila
Neurospora discreta FGSC 8579
Neurospora crassa OR74A
Neurospora tetrasperma FGSC 2509
Neurospora tetrasperma FGSC 2508
Cladophialophora psammophila CBS 110553
Coniosporium apollinis CBS 100218
Talaromyces stipitatus ATCC 10500
Aspergillus acidus
Penicillium chrysogenum
Penicillium brevicompactum
Blastomyces dermatitidis ATCC 26199
Uncinocarpus reesii 1704
Paracoccidioides sp. 'lutzii' Pb01
Coccidioides posadasii str. Silveira
Coccidioides immitis RS
Myriangium duriaei CBS 260.36
Piedraia hortae var. hortae
Dothistroma septosporum NZE10
Sphaerulina populicola
Aplosporella prunicola CBS 121167
Neofusicoccum parvum UCRNP2
Macrophomina phaseolina MS6
Botryosphaeria dothidea
Lophiostoma macrostomum
Phaeosphaeria nodorum SN15
Bipolaris oryzae ATCC 44560
Pyrenophora teres f. teres 0-1
Alternaria brassicicola

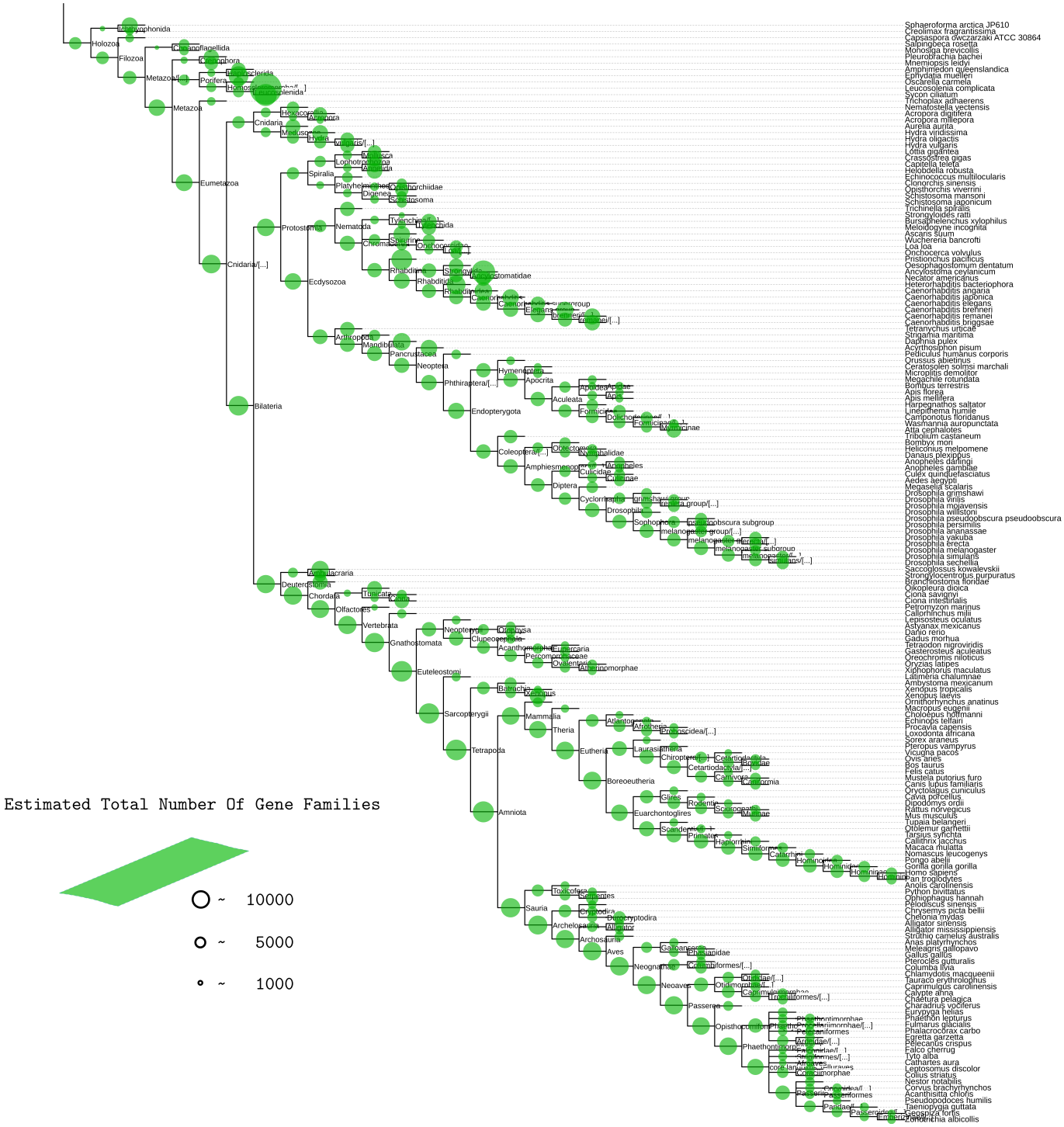


Fig. 3.24 Estimated total number of gene families associated to each ancestral node. Circle sizes are proportional to the number of individual gene families.

therefore, if such a global pattern exists on a bigger scale (when larger number of species are

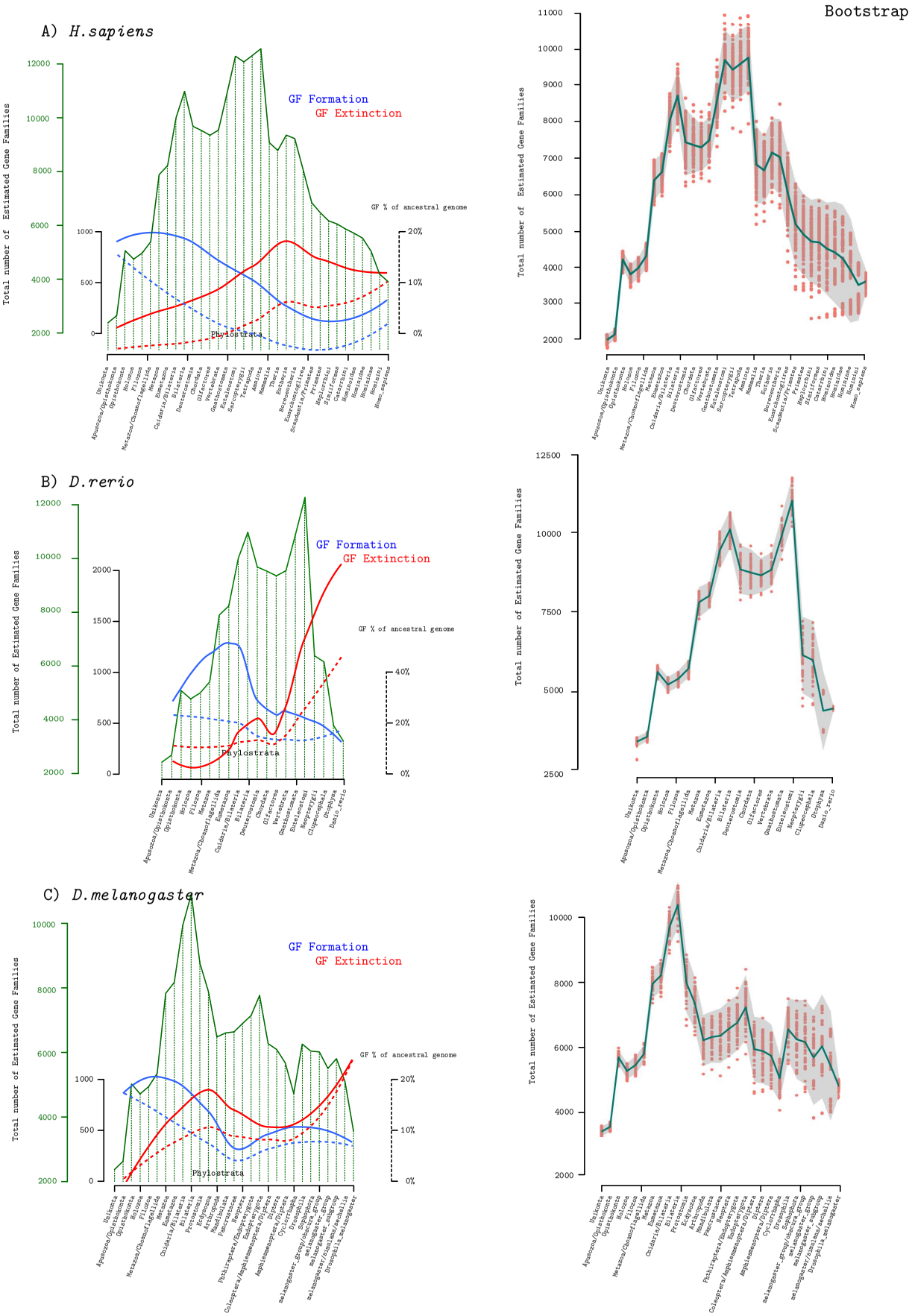
included in the analysis), it is expected to depreciate in case when a fast *QPhyloStrat* gene gain computation strategy is applied.

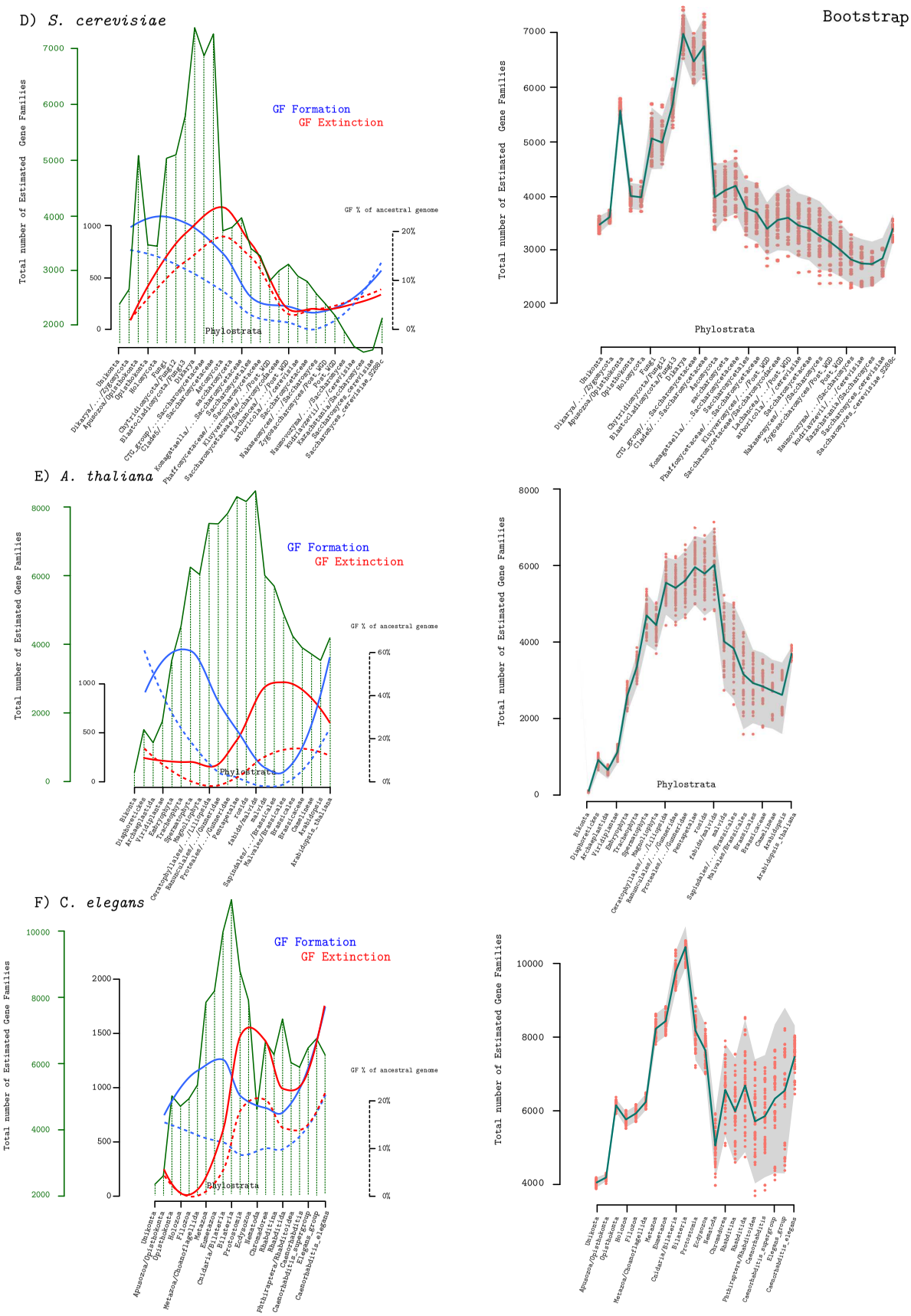
Next, what the obtained result in Figure 3.23 indicates is the presence of a pattern (arching pattern) when the effect of phylogeny is reduced to its minimum. Ergo, establishing the "zero point" distribution shape regarding phylogeny.

Finally, the effect of different initial gene family numbers used in calculation on the resulting distribution shape and its final count is obtained. Clearly, higher initial GFGE values increase the degree of arching, but it is difficult to estimate the true effect without having a more realistic case scenario with more species and their contributions to both GFGEs and GFLEs included in the calculation.

Therefore, using the information about computed GFGEs and GFLEs form all 383 species reconstruction analysis was repeated. The obtained result is summarized in figure 3.24. As in all previous similar depictions, the size of each circle is proportional to number of gene families (GFs). From Figure 3.24 it is clear that GFs associated to Eukaryota phylostrata have not been calculated due to the lack of information associated to both GFGEs and GFLEs at that point (caused by an immense computational challenge related to clustering genes into families).

By extracting six individual lineages from figure 3.24 (*H. sapiens, D. rerio, D. melanogaster, S. cerevisiae, C. elegans* and *A. thaliana*) the previously established "zero point" arching pattern becomes evident (Figure 3.25). All six lineages covering three major clades (animals, plants and fungi) exhibit the same distribution shape. By bootstrapping the computation hundred times (Figure 3.25 right column) robustness and stability of the produced pattern becomes indisputable. Further comparison of the obtained GF distribution and previously established gene family gain and loss patterns (Figure 3.25, left column, blue and red solid lines (dashed lines indicate their relative values)) reveals an already established negative correlation between GFGEs and GFLEs. However, what this comparative analysis shows is the evolutionary time period around which dominance between gene family gain and loss events flips. In animals this "flip" is related to the Cambrian period (or just right about it when the total number of GFs reached its maximum value).


A) H.sapiens
B) D.rerio
C) D.melanogaster
GF Formation
GF Extinction
GF % of ancestral genome
Phylostrata
Total number of Estimated Gene Families
Bootstrap



Description on the next page.

Fig. 3.25 The distribution of the estimated number of gene families at each lineage splitting event (phylostrata) in six different lineages. On the left side, the distribution of the total gene family counts are plotted as a function of phylostrata corresponding to a given phylogeny lineage splitting event. Red and blue lines depict local polynomial regression fitting curves for the total number of GFs (solid lines) and their normalized (normalization is carried out by dividing the number of events associated to each phylostrata by the total size (in number of GFs) of a corresponding ancestral genome) counterparts (dotted lines). Plots on the right side illustrate associated bootstrap (100 iterations) analysis results. Dark red dots represent the obtained, recalculated bootstrap values. The gray band indicates the 95% confidence interval and the green line connects bootstrap distribution means.

Moreover, a direct superposition of the information about the number of gene family gain and loss events across evolutionary time reveals the dominance of gain events in pre-Cambrian (Proterozoic eon) period followed by an increase in loss events across post-Cambrian (Phanerozoic eon) period.

This result has further implications on the evolutionary mode, in accord to which there is no predominant mode of evolution but is divided in two phases: the **progressive phase** (pre-Cambrian) dominated by the accumulation of gene families and the **reductive phase** steering the post-Cambrian evolution through gene family loss. This result is further analysed and discussed in Section 3.2.6.

3.2.5 Patterns and Trends in Genome Complexity Rates

As proposed in Section A.4, the total number gene families conserved at a given evolutionary distance, is a measurable proxy for quantifying genome complexity of an evolving species. The fact that in each of the six investigated lineages, a strong local and global (global being rise in complexity up to their respective maximum levels (Cambrian), and local referring to changes between two adjacent lineage points (phylostrata)), rise and fall in complexity can be observed.

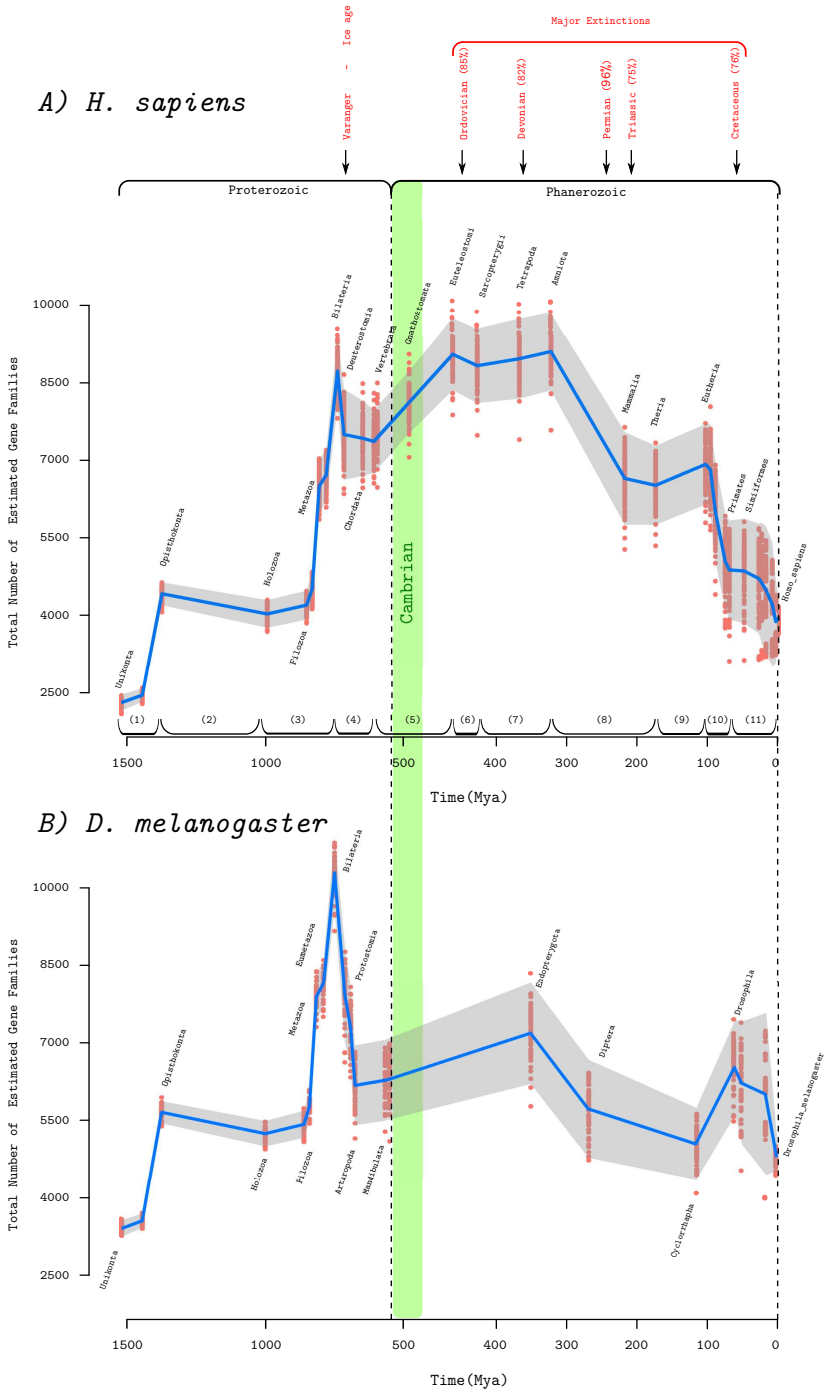


Fig. 3.26 Changes in the number of gene families in lineages leading *H. sapiens* (A) and *D. melanogaster* (B) as a function of geological time. Bootstrap (100 iterations) analysis was used to evaluate the robustness of the obtained family counts at a given time point, with gray bands indicating the 95% confidence interval for the obtained bootstrap values and a blue line connecting its distribution means. Light green bands indicate a period of known major cladogeneses in animals (the Cambrian period), while major extinctions (the big five) are labelled above the plot. Divergence times were estimated according to the TimeTree project (Hedges et al., 2006). The numbers above the x-axis in A, are decomposition landmarks for results depicted in Figures 3.27 and 3.28.

Using the information form the TimeTree project (Hedges et al., 2006), divergence times were mapped to lineage splitting events on a path leading to *H. sapiens*. The obtained result is located in Figure 3.26. What appears to be the rule is a local decrease in complexity prior to Cambrian that was followed by a an increase during it. Strikingly, the complexity change in *D. melanogaster* lineage following the emergence of bilateral ancestor is approximately 4 times higher (reduction in complexity) than the average change associated to periods after the Cambrian era. Though telescoped further in past, this result is in close vicinity to that reported by Lee et al. (2013) obtained through Bayesian (Drummond et al., 2012) and maximum likelihood (Sanderson, 2003) phylogenetic clock methods in analysing arthropods.

Once defined within a given time framework, it is possible to further measure the rate of genome complexity change. To do so, the complexity function (total number of gene families per node (phylostrata)) associated to *H. sapiens* lineage in Figure 3.26(A) has been divided into sections according to complexification (periods characterized by an increase in genome complexity) and simplification (periods characterized by a decrease in genome complexity) periods. Each further modelled by fitting the closest function that most accurately (highest correlation) described the observed complexity values.

The obtained results summarized in Figure 3.27, clearly show that each period can easily be modelled using two types of functions: linear and polynomial (exponential). Since both functions are in the same complexity class, essentially what this implies is a single underlying force causing the complexity changes and their rate through its strength and direction.

Moreover, by dividing periods into pre- and post- Cambrian eras (Figure 3.27 - green dashed line), a change in mode underlying the complexification process (a shift from polynomial (exponential) to its linear form and vice versa) is obvious.

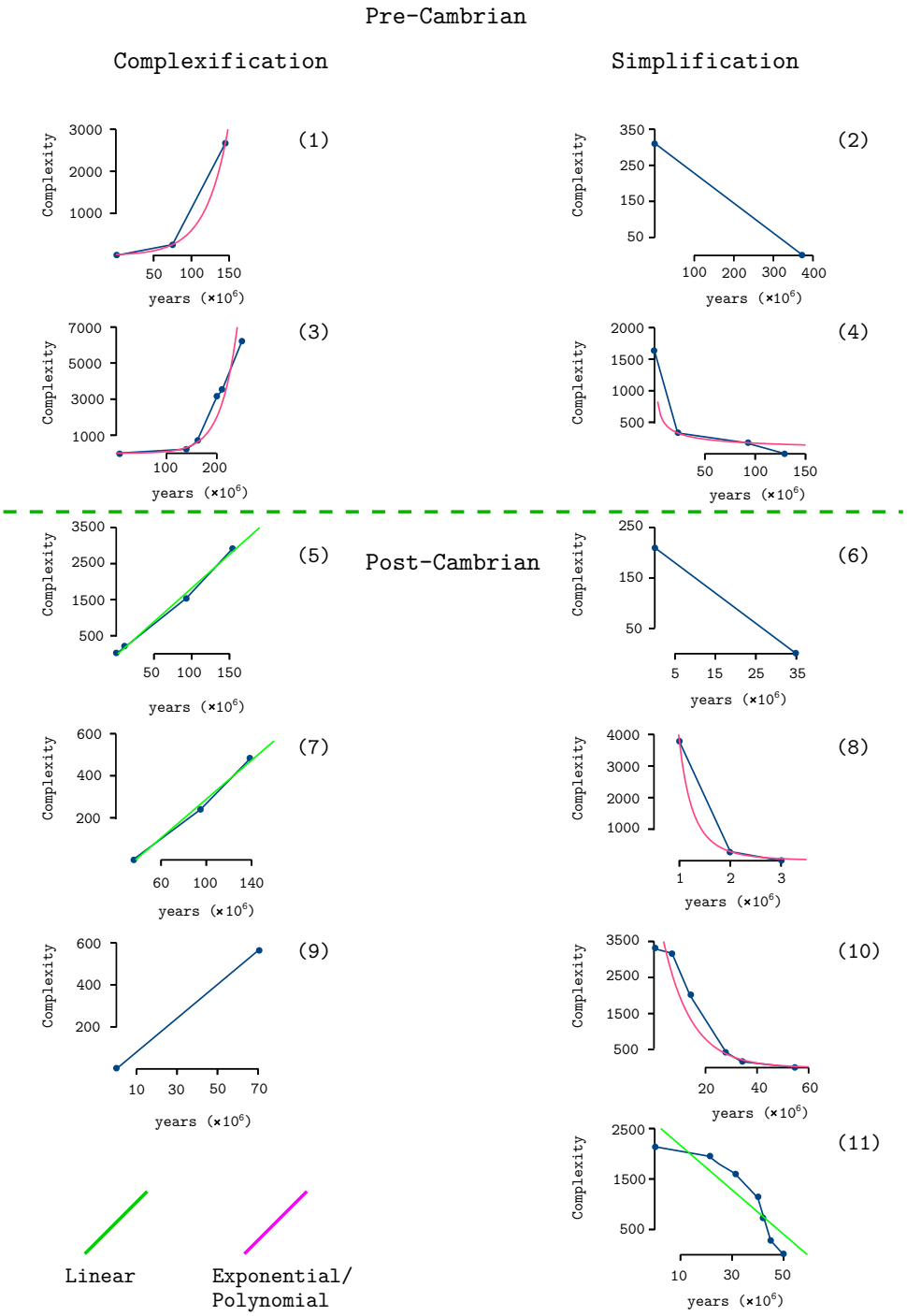


Fig. 3.27 Modelling rates of genome complexity change in *H. sapiens* lineage as a function of time. Each period (complexification and simplification) numerically labelled according to decomposition periods indicated in Figure 3.26 is separately fitted using three different models: linear (green), polynomially (pink) and exponential (pink) (the last two were coloured pink since no significant difference between correlation coefficients could be observed, thus rendering both equally likely to describe the underlying trends). The final model in each case was selected as the one best fitting the observed complexity values (having the highest correlation coefficient). The green dashed line indicates the separation point between pre- and post- Cambrian period. The left column contains plots associated to complexification periods while the right column contains plots associated to simplification periods.

To illustrate this observation, in Figure 3.28 average complexification/simplification rates associated to each time period analysed in Figure 3.27 are plotted as functions of geological time, connecting complexification (blue line) and simplification (red line) rates. The obtained result confirms previously made observations by separately analysing gene family gain and loss events in the previous section. However, this time a cumulative effect (resulting net change when both GFGEs and GFLEs are combined) with respect to geological time can be seen. The rate of complexification in Proterozoic eon dominates the evolution, while Phanerozoic is characterized by high simplification rates. Therefore, showing the evolutionary rates (as functions of the average genome complexity change) to be also bipartite with strong simplifications associated with the increasing lineage splitting, reaching its maximum around 100 Mya.

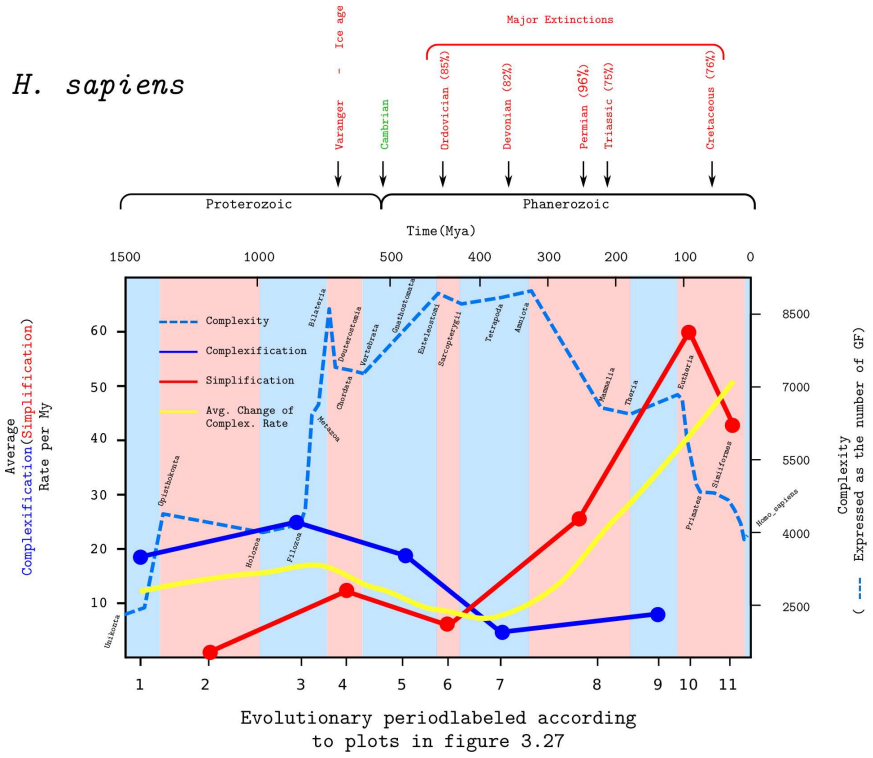


Fig. 3.28 Duration of complexification/simplification periods and their average rates in *H. sapiens* lineage. Average complexification/simplification rates plotted as a function of time. Moreover each pink period corresponds to a period dominated by simplification while each light blue period reflects the increase in average rate of complexity. The dashed line indicated the total number of families associated to each estimated time point. The yellow line estimates the average rate of complexity change (local polynomial regression fitting curve).

Estimating the average rate of complexity change (linear polynomial regression analysis - yellow line) shows that throughout the better part of evolutionary history, the change does not exceed 20 gene families (gained or lost) per million years, except during the last 100 million years (caused by 4-5 times higher simplification rates). Under the assumption that change in genome complexity as a function of time reflects a measurable proxy for calculating evolutionary rates, it follows that the rate of evolution has significantly increased in the last 100 million years or so.

3.2.6 Final Synthesis

With this new-found ability to observe changes in the number of gene families throughout evolutionary history of species, many new regularities, patterns and discoveries have emerged. Not only that these discoveries are fuelling new hypothesis regarding general patterns and trends of species evolution on a grand scale, but they are in accord with those previously proposed ones, e.g. Gould (1997); Lynch and Conery (2003); Wolf and Koonin (2013). Unlike many classic views related to evolution of biological complexity (discusses in Gould (1997) and Koonin (2011)), the evidence presented here demonstrate a global bell-shaped pattern of complexity across two major evolutionary epochs. In light of these results, here I propose a plausible explanation (a mechanistic sketch utilising various established models) for the underlying pattern.

During the Proterozoic eon, genomic evolution was clearly typified by punctuated (temporally brief) complexification periods engendering large genomic perturbations which intersect long, protracted in time, low rate simplification periods. When placed within a broader context (both evolutionary and ecological) the observation is not a surprising one. Given our current understanding of the time period, the better part of Proterozoic was dominated by mostly microscopic, metabolically divers generalists, living in a numerically versatile but uniform ecological niches. Uniformity being emphasized by their microscopic size. Therefore, such ecological generalists dependent on abiotic factors for their survival had only one way to evolve, increasing their genomic complexity by improving the vertical information transfer from their predecessors to their descendants. Better adopted individuals (probably

caused by fixating more mutational changes) that were able to more efficiently extract nutrients from their surroundings grew in number, eventually reaching the (carrying) capacity of the underlying niche (Hui, 2006). Once reached, the newly formed, better adapted species found themselves in a direct competition with their "ancestral" form thus engaging the evolutionary race. One of the well established evolutionary race concepts known as the "arms race" (*The Red Queen hypothesis (RQH)*) may be applied as an illustration for a mechanistic principle underlying the process. The red queen hypothesis was first proposed by Van Valen (1973) as an explanation for "The Law of Extinction", showing that, in many populations, the probability of extinction does not depend on the lifetime of the population, but is constant over millions of years. Essentially, what the hypothesis boils down to is a continuous evolution toward increasing genomic information in order to adapt and eventually reach some sort of an uneasy balance between "protagonists". However, in the event a balance cannot be established one of the competitors starts loosing this power struggle ultimately leading its population to extinction. In reality this is usually not a sudden event and spans over many generations, each time reducing the effective population size of the loosing side. In case the loosing side is the ancestral form mentioned above, reduction in size makes the shrinking population susceptible (by reducing the efficiency of natural selection) to accumulation of mildly deleterious insertions and gene duplication through drift (Lynch, 2006, 2007; Lynch and Conery, 2003). These additions in return as an effect increase the genome size, alleviating the emergence of new genes (potential gene family founders) (McLysaght and Guerzoni, 2015; Neme and Tautz, 2014; Tautz and Domazet-Lošo, 2011) that may provide the population with necessary advantage to get back into the race. Although the illustration here utilizes the RQH as a mechanism through which reduction in population size can be achieved, by no means the reduction is strictly limited to such a scenario. Therefore, the above required reduction can be achieved through other means (e.g. environmental factors) which may lead to the same outcome.

While the "ancestral" population is shrinking, descendant population by increasing its effective size, uplifts the power of purifying selection (Lynch and Conery, 2003). It has been reported on many occasions that highly successful forms like cyanobacterium *Prochlorococ-*

cus sp. (Dufresne et al., 2005; Partensky and Garczarek, 2010) and alpha-proteobacterium Candidatus *Pelagibacter ubique* (Giovannoni et al., 2005b; Morris et al., 2012), undergo genome streamlining, thus demonstrating the loss of genomic information (complexity) within constraints imposed by gene-specific purifying selection, is a reality. Moreover, though majority of loss might be neutral, there are strong recent indications that the process is adaptive. Hypothesis underlying this conjecture is termed "*The Black Queen hypothesis (BQH)*" (Morris, 2015; Morris et al., 2012). According to this hypothesis, which addresses the so-called "compensated trait loss" (Ellers et al., 2012) by natural selection, protagonists actively compete with each other in loosing genomic material (thus reducing the overall genomic complexity). The cause of this competition lies in their interactions through the environment (shaped by other communities). Competing sides partition benefits between each other through "leakiness", that is, the apparent altruistic behaviour in which one side adapts more quickly to nutrient assimilation from the environment (by eliminating genes involved in its production), wins. However, such victory does not imply the loss of a phenotype, since loss has been compensated through the environment, thus the only consequence for winners of this competition is increase in fitness (as a result of not wasting energy for nutrient self-production). Evidently such competition increases diversification and subsequent specialisation, which in return increases the number of existing species. Therefore, two major hallmarks of the black queen hypothesis in action would be, the increase in the number of species and reduced genomic complexity. Though currently only analysis conducted on microbial communities confirm fitness benefits associated to winners of such race (Cooper et al., 2001; Lee and Marx, 2012), there is no reason why scaling up the principle to include all levels of interactions between unicellular and multicellular eukaryotes, wouldn't hold (especially in well documented interactions between multicellular organisms and their resident microbiota (Ley et al., 2008; Rosenberg et al., 2007)). Therefore, evolution dominated by the principles of "*The Black Queen hypothesis*", over long periods of geological time is expected to result in long term reduction in genome complexity, accompanied by high speciation rates. Large number of species in a given ecological niche with a limited capacity will systematically reduce their individual effective population size, making it easier for a

population to again be affected by the drift, thus repeating the cycle. An interesting observation that follows from the above mechanistic sketch is that on a global scale, the increasing the number of species in a given environment as a result ought to exhibit the overall increase in genomic complexity accompanied by smaller effective size populations. Furthermore, as the environment (reachable niches) gets more populated, the above cycle is expected to repeat more frequently resulting with a potentially exponential rise in genome information (complexity). As a consequence, simplification periods between two complexifications for a given evolutionary path ought to exhibit shorter and shorter time-spans (ultimately eliminating them). Note how this scenario is almost the exact description of genomic complexity patterns associated to Proterozoic eon (Figures 3.26 and 3.28).

Increasing genomic information (complexity) increases the adaptedness of single cell organisms to various environments. Combined together with overpopulation of occupied niches, this scenario may eventually lead to a cooperation and ultimately the emergence of multicellular forms. Such organisms present a key transition event that shifts species form exploiting micro to macro environments (Bonner, 2004; Grosberg and Strathmann, 2007; King, 2004), thus opening new opportunities and consequently increasing the carrying capacity of the reachable niches to sustain a larger populations. Under this scenario, based on the model proposed by M. Lynch (Lynch and Conery, 2003), large populations are expected to decrease its genomic complexity. However, comparing the expectations with the obtained results (Figure 3.26), it appears that the expected decrease (at the dawn of multicellularity in animal clade - Metazoa) has been shifted toward the emergence of Bilateria, lasting until the beginning of Phanerozoic. A plausible explanation for the existing shift would be the apparent waiting period in which newly formed organisms were adapting, that is learning to utilize this new environment. This is far from an unlikely scenario since the initial multicellular forms are not expected to already at that point have a fully functional digestive track, hence due to nutritional reliance on their micro environment restricted in their effective population size. However, once adopted to the new environment, an overall decrease in genomic complexity is expected, al least until the environment's maximum load is reached again, or another key transition takes place (note that the above model does not exclude periodic com-

plexifications during that time since the competition is not only restricted to RQH effect). If one reflects back to Figure 3.25 the sketched expectation becomes the exact hallmark of Phanerozoic eon. What is even more interesting is the observation that throughout the entire eon, the capacity of the existing environment to sustain large populations apparently has not been reached (due to the prolonged trend of universal genome simplification). An alternative cause for this simplification trend might lay in the emergence of a new key transition, such as the appearance of cooperation between multicellular organisms (Bernard et al., 2016). This type of event would definitely represent an organizational shift, opening access to more niches and thus further increasing the carrying capacity of the reachable environment.

While it is evident that the full breadth of genomic evolution cannot be described using a simple "formula" and that there are many exception (e.g. key transition events associated to loss of gene families (Figure 3.21); which is not in contradiction to BHQ (Morris, 2015)), the above mechanistic sketch based on (but not limited to) two different types of "power struggles" (Morris, 2015; Van Valen, 1973), placed within an environmental context (Hui, 2006) and unified under one roof (the Lynch and Conery model (Lynch and Conery, 2003)), presents a plausible description of the obtained macroevolutionary pattern. Although the model proposed by Lynch and Conery came under a lot of criticism over the years (Daubin and Moran, 2004; Whitney and Garland, 2010; Yi, 2006) due to its heavy reliance on stochasticity as primary factor underlying the rapid change of genome complexity, the results reported here support the model quite well. However, the general view of evolution as an adaptive process has not been challenged here. Given that simplification periods dominate the evolution of genomic complexity, which in return are seen as predominantly governed by adaptations, the evolution according to here presented results can be seen as a process of constant thrive of species to "reconcile" with their environment by adapting to it. Needless to say how all observations stated so-far, based on the evidence presented in previous sections, also hold for both plants and fungi. Each clade clearly exhibits the rise and the fall of genome complexity.

However, in order to further test the plausibility of the above sketched, a first step would be to design an according simulation experiment to see whether the patterns could be recre-

ated or not. On a more general plan the study presented here requires further development of more precise mapping strategies for both gain and loss events to specific evolutionary periods and a rigorous theory underlying the entire hypothesis.

3.2.7 Future Directions

With the ever increasing amount of sequence data, particularly the number of species with sequenced genomes, the necessity to identify homologous relationships based on similarity has become central to bioinformatics and computational biology. Homology is one of the fundamental concepts in biology important for inferring common ancestry and reconstruction of gene (family) content of extinct species. To do so and to be able to compute a plausible sequence of historical events, rates of gain and loss at the level of genes and gene families need to be estimated.

Here a set of effective and efficient methods for gene family content reconstruction have been presented, allowing the computational tasks to include a large body of data in order to get high quality gain/loss mappings across a wide range of current and ancestral species. The utility of this approach was demonstrated by investigating global patterns and trends in genomic complexities across 383 lineages, revealing a completely new pattern of evolutionary change.

However, the volume of data and high diversity in the sequence information presented a tremendous computational challenge forcing many new heuristic solutions to be applied. These solutions, together with innate discrepancies and possible irregularities associated with the processed data, most certainly affected the final outcome. Therefore, in the next section I briefly reflect on potential future direction that can increase the utility and quality of computed results.

3.2.7.1 The Effect of Horizontal Gene Transfer

Lateral or horizontal gene transfer refers to non-genealogical transfer of genetic material between organisms. Probably the highest transfer ever to occur, was the one at the begin-

ning of eukaryote lineage during the endosymbiosis and subsequent genetic integrations of two prokaryotes that give rise to organelles such as mitochondria and plastids (Keeling and Palmer, 2008). This process set the foundation for complex life forms to emerge.

It has been suggested that HGT between higher eukaryotes might have little or no effect on their evolution (Keeling and Palmer, 2008). Furthermore, the recorded number of HGT events implies transfers are more common between recent lineages then between distant ones (Choi and Kim, 2007; Ge et al., 2005). The cause for this observation may be tribute to a recent increase in susceptibility to HGT, however, a more likely explanation is the inability to detect the ancient transfers.

A frequent question addressing the importance of horizontal gene transfer in evolution of eukaryotes is related to the quantity of genes acquired by HGT. Studies show that in modern prokaryotes, even a single HGT consisting of one or a few genes can increase the adaptability of the recipient organism to exploit new ecological niches by acquiring a new function in the process (Vogan and Higgs, 2011). A first step in identifying HGT in prokaryotes is to locate new genes (Ochman, 2001; Ragan, 2001) either through detecting bias in codon usage and different base composition in relation to other genes in the genome, or by detecting phylogenetic incongruence. Criteria based on codon usage bias and differential base composition strongly depends on the age of the transfer. Transferred genes can be ameliorated after only a few generations (Marri and Golding, 2008) due to the strong mutational bias. Phylogenetic incongruence, on the other hand can utilize this problem to a certain extent, but it's highly dependent on gene loss which is hard to distinguish from horizontal gene transfer under the commonly used "one gain multiple loss" scenario (Ragan, 2001).

Therefore, future research should be directed toward devising new strategies for distinguishing HGT and gene loss events. A straightforward strategy would be a type of likelihood measure based on the presence/absence of genes across different phylogenetic lineages (Figure 3.29) where the distance between gain and loss events (in terms of the number of taxonomic groups separating the two) may be used as a measure. Once HGT is identified across the entire eukarya domain it ought to be possible to estimate the rate of transfer events across eukaryote lineages and try to answer some basic questions regarding its propensity,

prevalence and importance in the evolution of eukaryote species.

A particularly important effect of HGT on general gain and loss computation is expected within microbial eukaryotes and fungi (Keeling and Palmer, 2008). Therefore, any analysis including these two groups of organisms should not be considered disregarding the lateral transfer of genetic material.

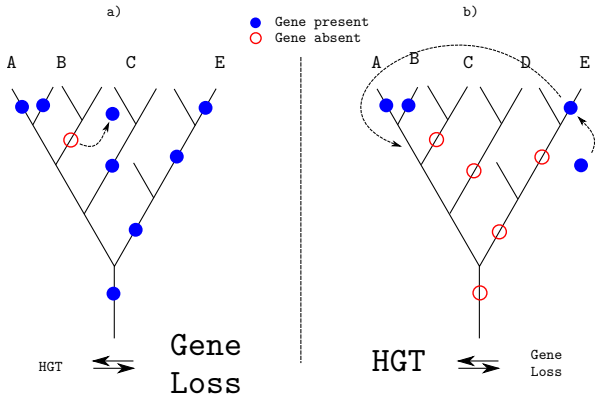


Fig. 3.29 Distinguishing between HGT and gene loss. Blue (filled) circles mark gene presence and red (empty) circles gene absence. Approach: Given a presence/absence scenario in species specific states, a) if ancestral states on path from A to E have more detected homologs (blue circles), then "gene loss" in clade C after A-C split is an reasonable explanation. On the other hand, b) if the number of detected homologs is small (red circles) and no evidence exists to suggest that a common ancestor of A and E had a gene present, then HGT event is a more likely explanation for the observed distribution

HGT has a strong impact on orthology computation. Recent studies (Dalquen et al., 2013; Sonnhammer et al., 2014) indicate that current orthology computation strategies preform badly in the presence of HGT events, due to their high mutual similarity and distant phylogenetic relationship. Therefore, the problem presents a future research direction requiring a novel solutions.

References

Abouelhoda, M. I., Kurtz, S., and Ohlebusch, E. (2002). The enhanced suffix array and its applications to genome analysis. *Algorithms in Bioinformatics. Lecture Notes in Computer Science.*, 2452:449–463.

Adami, C. (2002). What is complexity? *Bioessays.*, 24:1085–94.

Adami, C. and Cerf, N. (2000). Physical complexity of symbolic sequences. *Physica*, 137:62–69.

Adamowicz, S. J., Purvis, A., and Wills, M. A. (2008). Increasing morphological complexity in multiple parallel lineages of the crustacea. *Proceedings of the National Academy of Sciences*, 105:4786–4791.

Adl, S., Simpson, A., Farmer, M., Andersen, R., Anderson, O., Barta, J., Bowser, S., Brugerolle, G., Fensome, R., Fredericq, S., James, T., Karpov, S., Kugrens, P., Krug, J., Lane, C., Lewis, L., Lodge, J., Lynn, D., Mann, D., McCourt, R., Mendoza, L., Moestrup, O., Mozley-Standridge, S., Nerad, T., Shearer, C., Smirnov, A., Spiegel, F., and Taylor, M. (2005). The new higher level classification of eukaryotes with emphasis on the taxonomy of protists. *J. Eukaryot. Microbiol.*, 52:399–451.

Adl, S., Simpson, A., Lane, C., Lukes, J., Bass, D., Bowser, S., Brown, M., Burki, F., Dunthorn, M., Hampl, V., and et al. (2012). The revised classification of eukaryotes. *J. Eukaryot. Microbiol.*, 59:429–514.

Ahn, M., Cui, J., Irving, A., and Wang, L. (2016). Unique loss of the pyhin gene family in bats amongst mammals: Implications for inflammasome sensing. *Scientific Reports*, 6:21722.

Akerborg, O., Sennblad, B., Arvestad, L., and Lagergren, J. (2009). Simultaneous bayesian gene tree reconstruction and reconciliation analysis. *Proc Natl Acad Sci U S A*, 106:5714–9.

Albá, M. and Castresana, J. (2007). On homology searches by protein blast and the characterization of the age of genes. *BMC Evolutionary Biology*, 7:53.

Albalat, R. and Canestro, C. (2016). Evolution by gene loss. *Nat Rev Genet.*, 1:1.

Albert, V., Barbazuk, W., dePamphilis, C., Der, J., Leebens-Mack, J., Ma, H., Palmer, J., Rounsley, S., Sankoff, D., Schuster, S., and et al. (2013). The amborella genome and the evolution of flowering plants. *Science*, 342:1241089.

Altschul, S., Gish, W., Miller, W., Myers, E., and Lipman, D. (1990). Basic local alignment search tool. *J Mol Biol*, 215:403–410.

Altschul, S., Madden, T., Schäffer, A., Zhang, J., Zhang, Z., Miller, W., and Lipman, D. (1997). Gapped blast and psi-blast: a new generation of protein database search programs. *Nucleic Acids Res.*, 25:3389–3402.

Avisa, D. and Imamura, T. (2007). A list heuristic for vertex cover. *Operations Research Letters*, 35:201–204.

Baeza-Yates, R. and Ribeiro-Neto, B. (1999). *Modern information retrieval*. Addison-Wesley Longman.

Bellman, R. (1957). *Dynamic Programming*. Princeton University Press. Dover Publications; Reprint edition (2003).

Benjamini, Y. (2010). Discovering the false discovery rate. *Journal of the Royal Statistical Society*, 72:405–416.

Benjamini, Y. and Hochberg, Y. (1995). Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society*, 57:289–300.

Benson, D., Karsch-Mizrachi, I., Lipman, D., Ostell, J., and Wheeler, D. (2007). Genbank. *Nucleic Acids Res*, 35:D21–5.

Bernard, A., André, J., and Bredeche, N. (2016). To cooperate or not to cooperate: Why behavioural mechanisms matter. *PLoS Comput Biol.*, 12:e1004886.

Blomme, T., Vandepoele, K., De Bodt, S., Simillion, C., Maere, S., and Van de Peer, Y. (2006). The gain and loss of genes during 600 million years of vertebrate evolution. *Genome Biol.*, 7:R43.

Bonner, J. (2004). Perspective: the size-complexity rule. *Evolution*, 58:1883,90.

Bonner, J. T. (1988). *The evolution of complexity by means of natural selection*. Princeton University Press; First Edition edition.

Boussau, B., Karlberg, E., Frank, A., Legault, B., and Andersson, S. (2004). Computational inference of scenarios for alpha-proteobacterial genome evolution. *Proc Natl Acad Sci.*, 101:9722–9727.

Bradley, B. (2008). Reconstructing phylogenies and phenotypes: A molecular view of human evolution. *Journal of Anatomy*, 212:337–353.

Brigandt, I. (2002). Homology and the origin of correspondence. *Biology and Philosophy*, 17:389–407.

Brigandt, I. (2003). Homology in comparative, molecular, and evolutionary developmental biology: the radiation of a concept. *J Exp Zool B Mol Dev Evol.*, 299:9–17.

Bromham, L. (2016). *An Introduction to Molecular Evolution and Phylogenetics*. Oxford University Press.

Bromham, L. and Penny, D. (2003). The modern molecular clock. *Nature Reviews Genetics*, 4:216–224.

Brown, D. and Sugimoto, K. (1974). The structure and evolution of ribosomal and 5s dnas in xenopus laevis and xenopus mulleri. *Cold Spring Harbor Symp Quant Biol.*, 38:501–505.

Brown, M., Kolisko, M., Silberman, J., and Roger, A. (2012). Aggregative multicellularity evolved independently in the eukaryotic supergroup rhizaria. *Current Biology*, 22:1123–1127.

Brown, M., Spiegel, F., and Silberman, J. (2009). Phylogeny of the "forgotten" cellular slime mold, fonticula alba, reveals a key evolutionary branch within opisthokonta. *Mol Biol Evol*, 26:2699–2709.

Burki, F. (2014). The eukaryotic tree of life from a global phylogenomic perspective. *Cold Spring Harbor Perspectives in Biology*, 6:a016147.

Burki, F., Okamoto, N., Pombert, J.-F., and Keeling, P. (2012). The evolutionary history of haptophytes and cryptophytes: phylogenomic evidence for separate origins. *Proceedings of the Royal Society of London B: Biological Sciences*, 279:2246–2254.

Butler, A. and Saidel, W. (2000). Defining sameness: Historical, biological, and generative homology. *BioEssays*, 22:846–853.

Cai, J., Zhao, R., Jiang, H., and Wang, W. (2008). De novo origination of a new protein-coding gene in saccharomyces cerevisiae. *Genetics*, 179:487–496.

Capella-Gutiérrez, S., Marcet-Houben, M., and Gabaldón, T. (2012). Phylogenomics supports microsporidia as the earliest diverging clade of sequenced fungi. *BMC Biology*, 10:47.

Capra, J., Stolzer, M., Durand, D., and Pollard, K. (2013). How old is my gene? *Trends Genet.*, 29:659–68.

Capra, J., Williams, A., and Pollard, K. (2012). Proteinhistorian: tools for the comparative analysis of eukaryote protein origin. *PLoS Comput Biol*, 8:e1002567.

Carvunis, A., Rolland, T., Wapinski, I., Calderwood, M., Yildirim, M., Simonis, N., Charloteaux, B., Hidalgo, C., Barbette, J., Santhanam, B., Brar, G., Weissman, J., Regev, A., Thierry-Mieg, N., Cusick, M., and Vidal, M. (2012). Proto-genes and de novo gene birth. *Nature.*, 487:370–4.

Cavalier-Smith, T. (2009). Megaphylogeny, cell body plans, adaptive zones: Causes and timing of eukaryote basal radiations. *J. Eukaryot. Microbiol*, 56:26 – 33.

Cavalier-Smith, T. and Chao, E. E. (2010). Phylogeny and evolution of apusomonadida (protozoa: Apusozoa): new genera and species. *Protist*, 161:549–576.

Chase, M. and Reveal, J. (2009). A phylogenetic classification of the land plants to accompany apg iii. *Botanical Journal of the Linnean Society*, 161:122–127.

Chen, S., Zhang, Y., and Long, M. (2010). New genes in drosophila quickly become essential. *Science*, 330:1682–1685.

Choi, I. and Kim, S. (2007). Global extent of horizontal gene transfer. *Proc. Natl Acad. Sci.*, 104:489–4494.

Chou, H., Hayakawa, T., Diaz, S., Krings, M., Indriati, E., Leakey, M., Paabo, S., Satta, Y., Takahata, N., and Varki, A. (2002). Inactivation of cmp-n-acetylneuraminic acid hydroxylase occurred prior to brain expansion during human evolution. *Proc. Natl Acad. Sci. USA*, 99:11736–11741.

Clark, A., Eisen, M., Smith, D., Bergman, C., Oliver, B., Markow, T., Kaufman, T., Kellis, M., Gelbart, W., Iyer, V., and et al. (2007). Evolution of genes and genomes on the drosophila phylogeny. *Nature*, 450:203–218.

Cooper, V., Schneider, D., Blot, M., and Lenski, R. (2001). Mechanisms causing rapid and parallel losses of ribose catabolism in evolving populations of escherichia coli b. *J Bacteriol.*, 9:2834–41.

Cormen, T., Stein, C., Rivest, R., and Leiserson, C. (2001). *Introduction to Algorithms*. 2nd edn. McGraw-Hill Higher Education.

Csuros, M. (2010). Count: evolutionary analysis of phylogenetic profiles with parsimony and likelihood. *Bioinformatics.*, 26:1910–1912.

Dalquen, D., Altenhoff, A., Gonnet, G., and Dessimoz, C. (2013). The impact of gene duplication, insertion, deletion, lateral gene transfer and sequencing error on orthology inference: a simulation study. *PLoS One*, 8:e56925.

Daubin, V. and Moran, N. (2004). Comment on "the origins of genome complexity". *Science*, 306:978.

Daugherty, L., Seal, R., Wright, M., and Bruford, E. (2012). Gene family matters: expanding the hgnc resource. *Hum Genomics.*, 6:4.

Davies, W. (2007). Adaptive gene loss in vertebrates: Fhotosensitivity as a model case. *ELS. John Wiley and Sons, Ltd: Chichester*.

Davies, W., Cowing, J., Carvalho, L., Potter, I., Trezise, A., Hunt, D., and Collin, S. (2007). Functional characterization, tuning, and regulation of visual pigment gene expression in an anadromous lamprey. *FASEB Journal*, 21:2713–2724.

Dayhoff, M. (1976). Origin and evolution of protein superfamilies. *Fed Proc*, 35:2132–2138.

De Beer, G. (1971). *Homology: An Unsolved Problem.* Oxford University Pres, Glasgow.

De Bie, T., Cristianini, N., Demuth, J., and Hahn, M. (2006). Cafe: a computational tool for the study of gene family evolution. *Bioinformatics.*, 22:1269–71.

Delsuc, F., Brinkmann, H., Chourrout, D., and Philippe, H. (2006). Tunicates and not cephalochordates are the closest living relatives of vertebrates. *Nature*, 439:965–968.

Delsuc, F., Tsagkogeorga, G., Lartillot, N., and H., P. (2008). Additional molecular support for the new chordate phylogeny. *Genesis*, 46:592–604.

Demuth, J., De Bie, T., Stajich, J., Cristianini, N., and Hahn, M. (2006). The evolution of mammalian gene families. *PLoS ONE*, 1:e85.

Dietzl, G., Chen, D., Schnorrer, F., Su, K., Barinova, Y., Fellner, M., Gasser, B., Kinsey, K., Oppel, S., Scheiblauer, S., Couto, A., Marra, V., Keleman, K., and Dickson, B. (2007). A genome-wide transgenic rnai library for conditional gene inactivation in drosophila. *Nature*, 448:151–156.

Domazet-Lošo, T., Brajković, J., and Tautz, D. (2007). A phylostratigraphy approach to uncover the genomic history of major adaptations in metazoan lineages. *Trends Genet*, 23:533–9.

Domazet-Lošo, T. and Tautz, D. (2003). An evolutionary analysis of orphan genes in drosophila. *Genome Research*, 13:2213–9.

Donoghue, M. (1992). *Homology*. in E.F. Keller and E.A. Lloyd (eds.), Keywords in Evolutionary Biology , Harvard University Press, Cambridge, MA pp. 170-179.

Doolittle, W. F. (2012). A ratchet for protein complexity. *Nature*, 481:270–271.

Doyon, J., Ranwez, V., and Daubin, V. Berry, V. (2011). Models, algorithms and programs for phylogeny reconciliation. *Brief. Bioinform.*, 12:392–400.

Drummond, A., Suchard, M., Xie, D., and Rambaut, A. (2012). Bayesian phylogenetics with beauti and the beast 1.7. *Mol. Biol. Evol.*, 29:1969–1973.

Dufresne, A., Garczarek, L., and Partensky, F. (2005). Accelerated evolution associated with genome reduction in a free-living prokaryote. *Genome Biol.*, 6:R14.

Dujon, B. (1996). The yeast genome project: what did we learn? *Trends in Genetics*, 12:263–270.

Dunn, C., Giribet, G., Edgecombe, G., and Hejnol, A. (2014). Animal phylogeny and its evolutionary implications. *Annual Review of Ecology, Evolution, and Systematics*, 45:371–395.

Dunn, O. (1961). Multiple comparisons among means. *Journal of the American Statistical Association*, 56:52–64.

Dupont, P. (1977). Laplace and the indifference principle in the 'essai philosophique des probabilités'. *Rend. Sem. Mat. Univ. Politec. Torino*, 36:125–137.

D'Souza, G., Waschina, S., Pande, S., Bohl, K., Kaleta, C., , and Kost, C. (2014). Less is more: Selective advantages can explain the prevalent loss of biosynthetic genes in bacteria. *Evolution*, 68:2559–2570.

Ebersberger, I., Simoes, R. d. M., Kupczok, A., Gube, M., Kothe, E., Voigt, K., and Haeseler, A. v. (2012). A consistent phylogenetic backbone for the fungi. *Mol Biol Evol*, 29:1319–1334.

Edgar, R. (2010). Search and clustering orders of magnitude faster than blast. *Bioinformatics*, 26:2460–1.

Edgecombe, G., Giribet, G., Dunn, C., Hejnol, A., Kristensen, R., Neves, R., Rouse, G. W., Worsaae, K., and Sorensen, M. (2011). Higher-level metazoan relationships: recent progress and remaining questions. *Org Divers Evol*, 11:151–172.

Efron, B. and Tibshirani, R. (1993). *An Introduction to the Bootstrap.* Chapman and Hall/CRC; Softcover reprint of the original 1st ed.

Ellers, J., Kiers, E., Currie, C., McDonald, B., and Visser, B. (2012). Ecological interactions drive evolutionary loss of traits. *Ecol Lett.*, 15:1071–1082.

Farris, J. (1977). Phylogenetic analysis under dollo's law. *Syst. Zool*, 26:77–88.

Fawcett, T. (2006). An introduction to roc analysis. *Pattern Recognition Letters*, 27:861–874.

Fiore-Donno, A., Nikolaev, S., Nelson, M., Pawlowski, J., Cavalier-Smith, T., and Baldauf, S. (2010). Deep phylogeny and evolution of slime moulds (mycetozoa). *Protist*, 161:55–70.

Fischer, D. and Eisenberg, D. (1999). Finding families for genomic orfans. *Bioinformatics*, 15:759–762.

Fischer, J. and Heun, V. (2007). A new succinct representation of rmq-information and improvements in the enhanced suffix array. *Proceedings of the International Symposium on Combinatorics, Algorithms, Probabilistic and Experimental Methodologies*, page 459–470.

Fisher, R. A. (1922). On the interpretation of chi-squared from contingency tables, and the calculation of p. *Journal of the Royal Statistical Society*, 85:87–94.

Fitch, W. (1970). Distinguishing homologous from analogous proteins. *Syst Biol*, 19:99–113.

Fitch, W. and Ayala, F. (1995). *Tempo and Mode in Evolution: Genetics and Paleontology 50 Years After Simpson.* National Academies Press; First edition (January 26, 1995).

Flicek, P., Amode, M., Barrell, D., Beal, K., Billis, K., Brent, S., Carvalho-Silva, D., Clapham, P., Coates, G., Fitzgerald, S., Gil, L., Girón, C., Gordon, L., Hourlier, T., Hunt, S., Johnson, N., Juettemann, T., K ah ari, A., Keenan, S., Kulesha, E., Martin, F., Maurel, T., McLaren, W., Murphy, D., Nag, R., Overduin, B., Pignatelli, M., Pritchard, B., Pritchard, E., Riat, H., Ruffier, M., Sheppard, D., Taylor, K., Thormann, A., Trevanion, S., Vullo, A., Wilder, S. P., Wilson, M., Zadissa, A., Aken, B., Birney, E., Cunningham, F., Harrow, J., Herrero, J., Hubbard, T., Kinsella, R., Muffato, M., Parker, A., Spudich, G., Yates, A., Zerbino, D., and Searle, S. (2014). Ensembl. *Nucleic Acids Research*, 42:D749–D755.

Foret, S., Knack, B., Houliston, E., Momose, T., Manuel, M., Queinnec, E., Hayward, D., Ball, E., and Miller, D. (2010). New tricks with old genes: the genetic bases of novel cnidarian traits. *Trends Genet.*, 26:154–158.

Gabaldón, T. and Koonin, E. (2013). Functional and evolutionary implications of gene orthology. *Nat Rev Genet.*, 14:360–6.

Garey, M. and Johnson, D. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company.

Gazis, R., Miadlikowska, J., Lutzoni, F., Arnold, A., and Chaverri, P. (2012). Culture-based study of endophytes associated with rubber trees in peru reveals a new class of pezizomycotina: Xylonomycetes. *Molecular Phylogenetics and Evolution*, 65:294–304.

Ge, F., Wang, L., and Kim, J. (2005). The cobweb of life revealed by genome scale estimates of horizontal gene transfer. *PLoS Biol.*, 3:e316.

Geering, A., Maumus, F., Copetti, D., Choisne, N., Zwickl, D., Zytnicki, M., McTaggart, A., Scalabrin, S., Vezzulli, S., Wing, R., and et al. (2014). Endogenous florendoviruses are major components of plant genomes and hallmarks of virus evolution. *Nat. Commun.*, 5:5269.

Giovannoni, S., Tripp, H., Givan, S., Podar, M., Vergin, K., Baptista, D., Bibbs, L., Eads, J., Richardson, T., Noordewier, M., Rappé, M., Short, J., Carrington, J., and Mathur, E. (2005a). Genome streamlining in a cosmopolitan oceanic bacterium. *Science*, 309:1242–1245.

Giovannoni, S., Tripp, H., Givan, S., Podar, M., Vergin, K., Baptista, D., Bibbs, L., Eads, J., Richardson, T., Noordewier, M., Rappé, M., Short, J., Carrington, J., and Mathur, E. (2005b). Genome streamlining in a cosmopolitan oceanic bacterium. *Science*, 309:1242–1245.

Giribet, G., , and Edgecombe, G. (2012). Reevaluating the arthropod tree of life. *Annual Review of Entomology*, 57:167–186.

Gorecki, P., Burleigh, G., and Eulenstein, O. (2011). Maximum likelihood models and algorithms for gene tree evolution with duplications and losses. *BMC Bioinformatics.*, 12:S15.

Gould, S. (1997). *Full House: The Spread of Excellence from Plato to Darwin*. Three Rivers Press., New York.

Gould, S. and Eldredge, N. (1977). Punctuated equilibria: the tempo and mode of evolution reconsidered. *Paleobiology*, 3:115–151.

Grant, G. R. and Ewens, W. J. (2001). *Statistical methods in bioinformatics: An introduction.* Statistics for biology and health. Springer-Verlag, New York, Berlin, Heidelberg.

Greenberg, A., Moran, J., Coyne, J., and Wu, C. (2003). Ecological adaptation during incipient speciation revealed by precise gene replacement. *Science*, 302:1754–1757.

Grosberg, R. and Strathmann, R. (2007). The evolution of multicellularity: A minor major transition? *Annu. Rev. Ecol. Evol. Syst.*, 38:621–54.

Gumbel, E. (2013). *Statistics of extremes*. Echo Point Books & Media.

Guo, Y. (2013). Gene family evolution in green plants with emphasis on the origination and evolution of arabidopsis thaliana genes. *Plant J*, 73:941–951.

Gusfield, D. (1997). *Algorithms on Strings, Trees and Sequences.* Cambridge University Press.

Haag, K., James, T., Pombert, J.-F., Larsson, R., Schaer, T., Refardt, D., and Ebert, D. (2014). Evolution of a morphological novelty occurred before genome compaction in a lineage of extreme parasites. *Proceedings of the National Academy of Sciences*, 111:15480–15485.

Hahn, M., Demuth, J., and Han, S. (2007a). Accelerated rate of gene gain and loss in primates. *Genetics.*, 177:1941–1949.

Hahn, M., Han, M., and Han, S. (2007b). Gene family evolution across 12 drosophila genomes. *PLoS Genetics*, 3:e197.

Hauser, M., Mayer, C., and S oding, J. (2013). kclust: fast and sensitive clustering of large protein sequence databases. *BMC Bioinformatics*, 14:248.

He, M., Munro, J., and Nicholson, P. (2011). Dynamic range selection in linear space. *Algorithms and Computation*, 7074:160–169.

Hedges, S., Dudley, J., and Kumar, S. (2006). Timetree: a public knowledge-base of divergence times among organisms. *Bioinformatics*, 22:2971–2972.

Heinen, T. J., Staubach, F., H aming, D., and Tautz, D. (2009). Emergence of a new gene from an intergenic region. *Curr. Biol.*, 19:1527–1531.

Hemmrich, G. and Bosch, T. (2008). Compagen, a comparative genomics platform for early branching metazoan animals reveals early origins of of genes regulating stem cell differentiation. *BioEssays*, 20:1010–1018.

Hillis, D. M. (1994). *Homology in molecular biology, in Hall BK, editor. Homology: The Hierarchical Basis of Comparative Biology*. Academic Press, San Diego.

Hoballah, M. E., Gübitz, T., Stuurman, J., Broger, L., Barone, M., Mandel, T., Dell'Olivo, A., Arnold, M., and Kuhlemeier, C. (2007). Single gene-mediated shift in pollinator attraction in petunia. *Plant Cell*, 19:779–790.

Holland, J. (1996). *Hidden Order: How Adaptation Builds Complexity*. Basic Books.

Hottes, A. K., Freddolino, P., Khare, A., Donnell, Z., Liu, J., and Tavazoie, S. (2013). Bacterial adaptation through loss of function. *PLoS Genet.*, 9:e1003617.

Hughes, A. and Friedman, R. (2004). Differential loss of ancestral gene families as a source of genomic divergence in animals. *Proc Biol Sci*, 271:S107–9.

Hui, C. (2006). Carrying capacity, population equilibrium, and environment's maximal load. *Ecological Modelling.*, 192:317–320.

Ivancevic, A. M., Walsh, A. M., Kortschak, R. D., and Adelson, D. L. (2013). Jumping the fine line between species: Horizontal transfer of transposable elements in animals catalyses genome evolution. *Bioessays*, 35:1071–1082.

Jaccard, P. (1901). Étude comparative de la distribution florale dans une portion des alpes et des jura. *Bulletin de la Société Vaudoise des Sciences Naturelles*, 37:547–579.

James, T., Pelin, A., Bonen, L., Ahrendt, S., Sain, D., Corradi, N., and Stajich, J. (2013). Shared signatures of parasitism and phylogenomics unite cryptomycota and microsporidia. *Current Biology*, 23:1548–1553.

Jarvis, E., Mirarab, S., Aberer, A., Li, B., Houde, P., C, L., Ho, S., Faircloth, B., Nabholz, B., Howard, J., and et al. (2014). Whole-genome analyses resolve early branches in the tree of life of modern birds. *Science*, 346:1320–1331.

Jetz, W., Thomas, G., Joy, J., Hartmann, K., and Mooers, A. (2012). The global diversity of birds in space and time. *Nature*, 491:444–448.

Johnson, B., Borowiec, M., Chiu, J., Lee, E., Atallah, J., , and Ward, P. (2013). Phylogenomics resolves evolutionary relationships among ants, bees, and wasps. *Current Biology*, 23:2058–2062.

J.W, S., Sung, G.-H., Johnson, D., Hesse, C., O'Rourke, B., Serdani, M., Spotts, R., Lutzoni, F., Hofstetter, V., Miadlikowska, J., and et al. (2006). A five-gene phylogeny of pezizomycotina. *Mycologia*, 98:1018–1028.

Kaessmann, H. (2010). Origins, evolution, and phenotypic impact of new genes. *Genome Research*, 20:1313–1326.

Kamath, R. S., Fraser, A., Dong, Y., Poulin, G., Durbin, R., Gotta, M., Kanapin, A., Le Bot, N., Moreno, S., Sohrmann, M., Welchman, D., Zipperlen, P., and Ahringer, J. (2003). Systematic functional analysis of the caenorhabditis elegans genome using rnai. *Nature*, 421:231–237.

Karlin, S. and Altschul, S. (1990). Methods for assessing the statistical significance of molecular sequence features by using general scoring schemes. *Proc. Natl. Acad. Sci. USA*, 87:2264–2268.

Karp, R. M. (1972). Reducibility among combinatorial problems. *Complexity of Computer Computations. New York: Plenum*, 1:85–103.

Keeling, P. and Palmer, J. (2008). Horizontal gene transfer in eukaryotic evolution. *Nat Rev Genet.*, 9:605–18.

Khalturin, K., Anton-Erxleben, F., Sassmann, S., Wittlieb, J., Hemmrich, G., and Bosch, T. (2008). A novel gene family controls species-specific morphological traits in hydra. *PLoS Biol.*, 6:e278.

Khalturin, K., Hemmrich, G., Fraune, S., Augustin, R., and Bosch, T. (2009). More than just orphans: are taxonomically-restricted genes important in evolution? *Trends Genet.*, 25:404–13.

King, N. (2004). The unicellular ancestry of animal development. *Dev. Cell*, 7:313–25.

Kleisner, K. (2007). The formation of the theory of homology in biological sciences. *Acta Biotheor.*, 55:317–40.

Knuth, D. E. (1998). *The Art of Computer Programming*. Addison-Wesley Professional.

Kolmogorov, A. (1965). Three approaches to the definition of the concept "quantity of information". *Problems Inform. Transmission*, 1:1–7.

Kolmogorov, A. (1983). Combinatorial foundations of information theory and the calculus of probabilities. *Russ. Math. Surv.*, 38:29–40.

Kolmogorov, A. (1998). On tables of random numbers. *Theoretical Computer Science*, 25:387–395.

Koonin, E. (2005). Orthologs, paralogs, and evolutionary genomics. *Annu Rev Genet.*, 39:309–38.

Koonin, E. (2009). Evolution of genome architecture. *Int J Biochem Cell Biol.*, 41:298–306.

Koonin, E. (2011). *Logic of Chance, The: The Nature and Origin of Biological Evolution*. Upper Saddle River: FT Press.

Kortschak, R. D., Samuel, G., Saint, R., and Miller, D. J. (2003). Est analysis of the cnidarian acropora millepora reveals extensive gene loss and rapid sequence divergence in the model invertebrates. *Curr. Biol.*, 13:2190–2195.

Koskiniemi, S., Sun, S., Berg, O., and Andersson, D. (2012). Selection-driven gene loss in bacteria. *PLoS Genet.*, 8:e1002787.

Kumar, S. (2005). Molecular clocks: Four decades of evolution. *Nature Reviews Genetics*, 6:654–662.

Kunin, V., Goldovsky, L., Darzentas, N., and Ouzounis, C. (2005a). The net of life: reconstructing the microbial phylogenetic network. *Genome Res.*, 15:954–9.

Kunin, V. and Ouzounis, C. (2003). Genetrace-reconstruction of gene content of ancestral species. *Bioinformatics.*, 19:1412–6.

Kunin, V., Teichmann, S., Huynen, M., and Ouzounis, C. (2005b). The properties of protein family space depend on experimental design. *Bioinformatics*, 21:2618–22.

Kurtzman, C. and Robnett, C. (2013). Relationships among genera of the saccharomycotina (ascomycota) from multigene phylogenetic analysis of type species. *FEMS Yeast Research*, 13:23–33.

Kusserow, A., Pang, K., Sturm, C., Hrouda, M., Lentfer, J., Schmidt, H., Technau, U., von Haeseler, A., Hobmayer, B., Martindale, M., and Holstein, T. (2005). Unexpected complexity of the wnt gene family in a sea anemone. *Nature*, 433:156–160.

Lee, M. and Marx, C. (2012). Repeated, selection-driven genome reduction of accessory genes in experimental populations. *PLoS Genet.*, 8:e1002651.

Lee, M., Soubrier, J., and Edgecombe, G. (2013). Rates of phenotypic and genomic evolution during the cambrian explosion. *Current Biology*, 23:1889–1895.

Leonard, G. and Richards, T. (2012). Genome-scale comparative analysis of gene fusions, gene fissions, and the fungal tree of life. *PNAS*, 109:21402–21407.

Ley, R., Lozupone, C., Hamady, M., Knight, R., and Gordon, J. (2008). Worlds within worlds: evolution of the vertebrate gut microbiota. *Nat Rev Microbiol.*, 6:776–788.

Li, C. Y., Zhang, Y., Wang, Z., Zhang, Y., Cao, C., Zhang, P., Lu, S., Li, X., Yu, Q., Zheng, X., Du, Q., Uhl, G., Liu, Q., and Wei, L. (2010). A human-specific de novo proteincoding gene associated with human brain functions. *PloS Comput. Biol.*, 6:e1000734.

Li, L., Stoeckert, C. J., and Roos, D. (2003). Orthomcl: identification of ortholog groups for eukaryotic genomes. *Genome Res.*, 13:2178–89.

Li, W. and Godzik, A. (2006). Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences. *Bioinformatics*, 22:1658–9.

Librado, P., Vieira, F., and Rozas, J. (2012). Badirate: estimating family turnover rates by likelihood-based methods. *Bioinformatics*, 28:279–281.

Lipman, D. and Pearson, W. (1985). Rapid and sensitive protein similarity searches. *Science*, 227:1435–41.

Lloyd, S. (2001). Measures of complexity: a nonexhaustive list. *IEEE Control Systems*, 21:7–8.

Lynch, M. (2006). Streamlining and simplification of microbial genome architecture. *Annu Rev Microbiol.*, 60:327–49.

Lynch, M. (2007). *The Origins of Genome Architecture.* Sinauer Associates, Sunderland, MA.

Lynch, M. and Conery, J. (2000). The evolutionary fate and consequences of duplicate genes. *Science*, 290:1151–5.

Lynch, M. and Conery, J. (2003). The origins of genome complexity. *Science*, 302:1401–4.

Marri, P. and Golding, G. (2008). Gene amelioration demonstrated: the journey of nascent genes in bacteria. *Genome*, 5:164 –168.

Martin-Duran, J., Janssen, R., Wennberg, S., Budd, G., and Hejnol, A. (2012). Deuterostomic development in the protostome priapulus caudatus. *Current Biology*, 22:2161 – 2166.

McLysaght, A. and Guerzoni, D. (2015). New genes from non-coding sequence: the role of de novo protein-coding genes in eukaryotic evolutionary innovation. *Phil.Trans. R. Soc. B*, 370:20140332.

McShea, D. (2015). Three trends in the history of life: An evolutionary syndrome. *Evolutionary Biology*, 0:1–12.

McShea, D. and Hordijk, W. (2013). Complexity by subtraction. *Evolutionary Biology*, 40:504–520.

McShea, D. W. (1996). Metazoan complexity and evolution: Is there a trend? *Evolution*, 50:477–492.

McShea, D. W. (2000). Functional complexity in organisms: Parts as proxies. *Biology and Philosophy*, 15:641–668.

McShea, D. W. (2001). The hierarchical structure of organisms: A scale and documentation of a trend in the maximum. *Paleobiology*, 27:405–423.

McShea, D. W. (2002). A complexity drain on cells in the evolution of multicellularity. *Evolution*, 56:441–452.

Mendonca, A., Alves, R., and Pereira-Leal, J. (2011). Loss of genetic redundancy in reductive genome evolution. *PLoS Comput Biol*, 7:e1001082.

Meredith, R., Janecka, J., Gatesy, J., Ryder, O., Fisher, C., Teeling, E., Goodbla, A., Eizirik, E., Simao, T., Stadler, T., and et al. (2011). Impacts of the cretaceous terrestrial revolution and kpg extinction on mammal diversification. *Science*, 334:521–524.

Misof, B., Liu, S., Meusemann, K., Peters, R., Donath, A., Mayer, C., Frandsen, P., Ware, J., Flouri, T., Beutel, R., and et al. (2014). Phylogenomics resolves the timing and pattern of insect evolution. *Science*, 346:763–767.

Mitelman, F., Johansson, B., and Mertens, F. (2007). The impact of translocations and gene fusions on cancer causation. *Nature reviews. Cancer*, 7:233–45.

Moore, M., Soltis, P., Bell, C., Burleigh, J., and Soltis, D. (2010). Phylogenetic analysis of 83 plastid genes further resolves the early diversification of eudicots. *PNAS*, 107:4623–4628.

Moroz, L., Kocot, K., Citarella, M., Dosung, S., Norekian, T., Povolotskaya, I., Grigorenko, A., Dailey, C., Berezikov, E., Buckley, K., and et al. (2014). The ctenophore genome and the evolutionary origins of neural systems. *Nature*, 510:109–114.

Morris, J. (2015). Black queen evolution: the role of leakiness in structuring microbial communities. *Trends in Genetics*, 31:475–482.

Morris, J., Lenski, R., and Zinser, E. (2012). The black queen hypothesis: evolution of dependencies through adaptive gene loss. *mBio*, 3:e00036–12.

Nakhleh, L., Ruths, D., and Innan, H. (2009). *Gene trees, species trees, and species networks*. .

Navarro, G. (2001). A guided tour to approximate string matching. *ACM Computing Surveys*, 33:31–88.

Near, T., Eytan, R., Dornburg, A., Kuhn, K., Moore, J., Davis, M., Wainwright, P., Friedman, M., and Smith, W. (2012). Resolution of ray-finned fish phylogeny and timing of diversification. *Proceedings of the National Academy of Sciences*, 109:13698–13703.

Needleman, S. and Wunsch, C. (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J Mol Biol*, 48:443–53.

Nei, M., Gu, X., and Sitnikova, T. (1997). Evolution by the birth-and-death process in multigene families of the vertebrate immune system. *Proc Natl Acad Sci U S A.*, 94:7799–806.

Nei, M. and Hughes, A. (1992). *Balanced polymorphism and evolution by the birth-and-death process in the MHC loci*. Tsuji K, Aizawa M, Sasazuki T, editors. 11th Histocompatibility Workshop and Conference. Oxford Univ. Press; Oxford, UK.

Nei, M. and Rooney, A. (2005). Concerted and birth-and-death evolution in multigene families. *Annu Rev Genet*, 39:121–152.

Neme, R. and Tautz, D. (2013). Phylogenetic patterns of emergence of new genes support a model of frequent de novo evolution. *Bmc Genomics*, 14:117.

Neme, R. and Tautz, D. (2014). Evolution: dynamics of de novo gene emergence. *Curr Biol.*, 24:R238–40.

Neme, R. and Tautz, D. (2016). Fast turnover of genome transcription across evolutionary time exposes entire non-coding dna to de novo gene emergence. *eLife*, 5:e09977.

Nepusz, T., Sasidharan, R., and Paccanaro, A. (2010). Scps: a fast implementation of a spectral method for detecting protein families on a genome-wide scale. *BMC Bioinformatics*, 11:120.

Nordberg, H., Cantor, M., Dusheyko, S., Hua, S., Poliakov, A., Shabalov, I., Smirnova, T., Grigoriev, I., and Dubchak, I. (2014). The genome portal of the department of energy joint genome institute: 2014 updates. *Nucleic Acids Res.*, 42:D26–31.

Nosenko, T., Schreiber, F., Adamska, M., Adamski, M., Eitel, M., Hammel, J., Maldonado, M., M uller, W., Nickel, M., Schierwater, B., and et al. (2013). Deep metazoan phylogeny: When different genes tell different stories. *Molecular Phylogenetics and Evolution*, 67:223–233.

Ochman, H. (2001). Lateral and oblique gene transfer. *Curr. Opin. Genet. Dev.*, 11:616–619.

Ohno, S. (1970). *Evolution by gene duplication*. Springer.

Osorio, J. (2015). Functional genomics: the genetic essence of human cells. *Nat. Rev. Genet.*, 16:683.

Ouzounis, C., Kunin, V., Darzentas, N., and Goldovsky, L. (2006). A minimal estimate for the gene content of the last universal common ancestor—exobiology from a terrestrial perspective. *Res Microbiol.*, 157:57–68.

O'Leary, M., Bloch, J., Flynn, J., Gaudin, T., Giallombardo, A., Giannini, N., Goldberg, S., Kraatz, B., Luo, Z.-X., Meng, J., and et al. (2013). The placental mammal ancestor and the post–k-pg radiation of placentals. *Science*, 339:662–667.

Palmieri, N., Kosiol, C., and Schlötterer, C. (2014). The life cycle of drosophila orphan genes. *eLife*, 3:e01311.

Papsa, J., Medina-Chacóna, L., Marshall, W., Suga, H., and Ruiz-Trillo, I. (2013). Molecular phylogeny of unikonts: New insights into the position of apusomonads and ancyromonads and the internal relationships of opisthokonts. *Protist*, 164:2–12.

Partensky, F. and Garczarek, L. (2010). Prochlorococcus: advantages and limits of minimalism. *Ann Rev Mar Sci.*, 2:305–31.

Payne, S. and Loomis, W. (2006). Retention and loss of amino acid biosynthetic pathways based on analysis of whole-genome sequences. *Eukaryot Cell*, 5:272–6.

Pearson, K. (1895). Notes on regression and inheritance in the case of two parents. *Proceedings of the Royal Society of London*, 58:240–242.

Pearson, K. (1900). On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *Philosophical Magazine Series 5.*, 50:157 – 175.

Pearson, W. (2013). *An introduction to sequence similarity ("homology") searching*. by John Wiley and Sons, Inc.

Pearson, W. and Sierk, M. (2005). The limits of protein sequence comparison? *Curr Opin Struct Biol.*, 15:254–60.

Peters, R., Meusemann, K., Petersen, M., Mayer, C., Wilbrandt, J., Ziesmann, T., Donath, A., Kjer, K., Aspöck, U., Aspöck, H., and et al. (2014). The evolutionary history of holometabolous insects inferred from transcriptome-based phylogeny and comprehensive morphological data. *BMC Evolutionary Biology*, 14:52.

Prachumwat, A. and Li, W. (2008). Gene number expansion and contraction in vertebrate genomes with respect to invertebrate genomes. *Genome Res.*, 18:221–232.

Pruitt, K., Tatusova, T., and Brown, G.R.and Maglott, D. (2012). Ncbi reference sequences (refseq): current status, new features and genome annotation policy. *Nucleic Acids Res.*, 14:D130–D135.

Ptitsyn, A. and Moroz, L. (2012). Computational workflow for analysis of gain and loss of genes in distantly related genomes. *BMC Bioinformatics*, 13.

Putnam, N. H., Srivastava, M., Hellsten, U., Dirks, B., Chapman, J., Salamov, A., Terry, A., Shapiro, H., Lindquist, E., Kapitonov, V., Jurka, J., Genikhovich, G., Grigoriev, I., Lucas, S., Steele, R., Finnerty, J., Technau, U., Martindale, M., and Rokhsar, D. (2007). Sea anemone genome reveals ancestral eumetazoan gene repertoire and genomic organization. *Science*, 317:86–94.

Ragan, M. (2001). Detection of lateral gene transfer among microbial genomes. *Curr. Opin. Genet. Dev.*, 11:620–626.

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

Refulio-Rodriguez, N. and Olmstead, R. (2014). Phylogeny of lamiidae. *The American Journal of Botany*, 101:287–299.

Reis, M., Inoue, J., Hasegawa, M., Asher, R., Donoghue, P., and Yang, Z. (2012). Phylogenomic datasets provide both precision and accuracy in estimating the timescale of placental mammal phylogeny. *Proceedings of the Royal Society of London B: Biological Sciences*, page rspb20120683.

Rognes, T. (2001). Paralign: a parallel sequence alignment algorithm for rapid and sensitive database searches. *Nucleic Acids Res.*, 29:1647–1652.

Rosenberg, E., Koren, O., Reshef, L., Efrony, R., and Zilber-Rosenberg, I. (2007). The role of microorganisms in coral health, disease and evolution. *Nat Rev Microbiol.*, 5:355–62.

Rost, B. (1999). Twilight zone of protein sequence alignments. *Protein Eng.*, 12:85–94.

Ruiz-Trillo, I., Burger, G., Holland, P. W., King, N., Lang, B. F., Roger, A. J., and Gray, M. W. (2007). The origins of multicellularity: a multi-taxon genome initiative. *Trends Genet.*, 23:113–8.

Sadd, B., Barribeau, S., Bloch, G., Graaf, D. d., Dearden, P., Elsik, C., Gadau, J., Grimmelikhuijzen, C., Hasselmann, M., Lozier, J., and et al. (2015). The genomes of two key bumblebee species with primitive eusocial organization. *Genome Biology*, 16:76.

Sakarya, O., Kosik, K., and Oakley, T. (2008). Reconstructing ancestral genome content based on symmetrical best alignments and dollo parsimony. *Bioinformatics*, 24:606–12.

Sanderson, M. (2003). r8s: inferring absolute rates of molecular evolution and divergence times in the absence of a molecular clock. *Bioinformatics*, 19:301–302.

Sanjeev, A. and Boaz, B. (2009). *Computational Complexity: A Modern Approach*. Cambridge.

Schaeffer, S. (2007). Graph clustering. *Computer Science Review*, 1:27 – 64.

Schlötterer, C. (2015). Genes from scratch – the evolutionary fate of de novo genes. *Trends in Genetics*, 31:215–219.

Seetharam, A., , and Stuart, G. (2013). Whole genome phylogeny for 21 drosophila species using predicted 2b-rad fragments. *PeerJ*, 1:e226.

Shannon, C. and Weaver, W. (1949). *The Mathematical Theory of Communication.* Univ of Illinois Press.

Shapira, D. and Storer, J. (2007). Edit distance with move operations. *J. Discret. Algorithms*, 5:380–392.

Sharpton, T., Stajich, J., Rounsley, S., Gardner, M., Wortman, J., Jordar, V., Maiti, R., Kodira, C., Neafsey, D., Zeng, Q., Hung, C., McMahan, C., Muszewska, A., Grynberg, M., Mandel, M., Kellner, E., Barker, B., Galgiani, J., Orbach, M., Kirkland, T., Cole, G., Henn, M., Birren, B., and Taylor, J. (2009). Comparative genomic analyses of the human fungal pathogens coccidioides and their relatives. *Genome Res.*, 19:1722–31.

Shen, M., Davis, F., and Sali, A. (2005). The optimal size of a globular protein domain: A simple sphere-packing model. *Chemical Physics Letters*, 405:224–228.

Simon, H. (1969). *The Sciences of the Artificial*. The MIT Press.

Simpson, G. (1944). *Tempo and Mode in Evolution.* New York: Columbia Univ. Press.

Singh, N., Larracuente, A., Sackton, T., , and Clark, A. (2009). Comparative genomics on the drosophila phylogenetic tree. *Annual Review of Ecology, Evolution, and Systematics*, 40:459–480.

Sogin, M., Elwood, H., and Gunderson, J. (1986). Evolutionary diversity of eukaryotic small-subunit rrna genes. *1986*, 83:1383–1387.

Soltis, D., Smith, S., Cellinese, N., Wurdack, K., Tank, D., Brockington, S., Refulio-Rodriguez, N., Walker, J., Moore, M., Carlsward, B., and et al. (2011). Angiosperm phylogeny: 17 genes, 640 taxa. *American Journal of Botany*, 98:704–730.

Song, S., Liu, L., Edwards, S., , and Wu, S. (2012). Resolving conflict in eutherian mammal phylogeny using phylogenomics and the multispecies coalescent model. *PNAS*, 109:14942–14947.

Sonnhammer, E., Gabaldón, T., Sousa da Silva, A., Martin, M., Robinson-Rechavi, M., Boeckmann, B., Thomas, P., Dessimoz, C., and for Orthologs consortium, Q. (2014). Big data and other challenges in the quest for orthologs. *Bioinformatics*, 30:2993–2998.

Sonnichsen, B., Koski, L., Walsh, A., Marschall, P., Neumann, B., Brehm, M., Alleaume, A., Artelt, J., Bettencourt, P., Cassin, E., Hewitson, M., Holz, C., Khan, M., Lazik, S., Martin, C., Nitzsche, B., Ruer, M., Stamford, J., Winzi, M., Heinkel, R., Röder, M., Finell, J., Häntsch, H., Jones, S., Jones, M., Piano, F., Gunsalus, K., Oegema, K., Gönczy, P., Coulson, A., Hyman, A., and Echeverri, C. (2005). Full-genome rnai profiling of early embryogenesis in caenorhabditis elegans. *Nature*, 434:462–469.

Sprent, P. (1998). *Data driven statistical methods.* London: Chapman and Hall.

Stedman, H. H., Kozyak, B., Nelson, A., Thesier, D., Su, L., Low, D., Bridges, C., Shrager, J., Minugh-Purvis, N., and Mitchell, M. (2004). Myosin gene mutation correlates with anatomical changes in the human lineage. *Nature*, 428:415–418.

Struck, T., Wey-Fabrizius, A., Golombek, A., Hering, L., Weigert, A., Bleidorn, C., Klebow, S., Iakovenko, N., Hausdorf, B.v Petersen, M., and et al. (2014). Platyzoan paraphyly based on phylogenomic data supports a noncoelomate ancestry of spiralia. *Mol Biol Evol*, 31:1833–1849.

Swofford, D. and Maddison, W. (1987). Reconstructing ancestral character states under wagner parsimony. *Math. Biosci.*, 87:199–229.

Szathmáry, E., Jordan, F., and Pal, C. (2001). Molecular biology and evolution: Can genes explain biological complexity? *Science*, 292:1315 – 1316.

Tarjan, R. and Vishkin, U. (1984). Finding biconnected components and computing tree functions in logarithmic parallel time. *Proceedings of FOCS.*, page 12–20.

Tautz, D. (2014). The discovery of de novo gene evolution. *Perspect Biol Med.*, 57:149–61.

Tautz, D. and Domazet-Lošo, T. (2011). The evolutionary origin of orphan genes. *Nature Reviews Genetics*, 12:692–702.

Taylor, F. (1978). Problems in the development of an explicit hypothetical phylogeny of the lower eukaryotes. *Biosystems*, 10:67–89.

Technau, U., Rudd, S., Maxwell, P., Gordon, P., Saina, M., Grasso, L., Hayward, D., Sensen, C., Saint, R., Holstein, T., Ball, E., and Miller, D. (2005). Maintenance of ancestral complexity and non-metazoan genes in two basal cnidarians. *Trends in Genetics*, 21:633–639.

Torruella, G., Derelle, R., Paps, J., Lang, B., Roger, A., Shalchian-Tabrizi, K., and Ruiz-Trillo, I. (2012). Phylogenetic relationships within the opisthokonta based on phylogenomic analyses of conserved single-copy protein domains. *Mol Biol Evol*, 29:531–544.

Trautwein, M., Wiegmann, B., Beutel, R., Kjer, K., , and Yeates, D. (2012). Advances in insect phylogeny at the dawn of the postgenomic era. *Annual Review of Entomology*, 57:449–468.

Trifonov, E. and Berezovsky, I. (2003). Evolutionary aspects of protein structure and folding. *Current Opinion in Structural Biology*, 13:110–114.

Ukkonen, E. (1992). Approximate string matching with q-grams and maximal matches. *Theoretical Computer Science*, 92:191–211.

Valentine, J. W., Collins, A. G., and Meyer, C. P. (1994). Morphological complexity increase in metazoans. *Paleobiology*, 20:131–142.

Van Valen, L. (1973). A new evolutionary law. *Evolutionary Theory*, 1:1–30.

Vogan, A. and Higgs, P. (2011). The advantages and disadvantages of horizontal gene transfer and the emergence of the first species. *Biol Direct*, 6:1.

Wagner, A. (2008). Neutralism and selectionism: a network-based reconciliation. *Nat. Rev. Genet.*, 9:965–974.

Walsh, J. B. and Stephan, W. (2001). Multigene families: Evolution. *ELS*, 0:0–0.

Ward, P. (2014). The phylogeny and evolution of ants. *Annual Review of Ecology, Evolution, and Systematics*, 45:23–43.

White, J. K., Gerdin, A., Karp, N., Ryder, E., Buljan, M., Bussell, J., Salisbury, J., Clare, S., Ingham, N., Podrini, C., Houghton, R., Estabel, J., Bottomley, J., Melvin, D., Sunter, D., Adams, N., Sanger Institute Mouse Genetics Project, Tannahill, D., Logan, D., Macarthur, D., Flint, J., Mahajan, V., Tsang, S., Smyth, I., Watt, F., Skarnes, W., Dougan, G., Adams, D., Ramirez-Solis, R., Bradley, A., and Steel, K. (2013). Genome-wide generation and systematic phenotyping of knockout mice reveals new roles for many genes. *Cell*, 154:452–464.

Whitney, K. and Garland, T. (2010). Did genetic drift drive increases in genome complexity? *PLoS Genet.*, 6:e1001080.

Wiegmann, B., Trautwein, M., Winkler, I., Barr, N., Kim, J.-W., Lambkin, C., Bertone, M., Cassel, B., Bayless, K., Heimberg, A., and et al. (2011). Episodic radiations in the fly tree of life. *PNAS*, 108:5690–5695.

Wilk, M.B.and Gnanadesikan, R. (1968). Probability plotting methods for the analysis of data. *Biometrika*, 55:1–17.

Wissler, L., Gadau, J., Simola, D., Helmkampf, M., and Bornberg-Bauer, E. (2013). Mechanisms and dynamics of orphan gene emergence in insect genomes. *Genome Biology and Evolution*, 5:439–455.

Wolf, Y. and Koonin, E. (2013). Genome reduction as the dominant mode of evolution. *Bioessays*, 35:829–37.

Wood, B. and Richmond, B. (2000). Human evolution: taxonomy and paleobiology. *Journal of Anatomy*, 197:19–60.

Worheide, G., Dohrmann, M., Erpenbeck, D., Larroux, C., Maldonado, M., Voigt, O., Borchiellini, C., and Lavrov, D. (2012). *Chapter One - Deep Phylogeny and Evolution of Sponges (Phylum Porifera).* In Advances in Marine Biology. M.J.U., Manuel Maldonado and Xavier Turon Mikel A. Becerro, ed. (Academic Press).

Wyder, S., Kriventseva, E. V., Schroder, R., Kadowaki, T., and Zdobnov, E. M. (2007). Quantification of ortholog losses in insects and vertebrates. *Genome Biol.*, 8:R242.

Xu, D. and Nussinov, R. (1998). Favorable domain size in proteins. *Folding and Design*, 3:11–17.

Yang, Z., Yu, J., and Kitsuregawa, M. (2010). Fast algorithms for top-k approximate string matching. *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*, pages 1467–1473.

Yeung, R. (2006). *A First Course in Information Theory (Information Technology: Transmission, Processing and Storage).* Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Yi, S. (2006). Non-adaptive evolution of genome complexity. *Bioessays*, 28:979–82.

Zezula, P., Amato, G., Dohnal, V., and Batko, M. (2006). *Similarity Search; The Metric Space Approach.* Springer.

Zhang, H. (2003). Alignment of blast high-scoring segment pairs based on the longest increasing subsequence algorithm. *Bioinformatics*, 22:1391–6.

Zhou, Q., Zhang, G., Zhang, Y., Xu, S., Zhao, R., Zhan, Z., Li, X., Ding, Y., Yang, S., and Wang, W. (2008). On the origin of new genes in drosophila. *Genome Research*, 18:1446–1455.

Zobel, J., Moffat, A., and Ramamohanarao, K. (1998). Inverted files versus signature files for text indexing. *ACM Transactions on Database Systems*, 23:453–490.

Zufall, R. A. and Rausher, M. D. (2004). Genetic changes associated with floral adaptation restrict future evolutionary potential. *Nature*, 428:847–850.

Zurek, W. (1989). Algorithmic randomness and physical entropy. *Phys. Rev. A*, 40:4731–4751.

Appendix A

Supplementary material

A.1 Introduction

Supplementary materials in this section contain a computational appendix with basic definitions and concepts (Section A.2) required for understanding of newly introduced computation strategies, algorithms and distance measures that were utilized for executing fast search queries in Section A.3. Furthermore, the section contains raw information associated to gain and loss of gene families that are further processed and used in Chapter 3 together with additional results not shown in the main text of this thesis. Moreover, a complete list of genomes in *db_200514* database, with associated taxonomy information and repository location is summarized in Table A.4.

A.2 Basic Definitions and Concepts

A.2.1 Strings and k-mers

Throughout the thesis the notation "| |" refers to *size* and should not be confused with other definitions unless explicitly stated. In case of a set, the size will refer to the number of elements, whereas in the case of a string (a sequence of characters) the size will refer to its length. For string ω of length $|\omega|$, let $\omega[i]$ denote the i-th character of the string element of the alphabet $|\Sigma|$ and $\omega[i,j]$ its substring, for $i \leq x \leq j$ and $i, j \in [1, |\omega|]$ where $\omega[x] \in \Sigma$. When a substring starts at the beginning of the string (at position 1) and terminates before its end, the substring is called a prefix. On the other hand if the substring starts at a position $i \in [2..|\omega|]$ and terminates at the end of ω it is referred to as suffix.

A substring $\omega[i,j]$ of a fixed size is also called a $k-$mer (denoted as κ in the rest of the thesis) if $j-i+1=k$. Sometimes in a literature terms like *$k-$tuple or $k-$word* can also be found referring to a k size substring. Given an alphabet Σ and a string over that alphabet, a set of possible $k-$mer types ($\mathscr{K}_\kappa$) that can appear in that string is defined as:

$$\mathscr{K}_\kappa = \{\kappa_1, \kappa_2, ..., \kappa_L\} \tag{A.1}$$

where $L = |\Sigma|^k$. In practice, computing $k-$mers of a string ω can be described as a task of counting the number of overlapping substring occurrences of size k, by sliding a $k-$size "window" across the underlying string, starting at position 1 and terminating at position $|\omega|-k+1$. Thus an *occurrence count* of a $k-$mer κ is accordingly defined as the number of k-mers appearing in a given string.

By ordering the occurrence count values, using some pre-specified key (usually lexicographic order on the alphabet $|\Sigma|^k$), a surjective transformation is defined, assigning a string to a point in multidimensional integer space. The vector is defined as an *occurrence count vector*:

$$\vec{c_k^{\omega}} = (c_{k,1}^{\omega}, c_{k,2}^{\omega}, ..., c_{k,L}^{\omega}) \tag{A.2}$$

Three important facts follow from this definition. First, the transformed string has $|\Sigma|^k$ dimensions, the number of which is completely unrelated to its length $|\omega|$. Second, the sum of entries is not related to the content of ω. Third, all count values are non-negative.

As an example of the above stated definitions consider a string $\omega = TATATG$ over $\Sigma = \{A,T,G,C\}$. It follows that the string size $|\omega| = 6$ and the size of the alphabet is $|\Sigma| = 4$. Given $k = 3$, κ_3 in ω refers to $\omega[3,5] = TAT$. Furthermore, according to definition A.1, $\mathscr{K}_{\kappa} = \{AAA, AAC, AAG, AAT, ACA, ...\}$, and the number of elements it contains is equal to $|\Sigma|^k = 64$. Finally, the occurrence count vector computed by sliding a three letter "window" over a string ($|\omega| - k + 1 = 4$ slides) is:

$$\vec{c_k^{\omega}} = (0_{AAA}, \ldots, 1_{ATA}, 0_{ATC}, 1_{ATG}, \ldots, 1_{TAG}, 2_{TAT}, \ldots, 0_{TTT})$$

Note that the vector has a length of $L = |\Sigma|^k = 4^3 = 64$ and the zero values indicate k-mers, which are not present in ω.

A.2.2 Computational Complexity

In the first chapter of this thesis the notion of complexity was introduced. It was argued that an adaptive system such as a biological system requires a compatible complexity measure that is is a more suitable than Kolmongorov's complexity (Kolmogorov, 1998) (the measure of regularity associated to strings). In this section I introduce and describe an additional concept called computational complexity and the way it is quantified, relevant for asymptotic analysis that have been preformed on several occasions in this thesis.

Computational complexity describes how difficult a given computational task is with respect to its input size (Sanjeev and Boaz, 2009). Since each computational process is essentially a sequence of computational steps, computational complexity can be seen as a

measure of regularity (the number of repeated computational steps) and thus directly connected to Kolmongorov's complexity (Kolmogorov, 1998).

In computer science one is often interested in quantifying algorithms runtime boundaries, usually worst-case runtime performance pertaining to the input. Thus, calculating computational complexity is the process of characterizing how the running time of an algorithm grows with the input size (x) (Sanjeev and Boaz, 2009). Therefore, let $f(x)$ be a function of x (where x is the input size), that determines the number of computational steps (worst case scenario) required to complete some calculation. In computer science, the worst-case complexity (usually denoted in asymptotic notation using O) measures the resources (e.g. running time, memory) an algorithm requires in the worst-case scenario, thus it gives an upper bound on the resources required by the algorithm. Hence, the *upper bound* of an algorithm is denoted by $O(g(x))$, where $g(x)$ is a function of the input size $x \in \mathbb{Z}^+$ such that $f(x)$ grows no faster than $g(x)$, which means that there exists a positive constant c where:

$$|f(x)| \leq c \times |g(x)| \tag{A.3}$$

for all sufficiently large values of x. In practice $g(x)$ is formed by ignoring all constant factors and lower-order terms (in case $f(x)$ is a polynomial function), preserving only the highest-order term. However, sometimes this practice can be bent in a situation when a particular improvement in algorithm design is emphasized which depends on a constant factors and/or a lower order term.

Since computational complexity analyses in this thesis do not involve any other boundary type calculations aside from O, I restrict my introduction to computational complexity (asymptotic) analysis, to the above stated and encourage readers to seek more information on these and other related definitions in the appropriate (relevant) literature: e.g. Cormen et al. (2001) or Sanjeev and Boaz (2009).

A.2.3 Distance Computation

String searching has always been one of the most intriguing problems in computer science. It can generally be divided into exact and non-exact match finding (Gusfield, 1997). The former is usually associated to more traditional, well-defined data repositories which enforce strict data archiving strategies, whereas the latter is prevalent in modern, community-based information resources like social networks (FaceBook: www.facebook.com, LinkedIn: www.linkedin.com) and "world wide web" in general (Google: www.google.com), Bing: www.bing.com). Because of that, what is considered to be a well-matched, query-subject string pair using a traditional exact retrieval strategy, is often different from what such a pair ought to be in case of a non-exact search.

A non-exact match retrieval operation is usually based on quantifying the query-subject string pair proximity. Here, proximity I defined as (dis-)similarity between a query string and a string (subject) stored in a database. In order to quantify this (di-)similarity, a distance measure is required. In mathematical terms the intuitive notions of distance measures are formalized as *metrics*, and they are defined with respect to a given reference point, a metric space (D,d) (where D is a domain and $d : D \times D \to \mathbb{R}$ is a function). Thus a formal definition of a query search problem as formulated in Zezula et al. (2006) is:

Problem 2 *Let D be a domain and $d : D \times D \to \mathbb{R}$ a distance on D over a metric space (D,d). Given a set $X \subseteq D$ of n elements ($|X| = n$), structure the data in such way so that search queries can be efficiently executed.*

The stated problem broadly depicts a search as a process of obtaining objects from a collection (database), the order of which is defined through the distance function. Thus search can be seen as a type of ordering, ranking of objects form the database with respect to a given query data point.

A distance function operates on a metric space assigning a number to a pair of elements X and Y according to the following rules. Let X, Y and Z be elements belonging to a given

set, then:

$$d(X,Y) = 0 \Longleftrightarrow X = Y \qquad \forall X,Y \in D \qquad \text{(identity)}$$
$$d(X,Y) = d(Y,X) \qquad \forall X,Y \in D \qquad \text{(symmetry)}$$
$$d(X,Y) \leq d(X,Z) + d(Z,Y) \qquad \forall X,Y,Z \in D \qquad \text{(triangular inequality)}$$

Combing identity and triangular inequality forth rule is obtained:

$$d(X,Y) \geq 0 \qquad \forall X,Y \in D \quad \text{(nonnegativity)}$$

From nonnegativity it follows that $\forall X,Y \in D$ if $X \neq Y$ then $d(X,Y) > 0$, therefore if a distance function does not satisfy this property it is not a proper distance function and thus accordingly called *pseudo metric* (Zezula et al., 2006).

A.2.3.1 Edit Distance

Different kind of distance functions can be defined on different metric spaces. When talking about string searching, it is rather convenient to express the distance as an integer reflecting the number of edit operations ($\varepsilon = \{$*insert, delete, substitute* $\}$) which need to be preformed in order to convert one string into another. Edit distance ($d_e(q,s)$) is thus defined as a minimum number of such operations required for transforming a source sequence q into a target sequence s such that both letters $q[i], s[j] \in \Sigma$ for $i \in [1,|q|]$ and $j \in [1,|s|]$. Formally, according to Zezula et al. (2006) I define each edit operation ($w \in \varepsilon$) in the following way:

Definition 1 *Let ω be a string of size $|\omega|$ and let $\omega[i] \in \Sigma$ be a letter. Furthermore let letter $x \in \Sigma$, then:*

1. **insert** *operation is defined as inserting a letter x into a string ω at position i:*
   $ins(\omega,i,x) = \omega[1]\omega[2]...\omega[i-1]x,\omega[i]...\omega[|\omega|]$

2. **delete** *operation is defined as removing a letter at position i from ω :*
   $del(\omega,i) = \omega[1]\omega[2]...\omega[i-1]\omega[i+1]...\omega[|\omega|]$

3. **substitute** *operation is defined as replacing a letter at position i in* ω *with x :*
   $sub(\omega, i, x) = \omega[1]\omega[2], ..., \omega[i-1]x\omega[i+1]...\omega[|\omega|]$

As an example consider strings $q = ab$ and $s = aab$. In order to transform q into s one of the following set of edit operations needs to applied (Figure A.1):

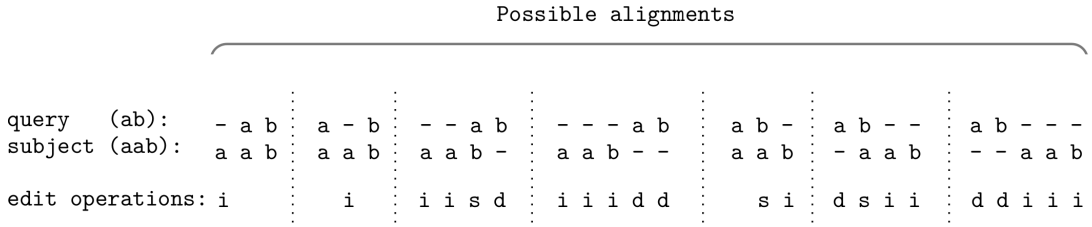


Fig. A.1 Example of edit operations: d - delete, i - insert, s - substitute

The number and the type of operations used, quantifies the distance between strings, such that each operation is penalized or reworded depending on the underlying model. Assuming that each edit operation has a cost value of 1 then each transformation in Figure A.1 has a particular distance value associated to it. Therefore, returning to the example, from left to right, transformational distances between q and s, given a different sequence of transformational events, are: 1, 1, 4, 5, 2, 4, 5.

The edit distance calculated using binary weights for cost values ($cost(w) = \{0,1\}$ *for* $w \in \varepsilon$) is called the unit-cost standard edit distance (Navarro, 2001; Shapira and Storer, 2007). Moreover, a generic version of such distance where real-number-value weights are assigned to each operation ($cost(w) \in \mathbb{R}$) instead of binary values, need not to be symmetric and therefore it is incorrect to define it as metric.

Since a unit-cost edit distance is a restricted version of its generic variant, I generalize the concept and formally define the edit distance as:

Definition 2 *Let q and s be two strings and let* $w \in \varepsilon$ *be an edit operation with an associated cost function* $cost(w)$. *Let* $W \in \varepsilon^*$ *define a set of edit operations with an associated cost function cost(W) such that* $cost(W) = \sum_i cost(w_i)$. *Let* $\mathscr{E} : \Sigma \times \varepsilon \rightarrow \Sigma$ *be the edit function.*

Then the edit distance (d_e) is defined as:

$$d_e(q,s) = min\{cost(W) : \mathscr{E}(q,W) = s\}$$

While it is relatively easy to find a set of edit operations (W) to transform one string into another, the challenge lies in finding a set which minimizes the overall value. A naive procedure would be to compute, as in figure A.1, all possible cases of edit operations and select the one with the minimum total cost value. Clearly this would be computationally unfeasible, even for short sequences. Therefore, a practical approach approximating the solution to this problem, is a strategy called dynamic programming (Bellman, 1957). Dynamic programming is a method for solving complex algorithmic problems by splitting them into a set of smaller, simpler ones. There are many different algorithms that make use of dynamic programming (Altschul et al., 1997; Bellman, 1957; Gusfield, 1997; Needleman and Wunsch, 1970), however, the emphasis here is on the algorithms designed for pairwise sequence alignment.

A.2.3.2 The k-mer Distance

The k-mer distance (d_k) between two strings q and s, such that $\forall i, j \in [1, \max(|q|, |s|)]$, $q[i], s[j] \in \Sigma$ and $\forall k \in [1, \max(|q|, |s|)]$, where c is an occurrence count is defined as (Ukkonen, 1992):

$$d_k(q,s) = \sum_{x \in |\Sigma|^k} |c^q_{k,x} - c^s_{k,x}|$$

where $|\cdot|$ notation in this particular case refers to the absolute value.

Unlike the edit distance where positional similarity between strings is emphasized, in (d_k) positional effects are restricted to a limited number of characters. $k-$mer distances are based on a content of a $k-$size substring and therefore less sensitive to large scale inversions, deletions and/or insertions. For example, if a block of characters is moved from one position in a string to another, due to a large number of insertion/deletion events, the edit distance will report a high value, although, from a certain perspective, the string has not changed significantly. $k-$mer distance, on the other hand, will remain, nearly identical and therefore

can be perceived as a metric of choice in such cases.

As an example consider a pair of strings $q = \texttt{ATTAT}$ and $s = \texttt{ATTAATT}$. Given an optimal alignment, an edit distance $d_e(q,s)$ between the two:

```
A T T A - T -
A T T A A T T
```

$d_e(q,s) = 2$ (2 insertions), whereas for $k = 3$, $d_k = |0-1|_{AAT} + |1-2|_{ATT} + |0-1|_{TAA} + |1-0|_{TAT} + |1-1|_{TTA} = 4$.

When executing search queries, the concept of distance (or a metric based on it) is often used to find "closely related" objects, that is, objects with the minimum pairwise distance to each other. In order to evaluate the quality of a search, one preforms sensitivity and specificity analysis using the obtained results (Fawcett, 2006). Sensitivity (frequency of true positive cases) of retrieved results based on the $k-$mer distance function depend on the value of k. For example, if $k = 1$, the distance $d_k(q,s)$ is computed as the occurrence count difference of single characters between two strings. Therefore, given strings q and s such that $|q| > 0$ and $|s| > 0$, under the uniform distribution assumption (of characters) in a string, the number of possible (q,s) pairs with the same d_k value, can be quite large and thus accordingly, sensitivity very low and specificity (frequency of true negative cases) high. On the other hand, if $k = \min(|q|,|s|)$ the k-mer distance takes its maximum value, with the opposite sensitivity and specificity trends. A more detailed introduction into quality measures (sensitivity and specificity) can be found in Section 2.7.

To be able to achieve a pre-specified balance between the two, it is necessary to estimate the appropriate value of k (the size of a k-mer). Therefore, I build upon Ukkonen's results (Ukkonen, 1992) on k-mer distance and derive a simple expression for estimation the length of a k-mer based on the expected number of k-mer occurrences and string size.

Let ω be a string where the characters are drawn uniformly at random from Σ. Then the

probability of finding a $k-$mer (κ_i) at some fixed position $i \in [1, |\omega| - k + 1]$ is:

$$P(\kappa) = \frac{1}{|\Sigma|^k} \tag{A.4}$$

and therefore an expected number of occurrences of κ is:

$$c_{k,i}^{\kappa} = \frac{|\omega| - k + 1}{|\Sigma|^k} \tag{A.5}$$

The equation holds if the expected number of occurrences $e(c_k^{\kappa}) = c_{k,i}^{\kappa}$, under the assumption that $1/e(c_k^{\kappa}) \ll |\Sigma|^k$ and $k/e(c_k^{\kappa}) \ll |\omega|/e(c_k^{\kappa})$. In such case both terms are lower order terms and thus can be neglected. Therefore, estimating the value of k, given the number of occurrences one expects to observe ($e(c_k^{\kappa})$), can easily be calculated using the following expression:

$$k \geq \left\lceil \log_{|\Sigma|}\left(\frac{|\omega|}{e(c_k^{\kappa})}\right) \right\rceil \tag{A.6}$$

The reason why the ceiling function is used in the expression is because the value of a k-mer size is an integer value.

Of course, whether the computed size of a $k-$mer is an appropriate one, strongly depends upon a primary question and sensitivity (specificity) one hopes to achieve, as well as on the algorithmic and technical limitations (often RAM restrictions).

A.3 New Distance Metric and Computation Strategies

A.3.1 Fast heuristic for computing the top most similar subjects to a given query string

Similarity searching on many-string collection databases is a central problem in bioinformatics and computational biology: e.g: Albá and Castresana (2007); Altschul et al. (1990, 1997); Capra et al. (2012); Edgar (2010), etc., where new, low cost technologies like high-

throughput next-generation sequencing, are rapidly increasing the amount of available information (Benson et al., 2007; Flicek et al., 2014). Therefore, faster, more efficient search strategies are required in order to cope with the growing problem.

The key idea behind the algorithm described in this section is based on the observation:

Observation 2 *Let k be a size of a substring, then two strings are more similar to each other if they share more k-sized substrings (k−mers) (Ukkonen, 1992)*

However, unlike in a conventional approach proposed by Ukkonen (1992), where variation in *k*-mer frequencies is used as a measure of distance, the strategy I propose here is based on computing the number of shared *k-mer types* (κ), with **same or similar in value** *k*-mer frequencies. Therefore, all features associated with *k*-mer frequency-based similarity computations are also valid for my *k*-mer type similarity search strategy. Moreover, I show in section A.3.1.2 that *k*-mer type distance (d_{kt}), upon which the entire computation is based, is in fact a lower bound on Ukkonen's *k*-mer distance and thus less accurate, however, allows for much faster distance computation.

Once the computation has been completed the top scoring database candidate strings are associated to each query string. This is a restricted version of a problem called "the top-X similar string searching problem" (Zezula et al., 2006) defined as:

Problem 3 *Let Q be a set of input strings and S be a set of database strings and let X be a number. Then, for each input sting find ($q \in Q$) find the X number of the closest (most similar) strings from the database ($s \in S$).*

A number of solutions has been proposed in order to efficiently solve this problem (Yang et al., 2010), however, none of which are based on the heuristic approach presented here.

The first part of this subsection deals with HD-index construction procedure (HD stands for High Dimensional) followed by a short description of the *k-mer type distance (d_{kt})* and the similarity function (*JaccScore*) based on it. Next, I describe the strategy utilizing d_{kt} and JaccScore for preforming the similarity search.

A.3.1.1 HD-index Construction Algorithm

The construction algorithm presented in this section is based on some well known solutions and data structures. However, it presents a crucial step in the overall computation, thus I feel it is necessary to be explained in detail. The first step in the index construction is to factor out (compute l-size substrings) the string ($s \in S$) into a set of overlapping substrings of equal size (l) starting from $s[1]$ and moving towards $s[|s|]$, such that the suffix of the left substring ($s[i..i+l-1]$) overlaps the prefix of its right neighbour ($s[j..j+l-1]$) with the overlap region equal to $\lfloor l/2 \rfloor$ ($j = i + \lfloor l/2 \rfloor - 1$). Each substring is further divided into smaller overlapping k-mers by sliding a k size window across the string, shifting it one character in each iteration to the right. An example of this process is illustrated in figure A.2

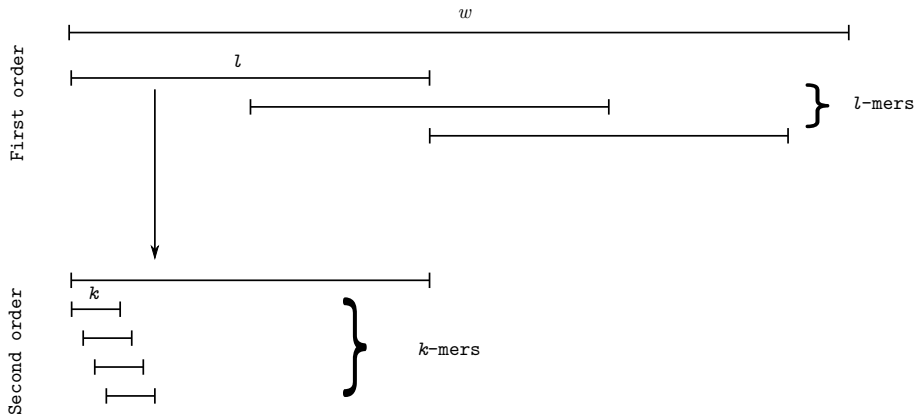


Fig. A.2 An example of first and second order string (ω) factorization process. The first factorization is done by shifting the l size character window alongside the string (from left to right) each time skipping $\lfloor \frac{l}{2} \rfloor$ characters. The second order factorization is done in a similar way, however sliding to the right of each l-mer substring by a single character.

Usually once k-mers are computed, an inverted list (Baeza-Yates and Ribeiro-Neto, 1999; Zobel et al., 1998) associating each k-mer type to a string identifier or even a pair *(string identifier, position in s)* is generated. However, the novel strategy I propose at this point relies on calculating the frequency of each computed k-mer and creating an inverted list with a two dimensional key. To achieve this the first step is to calculate all possible k-mer types given an alphabet Σ. From Section A.2.1 it follows that the number of possible κ can be computed using the information about the size of the alphabet and the length of a

k-mer as $|\Sigma|^k$. Moreover, by rearranging those k-mers according to some pre-specified order (lexicographic order) a k-mer vector is defined as in equation A.1.

Note that each κ in the vector has its unique position which is sometimes referred to as rank (R). Rank is defined as a injective mapping from the "k-mer space" to integer space thus allowing a reduction of a k characters to a single number (Abouelhoda et al., 2002). For example let $\Sigma = \{a,b,c\}$ and the size of a k-mer be $k = 2$, then the maximum number of possible k-mers is $|\Sigma|^k = 3^2 = 9$ and $\vec{\kappa} = \{aa,ab,ac,ba,bb,bc,ca,cb,cc\}$. Rank of each 2-mer is defined by its position in $\vec{\kappa}$: $R(aa) = 1, R(ab) = 2, ...R(cc) = 9$.
Next I compute the maximum number of times each k-mer can occur in a given string ($|\omega| - k + 1$). As an example consider the string $\omega = aaaaaaaaa$. The size of that string is 9. Given the size of a $k-$mer is 3, then the maximum number of times a k-mer $\kappa = aaa$ appears in a string is 7 which is equal to the total number of k-mers ω contains.

Using the two values described above (the size of the k-mer vector and the maximum number of times a k-mer can appear in any give string), it is possible to construct a two dimensional matrix (I here refer to as the *key* matrix) onto which any possible string ω of size $|\omega| \leq \varphi$ (where φ is some pre-specified integer value) can be mapped to. The mapping is done by locating the position in a matrix using the rank value of a k-mer and its frequency in ω. For example let $\omega = aabaa$ and $\Sigma = \{a,b,c\}$ with $k = 2$, then ω can be mapped onto a key matrix as depicted in Figure A.3:

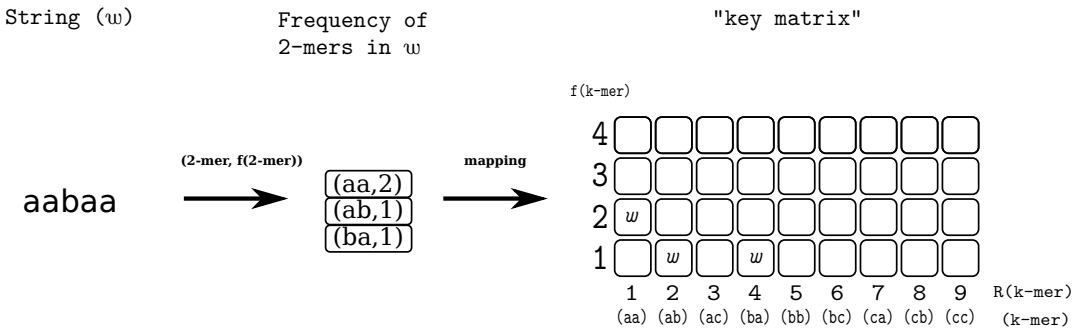


Fig. A.3 A general framework for mapping a string onto two dimensional matrix (key matrix)

However, by generalizing the principle to any set of strings Ω where $|\Omega| > 1$, collisions are expected to occur. That is, two strings $\omega_i, \omega_j \in \Omega$, can have an identical k-mer of the

same frequencies. In such case, mapping ω_i, ω_j onto a "key matrix" will assign two strings to a same location causing a "collision". In order to resolve this situation, strings are assigned to a list, each having a rank and a k-mer frequency value serving as an access key.

Therefore, once the initial l-mer is divided into smaller overlapping k-mers the frequency of each k-mer is calculated and the l-mer based on k-mer frequency and rank value is mapped onto a "key matrix" as described. This process is repeated for each l-mer in each string $s \in S$, ultimately constructing the so-called HD-index. The entire process is briefly summarized in Algorithm 1:

Algorithm 1 *Let S be a set of strings and $s \in S$ be a string from that set. Let $s_i^l = s[i..i+l-1]$ be a substring of s of size l. Let $\mathscr{K}'$ be the set of k-mers from s^l and Σ is the alphabet. HD-index computation is executed in the following way:*

- *Compute the number of possible k-mer types (κ) and their maximum frequencies within a string of size l.*
- *Allocate the space of the "key matrix" and a list associated to each position in it.*

for each *string $s \in S$* ***do***
  Create a set of l size overlapping substrings (s^l) such that two consecutive substrings overlap each other by $\lfloor l/2 \rfloor$.
  for each *substring $s^l \in s$* ***do***
    Compute a set of k-mers ($\mathscr{K}'$) such that each k-mer is shifted from its left neighbour by one
    for each *k-mers $\in \mathscr{K}'$* ***do***
      Compute its occurrence count (frequency) in s^l
      Using the occurrence count (frequency) and rank value of a k-mer, add string to each list, having $(R(\kappa), f(\kappa))$ pair as an access key in the "key matrix".
    end for
  end for
end for

Graphical representation illustrating the HD-index and its computation is depicted in Figure A.4.

The size of a substring (l) in the above description can vary in size, however according to results reported in previous analyses (Shen et al., 2005; Trifonov and Berezovsky, 2003; Xu and Nussinov, 1998) it is set to 300 characters since this length represents approximate double average size of a protein domain. This is an important factor from a biological perspective, especially given that homologues relations are in fact inferred on a level of a protein domain rather then the entire sequence (Altschul et al., 1990). Another parameter mentioned and used in HD-index construction is the size of a k-mer. Its value is computed using the equation A.6 by setting the size of $|\omega| = l$ and fixing the expected number of occurrences to 1 in order to maximize the sensitivity and reduce the number of expected collisions. Obtained calculation implies that the value of k should be set to a value larger than:

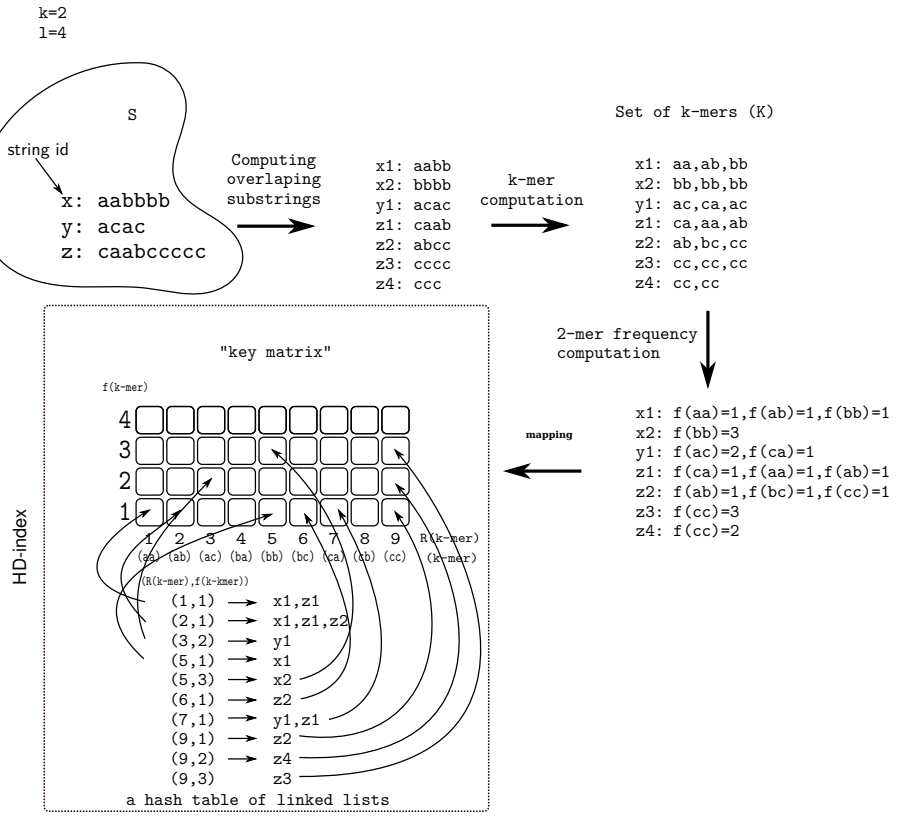


Fig. A.4 Simplified example of HD-index construction algorithm. k-mer size is $k = 2$, substring length is $l = 4$ and the alphabet is $\{a,b,c\}$

$$k \geq \left\lceil \log_{20}(\frac{300}{1}) \right\rceil \geq \lceil 1.903 \rceil \geq 2$$

Since k directly affects the size of memory footprint required for storing the "key matrix" (recall that the number of possible k-mers equals to $|\Sigma|^k$), it was necessary to calculate the most feasible, both technically acceptable and biologically meaningful value higher than 2. "Technically acceptable" implies that the application should be able to run on a 32-bit machine with ≈4GB (4000 MB) of RAM, whereas biologically meaningful refers to homology detection between distantly related sequences (e-value $= 10^{-3}$). Note that at such threshold identity tends to be substituted with positional similarity (Altschul et al., 1997; Li and Godzik, 2006), therefore homologous sequences will have less identical k-mers if the size of a k-mer increases.

Simple calculation (Table A.1) reveals what technically acceptable values for k are.

Table A.1 Memory footprint for different k-mer sizes. The required memory is computed as: $|\Sigma|^k \times (l-k+1) \times 32/8$ where 32 is the size of an integer value in bits, 8 is a size of a byte, $|\Sigma| = 20$ and $l = 300$

k	Memory	Acceptable
2	0.45(KB)	Yes
3	9 (MB)	Yes
4	180 (MB)	Yes
5	3.6 (GB)	Yes
6	72 (GB)	No

Using $k = 5$ value, the amount of occupied memory is still reachable within the above set framework, however as stated, high identity affects similarity detection in distantly related sequences therefore in accord to BLAST, sliding window size (window $= 3$) and the calculation preformed above (the lowest number higher than 2) the k-mer size was adjusted and accordingly set to 3.

A.3.1.2 A New Distance Metric and Similarity Measure

Once HD-index has been created, it is necessary to develop a procedure for retrieving the information stored in it. Strategy behind similarity searching used here relies on a count filtering approach introduced by Ukkonen (1992). The key observation upon which Ukkonen bases his strategy is that strings within a smaller edit distance (d_e) between each other share more $k-$mers and therefore are closer to each other. In order to quantify this observation, Ukkonen introduces a new k-mer distance measure (described in section A.2.3.2), which estimates the proximity between a pair of strings (q and s) by counting the number of shared k-mers. Moreover, the distance was proved to be a lower bound on edit distance with the relation between the two defined as (Ukkonen, 1992):

$$d_e(q,s) \geq \frac{d_k(q,s)}{2k} \tag{A.7}$$

This relation depends on a simple observation that each edit operation can cause only a limited number of changes in k-mer frequencies ($f(\kappa)$) in a given string. I formalize the relation in Lemma 1 :

Lemma 1 *Let ω be a string from Σ (the alphabet) and $\mathscr{K}'$ be a set of $k-$mers occurring in ω. Let κ and κ' be two k-mer types such that $\kappa \neq \kappa'$, then each edit operation on ω at position $i \in [1..|omega|]$ affects the frequency of each k-mer type (κ) overlapping the position (i) in one of the three following ways:*

1. $f(\kappa) = f(\kappa) + 1$
2. $f(\kappa) = f(\kappa) - 1$
3. $f(\kappa) = f(\kappa) + 1 \wedge f(\kappa') = f(\kappa') - 1$

Proof All three cases correspond to three different edit operations: case 1 corresponds to insertion, case 2 to deletion and case 3 to substitution. □

Thus according to Lemma 1: smaller the number of edit operations required to convert

one string into another, smaller the frequency distance between them. Therefore, smaller the number of changes in $f(\kappa)$ implies that two strings are closer to each other. The heuristic strategy behind my approach relies on another observation derived from that fact:

Observation 3 *The higher number of k-mers having the same or similar frequencies ($f(\kappa)$), the closer two strings are.*

Based on this observation, first I derive a new distance measure I call the $k-$**mer type distance** capturing the feature stated in observation 3.

Definition 3 *Let q,s be two strings over alphabet Σ and $k \in [1, min(|q|,|s|)]$ the size of a k-mer. I define the $k-$***mer type distance** *(d_{kt}) as the number of k-mer types (κ) satisfying the condition:*

$$|f_q(\kappa) - f_s(\kappa)| > r$$

for $r \leq min(|q|,|s|) - k + 1$.

In order to show the connection between the *$k-$mer type distance* and other closely related distances like *k*-mer and edit distance, first I prove d_{kt} to be a lower bound on d_e:

Lemma 2 *Let $d_e(q,s)$ be a unit-cost edit distance and $d_{kt}(q,s)$ the k-mer type distance. Than for $k \in [1, min(|q|,|s|)]$:*

$$d_e(q,s) \geq \frac{d_{kt}(q,s)}{k}$$

Proof Assume a unit-cost edit distance. Given a string s, each edit operation affects k overlapping $k-$mers such that it either affects the occurrence count of already existing $k-$mer types or it crests new ones. If already existing types are affected then there is no change in the number of $k-$mer types. On the other hand, if an edit operation generates new $k-$mer types then it increases (or decreases) the count of k number of $k-$mer types and decreases (or increases) at least one. Therefore, with each edit operation the $k-$mer type distance is

increased by at most $k+1$ times or remains unaffected. Thus $d_e(q,s) \geq d_{\mathrm{kt}}(q,s)/(k+1)$ and therefore $d_e(q,s) \geq d_{\mathrm{kt}}(q,s)/(k)$.

□

Next I establish its relation with respect to the $k-$mer distance (Ukkonen, 1992):

Lemma 3 *Let $d_k(q,s)$ be the $k-$mer distance and $d_{kt}(q,s)$ the $k-$mer type distance. Then for $k \in [1, min(|q|,|s|)]$:*

$$\frac{d_k(q,s)}{2k} \geq \frac{d_{kt}(q,s)}{k}$$

Proof Assume $d_k(q,s)$ is the $k-$mer distance, then a single change in k-mer frequency either affects an existing k-mer type or a non existing one. If the existing one is affected no change in the number of k-mer types can be observed. On the other hand:

Case 1: If the k-mer type occurs only once, decrease in its frequency will eliminate it from the set of occurring k-mer types thus increasing the k-mer type distance by one.

Case 2: If the frequency of a k-mer type is zero and the change increases it by one then the k-mer type distance is accordingly increased by one.

Case 3: If the k-mer type occurs only once, and decrease in its frequency eliminates it from the set of occurring k-mer types and at the same time increases the frequency of another k-mer type with the initial frequency zero the total change in k-mer type distance is two.

Therefore a change in k-mer frequency affects the k-mer type distance by the maximum value of two or it remains unaffected. Thus:

$$d_k(q,s) \geq 2d_{kt}(q,s)$$

By dividing the inequality with $2k$ it follows:

$$\frac{d_k(q,s)}{2k} \geq \frac{d_{kt}(q,s)}{k}.$$

□

Having proved d_{kt} distance is a lower bound on both d_e and d_k, and knowing from relation A.7 the relative position between d_e and d_k I conclude:

Theorem 1 *Let d_e be an edit distance and d_k a k-mer distance. Given the k-mer type distance d_{kt}, then:*

$$d_e(q,s) \geq \frac{d_k(q,s)}{2k} \geq \frac{d_{kt}(q,s)}{k}$$

Proof From relation A.7 $d_e(q,s) \geq \frac{d_k(q,s)}{2k}$, from Lemma 2, $d_e(q,s) \geq \frac{d_{kt}(q,s)}{k}$ and from Lemma 3 $\frac{d_k(q,s)}{2k} \geq \frac{d_{kt}(q,s)}{k}$, therefore:

$$d_e(q,s) \geq \frac{d_k(q,s)}{2k} \geq \frac{d_{kt}(q,s)}{k}$$

□

Jaccard distance metric (JD) is a measure of dissimilarity between sets (Jaccard, 1901). It is defied as the difference between the sizes of the union and intersection of sets divided by the size of their union. More formally, given two sets X and Y, Jaccard distance is defined as:

$$JD(X,Y) = \frac{|X \cup Y| - |X \cap Y|}{|X \cup Y|}$$

Since Jaccard distance (JD) is a metric on sets (Zezula et al., 2006), similarity measure derived from it, is complementary to the distance and therefore can be obtained by subtracting JD from 1, thus:

$$JS(X,Y) = 1 - \frac{|X \cup Y| - |X \cap Y|}{|X \cup Y|} = \frac{|X \cap Y|}{|X \cup Y|} \tag{A.8}$$

Jaccard similarity measure is also known as a Jaccard coefficient and here is used as a primary motivation behind the distance measure called JaccScore. JaccScore is essentially normalized version of Jaccard coefficient based on a k-mer type distance (d_{kt}). The normalization is achieved by multiplying the Jaccard similarity measure with the size of both query ($|q|$) and subject ($|s|$) string thus:

$$JaccScore(q,s) = |q| \times |s| \times JS(q,s) \tag{A.9}$$

According to the definition, JS equals the number of k-mer types shared between a query (q) and a subject (s) string divided by the total number of k-mer types of q and s. Thus, let $|q_{kt}|$ be the total number of k-mer types in q and $|s_{kt}|$ the total number of k-mer types in s, moreover let $|qs_{kt}|$ be the number of k-mer types shared between q and s, then:

$$JaccScore(q,s) = |q| \times |s| \times \frac{|qs_{kt}|}{|q_{kt}| + |s_{kt}| - |qs_{kt}|} \tag{A.10}$$

From the k-mer type definition, according to which d_{kt} the number of dissimilar k-mer types between two strings, it follows that $d_{kt}(q,s) = |q_{kt}| + |s_{kt}| - |qs_{kt}|$ and the total distance is $d_{kt}^{tot}(q,s) = |q_{kt}| + |s_{kt}|$. Therefore, JaccScore can be expressed in terms of d_{kt} as:

$$JaccScore(q,s) = |q| \times |s| \times \frac{d_{kt}^{tot}(q,s) - d_{kt}(q,s)}{d_{kt}(q,s)} \tag{A.11}$$

As an example of this new metric consider a following set of strings: $q = aabba$, $s_1 = accc$, $s_2 = aabab$, $s_3 = acaca$, $s_4 = acccb$. Clearly, the closest (most similar) to q is string s_2 since $d_e(q,s_2) = 2$ is the smallest distance value of all (under *cost(ins) = cost(del) = cost(sub) = 1*). In case of the k-mer distance (given that $k = 2$) for each query subject pair: $d_k(q,s_1) = 8$, $d_k(q,s_2) = 2$, $d_k(q,s_3) = 8$, $d_k(q,s_4) = 8$. It follows that the smallest distance is between q and s_2 which is in agreement with the result obtained by computing d_e. Next given the k-mer type distance for each given (q,s) yields the following result, $d_{kt}(q,s_1) = 6$,$d_{kt}(q,s_2) = 1$,$d_{kt}(q,s_3) = 6$,$d_{kt}(q,s_4) = 7$ again rendering (q,s_2) as the closest

pair. Evidently in all three cases the closest pair has been properly identified and is in accord with Theorem 1:

$$d_e(q,s_2) \geq \frac{d_k(q,s_2)}{2k} \geq \frac{d_{kt}(q,s_2)}{k} = 2 \geq \frac{1}{2} \geq \frac{1}{2}$$

However, the real gain and an advantage of d_{kt} metric over the other two becomes evident when used together with the index described in the previous section. Since k-mer types, with the same or similar frequencies can be compute using very few computational steps, one can easily use them in distance calculations, avoiding any additional "re-computations" as those done in a classic k-mer frequency distance calculations.

Once the d_{kt} value has been obtained Jaccard score is computed as:

$$JaccScore = |q| \times |s| \times \frac{d_{kt}^{tot}(q,s) - d_{kt}(q,s)}{d_{kt}(q,s)} = 4 \times 4 \times \frac{(4+3)-1}{1} = 96$$

A.3.2 HD-index Based Similarity Search Algorithm

Let S be a set of strings indexed using HD-index construction algorithm described in section A.3.1.1 and let $s \in S$ be a subject string. Let Q be a set to strings and let $q \in Q$ be a query string. HD-index based similarity search procedure begins in the same way as the HD-index construction process. Each query string $q \in Q$ is divided in a same way as are subject strings in section A.3.1.1, by splitting a it into a set of overlapping substring (q^l) of size $l = 300$. Also the length of the overlap needs to be $\lfloor l/2 \rfloor = 150$. Once a set of substrings $q^l \in q$ is computed, each substring is then further divided into a set of smaller k-mers ($\kappa \in \mathscr{K}''$) using the same k value and computation strategy as in the construction phase. For each $\kappa \in q^l$ an occurrence count vector (frequency) is calculated. Based on the frequency and the rank value of a k-mer $(R(\kappa), f(\kappa))$, a unique position in "key matrix" is located. The list of subject string

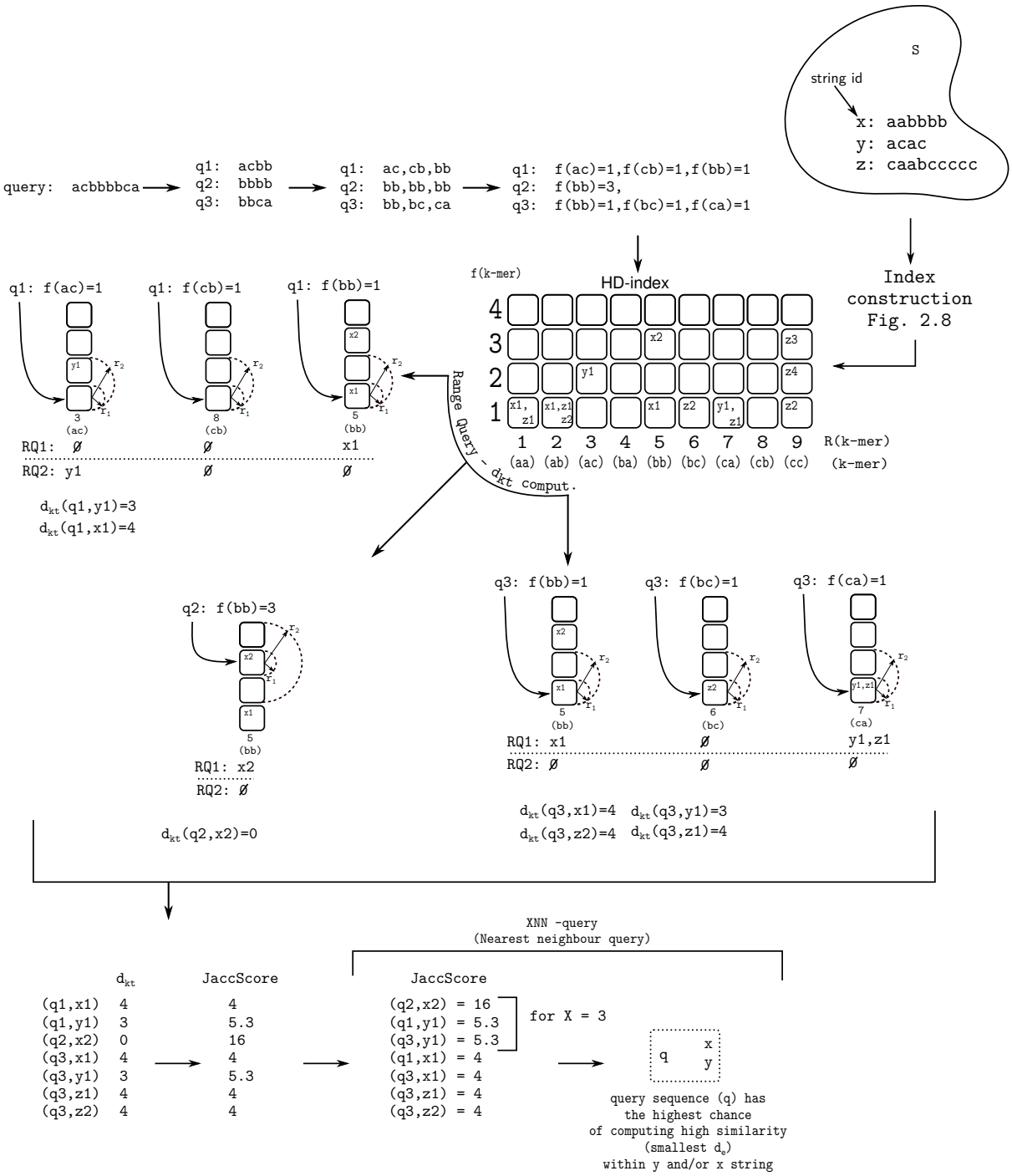


Fig. A.5 Simplified example of HD-index construction and search algorithm. k-mer size is 2, substring length is 4, range query radius is 2, X in nearest neighbour query is 3 and the alphabet is $\{a,b,c\}$

identifiers is then retrieved. As concluded in the previous section, the small number of edit operations will cause small variations in k-mer type distance, thus each subject string with k-mer frequencies similar to the one associated to a k-mer in a query string, is also informative and thus retrieved. The retrieval process is a linear time operation proportional to the number of subject identifiers since "the key matrix" is essentially an array and k-mer frequencies are pre-ordered. Therefore, subject identifiers with a k-mer sharing similar frequencies can be easily retrieved by moving "up" and "down" with respect to the y-axis ($f(\kappa)$) as depicted in Figure A.5. This "shifting" procedure is known as the *one-dimensional range query* (Zezula et al., 2006) and is used to collect all objects within a certain (pre-specified) range. For a more "hands on" illustration of the entire process, consider the example depicted in figure A.5.

After computing JaccScore for each pair of strings, based the JaccScore value, I implement a simple *insertion sort* algorithm to identify the set of closest (most similar) subject strings (highest JaccScore). This process is called the *nearest neighbour search* query (XNN) (Zezula et al., 2006), designed to identify the top X closest strings to a given query string. (Further information regarding range and nearest neighbour queries (either theoretical or implementation details), can be found in: Knuth (1998),Cormen et al. (2001) Zezula et al. (2006) and He et al. (2011)).

The procedure depicted in figure A.5 illustrates the computation process for one query sequence, however in practice I repeat the process for all $q \in Q$, assigning a small subset of subject sequences form S to each $q \in Q$ with the highest chance for being sufficiently similar with respect to edit distance and the alignment. In Algorithm 2 I summarizes the entire procedure:

Algorithm 2 (HD-search) *Let q be a query string $q \in Q$. Let $q_i^l = q[i..i+l-1]$ be a substring of q of size l and such that it overlaps with its closest neighbour by $\lfloor l/2 \rfloor$. Let s be a subject string $s \in S$. Let $\mathscr{K}'$ be the set of k-mers from q^l where k is the size of each k-mer and Σ is the alphabet. Let $r < |f_q(\kappa) - f_s(\kappa)|$ be a range cutoff where $f_q(\kappa)$ and $f_s(\kappa)$ is the frequency (occurrence count) of a k-mer type present in q and s respectively. Given that $R(\kappa)$ is the rank and $f(\kappa)$ the frequency (occurance count) of κ, HD-index base similarity*

computation is executed as follows:

for each *query string* $q \in Q$ ***do***

Divide q into a set of l size overlapping substrings

for each *substring* $q^l \in q$ ***do***

Divide q^l *into a set of k size overlapping substrings (k-mers) forming a set* $\mathscr{K}'$

for each $k-$*mer type* $\kappa \in \mathscr{K}'$ ***do***

$S_R \leftarrow \emptyset$*{*S_R *is a set of the retrieved subject strings}*

Compute its occurrence count (frequency): $f(\kappa)$

Compute its rank value: $R(\kappa)$

Using $R(\kappa)$ *and* $f(\kappa)$ *information, retrieve all subject substrings from the "key matrix" (*$Get(R(\kappa), f(\kappa))$*) and add them to* S_R*:* $S_R \leftarrow S_R \cup Get(R(\kappa), f(\kappa))$

for $i \in [1, r]$ ***do***

$f'(\kappa) \leftarrow f(\kappa) + i$

Retrieve all subject substrings from the "key matrix" associated to hash key defined by $(R(\kappa), f'(\kappa))$ *and add them to* S_R*:* $S_R \leftarrow S_R \cup Get(R(\kappa), f'(\kappa))$

$f'(\kappa)) \leftarrow f(\kappa)) - i$

Retrieve all subject substrings from the "key matrix" associated to hash key defined by a pair $(R(\kappa), f'(\kappa))$ *and add them to* S_R*:* $S_R \leftarrow S_R \cup Get(R(\kappa), f'(\kappa))$

end for

$J \leftarrow \emptyset$

for each (q, s_R) *pair, where* $s_R \in S_R$ ***do***

Compute $JaccScore(q, s_R)$ *and adda it to J:* $J \leftarrow J \cup JaccScore(q, s_R)$

end for

$TopX \leftarrow \emptyset$

Execute X nearest neighbour (XNN) search query: $TopX \leftarrow XNN(J)$

Report the retrieved results (TopX)

end for

end for

end for

As an end result of the construction and search process, for each query string a small set of candidate subject strings have been identified (TopX) which, most likely, contain a significant similarity region, that is, the smallest lower bound on edit distance (and thus, most likely, the smallest edit distance) to a query string. The software solution utilizing this search strategy together with all the heuristic and optimization techniques implemented is called **HD-Search** and can be found in `PhyloToolKit-XXX` available from:

https://github.com/RobertBakaric

A.3.3 Cladogram: Formal Definitions

Before describing algorithms further, it is my feeling the reader should be familiar with definitions and notations I use through the rest of my sections. In particular, the structure called cladogram is of the highest importance. Therefore:

Given a simple cladogram representation the time reference is defined with respect to different speciation events (lineage splitting events - internal nodes). For example, if the speciation event X precedes the speciation event Y then X is defined as older than Y.

Figure A.6 illustrates a cladogram representing relationships between hypothetical species. In a cladogram edges and nodes are classified into larger structures called *clades* and *lineages*. A lineage is a set of nodes and edges on a path from root to final species (leaf) node. Further, I distinguish between a *query species lineage* and a *branching lineage*. A query species lineage is a lineage that starts at root and ends in the node representing the focal (query) species. On the other hand, branching lineage begins in root but terminates in a leaf node not labelled as a query species. A clade is a set of edges and nodes (except the root) consisting of one ancestral node (internal node) and all its descendants (internal and leaf nodes connected with edges all descending from a given ancestral).

A tree-like structure used for depicting a cladogram implies an internal order between elements, in this case nodes and edges. Each node in a tree (except the root) has a parent, that is, a node preceding another node on a path from root to leaf. Since clades are sets of edges and nodes, the order on nodes implies the order on clades. As in the above case, a

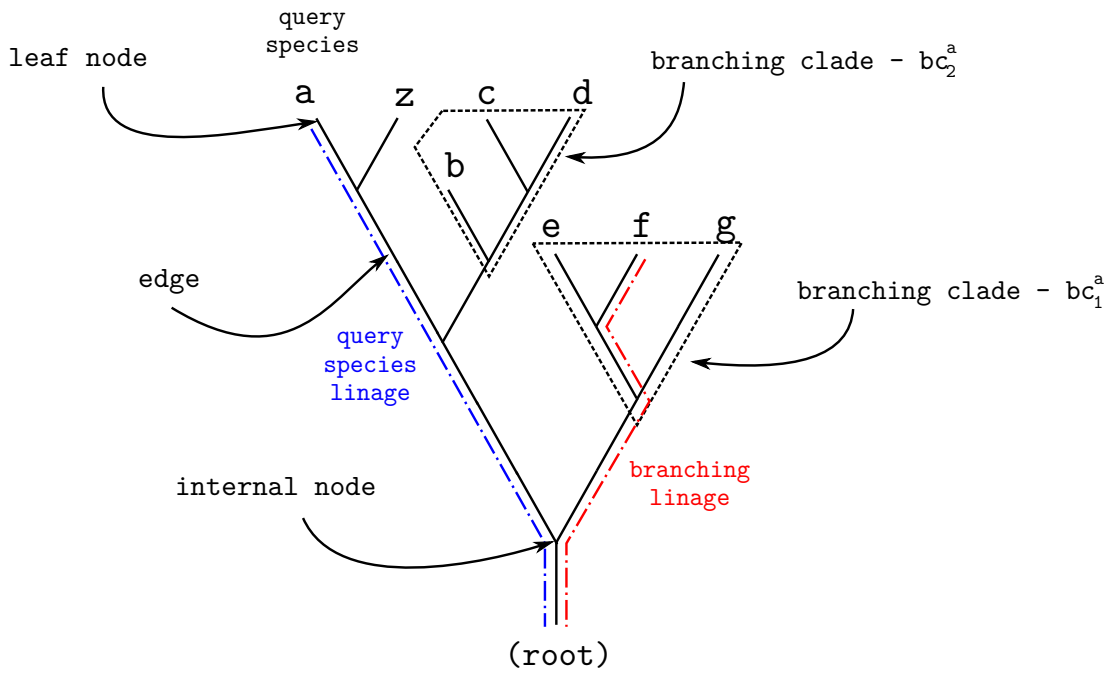


Fig. A.6 Example of a cladogram

clade X is said to be older than clade Y if the origin point of clade Y is a node in clade X, but not its origin point. A *branching clade* (*bc*) is a clade with the origin point being a child node (first descendent) of an internal node on a path from root to query species leaf node. Since nodes in query species lineage are ordered based on the time of their appearance (root - oldest, leaf - youngest), the order reflects on the origin of branching clades. As an example consider the following case: given two branching clades bc_1^a and bc_2^a for $i \in [1..3]$ such that i is an index of a node on a path from root ($i = 1$) to leaf ($a : i = 3$) (Figure A.6), I define a branching clade bc_1^a older in comparison to bc_2^a if its origin node is a descendent of an ancestral node in query species lineage older than the ancestral node, child of which is the origin point of bc_2^a. All the above introduced concepts are depicted in figure A.6.

A.3.4 QPhyloStrat Algorithm

Once the time reference (species phylogeny relations) has been established the similarity search process can begin. The algorithm I describe here is an equivalent in terms of the output and a general concept to the one employed by the phylostratigraphy approach, however,

the difference lies in the underlying computation strategy which improves on efficiency. In order to increase the speed of stratification (gena age computation), the new strategy relies on several accelerators one of which is the HD-index similarity search strategy described previously.

Let S be a set of strings each representing a gene from a genome of an organism whose species relationship is defined according to a given species phylogeny. In the introductory chapter of this thesis I established the connection between similarity, homology and common ancestry (Brief recap: significant similarity between two strings (sequences) implies homology for which the underlying cause is common ancestry). Therefore, a significant similarity between two sequences from distantly related species implies the existence of a sequence in their common ancestor from which the two have derived. Moreover, in accord with Dollo's parsimony model (Farris, 1977), given a query species and a branching clade I give the following observation:

Observation 4 *There is only one most recent common ancestor (MRCA) between a given query species and any species in a branching clade.*

The above implies that computing the origin point of a query species gene by locating a homolog within any species from a branching clade will trace the origin of that gene to exactly one ancestral point (their most recent common ancestor - an internal node in a query species linage). Therefore we have the following:

Observation 5 *It is sufficient to find a single homolog of a query species gene within a branching clade in order to compute its origin.*

To exploit this observation, sequences (strings) in S need to be searched in a specific order, that is, once the similarity search starts examining sequences from the genome of a specific branching clade, the search cannot skip to a sequence from another branching clade until the entire collection of sequences from that branching clade has been processed, or a match found. To achieve this, sequences (strings) in S need to be sorted according to a specified phylogeny and the search order carefully defined.

Unlike in the original phylostratigraphy approach (Domazet-Lošo et al., 2007) where no order on subject sequences exists and the search is done by exhaustively comparing each query species gene with the entire collection of sequences in S, the strategy I propose here starts by selecting a single query species gene and comparing it with sequences from genomes within the oldest branching clade. Since Dollo's parsimony model has been used as the underlying assumption, each gene has only one point of access (gain event), therefore if a significant similarity has been located between a selected query species sequence and a sequence from a genome within the first (oldest) branching clade, this access point has been located and no further search queries are required. Thus, if the uniform distribution of query species genes across different access points (nodes in query species linage) is assumed, the number of search cycles is reduced by the factor of 2 in comparison to the initial strategy where each query species gene is compared to each gene in the subject set, thus decreasing the overall computational time. It is important to note that, if the distribution is not uniform as assumed, then if a larger number of query species genes is assigned to earlier access points (closer to root) the computation speed will increase by a larger factor, whereas if the trend is opposite (larger number of query species genes found their homologs in latter access points closer to the query species leaf node) the computation speed will decrease and will be below 2. Rendering the described strategy as better suited for situations in which a larger fraction of genes have entered the genome early in the evolution.

Since the number of subject sequences in branching clades tends to be large, filtering strategy introduced in Section A.3.1 is applied to each genome (collection of genes - subject sequences) in the respective clade in order to identify X number of sequences that have the highest chance of containing the significant similarity region with respect to a given query species gene. The sequences are then verified using BLAST search to confirm the match. If a match is located, according to the Observation 5 no further search queries need to be executed and the procedure can proceed with the next query species gene. The entire approach speeds up the computation in two ways. First, and the most obvious is the filtering itself. It reduces the number of search queries from the entire set to subject sequences to a small fixed size subset of those having the highest chance for being positively verified by BLAST. Second,

the search process is executed on a collection of subject sequences on a "per genome" base, that is instead of pooling the entire set of subject sequences from a branching clade together, sequences are divided into smaller subsets each corresponding to an individual genome. This way if a hit in the first collection has been identified there is no need to examine the rest of the subject sequences within a given branching clade (according to the Observation 5), thus reducing the overall number of computational steps.

The next technique, which decreases the runtime, is a simple modification of the search strategy utilized by BLAST (Altschul et al., 1990). BLAST is a search algorithm designed to preform search queries on a large collection of sequences, reporting each significant match. Since it is sufficient to find a single significant similarity in order to compute query sequence point of origin, I modified the existing BLAST search strategy to take advantage of this fact. I call the algorithm **QuickBlast**.

QuickBlast, instead for reporting every significant match, terminates the search process as soon as the first match has been detected. Algorithm 3 summarizes the procedure.

Algorithm 3 (QuickBlast) *Let $q \in Q$ be a query string form Q and $s \in S$ a subject string from S. Let $E \in \mathbb{R}$ be a significance cutoff value and let $e(q,s)$ be a statistical significance of the alignment (e-value). Then given a query sequence q and a cut-off value E:*

for each $s \in S$ ***do***
  Align q and s.
  Compute significance of the alignment: $e(q,s)$.
  if *alignment is significant ($e(q,s) \leq E$)* ***then***
    Report a statistically significant match (q,s)
    break
  end if
end for

From a more technical perspective **QuickBlast** differs from regular BLAST in three ways. First, in order to decrease the runtime, BLAST concatenates several query sequences

together into batches which are then passed down to high scoring pair (HSP) computation (Zhang, 2003). **QuickBlast** does not. It avoids this procedure and preforms the alignment computation on each pair of sequences separately. As a result, the modification actually slows down the computation. Second, BLAST preforms the *e-value* computation after all HSP-s have been found and the total effective database size computed. **QuickBlast** precomputes the effective database size and calculates e-values "on-line", that is, as soon as HSP is identified the *e-value* computation takes place. And third, since e-value is computed immediately after HSP is identified, it is possible to use this information to decide if the search process should continue and analyse the next subject sequence or not.

Combined together, all the above mentioned accelerators make up the **QPhyloStrat** algorithm, summarized below:

Algorithm 4 (QPhyloStrat) *Let $q \in Q$ be a query string form Q and $s \in S$ a subject string from S. Let P_a be a set of nodes ($p_i^a \in P_a : i \in [1..|P_a|]$) on a path from root to query species leaf node (a), such that p_1^a is the root and $p_{|P_a|}^a$ is the leaf (a). Furthermore, let $ps_i^a \in PS_a$ be a phylostrata group such that $i \in [1..|P_a|]$. Let S be divided across $|P_a|$ number of equal size (equal in the number of elements) subsets $S_i^{bc^a}$ such that $s \in S_i^{bc^a} \subset S$ is a sequence from an organism of species within bc_i^a (where bc_i^a is a branching clade) with $S_i^{bc^a} \cap S_j^{bc^a} = \emptyset, \forall i, j \in [1..|P_a|] \wedge i \neq j$. Moreover, let each $S_i^{bc^a}$ be divided into a set of even smaller subsets called genomes $G \subset S_i^{bc^a}$ such that $G_t \cap G_h = \emptyset, \forall t, h \in [1..|S_i^{bc^a}|] \wedge t \neq h$. Further, let $E \in \mathbb{R}$ be a significance cut-off value and let HDSearch() be the similarity search algorithm described in section A.3.1.*

Then:

for each** sequence $q \in Q$ **do

  for** $i \in [1..|P_a|]$ **do

    for each** genome $G \in S_i^{bc^a}$ **do

      $X \leftarrow \emptyset$

      Compute candidates (the X number of most similar sequences in G to a given query sequence q): $X \leftarrow HDSearch(q, G)$

      Verify candidates: $s \leftarrow QuickBlast(q, X, E)$

if *a match has been found, that is, s has been defined* ***then***

Assign query species gene q to phylostrata group labelled as ps_i^a*:* $ps_i^a \leftarrow ps_i^a \cup q$ *{ Note that the value of i represents the relative age of query string q (relative with respect to the total number of phylostrata groups on a path from the root to species leaf node (a); lower the i, older the gene) }*

$y \leftarrow true$

break

end if

end for

if $y = true$ ***then***

break

end if

end for

end for

Clearly, the quality of the computation depends on the accuracy of filtering process (HD-index based similarity search). If selected candidates are not those with the highest chance of containing a significant similarity region with respect to a given query string, the origin (gene gain) computation cannot be carried out when required. If that occurs, the gene gain computation is pushed towards a more recent point in evolutionary history (phylostrata group with a higher *i* index) and the process repeated.

Once the age has been assigned to each gene and phylostrata groups defined, the computation of gene family gain events can begin. The process is described in the following section.

A.3.5 PhyloClust Algorithm

Clustering strategy I present in this section is based on pairwise sequence alignment. The general idea is to compute similarities between genes emerging in the same evolutionary epoch from which the number of families and thus the number GFGE's associated with that

period in the evolutionary history is estimated.

The estimation is done by a ranking process derived from the number of pairwise connections (significant similarities) each gene makes. For example let g_i and g_j be two genes such that $i \neq j$ and let g_i be a homologous to g_j, then the number of connections each gene has is equal to one (under the assumption no other gene exists). In the same way these values are computed given a larger set of genes. Based on the number of connections each gene has, genes are then sorted in a descending order according to the number of computed connections. Once sorted, a gene from the top of the list is selected as a cluster representative and each gene with which it has a connection, becomes a member of its cluster. The following cluster is then computed by moving down the list until a gene with no match to any of the previous cluster representatives is detected. This gene then becomes a new cluster representative and each gene with which a match has been established becomes a member of that cluster. Algorithm 5 summarizes the computation process that can applied to any of the existing phylostrata clusters (ps) computed in Algorithm 4, thus the associated phylostrata labels (p_i^a) have been omitted from the pseudocode.

Algorithm 5 *Let $g_i \in ps$ be a gene associated to phylostrata cluster ps such that $i \in [1..|ps|]$. Let C_{g_i} be a set of connections gene g_i makes with $g_j \in ps$, for $j \in [1..|ps|]$ and $i \neq j$. Moreover, let $C_{g_i} \subset C^{ps}$ and let $E \in \mathbb{R}$ be a significance cutoff value. Given that $e(q,s)$ is a function computing a statistical significance of the alignment (e-value), then the GFGE computation executes as follows:*

for $i \in [1..|ps|]$ ***do***
  for $j \in [1..|ps|]$ ***do***
    if $i \neq j$ ***then***
      Align g_i and g_j
      if *significant sequence similarity is found i.e. $e(g_i, g_j) \leq E$* ***then***
        Add gene g_j to cluster C_{g_i}: $C_{g_i} \leftarrow C_{g_i} \cup g_j$
      end if
    end if
  end for

end for

Sort C^{ps} in non-increasing order by the number of elements each $C_{g_i} \in C^{ps}$ contains.

for each $C_{g_i} \in C^{ps}$ *in this order* ***do***

if g_i *has not been assigned to a family in any of previous iterations* ***then***

Set g_i as a family representative and assign all genes from C_{g_i} as family members to g_i

end if

end for

By now, a careful reader has noticed that the above described problem is essentially the problem of finding a minimum vertex cover (where the identified vertices are essentially GFGE's). The decision version of this problem was first described by Karp (1972) and is one of the Karp's 21 NP-complete problems (problems whose solutions are easily verified, but for which no algorithms which efficiently find the solution are believed to exist). Also by carefully analysing the Algorithm 5 it should be obvious that the method used to compute the number of GFGE's is a typical "*list heuristic with static ordering*" (for more information on this heuristic please refer to (Avisa and Imamura, 2007)) and therefore, the computed number of GFGE's using the above approach does not guarantee its optimal value.

A.3.6 PhLoG Algorithm

GFLE computation is directly dependent on the results of GFGE computation. Once GFGE has been computed for each ancestral point (phylostrata) of the underlying phylogeny, the GFLE calculation can take place. The calculation starts by listing all the species included in the analysis, for which phylogeny relations are defined through the underlying phylogeny. Next, for each species GFLE of *GF* is computed by locating the most recent point (ancestral node) on the path from root to species leaf node, shared between a member of that family and selected species. Note that family *GF* in order to be considered in the analysis cannot contain a sequence form the selected species (that would be a contradiction since if a family contains a sequence from a selected species that family was not eliminated and thus no GFLE

occurred). Summary of the entire process can be found in Algorithm 6.

Algorithm 6 *Let PS^a be a set of phylostrata and let $ps \subset PS$ be a single phylostrata group in PS. Let $a \in A$ be a species from a set (A) such that their phylogeny relations are defined according to the underlying species tree T. Let $C_g \subset C^{ps}$ be a set of genes emerging from a GFGE whose origin has been traced back to phylostrata ps. Moreover, let G_a be a set of genes from species a and $\mathscr{T} : G \to A$ a function mapping a gene to its species.*

Then:

for each *species $a \in A$* ***do***
  for each *family cluster $C_g \subset C^{ps}$* ***do***
    $m \leftarrow$ (root) { Set m to root }
    if *$C_g \cap G_a = \emptyset$* ***then***
      for each *gene $g \in C_g$* ***do***
        Compute the most recent common ancestor (MRCA) of species a and species from which gene g comes from: $n \leftarrow MRCA(a, \mathscr{T}(g))$
        if *the ancestral species n is younger than m (appeared latter in the evolution)* ***then***
          $m \leftarrow n$
        end if
      end for
      Add m to GFLE of family C_g: $GFLE_{C_g} \leftarrow GFLE_{C_g} \cup m$
    end if
  end for
end for

In the above algorithm the MRCA computation can be implemented in constant time by pre-calculating the "Euler tour" (Tarjan and Vishkin, 1984) of the underlying species tree. Euler tour technique is a method from graph theory designed for tree representation. The tree in such representation is viewed as a directed graph that contains two directed edges for

each edge in the tree. The tree can then be represented as a Eulerian circuit of the directed graph, known as the Euler tour representation such that, at each node, its depth with respect to the root is computed and stored in an array (in a visiting order). Once calculated such array can then be used in $1\pm$ RMQ (Range Minimum Query) computation as described by Fischer and Heun (2007). Range Minimum Query computation is a constant time algorithm designed to locate a minimum value within an array, given two indexes. However, to be able to apply the $1\pm$ RMQ strategy to this particular situation, the depth values in the computation are required to be a unit distances, that is, the distance value between any two neighbouring nodes in the tree has to be equal. Therefore, in this particular case, during the "Euler tour" traversal, the distance is incremented (by 1) each time a child node is visited and decremented (by 1) when a parent is reached.

As a final result of the above computation is a set of GFLEs each defined with respect to the underlying species linage. Therefore, unlike GFGE which for a given gene family can occur only once in the evolution, GFLE is a multiple occurring event in accord to Dollo's parsimony model which underpins the overall gain and loss computations conducted in this thesis.

A.4 Measuring Genome Complexity

The physical complexity is a quantity which reflects the amount of information a system (usually a symbolic sequence) has about the environment within which it exists (Adami and Cerf, 2000). The genome is an example of one such system where the information is encoded within genes it consists of, therefore, physical complexity seems to be a reasonable choice for quantifying its complexity.

In their paper, in accord to Shannon and Weaver (1949), Adami and Cerf (2000) define physical complexity as a the number of meaningful units (bits) encoded in a symbolic string (ω). Where "meaningful" is defined with respect to the environment (ψ) obtainable through a set of rules authors refer to as a "program" of vanishing size. A measure in algorithmic complexity theory that reflects this concept is called "mutual complexity" ($K(\omega : \psi)$) (Kol-

mogorov, 1965, 1983; Zurek, 1989) and it holds that:

Definition 4 *Given the environment ψ with respect to which a symbolic sequence ω is encoded, mutual complexity shared between a sequence and its environment is:*

$$K(\omega : \psi) = K_0(\omega) - K(\omega|\psi) \tag{A.12}$$

where $K_0(\omega)$ is a complexity of a completely random symbolic sequence (genome) and $K(\omega|\psi)$ is a complexity of a sequence (genome) encoded with respect to environment (ψ).

Information, as Claude E. Shannon stated, is a quantity assigned to a collection of events describing certain properties of random processes (Shannon and Weaver, 1949). Thus, the information does not directly depend on some physical units and quantities of the system but on relative frequencies of events occurring within it. If the event is in abundance (high frequency), its uncertainty is low and thus highly informative in describing the environment. Shannon refers to this uncertainty as entropy (H). Decrease in entropy of a "symbolic sequence" (genome in this particular case) reflects the amount of information (I) such sequence contains about its environment. This concept is reflected through equation:

$$I(\Omega : \Psi) = H_{\max}(\Omega) - H(\Omega|\Psi) \tag{A.13}$$

In it $H_{\max}(\Omega)$ stands for the maximum entropy of a collection defined with respect to a non existing environment, thus a set of truly random sequences and $H(\Omega|\Psi)$ is the entropy of a collection Ω given a collection of corresponding environments (Ψ). The entropy (H) of any set (X) is defined through probability (p_i) of an occurring event in X (Shannon and Weaver, 1949) calculated as:

$$H(X) = -\sum_i p_i \log(p_i) \tag{A.14}$$

Since in the absence of selection all sequences ($\omega \in \Omega$) are equally probable,the probabilities are equal to $1/|\Omega|$ and thus the entropy takes its maximum value.

Furthermore, note that the entropy is defined as an average property and therefore the information measure in equation A.14 is the average information a sequence in Ω encodes about its environment. Since physical complexity is defined as the amount of information a system has about its environment, using the equation A.14 to calculate this amount gives the average physical complexity:

$$\begin{aligned}\overline{K(\omega:\psi)} &= \sum_{\omega}^{\Omega} p(\omega)\log(p(\omega)) \\ &\approx I(\Omega:\Psi) = H_{\max}(\Omega) - H(\Omega|\Psi)\end{aligned} \tag{A.15}$$

and thus:

$$\overline{K(\omega:\psi)} \approx H_{\max}(\Omega) - H(\Omega|\Psi) \tag{A.16}$$

In the rest of the text I denote $\overline{K(\omega:\psi)}$ as $\mathfrak{C}$ for simplicity.

Under selection, in the course of evolution, gene family will either be fixated with a genome at that point or will be eliminated. In case of fixation, (due to the lack of alternatives) the probability of a fixed family occurring within a genome is 1 and thus the associated uncertainty (entropy - H) is 0. Which means that the sum of all uncertainties associated to corresponding gene families is 0.

In a hypothetical situation in the absence of selection the probability of any possible occurring family within a genome is less than 1. Since there is no any "a priori" information in such case one must not prefer any family and assign equal probability to each possible alternative (Laplace principle of insufficient reasoning (Dupont, 1977)). It is clear that in such case one refers to the situation when the entropy of a genome will have its maximum value equal to:

$$\begin{aligned}H_{\max}(\Omega) &= -\sum_{i}^{L} \frac{1}{L}\log\frac{1}{L} \\ &= \log L\end{aligned} \tag{A.17}$$

which for $L = |\Sigma|^{|GF|}$ where Σ is an arbitrary large set of potential family alternatives (referred to as alphabet in computational science disciplines) and $|GF|$ the number of gene families, in logarithm base $|\Sigma|$, equals to:

$$\begin{aligned} H_{\max}(\Omega) &= \log_{|\Sigma|} |\Sigma|^{|GF|} \\ &= |GF| \end{aligned} \tag{A.18}$$

Accordingly, the average physical complexity of a genome within a given environment (Eq. A.16) is:

$$\mathfrak{C} \approx |GF| \tag{A.19}$$

Given the number of GFGEs and GFLEs, $|GF|$ can be obtained by summing up all gain and loss events that happened on a path from root to a given (ancestral) species node. Thus, let P_a be a set of nodes on a path from root to species a such that $p_i^a \in P^a$ for $i \in [1..|P_a|]$, then:

$$\mathfrak{C}_{p_i^a} = \sum_{j=1}^{i} (|GFGE_{p_j^a}| - |GFLE_{p_j^a}|) \tag{A.20}$$

where $|\cdot|$ refers to the number of elements; $|GFGE_{p_j^a}|$ the number of gain and $|GFLE_{p_j^a}|$ the number of loss events at node labelled as $j \in [1..i]$. Therefore, by computing the number of $|GF|$ at each phylostrata, according to equation A.19 the complexity of each ancestral genome has been estimated.

A.5 Quantitative Distribution of *D.melanogaster* Genes Across 31 Phylostrata

Table A.2 Comparing the obtained BLAST based pipeline results with those estimated from the phylogeny based (Figure 2.2) distribution of database sequences. Computed significance values for the obtained result, confirms that in 29 out of 31 cases the BLAST based pipeline distribution is significantly different from the background, when Bonferroni multiple testing correction is considered. Note: 55 *D. melanogaster* genes were not included in the analysis due to low complexity filtering

Id	Taxonomy Id	Species	Quant	Hit	Sample	Total	LogOdd	p-value	Bonferroni	FDR
1	131567		5364		8403637		-0.5	4.9383e-230	3.0617e-228	6.1235e-229
2	2759		3220		2584566		0.45	8.7149e-104	5.4032e-102	3.1784e-103
3	1648521		83		51809		0.62	3.9492e-07	2.4485e-05	4.9969e-07
4	1648523		21		10648		0.83	1.1530e-03	7.1486e-02	1.2765e-03
5	33154		185		1221832		-1.80	5.0189e-253	3.1117e-251	1.0372e-251
6	1648563		48		25840		0.77	3.1886e-06	1.9769e-04	3.8763e-06
7	1648564		67		10190		2.04	1.4497e-35	8.9881e-34	2.6436e-35
8	1648565		77		21004		1.45	1.6882e-24	1.0467e-22	2.6167e-24
9	33208		846		249868		1.41	2.3000e-198	1.4260e-196	2.0371e-197
10	6072		52		11572		1.65	1.2299e-20	7.6254e-19	1.7330e-20
11	1648588		159		147417		0.22	7.4147e-03	4.5971e-01	8.0651e-03
12	33213		326		2011920		-1.78	0.0000e+00	0.0000e+00	0.0000e+00
13	33317		60		194619		-1.03	7.2032e-22	4.4660e-20	1.0893e-21
14	1206794		39		456876		9.17	4.7854e-119	2.9669e-117	2.1192e-118
15	6656		56		18279		1.27	6.4715e-15	4.0123e-13	8.7225e-15
16	197563		73		15061		1.73	2.1911e-30	1.3585e-28	3.6716e-30
17	197562		108		31002		1.40	5.6802e-32	3.5217e-30	9.7826e-32
18	33340		205		36399		1.89	4.8203e-94	2.9886e-92	1.5729e-93
19	1648626		98		10870		2.36	2.3194e-63	1.4380e-61	5.1358e-63
20	33392		224		221800		0.16	2.3669e-02	1	2.4873e-02
21	1648627		45		16569		1.15	1.4425e-10	8.9435e-09	1.9029e-10
22	1648628		51		43596		0.30	4.4194e-02	1	4.4918e-02
23	7147		273		59482		1.69	1.3360e-104	8.2832e-103	5.5221e-104
24	480117		107		11549		2.39	3.3348e-70	2.0676e-68	8.6149e-70
25	7215		1265		45332		3.59	4.6000e-198	2.8520e-196	3.1689e-197
26	32341		100		15612		2.01	2.1641e-51	1.3417e-49	4.6267e-51
27	1648632		159		32907		1.73	2.3926e-64	1.4834e-62	5.9336e-64
28	32346		91		15160		1.95	1.1337e-44	7.0289e-43	2.1965e-44
29	32351		175		31294		1.88	5.2504e-80	3.2552e-78	1.5501e-79
30	1648634		54		31933		0.67	1.1042e-05	6.8460e-04	1.2678e-05
31	7227		247		14180		3.04	3.3400e-198	2.0708e-196	2.5885e-197

Drosophila melanogaster

13878

16052823

Table A.3 Comparing the obtained *QPhyloStrat* results with those estimated from the phylogeny based (Figure 2.2) distribution of database sequences. Computed significance values for the obtained result confirms that in all cases computed *QPhyloStrat* origin point distribution is significantly different from the background. Note: 55 *D. melanogaster* genes were not included in the analysis due to low complexity filtering

Id	Taxonomy Id	Species	Quant	Hit	Sample	Total	LogOdd	p-value	Bonferroni	FDR
1	131567	Drosophila melanogaster	3264	13878	8401537	16052823	-1.27	0.0000e+00	0.0000e+00	0.0000e+00
2	2759		2955		2584301		0.34	3.3069e-56	2.0503e-54	6.2130e-56
3	1648521		107		51833		0.87	5.1497e-15	3.1928e-13	6.2604e-15
4	1648523		21		10648		0.82	1.2112e-03	7.5094e-02	1.2112e-03
5	33154		512		1222159		-0.7	2.5248e-83	1.5654e-81	7.4542e-83
6	1648563		40		25832		0.58	1.0039e-03	6.2242e-02	1.0549e-03
7	1648564		38		10161		1.47	5.7325e-13	3.5542e-11	6.3467e-13
8	1648565		70		20997		1.35	4.0992e-20	2.5415e-18	5.0830e-20
9	33208		608		249630		1.06	4.1794e-107	2.5912e-105	1.3638e-106
10	6072		112		11632		2.42	5.9159e-75	3.6679e-73	1.3099e-74
11	1648588		307		147565		0.89	3.0673e-41	1.9017e-39	5.1398e-41
12	33213		726		2012320		-0.9	1.1433e-186	7.0885e-185	7.0885e-186
13	33317		99		194658		-0.5	6.2411e-09	3.8695e-07	6.7886e-09
14	1206794		100		456937		-1.39	4.0416e-72	2.5058e-70	8.3526e-72
15	6656		83		18306		1.66	2.9033e-32	1.8000e-30	4.5001e-32
16	197563		88		15076		1.92	3.3756e-42	2.0929e-40	5.9796e-42
17	197562		93		30987		1.25	3.6674e-23	2.2738e-21	4.7371e-23
18	33340		182		36376		1.77	1.7248e-75	1.0694e-73	4.2775e-75
19	1648626		116		10888		2.53	2.8333e-82	1.7566e-80	7.3194e-82
20	33392		351		221927		0.61	7.2318e-25	4.4837e-23	1.0936e-24
21	1648627		68		16592		1.56	2.7713e-24	1.7182e-22	3.9958e-24
22	1648628		92		43637		0.89	1.3503e-13	8.3719e-12	1.5796e-13
23	7147		306		59515		1.80	3.3959e-129	2.1055e-127	1.2385e-128
24	480117		98		11540		2.29	6.7114e-61	4.1611e-59	1.3003e-60
25	7215		1782		45849		3.97	5.8600e-198	3.6332e-196	5.1903e-197
26	32341		198		15710		2.70	5.9871e-153	3.7120e-151	3.3745e-152
27	1648632		258		33006		2.22	8.2026e-150	5.0856e-148	3.9120e-149
28	32346		190		15259		2.69	7.0187e-146	4.3516e-144	2.7197e-145
29	32351		489		31608		2.93	4.3200e-198	2.6784e-196	4.4640e-197
30	1648634		95		31974		1.24	2.4762e-23	1.5352e-21	3.3375e-23
31	7227		485		14418		3.73	1.2000e-198	7.4400e-197	1.4880e-197

A.6 Gene Family Gain Events

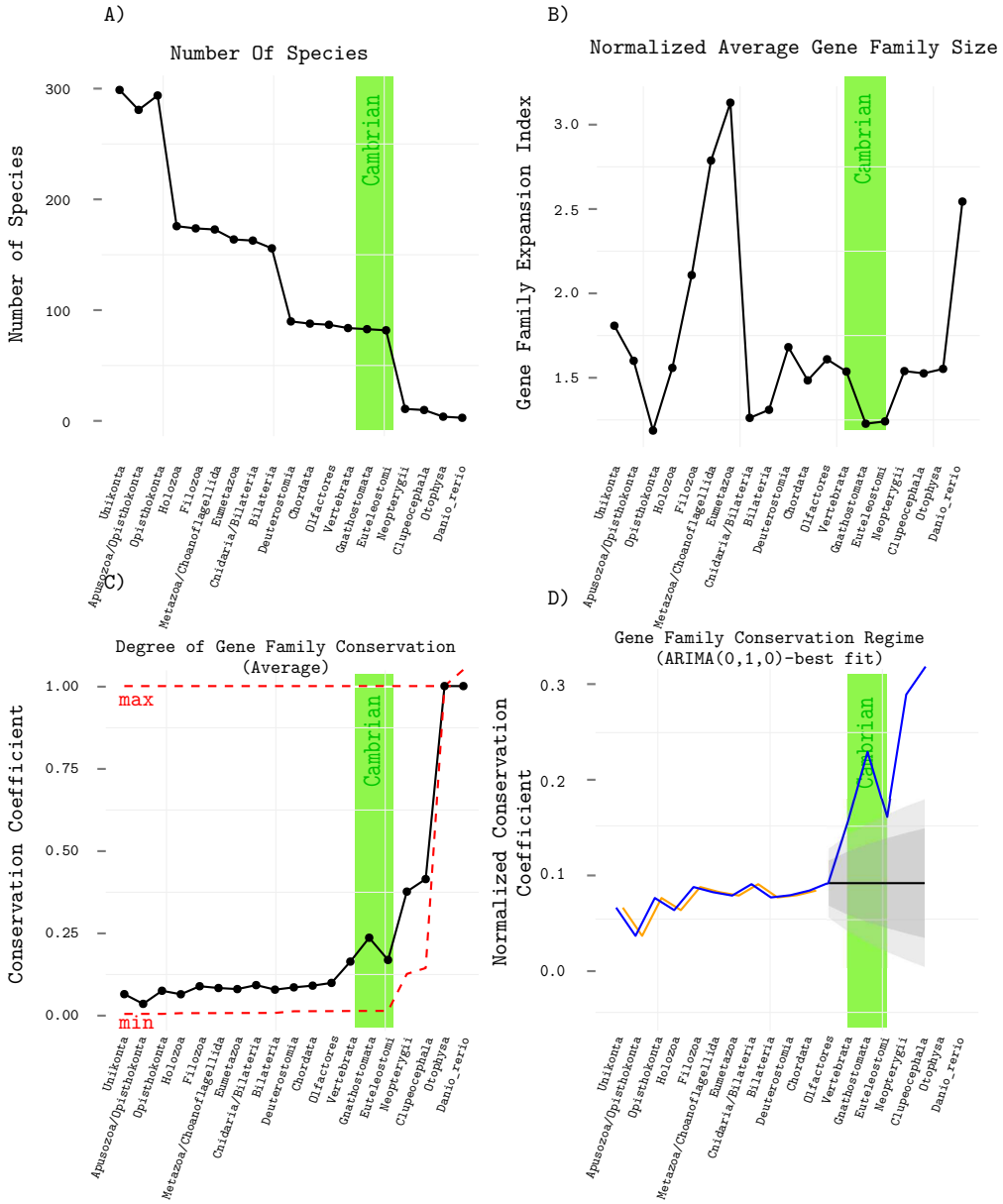


Fig. A.7 Statistical analysis of GFGEs from Unikonta to *D. rerio*. A) The number of species included in GFGE computation. B) The estimated average number of genes within a gene family. C) The degree of gene family conservation plotted as a black dotted line. The conservation coefficient is calculated as the average number of species within which a family has been preserved divided by the total number of species associated to GFGEs in a given phylostrata. Red dashed lines mark the borders (minimum and maximum conservation levels each family in a given phylostrata can have). Blue line depicts a normalized average conservation with respect to calculated borders (dashed red lines). D) Gene family conservation regime analysis. Based on the average gene family conservation levels form Unikonta to Olfactores, ARIMA(0,1,0) model was used to estimate 95% (light gray) and 80% (dark gray) confidence intervals within which conservation levels were expected to occur, if the same underlying evolutionary regime continued to influence the levels of gene family conservation. Orange line represents the ARIMA(0,1,0) data fit and the blue line, computed gene family conservation levels.

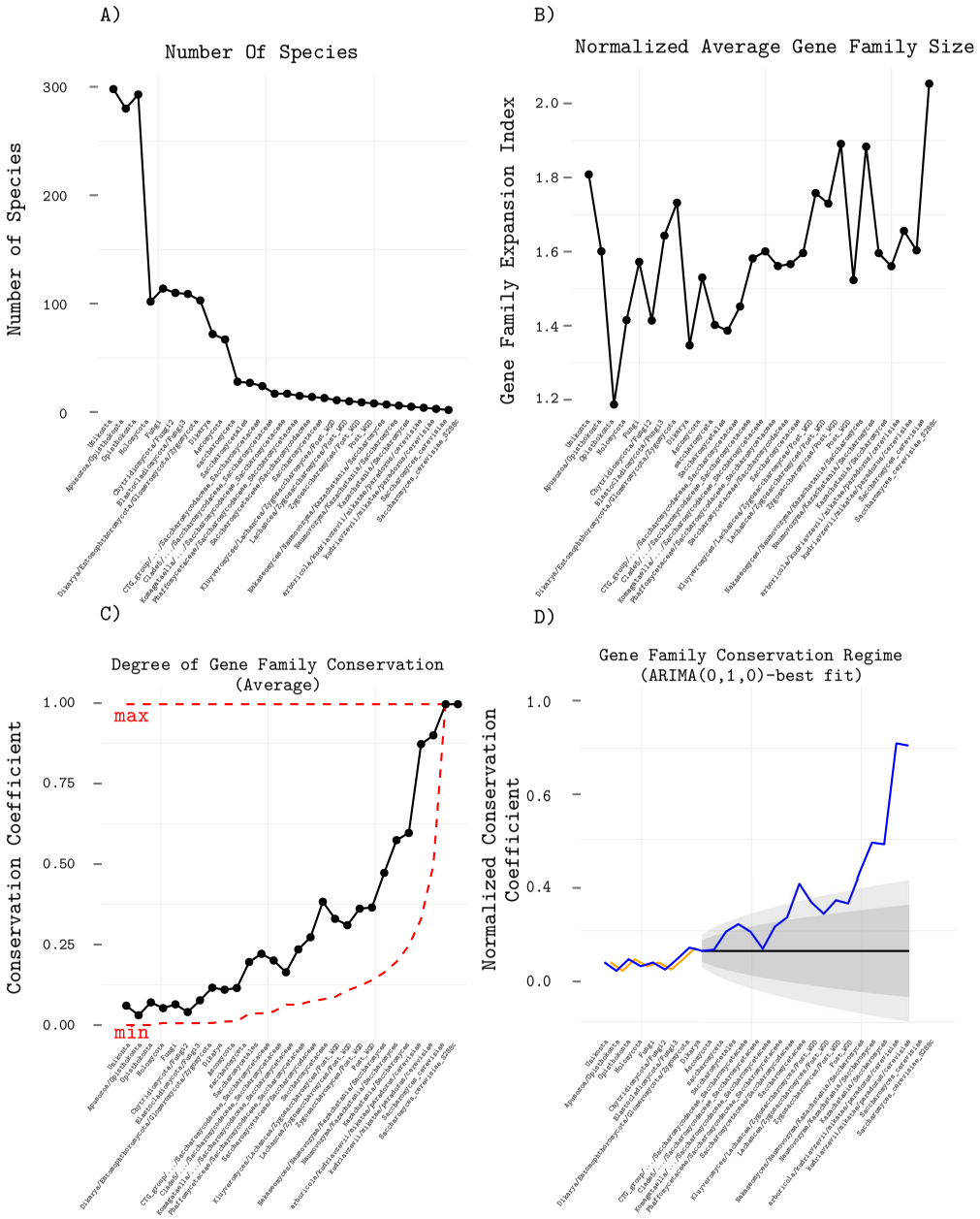


Fig. A.8 Statistical analysis of GFGEs from Unikonta to *S. cerevisiae*. A) The number of species included in GFGE computation. B) The estimated average number of genes within a gene family. C) The degree of gene family conservation plotted as a black dotted line. The conservation coefficient is calculated as the average number of species within which a family has been preserved divided by the total number of species associated to GFGEs in a given phylostrata. Red dashed lines mark the borders (minimum and maximum conservation levels each family in a given phylostrata can have). Blue line depicts a normalized average conservation with respect to calculated borders (dashed red lines). D) Gene family conservation regime analysis. Based on the average gene family conservation levels form Unikonta to Dikarya, ARIMA(0,1,0) model was used to estimate 95% (light gray) and 80% (dark gray) confidence intervals within which conservation levels were expected to occur, if the same underlying evolutionary regime continued to influence the levels of gene family conservation. Orange line represents the ARIMA(0,1,0) data fit and the blue line, computed gene family conservation levels.

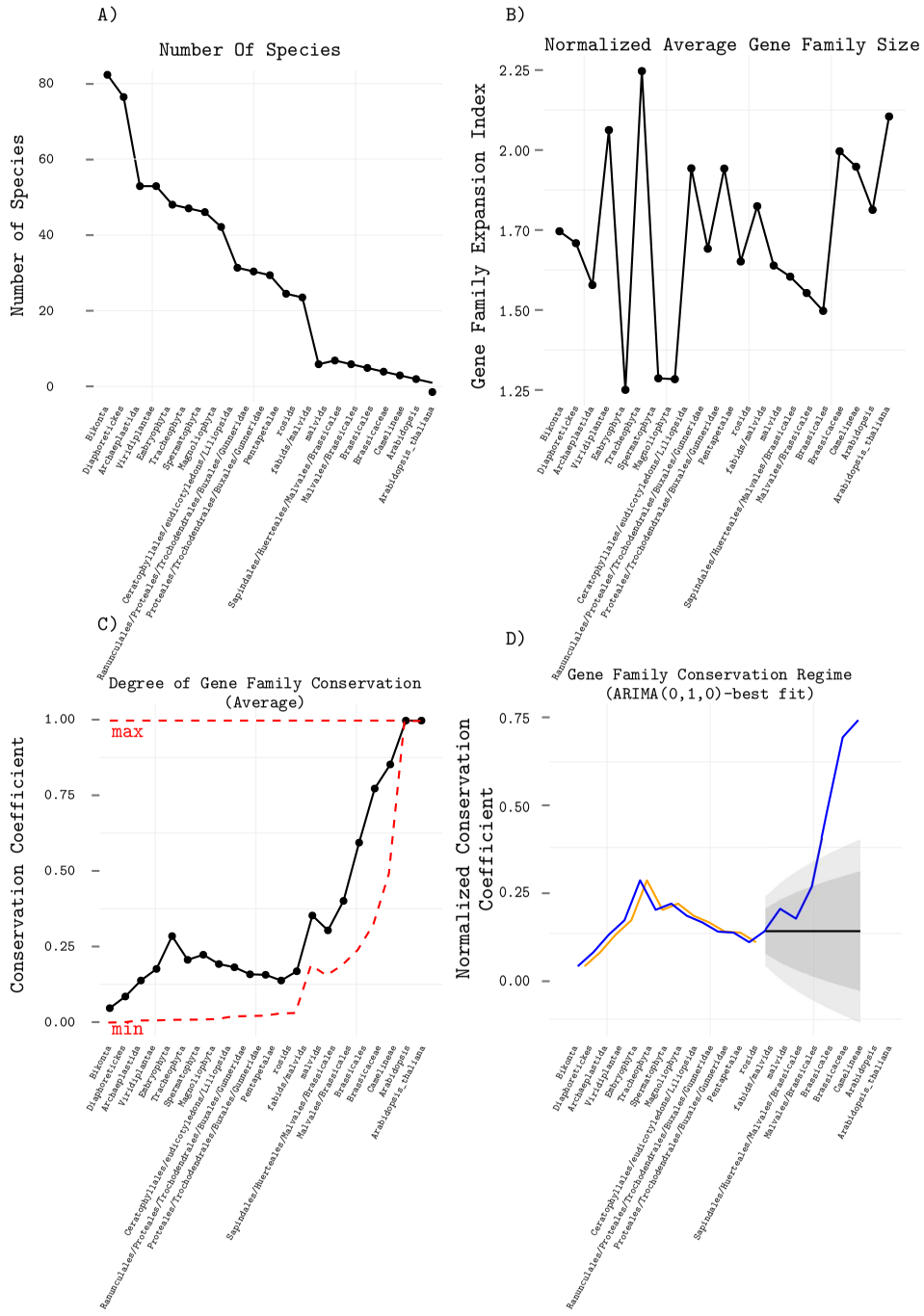


Fig. A.9 Statistical analysis of GFGEs from Unikonta to *A. thaliana*. A) The number of species included in GFGE computation. B) The estimated average number of genes within a gene family. C) The degree of gene family conservation plotted as a black dotted line. The conservation coefficient is calculated as the average number of species within which a family has been preserved divided by the total number of species associated to GFGEs in a given phylostrata. Red dashed lines mark the borders (minimum and maximum conservation levels each family in a given phylostrata can have). Blue line depicts a normalized average conservation with respect to calculated borders (dashed red lines). D) Gene family conservation regime analysis. Based on the average gene family conservation levels form Bikonta to Rosids, ARIMA(0,1,0) model was used to estimate 95% (light gray) and 80% (dark gray) confidence intervals within which conservation levels were expected to occur, if the same underlying evolutionary regime continued to influence the levels of gene family conservation. Orange line represents the ARIMA(0,1,0) data fit and the blue line, computed gene family conservation levels.

A.7 Gene Family Loss Events

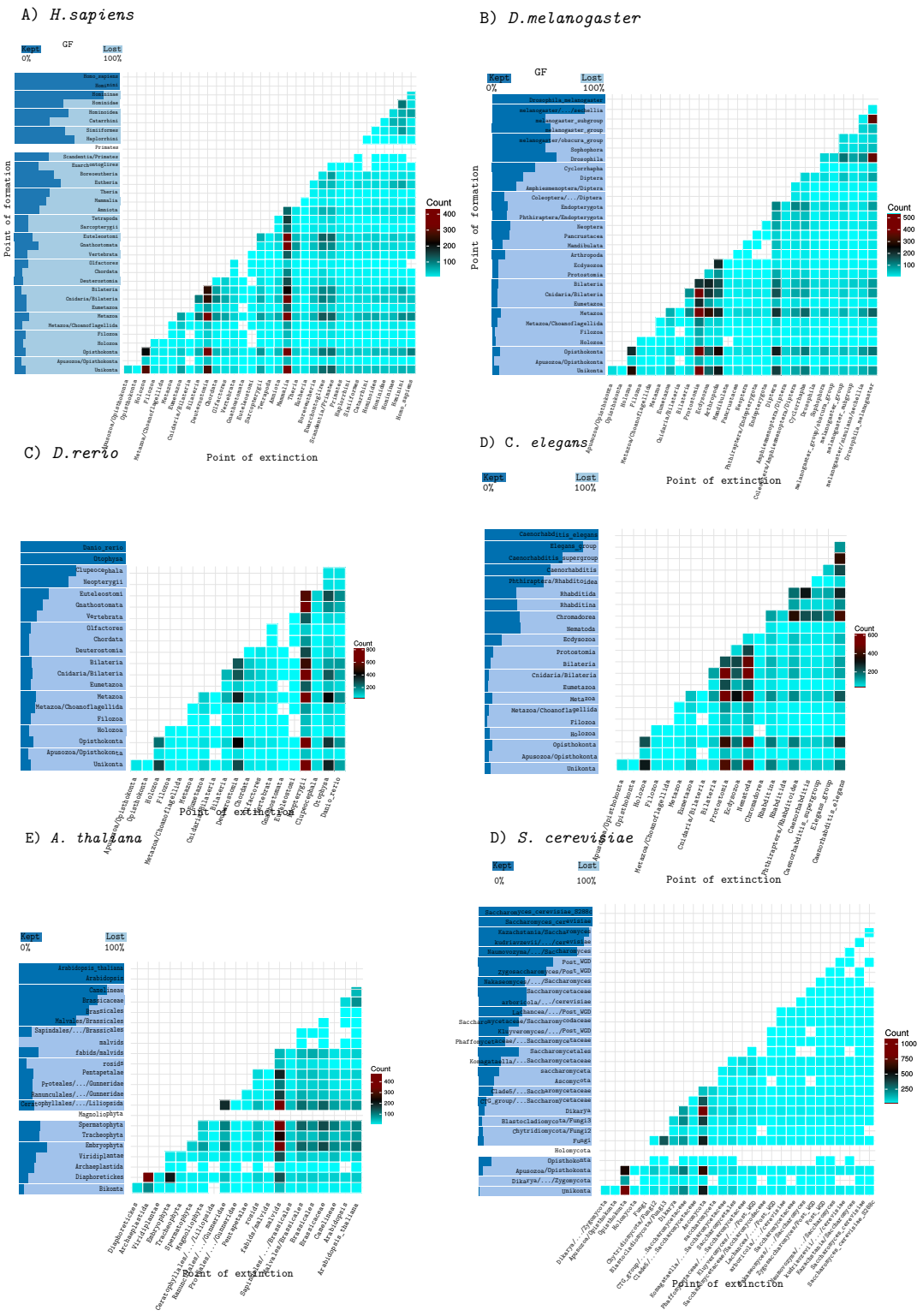


Fig. A.10 The number of loss events associated to each gene family gain event examined in six lineages: *H. sapiens, D. melanogaster, D. rerio, C. elegans, A. thaliana, S. cerevisiae*

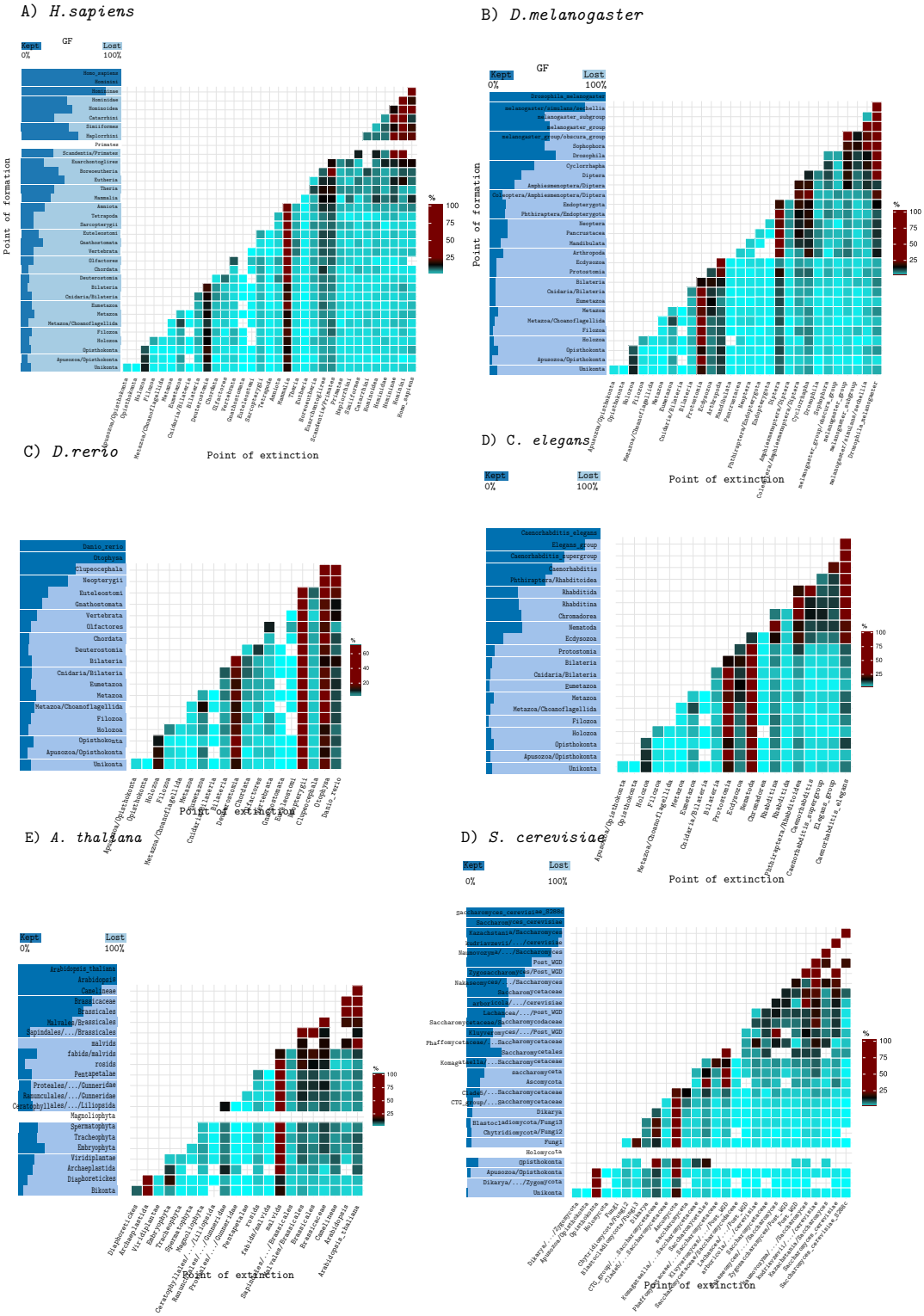


Fig. A.11 The number of gene family loss events associated to each family gain event normalized by the overall number of lost families associated to a given evolutionary period. Analysis involved *H. sapiens, D. melanogaster, D. rerio, C. elegans, A. thaliana* and *S. cerevisiae* lineage

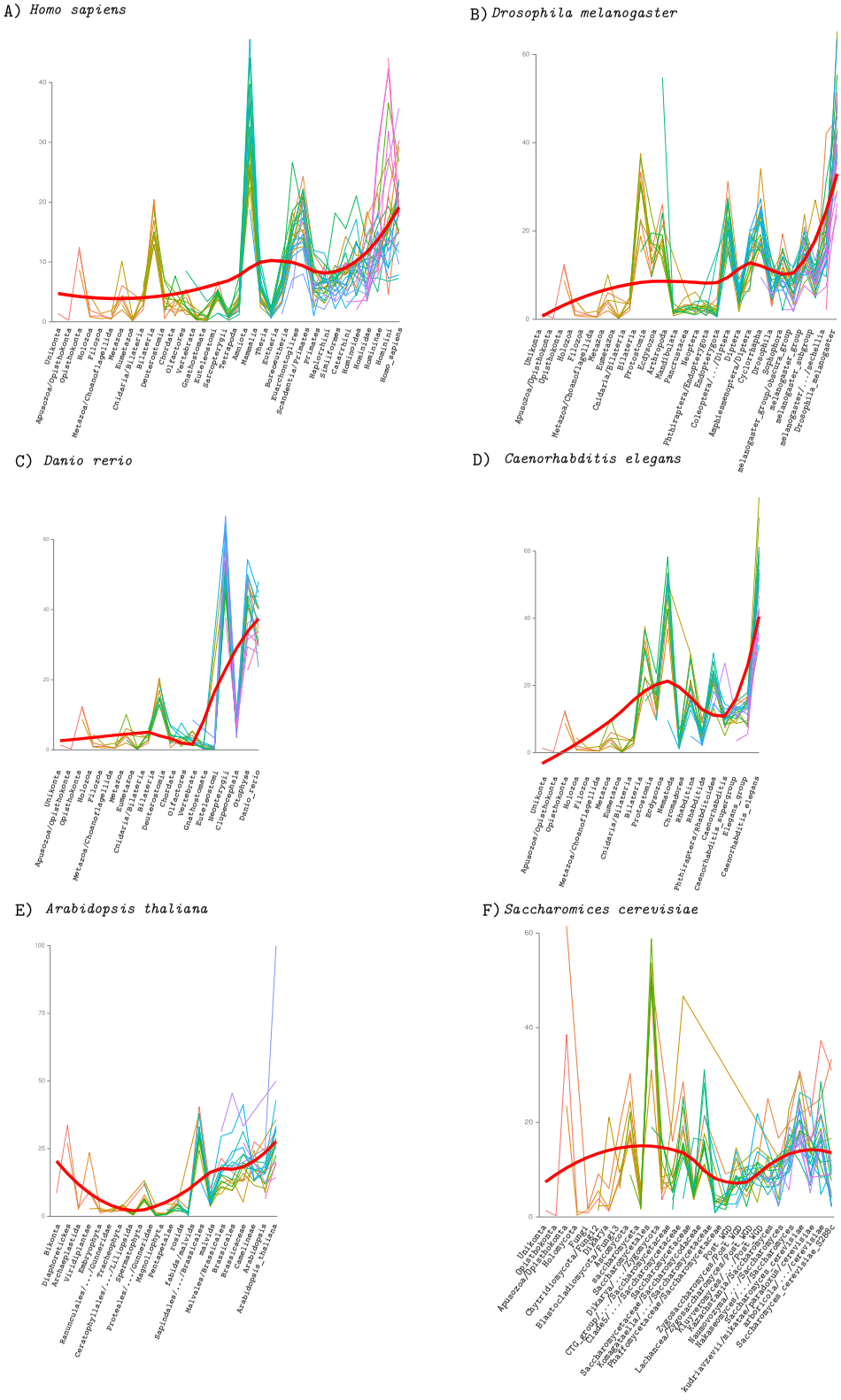


Fig. A.12 Loss rates of families emerging at specific evolutionary periods. Red line (local polynomial regression fitting curve) depicts a general trend regarding the loss rate. Noteworthy is a clear increase in rate of extinct families in all six lineages: *H. sapiens, D. melanogaster, D. rerio, C. elegans, A. thaliana, S. cerevisiae*

A.8 *db_200514* Database Content

Table A.4 Summary information of eukaryote species present in *db_200514* database. $(+)$ selected Eukaryote representatives, $(*)$ only longest splice variant included.

Number of genes	Species name	Taxonomy Id	Ass. info.	Source	Download date
18051	Acanthamoeba castellanii str. Neff	1257118	+	Ruiz Trillo group	03.04.2014
28283	Acanthisitta chloris	57068	+	na	03.04.2014
11202	Acidomyces richmondensis	245562		genome.jgi.doe.gov	15.01.2014
9521	Acremonium alcalophilum	398408	+	genome.jgi.doe.gov	15.01.2014
23677	Acropora digitifera	70779	+	marinegenomics.oist.jp	03.04.2014
5062	Acropora millepora	45264	+	na	03.04.2014
36195	Acyrthosiphon pisum	7029	+*	ensemblgenomes.org	03.12.2013.
16016	Aedes aegypti	7159	+*	ensemblgenomes.org	03.12.2013.
33849	Aegilops tauschii	37682	+*	ensemblgenomes.org	11.12.2013.
10438	Agaricus bisporus var. bisporus H97	936046		genome.jgi.doe.gov	15.01.2014
11289	Agaricus bisporus var. burnettii JB137-S8	597362	+	genome.jgi.doe.gov	15.01.2014
19343	Ailuropoda melanoleuca	9646	*	ensembl.org	02.12.2013
13804	Albugo laibachii Nc14	890382	*	ensemblgenomes.org	03.12.1013.
21622	Alligator mississippiensis	8496	+	na	03.04.2014
25127	Alligator sinensis	38654	+	na	03.04.2014
19446	Allomyces macrogynus ATCC 38327	578462	+	broadinstitute.org	09.12.2013
10688	Alternaria brassicicola	29001	+	genome.jgi.doe.gov	15.01.2014
18153	Amanita muscaria Koide BX008	946122		genome.jgi.doe.gov	15.01.2014
10354	Amanita thiersii Skay4041	703135		genome.jgi.doe.gov	15.01.2014
26846	Amborella trichopoda	13333	+	amborella.org	20.01.2014.
36644	Ambystoma mexicanum	8296	+	na	04.03.2014
9642	Amorphotheca resinae	5101	+	genome.jgi.doe.gov	15.01.2014
29883	Amphimedon queenslandica	400682	+*	ensemblgenomes.org	03.12.2013.
11327	Amyloporia sinuosa	1333626		genome.jgi.doe.gov	15.01.2014
15634	Anas platyrhynchos	8839	+*	ensembl.org	02.12.2013
65583	Ancylostoma ceylanicum	53326	+	na	03.04.2014
18596	Anolis carolinensis	28377	+*	ensembl.org	02.12.2013
11430	Anopheles darlingi	43151	+*	ensemblgenomes.org	03.12.2013.
12810	Anopheles gambiae	7165	+*	ensemblgenomes.org	03.12.2013.
15755	Anthostoma avocetta NRRL 3190	1405086		genome.jgi.doe.gov	15.01.2014
2606	Antonospora locustae HM-2013	1284279	+	genome.jgi.doe.gov	15.01.2014
26721	Apaloderma vittatum	57397		na	03.04.2014
18756	Apis dorsata	7462		na	03.04.2014
18166	Apis florea	7463	+	na	03.04.2014
10675	Apis mellifera	7460	+*	ensemblgenomes.org	03.12.2013.

11892	Aplanochytrium kerguelense PBS07	702273	+	genome.jgi.doe.gov	15.01.2014
12579	Aplosporella prunicola CBS 121167	1176127	+	genome.jgi.doe.gov	15.01.2014
9233	Aptenodytes forsteri	9233		na	03.04.2014
31277	Aquila chrysaetos canadensis	216574		na	03.04.2014
41063	Aquilegia coerulea	218851	+	genome.jgi.doe.gov	04.05.2014
32667	Arabidopsis lyrata subsp. lyrata	81972	+*	ensemblgenomes.org	11.12.2013.
27416	Arabidopsis thaliana	3702	+*	ensemblgenomes.org	11.12.2013.
14473	Armillaria mellea DSM 3731	1314977		genome.jgi.doe.gov	15.01.2014
16992	Arthrinium arundis NRRL 25634	1149870		genome.jgi.doe.gov	15.01.2014
11479	Arthrobotrys oligospora ATCC 24927	756982		genome.jgi.doe.gov	15.01.2014
7980	Arthroderma benhamiae CBS 112371	663331		genome.jgi.doe.gov	15.01.2014
8915	Arthroderma otae CBS 113480	554155		genome.jgi.doe.gov	15.01.2014
18542	Ascaris suum	6253	+	Wormbase	03.04.2014
17877	Ascobolus immersus RN42	1160509		genome.jgi.doe.gov	15.01.2014
6802	Ascoidea rubescens NRRL Y17699	983968	+	genome.jgi.doe.gov	15.01.2014
4776	Ashbya gossypii ATCC 10895	284811	+*	ensemblgenomes.org	03.12.2013.
13530	Aspergillus acidus	929029	+	genome.jgi.doe.gov	15.01.2014
10828	Aspergillus aculeatus ATCC 16872	690307		genome.jgi.doe.gov	15.01.2014
13000	Aspergillus brasiliensis	319629		genome.jgi.doe.gov	15.01.2014
11624	Aspergillus carbonarius ITEM 5010	602072		genome.jgi.doe.gov	15.01.2014
9121	Aspergillus clavatus NRRL 1	344612	*	ensemblgenomes.org	03.12.2013.
13487	Aspergillus flavus NRRL3357	332952	*	ensemblgenomes.org	03.12.2013.
9915	Aspergillus fumigatus A1163	451804	*	ensemblgenomes.org	03.12.2013.
9627	Aspergillus fumigatus Af293	330879	+*	ensemblgenomes.org	03.12.2013.
11277	Aspergillus glaucus	41413		genome.jgi.doe.gov	15.01.2014
11472	Aspergillus kawachii	40384		genome.jgi.doe.gov	15.01.2014
10534	Aspergillus nidulans FGSC A4	227321	*	ensemblgenomes.org	03.12.2013.
11197	Aspergillus niger ATCC 1015	380704		genome.jgi.doe.gov	15.01.2014
14068	Aspergillus niger CBS 513.88	425011	*	ensemblgenomes.org	03.12.2013.
12074	Aspergillus oryzae RIB40	510516	*	ensemblgenomes.org	03.12.2013.
10076	Aspergillus ruber	396024		genome.jgi.doe.gov	15.01.2014
13620	Aspergillus sydowii	75750		genome.jgi.doe.gov	15.01.2014
10402	Aspergillus terreus NIH2624	341663	*	ensemblgenomes.org	03.12.2013.
12322	Aspergillus tubingensis	5068		genome.jgi.doe.gov	15.01.2014
13228	Aspergillus versicolor	46472		genome.jgi.doe.gov	15.01.2014
12442	Aspergillus wentii	5066	+	genome.jgi.doe.gov	15.01.2014
41063	Aspergillus zonatus	41063		genome.jgi.doe.gov	15.01.2014
33318	Asterochloris sp. Cgr/DA1pho	763042		genome.jgi.doe.gov	04.05.2014
23042	Astyanax mexicanus	7994	+*	ensembl.org	11.12.2013
19518	Athalia rosae	37344		na	03.04.2014
18062	Atta cephalotes	12957	+*	ensemblgenomes.org	03.12.2013.
12127	Aulographum hederae	1176130		genome.jgi.doe.gov	15.01.2014
14859	Aurantiochytrium limacinum ATCC MYA-1381	717989	+	genome.jgi.doe.gov	15.01.2014
17631	Aurelia aurita	6145	+	compagen.org	03.04.2014

10594	Aureobasidium melanogenum	46634		genome.jgi.doe.gov	15.01.2014
10266	Aureobasidium namibiae	559561		genome.jgi.doe.gov	15.01.2014
11866	Aureobasidium pullulans EXF-150	1043002		genome.jgi.doe.gov	15.01.2014
10809	Aureobasidium subglaciale EXF-2481	1043005		genome.jgi.doe.gov	15.01.2014
11501	Aureococcus anophagefferens	44056		genome.jgi.doe.gov	04.05.2014
23577	Auricularia delicata TFB-10046 SS5	717982		genome.jgi.doe.gov	15.01.2014
6403	Babjeviella inositovora NRRL Y-12698	984486		genome.jgi.doe.gov	15.01.2014
17039	Backusella circina FSU 941	1314798	+	genome.jgi.doe.gov	15.01.2014
28235	Balearica regulorum gibbericeps	100784		na	03.04.2014
8732	Batrachochytrium dendrobatidis JAM81	684364	+	genome.jgi.doe.gov	15.01.2014
10513	Baudoinia compniacensis UAMH 10762	717646		genome.jgi.doe.gov	15.01.2014
10364	Beauveria bassiana ARSEF 2860	655819		genome.jgi.doe.gov	15.01.2014
29831	Beta vulgaris subsp. vulgaris	3555	+	na	03.04.2014
21708	Bigelowiella natans CCMP2755	753081	+	genome.jgi.doe.gov	15.01.2014
12720	Bipolaris maydis ATCC 48331	665024		genome.jgi.doe.gov	15.01.2014
9633	Bipolaris maydis C5	701091		genome.jgi.doe.gov	15.01.2014
12007	Bipolaris oryzae ATCC 44560	930090	+	genome.jgi.doe.gov	15.01.2014
12250	Bipolaris sorokiniana ND90Pr	665912		genome.jgi.doe.gov	15.01.2014
12894	Bipolaris victoriae FI3	930091		genome.jgi.doe.gov	15.01.2014
12857	Bipolaris zeicola 26-R-13	930089		genome.jgi.doe.gov	15.01.2014
15473	Bjerkandera adusta	5331		genome.jgi.doe.gov	15.01.2014
11444	Blastomyces dermatitidis ATCC 18188	653446		broadinstitute.org	13.01.2014
11211	Blastomyces dermatitidis ATCC 26199	447095	+	broadinstitute.org	13.01.2014
11539	Blastomyces dermatitidis ER-3	559297		broadinstitute.org	13.01.2014
11343	Blastomyces dermatitidis SLH14081	559298		broadinstitute.org	13.01.2014
6470	Blumeria graminis f. sp. hordei DH14	546991	*	ensemblgenomes.org	03.12.2013.
16933	Boletus edulis	36056		genome.jgi.doe.gov	15.01.2014
20930	Bombus impatiens	132113		na	03.04.2014
20552	Bombus terrestris	30195	+	na	03.04.2014
14623	Bombyx mori	7091	+*	ensemblgenomes.org	03.12.2013.
19994	Bos taurus	9913	+*	ensembl.org	02.12.2013
16526	Botryobasidium botryosum	264124		genome.jgi.doe.gov	15.01.2014
14998	Botryosphaeria dothidea	55169	+	genome.jgi.doe.gov	15.01.2014
10345	Botrytis cinerea B05.10	332648	*	ensemblgenomes.org	03.12.2013.
26552	Brachypodium distachyon	15368	+*	ensemblgenomes.org	11.12.2013.
50817	Branchiostoma floridae	7739	+	genome.jgi.doe.gov	15.01.2014
41025	Brassica rapa subsp. pekinensis	51351	+	ensemblgenomes.org	11.12.2013.
5636	Brettanomyces bruxellensis CBS 2499	747657	+	genome.jgi.doe.gov	15.01.2014
14219	Brugia malayi	6279	*	ensemblgenomes.org	03.12.2013.
27410	Buceros rhinoceros silvestris	175836		na	03.04.2014
18074	Bursaphelenchus xylophilus	6326	+	Wormbase	03.04.2014
33934	Caenorhabditis angaria	860376	+	Wormbase	03.04.2014
30667	Caenorhabditis brenneri	135651	+*	ensemblgenomes.org	03.12.2013.
21863	Caenorhabditis briggsae	6238	+	na	03.04.2014

20532	Caenorhabditis elegans	6239	+*	ensemblgenomes.org	03.12.2013.
29964	Caenorhabditis japonica	281687	+*	ensemblgenomes.org	03.12.2013.
31444	Caenorhabditis remanei	31234	+*	ensemblgenomes.org	03.12.2013.
20993	Callithrix jacchus	9483	+*	ensembl.org	02.12.2013
26800	Callorhinchus milii	7868	+	esharkgenome.imcb.a-star.edu.sg	26.03.2014
13177	Calocera cornea	29889	+	genome.jgi.doe.gov	15.01.2014
12378	Calocera viscosa	63146		genome.jgi.doe.gov	15.01.2014
30246	Calypte anna	9244	+	na	03.04.2014
17064	Camponotus floridanus	104421	+	hymenopteragenome.org	03.04.2014
6221	Candida albicans SC5314	237561	+	candidagenome.org	03.04.2014
5861	Candida arabinofermentans NRRL YB-2248	983967	+	genome.jgi.doe.gov	15.01.2014
5235	Candida glabrata CBS 138	284593	+	candidagenome.org	03.04.2014
5895	Candida tanzawaensis NRRL Y-17324	984487		genome.jgi.doe.gov	15.01.2014
5533	Candida tenuis	45596	+	genome.jgi.doe.gov	15.01.2014
6258	Candida tropicalis MYA-3404	294747	+	candidagenome.org	03.04.2014
19856	Canis lupus familiaris	9615	+*	ensembl.org	02.12.2013
32175	Capitella teleta	283909	+*	ensemblgenomes.org	03.12.2013.
29072	Caprimulgus carolinensis	279965	+	na	03.04.2014
9231	Capronia coronata CBS 617.96	1182541		broadinstitute.org	13.01.2014
10469	Capronia epimyces CBS 606.96	1182542		broadinstitute.org	13.01.2014
10123	Capsaspora owczarzaki ATCC 30864	595528	+	broadinstitute.org	09.12.2013
28447	Capsella rubella	81985	+	genome.jgi.doe.gov	04.05.2014
28047	Cariama cristata	54380		na	03.04.2014
27775	Carica papaya	3649	+	genome.jgi.doe.gov	04.05.2014
14188	Catenaria anguillulae PL171	765915	+	genome.jgi.doe.gov	15.01.2014
11467	Cathartes aura	43455	+	na	03.04.2014
18673	Cavia porcellus	10141	+*	ensembl.org	02.12.2013
14748	Cenococcum geophilum 1.58	794803		genome.jgi.doe.gov	15.01.2014
40480	Cerapachys biroi	443821		na	03.04.2014
12841	Ceratosolen solmsi marchali	326594	+	na	03.04.2014
12020	Cercospora zeae-maydis	135779		genome.jgi.doe.gov	15.01.2014
12125	Ceriporiopsis subvermispora B	914234	+	genome.jgi.doe.gov	15.01.2014
12966	Cerrena unicolor	90312		genome.jgi.doe.gov	15.01.2014
11124	Chaetomium globosum	38033		genome.jgi.doe.gov	15.01.2014
29403	Chaetura pelagica	8897	+	na	03.04.2014
19765	Chalara longipes BDJ	1379296		genome.jgi.doe.gov	15.01.2014
31137	Charadrius vociferus	50402	+	na	03.04.2014
39561	Chelonia mydas	8469	+	na	03.04.2014
19526	Chlamydomonas reinhardtii	3055	+	ensemblgenomes.org	11.12.2013.
27145	Chlamydotis macqueenii	187382	+	na	03.04.2014
55307	Chlorella variabilis	554065		genome.jgi.doe.gov	15.01.2014
12393	Choloepus hoffmanni	9358	+*	ensembl.org	02.12.2013
42569	Chrysemys picta bellii	8478	+	na	03.04.2014
16658	Ciona intestinalis	7719	+*	ensembl.org	02.12.2013

11616	Ciona savignyi	51511	+*	ensembl.org	02.12.2013
23438	Citrullus lanatus	3654	+	na	03.04.2014
33929	Citrus clementina	85681	+	genome.jgi.doe.gov	04.05.2014
46147	Citrus sinensis	2711	+	genome.jgi.doe.gov	04.05.2014
11415	Cladonia grayi	27339		genome.jgi.doe.gov	15.01.2014
10373	Cladophialophora carrionii CBS 160.54	1279043		broadinstitute.org	13.01.2014
13421	Cladophialophora psammophila CBS 110553	1182543	+	broadinstitute.org	13.01.2014
10118	Cladophialophora yegresii CBS 114405	1182544		broadinstitute.org	13.01.2014
5941	Clavispora lusitaniae ATCC 42720	306902		candidagenome.org	03.04.2014
14226	Clonorchis sinensis	79923	+	na	03.04.2014
10593	Coccidioides immitis H538.4	396776		broadinstitute.org	19.12.2013
10408	Coccidioides immitis RMSCC 2394	404692		broadinstitute.org	19.12.2013
10463	Coccidioides immitis RMSCC 3703	454286		broadinstitute.org	19.12.2013
9910	Coccidioides immitis RS	246410	+	genome.jgi.doe.gov	15.01.2014
7229	Coccidioides posadasii C735 delta SOWgp	222929		genome.jgi.doe.gov	15.01.2014
9964	Coccidioides posadasii RMSCC 3488	454284		broadinstitute.org	19.12.2013
10124	Coccidioides posadasii str. Silveira	443226	+	broadinstitute.org	19.12.2013
9629	Coccomyxa subellipsoidea C-169	574566	+	genome.jgi.doe.gov	15.01.2014
7347	Coemansia reversa NRRL 1564	763665		genome.jgi.doe.gov	15.01.2014
27081	Colius striatus	57412	+	na	03.04.2014
15777	Colletotrichum acutatum	27357		genome.jgi.doe.gov	15.01.2014
12020	Colletotrichum graminicola M1.001	645133	+*	ensemblgenomes.org	03.12.2013.
16172	Colletotrichum higginsianum IMI 349063	759273		genome.jgi.doe.gov	15.01.2014
32884	Columba livia	8932	+	na	03.04.2014
10635	Conidiobolus coronatus NRRL 28638	796925	+	genome.jgi.doe.gov	15.01.2014
13761	Coniophora puteana	80637	+	genome.jgi.doe.gov	15.01.2014
9308	Coniosporium apollinis CBS 100218	1168221	+	broadinstitute.org	13.01.2014
14245	Coprinopsis cinerea AmutBmut pab1-1	1132390		genome.jgi.doe.gov	15.01.2014
9651	Cordyceps militaris CM01	983644		genome.jgi.doe.gov	15.01.2014
20377	Cortinarius glaucopus AT 2004 276	1149754		genome.jgi.doe.gov	15.01.2014
30448	Corvus brachyrhynchos	85066	+	na	03.04.2014
24326	Corvus cornix cornix	932674		na	03.04.2014
26089	Crassostrea gigas	29159	+*	ensemblgenomes.org	03.12.2013.
7062	Creolimax fragrantissima	470921	+	multicellgenome.com	13.11.2013
13903	Cronartium quercuum f. sp. fusiforme G11	708437	+	genome.jgi.doe.gov	15.01.2014
11609	Cryphonectria parasitica EP155	660469		genome.jgi.doe.gov	15.01.2014
6967	Cryptococcus neoformans var. grubii H99	235443	+	genome.jgi.doe.gov	15.01.2014
6273	Cryptococcus neoformans var. neoformans JEC21	214684	+*	ensemblgenomes.org	03.12.2013.
7232	Cryptococcus vishniacii	89929	+	genome.jgi.doe.gov	15.01.2014
3886	Cryptosporidium hominis TU502	353151	+	cryptodb.org	04.05.2014
3805	Cryptosporidium parvum Iowa II	353152	+	cryptodb.org	04.05.2014
30669	Cuculus canorus	55661		na	03.04.2014
26746	Cucumis melo	3656	+	na	03.04.2014
30364	Cucumis sativus	3659	+	genome.jgi.doe.gov	04.05.2014

12439	Cucurbitaria berberidis CBS 394.84	1168544		genome.jgi.doe.gov	15.01.2014
18954	Culex quinquefasciatus	7176	+*	ensemblgenomes.org	03.12.2013.
12131	Curvularia lunata m118	977863		genome.jgi.doe.gov	15.01.2014
4998	Cyanidioschyzon merolae strain 10D	280699	*	ensemblgenomes.org	11.12.2013.
32167	Cyanophora paradoxa	2762	+	cyanophora.rutgers.edu	04.05.2014
6038	Cyberlindnera jadinii NRRL Y-1542	983966	+	genome.jgi.doe.gov	15.01.2014
13940	Cylindrobasidium torrendii	394432	+	genome.jgi.doe.gov	15.01.2014
11094	Cyphellophora europaea CBS 101466	1220924		broadinstitute.org	13.01.2014
23745	Cytauxzoon sp. Enebro	428574		na	04.05.2014
10242	Dacryopinax sp. DJM-731 SS1	745407		genome.jgi.doe.gov	15.01.2014
10959	Dactylellina haptotyla CBS 200.50	1284197		genome.jgi.doe.gov	15.01.2014
12199	Daedalea quercina	40437		genome.jgi.doe.gov	15.01.2014
11173	Daldinia eschscholtzii	292717		genome.jgi.doe.gov	15.01.2014
16254	Danaus plexippus	13037	+*	ensemblgenomes.org	03.12.2013.
26247	Danio rerio	7955	+*	ensembl.org	02.12.2013
30894	Daphnia pulex	6669	+*	ensemblgenomes.org	03.12.2013.
14803	Dasypus novemcinctus	9361	*	ensembl.org	02.12.2013
6272	Debaryomyces hansenii	4959		genome.jgi.doe.gov	15.01.2014
12290	Dichomitus squalens	114155	+	genome.jgi.doe.gov	15.01.2014
14546	Dictyocaulus viviparus	29172		na	03.04.2014
13212	Dictyostelium discoideum	44689	+*	ensemblgenomes.org	03.12.1013.
12394	Didymella exigua CBS 183.55	1150837		genome.jgi.doe.gov	15.01.2014
15948	Dioszegia cryoxerica	603311		genome.jgi.doe.gov	15.01.2014
15798	Dipodomys ordii	10020	+*	ensembl.org	02.12.2013
12857	Dirofilaria immitis	6287		na	03.04.2014
10299	Dissoconium aciculare	112489		genome.jgi.doe.gov	15.01.2014
12580	Dothistroma septosporum NZE10	675120	+	genome.jgi.doe.gov	15.01.2014
15069	Drosophila ananassae	7217	+*	ensemblgenomes.org	03.12.2013.
15044	Drosophila erecta	7220	+*	ensemblgenomes.org	03.12.2013.
14982	Drosophila grimshawi	7222	+*	ensemblgenomes.org	03.12.2013.
27227	Drosophila melanogaster	7227	+*	ensemblgenomes.org	03.12.2013.
14594	Drosophila mojavensis	7230	+*	ensemblgenomes.org	03.12.2013.
16874	Drosophila persimilis	7234	+*	ensemblgenomes.org	03.12.2013.
15876	Drosophila pseudoobscura pseudoobscura	46245	+*	ensemblgenomes.org	03.12.2013.
16467	Drosophila sechellia	7238	+*	ensemblgenomes.org	03.12.2013.
15413	Drosophila simulans	7240	+*	ensemblgenomes.org	03.12.2013.
14491	Drosophila virilis	7244	+*	ensemblgenomes.org	03.12.2013.
15512	Drosophila willistoni	7260	+*	ensemblgenomes.org	03.12.2013.
9779	Drosophila yakuba	7245	+*	ensemblgenomes.org	03.12.2013.
24441	Echinococcus granulosus	6210	+	na	03.04.2014
18213	Echinococcus multilocularis	6211	+	na	03.04.2014
16575	Echinops telfairi	9371	+*	ensembl.org	02.12.2013
16269	Ectocarpus siliculosus	2880		ncbi.nlm.nih.gov	04.05.2014
30623	Egretta garzetta	188379	+	na	03.04.2014

39125	Emiliania huxleyi CCMP1516	280463	+	genome.jgi.doe.gov	04.05.2014
1996	Encephalitozoon cuniculi GB-M1	284813	+	genome.jgi.doe.gov	15.01.2014
1847	Encephalitozoon hellem ATCC 50504	907965		genome.jgi.doe.gov	15.01.2014
1833	Encephalitozoon intestinalis ATCC 50506	876142		genome.jgi.doe.gov	15.01.2014
1831	Encephalitozoon romaleae SJ-2008	1178016	+	genome.jgi.doe.gov	15.01.2014
8113	Entamoeba histolytica HM-1 IMSS	294381	+*	ensemblgenomes.org	03.12.1013.
26052	Ephydatia muelleri	6052	+	na	03.04.2014
20449	Equus caballus	9796	*	ensembl.org	02.12.2013
14601	Erinaceus europaeus	9365	*	ensembl.org	02.12.2013
12290	Erythranthe guttata	4155	+	genome.jgi.doe.gov	04.05.2014
46315	Eucalyptus grandis	71139	+	genome.jgi.doe.gov	04.05.2014
27529	Eurypyga helias	54383	+	na	03.04.2014
29284	Eutrema halophilum	98038		genome.jgi.doe.gov	03.04.2014
11685	Eutypa lata UCREL1	1287681		genome.jgi.doe.gov	15.01.2014
26765	Exidia glandulosa	5219		genome.jgi.doe.gov	15.01.2014
13120	Exophiala aquamarina CBS 119918	1182545		broadinstitute.org	13.01.2014
16806	Falco cherrug	345164	+	na	03.04.2014
18954	Falco peregrinus	8954		na	03.04.2014
19493	Felis catus	9685	+*	ensembl.org	02.12.2013
9262	Fibroporia radiculosa TFFH 294	1078123		genome.jgi.doe.gov	15.01.2014
15303	Ficedula albicollis	59894	*	ensembl.org	02.12.2013
11244	Fistulina hepatica	40457		genome.jgi.doe.gov	15.01.2014
11333	Fomitiporia mediterranea	208960		genome.jgi.doe.gov	15.01.2014
13885	Fomitopsis pinicola FP-58527 SS1	743788		genome.jgi.doe.gov	15.01.2014
6289	Fonticula alba	691883	+	broadinstitute.org	09.12.2013
34274	Fopius arisanus	64838		na	03.04.2014
32831	Fragaria vesca	57918	+	genome.jgi.doe.gov	04.05.2014
28583	Fulmarus glacialis	30455	+	na	03.04.2014
14813	Fusarium fujikuroi IMI 58289	1279085		genome.jgi.doe.gov	15.01.2014
13317	Fusarium graminearum PH-1	229533	+*	ensemblgenomes.org	03.12.2013.
24739	Fusarium oxysporum CL57	660032		broadinstitute.org	19.12.2013
25216	Fusarium oxysporum Cotton	909454		broadinstitute.org	19.12.2013
24818	Fusarium oxysporum Fo47	660027		broadinstitute.org	19.12.2013
17696	Fusarium oxysporum f. sp. lycopersici 4287	426428	+*	ensemblgenomes.org	03.12.2013.
26719	Fusarium oxysporum f. sp. melonis 26406	1089452		broadinstitute.org	19.12.2013
26378	Fusarium oxysporum HDV247	909453		broadinstitute.org	19.12.2013
22487	Fusarium oxysporum II5	660034		broadinstitute.org	19.12.2013
24733	Fusarium oxysporum MN25	660028		broadinstitute.org	19.12.2013
26246	Fusarium oxysporum PHW808	660031		broadinstitute.org	19.12.2013
25666	Fusarium oxysporum PHW815	660033		broadinstitute.org	19.12.2013
14166	Fusarium verticillioides 7600	334819	*	ensemblgenomes.org	03.12.2013.
57136	Gadus morhua	8049	+*	ensembl.org	02.12.2013
14189	Gaeumannomyces graminis var. tritici R3-111a-1	644352	*	ensemblgenomes.org	03.12.2013.
21461	Galerina marginata	109633		genome.jgi.doe.gov	15.01.2014

15508	Gallus gallus	9031	+*	ensembl.org	02.12.2013
12910	Ganoderma sp. 10597 SS1	767862		genome.jgi.doe.gov	15.01.2014
20787	Gasterosteus aculeatus	69293	+*	ensembl.org	02.12.2013
26919	Gavia stellata	37040		na	03.04.2014
16433	Geospiza fortis	48883	+	na	03.04.2014
7364	Giardia lamblia ATCC 50803	184922	+*	ensemblgenomes.org	03.12.1013.
13083	Glarea lozoyensis ATCC 20868	1116229		genome.jgi.doe.gov	15.01.2014
11846	Gloeophyllum trabeum	104355		genome.jgi.doe.gov	15.01.2014
54174	Glycine max	3847	+*	ensemblgenomes.org	11.12.2013.
13902	Gonapodya prolifera	1123529	+	genome.jgi.doe.gov	15.01.2014
20962	Gorilla gorilla gorilla	9595	+*	ensembl.org	02.12.2013
77267	Gossypium raimondii	29730		genome.jgi.doe.gov	04.05.2014
8312	Grosmannia clavigera kw1407	655863		genome.jgi.doe.gov	15.01.2014
24840	Guillardia theta CCMP2712	905079	+	ensemblgenomes.org	03.12.1013.
9106	Gymnascella aurantiaca	78594		genome.jgi.doe.gov	15.01.2014
9779	Gymnascella citrina	37245		genome.jgi.doe.gov	15.01.2014
22057	Gymnopus luxurians	206324		genome.jgi.doe.gov	15.01.2014
24775	Haemonchus contortus	6289		Wormbase	03.04.2014
18969	Haliaeetus albicilla	8969		na	03.04.2014
25311	Haliaeetus leucocephalus	52644		na	03.04.2014
4800	Hanseniaspora valbyensis NRRL Y-1626	766949	+	genome.jgi.doe.gov	15.01.2014
18564	Harpegnathos saltator	610380	+	hymenopteragenome.org	03.04.2014
15382	Hebeloma cylindrosporum h7	686832	+	genome.jgi.doe.gov	15.01.2014
12452	Helianthus annuus	4232	+	na	03.04.2014
12669	Heliconius melpomene	34740	+*	ensemblgenomes.org	03.12.2013.
16526	Helobdella robusta	6412	+*	ensemblgenomes.org	03.12.2013.
13405	Heterobasidion annosum	13563		genome.jgi.doe.gov	15.01.2014
20964	Heterorhabditis bacteriophora	37862	+	Wormbase	03.04.2014
9233	Histoplasma capsulatum G186AR	447093		broadinstitute.org	19.12.2013
9532	Histoplasma capsulatum H143	544712		broadinstitute.org	19.12.2013
9428	Histoplasma capsulatum H88	544711		broadinstitute.org	19.12.2013
9251	Histoplasma capsulatum NAm1	339724		genome.jgi.doe.gov	15.01.2014
23264	Homo sapiens	9606	+*	ensembl.org	02.12.2013
24211	Hordeum vulgare subsp. vulgare	112509	+*	ensemblgenomes.org	11.12.2013.
14321	Hyaloperonospora arabidopsidis Emoy2	559515	+*	ensemblgenomes.org	03.12.1013.
14565	Hyaloperonospora parasitica	123356	+	na	03.04.2014
13270	Hydnomerulius pinastri	388859		genome.jgi.doe.gov	15.01.2014
20881	Hydra oligactis	6088	+	na	03.04.2014
22725	Hydra viridissima	6082	+	na	03.04.2014
32338	Hydra vulgaris	6087	+	na	03.04.2014
17911	Hypholoma sublateritium	71945		genome.jgi.doe.gov	15.01.2014
6005	Hyphopichia burtonii NRRL Y-1933	984485		genome.jgi.doe.gov	15.01.2014
11712	Hypoxylon sp. CI-4A	1001833		genome.jgi.doe.gov	15.01.2014
12256	Hypoxylon sp. CO27-5	1001938		genome.jgi.doe.gov	15.01.2014

12261	Hypoxylon sp. EC38	1001937	+	genome.jgi.doe.gov	15.01.2014
12352	Hysterium pulicare	100027		genome.jgi.doe.gov	15.01.2014
18826	Ictidomys tridecemlineatus	43179	*	ensembl.org	02.12.2013
20486	Ixodes scapularis	6945	*	ensemblgenomes.org	03.12.2013.
16419	Jaapia argillacea	202697		genome.jgi.doe.gov	15.01.2014
57136	Jatropha curcas	180498	+	na	03.04.2014
10756	Kazachstania africana CBS 2517	1071382	+	na	03.04.2014
5321	Kazachstania naganishii CBS 8797	1071383		na	03.04.2014
5076	Kluyveromyces lactis	28985	+	genome.jgi.doe.gov	15.01.2014
4952	Kluyveromyces marxianus DMKU3-1042	1003335	+	na	03.04.2014
5040	Komagataella pastoris GS115	644223	+*	ensemblgenomes.org	03.12.2013.
21066	Laccaria amethystina LaAM-08-1	1095629		genome.jgi.doe.gov	15.01.2014
29883	Laccaria bicolor	29883		genome.jgi.doe.gov	15.01.2014
10326	Lachancea thermotolerans CBS 6340	559295	+	na	03.04.2014
13774	Laetiporus sulphureus var. sulphureus	447506		genome.jgi.doe.gov	15.01.2014
19569	Latimeria chalumnae	7897	+*	ensembl.org	02.12.2013
29078	Leersia perrieri	77586	+*	na	03.04.2014
8308	Leishmania major strain Friedlin	347515	+*	ensemblgenomes.org	03.12.1013.
15581	Lentinus tigrinus ALCF2SS1-6	1328759		genome.jgi.doe.gov	15.01.2014
15380	Lentinus tigrinus ALCF2SS1-7	1328758		genome.jgi.doe.gov	15.01.2014
16742	Lentithecium fluviatile	690899		genome.jgi.doe.gov	15.01.2014
13870	Lepidopterella palustris	741139		genome.jgi.doe.gov	15.01.2014
32831	Lepisosteus oculatus	7918	+*	ensembl.org	11.12.2013
29042	Leptosomus discolor	188344	+	na	03.04.2014
12469	Leptosphaeria maculans	5022		genome.jgi.doe.gov	15.01.2014
12469	Leptosphaeria maculans JN3	985895	*	ensemblgenomes.org	03.12.2013.
5420	Leucoagaricus gongylophorus Ac12	1258663		genome.jgi.doe.gov	15.01.2014
92106	Leucosolenia complicata	433461	+	na	03.04.2014
12062	Lichtheimia hyalospora	420593	+	genome.jgi.doe.gov	15.01.2014
16116	Linepithema humile	83485	+	hymenopteragenome.org	03.04.2014
29883	Linum usitatissimum	4006	+*	genome.jgi.doe.gov	04.05.2014
8192	Lipomyces starkeyi NRRL Y-11557	675824	+	genome.jgi.doe.gov	15.01.2014
14908	Loa loa	7209	+*	ensemblgenomes.org	03.12.2013.
5799	Lodderomyces elongisporus NRRL YB-4239	379508		broadinstitute.org	03.04.2014
16160	Lophiostoma macrostomum	372055	+	genome.jgi.doe.gov	15.01.2014
23340	Lottia gigantea	225164	+*	ensemblgenomes.org	03.12.2013.
15030	Lotus japonicus	34305	+	na	03.04.2014
20033	Loxodonta africana	9785	+*	ensembl.org	02.12.2013
21905	Macaca mulatta	9544	+*	ensembl.org	02.12.2013
15801	Macrolepiota fuliginosa	201230	+	genome.jgi.doe.gov	15.01.2014
13806	Macrophomina phaseolina MS6	1126212	+	genome.jgi.doe.gov	15.01.2014
15290	Macropus eugenii	9315	+*	ensembl.org	02.12.2013
11054	Magnaporthe grisea	148305		genome.jgi.doe.gov	15.01.2014
12593	Magnaporthe oryzae 70-15	242507	+*	ensemblgenomes.org	03.12.2013.

11209	Magnaporthiopsis poae ATCC 64411	644358	*	ensemblgenomes.org	03.12.2013.
4286	Malassezia globosa	76773	+	genome.jgi.doe.gov	15.01.2014
3517	Malassezia sympodialis ATCC 42132	1230383		genome.jgi.doe.gov	15.01.2014
63517	Malus domestica	3750	+	genome.jgi.doe.gov	04.05.2014
28530	Manacus vitellinus	328815		na	03.04.2014
34151	Manihot esculenta	3983	+	genome.jgi.doe.gov	04.05.2014
44115	Medicago truncatula	3880	+*	ensemblgenomes.org	11.12.2013.
26046	Megachile rotundata	143995	+	na	03.04.2014
11461	Megaselia scalaris	36166	+*	ensemblgenomes.org	03.12.2013.
16372	Melampsora larici-populina 98AG31	747676	*	ensemblgenomes.org	03.12.2013.
16656	Melanconium sp. NRRL 54901	1155951	+	genome.jgi.doe.gov	15.01.2014
15881	Melanomma pulvis-pyrius	100047		genome.jgi.doe.gov	15.01.2014
14125	Meleagris gallopavo	9103	+*	ensembl.org	02.12.2013
18619	Meliniomyces bicolor E	1095630		genome.jgi.doe.gov	15.01.2014
20389	Meliniomyces variabilis F	1149755		genome.jgi.doe.gov	15.01.2014
14420	Meloidogyne hapla	6305		Wormbase	03.04.2014
20359	Meloidogyne incognita	6306	+	www6.inra.fr	03.04.2014
15974	Melopsittacus undulatus	13146		na	03.04.2014
26927	Merops nubicus	57421		na	03.04.2014
29585	Mesitornis unicolor	54374		na	03.04.2014
9849	Metarhizium acridum CQMa 102	655827		genome.jgi.doe.gov	15.01.2014
10583	Metarhizium robertsii	568076		genome.jgi.doe.gov	15.01.2014
5851	Metschnikowia bicuspidata var. bicuspidata NRRL YB-4993	869754		genome.jgi.doe.gov	15.01.2014
7364	Microbotryum violaceum p1A1 Lamole	683840	*	ensemblgenomes.org	03.12.2013.
16319	Microcebus murinus	30608	*	ensembl.org	02.12.2013
10660	Micromonas pusilla CCMP1545	564608		genome.jgi.doe.gov	15.01.2014
10107	Micromonas sp. RCC299	296587	+	genome.jgi.doe.gov	15.01.2014
19916	Microplitis demolitor	69319	+	na	03.04.2014
8907	Microsporum gypseum CBS 118893	535722		broadinstitute.org	19.12.2013
6903	Mixia osmundae IAM 14324	764103	+	genome.jgi.doe.gov	15.01.2014
16548	Mnemiopsis leidyi	27923	+	genome.gov	17.01.2014
8918	Monascus purpureus	5098		genome.jgi.doe.gov	15.01.2014
9650	Monascus ruber NRRL 1597	1155947		genome.jgi.doe.gov	15.01.2014
21327	Monodelphis domestica	13616	*	ensembl.org	02.12.2013
9196	Monosiga brevicollis	81824	+	genome.jgi.doe.gov	15.01.2014
15976	Mortierella elongata	310910	+	genome.jgi.doe.gov	15.01.2014
12569	Mortierella verticillata NRRL 6337	1069443		broadinstitute.org	09.12.2013
53984	Morus notabilis	981085	+	na	03.04.2014
12227	Mucor circinelloides B8987	1274786	+	na	03.04.2014
12227	Mucor circinelloides f. circinelloides 1006PhL	1220926		broadinstitute.org	09.01.2014
11719	Mucor circinelloides f. lusitanicus CBS 277.49	747725		genome.jgi.doe.gov	15.01.2014
36519	Musa acuminata subsp. malaccensis	214687	+*	ensemblgenomes.org	11.12.2013.
22780	Mus musculus	10090	+*	ensembl.org	02.12.2013
19910	Mustela putorius furo	9669	+*	ensembl.org	02.12.2013

9110	Myceliophthora thermophila	78579	+	genome.jgi.doe.gov	15.01.2014
19728	Myotis lucifugus	59463	*	ensembl.org	02.12.2013
10685	Myriangium duriaei CBS 260.36	1168546	+	genome.jgi.doe.gov	15.01.2014
5657	Nadsonia fulvescens var. elongata DSM 6958	857566		genome.jgi.doe.gov	15.01.2014
15753	Naegleria gruberi	5762	+	genome.jgi.doe.gov	04.05.2014
9053	Nannochloropsis gaditana CCMP526	1093141	+	www.nannochloropsis.org	04.05.2014
17085	Nasonia vitripennis	7425	*	ensemblgenomes.org	03.12.2013.
11199	Naumovozyma castellii CBS 4309	1064592	+	na	03.04.2014
11097	Naumovozyma dairenensis CBS 421	1071378		na	03.04.2014
19153	Necator americanus	51031	+	na	03.04.2014
15705	Nectria haematococca mpVI 77-13-4	660122	*	ensemblgenomes.org	03.12.2013.
13940	Nelumbo nucifera	4432	+	na	03.04.2014
2661	Nematocida parisii ERTm1	881290		genome.jgi.doe.gov	15.01.2014
24976	Nematostella vectensis	45351	+*	ensemblgenomes.org	03.12.2013.
10366	Neofusicoccum parvum UCRNP2	1287680	+	genome.jgi.doe.gov	15.01.2014
13164	Neolentinus lepideus	38799		genome.jgi.doe.gov	15.01.2014
10401	Neosartorya fischeri NRRL 181	331117	*	ensemblgenomes.org	03.12.2013.
27937	Nestor notabilis	176057	+	na	03.04.2014
9820	Neurospora crassa OR74A	367110	+*	ensemblgenomes.org	03.12.2013.
9948	Neurospora discreta FGSC 8579	510953	+	genome.jgi.doe.gov	15.01.2014
10380	Neurospora tetrasperma FGSC 2508	510951	+	genome.jgi.doe.gov	15.01.2014
11192	Neurospora tetrasperma FGSC 2509	510952	+	genome.jgi.doe.gov	15.01.2014
76379	Nicotiana benthamiana	4100	+	genome.jgi.doe.gov	04.05.2014
31677	Nipponia nippon	128390		na	03.04.2014
18575	Nomascus leucogenys	61853	+*	ensembl.org	02.12.2013
16006	Ochotona princeps	9978	*	ensembl.org	02.12.2013
25453	Oesophagostomum dentatum	61180	+	na	03.04.2014
5177	Ogataea angusta NCYC 495 leu1.1	638633	+	genome.jgi.doe.gov	15.01.2014
16703	Oidiodendron maius Zn	913774		genome.jgi.doe.gov	15.01.2014
17212	Oikopleura dioica	34765	+	OikoBase	03.04.2014
8171	Omphalotus olearius	72120		genome.jgi.doe.gov	15.01.2014
12169	Onchocerca volvulus	6282	+	na	03.04.2014
18637	Ophiophagus hannah	8665	+	ncbi.nlm.nih.gov	17.01.2014
8884	Ophiostoma piceae UAMH 11346	1262450		genome.jgi.doe.gov	15.01.2014
28085	Opisthocomus hoazin	30419		na	03.04.2014
32893	Opisthorchis viverrini	6198	+	na	03.04.2014
21437	Oreochromis niloticus	8128	+*	ensembl.org	02.12.2013
21698	Ornithorhynchus anatinus	9258	+*	ensembl.org	02.12.2013
19293	Orussus abietinus	222816	+	na	03.04.2014
19293	Oryctolagus cuniculus	9986	+*	ensembl.org	02.12.2013
34575	Oryza barthii	65489	*	na	03.04.2014
32037	Oryza brachyantha	4533	+*	ensemblgenomes.org	11.12.2013.
33164	Oryza glaberrima	4538	*	ensemblgenomes.org	11.12.2013.
35735	Oryza glumipatula	40148	*	na	03.04.2014

28413	Oryza longistaminata	4528	*	na	03.04.2014
29308	Oryza meridionalis	40149	*	na	03.04.2014
36313	Oryza nivara	4536	*	na	03.04.2014
31762	Oryza punctata	4537	*	na	03.04.2014
37071	Oryza rufipogon	4529	*	na	03.04.2014
40745	Oryza sativa Indica Group	39946	+*	ensemblgenomes.org	11.12.2013.
35681	Oryza sativa Japonica Group	39947	*	ensemblgenomes.org	11.12.2013.
19699	Oryzias latipes	8090	+*	ensembl.org	02.12.2013
11152	Oscarella carmela	386100	+	compagen.zoologie.uni-kiel.de	30.06.2014
7796	Ostreococcus 'lucimarinus'	242159	+	ftp.jgi-psf.org	15.01.2014
7725	Ostreococcus tauri	70448		genome.jgi.doe.gov	04.05.2014
19506	Otolemur garnettii	30611	+*	ensembl.org	02.12.2013
20921	Ovis aries	9940	+*	ensembl.org	11.12.2013
5675	Pachysolen tannophilus NRRL Y-2460	669874	+	genome.jgi.doe.gov	15.01.2014
26372	Panagrellus redivivus	6233		na	03.04.2014
70071	Panicum virgatum	38727	+	genome.jgi.doe.gov	04.05.2014
18759	Pan troglodytes	9598	+*	ensembl.org	02.12.2013
7876	Paracoccidioides brasiliensis Pb03	482561		genome.jgi.doe.gov	15.01.2014
8741	Paracoccidioides brasiliensis Pb18	502780		broadinstitute.org	19.12.2013
9136	Paracoccidioides sp. 'lutzii' Pb01	502779	+	broadinstitute.org	19.12.2013
39642	Paramecium tetraurelia	5888	+*	ensemblgenomes.org	03.12.1013.
14127	Passalora fulva	5499		genome.jgi.doe.gov	15.01.2014
9794	Patellaria atrata	703506		genome.jgi.doe.gov	15.01.2014
17968	Paxillus involutus ATCC 200175	664439		genome.jgi.doe.gov	15.01.2014
22065	Paxillus rubicundulus Ve08.2h10	930991		genome.jgi.doe.gov	15.01.2014
10772	Pediculus humanus corporis	121224	+*	ensemblgenomes.org	03.12.2013.
28872	Pelecanus crispus	36300	+	na	03.04.2014
18188	Pelodiscus sinensis	13735	+*	ensembl.org	02.12.2013
13606	Penicillium bilaiae ATCC 20851	1314792		genome.jgi.doe.gov	15.01.2014
11536	Penicillium brevicompactum	5074	+	genome.jgi.doe.gov	15.01.2014
12374	Penicillium canescens ATCC 10419	1314794		genome.jgi.doe.gov	15.01.2014
5076	Penicillium chrysogenum	5076	+	genome.jgi.doe.gov	15.01.2014
9118	Penicillium digitatum PHI26	1170229		genome.jgi.doe.gov	15.01.2014
11807	Penicillium expansum ATCC 24692	1314791		genome.jgi.doe.gov	15.01.2014
11300	Penicillium fellutanum ATCC 48694	1314795		genome.jgi.doe.gov	15.01.2014
12328	Penicillium glabrum DAOM 239074	1314793		genome.jgi.doe.gov	15.01.2014
10698	Penicillium lanosocoeruleum ATCC 48919	1346256		genome.jgi.doe.gov	15.01.2014
13671	Penicillium rubens Wisconsin 54-1255	500485		genome.jgi.doe.gov	15.01.2014
21758	Perkinsus marinus ATCC 50983	423536	+	Ruiz trillo group	04.05.2014
10415	Petromyzon marinus	7757	+*	ensembl.org	02.12.2013
10402	Phaeodactylum tricornutum CCAP 1055/1	556484	+*	ensemblgenomes.org	03.12.1013.
12391	Phaeosphaeria nodorum SN15	321614	+*	ensemblgenomes.org	03.12.2013.
28930	Phaethon lepturus	97097	+	na	03.04.2014
26339	Phalacrocorax carbo	9209	+	na	03.04.2014

13937	Phanerochaete carnosa HHB-10118-sp	650164	+	genome.jgi.doe.gov	15.01.2014
31638	Phaseolus vulgaris	3885	+	genome.jgi.doe.gov	04.05.2014
16170	Phlebia brevispora HHB-7030 SS6	747484	+	genome.jgi.doe.gov	15.01.2014
11891	Phlebiopsis gigantea	82310		genome.jgi.doe.gov	15.01.2014
12143	Phoenicopterus ruber ruber	9218		na	03.04.2014
16528	Phycomyces blakesleeanus	4837		genome.jgi.doe.gov	15.01.2014
38354	Physcomitrella patens	3218	+	ftp.jgi-psf.org	15.01.2014
19805	Phytophthora capsici LT1534	763924		genome.jgi.doe.gov	15.01.2014
26131	Phytophthora cinnamomi var. cinnamomi	622258		genome.jgi.doe.gov	15.01.2014
17785	Phytophthora infestans T30-4	403677	+*	ensemblgenomes.org	03.12.1013.
20822	Phytophthora parasitica INRA-310	761204		broadinstitute.org	23.12.2013
15605	Phytophthora ramorum	164328	+*	ensemblgenomes.org	03.12.1013.
18969	Phytophthora sojae	67593	*	ensemblgenomes.org	03.12.1013.
13487	Picea abies	3329	+*	congenie.org	20.01.2014.
23605	Picea sitchensis	3332	+	loblolly.ucdavis.edu	04.05.2014
5546	Pichia membranifaciens	4926	+	genome.jgi.doe.gov	15.01.2014
29768	Picoides pubescens	118200		na	03.04.2014
7572	Piedraia hortae var. hortae	147573	+	genome.jgi.doe.gov	15.01.2014
21583	Piloderma croceum F 1598	765440	+	genome.jgi.doe.gov	15.01.2014
34059	Pinus taeda	3352	+	na	03.04.2014
11767	Piriformospora indica DSM 11827	1109443	+	genome.jgi.doe.gov	15.01.2014
14648	Piromyces sp. E2	73868	+	genome.jgi.doe.gov	15.01.2014
21064	Pisolithus microcarpus 441	765257	+	genome.jgi.doe.gov	15.01.2014
22701	Pisolithus tinctorius Marx 270	870435		genome.jgi.doe.gov	15.01.2014
4855	Plasmodium berghei ANKA	5823	*	ensemblgenomes.org	03.12.1013.
5119	Plasmodium chabaudi	5825	+*	ensemblgenomes.org	03.12.1013.
5349	Plasmodium falciparum 3D7	36329	*	broadinstitute.org	05.01.2014
6314	Plasmodium falciparum 7G8	57266		broadinstitute.org	23.12.2013
6118	Plasmodium falciparum CAMP/Malaysia	5835		broadinstitute.org	03.01.2014
5139	Plasmodium falciparum Dd2	57267		broadinstitute.org	05.01.2014
5770	Plasmodium falciparum FCH/4	1036724		broadinstitute.org	03.01.2014
5462	Plasmodium falciparum HB3	137071		broadinstitute.org	05.01.2014
5041	Plasmodium falciparum IGH-CR14	580059		broadinstitute.org	05.01.2014
6317	Plasmodium falciparum MaliPS096 E11	1036727		broadinstitute.org	03.01.2014
6349	Plasmodium falciparum NF135/5.C10	1036726		broadinstitute.org	03.01.2014
5936	Plasmodium falciparum NF54	5843		broadinstitute.org	03.01.2014
6048	Plasmodium falciparum Palo Alto/Uganda	57270	+	broadinstitute.org	03.01.2014
3180	Plasmodium falciparum RAJ116	580058		broadinstitute.org	05.01.2014
6195	Plasmodium falciparum Santa Lucia	478859		broadinstitute.org	03.01.2014
6719	Plasmodium falciparum Tanzania <2000708>	1036725		broadinstitute.org	23.12.2013
5922	Plasmodium falciparum UGT5.1	1237627		broadinstitute.org	23.12.2013
6234	Plasmodium falciparum Vietnam Oak-Knoll <FVO>	1036723		broadinstitute.org	03.01.2014
5832	Plasmodium inui San Antonio 1	1237626		broadinstitute.org	05.01.2014
5102	Plasmodium knowlesi strain H	5851	*	ensemblgenomes.org	03.12.1013.

5160	Plasmodium vinckei petteri	138298		broadinstitute.org	05.01.2014
4954	Plasmodium vinckei vinckei	54757		broadinstitute.org	05.01.2014
6435	Plasmodium vivax Brazil I	1033975		broadinstitute.org	05.01.2014
6618	Plasmodium vivax India VII	1077284		broadinstitute.org	05.01.2014
6353	Plasmodium vivax Mauritania I	1035515		broadinstitute.org	05.01.2014
6667	Plasmodium vivax North Korean	1035514		broadinstitute.org	05.01.2014
5049	Plasmodium vivax Sal-1	126793		broadinstitute.org	05.01.2014
7508	Plasmodium yoelii 17X	1323249		broadinstitute.org	05.01.2014
13486	Pleomassaria siparia	100044		genome.jgi.doe.gov	15.01.2014
19524	Pleurobrachia bachei	34499	+	na	03.04.2014
12330	Pleurotus ostreatus PC15	1137138		genome.jgi.doe.gov	15.01.2014
12206	Pleurotus ostreatus PC9	1137139		genome.jgi.doe.gov	15.01.2014
13626	Plicaturopsis crispa	139390		genome.jgi.doe.gov	15.01.2014
3520	Pneumocystis jirovecii	42068	+	genome.jgi.doe.gov	15.01.2014
11753	Podiceps cristatus	345573		na	03.04.2014
10639	Podospora anserina S mat+	515849	+	podospora.igmors.u-psud.fr	03.04.2014
17189	Pogonomyrmex barbatus	144034		hymenopteragenome.org	03.04.2014
10582	Polychaeton citri	705562		genome.jgi.doe.gov	15.01.2014
18229	Polyporus arcularius	5639	+	genome.jgi.doe.gov	15.01.2014
12356	Polysphondylium pallidum PN500	670386	+	dictybase.org	03.04.2014
20424	Pongo abelii	9601	+*	ensembl.org	02.12.2013
73013	Populus trichocarpa	3694		ensemblgenomes.org	11.12.2013.
9113	Postia placenta Mad-698-R	561896		genome.jgi.doe.gov	15.01.2014
12541	Postia placenta MAD-698-R-SB12	670580	+	genome.jgi.doe.gov	15.01.2014
24642	Pristionchus exspectatus	1195656		na	03.04.2014
29644	Pristionchus pacificus	54126	+*	ensemblgenomes.org	03.12.2013.
9813	Procavia capensis	9813	+*	ensembl.org	02.12.2013
28923	Prunus mume	102107	+	na	03.04.2014
28701	Prunus persica	3760		genome.jgi.doe.gov	04.05.2014
13107	Pseudocercospora fijiensis	83344		genome.jgi.doe.gov	15.01.2014
9153	Pseudogymnoascus destructans 20631-21	658429		na	03.04.2014
19703	Pseudo-nitzschia multiseries	37319		genome.jgi.doe.gov	15.01.2014
18654	Pseudopodoces humilis	181119	+	na	03.04.2014
6543	Pseudozyma antarctica T-34	1151754	+	genome.jgi.doe.gov	15.01.2014
7472	Pseudozyma hubeiensis SY62	1305764		genome.jgi.doe.gov	15.01.2014
27227	Pterocles gutturalis	240206	+	na	03.04.2014
16990	Pteropus vampyrus	132908	+*	ensembl.org	02.12.2013
15800	Puccinia graminis f. sp. tritici CRL 75-36-700-3	418459	*	ensemblgenomes.org	03.12.2013.
15979	Puccinia graminis f. tritici	413621		broadinstitute.org	09.01.2014
18021	Puccinia striiformis f. sp. tritici PST-130	875184		genome.jgi.doe.gov	15.01.2014
20482	Puccinia striiformis f. sp. tritici PST-78	1165861		broadinstitute.org	09.01.2014
11638	Puccinia triticina 1-1 BBBD Race 1	630390	+*	ensemblgenomes.org	03.12.2013.
11538	Punctularia strigosozonata	202698	+	genome.jgi.doe.gov	15.01.2014
30341	Pygoscelis adeliae	9238		na	03.04.2014

11799	Pyrenophora teres f. teres 0-1	861557	+*	ensemblgenomes.org	03.12.2013.
12169	Pyrenophora tritici-repentis Pt-1C-BFP	426418	*	ensemblgenomes.org	03.12.2013.
13367	Pyronema omphalodes CBS 100304	1076935		genome.jgi.doe.gov	15.01.2014
46528	Pyrus x bretschneideri	225117		na	03.04.2014
15290	Pythium ultimum DAOM BR144	431595	*	ensemblgenomes.org	03.12.1013.
12614	Pythium ultimum var. sporangiiferum	115421		na	03.04.2014
25922	Python bivittatus	176946	+	ncbi.nlm.nih.gov	07.04.14
22941	Rattus norvegicus	10116	+*	ensembl.org	02.12.2013
15157	Rhizoctonia solani AG-1 IB	1108050		genome.jgi.doe.gov	15.01.2014
30282	Rhizophagus irregularis DAOM 181602	747089		genome.jgi.doe.gov	15.01.2014
17459	Rhizopus delemar RA 99-880	246409		broadinstitute.org	09.01.2014
17676	Rhizopus microsporus var. chinensis CCTCC M201021	1271458		genome.jgi.doe.gov	15.01.2014
10905	Rhizopus microsporus var. microsporus	86635		genome.jgi.doe.gov	15.01.2014
17467	Rhizopus oryzae	64495	+	genome.jgi.doe.gov	15.01.2014
15429	Rhodnius prolixus	13249	*	ensemblgenomes.org	03.12.2013.
7283	Rhodotorula graminis WP1	578459		genome.jgi.doe.gov	15.01.2014
12117	Rhytidhysteron rufulum	37885		genome.jgi.doe.gov	15.01.2014
31221	Ricinus communis	3988	+	genome.jgi.doe.gov	04.05.2014
18952	Rickenella mellea	50990		genome.jgi.doe.gov	15.01.2014
6350	Rozella allomycis CSF55	988480	+	genome.jgi.doe.gov	15.01.2014
7318	Saccharomyces arboricola H-6	1160507	+	na	03.04.2014
5974	Saccharomyces cerevisiae M3707	1149757		genome.jgi.doe.gov	15.01.2014
5984	Saccharomyces cerevisiae M3836	1162671		genome.jgi.doe.gov	15.01.2014
5969	Saccharomyces cerevisiae M3837	1162672		genome.jgi.doe.gov	15.01.2014
5991	Saccharomyces cerevisiae M3838	1162673		genome.jgi.doe.gov	15.01.2014
5989	Saccharomyces cerevisiae M3839	1162674	+	genome.jgi.doe.gov	15.01.2014
5381	Saccharomyces cerevisiae RM11-1a	285006		na	03.04.2014
6692	Saccharomyces cerevisiae S288c	559292	+*	ensemblgenomes.org	03.12.2013.
5994	Saccharomyces cerevisiae YB210	927258		genome.jgi.doe.gov	15.01.2014
3780	Saccharomyces kudriavzevii IFO 1802	226230	+	na	03.04.2014
34239	Saccoglossus kowalevskii	10224	+	ftp.jgi-psf.org	15.01.2014
7034	Saitoella complicata NRRL Y-17804	698492	+	genome.jgi.doe.gov	15.01.2014
11731	Salpingoeca rosetta	946362	+	broadinstitute.org	09.12.2013
18229	Saprolegnia diclina VS20	1156394	+	na	03.04.2014
20088	Saprolegnia parasitica CBS 223.65	695850		na	03.04.2014
8707	Sarcophilus harrisii	9305	*	ensembl.org	02.12.2013
5807	Scheffersomyces stipitis	4924		genome.jgi.doe.gov	15.01.2014
13469	Schistosoma japonicum	6182	+	www.chgc.sh.cn	03.04.2014
10627	Schistosoma mansoni	6183	+*	ensemblgenomes.org	03.12.2013.
10612	Schizochytrium aggregatum ATCC 28209	876976	+	genome.jgi.doe.gov	15.01.2014
16319	Schizophyllum commune H4-8	578458		genome.jgi.doe.gov	15.01.2014
13827	Schizophyllum commune Loenen D	1314666		genome.jgi.doe.gov	15.01.2014
15199	Schizophyllum commune Tattone D	1314663		genome.jgi.doe.gov	15.01.2014
5180	Schizosaccharomyces cryophilus OY26	653667	+	genome.jgi.doe.gov	15.01.2014

4878	Schizosaccharomyces japonicus yFS275	402676		genome.jgi.doe.gov	15.01.2014
4986	Schizosaccharomyces octosporus yFS286	483514		genome.jgi.doe.gov	15.01.2014
5143	Schizosaccharomyces pombe 972h-	284812	+*	ensemblgenomes.org	03.12.2013.
21012	Scleroderma citrinum Foug A	1036808		genome.jgi.doe.gov	15.01.2014
10175	Sclerotinia sclerotiorum 1980 UF-70	665079	*	ensemblgenomes.org	03.12.2013.
34799	Selaginella moellendorffii	88036	+*	ensemblgenomes.org	11.12.2013.
15312	Serendipita vermifera MAFF 305830	933852		genome.jgi.doe.gov	15.01.2014
17921	Serinus canaria	9135		na	03.04.2014
14495	Serpula lacrymans var. lacrymans S7.3	936435		genome.jgi.doe.gov	15.01.2014
12789	Serpula lacrymans var. lacrymans S7.9	578457	+	genome.jgi.doe.gov	15.01.2014
13805	Serpula lacrymans var. shastensis SHA21-2	690234	+	genome.jgi.doe.gov	15.01.2014
35471	Setaria italica	4555	+*	ensemblgenomes.org	11.12.2013.
11702	Setosphaeria turcica Et28A	671987		genome.jgi.doe.gov	15.01.2014
15653	Sistotrema brinkmannii HHB7604 ss-1	1328757	+	genome.jgi.doe.gov	15.01.2014
13657	Sistotremastrum suecicum	467971		genome.jgi.doe.gov	15.01.2014
9411	Sodiomyces alkalinus	1302862		genome.jgi.doe.gov	15.01.2014
34675	Solanum lycopersicum	4081	+*	ensemblgenomes.org	11.12.2013.
39021	Solanum tuberosum	4113	+*	ensemblgenomes.org	11.12.2013.
16522	Solenopsis invicta	13686		hymenopteragenome.org	03.04.2014
13187	Sorex araneus	42254	+*	ensembl.org	02.12.2013
34496	Sorghum bicolor	4558	+*	ensemblgenomes.org	11.12.2013.
5983	Spathaspora passalidarum NRRL Y-27907	619300		genome.jgi.doe.gov	15.01.2014
35274	Sphaerobolus stellatus	68786	+	genome.jgi.doe.gov	15.01.2014
18730	Sphaeroforma arctica JP610	667725	+	broadinstitute.org	09.12.2013
10233	Sphaerulina musiva SO2202	692275		genome.jgi.doe.gov	15.01.2014
9739	Sphaerulina populicola	215467	+	na	03.04.2014
9424	Spizellomyces punctatus DAOM BR117	645134		broadinstitute.org	09.12.2013
6673	Sporisorium reilianum SRZ2	999809	*	ensemblgenomes.org	03.12.2013.
5536	Sporobolomyces roseus	40563		genome.jgi.doe.gov	15.01.2014
10783	Sporormia fimetaria	718229		genome.jgi.doe.gov	15.01.2014
14072	Stereum hirsutum FP-91666 SS1	721885		genome.jgi.doe.gov	15.01.2014
14992	Strigamia maritima	126957	+*	ensemblgenomes.org	03.12.2013.
28842	Strongylocentrotus purpuratus	7668	+*	ensemblgenomes.org	03.12.2013.
8188	Strongyloides ratti	34506	+	Wormbase	03.04.2014
40016	Struthio camelus australis	441894	+	na	03.04.2014
22453	Suillus brevipes	48565		genome.jgi.doe.gov	15.01.2014
18316	Suillus luteus UH-Slu-Lm8-n1	930992		genome.jgi.doe.gov	15.01.2014
21630	Sus scrofa	9823	*	ensembl.org	02.12.2013
50731	Sycon ciliatum	27933	+	na	03.04.2014
10482	Symbiotaphrina kochii	40221		genome.jgi.doe.gov	15.01.2014
17488	Taeniopygia guttata	59729	+*	ensembl.org	02.12.2013
18523	Takifugu rubripes	31033	*	ensembl.org	02.12.2013
10638	Talaromyces marneffei ATCC 18224	441960		genome.jgi.doe.gov	15.01.2014
13252	Talaromyces stipitatus ATCC 10500	441959	+	genome.jgi.doe.gov	15.01.2014

14068	Taphrina deformans	5011	+*	genome.jgi.doe.gov	15.01.2014
13628	Tarsius syrichta	9478	+*	ensembl.org	02.12.2013
28832	Tauraco erythrolophus	121530	+	na	03.04.2014
21943	Tetrahymena borealis	5893	+	broadinstitute.org	09.01.2014
22562	Tetrahymena elliotti 4EA	1075773	+	broadinstitute.org	09.01.2014
26378	Tetrahymena malaccensis 436	1075772		broadinstitute.org	09.01.2014
24725	Tetrahymena thermophila SB210	312017	*	ensemblgenomes.org	03.12.1013.
26497	Tetramorium validiusculum	411796		na	03.04.2014
18224	Tetranychus urticae	32264	+*	ensemblgenomes.org	03.12.2013.
19602	Tetraodon nigroviridis	99883	+*	ensembl.org	02.12.2013
11674	Thalassiosira pseudonana CCMP1335	296543	+*	ensemblgenomes.org	03.12.1013.
10627	Thecamonas trahens ATCC 50062	461836	+	broadinstitute.org	09.12.2013
4079	Theileria parva strain Muguga	333668		ncbi.nlm.nih.gov	04.05.2014
44404	Theobroma cacao	3641	+	genome.jgi.doe.gov	04.05.2014
7988	Thermoascus aurantiacus	5087	+*	genome.jgi.doe.gov	15.01.2014
9813	Thielavia terrestris	35720		genome.jgi.doe.gov	15.01.2014
31508	Tinamus guttatus	94827		na	03.04.2014
8834	Togninia minima UCRPA7	1286976		genome.jgi.doe.gov	15.01.2014
4657	Tortispora caseinolytica NRRL Y-17796	767744		genome.jgi.doe.gov	15.01.2014
7988	Toxoplasma gondii ME49	508771	+*	ensemblgenomes.org	03.12.1013.
14296	Trametes versicolor	5325		genome.jgi.doe.gov	15.01.2014
8313	Tremella mesenterica	5217	+	genome.jgi.doe.gov	15.01.2014
16524	Tribolium castaneum	7070	+*	ensemblgenomes.org	03.12.2013.
14978	Trichaptum abietinum	40452		genome.jgi.doe.gov	15.01.2014
16380	Trichinella spiralis	6334	+*	ensemblgenomes.org	03.12.2013.
12586	Trichoderma asperellum CBS 433.97	1042311		genome.jgi.doe.gov	15.01.2014
14095	Trichoderma harzianum CBS 226.95	983964		genome.jgi.doe.gov	15.01.2014
10938	Trichoderma longibrachiatum ATCC 18648	983965	+	genome.jgi.doe.gov	15.01.2014
9113	Trichoderma reesei QM6a	431241	*	ensemblgenomes.org	03.12.2013.
9852	Trichoderma reesei RUT C-30	1344414		genome.jgi.doe.gov	15.01.2014
12400	Trichoderma virens Gv29-8	413071	*	ensemblgenomes.org	03.12.2013.
22885	Tricholoma matsutake 945	1095628		genome.jgi.doe.gov	15.01.2014
8907	Trichomonas vaginalis	5722	+	TrichDB.org	03.04.2014
8676	Trichophyton equinum CBS 127.97	559882		broadinstitute.org	19.12.2013
8707	Trichophyton rubrum CBS 118892	559305		genome.jgi.doe.gov	15.01.2014
8521	Trichophyton tonsurans CBS 112818	647933		broadinstitute.org	19.12.2013
8024	Trichophyton verrucosum HKI 0517	663202		genome.jgi.doe.gov	15.01.2014
11520	Trichoplax adhaerens	10228	+*	ensemblgenomes.org	03.12.2013.
14261	Trichuris suis	68888		na	03.04.2014
14565	Triticum aestivum	4565	+	ensemblgenomes.org	11.12.2013.
33424	Triticum urartu	4572	*	ensemblgenomes.org	11.12.2013.
8747	Trypanosoma brucei	5691	+*	ensemblgenomes.org	03.12.1013.
10228	Trypanosoma cruzi marinkellei	85056	+	TriTrypDB.org	03.04.2014
11858	Trypethelium eluteriae	364715		genome.jgi.doe.gov	15.01.2014

7496	Tuber melanosporum Mel28	656061	*	ensemblgenomes.org	03.12.2013.
19659	Tulasnella calospora MUT 4182	1051891		genome.jgi.doe.gov	15.01.2014
15471	Tupaia belangeri	37347	+*	ensembl.org	02.12.2013
9739	Tursiops truncatus	9739	*	ensembl.org	02.12.2013
26052	Tyto alba	56313	+	na	03.04.2014
7798	Uncinocarpus reesii 1704	336963	+	broadinstitute.org	03.04.2014
6522	Ustilago maydis 521	237631	+*	ensemblgenomes.org	03.12.2013.
11033	Vanderwaltozyma polyspora DSM 70294	436907	+	na	03.04.2014
10221	Verticillium alfalfae VaMs.102	526221		genome.jgi.doe.gov	15.01.2014
10535	Verticillium dahliae VdLs.17	498257		broadinstitute.org	09.01.2014
11765	Vicugna pacos	30538	+*	ensembl.org	02.12.2013
29927	Vitis vinifera	29760	+*	ensemblgenomes.org	11.12.2013.
11084	Volvariella volvacea V23	706473	+	genome.jgi.doe.gov	15.01.2014
15285	Volvox carteri	3067	+	ftp.jgi-psf.org	15.01.2014
4863	Wallemia ichthyophaga EXF-994	1299270		genome.jgi.doe.gov	15.01.2014
5284	Wallemia sebi	148960	+	genome.jgi.doe.gov	15.01.2014
24323	Wasmannia auropunctata	64793	+	na	03.04.2014
6423	Wickerhamomyces anomalus NRRL Y-366-8	683960	+	genome.jgi.doe.gov	15.01.2014
13093	Wilcoxina mikolae CBS 423.85	1314677		genome.jgi.doe.gov	15.01.2014
12746	Wolfiporia cocos MD-104 SS10	742152		genome.jgi.doe.gov	15.01.2014
19187	Wuchereria bancrofti	6293	+	broadinstitute.org	19.12.2013
10818	Xanthoria parietina 46-1-SA22	714311		genome.jgi.doe.gov	15.01.2014
43025	Xenopus laevis	8355	+	Xenbase	03.04.2014
18442	Xenopus <Silurana> tropicalis	8364	+*	ensembl.org	02.12.2013
20379	Xiphophorus maculatus	8083	+*	ensembl.org	02.12.2013
8205	Xylona heveae TC161	1328760		genome.jgi.doe.gov	15.01.2014
6448	Yarrowia lipolytica CLIB122	284591	*	ensemblgenomes.org	03.12.2013.
16015	Zasmidium cellare ATCC 36951	1080233		genome.jgi.doe.gov	15.01.2014
63480	Zea mays	4577	+*	ensemblgenomes.org	11.12.2013.
19226	Zonotrichia albicollis	44394	+	na	03.04.2014
21730	Zopfia rhizophila	160035		genome.jgi.doe.gov	15.01.2014
9925	Zygosaccharomyces bailii ISA1307	1355161	+	na	03.04.2014
5128	Zygosaccharomyces rouxii CBS 732	559307		na	03.04.2014
10931	Zymoseptoria tritici IPO323	336722	*	ensemblgenomes.org	03.12.2013.

Appendix B

Software documentation

B.1 Introduction

PhyloToolKit is a collection of C++ libraries and applications containing the above described algorithmic solutions implementing fast and efficient gene gain and loss computation strategies. Currently only the Beta version of the toolkit is available, implying some features regarding (usually) sporadic utilities are not fully functional, thus limited to a set of predefined parameters (clearly stated before or at runtime).

Primary emphasis regarding the implementation was on efficiency and flexibility (extensibility) of the underlying libraries. Therefore, in some cases classic C type solutions are implemented instead of their C++ alternatives (particularly in cases involving memory management).

It should be emphasized that the entire library is based on a ncbi toolkit and in principle can be updated by any future release simply by replacing the old source files. Latest version of PhyloToolKit is available form:

https://github.com/RobertBakaric/PhyloToolKit

In the following sections I provide examples on how to compile and use the tools found in the toolkit.

PhyloToolKit Documentation

Robert Bakarić
rbakaric@irb.hr
bakaric@evolbio.mpg.de
15.04.2015
PhyloToolKit-1.0

Abstract

This package contains a collection of C++ tools along with supporting Perl programs designed for fast and efficient computational analysis associated to gain and loss of genes and gene families. For more information about the underlying models and implemented algorithms see Chapter 2.

Availability: `https://github.com/RobertBakaric/`

Contents

1 Installation (Unix-Only)

The simplest way to compile and install the package is to:

1. Unpack the PhyloToolKit (`phylotoolkit-XXX.tar.gz`):

   ```
   tar -xvzf phylotoolkit-XXX.tar.gz
   ```

2. Change the current directory to `phylotoolkit-XXX`:

   ```
   cd phylotoolkit-XXX/
   ```

3. Configure the program for your system and install it (-L and -I are optional and should be specified if your boost library is not located on a default location):

   ```
   ./QuickInstallUnix.sh -L ./path/to/boost/lib/dir \
                         -I ./path/to/boost/include/dir
   ```

Once installed, your binaries should be in `./bin` directory. In case the installation needs to be carried out on a system with a particular (non-standard) architectural design, please modify the `QuickInstallUnix.sh` accordingly.

2 Synopsis

In this section, high level description of functions and classes used as part of the PhyloToolKit are located. For more information regarding their implementation and utilities, as well as those not a part of the class interface, please refer to the appropriate source file located in `./src` directory.

2.1 Tree

The class is designed to convert two integer vectors, with values arranged in a parent child relation, into an appropriate data structure capable of coping with desired queries and manipulations regarding node arraignments and their connections.

Functions:

`MakeTree` : Function converts two given integer vectors into the appropriate data structure used by other functions inside and outside the Tree class.

`make` : Explicit constructor.

`destroy` : Explicit destructor.

`DmpTree` : Function reverts the data structure created by the `MakeTree` and returns the stored information in a form of two integer vectors.

`GetPathVec` : For a given queried node, function returns a set of nodes on a path leading to the root.

Minimal usage example:

```
#include<vector>
#include<Tree.hpp>

    vector<int|long|unsigned|double> parent {0,1,1,2,1,2,3,4,6,3,2,67,67};
    vector<int|long|unsigned|double> child  {1,2,4,8,5,67,6,14,15,68,3,11,17};

    Tree<vector<Tint>, vector<Tint> > MyTree(parent,child);

    vector<int|long|unsigned|double> p-dash;
    vector<int|long|unsigned|double> c-dash;
    vector<int|long|unsigned|double> path;

    MyTree.DmpTree(p-dash,c-dash);      // creates two vectors

    path = MyTree.GetPathVec()      // returns the depth vector
```

2.2 TEdit

TEdit class relies on a Tree class inheriting all its features. Its utilities include manipulations regarding node connectivity and relations within data structure created by the Tree class.

Functions:

`make` :	Explicit constructor.
`destroy` :	Explicit destructor.
`SimpleTraverse` :	The function executes a simple recursive depth first tree traversal.
`DepthFirstTrav` :	Given a starting node, function conducts a depth first search traversal. The result is a set of nodes as they are visited during the traversal and their distance (depth) values. The two can be retrieved by invoking `GetDepthVec` and `GetVerticesVec` functions.
`GetDepthVec` :	The function returns the computed vector of node depth values (distance from root of the tree) as they are visited in the traversal.
`GetVerticesVec` :	The function returns the computed vector of nodes as they are visited during the tree traversal.
`GetPathToRoot` :	Given a node, the function returns a vector of nodes on a path from root to a given query node.
`GetBranchingLeafs` :	Given a node on a path from root to leaf (not including the two), the function returns a vector of leaf nodes within a clade starting at a that node.
`GetBranchingNodes` :	Given a node on a path from root to leaf (not including the two), the function returns a vector of nodes within a clade starting at a given node.
`ComputeBranchingClades` :	Given a phylogeny (a vector of nodes on a path from root to leaf) the function executes a simple tree traversal locating all leafs and

internal nodes within a given clade specified by a node within a given phylogeny.

`DumpTree` : The function invokes `Tree.DumpTree()` function and returns its output.

`AddNode` : Given a node and its parent location, function creates an edge between the two.

`RemoveNode` : Given a node, function removes it from the underlying tree together with all of its successor (child) nodes by relocating it to a new, null-routed location.

`EraseNode` : Given a node function removes the node and reattaches all outgoing edges to its parent node

`RelocateNode` : Given two nodes, function creates an edge between them and removes any existing ones between the relocated node and its previous parent.

`JoinNodes` : Given two nodes function creates an edge between them.

`CreateNode` : Given a tree, the function computes the next free node label (64-bit integer).

Minimal usage example:

```
#include<vector>
#include<TEdit.hpp>

    vector<int|long|unsigned|double> parent {1,1,2,1,2,3,4,6,3,2,67,67};
    vector<int|long|unsigned|double> child  {2,4,8,5,67,6,14,15,68,3,11,17};

/* Make Tree */

    /* Construction */
    TEdit<int|long|unsigned|double> MyTree(parent,child);


/*  Functions */

    MyTree.AddNode(CreateVertex(),8)  // creates a new node and adds it to node 8
                                      //    result: 8 -- 69 edge

    MyTree.EraseNode(4);  // it erases node 4 and makes an edge between 14 and 1

    MyTree.RemoveNode(67);// node 67 and all its child nodes are relocated to graph
                          // rotted at 0
                                      //    result: 0 -- 67 +- 11
                                                        +- 17

    MyTree.RelocateNode(5,6)          // assigns node 5 to node 6
                                             //    result:   1 +- 4 -- 14
                                                              +- 2 +- 8
                                                                   +- 3 +- 68
                                                                        +- 6 +- 15
                                                                             +- 5

    vector<int|long|unsigned|double> p-dash;
    vector<int|long|unsigned|double> c-dash;

    MyTree.DmpTree(p-dash,c-dash);      // creates two vectors
                 //results: p-dash:  0 67 67 1  1 2 2 3  3  4  6 6  8  8
                 //         c-dash: 67 11 17 2 14 8 3 6 68 14 15 5 69 70
```

2.3 ETT

Euler Tour Tree Representation class. Given a tree data structure (inherited from the Tree class), the main utility of this class is to compute the "euler tour" array by traversing it.

Functions:

`make` : Explicit constructor.

`destroy` : Explicit destructor. Destroys the local information and Tree class.

`Traverse` : Given a node, function computes the tour over the tree (graph) created by the Tree class.

`Clean` : Function is used for explicit removal of results obtained by the `Traverse` function.

`GetVertexIdVec` : Function returns the computed nodes as they are visited during the traversal.

`GetDepthVec` : Function returns the computed distance values of nodes from the root as they are visited during the traversal.

Minimal usage example:

```
#include<vector>
#include<ETourTec.hpp>

    vector<int|long|unsigned|double> parent {0,1,1,2,1,2,3,4,6,3,2,67,67};
    vector<int|long|unsigned|double> child  {1,2,4,8,5,67,6,14,15,68,3,11,17};

/* Make Graph */

    /* Construction */
    Graph<int|long|unsigned|double> graph(parent,child);
    /*  OR  */
    Graph<int|long|unsigned|double> graph;
    graph.make(parent,child);

/*  Make ETT  */

    /* Construction */
    EulerTour<int|long|unsigned|double> ett(0,parent,child);
    /*  OR  */
    EulerTour<int|long|unsigned|double> ett(parent,child);
    /*  OR  */
    EulerTour<int|long|unsigned|double> ett;
    et.make(0,parent,child);
      /*  or  */
    ett.make(parent,child);

/*  Functions */

    ett.Traverse(2)        // traverses the tree with 2 as a start and stop point

    ett.GetDepthVec()      // returns the depth vector
                           //    result for 2:      0 1 0 1 2 1 2 1 0 1 2 3 2 1 2 1 0
    ett.GetVertexIdVec()   // returns the set of vertices (vector)
                           //    result for 2:      2 8 2 67 11 67 17 67 2 3 6 15 6 3 68 3 2

    ett.Clean()            // cleans the treversed path (both dept and value)
```

2.4 STable

STable is a sparse table solution underlying search query algorithms. The class provides basis for the constant time MRCA (Most Recent Common Ancestor) queries sometimes referred to as LCA or RMQ queries.

Functions:

`make` : Explicit constructor. Given an array of integers, the function creates a sparse table for constant MRCA queries.

`destroy` : Explicit destructor.

Minimal usage example:

```
#include<vector>
#include<SparstTable.hpp>

    vector<int|long|unsigned|double> vec {1,5,23,7,8,3,12,5,3,44,56};

/* sparse table */
    /* Construct */
    STable<int|long|unsigned|double> sptab(vec);
    /* OR */
    STable<int|long|unsigned|double> sptab;
    sptab.make(vec)

    /* Explicite Destructor */computation
    sptab.destroy();
```

2.5 MRCA

MRCA class facilitates the constant time most recent common ancestor queries. Given a tree and a pair of nodes within it, MRCA calculates their most recent common ancestor, parent node from which the two descendent ones derived from (or their successive child nodes).

Functions:

`make` : Explicit constructor. Given an array of integers, the function creates a local map and calls STable constructor.

`destroy` : Explicit destructor. Destroys the local range map and all subsequent STable calss tables.

`MinVal` : Given two nodes the function returns the minimum value within a given range, that is the depth of the most recent common ancestor.

`MinPos` : Given two nodes the function returns the array position (index value) of a minimum value within a given range, that is the node representing the most recent common ancestor of the two.

Minimal usage example:

```
#include<vector>
#include<MRCA.hpp>

    vector<int|long|unsigned|double> vec {1,5,23,7,8,3,12,5,3,44,56}; // distance values
/* MRCA */
    /* Construct */
    MRCA<int|long|unsigned|double> mrca(vec);
    /* OR */
    MRCA<int|long|unsigned|double> mrca;
    mrca.make(vec)

    /* Explicite Destructor */
    mrca.destroy();


    /* MRCA Query for (5,3) */
    /* return value */
    mrca.MinVal(5-1,3-1); // returns 7
                          // (-1 is for vector index positions since indexing starts from 0)

    /* return position */
    mrca.MinPos(5-1,3-1); // returns 3 since indexing starts from 0
                          // (-1 is for vector index positions since indexing starts from 0)
```

2.6 Newick

Newick class is directly dependant on a Tree class inheriting all its data structures and functions. Its utilities include restructuring the underlying tree (from Tree class) in order to create standard, widely accepted (by other software tools) Newick tree file format.

Functions:

`GetNewick :` The function returns Newick tree format as a string computed during the class construction.

Minimal usage example:

```
#include<vector>
#include<Newick.hpp>

    vector<int|long|unsigned|double> parent {0,1,1,2,1,2,3,4,6,3,2,67,67};
    vector<int|long|unsigned|double> child  {1,2,4,8,5,67,6,14,15,68,3,11,17};
    unordered_map<int|long|unsigned|double,string> names; // 1 => root ...
    int|long|unsigned|double  start = 2;
    int|long|unsigned|double  depth = 3;

/* Newick */
    /* Construct */
    Newick<vector<Tint>, vector<Tint> > MyNewick(start,depth,parent,child,names);

    /* Get string */
    string mytree = MyNewick.GetNewick();
```

2.7 ASCIITree

As in the previous case, ASCIITree class is directly dependant on the Tree class, thus inheriting all its data structures and functions. Its utility like

Newick class includes restructuring the underlying tree (from Tree class) in order to create a modified version of the Newick file format that can be used and parsed by a TreeIllustrator.pl software tool.

Functions:

`PrintTree` : The function returns ASCIITree tree format as a string computed during the class construction. The string can than be passed to `TreeIllustrator.pl` in order to visualise it within the terminal session.

Minimal usage example:

```
#include<vector>
#include<ASCIITree.hpp>

    vector<int|long|unsigned|double> parent {0,1,1,2,1,2,3,4,6,3,2,67,67};
    vector<int|long|unsigned|double> child  {1,2,4,8,5,67,6,14,15,68,3,11,17};
    unordered_map<int|long|unsigned|double,string> names; // 1 => root ...
    int|long|unsigned|double  start = 2;
    int|long|unsigned|double  depth = 3;

/* ASCIITree */
    /* Construct */
    ASCIITree<vector<Tint>, vector<Tint> > MyASCIITree(start,depth,parent,child,names);

    /* Get string */
    string mytree = MyASCIITree.PrintTree();
```

2.8 ScoreMatrix

Class is designed to parse the BLOSUM or PAM matrices passed to a constructor as a string. Once the matrix has been obtained and constructed, constant time matching score values can be obtained by passing a single query residue (or a pair).

Functions:

`Score` : As an input, the function takes one or two amino acid residues (or nucleic, depending on the underlying scoring matrix) and returns either a score of a match or a self-match (if a single residue has been passed).

Minimal usage example:

```
#include<vector>
#include<Matrix.hpp>

    string MTX = "A R N D ... \n  A 4 -1 -2 ..." // BLOSUM62

/* ScoreMatrix */
    /* Construct */
    ScoreMatrix<int|long|unsigned|double> matrix(MTX);
```

```
/* Get score value */
matrix.Score("A");  // returns 4

matrix.Score("S","N");  // returns 1
```

2.9 PGI

PGI class is designed for managing the pgi indexing system used throughout the entire PhyloToolkit package.

Functions:

`ParsePgi` : The function extracts the **pgi**, **ti** and **pi** information from fasta header. The information is then used for indexing the entire record according to its taxonomy identifier (**ti**). Moreover, there are two variations of this function, one uses a single stringas an input, while the other can work with a vector of strings.

`GetTaxId` : Based on a **pgi** identifier the function returns the appropriate taxonomy identifier to which the a given **pgi** identifier is associated to.

`GetPgi` : Given a taxonomy identifier, the function returns a set of **pgi** identifiers associated to it.

Minimal usage example:

```
#include<vector>
#incluse<string>
#include<PhyloFasta.hpp>


    vector <string> pgi {"pgi|0000000000000077391|ti|7739|pi|0|", \\
                         "pgi|0000000000000077393|ti|7739|pi|0|"} ;


/* PGI */
    /* Construct */
    PGI<vector<string>, int|long|unsigned|double> MyPgi(pgi);

    /* Getters */
    MyPgi.GetTaxId(0000000000000077391);  // returns 7739

    MyPgi.GetPgi(7739);  // returns 0000000000000077391, 0000000000000077393
```

2.10 FastaSeq

Class is designed to preform reading and indexing fasta formatted sequence files. The underlying index is created to suit the requirements of functions and classes within PhyloToolKit package.

Functions:

`ReadFasta` : The function design allows reading fasta formatted files in three modes. Mode "**all**" refers to reading the entire fasta record. Mode

"**head**" only stores header information and "**body**" mode the corresponding sequence. Moreover, by specifying the record type as "pgi", header information is additionally indexed according to pgi, ti and pi identifiers.

`LoadHeaders` : The function is a pre-set instance of `ReadFasta` function with mode set to "head".

`LoadSequences` : The function is a pre-set instance of `ReadFasta` function with mode set to "body".

`GetFastaSeq` : The function is a getter returning a single sequence upon request using the record index.

`GetFastaHead` : The function is a getter returning a single header information upon request defined using the record index.

`GetHeaders` : The function returns the entire set of fasta headers of stored records.

`GetSequences` : The function returns the entire set of fasta body information of stored records.

`GetSequenceCount` : The function returns the number of records within a container.

`UpdateHeader` : Given a record index, the function assigns a new header to it.

`UpdateSequence` : Given a record index, the function assigns a new body (sequence) to it.

`AddFastaRecord` : The function adds a new fasta record to the container.

`RemoveFastaRecord` : The function removes a fasta record given the record index information from the container.

`DmpFastaSeq` : Given the output file location, the function prints out the entire set of records contained within it. Moreover, if a set of sequence header identifiers is provided (or record indices) the printing is restricted to only those (fasta records) specified.

Minimal usage example:

```
#include<vector>
#incluse<string>
#include<PhyloFasta.hpp>


/* Make FastaSeq object*/
    /* Construct */
    FastaSeq<string,vector<int>,int> MySeqCont("filename", \
                                                ["pgi"/"norm"], ["body"/"head"/"all"])

/*  Usage examples */
    string seq = MySeqCont.GetFastaSeq(34);            // returns the 34th sequence
    vector<string> headers = MySeqCont.GetHeaders();   // returns all headers
    MySeqCont.AddFastaRecord("header","HIKJAAAAIIII"); // adds record to the end
    MySeqCont.GetSequenceCount();                      // returns the total number of sequer
    MySeqCont.DmpFastaSeq("output.file");              // prints all records into output.fil
```

2.11 PhyloClust

Like many before, PhyloClust also inherits all features from the Tree class. Its functionality includes cluster computation from pairwise connectivity

of objects passed to it.

Functions:

`ComputeClusters` : As an input the function accepts two objects `FastaSeq` and `Tree`. Where `Tree` contains the information regarding pairwise connections between objects to be clustered and thus different from the one passed during the construction meant for creating the underlying phylogeny tree.

Minimal usage example:

```
#include<vector>
#include<Clust.hpp>
#include<PhyloFasta.hpp>

    vector<int|long|unsigned|double> parent {0,1,1,2,1,2,3,4,6,3,2,67,67};
    vector<int|long|unsigned|double> child  {1,2,4,8,5,67,6,14,15,68,3,11,17};

    vector<int|long|unsigned|double> sid {0,1,1,2};
    vector<int|long|unsigned|double> hit  {1,2,4,8};

/* Make FastaSeq object*/
    /* Construct */
    FastaSeq<string,vector<int>,int> MySeqCont("filename", \
                                              ["pgi"/"norm"], ["body"/"head"/"all"])

/* Make connectivity map */
    /* Construct */
    Tree<int|long|unsigned|double> ConnectivityMap(sid,hit);

/* PhyloClust */
    /* Construct */
    PhyloClust<int|long|unsigned|double> Clusters(parent,child);

    /* Compute clusters */
    vector<int|long|unsigned|double>ClRepresentatives = \
                               Clusters.ComputeClusters(Fasta,ConnectivityMap);
```

2.12 Blast2Seq

Blast2Seq calss is designed to serve as an interface for the ncbi+ library allowing a simple pairwise sequence alignment computation. The class currently supports protein alignments only and implements a single hit homology detection strategy referred to as the QuickBlast solution.

Functions:

`SetBlastOptions` : Function is designed to parse the input options and setup the blast search.

`RunBlast` : Given a pair of sequences, the function executes the blast search.

Minimal usage example:

```
#include<string>
#include<Blast.hpp>

    string query = "AAHHILKSTWYAAAAAAIIIII";
    string subject = "AAHHILKSIIIII";

/* Make Blast2Seq object*/
    /* Construct */
    Blast2Seq<int|long|unsigned|double> blast("options...")

   /* Search */
    blast.RunBlast(query,subject,"qid", "sid");  // returns: TRUE if a match is found
                                                 //          FALSE if not
```

2.13 SEG

SEG class is based in seg AA sequence filter. It encapsulates the filter for masking low compositional complexity regions.

Functions:

`SegFilt` : The function takes a string as an input and preforms low compositional complexity filtering by substituting identified regions with "X" symbols. The masked sequence is then returned back as a string object with X's where previously low compositional complexity regions were located.

Minimal usage example:

```
#include<string>
#include<SEG.hpp>

    string input = "AAHHILKSTWYAAAAAAIIIII";

/* Make SEG object*/
    /* Construct */
    SEG<int|long|unsigned|double> seg(options) // unordered_map<string, int> options

   /* Low compositional complexity masking */
    string seq = SegFilt(input);   // returns:  XXHHILKSTWYXXXXXXXXXXX
```

2.14 HDIndex

Upon construction, given a sequence set, class creates an index based on sequence k-mer type content.

Functions:

`MakeIndex` : Given the FastaSeq object, the function creates HD-index data structure.

Minimal usage example:

```
#include<string>
#include<PhyloFasta.hpp>
#include<HDIndex.hpp>


/* Make FastaSeq object*/
    /* Construct */
    FastaSeq<string,vector<int>,int> MyFastaSeq("filename", \
                                                ["pgi"/"norm"], ["body"/"head"/"all"]);

/* Make HDindex object*/
    /* Construct */
    HDIndex<int|long|unsigned|double> MyIndex(options...);  // k-mer formatting options

    MyIndex.MakeIndex(MyFastaSeq);
```

2.15 HDQuery

Upon creating HDIndex object, search is executed by calling functions in HDQuery class. Therefore, class contains a set to tools for querying the HD-index data structure.

Functions:

`CreateIndex` : The function invokes the HDIndex constructor in order to create HDindex data structure.

`Search` : Given a query sequence, the function executes the search, results of which can then be retrieved through `GetTopXScoreVec` and `GetTopXIdVec` functions.

`GetTopXScoreVec` : Function returns the X number of highest score values from query-subject pairwise comparison to a given query sequence.

`GetTopXIdVec` : Function returns the X number of closest subject sequence identifiers to a given query sequence. Closeness is defined through score values computed by comparing the k-mer type content between query and each subject sequence.

Minimal usage example:

```
#include<string>
#include<vector>
#include<PhyloFasta.hpp>
#include<HDIndex.hpp>


   string QS = "GHTKKIJKWYA";

/* Make FastaSeq object*/
    /* Construct */
    FastaSeq<string,vector<int>,int> MyFastaSeq("filename", \
                                                ["pgi"/"norm"], ["body"/"head"/"all"]);

/* Make HDIndex */
    /* Construct */
    HDQuery<int> hd(X,range,kmer,segsize,sccutoff,matrix);  // different options
    hd.CreateIndex(MyFastaSeq);
```

```
/* Execute the search for a given query */
hd.Search(QS);

/* Get results */
vector<Tint> topX = hd.GetTopXIdVec();
```

Note:

For documentation regarding functions used within the software solutions described below, please refer to the appropriate source file located in `.src/apps/`.

3 General Road Map

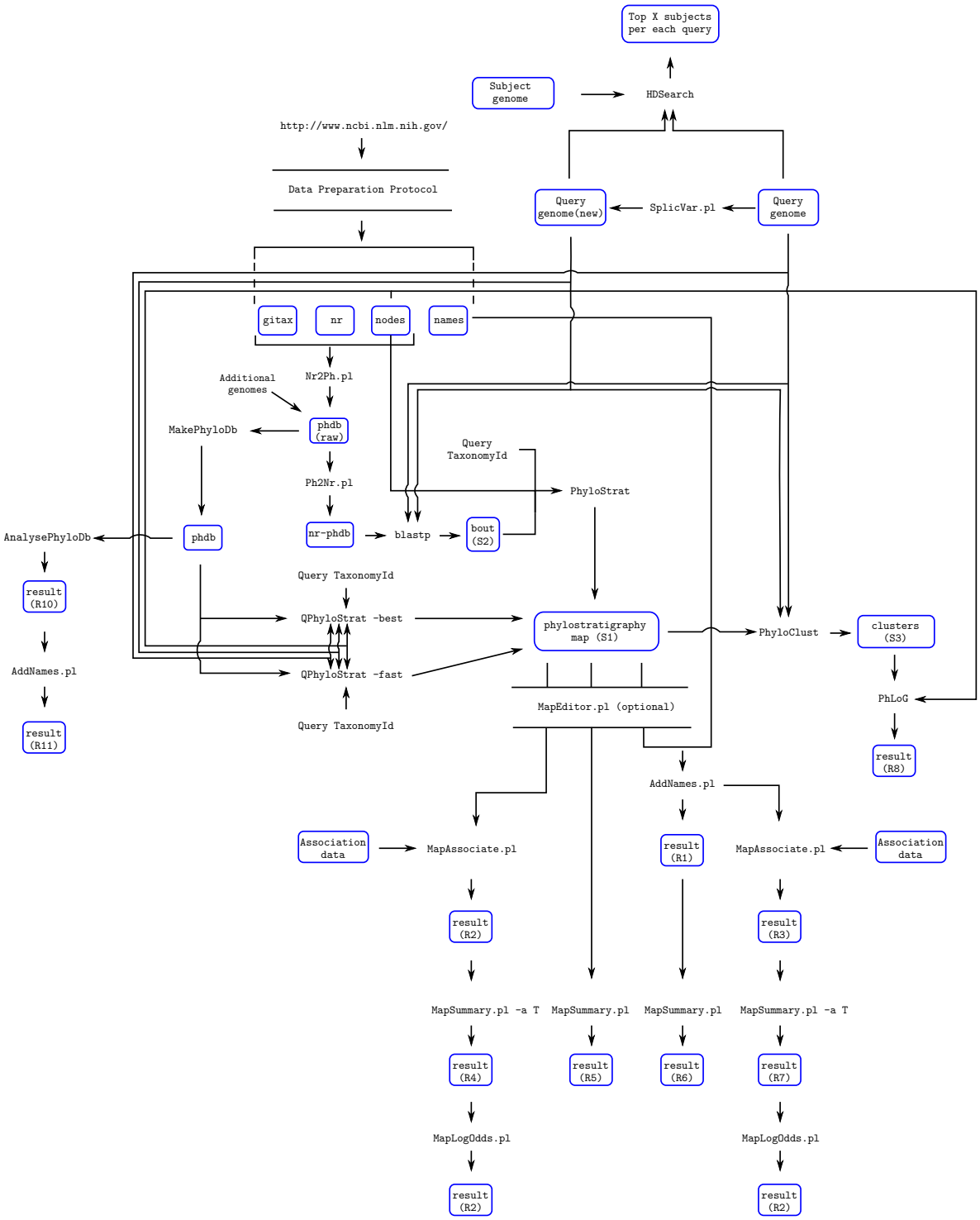


Figure 1: The figure summarizes a set of pipelines and the way they are interconnected in the process of obtaining a desired result.

4 Core Tools (C++)

4.1 MakePhyloDb

MakePhyloDb is a program designed for formatting protein and/or nucleotide databases used as an input for `QPhyloStrat`, `PhyloClust`, etc. tools. It operates on a set of flat-file fasta formatted DNA and Protein sequences, each named according to a unique taxonomy identifier to which each sequence in the corresponding file is associated to. Moreover, the program calculates basic content information related to each file and filters out the low complexity regions using SEG/XNU filtering approach.

4.1.1 Program options

In order to see program options type:

```
./bin/MakePhyloDb -h
```

Expected output:

```
Usage: ./program [options]

********************************************************************************
                                    MakePhyloDb
                                         by
                                   Robert Bakaric

CONTACT:
 Code written and maintained by Robert Bakaric,
 email: rbakaric@irb.hr , bakaric@evolbio.mpg.de

ACKNOWLEDGEMENT:

 SEG:
   Wootton, J. C. and S. Federhen (1993).  Statistics of local complexity in amino
   acid sequences and sequence databases.  Computers and Chemistry 17:149-163.

 XNU:
   Claverie, J.-M. and States, D.J. (1993). Information enhancement methods for
   large-scale sequence analysis. Computers and Chemistry 17:191-201.

LICENSE:
 The program is distributed under the GNU General Public License. You should have
 received a copy of the licence together  with this software. If not, see
 http://www.gnu.org/licenses/
********************************************************************************


Allowed options:
  -h [ --help ]                         produce help message
  -v [ --version ]                      print version information
  -d [ --database ] arg                 Path to database directory (required)
  -f [ --filter ] arg (=SEG)            Sequence filter [SEG/XNU]


  --SEG filter options--

  -w [ --window ] arg (=12)             SEG window size
  -L [ --locut ] arg (=2.2)             Low complexity cutoff (starter)
  -H [ --hicut ] arg (=2.5)             High complexity cutoff (starter)
  -x [ --maxXes ] arg (=0)              Maximum number of xxx  symbols
                                        tolerated (dynamically defined if left
                                        unchanged)
  -t [ --maxTrim ] arg (=100)           Maximum trimming of raw segment


  --XNU filter options--
```

```
  -p [ --pam ] arg (=-60)              PAM  matrix to use [-60] [-120] [-250]
  -P [ --probability ] arg (=0.01)     Probability cutoff
  -s [ --score ] arg (=5)              Score cutoff
  -m [ --max-search-offset ] arg (=4)  Maximum search offset
  -n [ --min-search-offset ] arg (=1)  Minimum search offset
```

4.1.2 Input Data

MakePhyloDb operates on a flat-file database, a collection of fasta formatted sequence files all having the same directory address. Therefore, input information for the program consists of a single directory path to a location where flat-fasta-files are located. The only requirement is that each file is named according to species taxonomy identifier the sequences in the file are members of. For example:

`./examples/data/phdb`:

```
100027   1182542  203908  339724  426428  5297    63146   717989  86635
100044   1182543  206324  341663  428574  5325    6326    718229  8665
100047   1182544  208960  344612  431241  5331    6334    71945   869754
1001833  1182545  214684  34506   43151   5346    63577   7209    870435
1001937  121224   214687  34740   431595  535722  638633  72120   875184
1001938  1220924  215467  347515  43179   54126   6412    7217    876142
10020    1220926  218851  34765   431895  544711  644223  721885  876976
10090    1230383  222929  35128   44056   544712  644352  7220    88036
10116    123356   225164  353151  441959  5457    644358  7222    881290
...
```

Each file having the following structure:

`./examples/data/phdb/1182544`:

```
>A1O7_00001T0 | A1O7_00001 | Cladophialophora yegresii CBS 114405 h...
MDPIDVEGNVNIAGKEEVVATHTERIVDDEKKHQEPTLTRAEIRRFMWKVDAHVLPMLGL
IYAISILDRINIGSAKVLGMQEDLNLGTQRYSIVLMIYFPGYALSDVPSNWILTKVEPRW
WLPFLTVAWGAVLTGMGFVHNWGILAFLRLLLGTLEGGILPGITFTIACWYSRHELHKRI
SFAYGIGVVASGLAGILSFGLGSMSGLRDMNGWRWIFSIEGGATMAVGCIAPFFVPKFPD
HTKWIKPDERVYLYNKLEKDRGDYKTGKVGWSSFVHTAKDWTLWAQGTIYCFNVGTANAV
GFFTPTIIKGLGYSGLQASLRSGYPFFAALGLLGITSYLSDKYQKRAIICIFNSFVMITG
FSIMREGFSNHVRYFGIFLATMGVHSNTPALLAFNQSNIVDSAGRAVSSGILIACGAIGG
IIGSLIFRGQDAPSYGPGIYTTIGLTAYMVLALSFMVYIYHSRNKKADRDGTHIAGVPGF
RYAL*
>A1O7_00002T0 | A1O7_00002 | Cladophialophora yegresii CBS 114405 h...
MAPSRLIDEDKLGFRSDSHSEDLSDDGAFHHPPPKRRRISTEEENATVPQTSLSRVKKID
HAPKTGSTPAAHPEIPTTSATFKSLNVALWLVHSLAAMAIQNPTEIQKACIPEILKGRDC
IGGSRTGTGKTVAFAVPILQKWAEDPFGIYAVVLTPTRELALQIFEQFQALGAPQNLKTV
LITGGSDMRPQALALAKRPHIVVATPGRLADHVLNSGKETTVGLSRTKVVVLDEADRLLA
PGQGSMLPDLNTCLGALPPSTYRLTLLFTATVTPEVRALKELPRPKERPPIFISETTDLS
DGAPQSSLIPATLSQTYLQIPMTHKDAFLHVLLQVPSLTKSPEPSIMIFVNRTNTADLLH
RTLLQLGHPVTALHSELAQSQRNRNLSDFRSQKARILIATDVASRGLDIPQVNLVINFDV
PRNPTDYVHRVGRTARAGRQGTSITLVGQRDVELILAIEAYVGSKLVKWTEEGVNVETRV
LKGRTLKDVAEARMEALRDVEGGKDVHGWRRKLKKDKKRSVAEASAS*
>A1O7_00003T0 | A1O7_00003 | Cladophialophora yegresii CBS 114405 h...
MSYQSFLRSTATTKNNTPSPGSIHRKPPPASAPSITPSGSSIHSSVTRNDTPTSAVTVPF
TPASSTSSPARPAASTPLPSTVSGTAHHDPARAPLTKEQIDVAVGTCLELQKTATSLHDK
RPFAALLLGPDNNTILLSHYSISHVQHAETELARLATIHFSQKYLASCTLVSTWEPCAMC
AGTIYWSNIGRLVYAASEEKLKDLTGGNNEENMTMSLPCREVLKHGQKDVEVIGPVSDWE
ESVVEESGKWWKEHQAQENAARLREGSVNGTDKPQSLSSMRHGTPTTWTGEETVLSRIDD
EGEYKAELDIDWMR*
>A1O7_00004T0 | A1O7_00004 | Cladophialophora yegresii CBS 114405 h...
MSRTKNKKPGLRAQLKRKRELEGSGDVDGKSAKRLRHSPQPAEQPASSPKSEHQSAQDGK
TAKQPSKLTSAKAAKTPRELPSAQDAVADPALLADRFAKYIQKYSPNSSPIELEEQYLPT
KAFLDTTVYDRDRVAANLPQFIERFSPEGKVGLSNCDDKASPHTLVVTSSGIRTADLYRE
LRVFQNEESKVGKLIAKHMKLRDNIEYMLGNKIGIAISTPFRFKQLVDADALKTGKLRRI
VVDGSFRDEKNNTIFTMPQTFNPLVVLLNEKTIRQRYGEGKGNIDILVF*
...
```

It is important to note that:

1. The header information in each file does not need to be formatted in any way, `MakePhyloDb` takes care of the formatting.
2. For file labels adequate taxonomy identifiers need to be used. Labelling files according to an identifier not present in the adjacency list (nodes file) will eliminate the file from any downstream computational analysis.

4.1.3 Usage Example

An example script can be found in `./examples/MakePhyloDb.sh`.

Running the script:

```
sh MakePhyloDb.sh
```

the following command is executed:

```
../bin/MakePhyloDb -d ../data/phdb
```

Upon completion the expected result is:

```
1000000.ff  1286976.ff  296543.ff  41413.ff   5518.ff    669874.ff  8083.ff
100027.ff   1287680.ff  296587.ff  4155.ff    554065.ff  670386.ff  8090.ff
100044.ff   1287681.ff  29730.ff   418459.ff  554155.ff  670580.ff  8128.ff
100047.ff   12957.ff    29760.ff   420593.ff  556484.ff  671987.ff  81824.ff
1001833.ff  1299270.ff  29883.ff   42068.ff   559292.ff  675120.ff  81972.ff
1001937.ff  1302862.ff  29889.ff   42254.ff   559297.ff  675824.ff  81985.ff
1001938.ff  13037.ff    30538.ff   423536.ff  559298.ff  67593.ff   82310.ff
10020.ff    1305764.ff  3055.ff    425011.ff  559305.ff  683840.ff  8296.ff
10090.ff    1314663.ff  30608.ff   426418.ff  559515.ff  683960.ff  83344.ff
10116.ff    1314666.ff  30611.ff   426428.ff  559561.ff  684364.ff  83485.ff
10141.ff    1314677.ff  3067.ff    428574.ff  559882.ff  686832.ff  8355.ff
...

info.caf
info.paf
```

Where the formatted fasta files contain an additional piece of information in their headers:

`./examples/data/phdb/1182544.ff`:

```
>pgi|0000000000011825441|ti|1182544|pi|0| A1O7_00001T0
MDPIDVEGNVNIAGKEEVVATHTERIVDDEKKHQEPTLTRAEIRRFMWKVDAHVLPMLGLIYAISILDRINIGSAKVLGM
QEDLNLGTQRYSIVLMIYFPGYALSDVPSNWILTKVEPRWWLPFLTVAWGAVLTGMGFVHNWGILAFLRLLLGTLEGGIL
PGITFTIACWYSRHELHKRISFAYGIGVVASGLAGILSFGLGSMSGLRDMNGWRWIFSIEGGATMAVGCIAPFFVPKFPD
HTKWIKPDERVYLYNKLEKDRGDYKTGKVGWSSFVHTAKDWTLWAQGTIYCFNVGTANAVGFFTPTIIKGLGYSGLQASL
RSGYPFFAALGLLGITSYLSDKYQKRAIICIFNSFVMITGFSIMREGFSNHVRYFGIFLATMGVHSNTPALLAFNQSNIV
DSAGRAVSSGILIACGAIGGIIGSLIFRGQDAPSYGPGIYTTIGLTAYMVLALSFMVYIYHSRNKKADRDGTHIAGVPGF
RYAL*
>pgi|0000000000011825442|ti|1182544|pi|0| A1O7_00002T0
MAPSRLIDEDKLGFRSDSHSEDLSDDGAFHHPPPKRRRISTEEENATVPQTSLSRVKKIDHAPKTGSTPAAHPEIPTTSA
TFKSLNVALWLVHSLAAMAIQNPTEIQKACIPEILKGRDCIGGSRTGTGKTVAFAVPILQKWAEDPFGIYAVVLTPTREL
ALQIFEQFQALGAPQNLKTVLITGGSDMRPQALALAKRPHIVVATPGRLADHVLNSGKETTVGLSRTKVVVLDEADRLLA
PGQGSMLPDLNTCLGALPPSTYRLTLLFTATVTPEVRALKELPRPKERPPIFISETTDLSDGAPQSSLIPATLSQTYLQI
PMTHKDAFLHVLLQVPSLTKSPEPSIMIFVNRTNTADLLHRTLLQLGHPVTALHSELAQSQRNRNLSDFRSQKARILIAT
DVASRGLDIPQVNLVINFDVPRNPTDYVHRVGRTARAGRQGTSITLVGQRDVELILAIEAYVGSKLVKWTEEGVNVETRV
LKGRTLKDVAEARMEALRDVEGGKDVHGWRRKLKKDKKRSVAEASAS*
>pgi|0000000000011825443|ti|1182544|pi|0| A1O7_00003T0
MSYQSFLRSTATTKNNTPSPGSIHRKPPPASAPSITPSGSSIHSSVTRNDTPTSAVTVPFTPASSTSSPARPAASTPLPS
TVSGTAHHDPARAPLTKEQIDVAVGTCLELQKTATSLHDKRPFAALLLGPDNNTILLSHYSISHVQHAETELARLATIHF
SQKYLASCTLVSTWEPCAMCAGTIYWSNIGRLVYAASEEKLKDLTGGNNEENMTMSLPCREVLKHGQKDVEVIGPVSDWE
```

```
ESVVEESGKWWKEHQAQENAARLREGSVNGTDKPQSLSSMRHGTPTTWTGEETVLSRIDDEGEYKAELDIDWMR*
>pgi|0000000000011825444|ti|1182544|pi|0| A1O7_00004T0
MSRTKNKKPGLRAQLKRKRELEGSGDVDGKSAKRLRHSPQPAEQPASSPKSEHQSAQDGKTAKQPSKLTSAKAAKTPREL
PSAQDAVADPALLADRFAKYIQKYSPNSSPIELEEQYLPTKAFLDTTVYDRDRVAANLPQFIERFSPEGKVGLSNCDDKA
SPHTLVVTSSGIRTADLYRELRVFQNEESKVGKLIAKHMKLRDNIEYMLGNKIGIAISTPFRFKQLVDADALKTGKLRRI
VVDGSFRDEKNNTIFTMPQTFNPLVVLLNEKTIRQRYGEGKGNIDILVF*
>pgi|0000000000011825445|ti|1182544|pi|0| A1O7_00005T0
MPASEKEGTNEDDRRGSSAIVEDDxxxxxxxxxxxxxSVKINVEGAFIVDDEMNAKNGTANEHVHWEHKDIRLPHHTDVV
SHVAVDIGGSLAKLVYFSRETGSMHGGGRLNFLNFETDRIDLCIDFIQELKKTQLKLNGSTPQELCVMATGGGAYKFYNR
MKEVLHVDVVQEDEMECLIIVGLDFFITEIPREVFTYSEEEPMQFADTRADIYPYLLVNIGSGVSMVKVSGPREFQRVGG
TSLGGGTFWGILSLLTGARTFDEMLRLAEKGDNAGVDMLVGDIYGGGYSKIGLKSTTIASTFGKVFRMKRLAERHAEDGE
GLFNGDDLSDDHEMHGHFKIEDMARSLLYAISNNIGQIAYLQSEKHNLRHIYFGGSFIRGHTQTMNTLSYAIKFWSKGEK
QAYFLRHEGYLGAVGAFLKRQPKNWGRRNSFDDIRLTKVLSKE*
...
```

The added identifier consisting of :

- pgi - phylogeny gene identifier
- ti - taxonomy identifier
- pi - phylostratigraphy identifier

Moreover, each formatted file has an extension `*.ff`. After formatting, two additional info files are created. `info.caf` contains the information about the total and effective size of each fasta file whereas the `info.paf` is a utility file with general summary information further used in downstream computational analysis.

`info` file examples:

`./examples/data/phdb/info.caf`:

```
#taxid  total   effective
1314663 5300    5124
88036   3637    3505
3694    3383    3347
9813    7037    5386
9606    1870    1868
3702    2649    2571
559305  4272    4178
554155  3373    3281
114155  2484    2323
41413   5931    5913
690234  3119    3044
1257118 5350    4881
380704  5006    4936
...
```

`./examples/data/phdb/info.paf`:

```
2015-4-15.12:41:56 :Database Created On:
692 :Number Of Genomes:
3073837 :Database Size:
2948150 :Effective Database Size:
```

Warning !! : DO NOT MODIFY THE CONTENT OF `info.*` FILES

4.1.4 Known problems

1. Destructor's not properly implemented, however, memory leaks are not possible.

4.1.5 Future work

1. Resolve bugs.
2. Implement XNU filter

4.2 AnalysePhyloDb

Given a query point (query species taxonomy identifier), `AnalysePhyloDb` program calculates the number of genomes and their corresponding sizes (the number of genes/sequences) for each branching clade defined according to the ancestral node identifier on a path from root to query species leaf. As an input the program requires the database to be pre-formatted using the `MakePhyloDb` software.

4.2.1 Program options

In order to see program options type:

```
./bin/AnalysePhyloDb -h
```

Expected output:

```
Usage: ./program [options]

********************************************************************************
                                  AnalysePhyloDb
                                        by
                                  Robert Bakaric

CONTACT:
 Code written and maintained by Robert Bakaric,
 email: rbakaric@irb.hr , bakaric@evolbio.mpg.de

ACKNOWLEDGEMENT:

LICENSE:
 The program is distributed under the GNU General Public License. You should have
 received a copy of the licence together  with this software. If not, see
 http://www.gnu.org/licenses/
********************************************************************************


Options:
  -h [ --help ]         produce help message
  -v [ --version ]      print version information
  -n [ --nodes ] arg    Nodes file (required)
  -d [ --database ] arg Location of the database (required)
  -t [ --taxid ] arg    Query species taxomony identifier (required)
```

4.2.2 Input Data

AnalysePhyloDb operates on a flat file database, previously processed using `MakePhyloDb` software. For an example see Section 4.1

4.2.3 Usage Example

An example script can be found in `./examples/MakePhyloDb.sh`.

Running the script:

```
sh AnalysePhyloDb.sh
```

the following command is executed:

```
../bin/AnalysePhyloDb -d ./data/nr-Test/TestDb-phdb/ -t 10090 \
                      -n ./data/nodes
```

Upon completion the expected result should look like this:

```
<ps>        1        0        131567
<ps>        2        1        2759
2         2759       117        3702
<ps>        3        1        1452644
3         1452644    60         44689
<ps>        4        0        1452646
<ps>        5        2        33154
5         33154      39         147573
5         33154      15         214684
<ps>        6        0        1452651
<ps>        7        1        1452652
7         1452652    45         595528
<ps>        8        1        1452653
8         1452653    42         431895
<ps>        9        1        33208
9         33208      80         27923
<ps>        10       1        6072
10         6072      250        10228
<ps>        11       1        33213
11         33213      79         6183
<ps>        12       1        33511
12         33511      227        10224
<ps>        13       1        7711
13         7711       346        7739
<ps>        14       0        1452661
<ps>        15       1        7742
15         7742       311        7757
<ps>        16       1        7776
16         7776       285        7868
<ps>        17       0        117570
<ps>        18       1        117571
18         117571     349        7918
<ps>        19       1        8287
19         8287       424        7897
<ps>        20       0        1338369
<ps>        21       1        32523
21         32523      438        8364
<ps>        22       0        32524
<ps>        23       1        40674
23         40674      296        9258
<ps>        24       1        32525
24         32525      561        9315
<ps>        25       1        9347
25         9347       400        9371
<ps>        26       1        1437010
26         1437010    770        9615
<ps>        27       1        314146
27         314146     453        9544
<ps>        28       0        314147
<ps>        29       1        9989
29         9989       587        10141
<ps>        30       0        33553
<ps>        31       0        337687
<ps>        32       0        10066
<ps>        33       0        39107
<ps>        34       0        10088
<ps>        35       0        862507
<ps>        36       1        10090
36         10090      724        10090
```

Output design (Fig. 2) is is structured such that a simple grep function can easily be used to further parse the data. For example, if only the number of genomes per branching clade is required, executing:

```
grep "<ps>" AnalysePhyloDb.out
```

will produce simple tabular output like:

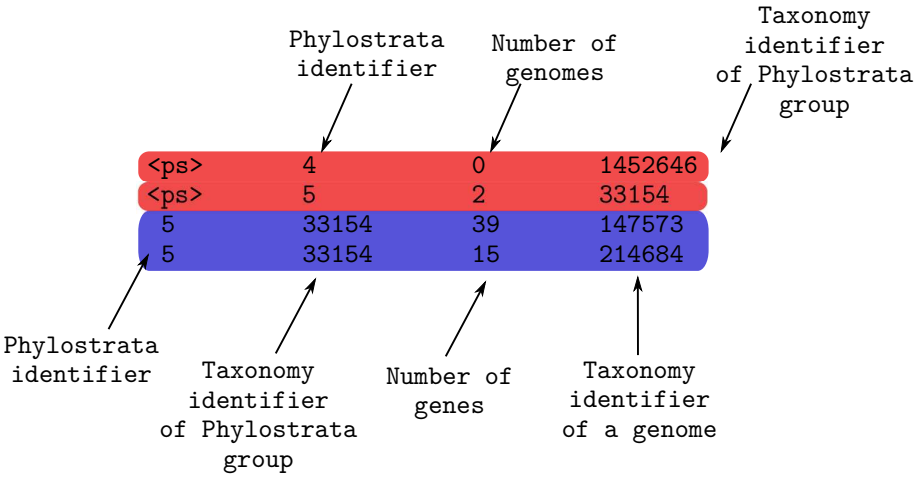


Figure 2: AnalysePhyloDb output description.

```
<ps>    1    0    131567
<ps>    2    1    2759
<ps>    3    1    1452644
<ps>    4    0    1452646
<ps>    5    2    33154
<ps>    6    0    1452651
<ps>    7    1    1452652
<ps>    8    1    1452653
<ps>    9    1    33208
<ps>    10   1    6072
<ps>    11   1    33213
<ps>    12   1    33511
<ps>    13   1    7711
<ps>    14   0    1452661
<ps>    15   1    7742
<ps>    16   1    7776
<ps>    17   0    117570
<ps>    18   1    117571
<ps>    19   1    8287
<ps>    20   0    1338369
<ps>    21   1    32523
<ps>    22   0    32524
<ps>    23   1    40674
<ps>    24   1    32525
<ps>    25   1    9347
<ps>    26   1    1437010
<ps>    27   1    314146
<ps>    28   0    314147
<ps>    29   1    9989
<ps>    30   0    33553
<ps>    31   0    337687
<ps>    32   0    10066
<ps>    33   0    39107
<ps>    34   0    10088
<ps>    35   0    862507
<ps>    36   1    10090
```

Thus listing all branching clades with the number of available genomes per each. From left to right, columns are labelled as :

1. "ps" tag
2. Phylostrata identifier
3. Number of genomes in a branching clade

4. Phylostrata taxonomy identifier

On the other hand, if a list of genes per each genome in each branching clade is required, then by executing:

```
grep -P "^\d+\t" AnalysePhyloDb.out
```

the user can extract genome specific information:

```
2       2759        117     3702
3       1452644     60      44689
5       33154       39      147573
5       33154       15      214684
7       1452652     45      595528
8       1452653     42      431895
9       33208       80      27923
10      6072        250     10228
11      33213       79      6183
12      33511       227     10224
13      7711        346     7739
15      7742        311     7757
16      7776        285     7868
18      117571      349     7918
19      8287        424     7897
21      32523       438     8364
23      40674       296     9258
24      32525       561     9315
25      9347        400     9371
26      1437010     770     9615
27      314146      453     9544
29      9989        587     10141
36      10090       724     10090
```

where now columns from left to right are defined as:

1. Phylostrata identifier
2. Phylostrata taxonomy identifier
3. Gene count
4. Branching clade genome taxonomy identifier

4.2.4 Known problems

Non reported

4.2.5 Future work

Upon request.

4.3 PhyloStrat

PhyloStrat is a fast and an efficient implementation of genome stratification method proposed by Domazet-Loso et al. (2007). Based on a blast output result, applying the Dollo's parsimony approach as an explanation to the observed presence/absence pattern of genes in current species, the program calculates the origin point for each query species gene used and reported in the analysis.

4.3.1 Program options

In order to see program options type:

```
./bin/PhyloStrat -h
```

Expected output:

```
Usage: ./program [options]

*******************************************************************************
                                    PhyloStrat
                                        by
                                   Robert Bakaric

CONTACT:
 Code written and maintained by Robert Bakaric,
 email: rbakaric@irb.hr , bakaric@evolbio.mpg.de

ACKNOWLEDGEMENT:


LICENSE:
 The program is distributed under the GNU General Public License. You should have
 received a copy of the licence together  with this software. If not, see
 http://www.gnu.org/licenses/
*******************************************************************************


Options:
  -h [ --help ]                         produce help message
  -v [ --version ]                      print version information
  -b [ --blast-file ] arg               Blast output file [tabular format]
  -n [ --nodes-file ] arg               Nodes file [tabular - two columns <child
                                          patent>]
  -t [ --taxid ] arg                    Taxonomy identifier of query species
  -e [ --evalue ] arg (=0.001) E-value cutoff treshold
```

4.3.2 Input Files

PhyloStrat requires two input files and a taxonomy identifier of a corresponding query species. One of the files is a species tree, adjacency list of node identifiers (taxonomy ids) divided in two columns, with left column being a child node and right one its parent, example of which can be found in:

`./examples/data/nodes`

```
2       131567
6       335928
7       6
9       32199
10      135621
11      1707
13      203488
14      13
16      32011
17      16
...
```

The other file is *tab* formatted blast output table:

`./examples/data/MouseSample.bout`:

```
MouseQuerySeq_38704436 pgi|38704436|ti|10090|pi|0| 100.00 359 0 0 1 359 1 359 0.0  622
MouseQuerySeq_38704436 pgi|38406037|ti|10020|pi|0| 92.44 357 27 0 3 359 2 358 0.0  581
MouseQuerySeq_38704436 pgi|39300935|ti|10141|pi|0| 89.08 357 36 1 3 359 2 355 0.0  545
MouseQuerySeq_38704436 pgi|29400391|ti|8839|pi|0| 78.61 360 66 2 3 359 2 353 2e-179  504
MouseQuerySeq_38704436 pgi|28808954|ti|8364|pi|0| 66.11 357 100 5 3 359 3 338 2e-134  389
MouseQuerySeq_38704436 pgi|26408981|ti|7897|pi|0| 63.97 358 116 4 3 359 2 347 5e-125  365
MouseQuerySeq_38704436 pgi|30302237|ti|9258|pi|0| 86.60 194 26 0 3 196 1 194 4e-113  331
MouseQuerySeq_38704436 pgi|16803081|ti|6183|pi|0| 42.53 174 74 2 37 184 7 180 1e-38  140
...
```

It is important to note that the second column in blast output is formatted as in the example, containing the information about :

- pgi - phylogeny gene identifier
- ti - taxonomy identifier
- pi - phylostratigraphy identifier

Query species gene identifier can be arbitrarily defined at this point.

4.3.3 Usage Example

A complete example script can be found in `./examples/PhyloStrat.sh`.

Running the script:

```
sh PhyloStrat.sh
```

the following command is executed:

```
../bin/PhyloStrat -b ./data/boutmin -n ./data/nodes -t 10090
```

Upon completion the expected result should look like this:

```
<Species taxomony Id>
10090
<Phylogeny used>
0 1
1 131567
2 2759
3 1452644
4 1452646
5 33154
6 1452651
7 1452652
8 1452653
9 33208
10 6072
11 33213
12 33511
13 7711
14 1452661
15 7742
16 7776
17 117570
18 117571
19 8287
20 1338369
21 32523
22 32524
23 40674
24 32525
25 9347
26 1437010
27 314146
```

```
28 314147
29 9989
30 33553
31 337687
32 10066
33 39107
34 10088
35 862507
36 10090
<Phylostratigraphy map>
MouseQuerySeq_38709112 2 2759
MouseQuerySeq_38713615 9 33208
MouseQuerySeq_38718975 1 131567
MouseQuerySeq_38704905 9 33208
MouseQuerySeq_38727057 2 2759
MouseQuerySeq_38709533 16 7776
MouseQuerySeq_38744814 1 131567
MouseQuerySeq_38727572 1 131567
MouseQuerySeq_38705387 9 33208
MouseQuerySeq_38735787 33 39107
MouseQuerySeq_38718587 2 2759
MouseQuerySeq_3876 36 10090
MouseQuerySeq_38736151 10 6072
MouseQuerySeq_38731984 1 131567
MouseQuerySeq_38714535 1 131567
MouseQuerySeq_38718132 2 2759
MouseQuerySeq_38700001 2 2759
MouseQuerySeq_38728030 13 7711
MouseQuerySeq_38714027 3 1452644
MouseQuerySeq_38723446 1 131567
MouseQuerySeq_38704436 9 33208
MouseQuerySeq_38700535 16 7776
MouseQuerySeq_38722609 1 131567
MouseQuerySeq_38709989 19 8287
MouseQuerySeq_38740706 2 2759
MouseQuerySeq_38723011 2 2759
MouseQuerySeq_38740230 10 6072
```

As we can see, the resulting output is divided into three sections. First section "`<Species taxomony Id>`" contains the information about query species taxonomy identifier used in the analysis. Second "`<Phylogeny used>`" shows the information about tree nodes used in stratification procedure. Colloquial term often used for this set of numbers is "phylogeny information". It should be noted that `PhyloStrat` always computes the longest set of nodes from root to specified leaf node (query species leaf node labelled by its taxonomy identifier). If the situation requires a subset of those, the two strategies can be applied in order to achieve this. One is to edit the species tree (`nodes` file) accordingly and the other is to use `MapEditor.pl` script to parse the obtained output (see section 5).

4.3.4 Known problems

1. Destructor is not properly implemented.
2. Error management not complete.

4.3.5 Future work

1. Resolve bugs.
2. Replace file streams with memory maps.
3. Disable syncing.

4.4 HDSearch

HDSearch software tool implements the filtering strategy for a fast retrieval of X number of sequences from a database that have the highest chance of containing a homologous region to a given query sequence. The filtering strategy is explained in the thesis and presents the underlying acceleration strategy used in `QPhyloStrat` program for fast genome stratification.

4.4.1 Program options

In order to see program options type:

```
./bin/HDSearch -h
```

Expected output:

```
Usage: ./program [options]

********************************************************************************
                                     HDSearch
                                        by
                                  Robert Bakaric

CONTACT:
 Code written and maintained by Robert Bakaric,
 email: rbakaric@irb.hr , bakaric@evolbio.mpg.de

ACKNOWLEDGEMENT:

LICENSE:
 The program is distributed under the GNU General Public License. You should have
 received a copy of the licence together  with this software. If not, see
 http://www.gnu.org/licenses/
********************************************************************************


Allowed options:
  -h [ --help ]                    produce help message
  -v [ --version ]                 print version information
  -q [ --query ] arg               Query file - fasta format (required)
  -s [ --subject ] arg             Subject file - fasta format (required)
  -k [ --kmer ] arg (=3)           K-mer size
  -g [ --segsize ] arg (=300)      Segment size
  -x [ --X ] arg (=10)             The number of closest sequences
  -r [ --range ] arg (=20)         Range query radius
  -m [ --matrix ] arg (=BLOSUM62) Scoring matrix
  -c [ --score ] arg (=17)         Cutoff score value
```

4.4.2 Input Data

HDSearch operates on two flat sequence files (fasta format), one of which is used as a query and the other as subject. In the search process for each query sequence the program computes a set of the most similar (according to JaccScore(d_{kt}) value) sequences from the subject file. An example of the input file can be found in `./examples/data/MouseSample.fa` and it should look like this:

```
>ENSMUSP00000017839 pep:known chromosome:GRCm38:11:80183851:80199757:1 ...
MAAVCSGNAVPVWLSEDDLSCIICQGLLDQPTTLPCGHSFCLRCLHDLWVSKRGAVDGCPWACPICRKGPLTKPKLHKN
PLLQDLVDKYLQAAREVEAGSEPEPAPAPRSAPQVTVQKSTTNVIQELTDMVRQLVDDVKSLQTQRPNLGSGQDNAQGT
PPTDSSSEGEHSLDSPKLVTFSISQKKIQEILHNLEEIQEKLQGSVPGRAPPRERVQEMTSSLCLLPDQRRPAPRKASH
LSLWAISPTFDLRTLSYNLEVSNNSRRVTVSRGDLHTYHWSPQRFSISQVFCSQALSSGQKYWEVDTRNCSHWAIGVAS
```

```
WGMKRDGMLGRTMDSWCIEWRGPGQFSAWAKMKKTDLQSDLPEVVGVWLDLESGELAFYAVADHERLLYECEVSSSSPL
HPAFWLYGLSPGNYLEIKQLNT
>ENSMUSP00000088808 pep:known chromosome:GRCm38:13:104173722:104178439:-1 ...
MDSCRMTTEVILHYRPYENDPKQLAKIAENVIQDFPTHPLPRFIPWFPYDESKLPLKPERLPPVISEEAAESVKQYLAI
SEPGVKSQSYDCTVDLLEFQPSSKLQHFIQSHTVKEQTNAAHLDKNSGKEKQHKQRSWSVSLASSHCPEKIFPLSRKLQ
ASLRTLHLHSFHRARWTLEYSVCNNQTLEDIWTKLNRLIRRDELPSCNATIQRQLGQIWVFCDIKCCEYVGNLLKERLS
LIGKIDLFVHKYGVIFSM
...
```

4.4.3 Usage Example

A complete example script can be found in `./examples/HDSearch.sh`.

Running the script:

```
sh HDSearch.sh
```

the following command is executed:

```
../bin/HDSearch -q examples/data/MouseSample.fa  -s examples/data/MouseSample.fa
```

Upon completion the expected result should look like this:

```
ENSMUSP00000017839 ENSMUSP00000017839 90000
ENSMUSP00000017839 ENSMUSP00000048309 3195
ENSMUSP00000017839 ENSMUSP00000103771 2678
ENSMUSP00000017839 ENSMUSP00000029971 2678
ENSMUSP00000017839 ENSMUSP00000125263 2347
ENSMUSP00000017839 ENSMUSP00000022225 2052
ENSMUSP00000017839 ENSMUSP00000070767 2045
ENSMUSP00000017839 ENSMUSP00000116947 1268
ENSMUSP00000017839 ENSMUSP00000088808 784
ENSMUSP00000088808 ENSMUSP00000088808 65025
ENSMUSP00000088808 ENSMUSP00000048309 1464
ENSMUSP00000088808 ENSMUSP00000125263 1265
ENSMUSP00000088808 ENSMUSP00000017839 784
ENSMUSP00000088808 ENSMUSP00000070767 708
ENSMUSP00000088808 ENSMUSP00000022225 701
ENSMUSP00000088808 ENSMUSP00000029971 626
ENSMUSP00000088808 ENSMUSP00000103771 390
ENSMUSP00000048309 ENSMUSP00000048309 90000
ENSMUSP00000048309 ENSMUSP00000125263 64090
ENSMUSP00000048309 ENSMUSP00000070767 4382
ENSMUSP00000048309 ENSMUSP00000022225 3653
ENSMUSP00000048309 ENSMUSP00000017839 3195
ENSMUSP00000048309 ENSMUSP00000103771 2764
ENSMUSP00000048309 ENSMUSP00000029971 2764
ENSMUSP00000048309 ENSMUSP00000116947 2084
ENSMUSP00000048309 ENSMUSP00000088808 1464
ENSMUSP00000125263 ENSMUSP00000125263 90000
ENSMUSP00000125263 ENSMUSP00000048309 64090
ENSMUSP00000125263 ENSMUSP00000022225 2772
ENSMUSP00000125263 ENSMUSP00000103771 2432
ENSMUSP00000125263 ENSMUSP00000029971 2432
ENSMUSP00000125263 ENSMUSP00000017839 2347
ENSMUSP00000125263 ENSMUSP00000070767 2288
ENSMUSP00000125263 ENSMUSP00000116947 1527
ENSMUSP00000125263 ENSMUSP00000088808 1265
ENSMUSP00000125263 ENSMUSP00000124760 109
ENSMUSP00000029971 ENSMUSP00000029971 90000
ENSMUSP00000029971 ENSMUSP00000103771 90000
ENSMUSP00000029971 ENSMUSP00000116947 44366
ENSMUSP00000029971 ENSMUSP00000048309 2764
ENSMUSP00000029971 ENSMUSP00000017839 2678
ENSMUSP00000029971 ENSMUSP00000125263 2432
ENSMUSP00000029971 ENSMUSP00000022225 1858
ENSMUSP00000029971 ENSMUSP00000124760 1554
ENSMUSP00000029971 ENSMUSP00000070767 1228
ENSMUSP00000103771 ENSMUSP00000103771 90000
ENSMUSP00000103771 ENSMUSP00000029971 90000
```

```
ENSMUSP00000103771 ENSMUSP00000116947 44366
ENSMUSP00000103771 ENSMUSP00000048309 2764
ENSMUSP00000103771 ENSMUSP00000017839 2678
ENSMUSP00000103771 ENSMUSP00000125263 2432
ENSMUSP00000103771 ENSMUSP00000022225 1858
ENSMUSP00000103771 ENSMUSP00000124760 1554
ENSMUSP00000103771 ENSMUSP00000070767 1228
ENSMUSP00000124760 ENSMUSP00000124760 2116
ENSMUSP00000124760 ENSMUSP00000103771 1554
ENSMUSP00000124760 ENSMUSP00000029971 1554
ENSMUSP00000124760 ENSMUSP00000116947 1545
ENSMUSP00000124760 ENSMUSP00000048309 169
ENSMUSP00000124760 ENSMUSP00000017839 126
ENSMUSP00000124760 ENSMUSP00000125263 109
ENSMUSP00000116947 ENSMUSP00000103771 44366
ENSMUSP00000116947 ENSMUSP00000029971 44366
ENSMUSP00000116947 ENSMUSP00000116947 44100
ENSMUSP00000116947 ENSMUSP00000048309 2084
ENSMUSP00000116947 ENSMUSP00000124760 1545
ENSMUSP00000116947 ENSMUSP00000125263 1527
ENSMUSP00000116947 ENSMUSP00000017839 1268
ENSMUSP00000116947 ENSMUSP00000070767 994
ENSMUSP00000116947 ENSMUSP00000022225 649
ENSMUSP00000022225 ENSMUSP00000022225 90000
ENSMUSP00000022225 ENSMUSP00000070767 66083
ENSMUSP00000022225 ENSMUSP00000048309 3653
ENSMUSP00000022225 ENSMUSP00000125263 2772
ENSMUSP00000022225 ENSMUSP00000017839 2052
ENSMUSP00000022225 ENSMUSP00000103771 1858
ENSMUSP00000022225 ENSMUSP00000029971 1858
ENSMUSP00000022225 ENSMUSP00000088808 701
ENSMUSP00000022225 ENSMUSP00000116947 649
ENSMUSP00000070767 ENSMUSP00000070767 90000
ENSMUSP00000070767 ENSMUSP00000022225 66083
ENSMUSP00000070767 ENSMUSP00000048309 4382
ENSMUSP00000070767 ENSMUSP00000125263 2288
ENSMUSP00000070767 ENSMUSP00000017839 2045
ENSMUSP00000070767 ENSMUSP00000103771 1228
ENSMUSP00000070767 ENSMUSP00000029971 1228
ENSMUSP00000070767 ENSMUSP00000116947 994
ENSMUSP00000070767 ENSMUSP00000088808 708
```

Evidently the output consists of three columns. From left to right columns are defined as:

1. Query sequence identifier
2. Subject sequence identifier
3. JaccScore value

4.4.4 Known problems

Non reported.

4.4.5 Future work

There is a potential to improve upon both speed and sensitivity of the search. For example, runtime improvements can be achieved by reimplementing the procedure for processing overlaps. Instead of separately processing each segment, each overlap can be simply re-indexed twice, thus reducing the processing and therefore the overall runtime in half.

Certainly the sensitivity improvements can be achieved by not only considering exact matching k-mers but also similar ones (where similarity is defined according to score values in BLOSUM or PAM)

4.5 QPhyloStrat

`QPhyloStrat` is a software tool with one hit strategy basic local alignment search algorithm (`QuickBlast` algorithm) implemented. It runs in two separate modes, best (-Q B) and fast (-Q F). The best mode preforms exhaustive `QuickBlast` search across the entire database section as defined by a query species phylogeny. The fast mode includes an additional pre-filtering step done using `HDSearch` strategy. Here the utility of both methods is demonstrated on a small demo case.

4.5.1 Program options

In order to see program options type:

```
./bin/QPhyloStrat -h
```

Expected output:

```
Usage: ./program [options]

*********************************************************************************
                                    QPhyloStrat
                                         by
                                   Robert Bakaric

CONTACT:
 Code written and maintained by Robert Bakaric,
 email: rbakaric@irb.hr , bakaric@evolbio.mpg.de
ACKNOWLEDGEMENT:
         http://www.ncbi.nlm.nih.gov/toolkit

LICENSE:
 The program is distributed under the GNU General Public License. You should have
 received a copy of the licence together  with this software. If not, see
 http://www.gnu.org/licenses/
*********************************************************************************


Allowed options

WARNING:  Options blastn, XNU, are still under construction and thus
cannot be used ! Current setup only supoorts blastp and SEG and DUST filtering:
  -h [ --help ]                         produce help message
  -v [ --version ]                      print version information
  -p [ --program ] arg (=blastp)        Type of BLAST program [blastp/blastn]
  -f [ --filter ] arg (=SEG)            Sequence filter [SEG/XNU/DUST]

  -t [ --taxid ] arg                    Query species taxonomy identifier
                                        (required)
  -n [ --nodes ] arg                    Nodes file [tabular - two columns <child
                                        \t parent>]  (required)
  -q [ --query ] arg                    Query file - fasta format (required)
  -d [ --database ] arg                 Database location - path (required)
  -e [ --evalue ] arg (=0.001)          Expectation value (E) threshold for
                                        saving hits
  -m [ --matrix ] arg (=BLOSUM62)       Scoring matrix
  -z [ --effective_db_size ] arg (=0)   Effective length of the database
  -O [ --gap_open ] arg (=6)            Cost to open a gap
  -E [ --gap_extend ] arg (=2)          Cost to extend a gap
  -W [ --wordsize ] arg (=3)            Word size
  -Y [ --gap_x_dropoff ] arg (=7)       X dropoff value for ungapped extensions
                                        in bits
  -Q [ --q_mode ] arg (=F)              QPhyloStrat mode: [fast(F)/best(B)]


  -- HD filter options --

  -k [ --kmer ] arg (=3)                K-mer size
```

```
-g [ --segsize ] arg (=300)         Segment size
-x [ --X ] arg (=10)                The number of closest sequences
-r [ --range ] arg (=20)            Range query radius
-c [ --score ] arg (=17)            Cutoff score value
```

4.5.2 Input files

The program requires a set of input parameters, examples of which can be found in `./examples/data/nr-Test` directory. These include:

`TestDb-phdb` A formatted database generated by the `MakePhyloDb` program (See an example in sec. 4.1)

`TestMouse.fa` A query species fasta file (See an example in sec. 4.4)

`nodes` Tab delimited two column adjacency list containing the information about taxonomic relations (See an example in sec. 4.3)

4.5.3 Usage Example

Running QPhyloStrat in best mode.

A complete example script can be found in `./examples/QPhyloStart-best.sh`.

Running the script:

```
sh QPhyloStart-best.sh
```

the following command is executed:

```
../bin/QPhyloStrat -t 10090 -n ./data/nodes  \
                   -q ./data/nr-Test/TestMouse.fa \
                   -d ./data/nr-Test/TestDb-phdb/ -Q B
```

The obtained result should look like:

```
<Species taxonomy Id>
10090
<Phylogeny used>
1 131567
2 2759
3 1452644
4 1452646
5 33154
6 1452651
7 1452652
8 1452653
9 33208
10 6072
11 33213
12 33511
13 7711
14 1452661
15 7742
16 7776
17 117570
18 117571
19 8287
20 1338369
21 32523
22 32524
23 40674
24 32525
25 9347
```

```
26 1437010
27 314146
28 314147
29 9989
30 33553
31 337687
32 10066
33 39107
34 10088
35 862507
36 10090
<Phylostratigraphy map>
pgi|38700001|ti|10090|pi|0| 2 2759
pgi|38700535|ti|10090|pi|0| 16 7776
pgi|38704436|ti|10090|pi|0| 2 2759
pgi|38704905|ti|10090|pi|0| 9 33208
pgi|38705387|ti|10090|pi|0| 9 33208
pgi|38709112|ti|10090|pi|0| 2 2759
pgi|38709533|ti|10090|pi|0| 16 7776
pgi|38709989|ti|10090|pi|0| 19 8287
pgi|38713615|ti|10090|pi|0| 9 33208
pgi|38714027|ti|10090|pi|0| 3 1452644
pgi|38714535|ti|10090|pi|0| 2 2759
pgi|38718132|ti|10090|pi|0| 2 2759
pgi|38718587|ti|10090|pi|0| 2 2759
pgi|38718975|ti|10090|pi|0| 2 2759
pgi|38722609|ti|10090|pi|0| 2 2759
pgi|38723011|ti|10090|pi|0| 2 2759
pgi|38723446|ti|10090|pi|0| 10 6072
pgi|38727057|ti|10090|pi|0| 2 2759
pgi|38727572|ti|10090|pi|0| 5 33154
pgi|38728030|ti|10090|pi|0| 9 33208
pgi|38731984|ti|10090|pi|0| 3 1452644
pgi|38735787|ti|10090|pi|0| 24 32525
pgi|38736151|ti|10090|pi|0| 10 6072
pgi|38740230|ti|10090|pi|0| 10 6072
pgi|38740706|ti|10090|pi|0| 2 2759
pgi|38744814|ti|10090|pi|0| 2 2759
```

In the same way by setting the `-Q` option to "F" the fast mode is set and the computation can take place. A complete example script can be found in `./examples/QPhyloStart-fast.sh`.

4.5.4 Known problems

Non reported.

4.5.5 Future work

Include threading option.

4.6 PhyloClust

PhyloClust is a simple clustering algorithm for estimating the number of GFGEs (Gene Family Gain Events) at each (individual species or a group) phylostrata level.

4.6.1 Program options

In order to see program options type:

```
./bin/PhyloClust -h
```

Expected output:

```
Usage: ./program [options]

*******************************************************************************
                                    PhyloClust
                                        by
                                   Robert Bakaric

CONTACT:
 Code written and maintained by Robert Bakaric,
 email: rbakaric@irb.hr , bakaric@evolbio.mpg.de

ACKNOWLEDGEMENT:
         http://www.ncbi.nlm.nih.gov/toolkit

LICENSE:
 The program is distributed under the GNU General Public License. You should have
 received a copy of the licence together  with this software. If not, see
 http://www.gnu.org/licenses/
*******************************************************************************


Options
:
  -h [ --help ]                         Produce help message
  -v [ --version ]                      Print version information
  -p [ --program ] arg (=blastp)        Type of BLAST program [blastp/blastn]
  -f [ --filter ] arg (=SEG)            Sequence filter [SEG/XNU/DUST]

  -t [ --taxid ] arg                    Query species taxonomy identifier
                                        (required)
  -d [ --database ] arg                 Database location - path (required)
  -c [ --cutoff ] arg (=99)             Cluster cutoff percent identity
  -r [ --maprep ] arg                   Map repository location - path (required)

  -e [ --evalue ] arg (=0.001)          Expectation value (E) threshold for
                                        saving hits
  -m [ --matrix ] arg (=BLOSUM62)       Scoring matrix
  -z [ --effective_db_size ] arg (=0)   Effective length of the database
  -O [ --gap_open ] arg (=6)            Cost to open a gap
  -E [ --gap_extend ] arg (=2)          Cost to extend a gap
  -W [ --wordsize ] arg (=3)            Word size
  -Y [ --gap_x_dropoff ] arg (=7)       X dropoff value for ungapped extensions
                                        in bits
  -q [ --quite ] arg (=0)               Quite mode
  -F [ --full ] arg (=0)                Full cluster report
```

4.6.2 Input data

The program requires two sets of input files, examples of which can be found in `./examples/data/nr-Test` directory. These include:

`TestDb-phdb` A formatted database generated with `MakePhyloDb` program (See an example in sec. 4.1)

`Test-MapRep` A set of gene gain maps. Each map is required to be properly named by changing its generic name (defined by a user) into `taxonomy_identifier.phmap`. Example:

`my_mouse_map_file` → `10090.phmap`

4.6.3 Usage Example

A complete example script can be found in `./examples/PhyloClust.sh`.

Running the script:

```
sh PhyloClust.sh
```

the following command is executed:

```
../bin/PhyloClust -t 6072 -d data/nr-Test/TestDb-phdb/ \
                  -r data/nr-Test/Test-MapRep/ -q true
```

The expected result should look like:

```
* pgi|000000000000954419|ti|9544|pi|0| pgi|000000000000954419|ti|9544|pi|0|,\
                                       pgi|000000000000954420|ti|9544|pi|0|,\
                                       pgi|000000000000954443|ti|9544|pi|0|,\
                                       pgi|000000000000954444|ti|9544|pi|0|,\
                                       pgi|000000000010090271|ti|10090|pi|0|,\
                                       pgi|000000000010090619|ti|10090|pi|0|,
* pgi|000000000000954442|ti|9544|pi|0| pgi|000000000000954419|ti|9544|pi|0|,\
                                       pgi|000000000000954420|ti|9544|pi|0|,\
                                       pgi|000000000000954442|ti|9544|pi|0|,\
                                       pgi|000000000000954443|ti|9544|pi|0|,\
                                       pgi|000000000000954444|ti|9544|pi|0|,\
                                       pgi|000000000010090271|ti|10090|pi|0|,
* pgi|000000000000954449|ti|9544|pi|0| pgi|000000000009544309|ti|9544|pi|0|,\
                                       pgi|000000000010090112|ti|10090|pi|0|,\
                                       pgi|000000000010090266|ti|10090|pi|0|,\
                                       pgi|000000000010090506|ti|10090|pi|0|,
...
```

Each line represents a cluster where the first identifier is a cluster representative (a sequence with the highest number of homology connections) and each following, the species sequence cluster representative. Each individual sequence can further be obtained by setting the `-full` option to `true`

Note: it is crucial to preserve the pgi-ti-pi identifier structure if the results are used as input data for `PhLoG` program, otherwise, sequence identifier is an arbitrary construct and can take any form as long as ASCII encoding is applied.

4.6.4 Known problems

Non reported.

4.6.5 Future work

Future work includes improving upon clustering technique by possibly including average of even complete-linkage clustering in order to improve upon method sensitivity.

4.7 PhLoG

PhLoG is a software tool designed to trace the gene family loss events (GFLEs). Given a set of gene family formation events it uses the phylogeny information about the relatedness of different species in order to locate the linage and/or a clade in which no evidence, no homologous gene, member of a given family gain event, exists.

4.7.1 Program options

In order to see program options type:

```
./bin/PhLoG -h
```

Expected output:

```
Usage: ./program [options]

********************************************************************************
                                        PhLoG
                                          by
                                    Robert Bakaric

CONTACT:
 Code written and maintained by Robert Bakaric,
 email: rbakaric@irb.hr , bakaric@evolbio.mpg.de

ACKNOWLEDGEMENT:

LICENSE:
 The program is distributed under the GNU General Public License. You should have
 received a copy of the licence together  with this software. If not, see
 http://www.gnu.org/licenses/
********************************************************************************


Options:
  -h [ --help ]            Produce help message
  -v [ --version ]         Print version information
  -t [ --taxid ] arg       Query species taxonomy identifier (required)
  -c [ --cluster_rep ] arg Cluster repository - path (required)
  -n [ --nodes ] arg       Nodes file (required)
```

4.7.2 Input data

The program requires a set of input files (outputs from PhyloClust tool) all placed within one directory. These files need to named according to a particular naming convention such that each file consists of a taxonomy identifier of phylostrata clusters it represents, followed by `.phclust` extension. As an example take a look at `./examples/data/clust/` directory:

```
117571.phclust  1452644.phclust  2759.phclust
33154.phclust   33208.phclust    6072.phclust
7711.phclust
```

In addition `nodes` file 4.3, containing the information between species is to be provided.

4.7.3 Usage Example

A complete example script can be found in `./examples/PhLoG.sh`.

Running the script:

```
sh PhLoG.sh
```

the following command is executed:

```
../bin/PhLoG -t 10090 -c data/clust/ -n data/nodes
```

Upon completion the expected result should look like this:

```
<Species taxomony Id>
10090
<GainLoss Summary Table>
<#Gain #Loss PS TaxId>
0 0 1 131567
37 0 2 2759
7 0 3 1452644
0 0 4 1452646
5 0 5 33154
```

```
0 0 6 1452651
0 0 7 1452652
0 0 8 1452653
7 0 9 33208
14 0 10 6072
0 0 11 33213
0 0 12 33511
1 0 13 7711
0 2 14 1452661
0 0 15 7742
0 0 16 7776
0 0 17 117570
3 0 18 117571
0 0 19 8287
0 0 20 1338369
0 0 21 32523
0 0 22 32524
0 0 23 40674
0 0 24 32525
0 0 25 9347
0 0 26 1437010
0 0 27 314146
0 7 28 314147
0 0 29 9989
0 0 30 33553
0 0 31 337687
0 0 32 10066
0 0 33 39107
0 0 34 10088
0 0 35 862507
0 0 36 10090
<Number of founders in 10090 with a possibility to be lost in the future: 65>
<Cluster representatives PSGain TaxId PSLoss TaxId>
pgi|0000000000000077391|ti|7739|pi|0| 9 33208 0 0
pgi|0000000000000077393|ti|7739|pi|0| 2 2759 0 0
pgi|0000000000000095442|ti|9544|pi|0| 9 33208 0 0
pgi|0000000000000100901|ti|10090|pi|0| 2 2759 0 0
pgi|0000000000000100902|ti|10090|pi|0| 9 33208 0 0
pgi|0000000000000100903|ti|10090|pi|0| 3 1452644 28 314147
pgi|0000000000000100906|ti|10090|pi|0| 9 33208 0 0
pgi|0000000000000773911|ti|7739|pi|0| 2 2759 0 0
pgi|0000000000000773935|ti|7739|pi|0| 2 2759 0 0
pgi|0000000000000773946|ti|7739|pi|0| 2 2759 0 0
pgi|0000000000000773956|ti|7739|pi|0| 2 2759 0 0
pgi|0000000000000773978|ti|7739|pi|0| 5 33154 14 1452661
pgi|0000000000000954419|ti|9544|pi|0| 10 6072 0 0
pgi|0000000000000954422|ti|9544|pi|0| 2 2759 0 0
pgi|0000000000000954430|ti|9544|pi|0| 18 117571 28 314147
pgi|0000000000000954442|ti|9544|pi|0| 10 6072 0 0
pgi|0000000000000954446|ti|9544|pi|0| 3 1452644 0 0
pgi|0000000000000954449|ti|9544|pi|0| 10 6072 0 0
pgi|0000000000000954451|ti|9544|pi|0| 2 2759 0 0
pgi|0000000000000954453|ti|9544|pi|0| 2 2759 0 0
pgi|0000000000000954464|ti|9544|pi|0| 3 1452644 0 0
pgi|0000000000000954467|ti|9544|pi|0| 2 2759 0 0
pgi|0000000000000954469|ti|9544|pi|0| 9 33208 0 0
pgi|0000000000000954479|ti|9544|pi|0| 3 1452644 0 0
pgi|0000000000000954480|ti|9544|pi|0| 9 33208 0 0
pgi|0000000000000954483|ti|9544|pi|0| 2 2759 0 0
pgi|0000000000000954484|ti|9544|pi|0| 2 2759 0 0
pgi|0000000000000954485|ti|9544|pi|0| 2 2759 28 314147
pgi|0000000000001009011|ti|10090|pi|0| 9 33208 0 0
pgi|0000000000001009016|ti|10090|pi|0| 18 117571 0 0
pgi|0000000000001009022|ti|10090|pi|0| 10 6072 0 0
pgi|0000000000007739106|ti|7739|pi|0| 2 2759 0 0
pgi|0000000000007739132|ti|7739|pi|0| 5 33154 28 314147
pgi|0000000000007739148|ti|7739|pi|0| 2 2759 0 0
pgi|0000000000007739176|ti|7739|pi|0| 2 2759 0 0
pgi|0000000000007739185|ti|7739|pi|0| 2 2759 0 0
pgi|0000000000007739270|ti|7739|pi|0| 2 2759 0 0
```

```
pgi|00000000000009544101|ti|9544|pi|0| 10 6072 0 0
pgi|00000000000009544132|ti|9544|pi|0| 18 117571 28 314147
pgi|00000000000009544143|ti|9544|pi|0| 10 6072 0 0
pgi|00000000000009544230|ti|9544|pi|0| 2 2759 0 0
pgi|00000000000009544252|ti|9544|pi|0| 2 2759 14 1452661
pgi|00000000000009544295|ti|9544|pi|0| 10 6072 0 0
pgi|00000000000010090109|ti|10090|pi|0| 2 2759 0 0
pgi|00000000000010090112|ti|10090|pi|0| 10 6072 0 0
pgi|00000000000010090115|ti|10090|pi|0| 2 2759 0 0
pgi|00000000000010090139|ti|10090|pi|0| 5 33154 28 314147
pgi|00000000000010090172|ti|10090|pi|0| 2 2759 0 0
pgi|00000000000010090182|ti|10090|pi|0| 2 2759 0 0
pgi|00000000000010090215|ti|10090|pi|0| 2 2759 0 0
pgi|00000000000010090244|ti|10090|pi|0| 13 7711 28 314147
pgi|00000000000010090262|ti|10090|pi|0| 2 2759 0 0
pgi|00000000000010090266|ti|10090|pi|0| 10 6072 0 0
pgi|00000000000010090271|ti|10090|pi|0| 10 6072 0 0
pgi|00000000000010090288|ti|10090|pi|0| 3 1452644 0 0
pgi|00000000000010090293|ti|10090|pi|0| 2 2759 0 0
pgi|00000000000010090297|ti|10090|pi|0| 10 6072 0 0
pgi|00000000000010090298|ti|10090|pi|0| 2 2759 0 0
pgi|00000000000010090376|ti|10090|pi|0| 5 33154 0 0
pgi|00000000000010090382|ti|10090|pi|0| 3 1452644 0 0
pgi|00000000000010090401|ti|10090|pi|0| 2 2759 0 0
pgi|00000000000010090421|ti|10090|pi|0| 5 33154 0 0
pgi|00000000000010090431|ti|10090|pi|0| 2 2759 0 0
pgi|00000000000010090475|ti|10090|pi|0| 2 2759 0 0
pgi|00000000000010090478|ti|10090|pi|0| 2 2759 0 0
pgi|00000000000010090480|ti|10090|pi|0| 2 2759 0 0
pgi|00000000000010090503|ti|10090|pi|0| 10 6072 0 0
pgi|00000000000010090504|ti|10090|pi|0| 2 2759 0 0
pgi|00000000000010090506|ti|10090|pi|0| 10 6072 0 0
pgi|00000000000010090542|ti|10090|pi|0| 2 2759 0 0
pgi|00000000000010090561|ti|10090|pi|0| 3 1452644 0 0
pgi|00000000000010090562|ti|10090|pi|0| 2 2759 0 0
pgi|00000000000010090619|ti|10090|pi|0| 10 6072 0 0
pgi|00000000000010090706|ti|10090|pi|0| 2 2759 0 0
```

Generated output consists of the information about query species taxonomy identifier, summary table and a full list of cluster representatives mapping their entrance and exit locations.

In summary table a brief overlook on how many GFGEs and GFLEs are associated to a given phylostrata group. It should be noted that the GFGE count includes only those not lost in the course of evolution. In order to extract the full information further parsing of *Cluster representatives table* is required in which, if the value is set to 0 in the loss column, the corresponding GFGE still has its family members present within the initial query species linage.

4.7.4 Known problems

Non reported

4.7.5 Future work

Upon request.

4.8 MakeTree

MakeTree is a simple tool for editing phylogeny information. As a result the Newick standard tree representation is generated, which can then further be passed into one of the free online tools for visualisation and further editing, or as an input to `TreeIlustrator.pl`.

4.8.1 Program options

In order to see program options type:

```
./bin/MakeTree -h
```

Expected output:

```
Usage: ./program [options]

********************************************************************************
                                    MakeTree
                                       by
                                  Robert Bakaric

CONTACT:
 Code written and maintained by Robert Bakaric,
 email: rbakaric@irb.hr , bakaric@evolbio.mpg.de
ACKNOWLEDGEMENT:
         http://www.ncbi.nlm.nih.gov/toolkit

LICENSE:
 The program is distributed under the GNU General Public License. You should have
 received a copy of the licence together  with this software. If not, see
 http://www.gnu.org/licenses/
********************************************************************************


Allowed options:
  -h [ --help ]                Produce help message
  -v [ --version ]             Print version information
  -n [ --nodes ] arg           Nodes file [tabular - two columns <child \t
                               parent>]  (required)
  -m [ --names ] arg           Names file [tabular]  (required)
  -i [ --instructions ] arg    Instrictions file [tabular] (required)
  -s [ --species ] arg         List of selected species
  -t [ --start ] arg (=131567) Start node
  -d [ --depth ] arg (=3)      Tree depth. (if set to -1 then full depth is
                               considered)
```

4.8.2 Input data

In order to generate the Newick tree, the program requires four input files. The first is the adjacency list containing the information about node child-parent relations. This file is very similar to the one that can be obtained from ncbi taxonomy repository, however much simpler. As an example consider `./examples/data/nodes` file:

```
2       131567
6       335928
7       6
9       32199
10      135621
11      1707
13      203488
14      13
16      32011
17      16
...
```

Second file is the names file. It contains the information about taxonomy identifiers located within the nodes file.

`examples/data/names`:

```
1       all     ~       synonym
1       root    ~       scientific_name
2       Bacteria        Bacteria_<prokaryote>   scientific_name
2       Monera  Monera_<Bacteria>       in-part
2       Procaryotae     Procaryotae_<Bacteria>  in-part
2       Prokaryota      Prokaryota_<Bacteria>   in-part
2       Prokaryotae     Prokaryotae_<Bacteria>  in-part
2       bacteria        bacteria_<blast2>       blast_name
2       eubacteria      ~       genbank_common_name
...
```

Note: names file is a four column tab delimited file with no empty columns. Other restrictions associated to that file are:

- No spaces allowed (spaces can be replaced with _)
- No round brackets allowed (in the above case "()" are replaced with "<>")
- No square brackets allowed (in the above case "[]" are replaced with "<>")
- No curly brackets allowed (in the above case "" are replaced with "<>")
- No dots (in the above case "." are replaced with "−")
- No semicolons allowed (in the above case ";" are replaced with "−")
- No colons allowed (in the above case ":" are replaced with "−")
- No non-ASCII characters allowed

The "names" file can be easily created from the ncbi version of the equivalent by introducing the mentioned changes or executing the `NCBIDataParser.pl` on it.

Third file is a list file. This file specifies which end species (leaf nodes the user wishes to see in his/hers final Newick tree). Again, this file is a two column tab delimited table with species taxonomy identifier in the first column and species name in the second (note that the second column does not necessarily needs to contain only species name. Moreover, it can be left empty if desired). Example:

`examples/data/species_list`:

```
1452671 Cheiracus sulcatus
100027  Hysterium pulicare
100044  Pleomassaria siparia
100047  Melanomma pulvis-pyrius
1001833 This species is strange...
...
```

Finally, the fourth file is an instruction list. This file contains the set of edit operations user wishes to apply to its tree. These are:

1. **Add(*)** add node (species node)
2. **Del** delete node
3. **Rel** relocate node
4. **Er** erase node

Add(*) function adds a new node to the existing tree. The proper syntax includes:

```
Add Robert_Bakaric 9606   This is how you add a node
```

where in the first column user specifies the type of operation, in the second column the node name is declared (no spaces) and in the third, taxonomy identifier of the parent node is defined. Taxonomy identifier assigned to a new species is automatically archived and user is not allowed to manipulate with it, as such action can cause cycles in the underlying phylogeny tree and ultimately lead to an error. By adding * symbol to Add function (Add*), the user explicitly declares that the added node is a leaf node (species name) required to be included in the visualisation (Newick) output

Del function deletes a specified node and the entire calde it represents. The proper syntax for this function is:

```
Del 10239  This is how you delete the entire Virus clade
```

Again, the first column is the edit operation one wishes to invoke upon the second column, where taxonomy identifier to which this operation should be applied is located.

Er Unlike deletion, erasing the node implies removing only that particular node from the tree leaving the clade attached to its parent. Syntax example for this function is:

```
Er 10239  This is how you delete only Virus \
          identifier attaching the clade to root (1)
```

Rel Relocation assumes moving a taxonomy identifier and its entire clade to a new location:

```
Rel    428574  9606    Hydra AEP has been moved to H.sapiens
```

The above function states that a user wishes to relocate a clade starting at node 428574 to 9606, making 9606 a parent of 428574 and all its descendants.

The additional syntax regarding the instruction file includes # symbol, which if placed at the beginning of a line declares it as a comment line and as such is not included in the editing process.

In all cases the last column in the instruction file is reserved for comments by default. Therefore, by default is treated as if the # symbol is placed before it.

4.8.3 Usage Example

A complete example script can be found in `./examples/MakeTree.sh`.

Running the script:

```
sh MakeTree.sh
```

the following command is executed:

```
../bin/MakeTree -n data/nodes -m data/names -i data/instructions \
                -s data/species_list -t 6072
```

Expected result should look like:

```
(((Trichoplax_adhaerens[10228]:1.0)Trichoplax[10227]:1.0)Placozoa[10226]:1.0,
((Chordata[7711]:1.0,Ambulacraria[1452662]:1.0)Deuterostomia[33511]:1.0,
(Lophotrochozoa[1206795]:1.0,Ecdysozoa[1206794]:1.0)Protostomia[33317]:1.0,
(Trematoda[6178]:1.0)Platyhelminthes[6157]:1.0)Bilateria[33213]:1.0,
((Hexacorallia[6102]:1.0)Anthozoa[6101]:1.0,(Semaeostomeae[6143]:1.0)Scyphozoa[6142]:1.0,
(Hydroida[37516]:1.0)Hydrozoa[6074]:1.0)Cnidaria[6073]:1.0)Eumetazoa[6072]:1.0
```

Moreover, modified nodes and names file are saved directly to HDD as `names.new` and `nodes.new`

4.8.4 Known problems

Non reported

4.8.5 Future work

Create a more intuitive GUI for the software.

5 Auxiliary Software (Perl)

5.1 MapEditor

`MapEditor` is a tool for phylogeny based editing of computed phylostratigraphy maps. It is designed to merge insufficiently "saturated" phylostrata groups.

5.1.1 Program options

In order to see program options type:

```
perl ./scripts/MapEditor.pl -h
```

Expected output:

```
********************************************************************************
                                    MapEditor.pl
                                         by
                                   Robert Bakaric

CONTACT:
 Code written and maintained by Robert Bakaric,
 email: rbakaric@irb.hr , bakaric@evolbio.mpg.de

ACKNOWLEDGEMENT:


LICENSE:
 The program is distributed under the GNU General Public License. You should have
 received a copy of the licence together  with this software. If not, see
 http://www.gnu.org/licenses/
********************************************************************************


Usage: ./program [options]

-m map file [query <tab> PhyloId <tab> TaxId]
-p phylogeny file [PhyloId <tab> Taxid]
-o output map [query <tab> PhyloId <tab> TaxId]
-h this message
```

5.1.2 Input Files

`MapEditor` takes two files. First one is a simple two column (tab separated) file containing integers with the defined new phylogeny order and the second one is a phylostratigraphy map. In the phylogeny file the first column contains phylogeny identifier. This number corresponds to the depth of a corresponding node as visited on a path from the root of the phylogeny tree to its leaf node. The second number is the taxonomy identified (unique node label). An example of a phylogeny input file can be found in `./phylotoolkit-xxx/examples/data` and it should look like this:

`./examples/data/PhyloForMapEdit`:

```
1 131567
2 6072
3 7742
4 8287
```

The second file is described in section 5.8.3

5.1.3 Usage Example

By executing:

```
perl ./scripts/MapEditor.pl -p ./examples/data/PhyloForMapEdit \
                            -m ./examples/data/MouseSample.map
```

The obtained result is a recomputed phylostratography map that should look like this:

```
<Species taxomony Id>
10090
<Phylogeny used>
1 131567
2 6072
3 7742
4 8287
<Phylostratigraphy map>
MouseQuerySeq_38709112 1 131567
MouseQuerySeq_38713615 1 131567
MouseQuerySeq_38718975 1 131567
MouseQuerySeq_38704905 1 131567
MouseQuerySeq_38727057 1 131567
MouseQuerySeq_38709533 2 6072
MouseQuerySeq_38744814 1 131567
MouseQuerySeq_38727572 1 131567
MouseQuerySeq_38705387 1 131567
MouseQuerySeq_38735787 2 6072
MouseQuerySeq_38718587 1 131567
MouseQuerySeq_3876 2 6072
MouseQuerySeq_38736151 2 6072
MouseQuerySeq_38731984 1 131567
MouseQuerySeq_38714535 1 131567
MouseQuerySeq_38718132 1 131567
MouseQuerySeq_38700001 1 131567
MouseQuerySeq_38728030 2 6072
MouseQuerySeq_38714027 1 131567
MouseQuerySeq_38723446 1 131567
MouseQuerySeq_38704436 1 131567
MouseQuerySeq_38700535 2 6072
MouseQuerySeq_38722609 1 131567
MouseQuerySeq_38709989 2 6072
MouseQuerySeq_38740706 1 131567
MouseQuerySeq_38723011 1 131567
MouseQuerySeq_38740230 2 6072
```

Note that only nodes already specified within a phylostratigraphy map are allowed to be defined in `./examples/data/PhyloForMapEdit` file.

5.2 AddNames

`AddNames` is a software tool created from adding corresponding "names" information to an array of program outputs within the toolkit.

5.2.1 Program options

In order to see program options type:

```
perl ./scripts/AddNames.pl -h
```

Expected output:

```
********************************************************************************
                                    AddNames.pl
                                        by
                                  Robert Bakaric

CONTACT:
```

```
 Code written and maintained by Robert Bakaric,
 email: rbakaric@irb.hr , bakaric@evolbio.mpg.de

ACKNOWLEDGEMENT:


LICENSE:
 The program is distributed under the GNU General Public License. You should have
 received a copy of the licence together  with this software. If not, see
 http://www.gnu.org/licenses/
*******************************************************************************


Usage: ./program [options]

-m Map file [query <tab> PhyloId <tab> TaxId]
-n Names file [PhyloId <tab> Name <tab> type]
-h This message
```

5.2.2 Input Files

Program requires two input files. One is the output of any program having a taxonomy identifier in the last column within its result and the other is a tab delimited names table:

```
names:

1       all     ~       synonym
1       root    ~       scientific_name
2       Bacteria        Bacteria_<prokaryote>   scientific_name
2       Monera  Monera_<Bacteria>       in-part
2       Procaryotae     Procaryotae_<Bacteria>  in-part
2       Prokaryota      Prokaryota_<Bacteria>   in-part
2       Prokaryotae     Prokaryotae_<Bacteria>  in-part
2       bacteria        bacteria_<blast2>       blast_name
...
```

5.2.3 Usage Example

By executing:

```
perl ./scripts/AddNames.pl -n ./examples/data/names \
                           -m ./examples/data/MouseSample.map
```

The expected output should look like:

```
<Species taxomony Id>
10090
<Phylogeny used>
0 1 ~ : root
1 131567 root : cellular organisms
2 2759 cellular organisms : Eukaryota
3 1452644 Eukaryota : Unikonta
4 1452646 Unikonta : Apusozoa/Opisthokonta
5 33154 Apusozoa/Opisthokonta : Opisthokonta
6 1452651 Opisthokonta : Holozoa
7 1452652 Holozoa : Filozoa
8 1452653 Filozoa : Metazoa/Choanoflagellida
9 33208 Metazoa/Choanoflagellida : Metazoa
10 6072 Metazoa : Eumetazoa
11 33213 Eumetazoa : Bilateria
12 33511 Bilateria : Deuterostomia
13 7711 Deuterostomia : Chordata
14 1452661 Chordata : Olfactores
15 7742 Olfactores : Vertebrata
16 7776 Vertebrata : Gnathostomata
17 117570 Gnathostomata : Teleostomi
18 117571 Teleostomi : Euteleostomi
```

```
19 8287 Euteleostomi : Sarcopterygii
20 1338369 Sarcopterygii : Dipnotetrapodomorpha
21 32523 Dipnotetrapodomorpha : Tetrapoda
22 32524 Tetrapoda : Amniota
23 40674 Amniota : Mammalia
24 32525 Mammalia : Theria
25 9347 Theria : Eutheria
26 1437010 Eutheria : Boreoeutheria
27 314146 Boreoeutheria : Euarchontoglires
28 314147 Euarchontoglires : Glires
29 9989 Glires : Rodentia
30 33553 Rodentia : Sciurognathi
31 337687 Sciurognathi : Muroidea
32 10066 Muroidea : Muridae
33 39107 Muridae : Murinae
34 10088 Murinae : Mus
35 862507 Mus : Mus
36 10090 Mus : Mus musculus
<Phylostratigraphy map>
MouseQuerySeq_38709112 2 2759 cellular organisms : Eukaryota
MouseQuerySeq_38713615 9 33208 Metazoa/Choanoflagellida : Metazoa
MouseQuerySeq_38718975 1 131567 root : cellular organisms
MouseQuerySeq_38704905 9 33208 Metazoa/Choanoflagellida : Metazoa
MouseQuerySeq_38727057 2 2759 cellular organisms : Eukaryota
MouseQuerySeq_38709533 16 7776 Vertebrata : Gnathostomata
MouseQuerySeq_38744814 1 131567 root : cellular organisms
MouseQuerySeq_38727572 1 131567 root : cellular organisms
MouseQuerySeq_38705387 9 33208 Metazoa/Choanoflagellida : Metazoa
MouseQuerySeq_38735787 33 39107 Muridae : Murinae
MouseQuerySeq_38718587 2 2759 cellular organisms : Eukaryota
MouseQuerySeq_3876 36 10090 Mus : Mus musculus
MouseQuerySeq_38736151 10 6072 Metazoa : Eumetazoa
MouseQuerySeq_38731984 1 131567 root : cellular organisms
MouseQuerySeq_38714535 1 131567 root : cellular organisms
MouseQuerySeq_38718132 2 2759 cellular organisms : Eukaryota
MouseQuerySeq_38700001 2 2759 cellular organisms : Eukaryota
MouseQuerySeq_38728030 13 7711 Deuterostomia : Chordata
MouseQuerySeq_38714027 3 1452644 Eukaryota : Unikonta
MouseQuerySeq_38723446 1 131567 root : cellular organisms
MouseQuerySeq_38704436 9 33208 Metazoa/Choanoflagellida : Metazoa
MouseQuerySeq_38700535 16 7776 Vertebrata : Gnathostomata
MouseQuerySeq_38722609 1 131567 root : cellular organisms
MouseQuerySeq_38709989 19 8287 Euteleostomi : Sarcopterygii
MouseQuerySeq_38740706 2 2759 cellular organisms : Eukaryota
MouseQuerySeq_38723011 2 2759 cellular organisms : Eukaryota
MouseQuerySeq_38740230 10 6072 Metazoa : Eumetazoa
```

Each "`X : Y`" marks a period labelled by a taxonomy identifier of "Y" node to which the origin of a gene has been traced to. That is, the gene traced to phylostrata 10 labelled as 6072 has its origin traced to (Metazoa : Eumetazoa) period.

5.2.4 Known problems

None reported.

5.2.5 Future work

Upon request.

5.3 MapAssociate

Program associates gene related terms to a given phylostratigraphy map.

5.3.1 Program options

In order to see program options type:

```
perl ./scripts/MapAssociate.pl -h
```

Expected output:

```
********************************************************************************
                                  MapAssociate.pl
                                        by
                                   Robert Bakaric

CONTACT:
 Code written and maintained by Robert Bakaric,
 email: rbakaric@irb.hr , bakaric@evolbio.mpg.de

ACKNOWLEDGEMENT:


LICENSE:
 The program is distributed under the GNU General Public License. You should have
 received a copy of the licence together  with this software. If not, see
 http://www.gnu.org/licenses/
********************************************************************************


Usage: ./program [options]

-m Phylostratigraphy map
-a Association file (<Gene_identifier Association_term>)
-h this message
```

5.3.2 Input files

Program requires two input files. The one being the output of PhyloStrat (Sec. 4.3) or QPhyloStrat (Sec. 4.5) and the other being a tab separated association file. The association file has two columns, first contains a list of gene identifiers and the second one the corresponding association term (identifier):

`./examples/data/MouseSampleAssociate.tsv`

```
MouseQuerySeq_38709112  a
MouseQuerySeq_38713615  a
MouseQuerySeq_38713615  t
MouseQuerySeq_38718975  a
MouseQuerySeq_38718975  b
MouseQuerySeq_38718975  h
MouseQuerySeq_38718975  r
MouseQuerySeq_38704905  f
MouseQuerySeq_38727057  t
MouseQuerySeq_38709533  t
MouseQuerySeq_38709533  h
MouseQuerySeq_38744814  t
MouseQuerySeq_38744814  h
MouseQuerySeq_38727572  t
MouseQuerySeq_38727572  r
MouseQuerySeq_38705387  t
MouseQuerySeq_38705387  r
MouseQuerySeq_38705387  b
MouseQuerySeq_38735787  f
MouseQuerySeq_38718587  f
MouseQuerySeq_3876      f
MouseQuerySeq_38736151  a
MouseQuerySeq_38736151  r
MouseQuerySeq_38731984  a
MouseQuerySeq_38731984  b
```

```
MouseQuerySeq_38714535  a
MouseQuerySeq_38714535  h
MouseQuerySeq_38718132  b
MouseQuerySeq_38718132  f
MouseQuerySeq_38700001  b
MouseQuerySeq_38728030  h
MouseQuerySeq_38728030  f
MouseQuerySeq_38714027  h
MouseQuerySeq_38714027  f
MouseQuerySeq_38723446  h
MouseQuerySeq_38704436  h
MouseQuerySeq_38700535  r
MouseQuerySeq_38722609  r
MouseQuerySeq_38722609  f
MouseQuerySeq_38709989  r
MouseQuerySeq_38740706  h
MouseQuerySeq_38740706  f
MouseQuerySeq_38723011  h
MouseQuerySeq_38740230  h
```

5.3.3 Usage Example

By executing:

```
perl ./scripts/MapAssociate.pl -a ./examples/data/MouseSampleAssociate.tsv
                               -m ./examples/data/MouseSample.map
```

The expected output should look like:

```
<Species taxomony Id>
10090
<Phylogeny used>
0 1
1 131567
2 2759
3 1452644
4 1452646
5 33154
6 1452651
7 1452652
8 1452653
9 33208
10 6072
11 33213
12 33511
13 7711
14 1452661
15 7742
16 7776
17 117570
18 117571
19 8287
20 1338369
21 32523
22 32524
23 40674
24 32525
25 9347
26 1437010
27 314146
28 314147
29 9989
30 33553
31 337687
32 10066
33 39107
34 10088
```

```
35 862507
36 10090
<Phylostratigraphy map>
MouseQuerySeq_38744814 t,h 1 131567
MouseQuerySeq_38713615 a,t 9 33208
MouseQuerySeq_38723446 h 1 131567
MouseQuerySeq_38714027 h,f 3 1452644
MouseQuerySeq_38740230 h 10 6072
MouseQuerySeq_38709989 r 19 8287
MouseQuerySeq_38705387 t,r,b 9 33208
MouseQuerySeq_38728030 h,f 13 7711
MouseQuerySeq_38709112 a 2 2759
MouseQuerySeq_38727572 t,r 1 131567
MouseQuerySeq_38700535 r 16 7776
MouseQuerySeq_38731984 a,b 1 131567
MouseQuerySeq_38722609 r,f 1 131567
MouseQuerySeq_38700001 b 2 2759
MouseQuerySeq_38709533 t,h 16 7776
MouseQuerySeq_38736151 a,r 10 6072
MouseQuerySeq_38740706 h,f 2 2759
MouseQuerySeq_38735787 f 33 39107
MouseQuerySeq_38723011 h 2 2759
MouseQuerySeq_38714535 a,h 1 131567
MouseQuerySeq_38718975 a,b,h,r 1 131567
MouseQuerySeq_38727057 t 2 2759
MouseQuerySeq_3876 f 36 10090
MouseQuerySeq_38704436 h 9 33208
MouseQuerySeq_38718587 f 2 2759
MouseQuerySeq_38704905 f 9 33208
MouseQuerySeq_38718132 b,f 2 2759
```

5.3.4 Known problems

None reported.

5.3.5 Future work

Upon request.

5.4 MapSummary

The program computes phylostratigraphy map summary information.

5.4.1 Program options

In order to see program options type:

```
perl ./scripts/MapSummary.pl -h
```

Expected output:

```
********************************************************************************
                                    MapSummary.pl
                                         by
                                    Robert Bakaric

CONTACT:
 Code written and maintained by Robert Bakaric,
 email: rbakaric@irb.hr , bakaric@evolbio.mpg.de

ACKNOWLEDGEMENT:


LICENSE:
 The program is distributed under the GNU General Public License. You should have
 received a copy of the licence together  with this software. If not, see
 http://www.gnu.org/licenses/
********************************************************************************
```

```
Usage: ./program [options]

-m Map file [query <tab> PhyloId <tab> TaxId]
-a Computes summary statistics for associated labels
(works iff associated labels are mapped)[def: F]
-h This message
```

5.4.2 Input Files

The output of PhyloStrat (Sec. 4.3) or QPhyloStrat (Sec. 4.5) is used as an input for `MapSummary`. Moreover, if genes in the initial phylostratigraphy map have been associated with corresponding additional descriptive information, option **-a T** can be used to count it in. An example of an input file can be found in section 5.8.3 (or 4.5.3)

5.4.3 Usage Example

By executing:

```
perl ./scripts/MapSummary.pl -m ./examples/data/MouseSample.map
```

The expected result should look like:

```
1 131567              7  25.93%
2 2759                7  25.93%
3 1452644             1   3.70%
9 33208               4  14.81%
10 6072               2   7.41%
13 7711               1   3.70%
16 7776               2   7.41%
19 8287               1   3.70%
33 39107              1   3.70%
36 10090              1   3.70%
```

From left to right columns are defined as:

1. PhyloStrata Identifier
2. Taxonomy Identifier
3. Number of genes within phylostrata defined in column 1
4. Fraction of genes within phylostrata defined in column 1 (%)

By executing the program on a map with names included:

```
perl ./scripts/MapSummary.pl -m ./examples/data/MouseSampleAddNames.ma
```

The expected output should look like:

```
1 131567 root : cellular organisms                  7 25.93%
2 2759 cellular organisms : Eukaryota               7 25.93%
3 1452644 Eukaryota : Unikonta                      1 3.70%
9 33208 Metazoa/Choanoflagellida : Metazoa          4 14.81%
10 6072 Metazoa : Eumetazoa                         2 7.41%
13 7711 Deuterostomia : Chordata                    1 3.70%
16 7776 Vertebrata : Gnathostomata                  2 7.41%
19 8287 Euteleostomi : Sarcopterygii                1 3.70%
33 39107 Muridae : Murinae                          1 3.70%
36 10090 Mus : Mus musculus                         1 3.70%
```

The third tab delimited column in this case represents the transition period corresponding phylostrata labelled with taxonomy identifier in the second column.

Moreover, if gene identifiers in phylostratigraphy mas have been previously associated with "gene-associated" information setting the option "-a" to TRUE (T) the obtained output should look like:

```
perl ./scripts/MapSummary.pl -a T   \
                        -m ./examples/data/MouseSampleAddNamesAddTerm.map
```

```
1 131567 root : cellular organisms                  (a) 3 50.00% (b) 2 40.00% (f) 1 11.11%
                                                    (h) 4 36.36% (r) 3 42.86% (t) 2 33.33%
2 2759 cellular organisms : Eukaryota               (a) 1 16.67% (b) 2 40.00% (f) 3 33.33% (h) 2 18.18%
                                                    (t) 1 16.67%
3 1452644 Eukaryota : Unikonta                      (f) 1 11.11% (h) 1 9.09%
9 33208 Metazoa/Choanoflagellida : Metazoa          (a) 1 16.67% (b) 1 20.00% (f) 1 11.11% (h) 1 9.09%
                                                    (r) 1 14.29% (t) 2 33.33%
10 6072 Metazoa : Eumetazoa                         (a) 1 16.67% (h) 1 9.09% (r) 1 14.29%
13 7711 Deuterostomia : Chordata                    (f) 1 11.11% (h) 1 9.09%
16 7776 Vertebrata : Gnathostomata                  (h) 1 9.09% (r) 1 14.29% (t) 1 16.67%
19 8287 Euteleostomi : Sarcopterygii                (r) 1 14.29%
33 39107 Muridae : Murinae                          (f) 1 11.11%
36 10090 Mus : Mus musculus                         (f) 1 11.11%
```

"(a) 3 50.00%" represents the summary statistics for associated term "a" in phylostrata 1.The first value "(a)" is the term label. The second number reflects terms occurrence count, while the third value is its corresponding fraction (%) with respect to the total number of genes associated to that particular association across the entire map.

5.5 MapExtract

The tool allows users to further parse the data obtained from a phylostratigraphy map or its summary table. It allows to extract any combination of phylostrata clusters and their associated data.

5.5.1 Program options

In order to see program options type:

```
perl ./scripts/MapExtract.pl -h
```

Expected output:

```
*********************************************************************************
                                    MapExtract.pl
                                         by
                                    Robert Bakaric

CONTACT:
 Code written and maintained by Robert Bakaric,
 email: rbakaric@irb.hr , bakaric@evolbio.mpg.de

ACKNOWLEDGEMENT:


LICENSE:
 The program is distributed under the GNU General Public License. You should have
```

```
 received a copy of the licence together  with this software. If not, see
 http://www.gnu.org/licenses/
*******************************************************************************


Usage: ./program [options]

-m Phylostratigraphy map
-s  Phylostratigraphy map, summary information table
-p Phylostra identifier
-t Associated data (functional traits ...)
-h this message
```

5.5.2 Input Files

Input files for this program are output files from either `PhyloStrat` (-m) or `MapSummary` (-s) programs. Moreover, once the input files are provided user can specify the phylostrata in which he/she is interested and the association term marked for extraction.

5.5.3 Usage Example

By executing:

```
perl MapExtract.pl -t b -s ../examples/data/MouseSummaryTerms
```

The obtained result should look like:

```
1 131567 b 2 40.00
2 2759 b 2 40.00
9 33208 b 1 20.00
```

In this example the program extracted each phylostrata containing an associated term b together with its corresponding information:

1. Phylostrat identifier
2. Taxonomy identifier
3. Associated term
4. Occurrence count of the associated term
5. Percentage

On the other hand if raw map is used as an input:

```
perl MapExtract.pl -p 2 -t b \
                   -m ../examples/data/MouseSampleAddNamesAddTerm.map
```

The expected result should look like:

```
MouseQuerySeq_38718132 b,f 2 2759 cellular organisms : Eukaryota
MouseQuerySeq_38700001 b 2 2759 cellular organisms : Eukaryota
```

Extracting all specified terms (option -t) from selected phylostrata (option -p). From left to right columns are labelled as:

1. Gene identifier
2. Associated term
3. Phylostrata identifier

4. Taxonomy identifier
5. Phylostrata label

5.6 MapLogOdds

The tool calculates Log-Odds ratio and its significance for each gene associated term in the obtained and associated phylostratigraphy map computed using either PhyloStrat (Sec. 4.3) or QPhyloStrat (Sec. 4.5) software.

5.6.1 Program options

In order to see program options type:

```
perl ./scripts/MapLogOdd.pl -h
```

Expected output:

```
*******************************************************************************
                                    MapLogOdd.pl
                                         by
                                   Robert Bakaric

CONTACT:
 Code written and maintained by Robert Bakaric,
 email: rbakaric@irb.hr , bakaric@evolbio.mpg.de

ACKNOWLEDGEMENT:


LICENSE:
 The program is distributed under the GNU General Public License. You should have
 received a copy of the licence together  with this software. If not, see
 http://www.gnu.org/licenses/
*******************************************************************************


Usage: ./program [options]

-s Map Summary file [MapSummary -a T]
-h This message
```

5.6.2 Input Files

`MapLogOdd` accepts only the summary output file from `MapSummary` program with `-a` option set to **T** (see section 5.4 for further information).

5.6.3 Usage Example

By executing:

```
perl ./scripts/MapLogOdd.pl -s ./examples/data/MouseSummaryTerms
```

The expected result should look like:

```
PhyloId TaxId Desc AssoTerm Quant Hit Sample Total LogOdds P-val Bonferroni
FDR
1 131567 root : cellular organisms t 2 6 15 44 -0.0 1.0000e+00 1 1
1 131567 root : cellular organisms f 1 9 15 44 -1.66 2.0989e-01 1 1
1 131567 root : cellular organisms r 3 7 15 44 0.45 8.9587e-01 1 1
1 131567 root : cellular organisms h 4 11 15 44 0.13 1.0000e+00 1 1
1 131567 root : cellular organisms b 2 5 15 44 0.29 1.0000e+00 1 1
1 131567 root : cellular organisms a 3 6 15 44 0.77 6.5415e-01 1 1
2 2759 cellular organisms : Eukaryota h 2 11 9 44 -0.1 1.0000e+00 1 1
```

```
2 2759 cellular organisms : Eukaryota f 3 9 9 44 0.88 5.2027e-01 1 1
2 2759 cellular organisms : Eukaryota r 0 7 9 44 -23.71 3.5096e-01 1 1
2 2759 cellular organisms : Eukaryota b 2 5 9 44 1.11 5.3432e-01 1 1
2 2759 cellular organisms : Eukaryota a 1 6 9 44 -0.2 1.0000e+00 1 1
2 2759 cellular organisms : Eukaryota t 1 6 9 44 -0.2 1.0000e+00 1 1
3 1452644 Eukaryota : Unikonta t 0 6 2 44 -23.71 1.0000e+00 1 1
3 1452644 Eukaryota : Unikonta a 0 6 2 44 -23.71 1.0000e+00 1 1
3 1452644 Eukaryota : Unikonta r 0 7 2 44 -23.71 1.0000e+00 1 1
3 1452644 Eukaryota : Unikonta f 1 9 2 44 1.45 7.4207e-01 1 1
3 1452644 Eukaryota : Unikonta h 1 11 2 44 1.16 8.8372e-01 1 1
3 1452644 Eukaryota : Unikonta b 0 5 2 44 -23.71 1.0000e+00 1 1
9 33208 Metazoa/Choanoflagellida : Metazoa a 1 6 7 44 0.06 1.0000e+00 1 1
9 33208 Metazoa/Choanoflagellida : Metazoa t 2 6 7 44 1.19 4.7683e-01 1 1
9 33208 Metazoa/Choanoflagellida : Metazoa f 1 9 7 44 -0.4 1.0000e+00 1 1
9 33208 Metazoa/Choanoflagellida : Metazoa b 1 5 7 44 0.32 1.0000e+00 1 1
9 33208 Metazoa/Choanoflagellida : Metazoa h 1 11 7 44 -0.7 8.5882e-01 1 1
9 33208 Metazoa/Choanoflagellida : Metazoa r 1 7 7 44 -0.1 1.0000e+00 1 1
10 6072 Metazoa : Eumetazoa r 1 7 3 44 1.07 8.2664e-01 1 1
10 6072 Metazoa : Eumetazoa h 1 11 3 44 0.44 1.0000e+00 1 1
10 6072 Metazoa : Eumetazoa b 0 5 3 44 -23.71 1.0000e+00 1 1
10 6072 Metazoa : Eumetazoa t 0 6 3 44 -23.71 1.0000e+00 1 1
10 6072 Metazoa : Eumetazoa a 1 6 3 44 1.28 7.2606e-01 1 1
10 6072 Metazoa : Eumetazoa f 0 9 3 44 -23.71 9.8837e-01 1 1
13 7711 Deuterostomia : Chordata h 1 11 2 44 1.16 8.8372e-01 1 1
13 7711 Deuterostomia : Chordata b 0 5 2 44 -23.71 1.0000e+00 1 1
13 7711 Deuterostomia : Chordata a 0 6 2 44 -23.71 1.0000e+00 1 1
13 7711 Deuterostomia : Chordata r 0 7 2 44 -23.71 1.0000e+00 1 1
13 7711 Deuterostomia : Chordata t 0 6 2 44 -23.71 1.0000e+00 1 1
13 7711 Deuterostomia : Chordata f 1 9 2 44 1.45 7.4207e-01 1 1
16 7776 Vertebrata : Gnathostomata r 1 7 3 44 1.07 8.2664e-01 1 1
16 7776 Vertebrata : Gnathostomata b 0 5 3 44 -23.71 1.0000e+00 1 1
16 7776 Vertebrata : Gnathostomata h 1 11 3 44 0.44 1.0000e+00 1 1
16 7776 Vertebrata : Gnathostomata f 0 9 3 44 -23.71 9.8837e-01 1 1
16 7776 Vertebrata : Gnathostomata a 0 6 3 44 -23.71 1.0000e+00 1 1
16 7776 Vertebrata : Gnathostomata t 1 6 3 44 1.28 7.2606e-01 1 1
19 8287 Euteleostomi : Sarcopterygii a 0 6 1 44 -23.71 1.0000e+00 1 1
19 8287 Euteleostomi : Sarcopterygii h 0 11 1 44 -23.71 1.0000e+00 1 1
19 8287 Euteleostomi : Sarcopterygii f 0 9 1 44 -23.71 1.0000e+00 1 1
19 8287 Euteleostomi : Sarcopterygii r 1 7 1 44 6.42 3.1818e-01 1 1
19 8287 Euteleostomi : Sarcopterygii b 0 5 1 44 -23.71 1.0000e+00 1 1
19 8287 Euteleostomi : Sarcopterygii t 0 6 1 44 -23.71 1.0000e+00 1 1
33 39107 Muridae : Murinae t 0 6 1 44 -23.71 1.0000e+00 1 1
33 39107 Muridae : Murinae r 0 7 1 44 -23.71 1.0000e+00 1 1
33 39107 Muridae : Murinae h 0 11 1 44 -23.71 1.0000e+00 1 1
33 39107 Muridae : Murinae a 0 6 1 44 -23.71 1.0000e+00 1 1
33 39107 Muridae : Murinae b 0 5 1 44 -23.71 1.0000e+00 1 1
33 39107 Muridae : Murinae f 1 9 1 44 6.08 4.0909e-01 1 1
36 10090 Mus : Mus musculus f 1 9 1 44 6.08 4.0909e-01 1 1
36 10090 Mus : Mus musculus b 0 5 1 44 -23.71 1.0000e+00 1 1
36 10090 Mus : Mus musculus h 0 11 1 44 -23.71 1.0000e+00 1 1
36 10090 Mus : Mus musculus r 0 7 1 44 -23.71 1.0000e+00 1 1
36 10090 Mus : Mus musculus t 0 6 1 44 -23.71 1.0000e+00 1 1
36 10090 Mus : Mus musculus a 0 6 1 44 -23.71 1.0000e+00 1 1.0000e+00
```

From left to right columns are defined as:

1. PhyloStrata Identifier
2. Taxonomy Identifier
3. Associated phylostrata Name
4. Associated Term
5. Number of genes associated to the term in column 4 within phylostrata defined in column 1
6. Log-Odds ratio

7. p-value
8. Bonferroni corrected p-value
9. FDR corrected p-value

5.6.4 Known problems

None reported

5.6.5 Future work

Upon request.

5.7 TreeIllustrator

Software tool for drawing tree-like data structures within a terminal session. The program is designed as a module for automatically parsing `MakeTree` output.

5.7.1 Program options

In order to see program options type:

```
perl ./scripts/TreeIlustrator.pl -n Tree.newick
```

Expected output:

```
*******************************************************************************
                                  TreeIlustrator.pl
                                         by
                                   Robert Bakaric

CONTACT:
 Code written and maintained by Robert Bakaric,
 email: rbakaric@irb.hr , bakaric@evolbio.mpg.de

ACKNOWLEDGEMENT:


LICENSE:
 The program is distributed under the GNU General Public License. You should have
 received a copy of the licence together  with this software. If not, see
 http://www.gnu.org/licenses/
*******************************************************************************


Usage: ./program [options]

-h Prints this message
-n Newick input file ( or stdin )
```

5.7.2 Input Files

A single line newick tree representation. Example:

```
(((Trichoplax_adhaerens[10228]:1.0)Trichoplax[10227]:1.0)Placozoa[10226]:1.0,
((Chordata[7711]:1.0,Ambulacraria[1452662]:1.0)Deuterostomia[33511]:1.0,
(Lophotrochozoa[1206795]:1.0,Ecdysozoa[1206794]:1.0)Protostomia[33317]:1.0,
(Trematoda[6178]:1.0)Platyhelminthes[6157]:1.0)Bilateria[33213]:1.0,
((Hexacorallia[6102]:1.0)Anthozoa[6101]:1.0,(Semaeostomeae[6143]:1.0)Scyphozoa[6142]:1.0,
(Hydroida[37516]:1.0)Hydrozoa[6074]:1.0)Cnidaria[6073]:1.0)Eumetazoa[6072]:1.0
```

5.7.3 Usage Example

By executing:

```
perl ./scripts/Nr2Ph.pl -n ./examples/data/Newick
```

The expected result is :

```
cellular_organisms[131567]
  ├─Bacteria[2]
  │ └─GroupA[1648577]
  │   └─GroupB[1648578]
  │     └─GroupC[1648579]
  │       └─GroupD[1648580]
  │         └─GroupE[1648581]
  └─Eukaryota[2759]
    ├─Unikonta[1648521]
    │ ├─Amoebozoa[554915]
    │ │ ├─Discosea[555280]
    │ │ │ └─Longamoebia[1485168]
    │ │ │   └─Centramoebida[555407]
    │ │ │     └─Acanthamoebidae[33677]
    │ │ │       └─Acanthamoeba[5754]
    │ │ │         └─Acanthamoeba_castellanii[5755]
    │ │ │           └─Acanthamoeba_castellanii_str._Neff[1257118]
    │ │ ├─Archamoebae[555406]
    │ │ │ └─Entamoebidae[33084]
    │ │ │   └─Entamoeba[5758]
    │ │ │     └─Entamoeba_histolytica[5759]
    │ │ │       └─Entamoeba_histolytica_HM-1:IMSS[294381]
    │ │ └─Mycetozoa[142796]
    │ │   └─Dictyosteliida[33083]
    │ │     ├─Polysphondylium[13641]
    │ │     │ └─Polysphondylium_pallidum[13642]
    │ │     │   └─Polysphondylium_pallidum_PN500[670386]
    │ │     └─Dictyostelium[5782]
    │ │       └─Dictyostelium_discoideum[44689]
    │ └─Apusozoa/Opisthokonta[1648523]
    │   ├─Apusozoa[554296]
    │   │ └─Apusomonadidae[172820]
    │   │   └─Thecamonas[877559]
    │   │     └─Thecamonas_trahens[529818]
    │   │       └─Thecamonas_trahens_ATCC_50062[461836]
    │   └─Opisthokonta[33154]
    │     ├─Holozoa[1648563]
    │     │ ├─Filasporea[1648566]
...
```

The tool is designed for quick illustration of changes introduced into a given phylogeny tree using `MakeTree` software.

5.7.4 Known problems

None reported

5.7.5 Future work

Upon request.

5.8 Nr2Ph

`Nr2Ph` is a simple nr database parser which is designed to split the nr database into a set of individual taxonomy based flat fasta files.

5.8.1 Program options

In order to see program options type:

```
perl ./scripts/Nr2Ph.pl -h
```

Expected output:

```
********************************************************************************
                                    Nr2Ph.pl
                                       by
                                 Robert Bakaric

CONTACT:
 Code written and maintained by Robert Bakaric,
 email: rbakaric@irb.hr , bakaric@evolbio.mpg.de

ACKNOWLEDGEMENT:


LICENSE:
 The program is distributed under the GNU General Public License. You should have
 received a copy of the licence together  with this software. If not, see
 http://www.gnu.org/licenses/
********************************************************************************


Usage: ./program [options]

-n nr fasta file
-t gene id - taxonomy id association table [GeneId <tab> Taxid]
-h this message
```

5.8.2 Input Files

Program requires two information sources the ncbi's "genome id"-"taxonomy id" relation table and the fasta formatted nr database.

5.8.3 Usage Example

By executing:

```
perl ./scripts/Nr2Ph.pl -n ./examples/data/nr_exe  \
                         -t ./examples/data/gitax_exe
```

As a result is a set of fasta files each labelled according to its corresponding taxonomy identifier all located within a `./phdb` directory

5.8.4 Known problems

None reported

5.8.5 Future work

Upon request.

5.9 Ph2Nr

Ph2Nr is a software tool designed to convert a given phylo-database into a classic nr-like data file, however, preserving the "pgi" indexing information.

5.9.1 Program options

In order to see program options type:

```
perl ./scripts/Ph2Nr.pl -h
```

Expected output:

```
********************************************************************************
                                      Ph2Nr.pl
                                         by
                                   Robert Bakaric

CONTACT:
 Code written and maintained by Robert Bakaric,
 email: rbakaric@irb.hr , bakaric@evolbio.mpg.de

ACKNOWLEDGEMENT:


LICENSE:
 The program is distributed under the GNU General Public License. You should have
 received a copy of the licence together  with this software. If not, see
 http://www.gnu.org/licenses/
********************************************************************************


Usage: ./program [options]

-i phdb - phylostratigraphy database (a collection of fasta files)
-h this message
```

5.9.2 Input Files

Program operated on a directory containing a set of phylo-db, preformatted fasta files.

5.9.3 Usage Example

By executing:

```
perl ./scripts/Ph2Nrh.pl -d ./examples/data/phdb
```

By redirecting stdout to a file (`perl ./scripts/Ph2Nrh.pl -d _examples/data/phdb > nr`) phdb is reformatted and as such can be used as an input for BLAST.

5.9.4 Known problems

None reported

5.9.5 Future work

Upon request.

5.10 SplicVar

Software for extracting splicing variants from fasta sequence files based on their size. Currently the program supports two modes (L and S), extracting either the longest splicing variant (L) or the shortest one (S).

5.10.1 Program options

In order to see program options type:

```
perl ./scripts/SplicVar.pl -h
```

Expected output:

```
********************************************************************************
                                    SplicVar.pl
                                        by
                                   Robert Bakaric

CONTACT:
 Code written and maintained by Robert Bakaric,
 email: rbakaric@irb.hr , bakaric@evolbio.mpg.de

ACKNOWLEDGEMENT:


LICENSE:
 The program is distributed under the GNU General Public License. You should have
 received a copy of the licence together  with this software. If not, see
 http://www.gnu.org/licenses/
********************************************************************************


Usage: ./program [options]

-f Fasta file (gene and transcript tags required)
-l longest (L), shortest (S)
-h this message
```

5.10.2 Input Files

As an input program takes a simple fasta file with `gene:` token in its header. Example:

```
>ENSP00000381386 pep:known chromosome:GRCh37:22:24313554:24316773:-1 gene:ENSG00000099977
 transcript:ENST00000398344 gene_biotype:protein_coding transcript_biotype:protein_coding
MPFLELDTNLPANRVPAGLEKRLCAAAASILGKPADRVNVTVRPGLAMALSGSTEPCAQLSISSIGVVGTAEDNRSHSA
HFFEFLTKELALGQDRILIRFFPLESWQIGKIGTVMTFL
>ENSP00000215773 pep:known chromosome:GRCh37:22:24313554:24322019:-1 gene:ENSG00000099977
 transcript:ENST00000350608 gene_biotype:protein_coding transcript_biotype:protein_coding
MPFLELDTNLPANRVPAGLEKRLCAAAASILGKPADRVNVTVRPGLAMALSGSTEPCAQLSISSIGVVGTAEDNRSHSA
HFFEFLTKELALGQDRILIRFFPLESWQIGKIGTVMTFL
...
```

5.10.3 Usage Example

By executing:

```
perl ./scripts/SplicVar.pl -f Fasta.fa -l L
```

Program extracts the longest splicing variant and prints it to standard output. By redirecting the output to a file user can save the modified set of fasta records into a new file.

5.10.4 Known problems

None reported

5.10.5 Future work

Upon request.

5.11 NCBIDataParser

Software designed for quick cleaning of the essential input files used throughout the entire toolkit. This usually is a starting point for any new toolkit user (see Sec. 6)

5.11.1 Program options

In order to see program options type:

```
perl ./scripts/NCBIDataParser.pl -h
```

Expected output:

```
*******************************************************************************
                                  NCBIDataParser.pl
                                         by
                                   Robert Bakaric

CONTACT:
 Code written and maintained by Robert Bakaric,
 email: rbakaric@irb.hr , bakaric@evolbio.mpg.de

ACKNOWLEDGEMENT:


LICENSE:
 The program is distributed under the GNU General Public License. You should have
 received a copy of the licence together  with this software. If not, see
 http://www.gnu.org/licenses/
*******************************************************************************


Usage: ./program [options]

-m names file (*.dmp)
-n nodes file (*.dmp)
-d nr database (multi fasta)
-h This message
```

5.11.2 Input Files

As an input program takes three different files `names.dmp`, `nodes.dmp` and multi fasta file `nr` examples of which are shown below:

`names.dmp` :

```
1 | all | | synonym |
1 | root | | scientific name |
2 | Bacteria | Bacteria <prokaryote>| scientific name |
2 | Monera | Monera <Bacteria>| in-part |
2 | Procaryotae | Procaryotae <Bacteria>| in-part |
2 | Prokaryota | Prokaryota <Bacteria>| in-part |
2 | Prokaryotae | Prokaryotae <Bacteria>| in-part |
2 | bacteria | bacteria <blast2>| blast name |
2 | eubacteria | | genbank common name |
2 | not Bacteria Haeckel 1894 | | synonym |
...
```

`nodes.dmp` :

```
1 | 1 | no rank | | 8 | 0 | 1 | 0 | 0 |0 | 0 | 0 | |
2 | 131567 | superkingdom | | 0 | 0 | 11 | 0 | 0| 0 | 0 | 0 | |
6 | 335928 | genus | | 0 | 1 | 11 | 1 | 0 |1 | 0 | 0 | |
7 | 6 | species | AC | 0 | 1 | 11 | 1 | 0 |1 | 1 | 0 | |
9 | 32199 | species | BA | 0 | 1 | 11 | 1 | 0 |1 | 1 | 0 | |
...
```

`nr :`

```
>gi|99600005|ti|431895|pi|0| MONBRDRAFT_00209T0 | MONBRDRAFT_00209 | Mon ...
EEVVLRAGSEGLGMSITGGTDRPLVAGDNSIFITDIVPHGAANRTGRLTPGDSIVSINGVSLENKTHGE
VVALLRQGGALNESSASIMMTHTETISLHRQHGRGLGFTIAGGQGSPHIAGDDGIFISKIIPDSAAKED
GRLAVGDRVLSVQGESCEKITHERAVEMLRNPASPIVLVVEHNAFHKATAELSRSLGLKKPAAVPITRT
GTLKSFGQVNASLQLEASPFDAADKMSDLRTVTLYKGKAGFGFSLLGPAKAGPAEEGEPVGIFISRILP
EGAAIESGQVFEGDQILSMNGQDLALASYRQAANLVKHITDGVMTLNLTANPGMYDLYKQRMAAVQANT
IETSKELNQIPQDSLCLRALFDYDPAQDSSVASNVTLHDVFVLIKDIHADWWEVQDVRTNVKGLIPSRA
RYDSHAMFGVGLTAREKKEEKARSILLTRLSFGRRKSSSNVSNGFSGRRRGVYEAVQLHQATTQEPRPL
LVLGPSKDHITDKLIDEYPTVFGSCVPHTTRDPRPGEREGEDYHFVSMAAMTKAIEDGEFIEAGQYRAH
LYGTSIASVQQVVQQQLSCILDVSVSAIPKLHAHKLFPIVIYLKPDSVSSLRQQNPHFSEETAREVFAL
SQQVERDYRHLFTKVISNLDLDSTYRRVLDTLSMQSREPFWAP
>gi|99600263|ti|431895|pi|0| MONBRDRAFT_03794T0 | MONBRDRAFT_03794 | Mon ...
KGEQCCICLSVFQDNDRILVLPCSHGFHHQCVGQWLRQQRRCPLCNRDPFSTD
>gi|99600294|ti|431895|pi|0| MONBRDRAFT_04813T0 | MONBRDRAFT_04813 | Mon ...
MGEAPPPPPPRWASRGMGVVLVAVVLAALVGWTSEHPQVQLALMRFFRDVSRAKAHARLYGRDNALAVM
LGQEELAASTWSEVPDTSFNEEDLQWLNGAESRPVYLALAGRVVYDVTAGRHKYGPGGSYHKLAGRDASR
PLALGCLTESCLTGSLQVAAAVEASLAADAFAEKTAARLRHEYKQQRQQRGGTGALVAIIQETQRDAVK
LYRRQRLDDVVTIIEALFLLLDEQMSLGGLDSAPLPLEMDAGRLILAWWAVCICHEQDREADQAIWQTN
FKVRRCS
>gi|99600405|ti|431895|pi|0| MONBRDRAFT_05431T0 | MONBRDRAFT_05431 | Mon ...
MAALTAELPTAVHDPLETACWESTRFVDDVSPHLRCPICLNVCLNPVACSTCDQVFGEHCWYQALAAHG
CCPTCRQKEHPFASPSRLARSFIGDYRVRCRHASEGCTEVLPLQEMLKHQAVCGHLQRPCPHCQVPVRA
SDAQQHEDECALRLVMCPHVGCGIQVPMHALAEHRGRCIHPRPSAGDQGTRPEAQHCCQFCHEHCSLAT
AMDTWCRPFLEHRLALAQLYLRRLQTAGEQQAALIKRSTTASRDLSNALRDATGALGYQTSQLRQQQET
IRSVGATRDHAVKLLRESETMRKTMSDVLRQKNICIQQLKAENELLRRKMDTQDVLLKVLMESPTRSGR
RVGSANGVQADSAAEAMELDEASGDKPKTD
...
```

As a result clean, reformatted file copies are created in the same directory all having the extension `*.fmt`.

Note that each file can be parsed separately by explicitly invoking the program on it as shown below.

5.11.3 Usage Example

By executing:

```
perl ./scripts/NCBIDataParser.pl -m ./examples/data/names.dmp \
                                 -n ./examples/data/nodes.dmp \
                                 -d ./examples/data/nr-Test/TestDb
```

Three resulting output files are created:

`names.dmp.fmt :`

```
1       all     ~       synonym
1       root    ~       scientific_name
2       Bacteria        Bacteria_<prokaryote>   scientific_name
2       Monera  Monera_<Bacteria>       in-part
2       Procaryotae     Procaryotae_<Bacteria>  in-part
2       Prokaryota      Prokaryota_<Bacteria>   in-part
2       Prokaryotae     Prokaryotae_<Bacteria>  in-part
2       bacteria        bacteria_<blast2>       blast_name
2       eubacteria      ~       genbank_common_name
2       not_Bacteria_Haeckel_1894       ~       synonym
...
```

nodes.dmp.fmt :

```
2       131567
6       335928
7       6
9       32199
...
```

nr.fmt :

```
>gi|99600005|ti|431895|pi|0| MONBRDRAFT_00209T0 | MONBRDRAFT_00209 | Monosiga brevicollis ..
EEVVLRAGSEGLGMSITGGTDRPLVAGDNSIFITDIVPHGAANRTGRLTPGDSIVSINGVSLENKTHGE
VVALLRQGGALNESSASIMMTHTETISLHRQHGRGLGFTIAGGQGSPHIAGDDGIFISKIIPDSAAKED
GRLAVGDRVLSVQGESCEKITHERAVEMLRNPASPIVLVVEHNAFHKATAELSRSLGLKKPAAVPITRT
GTLKSFGQVNASLQLEASPFDAADKMSDLRTVTLYKGKAGFGFSLLGPAKAGPAEEGEPVGIFISRILP
EGAAIESGQVFEGDQILSMNGQDLALASYRQAANLVKHITDGVMTLNLTANPGMYDLYKQRMAAVQANT
IETSKELNQIPQDSLCLRALFDYDPAQDSSVASNVTLHDVFVLIKDIHADWWEVQDVRTNVKGLIPSRA
RYDSHAMFGVGLTAREKKEEKARSILLTRLSFGRRKSSSNVSNGFSGRRRGVYEAVQLHQATTQEPRPL
LVLGPSKDHITDKLIDEYPTVFGSCVPHTTRDPRPGEREGEDYHFVSMAAMTKAIEDGEFIEAGQYRAH
LYGTSIASVQQVVQQQLSCILDVSVSAIPKLHAHKLFPIVIYLKPDSVSSLRQQNPHFSEETAREVFAL
SQQVERDYRHLFTKVISNLDLDSTYRRVLDTLSMQSREPFWAP
>gi|99600263|ti|431895|pi|0| MONBRDRAFT_03794T0 | MONBRDRAFT_03794 | Monosiga brevicollis ..
KGEQCCICLSVFQDNDRILVLPCSHGFHHQCVGQWLRQQRRCPLCNRDPFSTD
>gi|99600294|ti|431895|pi|0| MONBRDRAFT_04813T0 | MONBRDRAFT_04813 | Monosiga brevicollis ..
MGEAPPPPPPRWASRGMGVVLVAVVLAALVGWTSEHPQVQLALMRFFRDVSRAKAHARLYGRDNALAVM
LGQEELAASTWSEVPDTSFNEEDLQWLNGAESRPVYLALAGRVYDVTAGRHKYGPGGSYHKLAGRDASR
PLALGCLTESCLTGSLQVAAAVEASLAADAFAEKTAARLRHEYKQQRQQRGGTGALVAIIQETQRDAVK
LYRRQRLDDVVTIIEALFLLLDEQMSLGGLDSAPLPLEMDAGRLILAWWAVCICHEQDREADQAIWQTN
FKVRRCS
>gi|99600405|ti|431895|pi|0| MONBRDRAFT_05431T0 | MONBRDRAFT_05431 | Monosiga brevicollis ..
MAALTAELPTAVHDPLETACWESTRFVDDVSPHLRCPICLNVCLNPVACSTCDQVFGEHCWYQALAAHG
CCPTCRQKEHPFASPSRLARSFIGDYRVRCRHASEGCTEVLPLQEMLKHQAVCGHLQRPCPHCQVPVRA
SDAQQHEDECALRLVMCPHVGCGIQVPMHALAEHRGRCIHPRPSAGDQGTRPEAQHCCQFCHEHCSLAT
AMDTWCRPFLEHRLALAQLYLRRLQTAGEQQAALIKRSTTASRDLSNALRDATGALGYQTSQLRQQQET
IRSVGATRDHAVKLLRESETMRKTMSDVLRQKNICIQQLKAENELLRRKMDTQDVLLKVLMESPTRSGR
RVGSANGVQADSAAEAMELDEASGDKPKTD
...
```

Or just:

```
perl ./scripts/NCBIDataParser.pl -m ./examples/data/names.dmp
```

to get:

names.dmp.fmt :

```
1       all     ~       synonym
1       root    ~       scientific_name
2       Bacteria        Bacteria_<prokaryote>   scientific_name
2       Monera  Monera_<Bacteria>       in-part
2       Procaryotae     Procaryotae_<Bacteria>  in-part
2       Prokaryota      Prokaryota_<Bacteria>   in-part
2       Prokaryotae     Prokaryotae_<Bacteria>  in-part
2       bacteria        bacteria_<blast2>       blast_name
2       eubacteria      ~       genbank_common_name
2       not_Bacteria_Haeckel_1894       ~       synonym
...
```

The three files are now ready for further processing and information parsing.

5.11.4 Known problems

None reported

5.11.5 Future work

Upon request.

5.12 DbSync

Software designed for synchronizing the database content with phylogeny information.

5.12.1 Program options

In order to see program options type:

```
perl ./scripts/DbSync.pl -h
```

Expected output:

```
*******************************************************************************
                                    DbSync.pl
                                       by
                                  Robert Bakaric

CONTACT:
 Code written and maintained by Robert Bakaric,
 email: rbakaric@irb.hr , bakaric@evolbio.mpg.de

ACKNOWLEDGEMENT:


LICENSE:
 The program is distributed under the GNU General Public License. You should have
 received a copy of the licence together  with this software. If not, see
 http://www.gnu.org/licenses/
*******************************************************************************


Usage: ./program [options]

-n nodes file (*.new)
-d database (multi fasta file directory)
-h This message
```

5.12.2 Input Files

As an input program takes a formatted database directory containing genome flat-fasta-files with `*.ff` extension and a desired `nodes` file. The two should look like:

`./phdb/`

```
1000000.ff  1286976.ff  296543.ff  41413.ff   5518.ff    669874.ff  8083.ff
100027.ff   1287680.ff  296587.ff  4155.ff    554065.ff  670386.ff  8090.ff
100044.ff   1287681.ff  29730.ff   418459.ff  554155.ff  670580.ff  8128.ff
100047.ff   12957.ff    29760.ff   420593.ff  556484.ff  671987.ff  81824.ff
1001833.ff  1299270.ff  29883.ff   42068.ff   559292.ff  675120.ff  81972.ff
1001937.ff  1302862.ff  29889.ff   42254.ff   559297.ff  675824.ff  81985.ff
1001938.ff  13037.ff    30538.ff   423536.ff  559298.ff  67593.ff   82310.ff
10020.ff    1305764.ff  3055.ff    425011.ff  559305.ff  683840.ff  8296.ff
10090.ff    1314663.ff  30608.ff   426418.ff  559515.ff  683960.ff  83344.ff
10116.ff    1314666.ff  30611.ff   426428.ff  559561.ff  684364.ff  83485.ff
10141.ff    1314677.ff  3067.ff    428574.ff  559882.ff  686832.ff  8355.ff
...

info.caf
info.paf
```

`nodes`:

```
2       131567
6       335928
7       6
9       32199
10      135621
11      1707
13      203488
14      13
16      32011
17      16
...
```

5.12.3 Usage Example

By executing:

```
perl ./scripts/DbSync.pl -n ./examples/data/nodes  \
                         -d ./examples/data/nr-Test/TestDb-phdb
```

Given a parrent node, the program deletes child branches from `nodes` file that contain no genomes (specified within `phdb`). Moreover, genomes from `phdb` database assigned to parent nodes with leafs having associated genome files are removed also. Newly created file can then be used and a new adjacency list of taxonomy identifiers.

5.12.4 Known problems

None reported

5.12.5 Future work

Upon request.

6 Data Preparation Protocol

The protocol described here is design for preparing the data obtained from `http://www.ncbi.nlm.nih.gov/` repository. Once downloaded, using this protocol, the data gets processed and adapted for software tools introduced in sections above.

6.1 Download Data

The first step in data preparation is to retrieve the data from an on-line repository. In this particular situation the data is downloaded from `http://www.ncbi.nlm.nih.gov/`:

Three downloaded gzip files are:

1. `taxdump.tar.gz`
2. `nr.gz`
3. `gi_taxid_prot.dmp.gz`

Next each file is extracted by executing the following set of commands:

1. extracting taxonomy information (`names.dmp`, `nodes.dmp`):

   ```
   tar -xvzf taxdump.tar.gz
   ```

2. extracting nr database:

   ```
   gunzip nr.gz
   ```

3. extracting gi-taxonomy association table (`taxid`):

   ```
   gunzip gi_taxid_prot.dmp.gz
   ```

6.2 Reformat files

In order to betted organize your project it is always a good idea to create a new working directory to which relevant files should be relocated to and in which data processing takes place:

```
mkdir SandBox
cd SandBox/
mv ../names.dmp ./
mv ../nodes.dmp ./
mv ../nr ./
my ../gitax ./
```

Once relevant files have been downloaded and extracted, preparation starts by reformatting three files: `names.dmp`,`nodes.dmp` and `nr`. The program designed for this purpose is called `NCBIDataParser.pl` and is located in `./scripts/` directory. The complete description of its utility can be found in section 5.11:

By executing:

```
perl ../scripts/NCBIDataParser.pl -m names.dmp \
                                  -n nodes.dmp \
                                  -d nr
```

three `*.fmt` files should be generated and located in the above specified working directory (`./SandBox`):

```
SandBox/names.dmp.fmt
SandBox/nodes.dmp.fmt
SandBox/nr.fmt
```

6.3 Build the Database

Downloaded and formatted nr database is then further divided into a set of smaller multi-fasta record files, each containing a set of sequences associated to a given taxonomy identifier and also labelled by it. For this procedure a program called `Nr2Ph.pl` has been created (for a more detailed description of this tool, please refer to section 5.8). By executing:

```
perl ../scripts/Nr2Ph.pl -n nr -t gitax
```

a database (multi-file directory) called `./phdb` is created. In order to further prepare the database for tools like `QPhyloStrat` (Sec. 4.5) and/or `PhyloClust` (Sec. 4.6), the database needs to be stripped of sequence segments labelled as low compositional complexity regions. For that purpose the tool called `MakePhyloDb` was created (Sec. 4.1). The program is applied to the database by executing:

```
../bin/MakePhyloDb -d phdb
```

Once the execution is completed the data is prepared to be used as an input for software within a desired pipeline (Figure 1). Usually, if not certain the phylogeny (nodes file) is properly edited and synchronized with the database, after introducing changes with `MakeTree` software the synchronization can be achieved by executing:

```
perl ../scripts/DbSync.pl -d phdb -n nodes.dmp.new
```

As a result each genome within `phdb` not properly located within phylogeny tree will be reverted back to its original form and name.

```
9606.ff -> 9606
```

The user can then remove those files from the database or further tackle the problems within phylogeny tree in order to resolve it. If a decision has been made to delete unsynchronized genomes, a newly generated nodes file should be use in all downstream analysis. The file is located within the same directory as `nodes.dmp.new`, having the extension `*.sync` added to its original name:

```
nodes.dmp.new -> nodes.dmp.new.sync
```

Also before any additional steps are made the database should be formatted once more in order to recompute its actual and effective length. This is achieved by once again invoking `MakePhyloDb` on `./phdb`:

```
../bin/MakePhyloDb -d phdb
```

7 Acknowledgements

Claverie, J.-M. and States, D.J. (1993). Information enhancement methods for large-scale sequence analysis. Computers and Chemistry 17:191-201.

Domazet-Loso T, Brajkovic J, Tautz D. (2008). A phylostratigraphy approach to uncover the genomic history of major adaptations in metazoan lineages. Trends Genet. 23:533–539.

Fischer, J. and V. Heun (2006). "Theoretical and practical improvements on the RMQ-problem, with applications to LCA and LCE". Combinatorial Pattern Matching: 36-48.

Fischer, J. and Heun, V. (2007). A New Succinct Representation of RMQ-Information and Improvements in the Enhanced Suffix Array. Proceedings of the International Symposium on Combinatorics, Algorithms, Probabilistic and Experimental Methodologies. LNCS 4614. Springer. pp. 459-470.

Tomita, E.,Tanaka, A. and Takahashi, H. The wrst-case time complexity for generating all maximal cliques and computational experiments. (2006) Theor. Comput. Sci. 363:28-42

Wootton, J. C. and S. Federhen (1993). Statistics of local complexity in amino acid sequences and sequence databases. Computers and Chemistry 17:149-163.

Curriculum Vitae

Name	Robert Bakarić
Date and place of birth	01.07.1984., Kutina, Croatia
Nationality	Croatian

Education

Undergraduate studies

2003-2008	Diploma/MSc in Molecular Biology, University of Zagreb, Faculty of Science, Division of Biology, Zagreb, Croatia

Graduate studies

2008-2011	University of Zagreb, Faculty of Science, Zagreb, Croatia.
2011-2014	International Max Planck Research School, Ploen, Germany.
2011-2016	Christian-Albrecht-University of Kiel, Kiel, Germany

Stipends / Awards

2004-2008	Scholarship for excellent students. Croatian Ministry of Science, Education and Sports, Zagreb, Croatia.
2011-2014	International Max Planck Research School fellowship. Ploen, Germany

Work experience

2008-2016	Research Assistant, Ruder Bošković Institute, Zagreb, Croatia
2014-today	Bioinformatic consultant / Chief Technology Officer (CTO), Exaltum, Zagreb, Croatia

Professional memberships

2009-2010	Leader of "Young Researchers", section of "Croatian Society of Natural Sciences"
2010-2012	Acting secretary of the "Croatian Society for Theoretical and Mathematical Biology"

Scientific interest

Bioinformatics	Evolution
Next Generation Sequencing	Genomic Phylostratigraphy
Information theory	Statistics
Data Structures and Algorithms	Genetics

Declaration

I hereby declare:

i. that except where reference is made to the work of others and apart from my supervisors guidance, the content and design of this dissertation is product of my own work;

ii. that this thesis has not been submitted either partially or wholly as part of a doctoral degree to another examination body, and no other materials are published or submitted for publication;

iii. that the thesis has been prepared subject to the Rules of Good Scientific Practice of the German Research Foundation.

Robert Bakarić

Kiel, 2016.