Reconciliation with Segmental Duplication, Transfer, Loss and Gain

Yoann Anselmetti[1,2][0000−0002−6689−1163], Mattéo Delabre[1][0000−0003−4561−683X], and Nadia El-Mabrouk[1]

[1] Département d'informatique (DIRO), Université de Montréal
[2] Laboratoire CoBIUS, Département d'informatique, Université de Sherbrooke
mabrouk@iro.umontreal.ca

Abstract. We generalize the reconciliation approach, used for inferring the evolution of a single gene family given a species tree, to groups of co-localized genes, also called syntenies. More precisely, given a set $\mathcal{X}$ of syntenies in a set Σ of genomes, a tree T for $\mathcal{X}$ and a tree S for Σ, the problem is to find a most parsimonious history for $\mathcal{X}$ with respect to a given evolutionary model. We extend a previous model involving segmental duplications and losses, to also include segmental horizontal gene transfers (HGTs) and gene gains. We present a polynomial-time dynamic programming algorithm to solve the problem. We apply it to CRISPR-associated (Cas) gene syntenies. These genes are part of CRISPR-Cas systems, one of its members (CRISPR-Cas9) well-known as currently the most reliable and accurate molecular scissor technology for genome editing. The inferred evolutionary scenario is a plausible explanation of the diversification of this system into its different types. An implementation of the algorithm presented in this paper is available at: https://github.com/UdeM-LBIT/superrec2/releases/tag/rcg2022.

Keywords: Reconciliation · Synteny · CRISPR-Cas · Horizontal gene transfer.

1 Introduction

The incongruence between the tree of a given gene family and the phylogenetic tree of the corresponding species can be explained through *reconciliation* (an embedding of the gene tree into the species tree) by the fact that genes have been subject to events changing their occurrence in genomes, typically gene duplications (D) and gene losses (L) [12]. The standard parsimony criteria used to choose among all possible reconciliations is to minimize the number of duplications (D distance) and losses (DL distance) induced by the reconciliation. This can be computed in linear time by LCA-mapping [12, 31, 32].

Horizontal gene transfer (HGT), largely involved in shaping bacterial gene content, has also been considered in the analysis of gene families through reconciliation. In this case, the parsimony problem consists in finding a minimum scenario of duplication, loss and transfer events (DTL distance) explaining a gene

tree with respect to a given species tree. The problem of finding a most parsimonious acyclic DTL scenario has been shown NP-hard, becoming polynomial when the acyclicity requirement is relaxed [1, 27].

Although used successfully for many years, one of the major drawbacks of the reconciliation model is that gene families are considered to evolve independently from one another. While some work has been done on inferring the evolution of co-localized genes (such as operons in bacteria or paralogons), also called *syntenic groups of genes* (or simply *syntenies*) [2, 9], adjusting the computation of the evolutionary cost to favour co-evolution events—hence grouping individual events into single segmental ones [7]—or inferring the minimum number of "duplication episodes" defined as sets of single duplications mapped to the same node in the species tree [6, 23], none of these methods explicitly seek for an evolutionary scenario minimizing segmental duplication, loss and HGT events (see a recent review [8]).

The first attempt to generalize the reconciliation approach to a set of gene trees was described in [5]. Given a set of gene families grouped into ordered syntenies (i.e., ordered groups of genes), a gene tree for each gene family and a species tree, the *DL Super-reconciliation* problem was defined as finding an evolutionary scenario for the syntenies agreeing with the individual gene trees, whilst minimizing the number of segmental duplications and losses. The problem admits a solution only in the case of "consistent" gene trees and gene orders. It was shown that the associated decision problem is NP-hard, and that a two-steps method on the synteny tree (obtained as a supertree of the gene trees), first assigning an event labeling from the LCA-mapping and then inferring ancestral syntenies and losses, leads to an optimal solution. Moreover, ignoring gene orders, a polynomial-time algorithm exists for the second step.

In this paper, we describe DTL Super-reconciliation, generalizing the model to handle HGT and gene gain events. We restrict the problem to the unordered case, where syntenies are defined as unordered groups of genes. We introduce the evolutionary model in Section 3 and the formal optimization problem in Section 4. We show in Section 5 that the two-steps method for solving the DL Super-reconciliation problem does not apply in this case, then present a polynomial-time dynamic programming algorithm for DTL Super-reconciliation in Section 6. Finally, in Sections 7 and 8, we apply our algorithm to CRISPR-associated (Cas) gene syntenies. These genes are part of CRISPR-Cas systems, one of its members (CRISPR-Cas9) well-known as currently the most reliable and accurate molecular scissor technology for genome editing. The inferred evolutionary scenario leads to an interesting explanation of the diversification of this system into its different types, which opens the door to further investigations.

2 Preliminary Definitions

All trees are considered rooted. Given a tree T, we denote by $r(T)$ its root, $\mathrm{V}(T)$ its node set and by $\mathrm{L}(T) \subseteq \mathrm{V}(T)$ its leaf set. A node v' is an *ancestor* of v if v' is on the path from $r(T)$ to v; the *parent* $\mathrm{p}(v)$ of v, of which v is a *child*,

directly precedes v on this path. Conversely, v is a *descendant* of v'. Notice that a node v is both an ancestor and a descendant of itself; where this case needs to be excluded, we will talk about strict ancestors and descendants. Two nodes are *separated* in T if neither is an ancestor of the other. We denote by $\mathrm{E}(T)$ the set of edges of T, where each edge is represented by a pair of nodes $(\mathrm{p}(v), v)$. For any two nodes v_1 and v_2 of T, the node distance $\mathrm{D}_T^{\mathrm{node}}(v_1, v_2)$ is defined as the number of edges on the unique path from v_1 to v_2.

Given a node v of T, $T[v]$ is the *subtree* of T rooted at v (i.e., containing only the descendants of v). The *lowest common ancestor* (LCA) of a subset V of nodes, denoted $\mathrm{lca}_T(V)$, is the ancestor of all nodes in V that is the most distant from the root. A node is said to be *unary* if it has exactly one child and *binary* if it has exactly two. A *binary tree* is a tree where all internal (non-leaf) nodes are binary. If all internal nodes are unary or binary, then the tree is called *partially binary*. The two children of a binary node v are denoted v_l and v_r for the "left" and "right" child. Notice that the considered trees are unordered, and thus left and right are set arbitrarily.

If A is a set of labels (on a given finite alphabet), then any tree T such that there exists a one-to-one relation between A and $\mathrm{L}(T)$ is said to *be a tree for* A. In particular, a *species tree* S for a set Σ of species represents an ordered set of speciation events that led to Σ (i.e., each internal node of S represents an ancestral species preceding a speciation event). Similarly, a *gene tree* T for a gene family Γ is a branching history encoding each gene divergence that led to the gene family Γ. For a gene g of Γ, we denote by $s(g)$ the species of Σ the gene g belongs to.

Let $\mathcal{F}$ be a set of gene families. In this paper, a *syntenic group* or *synteny* X is a non-empty subset of $\mathcal{F}$ representing a group of co-localized genes, where the relative order of genes in the genomic region is ignored. The genes of a synteny are considered to all belong to different gene families (i.e., duplications inside a syntenic group are not allowed), therefore the genes are simply identified by their family Γ of $\mathcal{F}$. Given two syntenies X and Y, we say that there is a loss between X and Y if at least one of the gene families from X is absent in Y. Notice that, due to the possibility of gene gains, Y may contain genes not found in X. The loss indicator function $\mathrm{D}^{\mathrm{sub}}(X, Y)$ (*sub* for "subset") is therefore defined as $\mathrm{D}^{\mathrm{sub}}(X, Y) = 0$ if $X \subseteq Y$, otherwise $\mathrm{D}^{\mathrm{sub}}(X, Y) = 1$.

A *synteny family* is a set $\mathcal{X}$ of syntenies. A *synteny tree* T is a tree for a synteny family $\mathcal{X}$, $x(l)$ being the synteny of $\mathcal{X}$ associated to each leaf l of T. In this paper, the species and synteny trees are considered binary. See the left part of Figure 1 for an example of a synteny tree.

3 Evolutionary Histories for Syntenies

The *Super-reconciliation* framework introduced in [5] generalizes the reconciliation framework from a single gene family Γ to a family $\mathcal{X}$ of syntenies. More precisely, while an instance of a reconciliation problem is a tuple $\langle \Gamma, T, \Sigma, S \rangle$, T being a gene tree for Γ, an instance of a Super-reconciliation problem is a tuple

$\langle \mathcal{X}, \mathcal{F}, T, \Sigma, S \rangle$ where T is a synteny tree for the synteny family $\mathcal{X}$ on a set $\mathcal{F}$ of gene families. Notice that, while in [5] the synteny tree is inferred from a set of "consistent" gene trees, in this paper we start from the synteny tree itself, whatever the way used to infer it (e.g., from a set of gene trees, or alternatively from an alignment of the concatenated gene sequences).

The goal of the reconciliation approach is to infer a *correct* and *optimal* evolutionary history explaining T given S. Correctness depends on the considered evolutionary events, while optimality is stated as a parsimony criterion, given a cost function for the evolutionary events. As for Super-reconciliation, in addition to evolutionary events, ancestral syntenies should also be inferred.

The DL Super-reconciliation model [5], defined both on ordered and unordered syntenies, only involves speciations, duplications and losses. Here, we extend the unordered version to also allow for HGTs. We assume that evolution only takes place inside the S species tree, excluding speciations and transfers to and from extinct or unsampled lineages [26, 30]. Also notice that the constraints on HGTs do not exclude cyclic reconciliations. In addition, in order to avoid the unrealistic assumption of having all gene families present at the root, we allow for gene gains.

Definition 1 (Evolutionary history for syntenies). *Let $\mathcal{X}$ be a set of syntenies on a set $\mathcal{F}$ of gene families, Σ be the set of taxa to which these syntenies belong and S be the species tree for Σ. An $\mathcal{E}$ evolutionary history, with $\mathcal{E} \subseteq \{\text{Spe}, \text{Dup}, \text{HGT}, \text{Loss}, \text{Gain}\}$, is a partially binary tree with leaves mapped to $\mathcal{X}$ and where each internal node v corresponds to an event $e(X)$ of $\mathcal{E}$ with $X \subseteq \mathcal{F}$ being the synteny at v belonging to a genome $s(v)$ such that:*

- Spe *produces two syntenies Y and Z verifying $X = Y = Z$ and $s(Y)$, $s(Z)$ are the two children of $s(X)$ in S.*
- Dup (D) *produces two syntenies Y and Z verifying $Y = X$, $Z \subseteq X$, and $s(X) = s(Y) = s(Z)$.*
- HGT (T) *produces two syntenies Y and Z verifying $Y = X$, $Z \subseteq X$, $s(Y) = s(X)$ and $s(Z)$ is separated from $s(X)$.*
- Loss (L) *produces a single synteny Y verifying $Y \subset X$ and $s(Y) = s(X)$. A loss is* full *if Y is the empty synteny (i.e., $Y = \emptyset$) and* partial *otherwise.*
- Gain (G) *produces a single synteny Y verifying $X \subset Y$ and $s(Y) = s(X)$.*

We denote by $H = \langle H^{\text{tree}}, e, x, s \rangle$ such a history where H^{tree} is the supporting partially binary tree and for each of its nodes v, $e(v)$ is the event, $x(v)$ is the synteny, and $s(v)$ is the species to which $x(v)$ belongs. We have $\mathcal{X} = \{x(l) : l \in \text{L}(H^{\text{tree}})\}$ (Figure 1, center). Note that for an internal node v of H^{tree}, $x(v)$ is not necessarily in $\mathcal{X}$. In general, x is not an onto function, as not all possible syntenies on $\mathcal{F}$ are required to be represented in the history.

We next define a Super-reconciliation from a history. In Definition 2, the $\mathcal{E}$-Super-reconciliation obtained from H is $\langle T_H, e, x, s \rangle$ where T_H is the binary tree obtained from H^{tree} by removing edges adjacent to empty syntenies (due to full losses) and then removing unary nodes, and e, x and s are restrictions of the event, synteny and species mappings to the nodes of T_H (Figure 1, right).

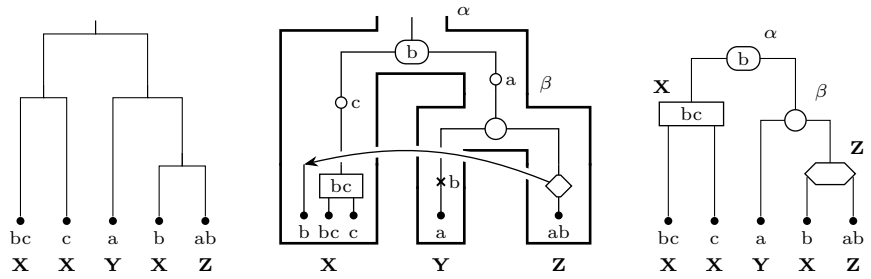

Fig. 1. (Left.) A synteny tree T for the family $\mathcal{X} = \{\{\mathrm{b}\}, \{\mathrm{b}, \mathrm{c}\}, \{\mathrm{c}\}, \{\mathrm{a}\}, \{\mathrm{a}, \mathrm{b}\}\}$ on the set of species $\Sigma = \{\mathbf{X}, \mathbf{Y}, \mathbf{Z}\}$. Under each leaf l are shown its associated synteny $x(l)$ and species $s(l)$. (Center.) An evolutionary history H^{tree} for T, embedded in the species tree S. For binary nodes, rounded rectangles correspond to Spe, plain rectangles to Dup, and chamfered rectangles to HGT. For unary nodes, circles correspond to Gain, and crosses to Loss. For a binary node v, the value of $x(v)$ is shown inside the shape (omitted when unchanged from its parent). For unary nodes, the synteny difference is shown beside the node. (Right.) The Super-reconciliation $\langle T, x, s\rangle$ obtained from H^{tree}. For each internal node v, we exhibit $e(v)$ (shape of the node), $s(v)$ (letter beside the node) and $x(v)$ (content of the node).

Definition 2 (Super-reconciliation). *Let $I = \langle \mathcal{X}, \mathcal{F}, T, \Sigma, S\rangle$ be an instance of a Super-reconciliation problem. An* $\mathcal{E}$-Super-reconciliation *for I is a tuple $\mathcal{R} = \langle T_H, e, x, s\rangle$ obtained from an $\mathcal{E}$ history $H = \langle H^{\mathrm{tree}}, e, x, s\rangle$. We say that H is a history for I leading to $\mathcal{R}$.*

Note that an *$\mathcal{E}$-Reconciliation* is defined as an $\mathcal{E}$-Super-reconciliation but on an instance $\langle \Gamma, T, \Sigma, S\rangle$. The next lemma, which will be required later for a simple inference of the event labeling from the species labeling, directly follows from Definition 1 and Definition 2.

Lemma 1 (Synteny and species trees coincide). *Let $I = \langle \mathcal{X}, \mathcal{F}, T, \Sigma, S\rangle$ and $\mathcal{R} = \langle T, e, x, s\rangle$ be a* DTLG *Super-reconciliation for I (i.e. a Super-reconciliation on the set of events $\mathcal{E} = \{\mathrm{Spe}, D, T, L, G\}$). Then s satisfies the following conditions: (1) neither $s(v_l)$ nor $s(v_r)$ is a strict ancestor of $s(v)$ (i.e. each one can either be a descendant of $s(v)$ or be separated from it in the case of HGTs); (2) at least one of $s(v_l)$ or $s(v_r)$ is a descendant of $s(v)$.*

Restrictions on gain points: Allowing for segmental gains and losses may lead to unrealistic optimal scenarios where full syntenies are lost and gained again. In order to ensure more realistic scenarios, we will assume that a gene family can only appear once in the history (i.e., not allowing for convergent evolution).

Definition 3 (Gain point). *Let T be a synteny tree for a synteny family $\mathcal{X}$ on $\mathcal{F}$ and x be a synteny assignment on the internal nodes of T. A node $v \in \mathrm{V}(T)$ is a* gain point *for $\Gamma \in \mathcal{F}$ in $\langle T, x\rangle$ iff $\Gamma \in x(v)$ and either $v = \mathrm{r}(T)$ or $\Gamma \notin x(\mathrm{p}(v))$. The set of gain points for Γ is denoted as* $\mathrm{Gain}_{\langle T,x\rangle}(\Gamma)$.

Moreover, as gain of function affects genes individually, we will restrict a gain to an event inserting a single gene in a synteny. Formally, we add a restriction to the Gain event of Definition 1 specifying that the synteny Y produced from a synteny X by a gain is such that $|Y| = |X| + 1$.

Consequently, each gene family $\Gamma \in \mathcal{F}$ can only be gained once in the history, leading to exactly $|\mathcal{F}|$ gains. In other words, we account for gene gains only to avoid imposing all gene families to be present at the root of T, but without including them in the cost function for inferring a most parsimonious history. Consequently, we can define our problem as a DTL (for D, T, and L events) rather than a DTLG problem (D, T, L, and G events) as follows, considering the above restriction on the Gain event.

Definition 4 (DTL Super-reconciliation). *Let $I = \langle \mathcal{X}, \mathcal{F}, T, \Sigma, S \rangle$. A* DTL *Super-Reconciliation for I is a* DTLG *Super-reconciliation $\langle T, e, x, s \rangle$ such that, for each $\Gamma \in \mathcal{F}$, $|\mathrm{Gain}_{\langle T,x \rangle}(\Gamma)| = 1$.*

4 Most Parsimonious Super-Reconciliations

In this paper, we assume a null cost for speciations and, according to the restriction on Gain events described above, we can ignore the cost of gains. We then define $\delta = \langle c_{\mathrm{Dup}}, c_{\mathrm{HGT}}, c_{\mathrm{Loss}} \rangle \in (\mathbb{R}^+ \cup \{\infty\})^3$ the cost for, respectively, a duplication, an HGT and a loss event. For a history $H = \langle H^{\mathrm{tree}}, e, x, s \rangle$ and a node $v \in \mathrm{V}(H^{\mathrm{tree}})$, we define $c_\delta(H, v)$ to be the sum of costs of events in the $H^{\mathrm{tree}}[v]$ subtree, up to and including v itself. The history's overall cost $c_\delta(H)$ is equal to $c_\delta(H, \mathrm{r}(H^{\mathrm{tree}}))$.

The goal is to find a most parsimonious history (i.e., a history of minimum cost), explaining a given synteny tree T with respect to a species tree S. From Definition 2, a Super-reconciliation $\mathcal{R}$ for an instance I represents not a single, but rather a set of histories H from which $\mathcal{R}$ can be obtained. In the rest of this section, we give some results allowing to reduce the problem to the exploration of the Super-reconciliation space rather than the history space.

The next lemma states that gain points can be inferred from the synteny tree. For each family $\Gamma \in \mathcal{F}$, denote by $\mathrm{L}(T)_\Gamma = \{l \in \mathrm{L}(T) : \Gamma \in x(l)\}$ the set of leaves whose corresponding synteny contains Γ and $\mathrm{lca}(\Gamma, T) = \mathrm{lca}_T(\mathrm{L}(T)_\Gamma)$.

Lemma 2 (Optimal gain point position). *Let $I = \langle \mathcal{X}, \mathcal{F}, T, \Sigma, S \rangle$. There is an optimal history H for I such that, in the Super-reconciliation $\mathcal{R} = \langle T, e, x, s \rangle$ obtained from H, for each $\Gamma \in \mathcal{F}$, $\mathrm{lca}(\Gamma, T)$ is the gain point for Γ in T.*

Proof. This proof and all subsequent ones can be found in the appendix.

We denote by $x^{\mathrm{gain}}(v)$ the set of genes gained at node v of T. We next introduce a way to assign a cost to a Super-reconciliation which, as we show in the subsequent lemma, matches the minimum cost of any history leading to that Super-reconciliation.

Definition 5 (Super-reconciliation cost). *Let $\mathcal{R} = \langle T, e, x, s\rangle$ be a Super-reconciliation for $I = \langle \mathcal{X}, \mathcal{F}, T, \Sigma, S\rangle$. The* cost *$C_\delta(\mathcal{R}, v)$ of $\mathcal{R}$ for the $T[v]$ subtree (or simply $C(\mathcal{R}, v)$ or $C(v)$ if no ambiguity) is defined recursively as follows:*

- *if v is a leaf, then $C_\delta(\mathcal{R}, v) = 0$;*
- *if $e(v) = \text{Spe}$ and v_l, v_r are the two children of v, then*

$$\begin{aligned} C_\delta(\mathcal{R}, v) = C_\delta(\mathcal{R}, v_l) + C_\delta(\mathcal{R}, v_r) \; + \\ c_{\text{Loss}} \times (\text{D}^{\text{sub}}(x(v), x(v_l)) + \text{D}^{\text{sub}}(x(v), x(v_r)) + \\ \text{D}_S^{\text{node}}(s(v), s(v_l)) + \text{D}_S^{\text{node}}(s(v), s(v_r)) - 2); \end{aligned}$$

- *if $e(v) = \text{Dup}$ and v_l, v_r are the two children of v, then*

$$\begin{aligned} C_\delta(\mathcal{R}, v) = c_{\text{Dup}} + C_\delta(\mathcal{R}, v_l) + C_\delta(\mathcal{R}, v_r) \; + \\ c_{\text{Loss}} \times (\min\left\{\text{D}^{\text{sub}}(x(v), x(v_l)), \text{D}^{\text{sub}}(x(v), x(v_r))\right\} \; + \\ \text{D}_S^{\text{node}}(s(v), s(v_l)) + \text{D}_S^{\text{node}}(s(v), s(v_r))); \end{aligned}$$

- *if $e(v) = \text{HGT}$, v' is the separated child, and v'' is the conserved one, then*

$$\begin{aligned} C_\delta(\mathcal{R}, v) = c_{\text{HGT}} + C_\delta(\mathcal{R}, v_l) + C_\delta(\mathcal{R}, v_r) \; + \\ c_{\text{Loss}} \times (\text{D}^{\text{sub}}(x(v), x(v'')) \; + \text{D}_S^{\text{node}}(s(v), s(v'')). \end{aligned}$$

The global cost of $\mathcal{R}$ is defined as $C_\delta(\mathcal{R}) = C_\delta(\mathcal{R}, r(T))$ (or simply $C(\mathcal{R})$ if no ambiguity).

Lemma 3 (Super-reconciliations minimize history cost). *Let $\mathcal{R} = \langle T, e, x, s\rangle$ be a Super-reconciliation for $I = \langle \mathcal{X}, \mathcal{F}, T, \Sigma, S\rangle$ and $\mathcal{H}$ be the set of histories leading to $\mathcal{R}$. Then, for any $C_\delta(\mathcal{R}) = \min_{H \in \mathcal{H}} c_\delta(H)$.*

Searching for a most parsimonious history is therefore equivalent to searching in the smaller space of super-reconciliations. Finally, the following definition makes the link between the values of the s mapping and the evolutionary events at the internal nodes of a Super-reconciliation.

Definition 6 (Min-event labeling). *Given $I = \langle \mathcal{X}, \mathcal{F}, T, \Sigma, S\rangle$, an internal node v of T, and three species σ, σ', σ'' of S, we define the* min-event *function $e^{\min}(\sigma, \sigma', \sigma'')$, used to label v if $s(v) = \sigma$, $s(v_l) = \sigma'$, and $s(v_r) = \sigma''$, as follows:*

- *If σ is an ancestor of both σ' and σ'', then*
  - *if σ' is separated from σ'' and $\sigma = \text{lca}(\sigma', \sigma'')$, then $e^{\min}(\sigma, \sigma', \sigma'') = \text{Spe}$;*
  - *if σ' and σ'' are not separated or $\sigma \neq \text{lca}(\sigma', \sigma'')$, then $e^{\min}(\sigma, \sigma', \sigma'') = \text{Dup}$.*
- *If either σ' or σ'' is separated from σ, then $e^{\min}(\sigma, \sigma', \sigma'') = \text{HGT}$.*
- *Species σ' and σ'' cannot be both separated from σ as per Lemma 1, therefore in such cases $e^{\min}(\sigma, \sigma', \sigma'')$ equals an error value not in $\{\text{Spe}, \text{Dup}, \text{HGT}\}$.*

Additionally, given a mapping s from $\mathrm{V}(T)$ to $\mathrm{V}(S)$, we define $e_s^{\min}$ so that for any internal node v of T, $e_s^{\min}(v) = e^{\min}(s(v), s(v_l), s(v_r))$.

The following lemma shows that the min-event labeling leads to the most parsimonious Super-reconciliation.

Lemma 4 (Min-event labeling is optimal). *Let $\mathcal{R} = \langle T, e, x, s\rangle$ and $\mathcal{R}^{\min} = \langle T, e_s^{\min}, x, s\rangle$ be two DTL Super-reconciliations for $I = \langle \mathcal{X}, \mathcal{F}, T, \Sigma, S\rangle$. Then, $C_\delta(\mathcal{R}^{\min}) \leq C_\delta(\mathcal{R})$.*

It follows that event labeling can be ignored, as it can be directly inferred from the species mapping. Therefore, from now on, a Super-reconciliation will be simply designed as a triplet $\langle T, x, s\rangle$. We are now ready to formally define the considered optimization problem.

δ-Super-Reconciliation Problem
Input: An input $I = \langle \mathcal{X}, \mathcal{F}, T, \Sigma, S\rangle$.
Output: A Super-reconciliation $\mathcal{R} = \langle T, x, s\rangle$ for I minimizing $C_\delta(\mathcal{R})$.

5 A Two-Steps Method

Finding a DL Reconciliation of a gene tree with a species tree is a classical problem [10, 12, 14, 22]. Given an instance $I = \langle \Gamma, T, \Sigma, S\rangle$, define its LCA-mapping from T to S as $s(v) = \mathrm{lca}_S(\{s(l) \mid l \in \mathrm{L}(T[v])\})$ for any $v \in \mathrm{V}(T)\backslash\mathrm{L}(T)$, and $s(l) = x$ for any $l \in \mathrm{L}(T)$, where x is the extant species to which l belongs. This mapping leads to an optimal DL Reconciliation (with constant costs on operations) and can be computed in time $\mathcal{O}(|\mathrm{V}(T)| + |\mathrm{V}(S)|)$ [13].

As for DL Super-reconciliation, an approach using two steps to infer an optimal $\mathcal{R} = \langle T, x, s\rangle$ for an instance $I = \langle \mathcal{X}, \mathcal{F}, T, \Sigma, S\rangle$ was presented in [5]:

1. Compute s as the LCA-mapping of T to S;
2. Infer x in a way minimizing segmental losses and with the constraint that $x(r(T)) = \mathcal{F}$ (as gains were not allowed).

Using this approach, an optimal DL Super-reconciliation can be computed in time $\mathcal{O}(|\mathcal{X}|\times|\mathcal{F}|)$. Crucially, this approach works because any optimal DL Super-reconciliation $\langle T, x, s\rangle$ is such that s is the LCA-mapping of T to S. Otherwise, we would not be able to compute s separately from x.

Unfortunately, this two-steps method does not work for solving the DTL Super-reconciliation problem. In fact, for a given input $I = \langle \Gamma, T, \Sigma, S\rangle$, the mapping s^{DT} from T to S allowing to minizime the duplication and loss cost is not necessarily the mapping s of an optimal DTL Super-reconciliation $\mathcal{R} = \langle T, x, s\rangle$. In particular, changing a speciation node (inferred from s^{DT}) to a HGT event may lead to less losses, and thus to a better cost in total. Figure 2 shows a counter example of the two-steps method for solving the DTL problem.

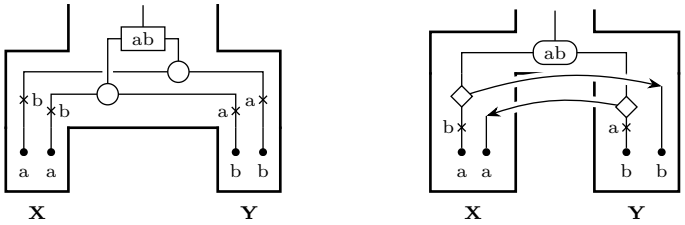

Fig. 2. Two solutions for the same DTL Super-reconciliation problem input. (Left.) Solution obtained by first computing s to minimize duplications and HGTs (yielding one such event) and then labeling internal nodes to minimize losses (yielding 4 losses). (Right.) A more parsimonious solution with two HGTs and only two losses.

6 A Dynamic Programming Algorithm for DTL Super-reconciliation

In this section, we introduce SuperDTL, a polynomial-time algorithm for solving the DTL Super-Reconciliation problem. Let $I = \langle \mathcal{X}, \mathcal{F}, T, \Sigma, S \rangle$ be any input for this problem. For any node v of T, we define $x^{\text{lca}}(v)$ to be the leafset of $T[v]$ excluding genes gained below v, namely:

$$x^{\text{lca}}(v) = \begin{cases} x(v) & \text{if } v \in \text{L}(G) \\ x^{\text{lca}}(v_l) \cup x^{\text{lca}}(v_r) \setminus (x^{\text{gain}}(v_l) \cup x^{\text{gain}}(v_r)) & \text{otherwise.} \end{cases}$$

For a synteny X and a node v of T, denote by $C_X(v)$ the minimum cost of a Super-reconciliation $\langle T[v], x, s \rangle$ between $T[v]$ and S in which $x(v) = X$. Notice that $x^{\text{lca}}(v)$ should be a subset of X as otherwise there would be a gene family with two or more gain points (by Definition 3). In other words, if $x^{\text{lca}}(v) \not\subseteq X$, then $C_X(v) = \infty$. However, X may contain genes in $\mathcal{F} \setminus x^{\text{lca}}(v)$, which may allow grouping losses in the evolutionary history thus leading to a lower cost. Testing all possible subsets of $\mathcal{F} \setminus x^{\text{lca}}(v)$ would be costly, but as shown in [5], this can be avoided due to a property that still holds in our case: all that matters is to know whether $x^{\text{lca}}(v)$ is included in X in a strict or not strict way, and the nature and number of "extra" genes is irrelevant for the computation of the optimal cost. In other words, the following lemma from [5] holds.

Lemma 5. *Let v be an internal node of T and X, Y be two subsets of $\mathcal{F}$ such that $x^{\text{lca}}(v) \subseteq X, Y$. Then $C_X(v) = C_Y(v)$.*

We therefore only need to consider two possibilities for $x(v)$. We define $\text{C}(v, \sigma)$ (respec. $\text{C}^\star(v, \sigma)$) to be the minimum cost of a Super-reconciliation $\langle T[v], x, s \rangle$ between $T[v]$ and S in which $s(v) = \sigma$ and $x(v) = x^{\text{lca}}(v)$ (respec. $x(v) \supsetneq x^{\text{lca}}(v)$). Algorithms 1 and 2, described below, provide a method for computing those two functions.

Algorithm 1 Computing the value of $\mathrm{C}(v, \sigma)$

function $\mathcal{C}(v, \sigma)$
  if $v \in \mathrm{L}(T)$ **then**
    return 0 **if** $\sigma = s(v)$ **else** ∞
  else
    $c \leftarrow \infty$
    for $\sigma', \sigma'' \in \mathrm{V}(S)^2$ **do**
      $\mathit{star}_l \leftarrow \infty$ **if** $x^{\mathrm{lca}}(v) \subseteq x^{\mathrm{lca}}(v_l)$ **else** 0
      $\mathit{star}_r \leftarrow \infty$ **if** $x^{\mathrm{lca}}(v) \subseteq x^{\mathrm{lca}}(v_r)$ **else** 0
      $\mathit{partial}_l \leftarrow \mathrm{D}^{\mathrm{sub}}(x^{\mathrm{lca}}(v), x^{\mathrm{lca}}(v_l))$
      $\mathit{partial}_r \leftarrow \mathrm{D}^{\mathrm{sub}}(x^{\mathrm{lca}}(v), x^{\mathrm{lca}}(v_r))$
      if $e^{\min}(\sigma, \sigma', \sigma'') = \mathrm{Spe}$ **then**
        $c \leftarrow \min\{c,\ \min\{\mathcal{C}^\star(v_l, \sigma') + \mathit{star}_l, \mathcal{C}(v_l, \sigma') + c_{\mathrm{Loss}} \times \mathit{partial}_l\} +$
        $\min\{\mathcal{C}^\star(v_r, \sigma'') + \mathit{star}_r, \mathcal{C}(v_r, \sigma'') + c_{\mathrm{Loss}} \times \mathit{partial}_r\} +$
        $c_{\mathrm{Loss}} \times (\mathrm{D}_S^{\mathrm{node}}(\sigma, \sigma') + \mathrm{D}_S^{\mathrm{node}}(\sigma, \sigma'') - 2)\}$
      else if $e^{\min}(\sigma, \sigma', \sigma'') = \mathrm{Dup}$ **then**
        $c \leftarrow \min\{c,\ \min\{\mathcal{C}^\star(v_l, \sigma') + \mathcal{C}^\star(v_r, \sigma'') + \mathit{star}_l + \mathit{star}_r,$
        $\mathcal{C}(v_l, \sigma') + \mathcal{C}^\star(v_r, \sigma'') + \mathit{star}_r,$
        $\mathcal{C}^\star(v_l, \sigma') + \mathcal{C}(v_r, \sigma'') + \mathit{star}_l,$
        $\mathcal{C}(v_l, \sigma') + \mathcal{C}(v_r, \sigma'') +$
        $c_{\mathrm{Loss}} \times \min\{\mathit{partial}_l, \mathit{partial}_r\}\} +$
        $c_{\mathrm{Dup}} + c_{\mathrm{Loss}} \times (\mathrm{D}_S^{\mathrm{node}}(\sigma, \sigma') + \mathrm{D}_S^{\mathrm{node}}(\sigma, \sigma''))\}$
      else if $e^{\min}(\sigma, \sigma', \sigma'') = \mathrm{HGT}$ **then**
        $\mathit{full} \leftarrow \mathrm{D}_S^{\mathrm{node}}(\sigma, \sigma'')$ **if** σ' is separated from σ **else** $\mathrm{D}_S^{\mathrm{node}}(\sigma, \sigma')$
        if σ' is separated from σ **then** $\mathit{partial}_l \leftarrow 0$
        if σ'' is separated from σ **then** $\mathit{partial}_r \leftarrow 0$
        $c \leftarrow \min\{c,\ \min\{\mathcal{C}^\star(v_l, \sigma') + \mathcal{C}^\star(v_r, \sigma''),$
        $\mathcal{C}(v_l, \sigma') + \mathcal{C}^\star(v_r, \sigma'') + \mathit{star}_r + c_{\mathrm{Loss}} \times \mathit{partial}_l,$
        $\mathcal{C}^\star(v_l, \sigma') + \mathcal{C}(v_r, \sigma'') + \mathit{star}_l + c_{\mathrm{Loss}} \times \mathit{partial}_r,$
        $\mathcal{C}(v_l, \sigma') + \mathcal{C}(v_r, \sigma'') + c_{\mathrm{Loss}}\} +$
        $c_{\mathrm{HGT}} + c_{\mathrm{Loss}} \times \mathit{full}\}$
    return c

Lemma 6 (Termination and correctness). *For any $v \in \mathrm{V}(T)$ and $\sigma \in \mathrm{V}(S)$, $\mathcal{C}(v, \sigma) = \mathrm{C}(v, \sigma)$ and $\mathcal{C}^\star(v, \sigma) = \mathrm{C}^\star(v, \sigma)$ (as computed by Algorithms 1 and 2 respectively).*

Algorithm SuperDTL computes the minimal cost of a Super-reconciliation for I by computing $\min_{\sigma \in \mathrm{V}(S)} \mathcal{C}(\mathrm{r}(T), \sigma)$ using Algorithm 1, which recursively invokes itself and Algorithm 2. Additionally, an actual solution can be constructed by keeping track of which σ', σ'' pairs and which of C or $\mathrm{C}^\star$ yield the minimum values of the min expressions in both algorithms. To make SuperDTL efficient, it should not be implemented as a naive recursion, but rather C and $\mathrm{C}^\star$ should be considered as dynamic programming tables with $|\mathrm{V}(T)| \times |\mathrm{V}(S)|$ entries each.

Algorithm 2 Computing the value of $\mathrm{C}^\star(v, \sigma)$

function $\mathcal{C}^\star(v, \sigma)$
  if $v \in \mathrm{L}(T)$ **then return** ∞
  else
    $c \leftarrow \infty$
    for $\sigma', \sigma'' \in \mathrm{V}(S)^2$ **do**
      if $e^{\min}(\sigma, \sigma', \sigma'') = \mathrm{Spe}$ **then**
        $c \leftarrow \min\{c, \min\{\mathcal{C}^\star(v_l, \sigma'), \mathcal{C}(v_l, \sigma') + c_{\mathrm{Loss}}\} +$
        $\min\{\mathcal{C}^\star(v_r, \sigma''), \mathcal{C}(v_r, \sigma'') + c_{\mathrm{Loss}}\} +$
        $c_{\mathrm{Loss}} \times (\mathrm{D}_S^{\mathrm{node}}(\sigma, \sigma') + \mathrm{D}_S^{\mathrm{node}}(\sigma, \sigma'') - 2)\}$
      else if $e^{\min}(\sigma, \sigma', \sigma'') = \mathrm{Dup}$ **then**
        $c \leftarrow \min\{c, \min\{\mathcal{C}^\star(v_l, \sigma') + \mathcal{C}^\star(v_r, \sigma''), \mathcal{C}(v_l, \sigma') + \mathcal{C}^\star(v_r, \sigma''),$
        $\mathcal{C}^\star(v_l, \sigma') + \mathcal{C}(v_r, \sigma''), \mathcal{C}(v_l, \sigma') + \mathcal{C}(v_r, \sigma'') + c_{\mathrm{Loss}}\}$
        $+ c_{\mathrm{Dup}} + c_{\mathrm{Loss}} \times (\mathrm{D}_S^{\mathrm{node}}(\sigma, \sigma') + \mathrm{D}_S^{\mathrm{node}}(\sigma, \sigma''))\}$
      else if $e^{\min}(\sigma, \sigma', \sigma'') = \mathrm{HGT}$ **then**
        $\mathit{full} \leftarrow \mathrm{D}_S^{\mathrm{node}}(\sigma, \sigma'')$ **if** σ' is separated from σ **else** $\mathrm{D}_S^{\mathrm{node}}(\sigma, \sigma')$
        $\mathit{partial}_l \leftarrow 0$ **if** σ' is separated from σ **else** 1
        $\mathit{partial}_r \leftarrow 0$ **if** σ'' is separated from σ **else** 1
        $c \leftarrow \min\{c, \min\{\mathcal{C}^\star(v_l, \sigma') + \mathcal{C}^\star(v_r, \sigma''),$
        $\mathcal{C}(v_l, \sigma') + \mathcal{C}^\star(v_r, \sigma'') + c_{\mathrm{Loss}} \times \mathit{partial}_l,$
        $\mathcal{C}^\star(v_l, \sigma') + \mathcal{C}(v_r, \sigma'') + c_{\mathrm{Loss}} \times \mathit{partial}_r,$
        $\mathcal{C}(v_l, \sigma') + \mathcal{C}(v_r, \sigma'') + c_{\mathrm{Loss}}\} +$
        $c_{\mathrm{HGT}} + c_{\mathrm{Loss}} \times \mathit{full}\}$
    return c

Using this implementation trick allows finding a minimal Super-reconciliation in polynomial time, as shown in the next theorem.

Theorem 1 (Time and space complexity). *Using* SuperDTL*, the* DTL Super-reconciliation *problem can be solved in polynomial time* $\mathcal{O}(|\mathcal{X}| \times |\Sigma|^3)$ *and space* $\mathcal{O}(|\mathcal{X}| \times |\Sigma|)$.

7 Application to CRISPR-associated (Cas) gene syntenies

7.1 Cas gene syntenies

Cas genes are part of CRISPR-Cas systems, one of its members, CRISPR-Cas9, being well-known as currently one of the most reliable and accurate "molecular scissor" biotechnology for genome editing. This technology, for which the Nobel Prize in Chemistry was awarded in 2020 [16], was derived from an adaptive bacterial immunity system targeting bacteriophages. The study and analysis of CRISPR-Cas systems over the past two decades has revealed their wide diversity

and led to their categorization into two classes: Class 1, composed of multisubunit effector proteins, and Class 2, composed of a single large effector protein. Each class is further divided into several types themselves composed of several subtypes (for more details see Supplementary Table 2 in Makarova *et al.* [20]). Although the discovery of new CRISPR-Cas systems is an ongoing process [18], the classification is generally stable.

As the function of CRISPR-Cas systems highly depends on the syntenic organization of Cas genes, elucidating the evolution of these systems is crucial. Many studies have been dedicated to reconstructing the evolutionary histories of individual Cas gene families such as Cas1 [15], or to inferring the evolution of Cas gene syntenies to elucidate the syntenic events leading to the diversity inside the different subtypes [17, 20, 21]. From these multiple phylogenetic analyses, a global scenario has been predicted for the evolutionary formation of CRISPR-Cas systems, with the latest scenario described by Koonin *et al.* [18]. However, to the best of our knowledge, none of these studies take into account the species tree topology. Moreover, several studies point to evidence of HGTs involving Cas genes between prokaryote species [11, 28, 15, 3], showing the need to take such events into consideration while inferring scenarios.

From this brief presentation, it appears that the DTL Super-reconciliation model is suitable for elucidating the evolution and radiation of CRISPR-Cas systems across prokaryotes.

7.2 Dataset

We used the Cas gene synteny subtypes from Class 1 of CRISPR-Cas systems described by Makarova *et al.* [20]. We limited the dataset to the 15 bacteria species and omitted archaea to avoid bias due to their underrepresentation in the dataset. Each of those species contains a Cas synteny. Taxonomical information of the 15 bacterial species has been recovered from the NCBI Taxonomy database [25] and the species tree topology is based on the phylogeny inferred by Coleman *et al.* [4].

We repurposed the phylogenetic tree given in Figure 1 of Makarova *et al.* [20] as our synteny tree. Some alterations of the syntenies were required to fit the constraints of the model. We considered Cas families Cas1–8 and Cas10–11. Since a part of Cas3, Cas3", can work as a standalone HD nuclease, we split Cas3 into Cas3' and Cas3". In [20], Cas10 and Cas8 share the same colour code as they provide similar functions in CRISPR-Cas complexes. Indeed, it was initially believed that Cas8 evolved from Cas10 [19]. Nevertheless, their sequence being extremely divergent, we decided to consider them as not homologous and to conserve separate Cas8 and Cas10 families. Finally, for syntenies with several copies of the same family, we conserved only one copy per synteny. The obtained Cas gene syntenies are illustrated in the left part of Figure 3.

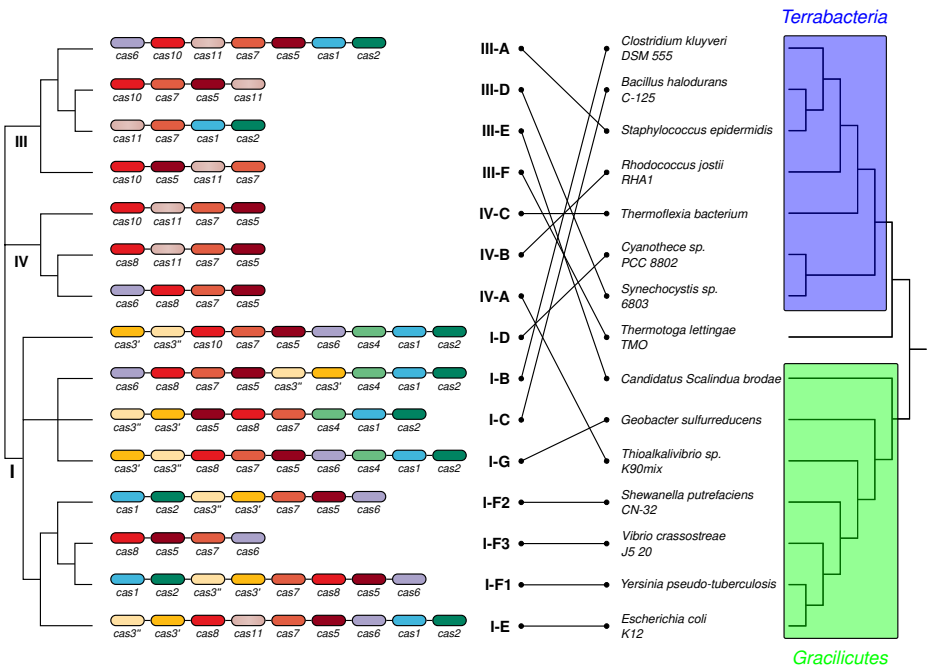

Fig. 3. Cas gene syntenies for the 15 considered bacterial species. (Left.) Phylogeny of Class 1 CRISPR-Cas systems with subtypes names, as presented in [20] with our preprocessing of syntenies as described in the text. (Right.) The species tree based on the topology inferred in [4] with representation of the two major groups of bacteria, Terrabacteria and Gracilicutes. Lines represent the correspondence between the Cas genes syntenies and the species they belong to, illustrating the high incongruence between the topology of the synteny tree and that of the species tree.

8 Results

8.1 DTL Super-reconciliation settings

We used SuperDTL to predict optimal DTL Super-reconciliations for the syntenies and trees depicted in Figure 3. Notice that the synteny tree is non-binary and contains three multifurcations, one at the root and two in the subtree of Type I CRISPR-Cas systems. As our algorithm can only be applied to binary trees, we test all possible "binarizations" of the synteny tree and retain the overall minimal solutions.

We tested different values for the $\delta = \langle c_{\mathrm{Dup}}, c_{\mathrm{HGT}}, c_{\mathrm{Loss}} \rangle$ cost model, in agreement with a classical assumption that HGTs are less frequent than duplications, which are less frequent than losses. The number of solutions obtained for each setting is given in Table 1. We observe that the number of solutions decreases as the HGT cost increases, reaching a minimum for $c_{\mathrm{HGT}} = 4$, and then increases again. For a given c_{HGT}, the results are largely stable for different c_{Dup} values.

$c_{\mathrm{HGT}} \backslash c_{\mathrm{Dup}}$	**1**	**1.5**	**2**	**2.5**	**3**
1	1376	1376	1376	1376	1376
2	132	132	132	132	132
3	112	60	60	60	60
4	32	32	32	32	48
5	288	288	320	32	32
6	320	288	288	288	288

Table 1. Number of solutions obtained by Algorithm SuperDTL for different values of $\delta = \langle c_{\mathrm{Dup}}, c_{\mathrm{HGT}}, c_{\mathrm{Loss}} \rangle$, with $c_{\mathrm{Loss}} = 1$.

8.2 An evolutionary scenario

We analyzed the 32 solutions generated for $c_{\mathrm{HGT}} = 4$ and $c_{\mathrm{Dup}} \in \{1, 1.5, 2, 2.5\}$. All solutions lead to the same resolution of the three multifurcations in the synteny tree, and the incongruence observed between the species and synteny tree topologies is mainly resolved by inferring six HGTs and one duplication. The 32 solutions only differ from one another by minor variations, such as gene losses located either before or after a speciation or HGT event, which do not change the overall inferred evolutionary history for the Cas gene syntenies.

Figure 4 is a representation of one of these solutions. The evolutionary reconstruction is broadly consistent with the CRISPR-Cas evolution scenario established by Koonin *et al* [18]. In both their scenario and ours, the CRISPR-Cas systems emerge from an initial adaptive immune system composed only of Cas genes involved in the effector complex (Cas5–8 and Cas10–11). We see from Figure 4 that the CRISPR-Cas emergence is inferred at the root of Terrabacteria. The scenarios diverge on the fact that, in our scenario, Type IV emerges directly from this ancestral adaptive immune system before the acquisition of Cas1 and Cas2 genes, while in Koonin *et al.*, Type IV emerges after the acquisition of Cas1 and Cas2 adaptation module genes and loss of several Cas genes. Aside from this difference, both scenarios are in agreement for Types I and III in terms of gene content in ancestral Cas syntenies. As it appears in Figure 4, Types I and III Cas syntenies emerge from Cyanobacteria after the acquisition of Cas1 and Cas2 by an ancestor of Cyanobacteria. Type I emerges from the branch leading to *Cyanothece sp. PCC 8802* with acquisition of Cas3 and Cas4, while Type III emerges from the branch leading to *Synechocystis sp. 6803* without further gene gains.

Type I was spread to the other bacteria with a HGT to the ancestor of Proteobacteria. According to Wang *et al.* [29], the Cyanobacteria ancestor is estimated between 2,230 and 3,000 Mya, while the last common ancestor (LCA) of Alpha-, Beta-, and Gammaproteobacteria is estimated between 2,360 and 2,620 Mya. These calibrations make time-plausible the lateral transfer of Type I

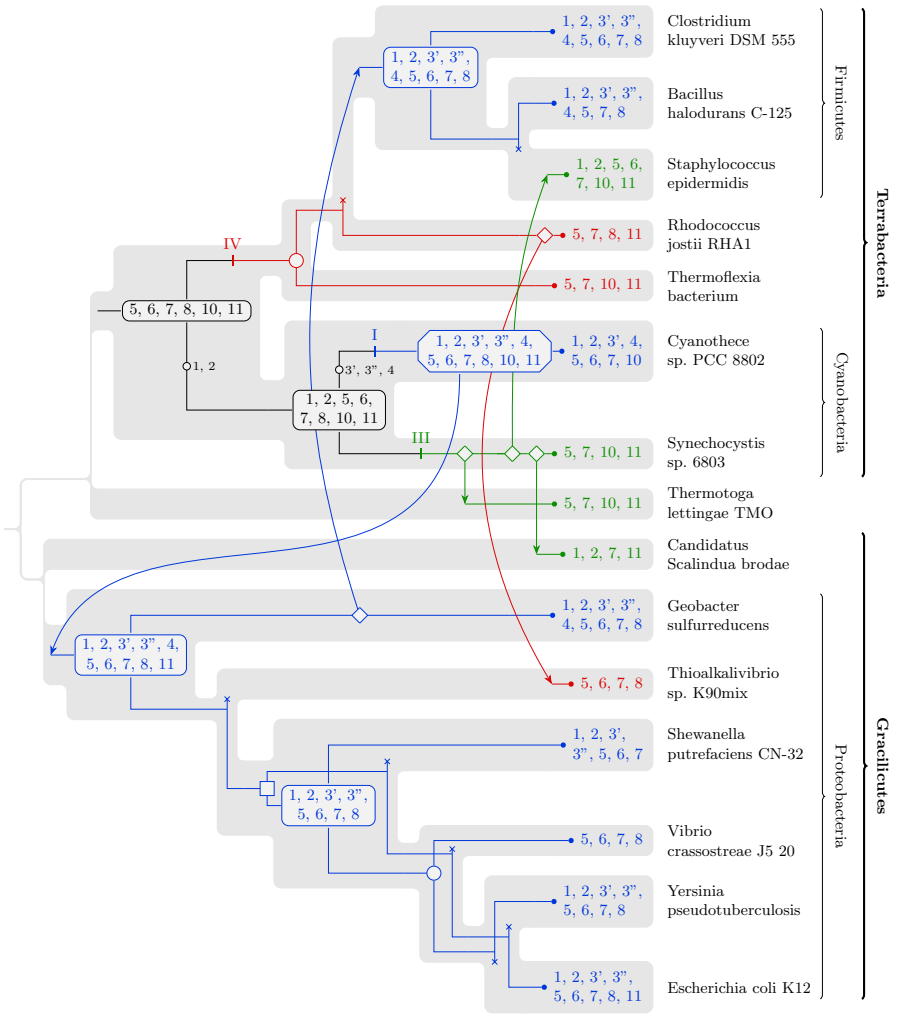

Fig. 4. Representation of one of the 32 solutions inferred by Algorithm SuperDTL reconstructing a DTL evolutionary history of the Class 1 Cas gene synteny dataset illustrated in Figure 3. Event costs set to $c_{\text{Loss}} = 1$, $c_{\text{HGT}} = 4$ and $c_{\text{Dup}} \in \{1, 1.5, 2, 2.5\}$ yield equal solutions. The red, blue, and green parts of the tree reflect the evolution of Cas synteny types IV, I, and III respectively. Gains of Cas1+Cas2 and Cas3(Cas3'+Cas3")+Cas4 are illustrated.

CRISPR-Cas from the *Cyanothece sp. PCC 8802* branch, close to the Cyanobacteria ancestor, to the LCA of Proteobacteria. A second HGT from *Geobacter sulfurreducens* brought Type I to the Firmicutes *Bacillus halodurans C-125* and *Clostridium kluyveri DSM 555*. Type III was spread across bacteria with a succession of three HGTs, which also seem time-consistent.

The analysis of the synteny evolutionary history across the Proteobacteria subtree shows an unexpected scenario highlighting a limitation of our model. The SuperDTL algorithm inferred an ancestral synteny duplication before the LCA of *S. putrefaciens*, *V. crassostreae*, *Y. pseudotuberculosis* and *E. coli* resulting in a succession of three consecutive full synteny losses along the branch to the synteny in *E. coli* which is an unlikely evolutionary scenario. An alternative evolutionary scenario would consist in a speciation in place of the duplication, copying the ancestral synteny to an unsampled or extinct species which would later be transferred back to *Escherichia coli*. Such a model of HGT events as combinations of a speciation event outside the species tree (to an unsampled or dead lineage) followed by a transfer back inside the species tree has been described in [26]. This alternative scenario would replace one duplication and three losses with one speciation and one transfer, which yields the same cost under the $c_{\text{Loss}} = 1$, $c_{\text{HGT}} = 4$ and $c_{\text{Dup}} \in \{1, 1.5, 2, 2.5\}$ model but would be less costly for $c_{\text{HGT}} < 4$. Consideration of unsampled and dead lineages in the DTL Super-reconciliation will therefore be necessary to infer better synteny evolutionary scenarios.

9 Conclusion

We have developed SuperDTL, the first exact algorithm for inferring most parsimonious evolutionary histories for a set of syntenies, given a phylogenetic tree of the syntenies and a phylogenetic tree of the corresponding species, for an evolutionary model accounting for segmental duplications, gains, losses and HGT events. We only presented the unordered version of the problem in this paper, but the algorithm developed in [5] for the ordered version of the DL Super-reconciliation model can also be extended to the DTL Super-reconciliation model. However, as rearrangements are not considered in the model, a DT or DTL Super-reconciliation only exists if extant gene orders are pairwise consistent, which is a strong constraint, not verified, for example, in the case of the Cas gene syntenies of Figure 3. Considering a unifying model accounting for both DTL and rearrangement events remains a challenge. The analysis of the synteny evolutionary scenario in Proteobacteria emphasized the need in DTL Super-reconciliation model to infer HGT to and from unsampled or extinct species to produce more realistic evolutionary scenarios. In addition, in order to avoid the unjustified constraint of having all genes at the root of the tree, we allowed for gene gains, which has been essential for the analysis of Cas gene syntenies. Although we defined the general evolutionary model in a way allowing for segmental gains (i.e. gains of a group of genes), we only considered single gene gains

in our algorithm. An extension is however possible and may be considered for future work.

The reconstruction of an evolutionary history for Cas gene syntenies using SuperDTL provides a first attempt to reconcile the evolutionary scenario of Cas syntenies in the context of the evolution of bacterial species. However, several improvements could be brought to the considered Cas synteny dataset to better reconstruct its evolutionary history. First, we excluded archaea from our dataset while several studies show evidence for an emergence of CRISPR-Cas systems from archaeal species followed by horizontal transfers to bacteria. This is supported by the fact that most considered archaea have a complete CRISPR-Cas system while only part of bacteria have one. In fact, as of the latest update from January 21, 2021 of the CRISPRCasdb database[3] [24], 70.64% of analyzed archaeal species versus 36.27% of analyzed bacteria had a complete CRISPR-Cas system. A phylogenetic dating approach could also be used to produce a dated phylogeny of prokaryotes and to constrain HGT events. A larger dataset with species sampling representing the diversity of archaea and bacteria with dated phylogeny and Cas synteny tree based on synteny content and sequence divergence is required to further elucidate the evolutionary history of the CRISPR-Cas system in prokaryotes.

[3] Available at `https://crisprcas.i2bc.paris-saclay.fr/MainDb/StrainList`

Bibliography

[1] Mukul S. Bansal, Eric J. Alm, and Manolis Kellis. Efficient algorithms for the reconciliation problem with gene duplication, horizontal transfer and loss. *Bioinformatics*, 28(12):i283–i291, 2012.

[2] S. Bérard, C. Gallien, B. Boussau, G.J. Szollosi, V. Daubin, and E. Tannier. Evolution of gene neighborhoods within reconciled phylogenies. *Bioinformatics*, 28(18):i382-i388, 2012.

[3] Sajib Chakraborty, Ambrosius P. Snijders, Rajib Chakravorty, Musaddeque Ahmed, Ashek Md. Tarek, and M. Anwar Hossain. Comparative network clustering of direct repeats (DRs) and cas genes confirms the possibility of the horizontal transfer of CRISPR locus among bacteria. *Molecular Phylogenetics and Evolution*, 56(3):878–887, September 2010.

[4] Gareth A. Coleman, Adrián A. Davín, Tara A. Mahendrarajah, Lénárd L. Szánthó, Anja Spang, Philip Hugenholtz, Gergely J. Szöllősi, and Tom A. Williams. A rooted phylogeny resolves early bacterial evolution. *Science*, May 2021.

[5] Mattéo Delabre, Nadia El-Mabrouk, Katharina T. Huber, Manuel Lafond, Vincent Moulton, Emmanuel Noutahi, and Miguel Sautie Castellanos. Evolution through segmental duplications and losses: a super-reconciliation approach. *Algorithms for Molecular Biology*, 15(12), May 2020.

[6] R. Dondi, M. Lafond, and C. Scornavacca. Reconciling multiple genes trees via segmental duplications and losses. *Algorithms for Molecular Biology*, 14, 2019.

[7] Wandrille Duchemin. *Phylogeny of dependencies and dependencies of phylogenies in genes and genomes*. Theses, Université de Lyon, December 2017.

[8] Nadia El-Mabrouk. Predicting the evolution of syntenies—an algorithmic review. *Algorithms*, 14(5):152, 2021.

[9] W. Duchemin *et al.* DeCoSTAR: Reconstructing the ancestral organization of genes or genomes using reconciled phylogenies. *Genome Biol. Evol.*, 2017.

[10] Walter M. Fitch. Distinguishing homologous from analogous proteins. *Systematic Zoology*, 19(2):99, June 1970.

[11] James S. Godde and Amanda Bickerton. The repetitive DNA elements called CRISPRs and their associated genes: Evidence of horizontal transfer among prokaryotes. *Journal of Molecular Evolution*, 62(6):718–729, June 2006.

[12] M. Goodman, J. Czelusniak, G. W. Moore, A. E. Romero-Herrera, and G. Matsuda. Fitting the gene lineage into its species lineage, a parsimony strategy illustrated by cladograms constructed from globin sequences. *Systematic Biology*, 28(2):132–163, June 1979.

[13] Pawel Górecki and Jerzy Tiuryn. DLS-trees: A model of evolutionary scenarios. *Theoretical Computer Science*, 359(1-3):378–399, 2006.

[14] Roderic Guigo, Ilya Muchnik, and Temple F. Smith. Reconstruction of ancient molecular phylogeny. *Molecular Phylogenetics and Evolution*, 6(2):189–213, October 1996.

[15] Philippe Horvath, Anne-Claire Coûté-Monvoisin, Dennis A. Romero, Patrick Boyaval, Christophe Fremaux, and Rodolphe Barrangou. Comparative analysis of CRISPR loci in lactic acid bacteria genomes. *International Journal of Food Microbiology*, 131(1):62–70, April 2009.

[16] Martin Jinek, Krzysztof Chylinski, Ines Fonfara, Michael Hauer, Jennifer A. Doudna, and Emmanuelle Charpentier. A programmable dual-RNA–guided DNA endonuclease in adaptive bacterial immunity. *Science*, 337(6096):816–821, August 2012.

[17] Eugene V. Koonin and Kira S. Makarova. Origins and evolution of CRISPR-Cas systems. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 374(1772):20180087, May 2019.

[18] Eugene V. Koonin and Kira S. Makarova. Evolutionary plasticity and functional versatility of CRISPR systems. *PLOS Biology*, 20(1):e3001481, January 2022.

[19] Kira S. Makarova, L. Aravind, Yuri I. Wolf, and Eugene V. Koonin. Unification of Cas protein families and a simple scenario for the origin and evolution of CRISPR-Cas systems. *Biology Direct*, 6(1):38, July 2011.

[20] Kira S. Makarova, Yuri I. Wolf, Jaime Iranzo, Sergey A. Shmakov, Omer S. Alkhnbashi, Stan J. J. Brouns, Emmanuelle Charpentier, David Cheng, Daniel H. Haft, Philippe Horvath, and et al. Evolutionary classification of CRISPR–Cas systems: a burst of class 2 and derived variants. *Nature Reviews Microbiology*, 18(2):67–83, February 2020.

[21] Ana Moya-Beltrán, Kira S. Makarova, Lillian G. Acuña, Yuri I. Wolf, Paulo C. Covarrubias, Sergey A. Shmakov, Cristia Silva, Igor Tolstoy, D. Barrie Johnson, Eugene V. Koonin, and et al. Evolution of type IV CRISPR-Cas systems: Insights from CRISPR loci in integrative conjugative elements of acidithiobacillia. *The CRISPR Journal*, 4(5):656–672, October 2021.

[22] Roderic D. M. Page. Maps between trees and cladistic analysis of historical associations among genes,organisms, and areas. *Systematic Biology*, 43(1):58–77, 1994.

[23] J. Paszek and P. Gorecki. Efficient algorithms for genomic duplication models. *IEEE/ACM Trans. Comput. Biol. Bioinform.*, 2017.

[24] Christine Pourcel, Marie Touchon, Nicolas Villeriot, Jean-Philippe Vernadet, David Couvin, Claire Toffano-Nioche, and Gilles Vergnaud. CRISPRCasdb a successor of CRISPRdb containing CRISPR arrays and cas genes from complete genome sequences, and tools to download and query lists of repeats and spacers. *Nucleic Acids Research*, 48(D1):D535–D544, January 2020.

[25] Conrad L. Schoch, Stacy Ciufo, Mikhail Domrachev, Carol L. Hotton, Sivakumar Kannan, Rogneda Khovanskaya, Detlef Leipe, Richard Mcveigh, Kathleen O'Neill, Barbara Robbertse, Shobha Sharma, Vladimir Soussov, John P. Sullivan, Lu Sun, Seán Turner, and Ilene Karsch-Mizrachi. NCBI

taxonomy: a comprehensive update on curation, resources and tools. *Database*, 2020, August 2020.

[26] Gergely J. Szöllősi, Eric Tannier, Nicolas Lartillot, and Vincent Daubin. Lateral gene transfer from the dead. *Systematic Biology*, 62(3):386–397, February 2013.

[27] A. Tofigh, M. Hallett, and J. Lagergren. Simultaneous identification of duplications and lateral gene transfers. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 8(2):517–535, March 2011.

[28] Gene W. Tyson and Jillian F. Banfield. Rapidly evolving CRISPRs implicated in acquired resistance of microorganisms to viruses. *Environmental Microbiology*, 10(1):200–207, 2008.

[29] Sishuo Wang, Andrew Meade, Hon-Ming Lam, and Haiwei Luo. Evolutionary timeline and genomic plasticity underlying the lifestyle diversity in rhizobiales. *mSystems*, 5(4):e00438–20, July 2020.

[30] Samson Weiner and Mukul S. Bansal. Improved duplication-transfer-loss reconciliation with extinct and unsampled lineages. *Algorithms*, 14(8):231, August 2021.

[31] L. Zhang. On a Mirkin–Muchnik–Smith conjecture for comparing molecular phylogenies. *J. Comput. Biol.*, 4(2):177–187, 1997.

[32] C. M. Zmasek and S. R. Eddy. A simple algorithm to infer gene duplication and speciation events on a gene tree. *Bioinformatics*, 17:821– 828, 2001.

A Additional Content for Section 4 ("Most Parsimonious Super-Reconciliations")

Lemma 2 (Optimal gain point position). *Let $I = \langle \mathcal{X}, \mathcal{F}, T, \Sigma, S \rangle$. There is an optimal history H for I such that, in the Super-reconciliation $\mathcal{R} = \langle T, e, x, s \rangle$ obtained from H, for each $\Gamma \in \mathcal{F}$, $\mathrm{lca}(\Gamma, T)$ is the gain point for Γ in T.*

Proof. Let $H = \langle H^{\mathrm{tree}}, e, x, s \rangle$. Let v be the node corresponding to $\mathrm{lca}(\Gamma, T)$ in H^{tree}. The fact that $\mathrm{lca}(\Gamma, T)$ is the gain point for Γ in T means that the gene family Γ is gained on the branch (u, v) where u is the root of H^{tree} or the node just preceding v in H^{tree}. The result then follows from the fact that: (1) Γ cannot be gained on a node of the left subtree $H^{\mathrm{tree}}[v_l]$ of $H^{\mathrm{tree}}[v]$ as in that case, Γ should also have been gained on another node on the right subtree $H^{\mathrm{tree}}[v_r]$ of $H^{\mathrm{tree}}[v]$ (or conversely); (2) thus Γ can only be gained on a node between the root r of H^{tree} and v; moving the gain point to an ancestor of v cannot decrease the number of losses. □

Lemma 7. *Let $H = \langle H^{\mathrm{tree}}, e, x, s \rangle$ be a history, and $v \neq w$ be two nodes of H^{tree} such that none of the nodes on the path from v to w (excluding v and w themselves) is a* HGT *event and such that $s(v)$ and $s(w)$ are not separated. Then, there are at least $\mathrm{D}_S^{\mathrm{node}}(s(v), s(w))$ speciation nodes on the path from v to $\mathrm{p}(w)$.*

Proof. This follows from the constraints of Definition 1, which states that the only way to descend in the species tree, excluding HGT events, is through Spe events. □

Lemma 8. *Let $H = \langle H^{\mathrm{tree}}, e, x, s \rangle$ be a history, and v, w_l, w_r be three nodes of H^{tree} such that v is a binary node, w_l descends from v_l and w_r from v_r, and none of the nodes on the paths from v to w_l and v to w_r (excluding v, w_l, and w_r themselves) are* Dup *or* HGT *events. Then:*

1. *If $e(v) =$ Spe, there are at least $\mathrm{D}^{\mathrm{sub}}(x(v), x(w_l)) + \mathrm{D}^{\mathrm{sub}}(x(v), x(w_r))$ loss events on the path from w_l to w_r.*
2. *If $e(v) =$ Dup, there are at least $\min\{\mathrm{D}^{\mathrm{sub}}(x(v), x(w_l)), \mathrm{D}^{\mathrm{sub}}(x(v), x(w_r))\}$ loss events on the path from w_l to w_r.*
3. *If $e(v) =$ HGT, if $s(w_l)$ (resp. w_r) is not separated from $s(w)$ there is at least $\mathrm{D}^{\mathrm{sub}}(x(v), x(w_l))$ (resp. w_r) loss event on the path from w to w_l (resp. w_r).*

Proof. This follows from the constraints of Definition 1, which states that the only way to loose part of a synteny, excluding Dup and HGT events, is through Loss events, and that Dup events allow loosing part of their synteny on either of their children, while HGT events allow loosing part of their synteny on their conserved child. □

Lemma 3 (Super-reconciliations minimize history cost). *Let $\mathcal{R} = \langle T, e, x, s \rangle$ be a Super-reconciliation for $I = \langle \mathcal{X}, \mathcal{F}, T, \Sigma, S \rangle$ and $\mathcal{H}$ be the set of histories leading to $\mathcal{R}$. Then, for any $C_\delta(\mathcal{R}) = \min_{H \in \mathcal{H}} c_\delta(H)$.*

Proof. Let $H = \langle H^{\text{tree}}, e, x, s\rangle$ be a history leading to $\mathcal{R}$. Let us first prove that $c_\delta(H) \geq C_\delta(\mathcal{R})$, by structural induction on T. Let v be a node of T and v' be its corresponding node in H^{tree}. If v is a leaf, then $c_\delta(H, v') \geq C_\delta(\mathcal{R}, v) = 0$; otherwise, let v'_l and v'_r be the nodes in H^{tree} corresponding to v_l and v_r respectively, and assume $c_\delta(H, v'_l) \geq C_\delta(\mathcal{R}, v_l)$ and $c_\delta(H, v'_r) \geq C_\delta(\mathcal{R}, v_r)$.

- If $e(v) = \text{Spe}$, by Lemma 7, there are at least $\text{D}_S^{\text{node}}(s(v), s(v_l)) - 1$ speciation nodes on the path from $(v')_l$ to v'_l and at least $\text{D}_S^{\text{node}}(s(v), s(v_r)) - 1$ speciation nodes on the path from $(v')_r$ to v'_r. Each of those speciation nodes must have at least a full loss child, for otherwise v'_l or v'_r would not correspond to v_l and v_r. By Lemma 8.1, there are at least $\text{D}^{\text{sub}}(x(v), x(v_l)) + \text{D}^{\text{sub}}(x(v), x(v_r))$ loss events on the path from v'_l to v'_r.
- If $e(v) = \text{Dup}$, by Lemma 7, there are at least $\text{D}_S^{\text{node}}(s(v), s(v_l))$ speciation nodes on the path from $(v')_l$ to v'_l and at least $\text{D}_S^{\text{node}}(s(v), s(v_r))$ speciation nodes on the path from $(v')_r$ to v'_r, each of which must have a full loss child as per the same argument as above. By Lemma 8.2, there are at least $\min\{\text{D}^{\text{sub}}(x(v), x(v_l)), \text{D}^{\text{sub}}(x(v), x(v_r))\}$ loss events on the path from v'_l to v'_r.
- If $e(v) = \text{HGT}$, assume w.l.o.g. that v_l is the conserved child. By Lemma 7, there are at least $\text{D}_S^{\text{node}}(s(v), s(v_l))$ speciation nodes on the path from $(v')_l$ to v'_l, each of which must have a full loss child as per the same argument as above. By Lemma 8.3, there are at least $\text{D}^{\text{sub}}(x(v), x(v_l)$ loss events on the path from v' to v'_l.

In all three cases, $c_\delta(H, v') \geq C_\delta(\mathcal{R}, v)$, concluding the first part of the proof. Additionally, it is easy to see that a history $H_\mathcal{R}$ can be constructed from $\mathcal{R}$ such that $H_\mathcal{R} \in \mathcal{H}$ and $c_\delta(H_\mathcal{R}) = C_\delta(\mathcal{R})$, by inserting speciation nodes and loss nodes in the locations described above. □

Lemma 4 (Min-event labeling is optimal). *Let $\mathcal{R} = \langle T, e, x, s\rangle$ and $\mathcal{R}^{\min} = \langle T, e_s^{\min}, x, s\rangle$ be two DTL Super-reconciliations for $I = \langle \mathcal{X}, \mathcal{F}, T, \Sigma, S\rangle$. Then, $C_\delta(\mathcal{R}^{\min}) \leq C_\delta(\mathcal{R})$.*

Proof. Given s, the choice for e is constrained by Definition 1, as evidenced by Lemma 1. For any internal node v of T, if either $s(v_l)$ or $s(v_r)$ is separated from $s(v)$, then it must be that $e(v) = \text{HGT}$. If $s(v_l)$ and $s(v_r)$ are not separated (i.e., either one is an ancestor of the other), then it must be that $e(v) = \text{Dup}$. Finally, if $s(v_l)$ is separated from $s(v_r)$, then $e_s^{\min}(v) = \text{Spe}$, but it would also be valid to set $e(v)$ to Dup. However, in that case, replacing Dup back with Spe would save two full losses at the cost of adding at most one partial loss, which cannot lead to a more costly Super-reconciliation since speciations have a null cost and $c_{\text{Dup}}, c_{\text{Loss}} \geq 0$. □

B Additional Content for Section 6 ("A Dynamic Programming Algorithm for DTL Super-reconciliation")

Lemma 6 (Termination and correctness). *For any $v \in \mathrm{V}(T)$ and $\sigma \in \mathrm{V}(S)$, $\mathcal{C}(v,\sigma) = \mathrm{C}(v,\sigma)$ and $\mathcal{C}^\star(v,\sigma) = \mathrm{C}^\star(v,\sigma)$ (as computed by Algorithms 1 and 2 respectively).*

Proof. First, note that both algorithms terminate even though they are mutually recursive, since any call to $\mathcal{C}(v,\sigma)$ or $\mathcal{C}^\star(v,\sigma)$ calls $\mathcal{C}$ and $\mathcal{C}^\star$ only on v_l and v_r. We proceed by structural induction to prove correctness. Let v be a leaf of T. Leaves cannot be labeled by any synteny or mapped to any species other than the ones specified in the problem input, therefore $\mathrm{C}(v,s(v)) = 0 = \mathcal{C}(v,s(v))$, $\mathrm{C}(v,\sigma) = \infty = \mathcal{C}(v,\sigma)$ for any $\sigma \neq s(v)$, and $\mathrm{C}^\star(v,\sigma) = \infty = \mathcal{C}^\star(v,\sigma)$ for any $\sigma \in \mathrm{V}(S)$.

Now, let v be an internal node and assume that for any $\sigma \in \mathrm{V}(S)$, $\mathcal{C}^\star(v_l,\sigma) = \mathrm{C}^\star(v_l,\sigma)$, $\mathcal{C}^\star(v_r,\sigma) = \mathrm{C}^\star(v_r,\sigma)$, $\mathcal{C}(v_l,\sigma) = \mathrm{C}(v_l,\sigma)$, and $\mathcal{C}(v_r,\sigma) = \mathrm{C}(v_r,\sigma)$. Let σ be any node of the species tree. Both $\mathcal{C}^\star(v,\sigma)$ and $\mathcal{C}(v,\sigma)$ explore all possible pairs $\sigma', \sigma'' \in \mathrm{V}(S)^2$ that can be mapped respectively to v_l and v_r, and return the minimum cost of all those options.

Consider the computation of $\mathcal{C}^\star(v,\sigma)$ by Algorithm 2. Since in that case $x(v) \supsetneq x^{\mathrm{lca}}(v)$, then $x(v) \not\subseteq x^{\mathrm{lca}}(v_l)$. Therefore, setting $x(v_l) = x^{\mathrm{lca}}(v_l)$ (which costs $\mathcal{C}(v_l,\sigma') = \mathrm{C}(v_l,\sigma')$) would imply that $\mathrm{D}^{\mathrm{sub}}(x(v),x(v_l)) = 1$. On the contrary, setting $x(v_l) \supsetneq x^{\mathrm{lca}}(v_l)$ (which costs $\mathcal{C}^\star(v_r,\sigma') = \mathrm{C}^\star(v_r,\sigma'')$) would give us the freedom to choose $x(v_l)$ such that $x(v) \subseteq x(v_l)$, implying that $\mathrm{D}^{\mathrm{sub}}(x(v),x(v_l)) = 0$. The same logic holds for v, v_r, and σ''. Note that each of the three cases in the innermost loop uses the same cost computation formula as Definition 5, albeit adapted to test all four options of setting $x(v_l)$ and $x(v_r)$ to either x^{lca} or a superset of it. By Lemma 5, those are the only four options to consider, so $\mathcal{C}^\star(v,\sigma) = \mathrm{C}^\star(v,\sigma)$.

Consider the computation of $\mathcal{C}(v,\sigma)$ by Algorithm 1. In that case, $x(v) = x^{\mathrm{lca}}(v)$, therefore setting $x(v_l) \supsetneq x^{\mathrm{lca}}(v_l)$ is only allowed if $x^{\mathrm{lca}}(v) \not\subseteq x^{\mathrm{lca}}(v_l)$, for otherwise there would be at least one gene family with two or more gain points. When setting $x(v_l) = x^{\mathrm{lca}}(v_l)$ (which costs $\mathcal{C}(v_l,\sigma') = \mathrm{C}(v_l,\sigma')$), the presence of a loss depends on the value of $\mathrm{D}^{\mathrm{sub}}(x^{\mathrm{lca}}(v),x^{\mathrm{lca}}(v_l))$. On the other hand, setting $x(v_l) \supsetneq x^{\mathrm{lca}}(v_l)$ (which costs $\mathcal{C}^\star(v_r,\sigma') = \mathrm{C}^\star(v_r,\sigma'')$) would give us the freedom to choose $x(v_l)$ such that $x(v) \subseteq x(v_l)$, implying that $\mathrm{D}^{\mathrm{sub}}(x(v),x(v_l)) = 0$. The same logic holds for v, v_r, and σ''. Note that this algorithm also follows Definition 5, handling all four options mentioned previously, and excluding disallowed cases. So $\mathcal{C}(v,\sigma) = \mathrm{C}(v,\sigma)$. □

Theorem 1 (Time and space complexity). *Using SuperDTL, the* DTL Super-reconciliation *problem can be solved in polynomial time $\mathcal{O}(|\mathcal{X}| \times |\Sigma|^3)$ and space $\mathcal{O}(|\mathcal{X}| \times |\Sigma|)$.*

Proof. Both dynamic programming tables have exactly $|\mathrm{V}(T)| \times |\mathrm{V}(S)|$ entries each. Computing a single entry of one of the tables takes time $\mathcal{O}(|\mathrm{V}(S)|^2)$, pro-

vided all the other required entries are made available in constant time and set and tree operations are implemented in an efficient way. A single bottom-up traversal of the synteny tree is enough to fill in both tables using Algorithms 1 and 2, therefore, computing the tables takes time $\mathcal{O}(|\mathrm{V}(T)| \times |\mathrm{V}(S)|^3)$. To compute the overall minimal cost is to compute the minimum value of the C table in column $\mathrm{r}(T)$, which can be done in time $\mathcal{O}(\mathrm{V}(S))$. To construct an optimal solution, pointers to optimal σ', σ'' pairs and to which of C and $\mathrm{C}^\star$ is used must be tracked for each table entry, taking up a constant time and space for each entry. After the tables have been computed, tracing back those pointers allows constructing a solution by following $\mathcal{O}(|\mathrm{V}(T)|)$ pointers. The overall time and space complexities are therefore $\mathcal{O}(|\mathrm{V}(T)| \times |\mathrm{V}(S)|^3)$ and $\mathcal{O}(|\mathrm{V}(T)| \times |\mathrm{V}(S)|)$, or, equivalently, $\mathcal{O}(|\mathcal{X}| \times |\Sigma|^3)$ and $\mathcal{O}(|\mathcal{X}| \times |\Sigma|)$. □