1-1-2011

The Cophylogeny Reconstruction Problem is NP-Complete

Yaniv J. Ovadia '10
Harvey Mudd College

Daniel Fielder '11
Harvey Mudd College

Chris Conow
California State Polytechnic University - Pomona

Ran Libeskind-Hadas
Harvey Mudd College

Recommended Citation

Y. Ovadia, D. Fielder, C. Conow, and R. Libeskind-Hadas, "The Cophylogeny Re- construction Problem is NP-Complete," Journal of Computational Biology, Vol. 18, No. 1, January 2011, pp. 59-65.

The Cophylogeny Reconstruction Problem is NP-Complete[*]

Y. Ovadia and D. Fielder and C. Conow and R. Libeskind-Hadas

September 7, 2009

1 Introduction

The cophylogeny reconstruction problem arises in the study of host-parasite relationships. Specifically, we are given a host tree H, a parasite tree P, and a function φ mapping the leaves (extant taxa) of P to the leaves of H. Four biologically plausible operations are considered: cospeciation, duplication, host switching, and loss (Figure 1). A host switch is permitted in conjunction with a duplication event but not with a cospeciation event [1].

A feasible solution is an extension of φ that maps each internal node of the parasite tree to a vertex or edge of the host tree and can be constructed using the four types of events. The objective of the cophylogeny reconstruction problem is to find one or more "optimal" solutions that reconcile the parasite tree and the host tree with respect to these operations. One notion of optimality simply assigns a cost to each of the four types of events and then seeks to minimize the total cost. Since it is often difficult to estimate appropriate costs for each of the four types of operations, an alternative approach is to find a Pareto optimal set of solutions [1]. In this case, a vector is associated with each solution where the entries indicate the number of cospeciation, duplication, loss, and host switch events, respectively. Let $v = (c, d, \ell, h)$ and $v' = (c', d', \ell', h')$ be the cost vectors of two solutions. We say that v is *strictly less than* v' if $c \leq c'$, $d \leq d'$, $\ell \leq \ell'$, and $h \leq h'$, and at least one of these relationships is a strict inequality. A solution is said to be in the Pareto optimal set if its cost vector is some v such that there is no solution with cost vector strictly less than v. The optimization version of the cophylogeny reconstruction problem is to find the maximal Pareto optimal set. The corresponding decision version of this problem, henceforth denoted CRDP (Cophylogeny Reconstruction Decision Problem) asks: For a given cost vector v, is there a solution whose cost vector is strictly less than v?

While practitioners have generally assumed for over a decade that this problem is computationally intractable, no rigorous proof of this conjecture has been found. Recently, Libeskind-Hadas and Charleston have made progress in this direction by showing that a slightly more general version of this problem is NP-complete [2]. Their proof assumes that the host phylogeny may be reticulate, meaning that it can contain cycles (corresponding to biological hybridization events) rather than a true tree. In addition, the proof assumes that ranges can be stipulated for the relative times of events in the two trees. These two assumptions are necessary in the proof in order to construct appropriate gadgetry. The problem of whether the classical cophylogeny reconstruction problem is NP-complete was, therefore, left open.

[*]This work was supported by the U.S. National Science Foundation under grant 0753306.

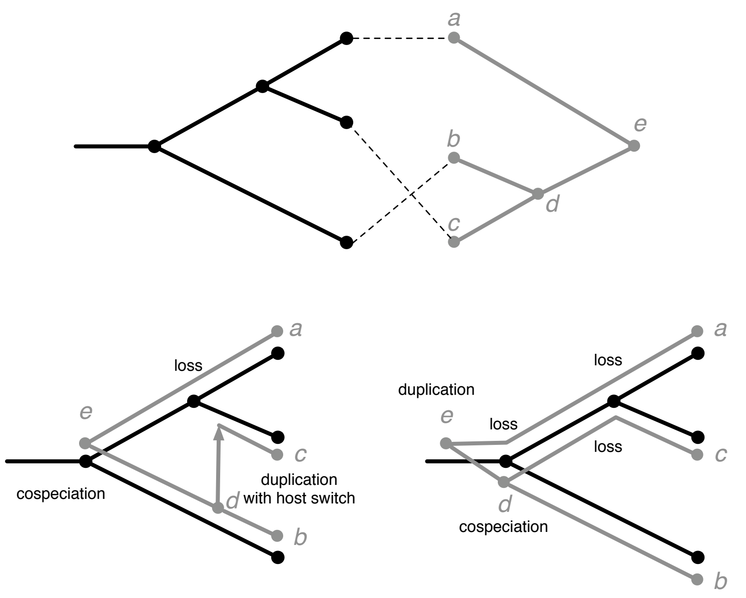

Figure 1: A simple tanglegram with host tree in black at left and parasite tree in gray on right. The associations φ between tips is shown in dotted lines. Below are two possible reconstructions that explain the relationship between H and P with events labeled.

In this paper, we prove that the decision version of the cophylogeny reconstruction problem is indeed NP-complete. The proof relies on some ideas introduced in Libeskind-Hadas and Charleston's proof for the reticulate timed problem, but is substantially more involved due to the fact that we can no longer exploit reticulation or timing.

2 Terms and Definitions

The following terms and definitions will be used throughout the remainder of the paper.

- $V(T)$ denotes the vertices in rooted tree T,
- $L(T)$ denotes the leaves or *tips* of rooted tree T,
- A host tree is a rooted regular binary tree with an additional edge (d, r) where d is called the "dummy root" and r is the original root of the regular binary tree. This is required in order to account for events in the parasite phylogeny that may predate the most recent common ancestor in the host phylogeny.

We now formally state the decision problem.

Definition 1 *An instance of the Cophylogeny Reconstruction Decision Problem (CRDP) is a 4-tuple* (H, P, φ, B) *where*

- *H and P are the rooted host and parasite trees,*

- *$\varphi : L(P) \to L(H)$ maps the tips of P to the tips of H,*
- *B is a 4-tuple (B_C, B_D, B_S, B_L) of upper bounds on the number of cospeciation, duplication, loss, and host switch events respectively.*

The decision question is: Does there exist a mapping Φ that extends φ and whose cost is strictly less than B?

A natural generalization of this problem allows an extant parasite to be mapped to some arbitrary number of tips in the host tree. This problem, called the *Generalized Cophylogeny Reconstruction Decision Problem (GCRDP)*, is stated as follows:

Definition 2 *An instance of the Generalized Cophylogeny Reconstruction Decision Problem (GCRDP) is a 4-tuple (H, P, φ, B) where*

- *H and P are the rooted host and parasite trees,*
- *$\psi : L(P) \to 2^{L(H)}$ maps the tips of P to sets of tips of H,*
- *B is a 4-tuple (B_C, B_D, B_S, B_L) of upper bounds on the number of cospeciation, duplication, loss, and host switch events respectively.*

The decision question is: Does there exist a mapping Ψ that extends ψ and whose cost is strictly less than B?

We first prove that GCRDP is NP-complete via a reduction from 3SAT. We then show an instance of GCRDP constructed in the reduction from 3SAT can be transformed into a corresponding instance of CRDP with the same answer, thus implying that CRDP is also NP-complete.

3 Polynomial Time Reduction of 3SAT to GCRDP

Given an instance of 3SAT with n variables $x_1, \dots, x_n$ and m clauses $C_1, \dots, C_m$ where n is a power of 2 and each clause contains at most a single instance of each variable[1], we construct an instance of GCRDP for which a solution exists if and only if a solution exists to the 3SAT instance.

A basic gadget in the reduction is the *k-thorn gadget*, illustrated in Figure 2, consisting of a path of length k where each vertex on the path, except the last, has one tip child. The last vertex on the path may have one tip child and a second child in another gadget or may have two tip children. The value of k will be determined later.

If y and z are two k-thorn gadgets where y is in the parasite tree and z is in the host tree, then we say that y associates with z if φ maps each tip of y to the corresponding tip of z or, in the case of GCRDP, if ϕ maps each tip of y to a set which contains the corresponding tip of z. Similarly, we say that a solution mapping Ψ maps thorn y to thorn z to mean the solution maps each vertex of y to the corresponding vertex in z.

In our reduction, variables in the 3SAT instance will be represented in the host tree while clauses will be represented in the parasite tree. The tip mapping function will be used to encode the literals that appear in each clause.

[1]It is easily seen that any instance of 3SAT can be expressed as an equivalent instance with these two properties using a simple padding of variables and clauses.

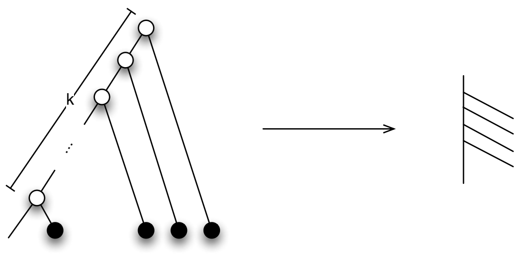

Figure 2: The k-thorn gadget is drawn to the left with filled-in tip vertices. Henceforth, we represent this symbolically as shown to the right.

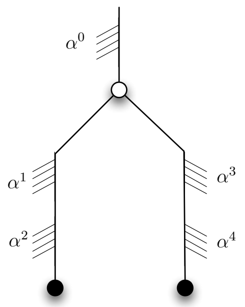

Figure 3: The assignment gadget is set to **true** if α^1 occurs prior to α^3 and is set to **false** otherwise.

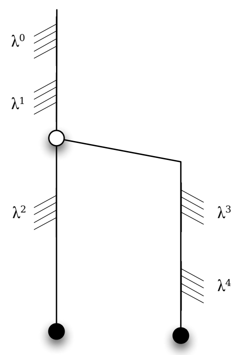

Figure 4: A generic literal gadget.

The primary gadget in the host tree is called an *assignment gadget*. There will be four such gadgets in the host tree for each variable in the 3SAT instance (only one of which determines the variable's assignment). This gadget, shown in Figure 3, begins with a k-thorn gadget α^0. This thorn's non-tip child has two descendant edges; one to thorn gadget α^1 followed by thorn gadget α^2, and the other which leads to α^3 followed by α^4. The final thorns of α^2 and α^4 have two tip children. This host tree gadget will represent the value **true** if α^1 occurs before α^3 and **false** otherwise. These relative times will be induced by the mapping of corresponding gadgets in the parasite tree, which we describe below.

The host tree, depicted in Figure 5, begins with a dummy root whose single child is a k-thorn τ^0 where $k = B_D + \sum_{l \in L(P)}(|\psi(l)| - 1) + 1$ (this value will be defined as $B'_D + 1$ later). This k-thorn leads to a vertex which initiates a sequence of bifurcations forming a balanced binary tree culminating with n vertices, one for each variable in the 3SAT instance. Each of these n vertices is the root of a k-thorn gadget τ^1_i of size $k = B_L + 1$. Each such thorn's non-tip child leads to a balanced binary tree with four children which serve as roots for assignment gadgets α_i, β_i, γ_i, and δ_i.

The primary gadget in the parasite tree is called a *literal gadget* and is depicted in Figure 4 Each clause in a 3SAT instance contains exactly three literals and thus there will be three literal gadgets per clause in the parasite tree. Let λ denote a literal gadget. λ begins with k-thorn gadget λ^0 which shares its root, followed by an additional thorn λ^1. The non-tip child of λ^1 is incident to one edge which leads to a k-thorn λ^2 followed by a tip, and another edge which leads to k-thorns λ^3 and λ^4 in series followed by another tip vertex.

The literal gadgets in the parasite tree will be mapped to assignment gadgets in the host tree using the tip mapping function ψ. In particular, this mapping will depend on whether a given literal, represented by a literal gadget in the parasite tree, appears unnegated or negated in a given clause. In other words, the tip mapping function will encode whether a given literal satisfies its clause by being **true** or **false**.

Henceforth, we will refer to a literal gadget from clause C_j that is the negated or unnegated form of variable x_i as $\lambda_{i,j}$.

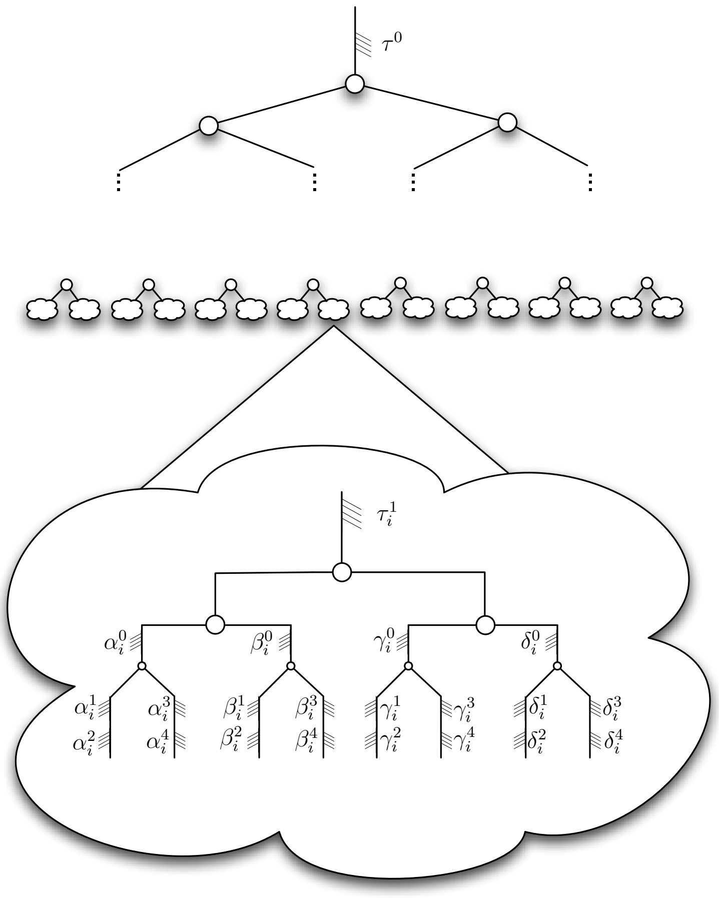

Figure 5: A sample host tree derived from a 3SAT instance. The balloon shows the group of four assignment gadgets for variable x_i.

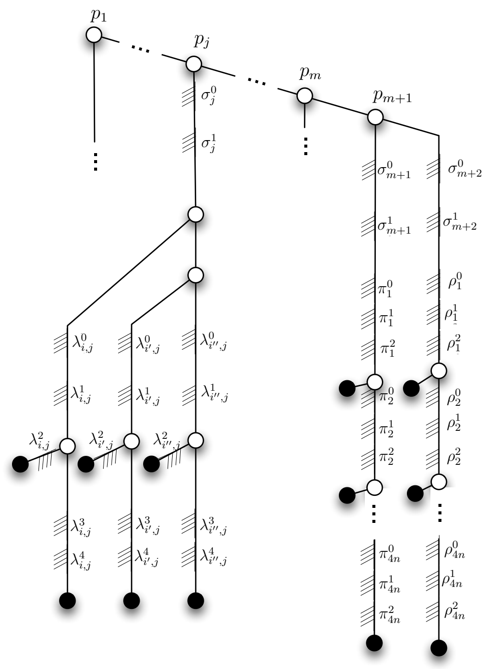

Figure 6: A sample parasite tree derived from a 3SAT instance. On the left is the gadgetry corresponding to clause C_j containing literals of variables $x_i, x_{i'}$, and $x_{i''}$.

3.0.1 Budget Tuple

The budget tuple $B = (B_C, B_D, B_S, B_L)$ is defined as follows

$$\begin{aligned}
B_C &= \infty \\
B_D &= 4m + 8n - 1 \\
B_S &= 3m + 8n - 2 \\
B_L &= (m+2)\log(n) + 4m + 8n + 4
\end{aligned}$$

3.0.2 Parasite Tree

The parasite tree, illustrated in Figure 6, begins with a path $(p_1, \ldots p_{m+1})$ where p_1 is the root. Henceforth, for $1 \leq j \leq m$, the jth parasite subtree refers to the subtree containing descendants of p_j's non-path edge. We arbitrarily label the subtrees containing descendants of p_{m+1}'s edges as the $(m+1)$th and $(m+2)$th parasite subtrees.

Each of the first m parasite subtrees correspond to one of the m clauses. Consider an arbitrary clause C_j and its corresponding subtree. The jth subtree begins with a thorn gadget of size $k = B'_D + 1$ labeled σ_j^0 followed by an additional k-thorn labeled σ_j^1 of size $k = B_L + 1$. A path of length 2 follows the non-tip child of σ_j^1 which is followed by three edges followed by literal gadgets labeled $\lambda_{i,j}, \lambda_{i',j}$, and $\lambda_{i'',j}$ each representing a literal contained in C_j.

The last two parasite subtrees do not correspond to clauses. Instead they serve to force an order between assignment gadgets as we will describe later. The branches lead to the two thorns $\sigma_{m+1}^0, \sigma_{m+1}^1$ and $\sigma_{m+2}^0, \sigma_{m+2}^1$ respectively where the 0th thorns are of size $k = B'_D + 1$, and the 1st is of size $k = B_L + 1$. These branches are then followed by n k-thorn triples ($k = B_L + 1$) in series $\pi_i^0, \pi_i^1, \pi_i^2$ and $\rho_i^0, \rho_i^1, \rho_i^2$ respectively for $1 \le i \le n$, and each triple except the last is followed by a vertex with one tip child.

3.0.3 Tip Mappings

We begin by providing definitions of how literal gadgets are mapped to assignment gadgets in negated and unnegated forms. This, in turn, will be used to describe the mapping function ψ.

If we wish to associate λ with α in unnegated form, then the tips are mapped as follows where each parasite thorn, host thorn pair indicates that the parasite thorn is associated with the host thorn.

$$(\lambda^0, \alpha^0), (\lambda^1, \alpha^1), (\lambda^2, \alpha^2), (\lambda^3, \alpha^3), (\lambda^4, \alpha^4)$$

If, instead, we wish to associate λ with an assignment gadget α in negated form, then we use the following set of associations.

$$(\lambda^0, \alpha^0), (\lambda^1, \alpha^3), (\lambda^2, \alpha^4), (\lambda^3, \alpha^1), (\lambda^4, \alpha^2)$$

Figure 7, which depicts the intended solution mapping of a literal gadget onto an assignment gadget, also serves to depict the tip mapping between these gadgets.

The parasite tip mappings are defined by the following:

- σ_j^0 is associated with τ^0 for $1 \le j \le m$.
- σ_j^1 is associated with τ_i^1 for all i, j where $x_i \in C_j$ or $\overline{x_i} \in C_j$.
- The $(m+1)$th branch maps its sequence of thorns to the 0th, 1st, and 2nd thorn of each assignment gadget as described by the following parasite thorn, host thorn associations (for $1 \le i \le n$):

$$(\pi_{4i-3}^0, \alpha_i^0), (\pi_{4i-3}^1, \alpha_i^1), (\pi_{4i-3}^2, \alpha_i^2)$$
$$(\pi_{4i-2}^0, \beta_i^0), (\pi_{4i-2}^1, \beta_i^1), (\pi_{4i-2}^2, \beta_i^2)$$
$$(\pi_{4i-1}^0, \gamma_i^0), (\pi_{4i-1}^1, \gamma_i^1), (\pi_{4i-1}^2, \gamma_i^2)$$
$$(\pi_{4i}^0, \delta_i^0), (\pi_{4i}^1, \delta_i^1), (\pi_{4i}^2, \delta_i^2)$$

- Similarly, the $(m+2)$th branch's thorn tips map to the 0th, 1st, and 2nd thorn of each assignment gadget (for $1 \le i \le n$):

$$(\rho_{4i-3}^0, \alpha_i^0), (\rho_{4i-3}^1, \alpha_i^3), (\rho_{4i-3}^2, \alpha_i^4)$$
$$(\rho_{4i-2}^0, \beta_i^0), (\rho_{4i-2}^1, \beta_i^3), (\rho_{4i-2}^2, \beta_i^4)$$
$$(\rho_{4i-1}^0, \gamma_i^0), (\rho_{4i-1}^1, \gamma_i^3), (\rho_{4i-1}^2, \gamma_i^4)$$
$$(\rho_{4i}^0, \delta_i^0), (\rho_{4i}^1, \delta_i^3), (\rho_{4i}^2, \delta_i^4)$$

- Recall that every k-thorn triple (except the last) of form $\pi^0_{4i-3}, \pi^1_{4i-3}, \pi^2_{4i-3}$ $(1 \leq i \leq n)$ is followed by a vertex with exactly one tip child. This tip maps to a singleton set containing the tip child of α^2_i which is not already mapped to by the last thorn in π^2_{4i-3}.
- Similarly, the tips following thorn triples of form π_{4i-2}, π_{4i-1}, and π_{4i} $(1 \leq i \leq n)$ map to the tip children of β, δ, and γ which are not already mapped to by the last thorn of the triple respectively.
- The prior two mappings apply similarly for the $(m+2)$th parasite subtree except the assignment gadget thorns use superscripts $0, 3, 4$ in place of $0, 1, 2$.
- For all $1 \leq i \leq n$ and $1 \leq j \leq m$, where $x_i \in C_j$ we associate $\lambda_{i,j}$'s tips with α_i's tips in its unnegated form as defined earlier.
- For all $1 \leq i \leq n$ and $1 \leq j \leq m$, where $\overline{x_i} \in C_j$ we associate $\lambda_{i,j}$ with α_i in its negated form as defined earlier.
- For $1 \leq i \leq n$ and $1 \leq j \leq m$, if $\lambda_{i,j}$'s parent is the first vertex of the length 2 path following σ^1_j, then $\lambda_{i',j}$'s and $\lambda_{i'',j}$'s tips map to γ_i's and δ_i's tips respectively, where $\lambda_{i,j}, \lambda_{i',j}$, and $\lambda_{i'',j}$ are distinct literal gadgets of the jth parasite subtree, and $i' < i''$.
- For $1 \leq i \leq n$ and $1 \leq j \leq m$, if $\lambda_{i,j}$'s parent is not the first vertex of the path following σ^1_j, then its sibling literal gadget $\lambda_{i',j}$'s tips map to β_i, and the tips of $\lambda_{i'',j}$ map to γ_i.

3.1 Proof of Correctness

3.1.1 Forward Direction

Assume that the given 3SAT instance is satisfiable. Let $C_1, \ldots, C_m$ denote the m clauses, and let $x_1, \ldots, x_n$ denote the n variables. We show the existence of a solution to the constructed GCRDP instance by describing a solution Ψ which incurs a cost tuple less than B. For convenience, define $s(j) \in \{1, \ldots, n\}$ $(1 \leq j \leq m+2)$ to be the index of a variable which satisfies clause C_j. If multiple variables satisfy C_j, let $s(j)$ be the least such index. We arbitrarily assign $s(m+1) = s(m+2) = 1$.

First we describe the mapping of a literal gadget onto an assignment gadget as shown in Figure 7. When we write that a literal gadget λ associates with an assignment gadget α, and ψ has been defined such that λ's tips are mapped to α's tips in unnegated form, the following is intended:

- For each of the following parasite thorn, host thorn pairs, we map the parasite gadget to the host gadget each at cost of $B_L + 1$ cospeciations.

$$(\lambda^0, \alpha^0), (\lambda^1, \alpha^1), (\lambda^2, \alpha^2), (\lambda^3, \alpha^3), (\lambda^4, \alpha^4)$$

- The edge between λ^0 and λ^1 incurs a loss as it passes over the non-tip child of α^0.
- The parent vertex of thorns λ^2 and λ^3 is mapped to the edge between α^1 and α^2 and incurs a duplication followed by a host switch onto the edge preceding α^3.

If ψ maps λ's tips to α's tips in negated form, then we apply the following modifications to the mapping above.

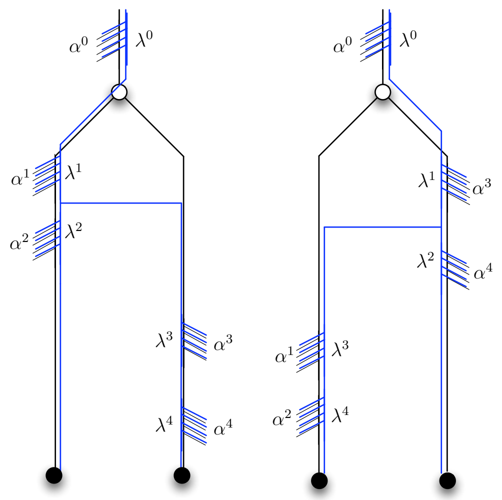

Figure 7: On the left, we show a literal gadget λ mapping onto an assignment gadget α in unnegated form. The mapping for negated form is shown to the right.

- The gadgets are mapped to each other according to the pairs

$$(\lambda^0, \alpha^0)(\lambda^1, \alpha^3)(\lambda^2, \alpha^4)(\lambda^3, \alpha^1)(\lambda^4, \alpha^2)$$

- The parasite tree incurs a loss by passing over the other child of the non-tip child of α^0.
- The duplication and host switch events occur on the edge between α^3 and α^4, and the switch lands on the edge prior to thorn α^1.

Regardless of form, this mapping costs $5(B_L + 1)$ cospeciations, 1 duplication, 1 loss, and 1 host switch.

First, let all vertices of the path $(p_1, \ldots, p_{m+2})$ map to the host edge which precedes τ^0 incurring a cost of $m + 1$ duplications. For the jth parasite subtree $(1 \leq j \leq m + 2)$, we map the k-thorn σ_j^0 to τ^0, assign the child edge to the path through the binary subtree between τ^0 and $\tau_{s(j)}^1$, and map σ_j^1 to $\tau_{s(j)}^1$. This accumulates $(m + 2) \times (B'_D + 1 + B_L + 1)$ cospeciation, $m + 1$ duplication, and $(m + 2) \times (\log_2(n))$ loss events.

For $1 \leq j \leq m$, the length two path following σ_j^1 codiverges at vertices of the balanced binary subtree following $\tau_{s(j)}^1$ as depicted in Figure 8, incurring two cospeciations and one loss per parasite subtree associated with a 3SAT clause. The literal gadget $\lambda_{s(j),j}$ then maps to assignment gadget $\alpha_{s(j)}$. Since the given 3SAT instance is known to be satisfiable, no two values j and j' exist such that $i = s(j) = s(j')$ and $x_i \in C_j$ and $\overline{x_i} \in C_{j'}$. Thus we may apply the mapping defined earlier between literal and assignment gadgets and incur a cost tuple of $(5(B_L + 1), 1, 1, 1)$ for each of the m clauses (recall that this is the tuple we derived after describing how to map a literal gadget to an assignment gadget). The other two literal gadgets of the jth parasite subtree each map to one of $\beta_{s(j)}$, $\gamma_{s(j)}$, or $\delta_{s(j)}$ in unnegated form accumulating an additional cost of $(10(B_L+1), 2, 2, 2)$ for each clause.

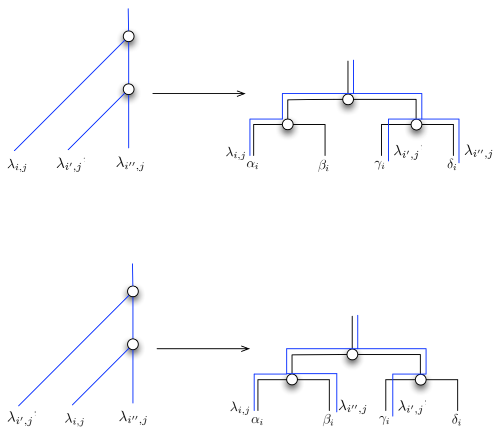

Figure 8: The upper figure depicts the mapping used if the satisfying literal for clause C_j is a child of the first vertex in the length two path. The lower figure illustrates the mapping if the satisfying literal is a child of the second vertex. Note that in both cases, we incur two cospeciations and one loss.

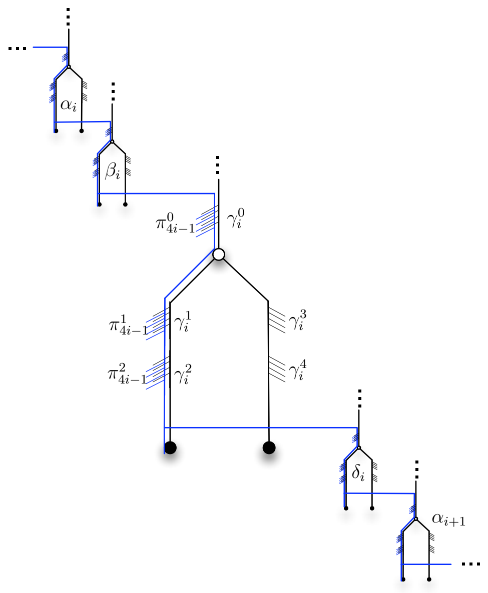

Figure 9: Here we depict the mapping of the $(m+1)$th parasite subtree along the k-thorns with superscripts 0, 1, and 2 of every assignment gadget. The expanded assignment gadget shows the mapping details onto γ_i. This is similar to the mapping for the $(m+2)$th parasite subtree which codiverges along k-thorns of superscripts 0, 3, and 4.

For the $(m+1)$th and $(m+2)$th parasite subtrees, we map the edge following σ_j^1 to the path between τ_1^1 and α_1^0 at a cost of 2 loss events for each subtree. The first k-thorn triple of the $(m+1)$th parasite subtree maps to the 0, 1, and 2 superscripted thorns of α_1, and the vertex which follows this triple host switches to the edge preceding assignment gadget β_1. This continues through assignment gadgets $\beta_1, \gamma_1, \delta_1, \alpha_2, \ldots, \delta_n$ as depicted in Figure 9. Since each of the two parasite subtrees has $4n$ k-thorn triples, and $4n-1$ vertices to following the triples in order to perform the host switches, an additional cost of $24n(B_L+1)$ cospeciation, $8n-2$ duplication, $8n$ loss, and $8n-2$ host switch events is incurred.

The total cost of this mapping is

$$(m+2)(B'_D+1) + (m+2)(B_L+1) + 15m(B_L+1) + 2m + 24nk \textbf{ cospeciations}$$

$$4m + 8n - 1 \textbf{ duplications}$$

$$(m+2)\log n + 4m + 8n + 4 \textbf{ losses}$$

$$3m + 8n - 2 \textbf{ host switches}$$

Since this cost is within the budget tuple B, it follows that any instance of GCRDP constructed from a satisfiable 3SAT instance has a valid solution.

3.1.2 Reverse Direction

Now we will show that the existence of a valid solution to an instance of GCRDP describing an instance of 3SAT implies that the 3SAT instance is satisfiable.

First note that every parasite tree k-thorn must codiverge at least once since every such thorn is made up of at least $B_L + 1 > B_D + 1$ vertices. It follows that for every $1 \le j \le m+2$, Ψ must map at least one vertex of σ_j^0 to its corresponding vertex on τ^0 yielding a cospeciation event, and since the vertices, $p_1, \ldots, p_{m+1}$ are ancestors of σ_1^0, they must incur $m+1$ duplications.

Consider the $(m+1)$th and $(m+2)$th parasite subtrees of the parasite tree. In order for a solution to map any non-k-thorn vertex of these branches as a cospeciation event, we must map it to a common ancestor of two assignment gadgets. However, such a mapping would exceed the loss budget as the edge from this vertex to its tip child must work through one side of an assignment gadget at a cost of $3 \times (B_L + 1)$ losses. Thus each of these $2 \times (4n-1)$ vertices must duplicate and host switch. The purpose of this mapping is to force an ordering between all assignment gadgets such that given any pair of assignment gadgets in the host tree, the k-thorns of one must occur before all k-thorns of the other.

Note that we have now accounted for all but $3m$ duplications, and all but $3m$ host switches. We claim that any solution must incur at least one duplication and host switch as it maps each literal gadget to the host tree. Any mapping of a literal gadget $\lambda_{i,j}$ which incurs no duplication or host switch events must map the k-thorn $\lambda_{i,j}^0$ as a cospeciation event to $\alpha_{i'}^0$, $\beta_{i'}^0$, $\gamma_{i'}^0$, or $\delta_{i'}^0$ for some i' possibly equal to i. Without loss of generality, we will assume that the assignment gadget is labeled $\alpha_{i'}^0$. In order to avoid host switches we must map all vertices of $\lambda_{i,j}$ to vertices of $\alpha_{i'}$. Now consider the parent of $\lambda_{i,j}^2$ and $\lambda_{i,j}^3$ which we will refer to as w. A solution which maps this vertex as a cospeciation must map it to the parent of $\alpha_{i'}^1$ and $\alpha_{i'}^2$, but such a solution requires that w's child lineage which contains $\lambda_{i,j}^2$ incur at least $B_L + 1$ loss events. Thus any valid solution must duplicate and host switch at least one vertex of each $\lambda_{i,j}$, and since we have used all but $3m$ duplications and host switches, each literal gadget duplicates and host switches *exactly* once.

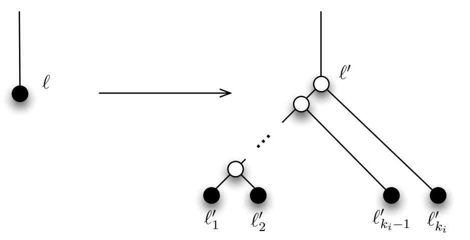

Figure 10: Here we show a sample transformation of a parasite tip vertex to a disjunctive tip gadget.

We have now accounted for all host switches and duplications, so Φ must map all remaining parasite vertices as cospeciation events. This implies that each vertex of a σ_j^1 k-thorn gadget must map to its corresponding vertex on some τ_i^1 as a cospeciation, and that the length 2 path which follows σ_j^1 must codiverge with vertices of the balanced tree following τ_i^1.

Consider some $\lambda_{i,j}$ which, without loss of generality, maps to an assignment gadget $\alpha_{i'}$. Recall that any attempt to codiverge the parent vertex of $\lambda_{i,j}^2$ and $\lambda_{i,j}^3$ will exceed the loss budget, so this vertex must duplicate and host switch while all other vertices of the literal gadget codiverge with vertices of α_i.

Furthermore, we claim that this host switch cannot land outside of the assignment gadget $\alpha_{i'}$. Recall that the solution mapping of the $(m+1)$th and $(m+2)$th parasite subtrees imply that for any pair of assignment gadgets in the host tree, all non-tip vertices of one gadget must occur before all non-tip vertices of the other. Thus a host switch which lands outside of α_i will place a switched lineage with $2 \times (B_L + 1)$ internal vertices onto an edge that either precedes an assignment gadget where any mapping will incur at least $B_L + 1$ loss events or onto an edge which precedes a tip vertex resulting in an excess of duplication events. Since the switched parasite lineage will contain 2 k-thorns of size $k = B_L + 1$, the host switch must land on the edge preceding either $\alpha_{i'}^1$ or $\alpha_{i'}^3$ depending on ψ's definition.

Note that this host switch requires that either all vertices of $\alpha_{i'}^1$ occur prior to all vertices of $\alpha_{i'}^3$ or that the reverse ordering occurs. If this condition cannot be met, it follows that no satisfactory mapping exists to solve the GCRDP instance. Thus for any solvable instance, a solution exists for the associated 3SAT instance which we can obtain by assigning x_i to true if α_i^1 occurs prior to α_i^3 and to false if we find the reverse ordering. To prove that this solution will satisfy the 3SAT clauses, consider a clause C_j. Note that ψ is defined such that the property noted earlier where the literal gadgets of a single clause must map to the assignment gadgets of the same variable implies that some gadget $\lambda_{i,j}$ exists which maps to α_i. Since the solution mapping is valid, an ordering is imposed between α_i^1 and α_i^3 which assigns the variable x_i appropriately such that C_j is satisfied. This is the case for all C_j and therefore a solution to the GCRDP instance implies the existence of an assignment which satisfies the 3SAT clauses.

4 Polynomial Time Reduction of GCRDP to CRDP

We now show that an instance of GCRDP constructed from a 3SAT instance, as described above, can be transformed into an instance of CRDP for which a solution exists if and only if the DCRP instance is solvable.

Given an instance of GCRDP (H, P, ψ, B) corresponding to a 3SAT instance, we construct a CRDP instance (H', P', φ, B') as follows. The transformation leaves the host tree unchanged so $H' = H$. The parasite tree is modified, as illustrated in Figure 10 such that each tip $\ell \in L(P)$ is replaced with a *disjunctive tip gadget*. This gadget is similar in form to a k-thorn of size $|\psi(\ell)| - 1$ all of whose children are tips labeled $\ell'_1, \ell'_2, \ldots, \ell'_{|\psi(\ell)|}$ in order of decreasing depth, and whose root we label ℓ'. The tips are mapped by φ to a unique element in the set $\psi(\ell)$, and we stipulate that $\varphi(\ell'_1)$ and $\varphi(\ell'_2)$ are the most distant pair of host vertices in $\psi(\ell)$ with respect to distance in the host tree.

Finally, we define

$$B' = (B_C, B_D + \sum_{\ell \in L(P)} (|\psi(\ell)| - 1), B_S + \sum_{\ell \in L(P)} (|\psi(\ell)| - 1), B_L).$$

4.1 Proof of Correctness

Given a solution, Ψ, for a GCRDP instance (H, P, ψ, B), we construct a solution to the CRDP instance (H', P', φ, B') as follows. Let $\Phi(p) = \Psi(p)$ for all $p \in P \cap P'$. For every disjunctive tip gadget constructed to replace a tip $\ell \in L(P)$, we map the vertices of the unique path from ℓ' to ℓ'_i to the edge preceding h_i where $h_i = \Psi(\ell)$, $\varphi(l'_i) = h_i$, and every internal vertex v of the disjunctive tip gadget host switches to the edge preceding some tip $h_{i'}$, where some descendant of v maps to $h_{i'}$. The mapping of vertices $p \in P \cap P'$ accumulates a cost of at most (B_C, B_D, B_S, B_L) while the disjunctive tip gadgets add $\sum_{\ell \in L(P)}(|\psi(\ell)| - 1)$ duplications and the same number of host switches. The total cost is upper bounded by B', and thus there exists a solution to the CRDP instance.

Now we demonstrate that the transformation above yields a solvable instance of CRDP only if the transformed GCRDP instance is solvable. First we show that every internal vertex of a disjunctive tip gadget, which takes the place of a tip other than a thorn tip of some σ^1_j, must be mapped as a duplication and host switch event. It will then be demonstrated that the vertices $p_1, \ldots, p_{m+1}$ must be mapped to the edge child of the host dummy root at a cost of $m + 1$ duplications, and finally, that the solution must map each σ^1_j to some τ^1_i as cospeciation events. With these observations, we then show that defining $\Psi(p) = \Phi(p)$ for $p \in P \cap P'$ and $\Psi(\ell) = \Phi(\ell')$ for $\ell \in L(P)$, will yield a valid solution to the GCRDP instance.

Consider an internal vertex v of a disjunctive tip gadget, which replaces a tip that is not part of some σ^1_j. Note that in our definition of ψ, no two vertices exist together in some $\psi(\ell)$ which are both descendants of the same τ^1_i k-thorn gadget, so in order to map v such that only a cospeciation is incurred, it must be mapped to an ancestor of two gadgets of form τ^1_i which would incur $B_L + 1$ losses yielding an invalid solution. Thus it follows that these vertices, of which there are

$$\sum_{l \in L(P) - \{\text{thorn tips of some } \sigma^1_j\}} (|\psi(\ell)| - 1)$$

must each incur a duplication and host switch. Furthermore, for any such mapping, it is always desirable to perform these host switches on edges which precede tips in order to avoid loss events, which yields a mapping similar to that described earlier.

We then note that in order to map a vertex of the initial parasite tree path, p_j, to a descendant of τ^0, σ^0_j must duplicate and host switch all of its thorns in order to ensure that its tips properly map with the tips of τ^0. Since we specified that $k = B'_D + 1$ for the k-thorns, τ^0 and all σ^0_j

$(1 \leq j \leq m+2)$, host switching all thorns of any σ_j^0 will incur too many duplications for even the CRDP instance, and therefore the path $(p_1, \ldots, p_{m+1})$ must accumulate $m+1$ duplications.

Subtracting the host switch and duplication events due to the disjunctive tip gadgets below thorns of form σ_j^1 and the length $m+1$ path, the remaining cost budget for both host switches and duplications is

$$3m + 8n - 2 + \sum_{\ell \in \{\text{thorn tips of some } \sigma_j^1\}} (|\psi(\ell)| - 1).$$

Since every host switch requires an accompanying duplication, any additional duplication which does not incur a host switch is no less expensive than a duplication followed by a host switch.

If we map each σ_j^1 to some τ_i^1 $(1 \leq i \leq n$ and $1 \leq j \leq m+2)$ and host switch the disjunctive tip gadgets as described earlier, then each thorn contributes a cost of two duplications and host switches. Now if we consider any mapping of a thorn vertex in σ_j^1 which avoids this host switch, we will show that such a mapping incurs an additional duplication and some number of loss events. Let u and v be the two internal vertices of the disjunctive tip gadget associated with a thorn tip of σ_j^1 where v is a descendant of u, let ℓ'_1, ℓ'_2, and ℓ'_3 be the disjunctive gadget's tips in order of increasing depth, and label $\varphi(\ell'_k) = h_k$ $(k = 1, 2, 3)$. In order to avoid a single host switch at u or v, Φ must map this vertex to a common ancestor vertex or edge of h_1 and h_2. Mapping to the latest common ancestor would yield a cospeciation, but the parent of u (an internal vertex of the k-thorn gadget) would have to incur a duplication (else exceed B_L losses) while an earlier common ancestor would yield an extra duplication. In addition to this, the edges (v, ℓ'_1) and (v, ℓ'_2) would each incur at least one loss at the parents of h_1 and h_2. To avoid a host switch at u, the vertex must map to a common ancestor of h_1, h_2, and h_3, but recall that ℓ'_1 and ℓ'_2 are defined such that h_1 and h_2 are pairwise the most distant vertices in the image of this disjunctive tip gadget's tips. It follows that the common ancestor of h_1, h_2, and h_3 is the same as the common ancestor of h_1 and h_2, so if we wish to save host switches at u and v, we cannot codiverge at u, and must incur an extra duplication in addition to those accumulated by avoiding the host switch at v.

Thus, the proposed solution, Ψ, to the GCRDP instance solution, where $\Psi(p) = \Phi(p)$ for $p \in P \cap P'$ and $\Psi(\ell) = \Phi(\ell')$ for $\ell \in L(P)$ will incur a cost within the budget tuple B. Furthermore, a valid CRDP instance solution requires that $\Phi(p) = h$ implies that there exists a tip descendant p' of p and a tip descendant h' of h such that $\varphi(p') = h'$, so it follows that Ψ is valid in that it properly extends the tip mapping function ψ.

References

[1] Michael Charleston. Jungles: A new solution to the hostparasite phylogeny reconciliation problem. *Mathematical Biosciences*, 149:191–223, 1998.

[2] Ran Libeskind-Hadas and Michael Charleston. On the computational complexity of the reticulate cophylogeny reconstruction problem. *Journal of Computational Biology*, 16(1):105–117, 2009.