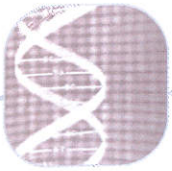
1-1-2010

# Figs, Wasps, Gophers, and Lice: A Computational Exploration of Coevolution

Ran Libeskind-Hadas
*Harvey Mudd College*



CHAPTER TWELVE

# Figs, wasps, gophers, and lice: a computational exploration of coevolution

**Ran Libeskind-Hadas**

This chapter explores the topic of coevolution: the genetic change in one species in response to the change in another. For example, in some cases, a parasite species might evolve to specialize with its host species. In other cases, the relationship between two species may be mutually beneficial and coevolution may serve to strengthen the benefits of that relationship.

One important way to study the coevolution of species is through a computational technique called *cophylogeny reconstruction*. In this technique, we first obtain the evolutionary (phylogenetic) trees for the two species and then try to map one tree onto the other in the "simplest" (most parsimonious) possible way. We can then use these mappings to determine how likely it is that the two species coevolved.

This chapter begins with descriptions of several pairs of species that are believed to have coevolved: figs and the wasps that polinate them; gophers and the lice that infest them; and a bird species that "tricks" another species to tend to its young. Next, we describe the cophylogeny reconstruction problem, its computational complexity, and a technique for finding good solutions for this problem. Finally, the reader is invited to use this computational method – through a freely accessible software package called Jane – to investigate the relationships between the pairs of species described at the beginning of the chapter.

*Bioinformatics for Biologists*, ed. P. Pevzner and R. Shamir. Published by Cambridge University Press.

# 1 Introduction

I can understand how a flower and a bee might slowly become, either simultaneously or one after the other, modified and adapted in the most perfect manner to each other, by the continued preservation of individuals presenting mutual and slightly favourable deviations of structure. (Charles Darwin, *The Origin of Species*)

The prescient thought experiment that Darwin describes in *The Origin of Species* is, in fact, borne out in bees and flowers (as documented in the book *The Sex Life of Flowers* [1]). One particularly interesting example is the symbiotic relationship between figs, their tiny flowers, and the miniature wasps that pollinate them.[1]

The story goes something like this. The flowers or "florets" of a fig are in its interior and are protected by the fig's thick membrane. Pollinating a fig is a real challenge! However, each fig species has a species of wasp (usually just one species, but sometimes more) that pollinates it. When a female wasp of the right species finds a fig that she likes, she tunnels into the interior, generally losing her wings in the process. Once inside, she lays her eggs on some of the tiny interior flowers, and, in the process, pollinates the fig. As the host fig develops, the wasp eggs hatch and the larvae feed on the fig tissue. After several weeks, the wasps reach maturity. The wingless males have a short life with only two objectives: they mate with the females and then burrow holes to help the females escape from the fig. The males then die inside the fig and the females fly off in search of their own fig homes to repeat the reproductive cycle. This bizarre story is true [2, 3] and not merely a *fig*ment of our imagination!

Biologists refer to the genetic change of one species in response to the change in another as *coevolution*. In the case of figs and wasps, the coevolution is known as *mutualism* since the two species are mutually dependent on one another for their survival. While there are several hundred varieties of figs (*Ficus*) and fig wasps, many pairs of fig and wasp species have become highly specialized to one another over approximately 60 million years of evolution.

Coevolution is not always mutually beneficial. For example, there are a variety of species of pocket gophers and an equal variety of lice that have specialized to their particular gopher hosts. This form of coevolution, known as *parasitism*, is a sort of evolutionary war: the gophers have evolved to defend themselves from the parasitic lice and the lice have evolved along with them to defeat their hosts' defenses [4].

A truly bizarre form of parasitism arises between finches from the family Estrildidae and another family commonly known as indigobirds [5, 6]. Each species of indigobird has evidently specialized to exploit a specific finch host species. The parasitic indigobirds very slyly lay their eggs in the nests of the host finches. The indigobird eggs look

[1] Wasps are not bees, but they are in the same order called Hymenoptera.

virtually identical to the corresponding host finch eggs and the juvenile indigobirds have markings and begging behaviors that are nearly identical to those of their finch nestmates. In this way, the parasitic indigobirds trick the host finches into caring for their eggs and feeding their young!

Finally, an urgent and compelling case of parasitism is the evolution of HIV. Studies of the evolutionary history of HIV indicate that it has close relatives including SIV (simian immunodeficiency virus) that infects non-human primates and FIV (feline strains) that infects cats. Interestingly, SIV and FIV do not appear to have deleterious effects on their hosts. By understanding the relationships between these different parasite viruses and their human, non-human primate, and feline hosts, researchers hope to develop better treatments and, ultimately, vaccines against HIV [7].

Indeed, there are countless cases of coevolution that have been studied, both of mutually beneficial and parasitic types. How do biologists determine whether two taxa coevolved and, if there is evidence that they did, what did that coevolution look like? This is known as the *cophylogeny problem* and is the topic of this chapter.

## 2 The cophylogeny problem

While we will soon examine figs and wasps, gophers and lice, and finches and indigobirds, let's begin with a simpler case of contrived taxa that we'll call Groodies and Cooties. (Google "Purves Groody" to learn about Groodies.)

Imagine that biologists have observed that Cooties are parasites of their Groody hosts and have constructed evolutionary histories, or *phylogenetic trees*, for Groodies and similarly for Cooties as shown in Figure 12.1.[2] The Groody tree is shown in black on the left and the Cootie tree is shown in blue on the right. From now on, we'll refer to one of the trees as the *host tree* (the Groody tree, in our example) and the other the *parasite tree* (the Cootie tree in this case).

The *nodes* in a tree represent hypothesized ancestral species. The end nodes, or "tips," of each tree represent the currently living, or *extant*, species. In Figure 12.1, we've given these names Groody 1 through 4 and Cootie 1 through 4. All the other nodes in the trees represent hypothesized species, named $X$, $Y$, $Z$ in the Groody tree and $x$, $y$, $z$ in the Cootie tree. More precisely, those nodes represent speciation events when the hypothesized ancestral species divided into two new species. Therefore, an *edge* in the tree can be thought of as the lifetime of the species with the node at the end

[2] The construction of phylogenetic trees is itself a fascinating and important field in computational biology, but here we'll assume that the phylogenetic trees have already been constructed using one of several known techniques.

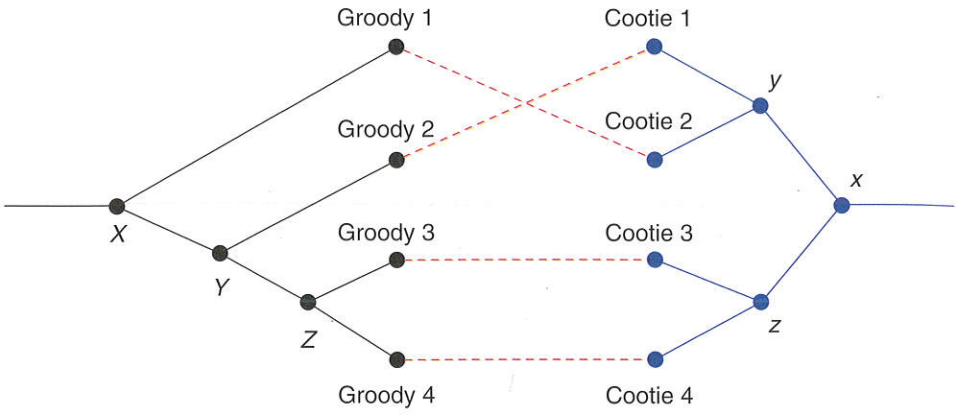


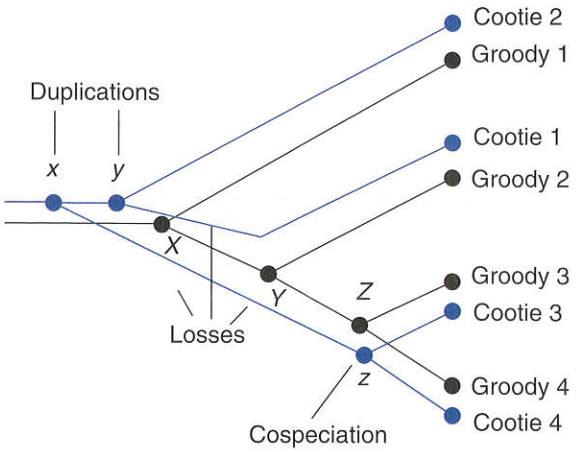

**Figure 12.1** A tanglegram for Groodies and Cooties.

of that edge indicating the speciation event. Finally, the associations between the tips of the Groody and Cootie trees are indicated by dotted lines. A figure like this showing two phylogenetic trees and the associations between their tips is called a *tanglegram*.

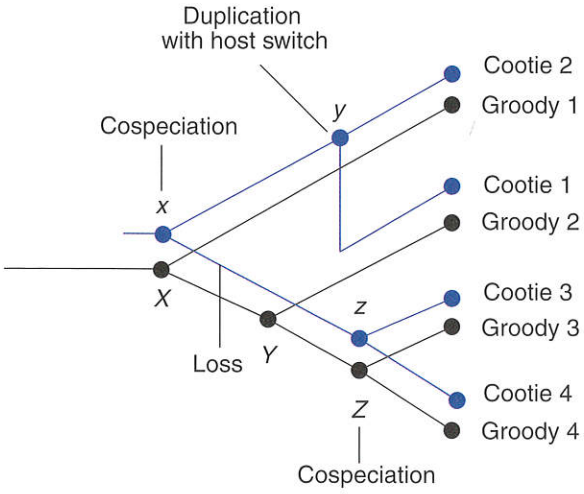
You might expect that coevolution should imply that the Groody and Cootie trees are exactly identical. However, such perfect congruence almost never happens even for species that have coevolved. Figure 12.2(a) and (b) show two possible ways in which the species might have coevolved. In each case, the Cootie tree in blue is superimposed on the Groody tree in black. Each of these is called a *reconstruction* since it attempts to reconstruct the histories of the two species.

In the reconstruction in Figure 12.2(a), we see that Cootie speciation event $z$ occurs at the same time as Groody event $Z$. This is called a *cospeciation* event and corresponds to two lineages speciating contemporaneously. For example, consider a species of louse living on a species of gopher. Imagine that the gopher species becomes geographically distributed with one population living in a warmer climate and another in a colder climate. Eventually, the gopher species splits into two new species, one with short hair and one with thick long hair adapted for the colder climate. The parasitic louse species may also split to specialize to the two new species of gophers – one new louse species may adapt to the short-haired gophers and the other to the thick long-haired gophers. In general, if two species coevolved, we would expect to see a significant number of cospeciation events between their two phylogenetic trees.

Notice that in Figure 12.2(a), events $x$ and $y$ in the Cootie tree occurred in the "prehistory" of the Groody species, that is, before the first inferred Groody speciation event. Speciation events in the Cootie tree that are not contemporaneous with speciation events in the host tree are called *duplications*. Duplications suggest that the Cootie speciation was independent of the Groody speciation, which does not contribute to




(a)




(b)

**Figure 12.2** Two possible reconstructions of the Cootie tree on the Groody tree.

evidence of coevolution of the two species. Finally, the edge from $y$ to Cootie 1 passes through $X$ and $Y$ as does the edge from $x$ to $z$. These are called *loss* events. Loss events may be due to a failure of the Cootie lineage to speciate, or there may have been a speciation but one of the lineages became extinct.

The reconstruction in Figure 12.2(b) suggests another possible way in which the two species may have coevolved with two cospeciation events ($x$ maps to $X$ and $z$ maps to $Z$), a loss event at $Y$, and a duplication event where $y$ occurs independently of a speciation event in the Groody tree. Another interesting thing happens here: one of the two descendant lineages from $y$ switches to a different part of the Groody tree.

This is called a *host switch*, or *horizontal transfer* event; such events are thought to be quite common in evolution. For example, it is known that one strain of HIV host switched from chimpanzees to humans sometime around the end of the nineteenth century [7].

There are many other possible reconstructions of these two phylogenetic trees and biologists would like to know which reconstructions, if any, are most plausible under the assumption that the two species coevolved. One approach is to estimate the relative likelihood of each of the four types of events (cospeciation, duplication, host switch, and loss) assuming coevolution has occurred and assign each such event a numerical "cost" so that likely events have low cost and unlikely ones have a higher cost. For example, cospeciation is a very likely event under the assumption that our two species coevolved, so the cost of this event might be 0 whereas duplication is a much less likely event and would therefore have some positive cost.

Now our objective becomes that of finding a reconstruction of minimum total cost under the given cost scheme. This is called the *cophylogeny reconstruction problem*. If there exists a reconstruction of very low cost, this gives strong evidence of coevolution. For example, imagine that cospeciation is assigned a cost of 0 and each duplication, host switch, and loss is assigned cost 1. Then, in the reconstruction in Figure 12.2(a), the total cost is 5 (2 duplications plus 3 losses), whereas in the reconstruction in Figure 12.2(b) the total cost is 3 (1 duplication, 1 loss, and 1 host switch). You might be wondering if there is a better reconstruction for the Groody and Cootie trees. The answer is yes, there is a reconstruction of cost 2 and you might want to pause here to find it. (Note that event $x$ in the Cootie tree could be associated with something after $X$ in the Groody tree. Moreover, the edge leading into $x$ is not considered to be involved in loss events because we have no putative ancestor for $x$.)

Imagine that we enumerated every possible reconstruction of the Groody and Cootie trees and, for each one, we computed its total cost. We then selected the reconstruction of minimum total cost. In our example, that cost is 2. How do we know whether that cost of 2 suggests coevolution? Certainly, if the cost had been 0, we'd probably feel pretty confident that there was coevolution here because that would mean that the two trees were identical. However, is a cost of 2 suggestive of that as well?

One way to find out is to use a basic idea in statistical hypothesis testing. Specifically, we can formulate the *null hypothesis* that the two phylogenies and the associations of their tips were random. Under this hypothesis, we'd like to measure the probability that there was a reconstruction of cost 2 or less. We can do so by writing a computer program that generates random pairs of trees and associations between their tips.[3]

[3] There is some controversy on the issue of what should be randomized in such tests. Generally, the host tree is not modified but the parasite tree is randomized. Another school of thought is that neither tree should be changed but only the associations between the tips should be randomized.

Next, we find the reconstruction of least cost and record that value. We repeat this computational experiment some large number of times, say 100 times. Imagine that we did this and discovered that for 96% of these random pairs, the cost of a minimum reconstruction was 3 or higher and in only 4% were the minimum costs 2 or less. In this case, we would say that the *p-value* is 0.04 because the probability of doing at least as well as 2, assuming that the trees were just random, is 0.04. If the $p$-value is low (typically less than or equal to 0.05), then we can reject the null hypothesis that the pairs of trees were simply random.

## 3 Finding minimum cost reconstructions

Our statistical hypothesis testing depends on our ability to solve the cophylogeny reconstruction problem. Moreover, once biologists are confident that a pair of species coevolved, they would like to see what minimum cost reconstructions look like to get a sense of some plausible ways in which the species coevolved.

Unfortunately, there are far too many different possible reconstructions for a pair of phylogenetic trees for us to enumerate them all. The number of possible reconstructions for two trees, each with $n$ tips, can be shown to be an exponential function of $n$. Just to get a sense of how bad that is, imagine that there were "only" $2^n$ possible reconstructions for a pair of host and parasite trees with $n$ tips each. (The actual number of reconstructions can be significantly larger than this!) If we have a pair of trees with 100 tips each (small relative to some of the trees that biologists would like to evaluate), we have $2^{100}$ reconstructions to evaluate. Even if we had a supercomputer capable of examining a billion reconstructions per second, it would take over 40 *trillion* years to examine them all! Considering that the sun will burn out in about nine *billion* years, this is very very bad news.

"Let's just wait a few years for faster computers; they should be able to do the job!" you might be thinking to yourself. Let's explore that for a moment. Under the very optimistic assumption that computers get twice as fast every year, waiting 20 years would result in computers that are about one million times faster than they are now. With such a fast computer we could solve the problem for trees with 100 tips in a mere 40 *million* years! In the off chance that this seems like a significant improvement, consider that if we increased the number of tips in the trees from 100 to 120, we'd be back to taking 40 *trillion* years to solve the problem, even with our super-fast futuristic computer. Considering that biologists have developed cophylogeny data sets in which the trees have over 200 tips, it appears that we're in serious trouble if we try to solve the problem this way. The moral of this story is that computational methods that

consider an exponential number of possibilities are useless for even relatively small phylogenetic trees.

For some computational problems, there are clever ways of finding the desired optimal solution without brute-force examination of every possible option. For example, you've probably used a program like Mapquest or Google Maps and asked for driving directions from one location to another. Those programs can find the shortest path between two locations without actually looking at every one of the large number of different paths. Computer scientists have found very clever algorithms that are absolutely guaranteed to find you a shortest path and the computation time is lightning fast.

It would be nice if this was possible for the cophylogeny reconstruction problem. Unfortunately, this appears to be very unlikely. The cophylogeny reconstruction problem was recently shown to be *NP-hard*, which essentially means that a fast algorithm for solving the cophylogeny reconstruction problem probably doesn't exist [8].

So what is to be done about the cophylogeny reconstruction problem? If the NP-hardness of the problem meant that there was absolutely no hope, then evolutionary biologists would be very disappointed and this chapter would be over. Fortunately, computational biologists have developed several strategies for solving the cophylogeny problem *reasonably well*. One approach is to try to use clever computational techniques to avoid examining certain reconstructions that can't be optimal. Professor Michael Charleston, at the University of Sydney in Australia, has developed a technique called *jungles* [9] that does exactly this. This approach still takes exponential time in many cases so it can only be used with relatively small trees. The technique has been implemented in a software tool called TreeMap [10].

Another approach is to use *heuristics*. A heuristic is a computational method that doesn't guarantee an optimal solution but foregoes optimality for efficiency. For example, Professors Daniel Merkle and Martin Middendorf at the University of Leipzig in Germany developed a very fast heuristic [11] used in a package called *Tarzan* [12]. (First there were jungles and then there was Tarzan.) Tarzan is known to find solutions that are not necessarily optimal and sometimes even finds solutions that don't quite make sense biologically (e.g. reconstructions that are impossible because they require a speciation event $x$ to occur before another speciation event $y$ but also for $y$ to occur before $x$, creating an irreconcilable inconsistency). Nonetheless, Tarzan often finds very good solutions and can handle very large phylogenetic trees.

We have recently developed a different kind of heuristic for cophylogeny reconstruction that uses a paradigm, called *genetic algorithms*, that computer science has borrowed from biology. The irony here is that we are trying to use computational methods to solve a biological problem but the computational method was one that computer scientists learned from biology! Unlike jungles, but like Tarzan, our approach does not guarantee optimal solutions. However, our approach is guaranteed to always produce

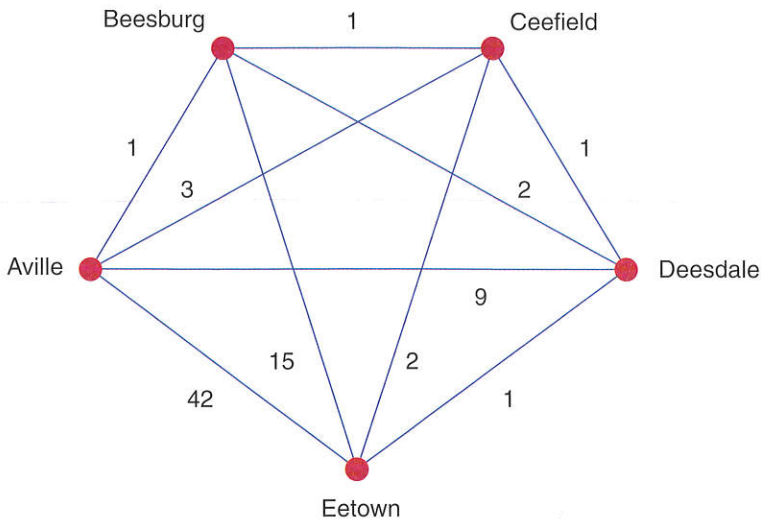



**Figure 12.3** Cities and flight costs.

good and biologically reasonable solutions in a reasonable amount of time. Continuing the jungles and Tarzan theme, our software is called *Jane*. In section 5, we explain how Jane works. Then, you'll have a chance to try it out for the fig/wasp, gopher/louse, and finch/indigobird relationships. In the meantime, you can download Jane from http://www.cs.hmc.edu/~hadas/jane.

## 4 Genetic algorithms

In this section we'll examine genetic algorithms. In the next, we'll see how Jane uses genetic algorithms to solve the cophylogeny problem. Finally, we'll use Jane to explore some real data in coevolution.

To explain the concept of a genetic algorithm – the key idea behind the Jane software – we now take a short aside to discuss a famous computational problem called the *Traveling Salesperson Problem*. The problem goes like this. Imagine that you are a salesperson who needs to travel to a set of cities to show your products to potential customers. The good news is that there is a direct flight between *every pair of cities* and, for each pair, you are given the cost of flying between those two cities. Your objective is to start in your home city, visit each city *exactly once*, and return back home. For example, consider the set of cities and flights shown in Figure 12.3 and imagine that your start city is Aville.

A tempting approach to solving this problem is to use an approach like this: starting at our home city, Aville, fly on the cheapest flight. That's the flight of cost 1 to Beesburg.

From Beesburg, we could fly on the least expensive flight to a city that we have not yet visited, in this case Ceefield. From Ceefield we would then fly on the cheapest flight to a city that we have not yet visited. (Remember, the problem stipulates that you only fly to a city once, presumably because you're busy and you don't want to fly to any city more than once – even if it might be cheaper to do so.) So now, we fly from Ceefield to Deesdale and from there to Eetown. Uh oh! Now, the constraint that we don't fly to a city twice means that we are forced to fly from Eetown to Aville at a cost of 42. The total cost of this "tour" of the cities is $1 + 1 + 1 + 1 + 42 = 46$. This approach is called a "greedy algorithm" because at each step it tries to do what looks best at the moment, without considering the long-term implications of that decision. This greedy algorithm didn't do so well here. For example, a much better solution that goes from Aville to Beesburg to Deesdale to Eetown to Ceefield to Aville has a total cost of $1 + 2 + 1 + 2 + 3 = 9$. In general, greedy algorithms are fast, but often fail to find optimal or even particularly good solutions.

It turns out that finding the optimal tour for the Traveling Salesperson Problem is very difficult. Of course, we could simply enumerate every one of the possible different tours, evaluate the cost of each one, and then find the one of least cost. However, there are a huge number (exponential or worse!) of different tours and this approach is not viable for even a moderate number of cities. Like the cophylogeny reconstruction problem, the problem is in the category of NP-hard problems – problems for which there is strong evidence that no fast algorithms exist. So, we are in the same predicament for the Traveling Salesperson Problem as for cophylogeny reconstruction.

Now for the clever idea that computer scientists borrowed from biology. Let's call the cities in Figure 12.3 by their first letters: $A$, $B$, $C$, $D$, and $E$. We can represent a tour by sequence of those letters in some order, beginning with $A$ and with each letter appearing exactly once. For example, the tour Aville to Beesburg to Deesdale to Eetown to Ceefield and back to Aville would be represented as the sequence $ABDEC$. Notice that we don't include the $A$ at the end because it is implied that we will return to $A$ at the end.

Now, let's imagine a collection of some number of orderings such as $ABDEC$, $ADBCE$, $AECDB$, and $AEBDC$. Let's think of each such ordering as an "organism" and the collection of these orderings as a "population." Pursuing this biological metaphor further, we can evaluate the "fitness" of each organism/ordering by simply computing the cost of flying between the cities in that given order.

Now let's push this idea one step further. We start with a population of organisms/orderings. We evaluate the fitness of each organism/ordering. Now, some fraction of the most fit organisms "mate," resulting in new "child" orderings where each child has some attributes from each of its "parents." We now construct a new population of such children for the next generation. Hopefully, the next generation will be more

fit – that is, it will, on average, have less expensive tours. We repeat this process for some number of generations, keeping track of the most fit organism (least cost tour) that we have found and report this tour at the end.

"That's a cute idea," we hear you say, "but what's all this about mating traveling salesperson orderings?" That's a good question – we're glad you asked! There are many possible ways we could define the process by which two parent orderings give rise to a child ordering. For the sake of example, we'll describe a very simple (and not very sophisticated) method; better methods have been proposed and used in practice.

Imagine that we select two parent orderings from our current population to reproduce (we assume that any two orderings can mate): $ABDEC$ and $ACDEB$. We choose some point at which to split the first parent's sequence in two, for example as $ABD|EC$. The offspring ordering receives $ABD$ from this parent. The remaining two cities to visit are $E$ and $C$. In order to get some of the second parent's "genome" in this offspring, we put $E$ and $C$ in the order in which they appear in the second parent. In our example, the second parent is $ACDEB$ and $C$ appears before $E$, so the offspring is $ABDCE$.

Let's do one more example. We could have also chosen $ACDEB$ as the parent to split, and split it at $AC|DEB$, for example. Now we take the $AC$ from this parent. In the other parent, $ABDEC$, the remaining cities $DEB$ appear in the order $BDE$, so the offspring would be $ACBDE$.

In summary, a genetic algorithm is a computational technique that is effectively a simulation of evolution with natural selection. The technique allows us to find good solutions to hard computational problems by imagining candidate solutions to be metaphorical organisms and collections of such organisms to be populations. The population will generally not include every possible "organism" because there are usually far too many! Instead, the population comprises a relatively small sample of organisms and this population evolves over time until we (hopefully!) obtain very fit organisms (that is, very good solutions) to our problem.

Just as evolution makes no promises that it results in optimally fit organisms, this technique cannot guarantee that the solutions that it finds will be optimal. However, carefully crafted genetic algorithms have been shown to find very good solutions to some very hard problems. Now, let's see how these ideas are used in Jane.

## How Jane works

Earlier, we noted that the cophylogeny reconstruction problem is computationally very hard; the only known approaches for solving this problem would take nearly an eternity. On the other hand, here's some good news: if we happen to know the order in which

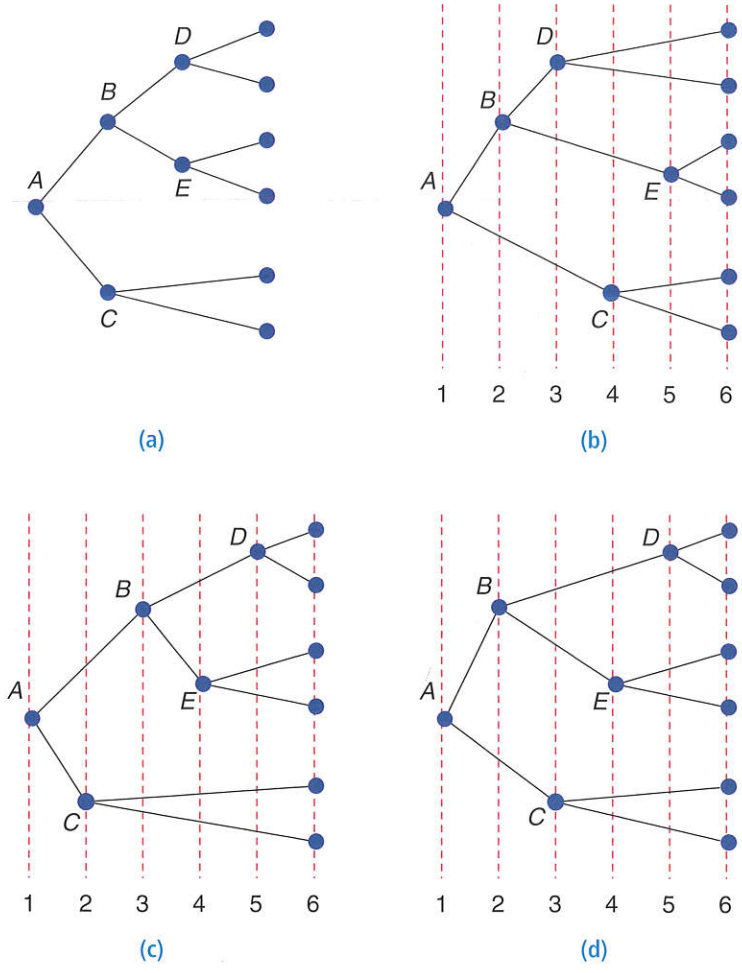



**Figure 12.4** (a) A host tree and three different possible orderings of the speciation events shown in (b), (c), and (d).

speciation events occurred in the host phylogeny, the problem turns out to be solvable very quickly!

What do we mean by the order of the speciation events? Consider the host phylogeny shown in Figure 12.4(a). Obviously, speciation event *A* occurred before speciation events *B* and *C*. Similarly, speciation event *B* occurred before speciation events *D* and *E*. However, which speciation event occurred first: *B* or *C*? Similarly, did *D* occur before *E*, or vice versa? There are many possible orderings for these events and three of them are shown in Figure 12.4(b), (c), and (d). Recall that we assume that all of the tips of the tree occur at the same time – that is, at current time.

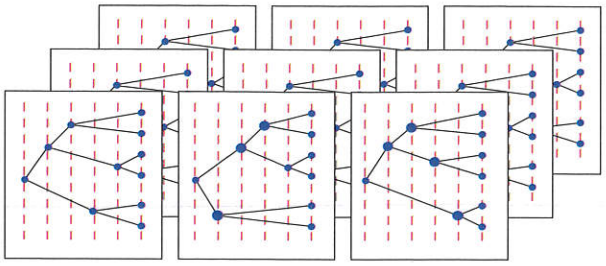
Surprisingly, if we happen to know the ordering of the speciation events in the host tree, even if we know nothing about the ordering of the events in the parasite tree, then we can find a least-cost solution in next-to-no-time using a clever computational technique called *dynamic programming* [8]. While we won't go into that technique here, it is one of the mostly widely used methods in computational biology. For example, sequence alignment, RNA folding, and various other computational biology problems can be solved using this technique. In the case of cophylogeny reconstruction, we can solve the problem in about one second (on a typical laptop computer) when the host and parasite trees have 100 tips each. That's fast!

"Wait a second!" we hear you exclaim. "Why does the ordering of the speciation events in the host tree matter at all?" Take a look again at Figures 12.4(c) and (d). In these figures let $(A, C)$ denote the edge from node $A$ to node $C$ and let $(B, E)$ denote the edge from node $B$ to node $E$. Notice that in the ordering shown in (c), speciation event $C$ occurs before speciation event $B$. Thus, a parasite that duplicates on edge $(A, C)$ cannot host switch to edge $(B, E)$ because $(A, C)$ ends before $(B, E)$ begins. On the other hand, in the ordering shown in (d), such a switch is possible because speciation event $c$ occurs after speciation event $B$ so edges $(A, C)$ and $(B, E)$ overlap in time. It might be that the best solution (the one that minimizes the total cost of the cospeciation, duplication, host switch, and loss events) requires a switch from $(A, C)$ to $(B, E)$, in which case the ordering in (c) might not be as "good" as the ordering in (d).

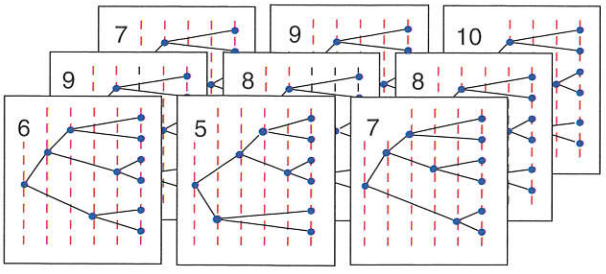
There's just one problem. How do we know the order in which the speciation events occurred in the host tree? If we're very lucky, we might have this information from the fossil record, but generally we will have little or no reliable information on the orderings of these events. Perhaps we could just try out all possible orderings of the host tree events and see which one permits us to find the best reconstruction of the parasite tree on the host tree? Unfortunately, there are way too many different orderings of the host (an exponential number, to be specific!), so that's totally impractical.

This is essentially the same problem that we had in the Traveling Salesperson Problem; there were too many possible orderings of the cities to explore them all. So, we used a genetic algorithm that kept a population that was a relatively small sample of the totality of all possible orderings and we artificially "evolved" better solutions.

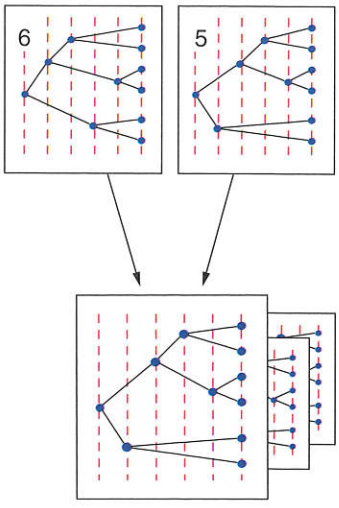
The Jane software package does exactly this for the cophylogeny reconstruction problem. It starts with a population comprising some relatively small population of random orderings of the speciation events in the host tree as illustrated in Figure 12.5(a). For each such ordering of events in the host tree, we use our very fast dynamic programming algorithm to find the best solution for reconstructing the parasite tree on the host tree with this particular ordering of events. The cost of the best solution can be thought of as the fitness for that ordering. Figure 12.5(b) shows the orderings scored



(a) The genetic algorithm maintains a population of "organisms," each of which is a different ordering of the events in the host tree.




(b) A very fast dynamic programming algorithm is used to find the best reconstruction of the parasite tree onto each of the orderings of the host tree. The cost of that reconstruction is used as the fitness of that ordering. Example fitness scores are shown in the upper left corner of each ordering.




(c) Two orderings are chosen at random, but biased in favor of orderings with lower cost (better fitness). These orderings are then "mated" to construct a new offspring ordering that maintains some properties of its parent orderings. This offspring ordering is placed into the population for the next generation.

(d) The parents are placed back into their mating population and the mating process is repeated until a new population of orderings of the desired size is constructed. We now go back to step (a) using this new generation as the mating population.

**Figure 12.5** The steps of the genetic algorithm used by Jane.

by their fitness. Keep in mind that in this context, a lower-cost solution is more fit than a higher-cost solution.

Next, we repeatedly choose pairs of orderings to "mate." While a pair of orderings is chosen at random, our random choice is biased to prefer more fit (lower-cost) orderings

to less fit (higher-cost) ones. That is, we tend to prefer orderings of the speciation events in the host tree that permit us to find better solutions. We mate that pair of orderings in some way, resulting in a new ordering that preserves some attributes from each of its two parent orderings.[4] The offspring is a new ordering of the host tree events that has some attributes from each of its two parent orderings. Our hope is that this new ordering of the speciation events in the host tree might be one for which there exists an even better solution. This is illustrated in Figure 12.5(c).

We repeat this process of constructing new offspring orderings until we've built a population of new orderings of some desired size. This is our next generation as illustrated in Figure 12.5(d). We now start all over again with this new population serving as the mating population. This process is iterated for a user-specified number of generations. At the end, we report the best solutions that were found during this evolutionary process.

## See Jane run

Now that we have an understanding of the computational challenge posed by the cophylogeny reconstruction problem, and the approach taken by Jane, let's try using Jane on some real cophylogeny data for figs and wasps and for gophers and lice. If you haven't done so already, download Jane from the website http://www.cs.hmc.edu/~hadas/jane. After you download it you can simply click on the the icon for that file and Jane will start up on your computer. From the Jane page, there is also a link that contains several example trees for you to download. One file is for figs and wasps, one is for pocket gophers and chewing lice, and one is for finches and indigobirds. You may also wish to read the Jane tutorial on the website, but the following is a self-contained demonstration of Jane in action.

Now click on Jane to start the program. You'll see the Jane window shown in Figure 12.6. In the "File" menu at the top of the Jane window, select "Open Trees" and find the Ficus-Ceratosolen.tree file that you downloaded from the Jane site. These are trees for figs and wasps that pollinate them. When the file loads, you'll see that the Jane window reports that the trees have 16 tips each. Notice that there are sliders in the Jane window that let you choose the "Number of Generations" (the number of generations of the genetic algorithm) and the "Population Size" (the number of tree orderings in each population maintained by the genetic algorithm). The defaults for both of these values are 30, which is fine for now. Click "Go" to start Jane running.

[4] We won't go into the details of the mating of orderings here, but if you're interested, you can find a detailed description online at [13].

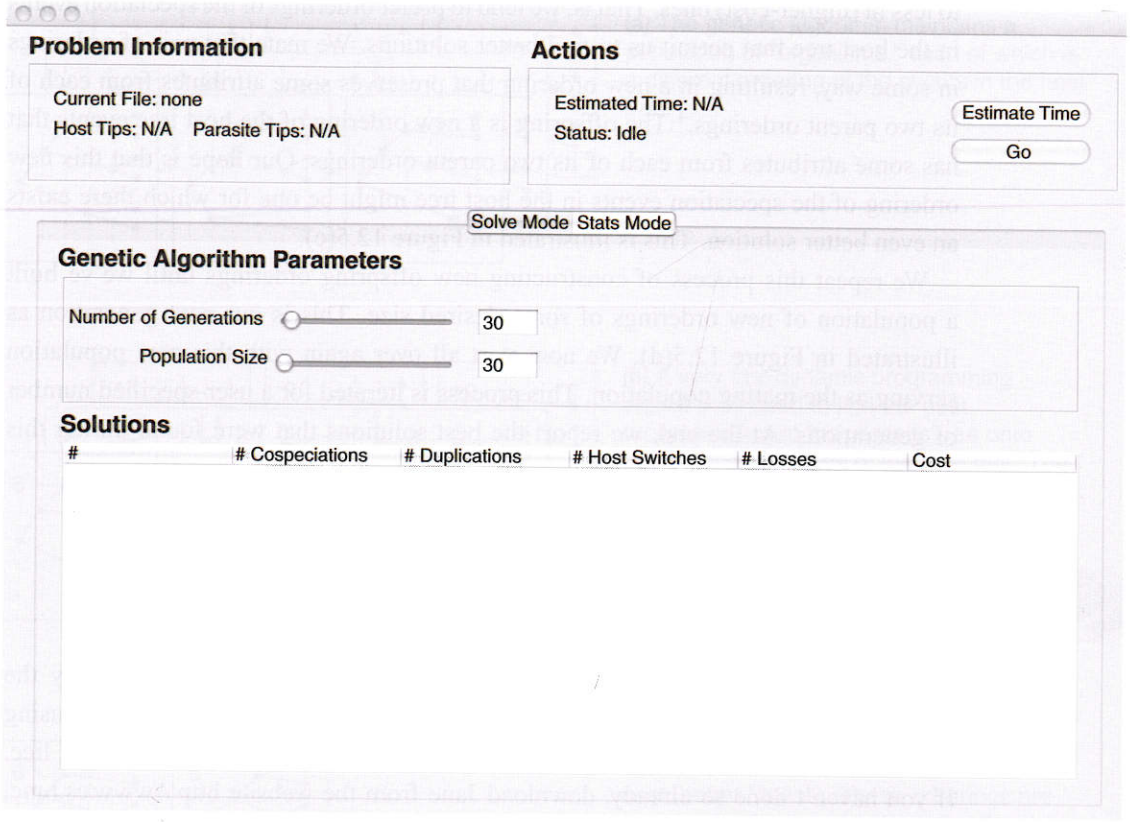



**Figure 12.6** The Jane window.

Within a second or so, Jane will complete the genetic algorithm and will display a list of solutions in the "Solutions" window. (Since there is some randomness employed in the genetic algorithm, you won't necessarily get exactly the same solutions that are shown here, nor will you necessarily get the same solutions each time you run Jane.) Jane presents you with a list of best solutions that it found along with their costs. By default, Jane assumes that cospeciations have cost 0, duplications and host switches have cost 1, and losses have cost 2. While these values have been used in many studies, biologists often try to infer appropriate relative values of these costs from other biological data. The values of these parameters can be changed in the "Settings" menu in Jane.

Coming back to our example, you can see that these solutions had 9 cospeciations, 12 duplications, 6 host switches, and 1 loss for a total cost of $9 \times 0 + 12 \times 1 + 6 \times 1 + 1 \times 2 = 20$. These are valid solutions, but since Jane uses a heuristic, there is no guarantee that they are optimal solutions.

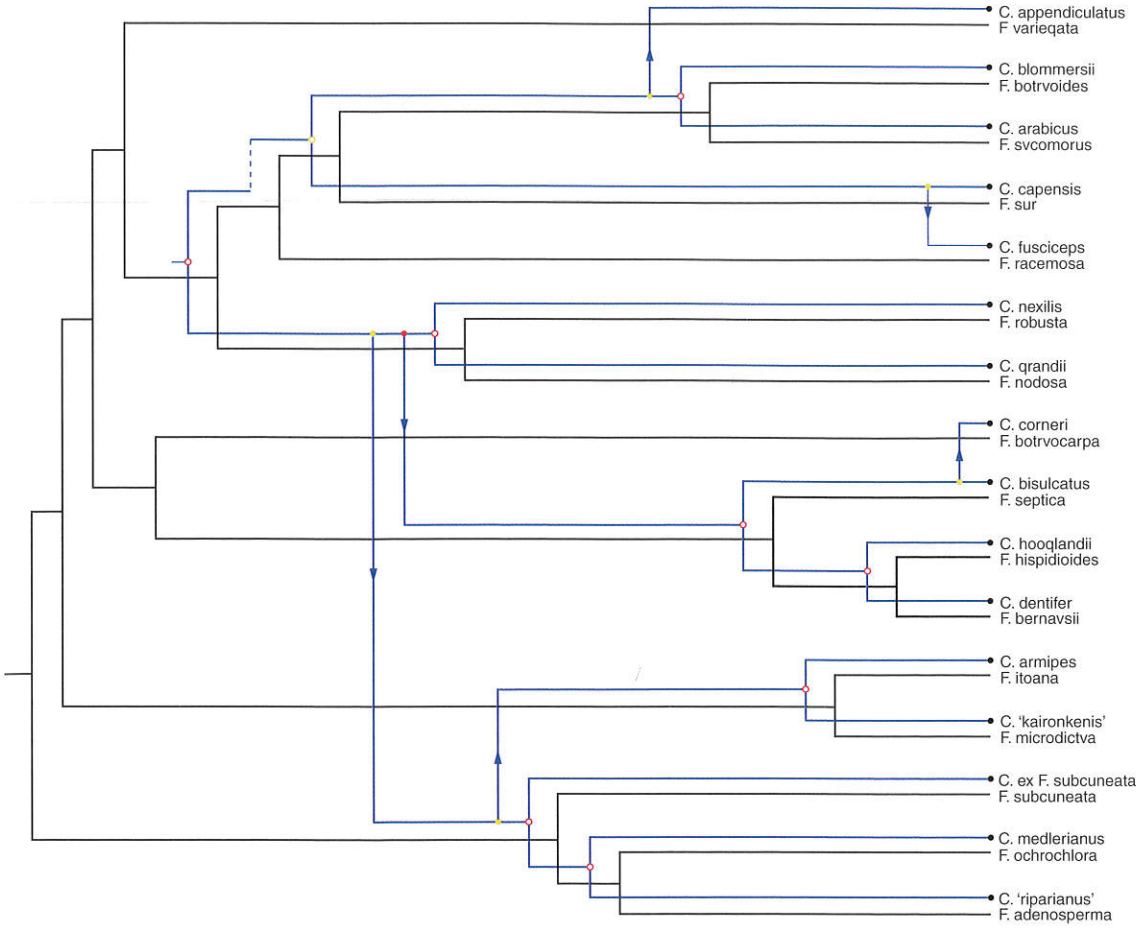



**Figure 12.7** A sample solution found by Jane.

Now, click on a solution to see what it looks like. You will see a new window with a solution that might look something like the one shown in Figure 12.7. The black tree is the host tree and the blue tree is the parasite tree. The hollow dots indicate cospeciation events while the solid red dots indicate duplication events. Some duplication events are accompanied by host switches as can be seen by the edges with arrows on them. Finally, loss events are indicated by dashed lines. To learn more about the meaning of the colors of the nodes, please read the tutorial on the Jane website. (You might notice that there appear to be only 6 duplications rather than 12. In this cost model, each duplication actually counts as two duplications – one for each of the two child species that result from the duplication event.) Try this out for the gopher_louse.tree file that you downloaded.

Next, let's take a look at the finch and indigobird data set in the file Vidua.tree. The trees here are larger than the others that you've experimented with previously; the host tree has 33 tips and the parasite tree has 21 tips (some host species have no parasites). Open this file in Jane and, this time, choose the "Number of Generations" used in the genetic algorithm to be small – let's try 3 generations. Similarly, let's use a small population size in the genetic algorithm – let's make it 4. Click on "Go" and Jane will run its genetic algorithm for 3 generations with 4 orderings per generation. You'll see some solutions reported in the "Solutions" window – these are the best solutions that resulted from our artificial evolution of solutions in this case. Note the cost of these solutions.

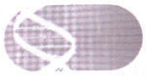
As biologists, we know that natural selection works slowly and more effectively in large populations. So, let's now increase the "Number of Generations" to a larger value – say 20 – and let's increase the size of the population in each generation to something larger as well, perhaps 100. Now, click "Go" again. The old solutions will still be listed here, but below them will be the new solutions found from this longer and larger evolutionary simulation. Take a look at the cost of these solutions! You should see that much better solutions were found in this second run.

Now, you can perform a statistical experiment to get a sense of whether or not the cost of the best solution found by Jane is suggestive of coevolution. More precisely, you can test the null hypothesis that the best solution found for the observed data – that is, the least-cost mapping of the given parasite tree onto the host tree given the observed mapping between the tips of the parasite tree and the tips of the host tree – is no better than we would find for random trees and tip mappings. If that's true, then the case for coevolution for these species is weak. If it's false, we are likely to accept that coevolution was at work here.

To try this out for yourself, click on the "Stats Mode" tab in the middle of the Jane window. By clicking "Go," Jane will find the best solution it can for the observed data and compare it with the best solution it can find for 50 random samples, each of which is the same pair of trees but with a completely random mapping between the tips of the host and parasite trees. The histogram at the bottom right shows the costs of the 50 samples: our original tip mapping is indicated in the histogram in red and the 50 random mappings are indicated by blue bars. If the majority of the random samples have higher cost than the original mapping, it is likely that the low cost for the observed tip mapping is not due to randomness. In particular, if 5% or fewer of the random solutions are better than the observed, this is considered strong evidence against the null hypothesis. Notice that you can change the sample size from 50 to any value that you like. Try it!

You can also test an alternative null hypothesis that the solution for the observed data is no better than random when the parasite tree *and* the tip mapping are randomized.

To do so, click on the "Random Parasite Tree" button in the "Statistical Parameters" panel and then press "Go" again. Now, try these computational experiments all over again with the other data sets. You will discover that, indeed, the case for coevolution is very compelling in each case.



## DISCUSSION

**This chapter has explored aspects of the field of cophylogeny – the study of the evolutionary associations of species. Since we can't travel backwards in time to study these relationships *in vivo*, we do the next best thing and study them *in silico* – that is, using computational methods. We've explored one computational approach for cophylogeny reconstruction and the *Jane* software that uses this approach.**

**Using computational tools, biologists are developing a better understanding of how parasites such as HIV and malaria have coevolved with their primate hosts which may ultimately lead to new approaches to combatting these diseases. Professor Michael Charleston, one of the leading researchers in the field of cophylogeny writes: "The global melt-down of ecological diversity is leading to greater chances of unrelated organisms interacting, leading in turn to greater potential of new pathogens crossing the species barrier into the human population. Understanding the way in which such cross species transmissions occur is of fundamental importance and it is through phylogenetic tools such as cophylogenetic maps which will shed the light we need."[14]**

**In addition to this pragmatic need, cophylogeny allows us to explore some of the beautiful and surprising ways that nature works, as Darwin himself imagined over 150 years ago.**

## QUESTIONS

(1) The Jane website (http://www.cs.hmc.edu/~hadas/jane) contains a number of sample host and parasite trees, including several that were discussed in this chapter. If you haven't done so already, download the "Ficus and Ceratsolen" file (called Ficus-Ceratosolen.tree) for the fig/wasp mutualism. Open this file in Jane and you will see in the upper-left corner of the Jane panel that these trees both have 16 tips.

(a) Use Jane to find solutions for this pair of trees. You may use the default settings of 30 generations and a population size of 30. Jane will present a number of different solutions found. Click on a solution to view it. Then, click on another solution to view it. Finally, click on a third solution. You will now have three solution windows open. These solutions will differ in some places but will agree in others. Describe where these solutions differ.

(b) Next, enter "Stats Mode" and click the "Go" button. Take a look at the histogram produced. The dashed red line shows the cost of the best solution found for the original data and the blue bars indicate the best solutions found for 50 random samples. What do these results suggest?

(2) Using the Ficus–Ceratosolen data set, make a note of the number of cospecation, duplication, host switch, and losses in the solutions found by Jane. (If you are still in "Stats Mode," you will need to go back to "Solve Mode" to do this.) Jane allows biologists to set the relative costs of each of these four event types. This is done by clicking on the "Settings" menu and selecting "Set Costs." (You will be asked if you would like to clear the solution table. Click "Yes".) Now, change the cost of a loss (sorting) event from 2 to 1, click "Go" to re-solve the problem, and note the number of each of the four event types used in the best solutions found. Explain why the solutions to the first case differ from the second case.

(3) Do a web search for "cophylogeny" and/or "host parasite" to find at least one more example of a host-parasite system. Briefly describe this system and the results found by the authors.

## REFERENCES

[1] B. Meeuse and S. Morris. *The Sex Life of Flowers.* Facts on File, 1984.

[2] figweb. http://www.figweb.org/.

[3] G. D. Weiblen and G. W. Bush. Polination in fig pollinators and parasites. *Molec. Ecol.*, 11:1573–1578, 2002.

[4] M. S. Hafner and S. A. Nadler. Phylogenetic trees support the coevolution of parasites and their hosts. *Nature*, 332:258–259, 1988.

[5] J. DaCosta and M. Sorenson. http://www.indigobirds.com.

[6] M. D. Sorenson, C. N. Balakrishnan, and R. B. Payne. Clade-limited colonization in brood parasitic finches (*Vidua* spp.). *System. Biol.*, 53:140–153, 2004.

[7] Understanding evolution: HIV's not-so-ancient history. http://evolution.berkeley.edu/evolibrary/news/081101_hivorigins.

[8] R. Libeskind-Hadas and M. Charleston. On the computational complexity of the reticulate cophylogeny reconstruction problem. *J. Comput. Biol.*, 16(1):105–117, 2009.

[9] M. Charleston. Jungles: A new solution to the hostparasite phylogeny reconciliation problem. *Math. Biosci.*, 149:191–223, 1998.

[10] Michael Charleston. TreeMap. http://www.it.usyd.edu.au/ mcharles/software/treemap/treemap.html.

[11] D. Merkle and M. Middendorf. Reconstruction of the cophylogenetic history of related phylogenetic trees with divergence timing information. *Theor. Biosci.*, 123(4):277–299, 2005.

[12] D. Merkle and M. Middendorf. Tarzan. http://pacosy.informatik.uni-leipzig.de/pv/Software/Tarzan/PV-Tarzan.engl.html.

[13] C. Conow, D. Fielder, Y. Ovadia, and R. Libeskind-Hadas. Jane: A new tool for cophylogeny reconstruction problem. *Algorith. Mol. Biol.*, 5(16), 2010. http://www.almob.org/content/5/1/16.

[14] M. Charleston. Principles of cophylogeny maps. In M. Lässig and A. Valleriani (eds) *Biological Evolution and Statistical Physics.* Springer-Verlag, 2002.