UC Davis Computer Science Technical Report CSE-2005

On the Full-Decomposition Optimality Conjecture for Phylogenetic Networks

Dan Gusfield

January 25, 2005

On the Full-Decomposition Optimality Conjecture for Phylogenetic Networks

Dan Gusfield*

Keywords: Molecular Evolution, Phylogenetic Networks, Perfect Phylogeny, Ancestral Recombination Graph, Recombination, Gene-Conversion, SNP

Abstract

Phylogenetic networks are models of evolution that go beyond trees, allowing biological operations that are not consistent with tree-like evolution. One of the most important of these biological operations is (meiotic) recombination between two sequences (homologous chromosomes). The algorithmic problem of reconstructing a history of recombinations, or determining the minimum number of recombinations needed, has been studied in a number of papers [10, 11, 12, 24, 23, 25, 16, 13, 7, 14, 6, 9, 8, 18, 19, 15].

*In an earlier paper [**?**], we established that for any set of sequences M, there is a phylogenetic network deriving M, that has a natural "fully decomposed" structure. That paper also conjectured that there is always a fully decomposed network for M which minimizes the number of recombination used, over all possible phylogenetic networks for M. That conjecture is still open. In this paper, we prove a mathematically weaker, but biologically motivated, version of that conjecture.*

1 Introduction to Phylogenetic Networks and Problems

With the growth of genomic data, much of which does not fit ideal evolutionary-tree models, and the increasing appreciation of the genomic role of such phenomena as recombination, recurrent and back mutation, horizontal gene transfer, cross-species hybridization, gene conversion, and mobile genetic elements, there is greater need to understand the algorithmics and combinatorics of phylogenetic networks on which extant sequences were derived [21, 22]. Recombination is particularly important in deriving chimeric sequences in a population of individuals of the same species. Recombination in populations is the key element underlying techniques that are widely hoped to locate genes influencing genetic diseases.

Formal definition of a phylogenetic network

There are four components needed to specify a phylogenetic network that allows multiple-crossover recombination (see Figure 1).

*Dept. of Computer Science, 3051 Engineering II, University of California, One Shields Avenue, Davis, CA 95616. Email: gusfield@cs.ucdavis.edu Research Supported by NSF grant EIA-0220154

A phylogenetic network N is built on a directed acyclic graph containing exactly one node (the root) with no incoming edges, a set of internal nodes that have both incoming and and outgoing edges, and exactly n nodes (the leaves) with no outgoing edges. Each node other than the root has either one or two incoming edges. A node x with two incoming edges is called a *recombination* node.

Each integer (site) from 1 to m is assigned to exactly one edge in N, but for simplicity of exposition, none are assigned to any edge entering a recombination node. There may be additional edges that are assigned no integers. We use the terms "column" and "site" interchangeably.

Each node in N is labeled by an m-length binary sequence, starting with the root node which is labeled with some sequence R, called the "root" or the "ancestral" sequence. For a node v in N, we define S_v as the sequence that labels node v. Since N is acyclic, the nodes in N can be topologically sorted into a list, where every node occurs in the list only after its parent(s). Using that list, we can constructively define the sequences that label the non-root nodes, in order of their appearance in the list, as follows:

a) For a non-recombination node v, let e be the single edge coming into v. The sequence labeling v is obtained from the sequence labeling v's parent by changing the state (from 0 to 1, or from 1 to 0) of the value at site i, for every integer i on edge e. This corresponds to a mutation at site i occurring on edge e.

b) For the recombination at node x, let Z and Z' denote the two m-length sequences labeling the parents of x. Then the "recombinant sequence" X labeling x can be any m-length sequence provided that at every site i, the character in X is equal to the character at site i in (at least) one of Z or Z'.

The "event" that creates X from Z and Z' is called a "multiple-crossover recombination". To fully specify the event, we must specify for every position i whether the character in X "comes from" Z or Z'. This specification is forced when the characters in Z and Z' at position i are different. When they are the same, a choice must be specified. For a given event, we say that a *crossover* occurs at position i if the characters at positions $i-1$ and i come from different parents. It is easy to determine the minimum number of crossovers needed to create X by a recombination of Z and Z'.

The sequences labeling the leaves of N are the extant sequences, i.e., the sequences that can be observed. We say that an (n, m)-phylogenetic network N *derives (or explains)* a set of n sequences M if and only if each sequence in M labels one of the leaves of N.

With these definitions, the classic "perfect phylogeny" [4] is a phylogenetic network without any recombinations. That is, each site mutates exactly once in the evolutionary history, and these is no recombination between sequences.

There are two restricted forms of recombination that are of particular biological interest. One is where X is formed from a *prefix* of one of its parent sequences (Z or Z') followed by a *suffix* of the other parent sequence. This is called "single-crossover recombination" since it uses exactly one crossover, and it is the definition of recombination used in [7, 9, 8]. The other case is when X is formed from a prefix of one parent sequence, followed by an internal segment of the other parent sequence, followed by a suffix of the first parent sequence. This is a two-crossover recombination and is usually called "gene-conversion". It is believed [1] that during meiosis, single-crossover recombination is the dominant form of recombination occurring in intervals of DNA contained between neighboring genes, while gene-conversion is the dominant form of recombination in intervals of DNA contained inside a single gene. At a different biological scale, what we have defined as two-crossover recombination models "lateral gene-transfer" or "hybrid speciation", and the main result in this paper applies to those, and other biological models of "reticulate evolution".

What we have defined here as a phylogenetic network with single-crossover recombination is the digraph part of the stochastic process called an "ancestral recombination graph (ARG)" in the population

genetics literature. (see [20] for example).

In the context of meiotic recombination, the assumption that the sequences are binary is motivated today by the importance of SNP data, where each site can take on at most two states (alleles) [2]. In the context of macroevolution, complex evolutionary characters are usually considered to be binary (either present or absent)[3].

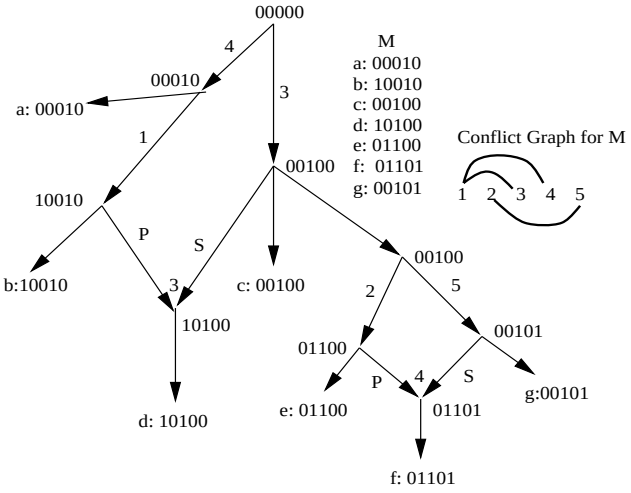

Figure 1. A phylogenetic network that derives the set of sequences M. The two recombinations shown are single-crossover recombinations, and the forced crossover point is written above the recombination node. In general the recombinant sequence exiting a recombination node may be on a path that reaches another recombination node, rather than going directly to a leaf. Also, in general, not every sequence labeling a node also labels a leaf.

1.1 Rooted and Root-Unknown problems

Problems of reconstructing phylogenetic networks, given an input set of binary sequences M, can be addressed either in the rooted case, or the root-unknown case. In the *rooted* phylogenetic network problem, a required root or ancestral sequence R for the network is specified in advance. In the *root-unknown* phylogenetic network problem, no ancestral sequence is specified in advance, and the algorithm must select an ancestral sequence.

2 A Fundamental Decomposition Theory for Phylogenetic Networks and Incompatible Characters

In this section we review the main result of an earlier paper [**?**], that for any input M, there always is a phylogenetic network of a particular, natural structure.

In a phylogenetic network N, let w be a node that has two paths out of it that meet at a recombination node x. Those two paths together define a "recombination cycle" Q. Node w is called the "coalescent node" of Q, and x is the recombination node of Q. In Figure 1, the nodes labeled 00000 and 00100 are coalescent nodes of two different recombination cycles.

If a recombination cycle in a phylogenetic network N is not isolated (a "gall" in the termonology of [7, 9] and other papers), it shares at least one edge with some other recombination cycle. We can add another cycle to that blob if the new cycle shares an edge with at least one cycle already on the blob. Continuing in this way, we ultimately get a maximal set of recombination cycles in N that form a single connected subgraph of N, and each cycle shares at least one edge with some other cycle in the set. We call such a maximal set of cycles a "blob".

Clearly, because of maximality, the blobs in a phylogenetic network N are well-defined. Moreover, if we contract each blob in N to a single point, the resulting network is a directed tree T'. This follows because if the resulting graph had a cycle (in the underlying undirected graph) that cycle would correspond to a recombination cycle which should have been contracted. We call T' a "tree of blobs" or a "blobbed tree". So every phylogenetic network N can be viewed as a blobbed tree. The edges in T' are called "tree edges" of N. Note that in each blob B, either there is exactly one node v with no edges directed into it, or there is exactly one node v with an edge directed into it from a node not in B. In the first case, v is the root of N and the "root" of B. In the second case, v is the root of B.

Given a set of binary sequences M, two columns i and j in M are said to be *incompatible* if and only if there are four rows in M where columns i and j contain all four of the ordered pairs 0,1; 1,0; 1,1; and 0,0. For example, in Figure 1 columns 1 and 3 of M are incompatible because of rows a, b, c, d. The test for the existence of all four pairs is called the "four-gamete test" in the population genetics literature. A site that is not involved in any incompatibility is called a "compatible site".

Given a sequence S, two columns i and j in M are said to *conflict (relative to S)* if and only if columns i and j contain all three of the above four pairs that differ from the i, j pair in S.

The classic Perfect Phylogeny theorem (in the terminology of this paper) is that there is a root-unknown phylogenetic network without any recombination cycles, that derives a set of binary sequences M, if and only if there is no incompatible pair of columns. Similarly, there is a phylogenetic network with ancestral sequence S, without any recombination cycles, that derives M, if and only if there is no pair of columns that conflict relative to S. For one exposition of this classic result, see [5].

Incompatibility and Conflict Graphs

We define the "incompatibility graph" $G(M)$ for M as a graph containing one node for each column (site) in M, and an edge connecting two nodes i and j if and only if columns i and j are incompatible. Similarly, given a sequence S, we define the "conflict graph" $G_S(M)$ for M (relative to S) as a graph containing one node for each column in M, and an edge connecting two nodes i and j if and only if columns i and j conflict relative to S. Figure 1 shows the conflict graph relative to the all-zero sequence S. This conflict graph is also the incompatibility graph for M.

Given a sequence S, we define the "conflict graph" $G_S(M)$ for M (relative to S) as a graph containing one node for each column in M, and an edge connecting two nodes i and j if and only if columns i and j conflict relative to S. Figure 1 shows the conflict graph relative to the all-zero sequence S. This conflict graph is also the incompatibility graph for M. When S is the all-zero sequence, the conflict graph is denoted $G_0(M)$.

A "connected component" (or "component" for short), C, of a graph is a maximal subgraph such that for any pair of nodes in C there is at least one path between those nodes in the subgraph. A "trivial" component has only one node, and no edges. The conflict graph in Figure 1 has two components. In [**?**], we proved the following fundamental result:

Theorem 2.1 *Let $G(M)$ be the incompatibility graph for M. Then, there is a phylogenetic network N that derives M where every blob contains all and only the sites of a single non-trivial connected component of $G(M)$, and every compatible site is on a tree edge of N.*

Stated another way, for any input M, there is a blobbed-tree that derives M, where the blobs are in one-one correspondence with the non-trivial connected components of $G(M)$, and if B is the blob corresponding to component C, then B contains all and only the sites in C. We say that a network is "fully-decomposed" (relative to $G(M)$) if it has the above structure. Theorem 2.1 says that for every M, there is always a fully-decomposed network for M. However, it is not true that every phylogenetic network for every input M is fully decomposed.

There is an analogous theorem to Theorem 2.1 in the case that the ancestral sequence S is known in advance. In that case, there is a phylogenetic network N that derives M, with ancestral sequence S, where the blobs in N are in one-one correspondence with the non-trivial connected components of $G_S(M)$, and any non-conflicting site is on a tree edge of N.

In [**?**] we stated the following conjecture, which still remains open.

> **Conjecture**: For any M, there is always a fully-decomposed phylogenetic network for M which has the minimum number of recombinations, over all possible phylogenetic networks for M.

We call this the "Full-Decomposition Optimality Conjecture".

3 Main Result: Progress on the Full-Decomposition Optimality Conjecture

In this paper, we state and prove a weaker, but biologically motivated, version of the full-decomposition optimality conjecture.

For any node u in a network N, let S_u denote the sequence labeling u in N. A node is considered an ancestor of itself, and a blob is considered ancestral to every node in the blob, and to every node that can be reached by a directed path from some node on the blob.

Let N be an optimal phylogenetic network for M, i.e., one that minimizes the number of recombinations over all phylogenetic networks for M. For ease of exposition, we assume that every node in N is labeled with a distinct sequence, every edge in N contains one or more sites, except for edges directed into a recombination node (these contain no sites). It is always possible to create these conditions by merging nodes with the same label, and contracting any edge that contains no site.

We consider the case when the ancestral sequence is known, and in M. For simplicity, let the ancestral sequence be the all-zero sequence. A simple lower bound on the number of needed recombinations in any network that derives M is the number of distinct rows of M, minus the number of distinct columns of M, minus one. This is called the "haplotype" bound [16].

An equivalent way to say that the haplotype bound is tight is that in a network N for M, a) no edge contains more than one site, and b) each node v in N is "visible", meaning that every sequence generated in N is in M. In that case, N is optimal, i.e., it uses the minimum possible number of recombinations over all phylogenetic networks for M. Note however, that visibility by itself does not imply that the haplotype bound is tight, and there are optimal networks where each node is visible, but the haplotype bound is not tight.

So a sufficient (but not necessary) condition for the visibility of all nodes is that the "haplotype lower bound" [16] on the minimum number of recombinations equals the true minimum. Simon Myers has shown [17] that under the neutral coalescent model with recombination, the expected difference between the haplotype bound and the true minimum is bounded by a *constant* as the number of sequences goes to infinity. Thus, there may be optimal phylogenetic networks where all nodes are visible, more often than might at first be assumed.

In this paper we establish the following

Theorem 3.1 *If there is a network N for M where conditions a) and b) are satisfied, then there is a fully-decomposed phylogenetic network DN for M that minimizes the number of recombinations used over all phylogenetic networks for M.*

We will demonstrate a one-one correspondence between the nodes of N and the nodes of DN, where each pair of corresponding nodes has the same sequence label, and where a node in N is a recombination node if and only if its corresponding node in DN is a recombination node.

The proof can be modified to show that weaker conditions suffice for the existence of DN. For example, it is sufficient that there is an optimal network N for M in which every node is visible. That is, we can allow edges that contain more than one site. The weakest sufficient condition is that for every pair of sites i, j, if an i, j state-pair occurs at some node in N, then it occurs in some sequence in M. The proof can be further extended to show that for any N that satisfies this condition (whether N is optimal or not) there is a fully-decomposed network which uses the same number or fewer recombinations as does N. We will sketch some of these extensions later. But for now we assume that network N is optimal, and that it contains exactly one site per edge except for edges that enter recombination nodes, and that each node in N is visible.

The proof strategy

First, since N is a DAG we can examine the nodes of N in some topological order, i.e., where a node is examined only after all of its ancestors have been examined. At every step in this examination, let N' be the subnetwork of N that has been examined, and let M' denote the subset of sequences of M that are generated in N' (i.e., the set of sequences that label the nodes in N that have been examined). Let $G_0(M')$ denote the conflict graph for M'.

We will prove inductively that there is a fully-decomposed phylogenetic network DN', relative to $G_0(M')$, for M' that uses the same number of recombinations as does N'. More precisely, we prove inductively that there is a one-one correspondence between nodes in N' and in DN' such that for any pair of corresponding nodes, both nodes in the pair have the same label, and one node in the pair is a recombination node if and only if both nodes are. We also maintain the inductive claim that every edge in DN' contains one site, except for edges directed into a recombination node. We call this the "edge claim".

The basis Initially, N' consists of the root node of N, which is a "tree", and clearly during some additional portion of the examination, N' remains a tree. During that time, we make DN' identical to N'. Since N' is a tree, there are no conflicts in M', and since DN' is a tree, there are no cycles or recombinations, so DN' is fully-decomposed and has the same number (zero) of recombinations as does N'. Since DN' is identical to N', it generates M' and the edge claim holds.

The inductive step Assume that the claims hold inductively to some point in the examination, and suppose v is the next node examined in N. In order to talk about how the sequences, networks and

graphs are modified in an inductive step, we use $M', N', DN', G_0(M')$ to denote those objects at the start of the inductive step, and use $M'', N'', DN'', G_0(M'')$ for those objects at the end of the inductive step. M'' is always M' after the addition of sequence S_v. There are three cases to consider.

Case 1: Node v has one parent u in N (labeled with sequence S_u) and the edge into v contains site i. By construction, DN' has a node D_u labeled S_u, and we extend DN' by adding an edge out of D_u to a new node D_v and put site i on edge (D_u, D_v). The resulting DN'' generates all the sequences in M'', and the edge claim is clearly satisfied. Since the node D_v in DN'' is a leaf, the blob structure of DN' is identical to that in DN''. Since there is only one sequence in M'' with a 1 at site i, site i is not involved in any conflicts in M'', so the set of conflicts in M' and M'' is the same, and so the set of non-trivial connected components is the same in $G_0(M')$ and $G_0(M'')$. This proves the inductive step when Case 1 occurs.

Case 2: Node v is a recombination node in N with parent nodes a and b. By induction, there are two nodes D_a and D_b labeled S_a and S_b in DN'. Suppose that D_a and D_b nodes are in the same blob in DN'. We add a new recombination node D_v to DN' and direct two edges into D_v from D_a and D_b. The resulting DN'' generates M'', the edge claim holds, and the blob structure for DN' and DN'' is identical. We must show now that the partition of nodes into connected components in $G_0(M')$ is the same in $G_0(M'')$, although $G_0(M'')$ may have some edges that $G_0(M')$ does not.

For contradiction, suppose that the non-trivial connected components of the two conflict graphs are different. Since all conflicts in M' are also in M'', the edges of $G_0(M')$ are a subset of the edges in $G_0(M'')$. So if the connected components of the two graphs are different, there must be an edge (i, j) in $G_0(M'')$ that is not in $G_0(M')$, and i and j must be in two different connected components (possibly trivial components) of $G_0(M')$. So the addition of (i, j) to $G_0(M')$ causes the merge of two connected components of $G_0(M')$. Now by the inductive claim, since i and j are in different connected components of $G_0(M')$, they are not in the same blob of DN', and since the blob structure of DN' is identical to that of DN'', they are not in the the same blob in DN''. Hence in DN'', i and j are not contained together in any recombination cycle of DN''. But DN'' generates M'', and the most basic fact about conflicting sites is that for any set of sequences $\overline{M}$, two sites can be in conflict *only* if they are contained in some common cycle in *every* phylogenetic network for $\overline{M}$ (see [9] for a complete proof). So i and j cannot be in conflict in M'' and hence the component structure of $G_0(M)$ is identical to the component structure of $G_0(M'')$.

Case 3: Node v is a recombination node in N with parent nodes a and b in N'. Assume that S_a is the prefix sequence and S_b is the suffix sequence at the recombination at v. Let r be the crossover point of the recombination at v. By induction, there are two nodes D_a and D_b labeled S_a and S_b in the current DN'. Suppose the corresponding nodes are *not* in the same blob in DN' (either node may be in no blob).

If we were to add a new recombination node D_v to DN' and direct two edges into D_v from D_a and D_b, with a recombination crossover at r, the resulting network would generate M'', but it may not be the network we seek. We define a "hyper-cycle" as a cycle containing tree edges and blobs (see Figure 2). The addition of D_v and edges (D_a, D_v) and (D_b, D_v) to DN' creates a unique hyper-cycle that also defines a blob in the resulting network. However, it is not true in general that the sites in this blob would all be contained in one connected component of $G_0(M'')$. In order to achieve that property, we have to be more careful in the selection of the parents for D_v.

Creating DN'' in Case 3

Note that node D_a is either the root of DN', or the head (at the arrow end) of a directed tree edge in DN', or is in a blob B but is not the root node of B. The same is true of D_b. If a node z is the head of a tree edge (q, z) in DN', then we define $sup(z)$ to be q. If a node z is in a blob B and is not the root node w of B, we define $sup(z)$ to be w. See Figure 3. In Case 3, we first execute procedure UP.

Procedure UP

Set z_a to D_a and set z_b to D_b.

While ((z_a is not an ancestor of z_b in DN') and (S_v can be formed by a recombination between $S_{sup(z_a)}$ and S_{z_b})) {

set $z_a = sup(z_a)$; }

/* Conceptually, each iteration moves the prefix parent of D_v up the original hyper-cycle, and (except possibly for the last iteration) removes an edge or a blob from the hyper-cycle, while preserving all the sequences generated./*

While ((z_b is not an ancestor of z_a in DN') and (S_v can be formed by a recombination between $S_{sup(z_a)}$ and S_{z_b})) {

set $z_b = sup(z_b)$ }

/* These moves change the suffix parent of D_v and (generally) remove edges or blobs from the original hyper-cycle, while preserving all the sequences generated. /*

End of UP

After Procedure UP is finished, we have the following exclusive possibilities: either 3a) z_a and z_b are different nodes, but on the same blob in DN'; or 3b) they are the same node; or 3c) one is a strict ancestor of the other; or 3d) neither node is an ancestor of the other, but one node is on a blob that is strictly ancestral to the other node; or 3e) none of the above. We will show how to create DN'' in each of these subcases, and complete the proof of the inductive step in each subcase.

Subcase 3a) When z_a and z_b are on the same blob in DN', but are different nodes, create DN'' from DN' by adding node D_v and adding directed edges (z_a, D_v) and (z_b, D_v). S_{z_a} is the prefix sequence and S_{z_b} is the suffix sequence in this recombination, and r is the crossover point. The situation now is the same as in Case 2, and so the inductive step for subcase 3a) is proved.

Subcase 3b) When z_a and z_b are the same node D_u, then both parent sequences are the same, so $S_v = S_u$, and since node D_u is in DN', we can create a network D'' that generates M'', by adding an edge from D_u to D_v in DN' with no mutation on it. Note that now D_v is not a recombination node, but there is a one-one correspondence between nodes in N'' and in D'', such that the corresponding nodes have the same labels. Now consider replacing N'' with D'' in N, meaning that the subgraph N'' is removed from N, the subgraph D'' is added to N, and for any directed edge (p, q), where p is in N'' and q is not in N'', replace (p, q) with (D_p, q). This creates a network that generates M using one fewer recombinations than does N, which is a contradiction. Hence, subcase 3b) cannot happen when N is optimal.

Subcases 3c), 3d) and 3e)

In subcases 3c) and 3d) one of z_a or z_b (WLOG z_a) is an ancestor of the other (z_b) or is in a blob B which is strictly ancestral to the other (z_b). When this happens, we execute Procedure DOWN to see if z_a should be moved closer to z_b.

If z_a is the tail of a tree edge (z_a, q) in DN' and node q is an ancestor of z_b, then define $inf(z_a)$ to be q. Else, if z_a is in a blob B that is ancestral to z_b, then define $inf(z_a)$ to be the last node in B on any path from B to z_b. Note that $inf(z_a)$ is unique and well-defined. If both conditions hold, the first definition applies, and even if there are many paths from B to z_b, they all have the same last node on B or else B would not be maximal. Hence, $inf(z_a)$ is well-defined. See Figure 4.

Procedure DOWN

While ((z_a is not equal to z_b) and (S_v can be formed by a recombination between $S_{inf(z_a)}$ and S_{z_b})) {
set $z_a = inf(z_a)$; }

/* Each change moves z_a closer to z_b, and removes an edge or a blob from the hyper-cycle, while preserving all of the sequences generated. /*

end of DOWN

In subcase 3e) z_a and z_b are different nodes by definition. In subcases 3c) and 3d) Procedure DOWN could end with $z_a = z_b$. If that happens, then we have another instance of subcase 3b) where the inductive step has been proven, so assume z_a and z_b are different at the end of Procedure DOWN. Then, in all three subcases 3c), 3d) and 3e), create DN'' from DN' by extending an edge from z_a to D_v and an edge from z_b to D_v, making node D_v a recombination node with the crossover point r. Clearly, DN'' creates the sequences in M'', the edge claim holds, and the number of recombinations used in DN'' is equal to the number of recombinations in N''. We now need to prove that the blob structure of DN'' corresponds correctly to the connected component structure in $G_0(M'')$, which would complete the proof of the inductive step.

Define the ordered set of nodes $L_b = z_b, sup(z_b), sup(sup(z_b))..., z_{tb}$, where z_{tb} is the first node such that $sup(z_{tb})$ is an ancestor of both z_a and z_b. (see Figure 5).

Similarly, define the ordered set of nodes $L_a = z_a, sup(z_a), sup(sup(z_a))..., z_{ta}$, where z_{ta} is the first node such that $sup(z_{ta})$ is an ancestor of both z_a and z_b.

Let H be the hyper-cycle in DN'' created by the addition of node D_v to DN' and the addition of edges from z_a and z_b to D_v. Note that node $sup(z_{tb}) = sup(z_{ta})$, and it is the "lowest" node that is an ancestor of every node on H That is, any other node that is there is an ancestor of every node on H, is also an ancestor of $sup(z_{tb})$. See Figure 5. Let H_b be the part of the hyper-cycle between $sup(z_{tb})$ and z_b (see Figure 5). Note that L_b is in DN', and that the head (the arrow end) of every tree edge on H_b is in L_b, and one node in every blob B in H_b, other than the root node of B, is in L_b. Recall, for the following Lemmas, that we are examining subcases 3c), 3d) and 3e), and that $z_a \neq z_b$.

Lemma 3.1 *For any node z on L_b, the set of sites with state 1 in $S_{sup(z)}$ must be a strict subset of the set of sites with state 1 in S_z.*

Proof The claim is trivially true if z is not a recombination node, since then the unique edge into z must contain a site. So, assume z is a recombination node in a blob B in DN'. Now, every site with state 1 at the root node w of B retains the state 1 at each node in B (because at any recombination in B, both parents have state 1 at that site), so the set of sites with state 1 in S_w is a subset of the set of sites with state 1 in S_z. If that subset is not strict, then $S_z = S_w$, and we can create a network D' from DN', where the recombination at node z in DN' is replaced by an edge from w to z. But if we then replace N' in N with D', we obtain a network that generates M and uses fewer recombinations than does N, contradicting the optimality of N. □

For any node z on L_b, let W_z be the set of sites with state 1 in S_z but state 0 in $S_{sup(z)}$. Lemma 3.1 says that W_z is never empty.

Lemma 3.2 *Let z be any node on L_b other than z_b, and let q be any node in L_b below z. If j is any site in W_z with state 0 in S_v, and if i is any site in W_q with state 1 in S_v, then sites i and j conflict in M''.*

Proof WLOG, assume $i < j$. Since q is below z and i is in W_q, application of Lemma 3.1 (repeated if needed) implies that site i has state 0 in S_z. Then the pair i, j have states 0,0 at the root of DN'', state 0,1 in S_z, states $1, 1$ in S_q, and state $1, 0$ in S_v. By the full visibility assumption and the fact that DN'' generates M'', these four state pairs exist in M'', and hence i and j conflict in M''. □

Theorem 3.2 *Let $z'_b \neq z_b$ be the last node on L_b with the property that some site in $W_{z'_b}$ has state 0 in S_v. Then all sites on all blobs and edges in the part of H_b between $sup(z'_b)$ and z_b are together in a single connected component of $G_0(M'')$.*

Proof We assume that z'_b exists, since the theorem is vacuously true otherwise. Note that there must be a site i_b in W_{z_b} which has state 1 in S_v. If not, then Procedure UP would also have moved z_b to $sup(z_b)$. So, by Lemma 3.2, for every node z on L_b other than z_b, where some site j has state 0 in S_v, states i_b and j conflict. It follows that sites i_b and j' conflict for some site j' in $W_{z'_b}$. Now if z is a node below z'_b and *no* sites in W_z have state 0 in S_v, then let i be any site in W_z. By Lemma 3.2, states i and j' conflict in M''. So, some site j' in $W_{z'_b}$ conflicts with site i_b in W_{z_b}, and for every other node z between z'_b and z_b, some site in W_z either conflicts with site j' in $W_{z'_b}$ or conflicts with site i_b in W_{z_b}. See Figure 6. Further, if node z in L_b is a recombination node in a blob B in DN', then by the inductive hypothesis, every site in W_z is in one connected component of $G_0(M')$ together with every site in B, and they continue to be together in one connected component of $G_0(M'')$, since all conflicts in M' are also in M''. Finally, when node z in L_b is the head of a tree edge in DN', so trivially, all sites on that edge are in the same component of $G_0(M'')$, and all sites on that edge are in W_z. It follows that all sites on all blobs and edges in the part of H_b between $sup(z'_b)$ and z_b are together in a single connected component of $G_0(M'')$. □

If $z'_b = z_{tb}$, then Theorem 3.2 proves that all sites on the path H_b are in a single connected component of $G_0(M'')$. Otherwise, let L'_b be the subset of L_b from node $Sup(z'_b)$ to z_{tb}. Clearly, if z'_b is not z_{tb}, then z_{tb} is in L'_b. To fully prove that all sites on H are in one connected component of $G_0(M'')$, we must examine the subcases 3c), 3d) and 3e) separately.

Finishing the inductive step for Subcase 3c)

WLOG, assume that node z_a is a strict ancestor of z_b. Note that z_{tb} must be $inf(z_a)$. See Figure 7. By Lemma 3.1 the set of sites with state 1 in $S_{inf(z_a)}$ is a strict superset of the set of sites with state 1 in S_{z_a}. So if all the sites in $W_{inf(z_a)}$ have state 1 in S_v, a recombination between $S_{inf(z_a)}$ and S_{z_b} at crossover point r results in the same sequence, namely S_v, that results from a recombination of S_{z_a} and S_{z_b} at point r. But in that case, Procedure DOWN would have moved z_a to $inf(z_a)$. Hence, some site(s) in $W_{inf(z_a)}$ have state 0 in S_v, and z'_b must be z_{tb}. Then, by Theorem 3.2, all the sites on H_b are in a single connected component of $G_0(M'')$. But, there is a direct edge from z_a to D_v and it contains no sites, so all sites in H are in H_b, and the inductive step has been proved in subcase 3c).

Finishing the inductive step for Subcase 3d)

WLOG, assume that node z_a is on a blob B that is strictly ancestral to z_b, but Z_a is not an ancestor of Z_b. Note that z_{tb} must be $inf(z_a)$. See Figure 8. Since there is only a single edge from z_a to D_v with no sites on it, if $z'_b = z_{tb}$, then by Theorem 3.2 all sites in H would be in a single connected component of $G_0(M'')$ and the inductive step for subcase 3d) would be proven. So, assume that $z'_b \neq z_{tb}$, and hence all sites in $W_{z_{tb}}$ have state 1 in S_v.

Since all sites in $S_{z_{tb}}$ have state 1 in S_v, if there is no site $j \in W_{z_a} - W_{z_{tb}}$ which has state 1 in S_v, a recombination between $S_{z_{tb}}$ and S_{z_b} at crossover point r would result in the same sequence, namely S_v, that results from a recombination of S_{z_a} and S_{z_b} at point r. But, z_{tb} is $inf(z_a)$, so if those two recombinations both yield S_v, Procedure DOWN should have moved z_a to $inf(z_a)$, hence there must be some site $j \in W_{z_a} - W_{z_{tb}}$ which has state 1 in S_v. But then sites j and i_b (defined in Theorem 3.2) conflict in M'': assuming $i_b < j$, those state-pairs are 0,0 at the root of DN'', they are 0,1 in S_{z_a}, they are 1,0 in S_{z_b}, and 1,1 in S_v. By similar reasoning, site j conflicts with every site i in W_z for every node z between z_{tb} and $sup(z'_b)$ inclusive. Finally, since site j is on some edge in blob B, as are all sites in $W_{z_{tb}}$, and all sites in a blob are inductively together in a single connected component of $G_0(M')$ (and hence of $G_0(M'')$), all sites in H are in a single connected component of $G_0(M'')$, and the inductive step for subcase 3d) is proven.

Finishing the inductive step for Subcase 3e)

We define L_a, z_{ta}, z'_a, L'_a and H_a, in ways analogous to the definitions of L_b, z_{tb}, z'_b, L'_b and H_b.

In subcase 3e) neither z_a nor z_b is an ancestor of the other, and neither is on a blob that is ancestral to the other. By arguments that are symmetric to those given above for z_b, W_{z_a} is not empty and must contain a site i_a which has state 1 in S_v.

Since neither z_a nor z_b is an ancestor of the other, and neither is on a blob that is ancestral to the other, site i_a (respectively i_b) must be on an edge e, or in a blob B in DN', with the property that there is no directed path in DN' from e or B to z_b (respectively z_a). Therefore, sites $i_b \in W_{z_b}$ and $i_a \in W_{z_a}$ conflict in M''.

If z is any site on L_b, then by Lemma 3.2, every site in W_z that has state 0 in S_v conflicts with i_b. Similarly, if z is any site on L_a, then every site in W_z that has state 0 in S_v conflicts with i_a. Further, if z is any site on L_b other than z_{tb} (the reason for its exclusion will be explained below), then by the same reasoning used to conclude that i_a and i_b conflict in M'', every site W_z which has state 1 in S_v is in conflict with site i_a in M''. Similarly, if z is any site on L_a other than z_{ta}, then, every site W_z which has state 1 in S_v conflicts with site i_b in M''. It follows that all sites on all blobs and edges in the part of H_b between z_{tb} and z_b are in a single connected component of $G_0(M'')$ together with all sites on all blobs and edges in the part of H_a between z_{ta} and z_a.

Note that in subcase 3e) either z_{ta} and z_{tb} have the same single parent node, or they are both in the same blob B, and neither is the root node of B (if both are in B and one was the coalescent node of B then it would be an ancestor of both z_a and z_b violating the definition of z_{ta} or z_{tb}). See Figure X.

Let j be any site in $W_{z_{ta}}$ and assume it has state 1 in S_v. The reason that we could not include z_{ta} with the other nodes in the above argument, and conclude that j is in conflict with site i_b, is that the edge containing site j may be on a directed path to z_b, and hence we could not conclude that the (i_b, j) state-pair of $(1, 0)$ exists in M''. But, that situation is only possible when z_{ta} and z_{tb} are together on some blob B in DN', and neither is the root node of B, as shown in Figure Xb. So when the situation is as shown in Figure Xa, we can conclude that all sites on H are in a single connected component of $G_0(M'')$ and the induction step is proved for subcase 3e).

So assume that z_{ta} and z_{tb} are together on B as shown in Figure Xa. If some site j in $W_{z_{ta}}$ (or $W_{z_{tb}}$)

has state 0 in S_v, then site j conflicts with i_a (or i_b) as established earlier. Now j is contained on some site in B, and since z_{ta} and z_{tb} are together on B, and all sites on B are together in a connected component of $G_0(M'')$, it follows that all sites in H are together in a single connected component of $G_0(M'')$.

So, the remaining situation is that z_{ta} and z_{tb} are together on B, and every site in $W_{z_{ta}}$ and $W_{z_{tb}}$ has state 1 in S_v. If there is some site j in $W_{z_{ta}}$ that is not in $W_{z_{tb}}$, then there is no directed path to z_{tb} from the edge in B that contains site j, and so j conflicts with i_b, and again we can conclude that all sites on H are in one connected component of $G_0(M'')$. The same conclusion holds if there is a site j in $W_{z_{ta}}$ that is not in $W_{z_{tb}}$. So, assume that that $W_{z_{tb}} = W_{z_{ta}}$. In that situation, $S_{z_{ta}} = S_{z_{tb}}$, and both z_{ta} and z_{tb} are recombination nodes in B in DN'. But we could then remove the edges into one of those nodes, say z_{ta}, and add an edge from z_{tb} to it, and still generate all the sequences in M' while reducing the number of recombinations by one. If we replace N' in N with this modified DN', the result would be a network that produces M with fewer recombinations than does N, a contradiction.

So in all occurrences of subcase 3e), all the sites in H are in a single connected component of $G_0(M'')$, and the inductive step is proved for subcase 3e).

This completes the proof of the inductive step for Case 3), and so Theorem 3.1 is proved.

4 Extending the proof

It is immediate that we can remove the assumption that the ancestral sequence is known in advance - the given proof applies to whatever sequence is the ancestral sequence in the optimal network N.

We assumed in the statement of Theorem 3.1 that every edge in the optimal N contains at most one site. The proof, as given, may break down if that were not the case. For example, consider the situation in case 3e) if z_b is the head of a tree edge containing more than one site. It remains true that some site in W_{z_b} must have state 1 in S_v, but some sites in W_{z_b} may have state 0 in S_v. In M'', those sites will not be in conflict with any sites in H, and the inductive step will fail. A similar problem can arise in case 3c) if edge (z_a, z_b) exists and has sites with state 1 in S_v and sites with state 0 in S_v. The solution is the following: in the inductive step when node v in N is examined, if e is an edge in DN' which has a site with state 1 in S_v and also a site with state 0 in S_v, then we divide e into two edges e_1 followed by e_2. We place on e_1 all sites on e that have state 1 in S_v, and place on e_2 all sites on e that have state 0 in S_v. Note that there are no other edges created, so this network does not have the node visibility property, but this will not matter.

Then we allow Procedures UP and DOWN to operate on the modified network, exactly as before. For example, consider the situation mentioned above that in case 3e) some sites on the edge e into z_b have state 0 in S_v. After dividing e into e_1 and e_2, with the sites of state 0 in S_v on e_2, Procedure UP will move z_b to the head of e_1 and all the sites in e with state 0 in S_v will be off the hyper-cycle. Note that the node at the head of e_1 has a sequence label that is not in M'', so not all nodes in DN'' are visible. That is ok, since in the proof we only require that all nodes in N are visible. A full proof of the correctness of this approach requires a close examination of the existing proof and is left for later.

In order to avoid the assumption that N is optimal, we must execute another procedure to assure that $S_z \neq S_{sup_z}$ and $S_z \neq S_{inf_z}$ for every node z on H_b or H_a.

Finally, it is clear from the proof, that the assumption that all nodes in N are visible is stronger than is needed. It is sufficient that for every pair of sites i, j, any i, j state pair that exists at a node in N appears in some sequence in M.

References

[1] A. Berry and A. Barbadilla. Gene conversion is a major determinant of genetic diversity at the DNA level. In R.S. Singh and C.B. Krimbas, editors, *Evolutionary Genetics: From Molecules to Morphology*, pages 102–123. Cambridge University Press, 1999.

[2] A. Chakravarti. It's raining SNP's, hallelujah? *Nature Genetics*, 19:216–217, 1998.

[3] J. Felsenstein. *Inferring Phylogenies*. Sinauer, Sunderland, MA., 2004.

[4] D. Gusfield. Efficient algorithms for inferring evolutionary history. *Networks*, 21:19–28, 1991.

[5] D. Gusfield. *Algorithms on Strings, Trees and Sequences: Computer Science and Computational Biology*. Cambridge University Press, Cambridge, UK, 1997.

[6] D. Gusfield. Optimal, efficient reconstruction of Root-Unknown phylogenetic networks with constrained recombination. Technical report, Department of Computer Science, University of California, Davis, CA, 2004.

[7] D. Gusfield, S. Eddhu, and C. Langley. Efficient reconstruction of phylogenetic networks (of SNPs) with constrained recombination. In *Proceedings of 2'nd CSB Bioinformatics Conference*, Los Alamitos, CA, 2003. IEEE Press.

[8] D. Gusfield, S. Eddhu, and C. Langley. The fine structure of galls in phylogenetic networks. *INFORMS J. on Computing, special issue on Computational Biology*, 16:459–469, 2004.

[9] D. Gusfield, S. Eddhu, and C. Langley. Optimal, efficient reconstruction of phylogenetic networks with constrained recombination. *J. Bioinformatics and Computational Biology*, 2(1):173–213, 2004.

[10] D. Gusfield and D. Hickerson. A new lower bound on the number of needed recombination nodes in both unrooted and rooted phylogenetic networks. Report UCD-ECS-2004-06. Technical report, University of California, Davis, 2004.

[11] J. Hein. Reconstructing evolution of sequences subject to recombination using parsimony. *Math. Biosci*, 98:185–200, 1990.

[12] J. Hein. A heuristic method to reconstruct the history of sequences subject to recombination. *J. Mol. Evol.*, 36:396–405, 1993.

[13] R. Hudson and N. Kaplan. Statistical properties of the number of recombination events in the history of a sample of DNA sequences. *Genetics*, 111:147–164, 1985.

[14] J. D. Kececioglu and D. Gusfield. Reconstructing a history of recombinations from a set of sequences. *Discrete Applied Math.*, 88:239–260, 1998.

[15] B. Moret, L. Nakhleh, T. Warnow, C.R. Linder, A. Tholse, A. Padolina, J. Sun, and R. Timme. Phylogenetic networks: Modeling, reconstructibility, and accuracy. *IEEE/ACM Transactions on Computatational Biology and Bioinformatics*, pages 13–23, 2004.

[16] S. R. Myers and R. C. Griffiths. Bounds on the minimum number of recombination events in a sample history. *Genetics*, 163:375–394, 2003.

[17] Simon Myers. *The detection of recombination events using DNA sequence data*. PhD thesis, University of Oxford, Oxford England, Department of Statistics, 2003.

[18] L. Nakhleh, J. Sun, T. Warnow, C.R. Linder, B.M.E. Moret, and A. Tholse. Towards the development of computational tools for evaluating phylogenetic network reconstruction methods. In *Proc. of 8'th Pacific Symposium on Biocomputing (PSB 03), pages 315-326*, 2003.

[19] L. Nakhleh, T. Warnow, and C.R. Linder. Reconstructing reticulate evolution in species - theory and practice. In *Proc. of 8'th Annual International Conference on Computational Molecular Biology*, pages 337–346, 2004.

[20] M. Norborg and S. Tavare. Linkage disequilibrium: what history has to tell us. *Trends in Genetics*, 18:83–90, 2002.

[21] D. Posada and K. Crandall. Intraspecific gene genealogies: trees grafting into networks. *Trends in Ecology and Evolution*, 16:37–45, 2001.

[22] M. H. Schierup and J. Hein. Consequences of recombination on traditional phylogenetic analysis. *Genetics*, 156:879–891, 2000.

[23] Y. Song and J. Hein. On the minimum number of recombination events in the evolutionary history of DNA sequences. *Journal of Mathematical Biology*, 48:160–186, 2003.

[24] Y. Song and J. Hein. Parsimonious reconstruction of sequence evolution and haplotype blocks: Finding the minmimum number of recombination events. In *Proc. of 2003 Workshop on Algorithms in Bioinformatics*, Berlin, Germany, 2003. Springer-Verlag LNCS.

[25] L. Wang, K. Zhang, and L. Zhang. Perfect phylogenetic networks with recombination. *Journal of Computational Biology*, 8:69–78, 2001.

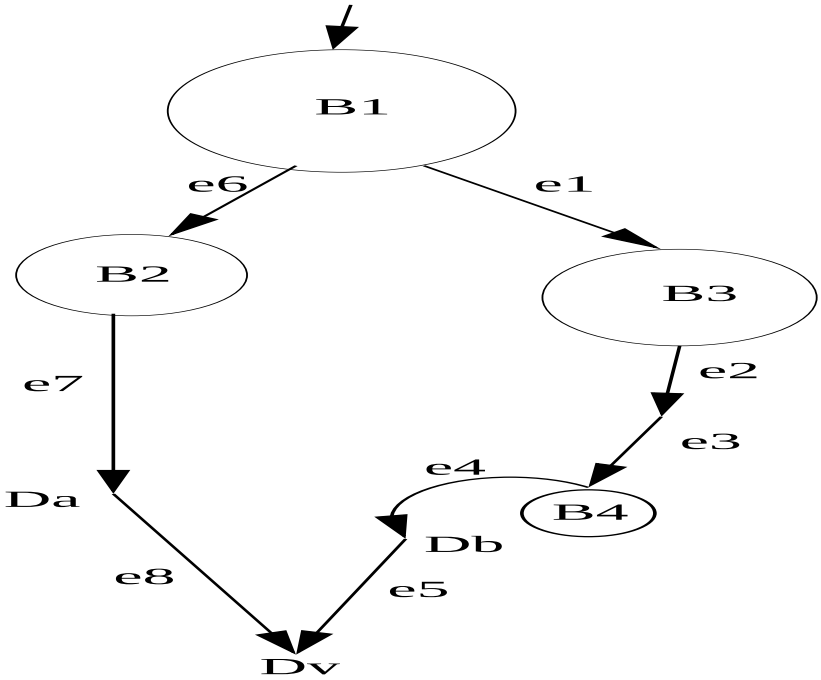

Figure 2. Figure for the proof

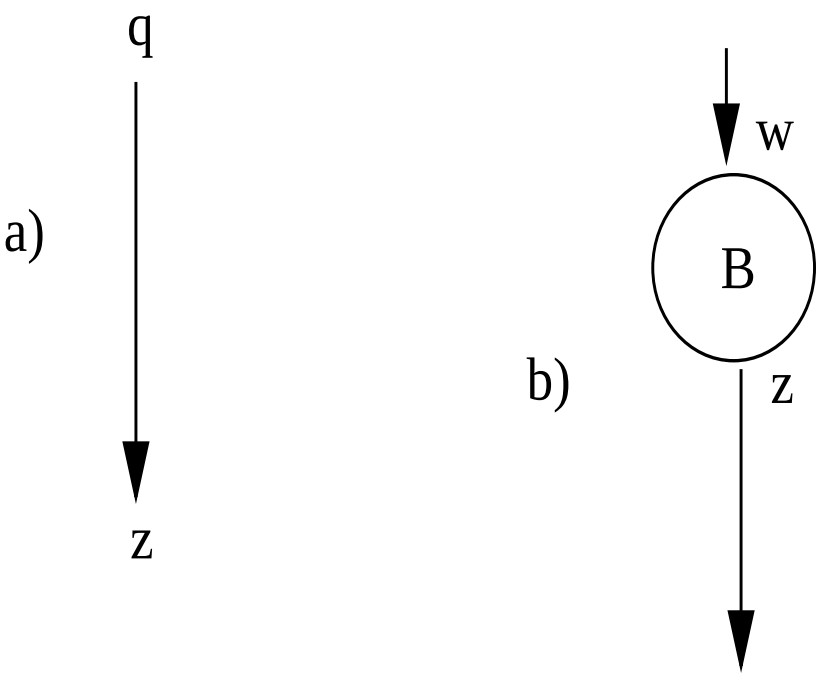

Figure 3. Figure for the proof

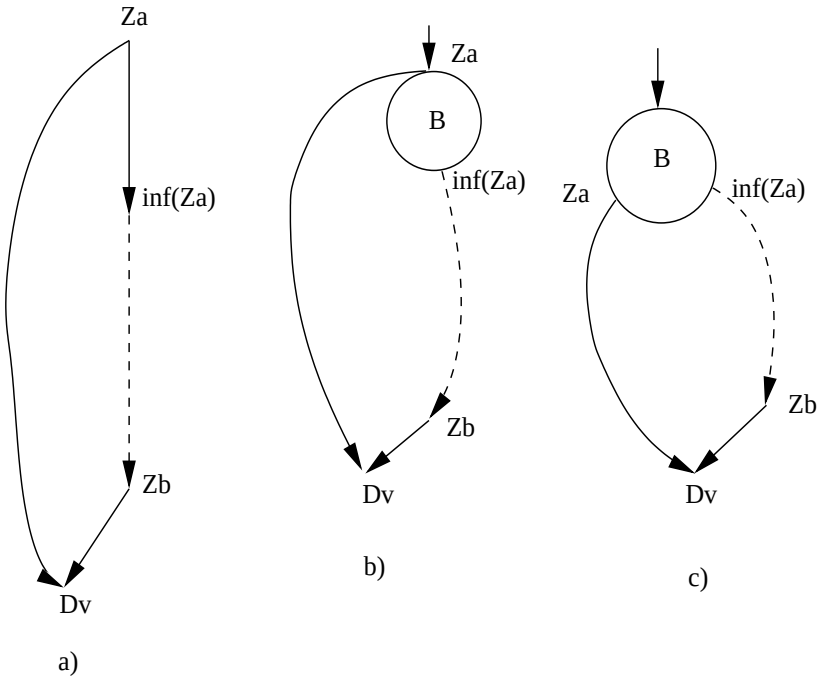

Figure 4. Figure for the proof

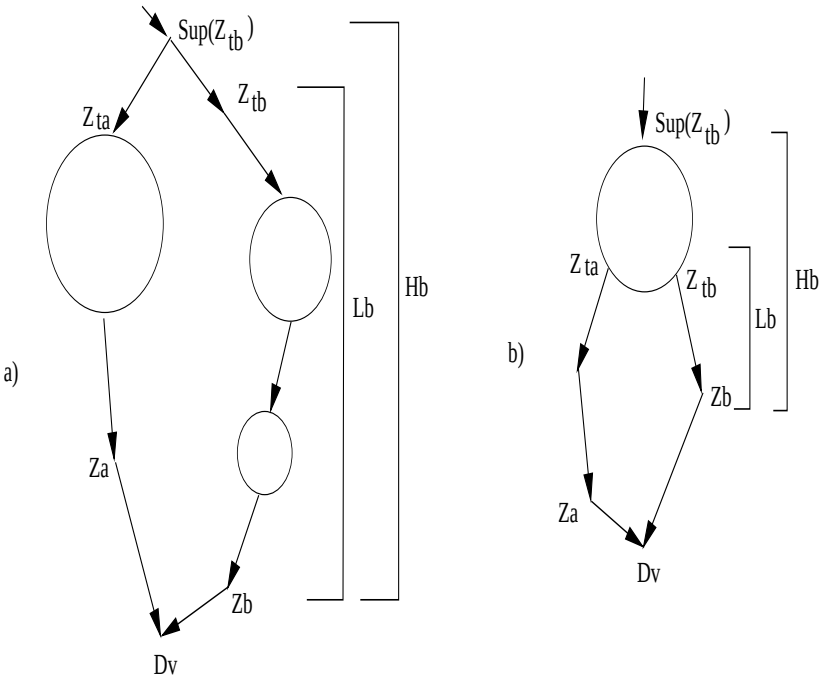

Figure 5. Figure for the proof

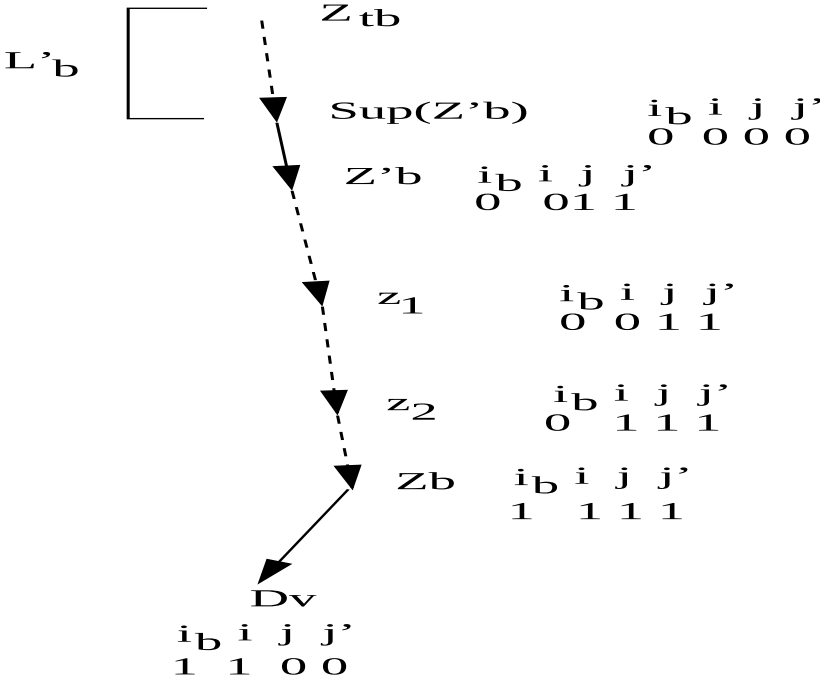

Figure 6. Figure for the proof

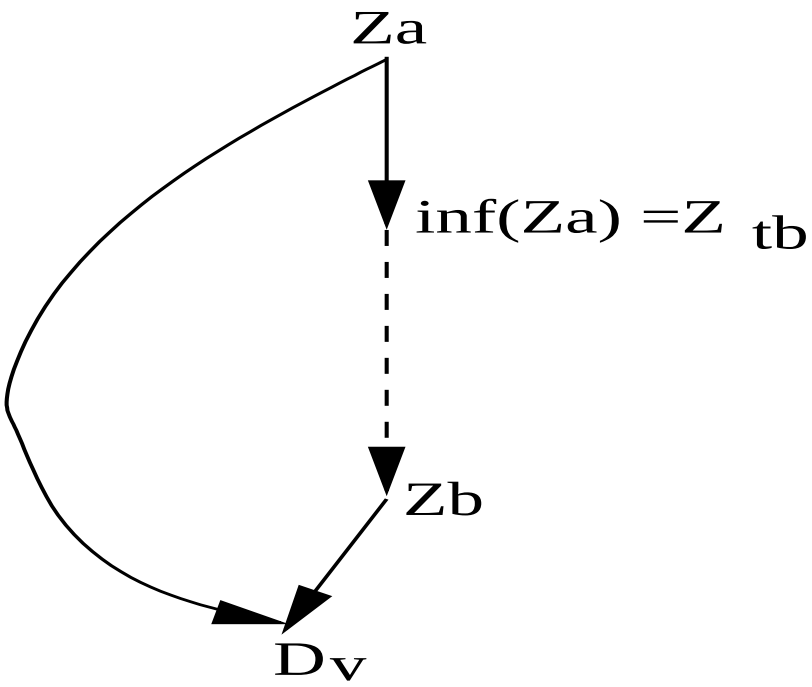

Figure 7. Figure for the proof

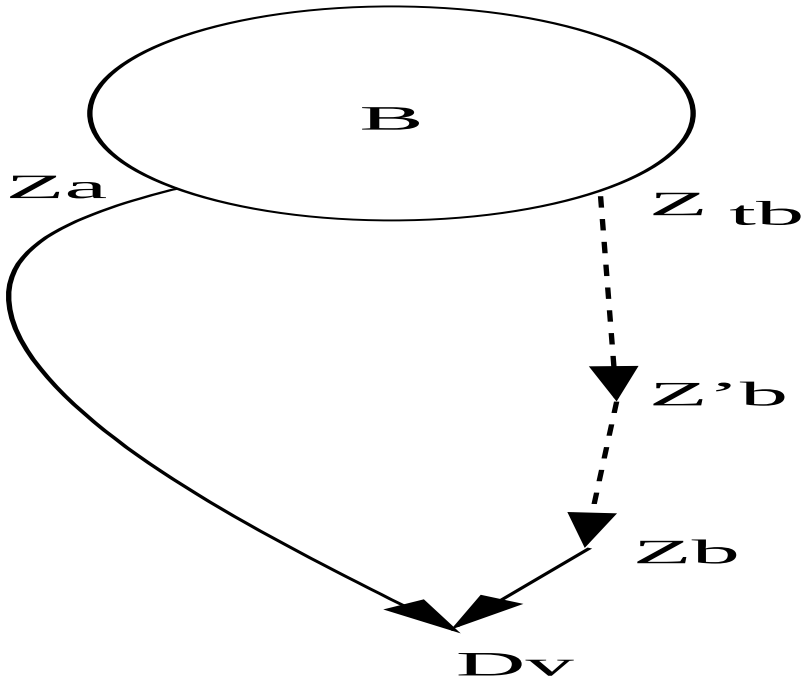

Figure 8. Figure for the proof