Finding the maximum phylogenetic diversity score on phylogenetic networks

Bachelor of Science
Applied Mathematics thesis

Victor J. Veenman

Finding the maximum phylogenetic diversity score on phylogenetic networks

Bachelor of Science Applied Mathematics thesis

by

Victor J. Veenman

to obtain the degree of Bachelor of Science
at the Delft University of Technology,

Student number: 4833228
Project duration: February 23, 2022 – June 29, 2022
Thesis committee: Dr. M.E.L. Jones, TU Delft, supervisor
Dr. J. W. van der Woude TU Delft, Graduation committee

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

TU Delft

Contents

Layman's-summary iv

Abstract v

Acknowledgements vi

1 Introduction 1
- 1.1 Context and relevance . . . 1
- 1.2 Problem description and research objectives . . . 1
- 1.3 Structure of the thesis . . . 1

2 Literature review and background information 2
- 2.1 Integer Linear Programs . . . 2
- 2.2 What is phylogenetic diversity . . . 3
- 2.3 Phylogenetic networks . . . 3
  - 2.3.1 Biological representation of phylogenetic networks . . . 4
- 2.4 Phylogenetic diversity scores . . . 5
  - 2.4.1 Phylogenetic score on trees . . . 5
  - 2.4.2 Phylogenetic score functions on networks . . . 5
  - 2.4.3 Maximizing PD score functions . . . 7

3 ILP formulations of Phylogenetic networks 8
- 3.1 ILP representation of Max-AllPath-PD on tree networks . . . 8
- 3.2 ILP representation of Max-AllPaths-PD on networks . . . 9
- 3.3 ILP representation of Max-Network-PD on networks . . . 9
  - 3.3.1 Rewriting γ_e to linear constraints . . . 10
    - 3.3.1.1 γ_e above reticulations and leaves . . . 10
    - 3.3.1.2 γ_e above tree vertices . . . 10
  - 3.3.2 The ILP formulation for networks with one reticulation . . . 14
  - 3.3.3 ILP formulation for networks with multiple reticulations . . . 15

4 Conclusion and discussion 16
- 4.1 Conclusion . . . 16
- 4.2 Recommendations for future research . . . 16

Bibliography vii

Appendices viii

Layman's summary

Conservation biology focusses on preservation of nature and earth's biodiversity. Since it is not possible to preserve all species on earth, it is important to know which species should get priority. This is where phylogenetic diversity comes in. Phylogenetic diversity is a measure for biodiversity. By checking which group of species has the greatest diversity, that group can be selected to have priority. This thesis will create an optimization model that will look for the group of species with the highest diversity. For this, multiple types of networks and phylogenetic score functions are considered.

Abstract

Due to human advancement in recent centuries, extinction rates of animals and plants around the earth have greatly risen. Conservation biology, the study of conservation of nature and earth's biodiversity, aims to protect species from these increasing extinction rates.
Phylogenetic diversity is a measure for biodiversity that can help in selecting which species to prioritize in preserving diversity. This is necessary because there are bounds on how many of these species can be preserved, due to costs connected to this preservation. To aid in selecting the subset of species with maximum diversity, an ILP can be formulated to find this subset. In this thesis these ILP's will be formulated for different phylogenetic networks and functions that give a diversity score to these subsets.

An easy, but less realistic, function for calculating the diversity of a set is AllPath-PD. This function adds up all the weights of edges that go from the root node to the selected leaves. These selected leaves are the species in the subset.
A more realistic function is Network-PD. This function multiplies the weights of the edges by another value, γ, that represents how much of that edge can be expected to be found in the selected set of species. This function has to be used in the objective function of the ILP, but it is non linear. Therefore it must be rewritten. In networks with one reticulation, which is a vertex with two incoming edges and one outgoing edge, this is possible with several tricks after observing the behaviour of this γ on different edges. This γ can only attain certain values and by using linear constraints all these possible values can be covered.
For networks with multiple reticulations there are more values that this γ can attain, making it harder and maybe even impossible to use linear constraints to get all those values. This thesis will briefly discuss a few possible ideas that can help in making an ILP for these networks as well.
The solutions of the ILP's will be a score, which is the diversity score for the set with the highest diversity. The decision variables used for that solution can easily be translated into which species would be chosen in this subset with highest diversity.

Acknowledgements

For this thesis I would like to thank my supervisor Mark Jones. Whenever I got stuck we would have a brainstorm session where new ideas came through. If I overlooked something, or if I made an error in one of the new ideas, he would tell me that very early, saving me a lot of time trying something that was wrong from the start.

1 Introduction

1.1. Context and relevance

Because of human advancement, extinction rates of animals and plants have skyrocketed. Conservation biology is the study of the conservation of nature and earth's biodiversity. The aim is to protect species from these increasing extinction rates. Conserving all species might not be possible, but in order to conserve the biodiversity as best as possible, a group of species can be selected that will have optimal biodiversity.
This is where the subject of this thesis comes in. Integer linear programming is a mathematical optimization program for which many solvers are available. Therefore it is desired to reduce the problem of finding this group of species to a problem that can be solved with integer linear programming.

1.2. Problem description and research objectives

As mentioned before, the problem of finding a group of species with optimal biodiversity is a big problem in conservation biology. In a mathematical context this problem will look similar, but with different terms. If there is a phylogenetic network, which will be properly defined in chapter 2, with species as the leaves, the problem for finding an optimal set can be described as:

Maximize	Phylogenetic score of the selected species in the group
Subject to	The group of selected species contains at most k species

This thesis will aim to formulate this problem in such a way that it can be solved using these integer linear programming solvers. This will make it possible to get this desired group of species for any given network.

1.3. Structure of the thesis

In order to formulate the problem of finding a group of species with optimal biodiversity, it must first be clear how to measure this biodiversity. In chapter 2 phylogenetic diversity, a measure for biodiversity, will be introduced. Furthermore, it will define the way the phylogenetic networks look and how phylogenetic diversity scores can be calculated for groups of species.
In chapter 3 the problems are formulated. At first for the easiest and least realistic networks and score functions. After that an extension is made for more complicated networks and functions in order to get results that will represent reality more closely.
Chapter 4 goes over the results from chapter 3 and discusses how this can be used and what research can still be done in order to improve the current formulations.

It is recommended to view this thesis in colour, since many examples make use of colours. This helps in understanding the different observations and functions.

2

Literature review and background information

For this thesis some background information is necessary. First of all a brief explanation of ILP's. Secondly, phylogenetic diversity will be introduced together with what it can be used for. And lastly, different types of phylogenetic networks will be discussed. On these networks a phylogenetic score function can be used. These scores are what the formulated ILP's will be optimizing.

2.1. Integer Linear Programs

Integer linear programming is a mathematical optimization problem. An integer linear programming problem, refered to as ILP from now on, is very commonly used within optimization. This is because they have many solvers available that have been improved to find a solution in the shortest possible time. That is why it can be beneficial to take the time to reduce your problem into an ILP form.
ILP's consist of four parts:

- **Decision variables**: The decision variables are variables that are set to be able to attain certain values. The ILP will then try to optimize the objective function subject to the constraints. The values that these variables get in this optimal solution are the solution to the ILP. In an ILP these values must be, like the name suggests, integers.
- **Objective function**: This is the function that you try to optimize. Depending on the problem, you try to maximize or minimize the function. In this thesis, the ILP will be maximizing this objective function. As the name suggests, it is important that this is a linear function. It has to depend linearly on the decision variables. The decision variables are explained later.
- **Constraints**: These are equalities or inequalities that also depend linearly on the decision variables. These constraints ensure that not every combination of the decision variables is a feasible solution.
- **Parameters**: Parameters are variables that will not vary within the ILP. The values that these parameters will have, are calculated in advance. Because of that, the objective function and constraints are allowed to depend on these parameters in a nonlinear way.

In equation (2.1) an example can be seen for a very simple ILP. The values for α and β are known. In this case they are $\alpha = 2$ and $\beta = 3$. For this example the optimal solution is at $(x, y) = (3, 7)$, where the objective function has value $\alpha x + \beta y = 27$.

$$\begin{aligned} \text{Maximize} \quad & \alpha x + \beta y \\ \text{Subject to} \quad & \alpha y \leq 14 \\ & y - x \geq 4 \\ \text{Decision variables} \quad & x, y \in \mathbb{Z} \\ \text{Parameters} \quad & \alpha, \beta \in \mathbb{R} \end{aligned} \tag{2.1}$$

2.2. What is phylogenetic diversity

Biodiversity is a biological term used to express variety of life on earth. Phylogenetic diversity is a measure for this biodiversity. By looking at inheritable traits of different species, a value can be assigned to the relation between a predecessor and its descendant. A higher number of different traits and the importance of those traits will increase this value. Since not all the traits are equally different, there is no one to one correlation between the number of different traits and the value given to the phylogenetic diversity between the two species. With values assigned to all the different relations, a directed graph can be constructed for the tree of life.

Table 2.1 and figure 2.1 are a simplified example of how this works. In the table several strings of ones and zeroes can be seen. These values represent the inheritable traits of the species. If the species do have these traits, the value will be one, and zero otherwise. These species are then illustrated in figure 2.1 by looking at the traits and then checking where there is a difference. That is where branches split off. For every trait that is present in all the underlying species, there will be a small tick on the path. That way the number of ticks on the path is equal to the number of traits that are present for that species.[1].

Species	String of traits
0	00000000000000000000
1	01110000100000000000
2	01110011000001000000
3	01110011001110100000
4	01110011001110010000
5	10001100000000000001
6	10001100000000001000
7	10001000000000000110

Table 2.1: An example of seven species and their inheritable traits.

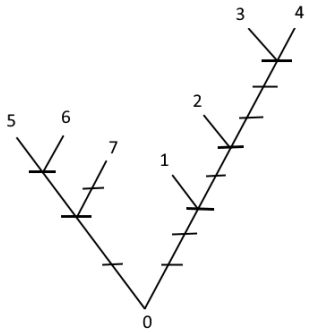


Figure 2.1: A visualisation for the species in table 2.1. Every tick on the path represents a trait present in the underlying species.

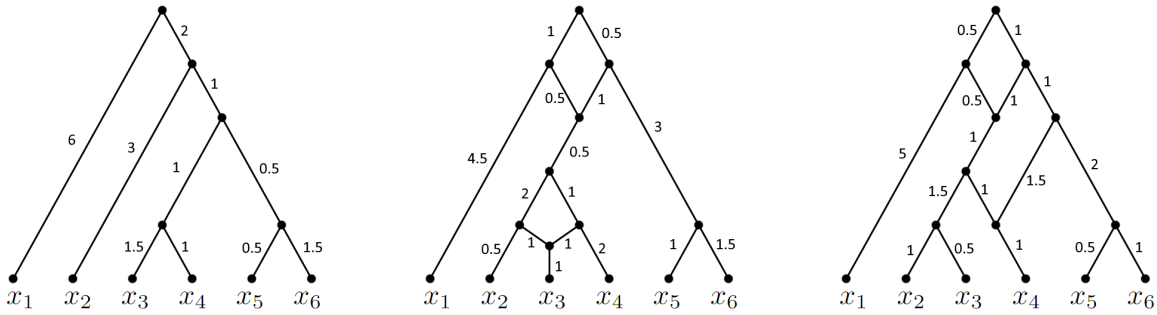
2.3. Phylogenetic networks

In this thesis three types of diversity networks will be used. A general rooted binary phylogenetic network, a simplified version of this network, which is the rooted binary phylogenetic tree network and lastly there are level-1 networks. Level-1 networks are a subclass of networks, so they have an extra constraint. The phylogenetic tree will be used to formulate a basic ILP, which can then be extended for the more realistic networks.

Definition 1. A *rooted binary phylogenetic network* $\mathcal{N}$ on X is a rooted directed acyclic graph with no parallel arcs satisfying the following properties[2]:

(i) The unique root has in-degree zero and out-degree two;

(ii) A vertex with out-degree zero has in-degree one. The set of all vertices with out-degree zero is X;



(a) Phylogenetic tree $\mathscr{T}$ on X.

(b) Phylogenetic level-1 network $\mathscr{N}$ on X.

(c) Phylogenetic network $\mathscr{N}$ on X.

Figure 2.2: Three phylogenetic networks on $X = \{x_1, \ldots, x_6\}$ with their edge weights.

(iii) All other vertices have either in-degree one and out-degree two, or in-degree two and out-degree one.

The phylogenetic network has a root, from which all leaves can be reached. There are three types of vertices if the root is not included. These vertices are *leaves*, which have one incoming edge and no outgoing edges. There are *tree vertices*. These have one incoming edge and two outgoing edges. Lastly there are *reticulations*. Reticulations have two incoming edges and one outgoing edge.
From these reticulations it is obvious that cycles will be formed. Note that there are no **directed** cycles, but because of the reticulations there will be cycles in the undirected graph. Therefore these cycles can be considered *undirected cycles*.
A *Rooted binary phylogenetic level-1 network* $\mathscr{N}$ on X is a phylogenetic network for which its underlying cycles are vertex disjoint. This is equivalent to saying that it is a phylogenetic network for which its underlying cycles have no shared edges.

Definition 2. A *Rooted binary phylogenetic tree network* $\mathscr{T}$ on X is a rooted directed tree with no paralel arcs satisfying the following properties:

(i) The unique root has in-degree zero and out-degree two;

(ii) A vertex with out-degree zero has in-degree one. The set of all vertices with out-degree zero is X;

(iii) All other vertices have in-degree one and out-degree two.

The definition of a Tree network is very similar to the definition of a general network. The only difference is in the third part. A tree network does not allow reticulations. So all vertices are either leaves or tree vertices except for the root.

In figure 2.2 three networks can be seen. The edge weights represent the phylogenetic diversity between the two connected vertices. The first of these networks is a tree network. Every branch can only split off into more branches and can never combine with another branch.
Secondly there is a level-1 network. There are 2 reticulations, both of which have cycles that do not have any shared edges or vertices.
Lastly there is a network that is no tree, since it has reticulations. But the cycles formed by these reticulations have shared vertices, so it is not a level-1 network. Since it does satisfy the constraints for a network, it will be seen as a phylogenetic network.

2.3.1. Biological representation of phylogenetic networks

Each part of a phylogenetic network has a biological meaning behind it. Like mentioned before, the edges between vertices represent inheritance between those two species. The weight of that edge will correspond with the phylogenetic diversity between those two species.
The root is the main ancestor of all species in the network.
Each tree vertex is a place where two new species evolve from the same ancestor.

Reticulations represent hybridization between two parent species. The result is a child species that has genetic traits from both parents.
Lastly there are the leaves. These are species that have not evolved any further yet. Those are also the species that are present on earth right now.
In general a phylogenetic network represents the tree of life with the relative diversity between two species as weights on the edges.

2.4. Phylogenetic diversity scores

Since the problem requires the diversity of a subset of leaves to be as large as possible, it is necessary to define how this can be calculated for a set of leaves. This section will first show how this is defined for trees and then extend this to general functions on networks.

2.4.1. Phylogenetic score on trees

Faith [1] defines the phylogenetic diversity between two species as the sum of the weights of the edges on the path between the two species. To get the phylogenetic diversity of a set, first two of the species in that set are selected. Their diversity is calculated as described. This will be the score of the set of two species. This set is called S. After that the gain function 2.2 is used to calculate the gain from this new species x. This gain is added to the score. Now the score of the new set of three species is known. Set S will be updated to be this new set. This gain function is used for all species that are added to the set.

$$G = \min_{i,j \in S} \frac{1}{2}(D_{i,x} + D_{j,x} - D_{i,j}) \tag{2.2}$$

In this function, $D_{i,j}$ is the diversity between species i and j. This gain function is a clever trick to add only the weights of the edges that were not already used in the set. Since x will go through a path that will eventually split off to i or j, the path up to that point is counted twice. The path between i and j is counted only once, but was already used for the score in the current set. Therefore subtracting the value for that path and then dividing by 2 results in only adding the weights of new edges.

2.4.2. Phylogenetic score functions on networks

On the defined phylogenetic networks, a score can be given to a subset of leaves. This score will loosely represent the diversity within the tree of life that is covered by the selected subset of leaves. This thesis looks at two different score functions. One of these is easier to calculate and use in an ILP formulation. The other function is more realistic.
These functions and the ILP's in this thesis will make use of the term *PD path*. This is the collection of paths between a leaf in the selected subset and the root vertex. In figure 2.3 the coloured edges and vertices are on this PD path.

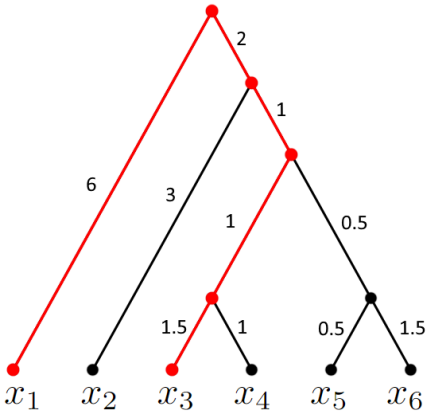
The first PD score function is the AllPaths-PD function[2]. In this function you sum up all the weights of all edges on the PD path following from the selected subset of leaves S on network $\mathcal{N}$. This AllPath-PD function would give the same result on trees as Faith's method in section 2.4.1.

$$\text{AllPaths-PD}_{\mathcal{N}}(S) = \sum_{e \in \text{Anc}(S)} w(e) \tag{2.3}$$

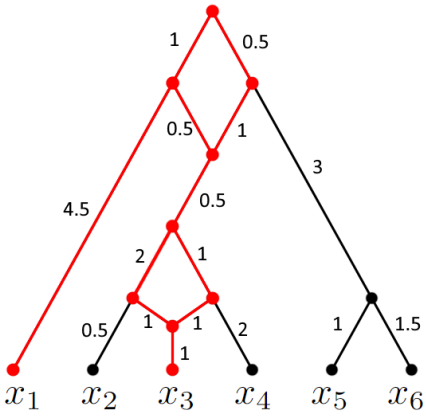
Here Anc(S) is the set of ancestral edges leading from a leaf in the selected set S to the root.
For example, the PD score for subset $S = \{x_1, x_3\}$ in figure 2.3b using the AllPaths-PD function gives us:

$$\text{AllPaths-PD}_{\mathcal{N}}(\{x_1, x_3\}) = 14$$

For tree networks, this function works intuitively, but when looking at general networks this function will give a slight bias towards leaves that have their PD path going through many reticulations. Since both edges upwards from a reticulation will be fully counted, this will be more likely to give a higher score.
In reality a descendant with two parent vertices will inherit only a part of both parents, so the edges to those parents should not be given the full score.



(a) Phylogenetic tree $\mathcal{T}$ on X with selected subset of leaves $S = \{x_1, x_3\}$



(b) Phylogenetic level-1 network $\mathcal{N}$ on X with selected subset of leaves $S = \{x_1, x_3\}$

Figure 2.3: Two phylogenetic networks with a selected subset of leaves $S = \{x_1, x_3\}$. The coloured edges and vertices are on the PD path.

That is where the second, more realistic function comes in. It still has many similarities with the AllPaths-PD function. This function is the Network-PD function[2].

Parameter $p(e)$ can be introduced as the expected proportion of features from parent u that are present in child v for an edge $e = (u, v)$. The value of this proportion is always between zero and one, so $p(e) \in [0, 1]$. This parameter will aid in expressing the partial inheritance with reticulations.

The new function also looks at all ancestral edges leading from the selected leaves to the root, but not all weights will be added equally. They will be multiplied by a term $\gamma(S, e)$. This $\gamma(S, e)$ represents how much of the features of that parent node from the edge can be found within leaves of S. This term $\gamma(S, e)$ with $e = (u, v)$ can be determined as follows[2]:

$$\gamma(S,e) = \begin{cases} 1 & \text{if } v \text{ is a leaf and } v \in S \\ 0 & \text{if } v \text{ is a leaf and } v \notin S \\ \gamma(S,e) = \gamma(S,f_1) + \gamma(S,f_2) - \gamma(S,f_1)\gamma(S,f_2) & \text{if } v \text{ is a tree vertex with outgoing edges } f_1 \text{ and } f_2 \\ \gamma(S,e) = \gamma(S,f)\,p(e) & \text{if } v \text{ is a reticulation vertex with outgoing edge } f \end{cases} \tag{2.4}$$

With this new term $\gamma(S, e)$ the Network-PD fuction can now be defined as:

$$\text{Network-PD}_{\mathcal{N}}(S) = \sum_{e \in \mathcal{N}} \gamma(S,e)\,w(e) \tag{2.5}$$

Reasoning behind the construction of γ
The γ fuction as defined in equation (2.4) looks rather complicated, but each part is defined in a way that it would make sense. First of all, the edges above leaves. Here $\gamma(S, e)$ is equal to 1 if the leaf is in S, or equal to 0 if the leaf is not in S. This is similar to how AllPath-PD decided to include an edge in the sum or to not include it in the sum.
Secondly there are the edges above a tree vertices. γ loosely refers to the expected proportion of features from the parent node of an edge that are present in leaves of set S. Therefore it has to check how much of the edges below it are included in set S. Those proportions can be added together. Every part that was included in both the outgoing edges is now counted twice. That is why $\gamma(S, f_1)\gamma(S, f_2)$ is subtracted.
Lastly there are the edges above a reticulation. These look at the γ of the outgoing edge and then multiply that with the term $p(e)$, which is the inheritance proportion of edge e that is present in outgoing edge f.
With this construction, the values of $\gamma(S, e)$ correspond with the proportion of that edge that can be found within the leaves contained in set S.

2.4.3. Maximizing PD score functions

With the functions from section 2.4.2 two optimization problems can be defined. In general the problem describes finding a subset of leaves in network $\mathcal{N}$ with cardinality k that maximizes the PD score. This problem is useful for conservation biology. Here they try to determine which k species maximize the biodiversity of a group. Since Phylogenetic diversity is a measure of this biodiversity, it makes sense that these optimization problems will lead to solutions that can be used in conservation biology. The results from the problems defined below can be used to choose which species have to be prioritized for conservation [1].
The first of the two problems will be called *Max-AllPath-PD*[2].

Max-AllPath-PD($\mathcal{N}$, k) :

Input : Phylogenetic network $\mathcal{N}$ on X and positive integer k

Objective : Determine the maximum value of AllPaths-PD$_{\mathcal{N}}(S)$ over all subsets $S \subseteq X$ of cardinality k.

The second problem, *Max-Network-PD*, uses the second PD score function[2]:

Max-Network-PD($\mathcal{N}$, p, k) :

Input : Phylogenetic network $\mathcal{N}$ on X, inheritance proportion function p and positive integer k

Objective : Determine the maximum value of Network-PD$_{\mathcal{N},p}(S)$ over all subsets $S \subseteq X$ of cardinality k.

3

ILP formulations of Phylogenetic networks

In this chapter the problems from section 2.4.3 will be reduced to different ILP formulations.
The different types of networks and functions result in different ILP's, each with their own advantages and disadvantages. At first an ILP representation for Max-AllPath-PD on tree networks is created. This will be the basis for the other ILP's.

3.1. ILP representation of Max-AllPath-PD on tree networks

Let $\mathcal{T}$ be a rooted binary phylogenetic tree network with vertex set V, corresponding and set E for its edges. Here the weights of the edges are written as $w(e)$. Each edge can be seen as an ordered pair of 2 vertices, which are its starting and endpoint, $e = (u, v)$, $\{u, v\} \in V$. We can formulate an ILP for maximizing the All-Path-PD score of a subset of species with cardinality at most k:

$$
\begin{aligned}
&\text{Maximize} && \sum_{e \in E} y_e w(e) && \\
&\text{Subject to} && \sum_{v \in V} l_v x_v \leq k && \\
& && y_{(u,v)} = x_v && \forall (u,v) \in E \\
& && x_u \geq y_{(u,v)} && \forall (u,v) \in E \\
& && \sum_{(v,u) \in E} y_{(v,u)} \geq x_v (1 - l_v) && \forall v \in V \\
&\text{Decision variables} && y_e = \begin{cases} 1 & e \text{ an edge on the PD path} \\ 0 & e \text{ an edge not on the PD path} \end{cases} && \\
& && x_v = \begin{cases} 1 & v \text{ a vertex on the PD path} \\ 0 & v \text{ a vertex not on the PD path} \end{cases} && \\
&\text{Parameters} && l_v = \begin{cases} 1 & v \text{ is a leaf} \\ 0 & v \text{ is not a leaf} \end{cases} &&
\end{aligned}
\tag{3.1}
$$

The objective function is the sum of the weights of all edges that are on the PD path. To find a maximum All-Path-PD score it makes sense to try to maximize this objective function. Whenever a solution is found with maximized objective function, the leaves with $x_v = 1$ can be selected to find set S that maximizes the AllPaths-PD function.

There are two sets of decision variables. One for vertices on the PD path and one for edges on the PD path. The constraints will enforce these two decision variables to be combined in a way that is possible with the definition of the All-Path-PD function. There is one parameter, that checks whether a vertex is a leaf or not. Since this does not change for different choices of PD paths in the graph, this is not a decision variable.

There are four constraints, each with a different purpose. The first constraint enforces the cardinality of the selected leaf set to be at most k. Since $l_v = 1$ when the vertex is a leaf and $x_v = 1$ if the vertex is on the PD-Path, their product is equal to 1 if and only if the vertex is a leaf on the PD-Path. This selection of leaf nodes has to be bounded by the definition of the problem. Since the All-Path-PD function is increasing with respect to the size of the selected leaf set, this constraint can also be written as

$$\sum_{v \in V} l_v x_v = k.$$

The second constraint is here to be sure that the edge between a vertex on the PD path and its parent is always on the PD path. This enforces the path upwards to the root to always be on the PD path for every selected leaf.

The third constraint is also needed to have paths going all the way back to the root node. Without the third constraint it would be possible for some vertex (and their path down to a leaf) to be on the PD-Path. Then from the second constraint, the edge leading to that vertex has to be on the path as well, but the parent vertex is not mandatory to be on the path. In practice, the maximized path will never have this happen, since adding edges adds to the score. But for the clarity of the ILP representation it is better to include it.

The last constraint is there to make sure that paths go down all the way to a leaf. That way an edge can not be added to the PD path without adding at least one leaf with it. If this constraint was not here, it was possible to have edges leading from the main PD path to other vertices, that are not leaves. That way they would not add to the first constraint, but they would add a value to the objective function. The way this constraint is built is as follows. $x_v(1 - l_v) = 1$ if $l_v = 0$ and $x_v = 1$, so whenever a vertex is on the PD path, but not a leaf. Then at least one of the edges leading from that vertex must be on the PD path as well. If $l_v = 1$, the right hand side of the constraint will be 0, because leaves can not have child vertices. If $x_v = 0$ the right hand side will also be equal to 0, because the child vertices of any vertex that is not on the PD path, can not be on the PD path. Purely from this constraint it is possible for an edge leading from such a vertex to be on the PD path, since $1 > 0$, but the second constraint makes that impossible.

3.2. ILP representation of Max-AllPaths-PD on networks

There is one difference between the networks and the tree networks. This difference is that networks can have reticulations. These are the vertices with in-degree two and out-degree one. In this case the ILP formulation would have to include both the ingoing edges into the PD path. But the second constraint of the ILP representation for tree networks already made sure that this would happen. Therefore the same formulation can be used as an ILP representation of AllPaths-PD on networks.

3.3. ILP representation of Max-Network-PD on networks

Since Network-PD on trees will result in the same problem as AllPaths-PD on trees, it will not be interesting to look at that problem.
Network-PD and AllPaths-PD differ only in the term γ_e. Since $\gamma(S, e) = 0$ if and only if $y_e = 0$ the objective function could be changed to:

$$\text{Maximize} \quad \sum_{e \in E} \gamma_e w(e)$$

Where $\gamma_e = \gamma(S, e)$, but since S is not known within the ILP, it would not make sense to have that as a variable in the objective function. From this point on γ_e will be used to describe $\gamma(S, e)$.
It would be desirable to have a constraint that calculates γ_e within this ILP, but this constraint would not be able to directly express γ_e, because it depends on γ_{f_1} and γ_{f_2} non-linearly whenever f_1 and f_2 are outgoing edges from a tree node. Therefore some other trick must be found to express γ_e in such a way that it depends on γ_{f_1} and γ_{f_2} linearly.
This section will focus on trying to find this linear dependency. To do this, we first consider networks with only one reticulation, since this will make observing the behaviour of γ_e easier. After that it can be extended to multiple reticulations.

γ_{f_1}	γ_{f_2}	γ_e
0	0	0
1	0	1
1	1	1
0	1	1
x	1	1
1	y	1
x	0	x
0	y	y
x	y	$x+y-xy$

Table 3.1: All options for the γ-values of outgoing edges of tree vertices and the corresponding value for γ_e.

3.3.1. Rewriting γ_e to linear constraints

When looking closely at the behaviour of the γ-function, something interesting can be seen. It will attain the values 1 and 0 relatively often and in general, $\gamma_e \in [0,1]$. The only place where this γ_e is not linearly dependent on γ_f for outgoing edges f, is for edges directly above a tree node. Therefore rewriting this γ_e will be split into two main parts. The first part will look at γ_e for edges e that are above reticulations or leaves. The second part looks at the edges above tree vertices.

3.3.1.1. γ_e above reticulations and leaves

Looking at edges e that are leading into leaves will give the easiest case. Since $\gamma_{(u,v)} = 1$ whenever leaf $v \in S$ and $\gamma_{(u,v)} = 0$ whenever leaf $v \notin S$, it is easy to see that this can be put into a single constraint:

$$\gamma_{(u,v)} = x_v \text{ when } v \text{ is a leaf}$$

Secondly there are the edges leading into reticulations. From the definition of $\gamma(S,e)$ in equation (2.4) it can be seen that this γ_e depends linearly on γ_f where f is the outgoing edge from the reticulation. $p(e)$ is just a parameter, so again a single constraint will work:

$$\gamma_{(u,v)} = \gamma_{(v,w)}\, p((u,v)) \quad \forall v \text{ where } v \text{ is a reticulation}$$

3.3.1.2. γ_e above tree vertices

To understand how γ_e behaves, it can be useful to look at all the possible options. These options are shown in table 3.1. Here x and y can be any value in the interval $(0,1)$. For any row in this table except the last row, it is clear that $\gamma_e = \max\{\gamma_{f_1}, \gamma f_2\}$. The last row is more complicated.
Luckily the case of the last row can only happen if both outgoing edges lead to a reticulation. When looking at networks with only one reticulation, this means that the case of the last row can happen at most once in the network.
First γ_e for all other tree vertices can be rewritten to linear constraints.

γ_e above regular tree vertices

All the regular tree vertices, so the vertices that do not have both outgoing edges leading to the reticulation, have $\gamma_e = \max\{\gamma_{f_1}, \gamma f_2\}$. Sadly the maximum is still not a linear function, so multiple constraints have to be used to simulate the maximum. For edge $e = (u,v)$ the following will do the same as using the maximum:

$$\begin{aligned} & m_e \geq \gamma(S,(v,w)) \qquad \forall (v,w) \in E \\ & m_e \leq 1 \\ & m_e \leq \sum_{(v,w)\in E} \gamma(S,(v,w)) \end{aligned}$$

The first constraint here makes sure that the value can not be smaller than the largest of the two γ-values.
If either one of the outgoing γ-values is equal to 1, the new γ_e must also be one, so for those cases, this second constraint finalizes it.
The only case left is when the result will be something in the interval (0,1). In other words, whenever either one of the outgoing γ-values is in that interval and the other value is zero. Therefore by summing both values,

the exact γ-value will be found.

γ_e above tree vertices with both outgoing edges leading to reticulations
The last case remaining is γ_e whenever e is the edge leading into a tree vertex where both outgoing edges lead into a reticulation. In figure 3.1 two examples can be seen. The red edges and vertices are the places where this last case is used.
To better understand how these edges and vertices work, a schematic view of a network with one reticulation can be seen in figure 3.2. Table 3.2 explains how these values can be attained at the different edges. Something that is worth noting is that in this table, there are two options left out in comparison to table 3.1. This is because in networks with one reticulation, it would never be possible for one of the outgoing edges to have $\gamma_{f_1} = 0$ and the other having $\gamma_{f_2} = y$. This is because values x and y that are in the interval $(0,1)$ can only happen whenever the edge below the reticulation has $\gamma_e = 1$. Therefore either both edges leading into the reticulation have this value x or y, or neither of them do.
This leaves us with three possible values and cases for this edge above the tree vertex where both outgoing edges lead into a reticulation.

- $\gamma_e = 1$, whenever either one of the outgoing edges has $\gamma_f = 1$
- $\gamma_e = 0$, whenever both the outgoing edges have $\gamma_f = 0$
- $\gamma_e = g$, whenever the outgoing edges have $\gamma_{f_1} = x$ and $\gamma_{f_2} = y$. Then $g = x + y - xy$

To express these choices in linear constraints, a new decision variable is needed.
This decision variable will be a check to see if the γ-value on an edge is equal to 1 or not. This new decision variable will be z_e:

$$z_e = \begin{cases} 1 & \gamma_e = 1 \\ 0 & \gamma_e \neq 1 \end{cases}$$

To make sure this decision variable actually works the way it is supposed to, a few constraints have to be added.

$$\begin{aligned}
z_{(u,v)} &= 0 && \text{if } v \text{ is a reticulation} \\
z_{(u,v)} &= x_v && \text{if } v \text{ is a leaf} \\
z_{(u,v)} &\geq z_{(v,w)} && \forall (v,w) \in E, \text{ where } v \text{ is not a reticulation} \\
z_e &\leq 1 && \forall e \in E \\
z_{(u,v)} &\leq \sum_{(v,w)\in E} z_{(v,w)}
\end{aligned}$$

The first two constraints are derived directly from the definition of γ. The third constraint is used for edges e leading into tree vertices. Then if any of the outgoing edges has $\gamma_f = 1$, γ_e must be equal to 1 as well. The fourth constraint is there to make sure that z_e can not be greater than 1. The last constraint is used for the tree vertices where there are two outgoing edges with $\gamma_f \in (0,1)$, so then $z_f = 0$, but z_e should also be 0.
Now this decision variable can be used for calculating γ_e in the ILP.

Whenever both outgoing edges of the tree vertex have γ_f equal to 0, the values of y_f are also equal to 0. By setting:

$$\gamma_{(u,v)} \leq y_{(v,w)} \quad \forall (v,w) \in E,$$

it is ensured that the gamma will also be equal to 0 in that case. This does not effect any other values, since $\gamma_e = 0$ whenever $y_e = 0$, and $\gamma_e \leq 1$ in all other cases.
Secondly, γ_e should be equal to 1 if any values of γ_f, with f the outgoing edges, is equal to 1. This will already happen, since this is a maximization problem, where γ_e is a positive variable in the objective function. Therefore, as long as there are no constraints stopping the γ_e from getting bigger, it will get its highest possible value. This highest value is 1, because of the previous constraint.

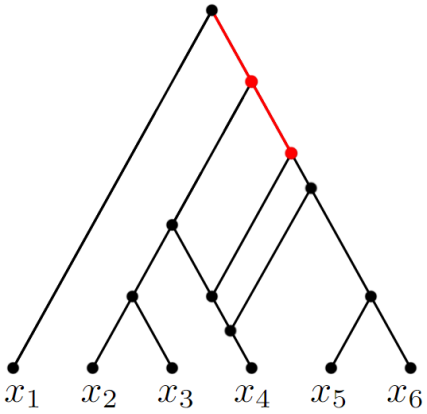
γ_{f_1}	γ_{f_2}	γ_e
0	0	0
1	0	1
1	1	1
0	1	1
x	1	1
1	y	1
x	y	$g = x + y - xy$

Table 3.2: The options for edges leading into vertices that have both outgoing edges leading to the same reticulation.

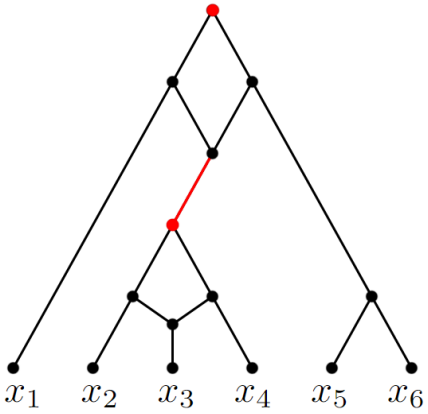
Lastly, if $\gamma_{f_1} = x$ and $\gamma_{f_2} = y$, both $z_{f_1} = 0$ and $z_{f_2} = 0$. Therefore $\sum z_f = 0$ for outgoing edges f. This leads to the conditional constraint:

$$\gamma_{(u,v)} \leq g + M(\sum_{w \in V} z_{(v,w)})$$

Here $g = x + y - xy$ and M is a sufficiently large number. Since $\gamma_e \leq 1$, $M = 1$ would suffice. Whenever one of the outgoing edges f has $\gamma_f = 1$, this constraint will no longer bound the γ, but if both $\gamma_f = 0$, γ_e would already have the stronger bound in $\gamma_e \leq y_f$.
There is still one problem. That is because this unknown value $g \in (0,1)$ has to be calculated. This can be done using a programming language before running the ILP. In that case this possible value g will just be a parameter. This is possible to do, because g only has one possible value, which is $x + y - xy$. Here x and y are the values $p(e)$ for edges e going into the reticulation.



(a) A phylogenetic network with red vertices and edges where the γ function can not be computed with the maximum



(b) A phylogenetic level-1 network with red vertices and edges where the γ function can not be computed with the maximum

Figure 3.1: 2 phylogenetic networks with red vertices and edges where the γ function can not be computed with the maximum. Nothing happens for the root node, since there is no edge above it.

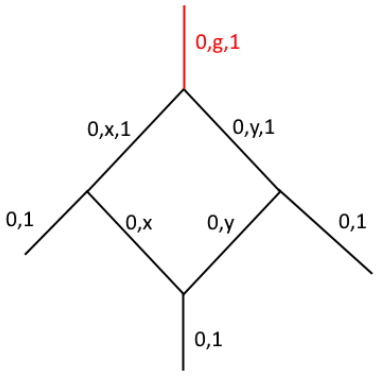


Figure 3.2: The possible γ-values that edges can attain around a cycle in a network with one reticulation.

3.3.2. The ILP formulation for networks with one reticulation

$$
\begin{aligned}
&\text{Maximize} && \sum_{e\in E} \gamma_e w(e) \\
&\text{Subject to} && \sum_{v\in V} l_v x_v \leq k \\
& && y_{(u,v)} = x_v && \forall (u,v)\in E \\
& && x_u \geq y_{(u,v)} && \forall (u,v)\in E \\
& && \sum_{(v,u)\in V} y_{(v,u)} \geq x_v(1-l_v) && \forall v\in V \\
& && m_{(u,v)} \geq \gamma_{(v,w)} && \forall (v,w)\in E \\
& && m_e \leq 1 \\
& && m_{(u,v)} \leq \sum_{(v,w)\in E} \gamma_{(v,w)} \\
& && \gamma_{(u,v)} = m_{(u,v)} && \forall (u,v)\in E \text{ where } v \text{ is a tree vertex that} \\
& && && \text{does not have both outgoing edges leading to a reticulation} \\
& && \gamma_{(u,v)} = m_{(u,v)}\, p((u,v)) && \forall (u,v)\in E \text{ s.t. } v \text{ is a reticulation} \\
& && z_{(u,v)} = 0 && \text{if } v \text{ is a reticulation} \\
& && z_{(u,v)} = x_v && \text{if } v \text{ is a leaf} \\
& && z_{(u,v)} \geq z_{(v,w)} && \forall (v,w)\in E \\
& && z_e \leq 1 && \forall e\in E \\
& && z_{(u,v)} \leq \sum_{(v,w)\in E} z_{(v,w)} && \forall (u,v)\in E \\
& && \gamma_{(u,v)} \leq y_{(v,w)} && \forall (v,w)\in E \\
& && \gamma_{(u,v)} \leq g + M\left(\sum_{w\in V} z_{(v,w)}\right) \\
&\text{Decision variables} && y_e = \begin{cases} 1 & e \text{ an edge on the PD path} \\ 0 & e \text{ an edge not on the PD path} \end{cases} \\
& && x_v = \begin{cases} 1 & v \text{ a vertex on the PD path} \\ 0 & v \text{ a vertex not on the PD path} \end{cases} \\
& && z_e = \begin{cases} 1 & \gamma_e = 1 \\ 0 & \gamma_e \neq 1 \end{cases} \\
&\text{Parameters} && l_v = \begin{cases} 1 & v \text{ is a leaf} \\ 0 & v \text{ is not a leaf} \end{cases} \\
& && p(e) && \text{Inheritance proportion function} \\
& && g && g \text{ is calculated in advance}
\end{aligned}
\tag{3.2}
$$

The new constraints, decision variables and parameters in this ILP have been explained in the previous sections, but there is still one thing to note about the formulation.
The final constraint is meant to be used for calculating only the edge above the tree vertex that starts a cycle. It does satisfy all other edges however. Edges e that have a reticulation below them always satisfy this, because there $\gamma_e \leq \gamma_f$, where f is the edge going out of the reticulation, because $p(e) \in (0,1)$. For edges above a tree vertex this also works. For any edge in the network, the only γ values in the interval $(0,1)$ that are unequal to g are the values x and y from table 3.2. Since $g = x + y - xy$ and $y - xy \geq 0$, because $x \in [0,1]$, it is clear that

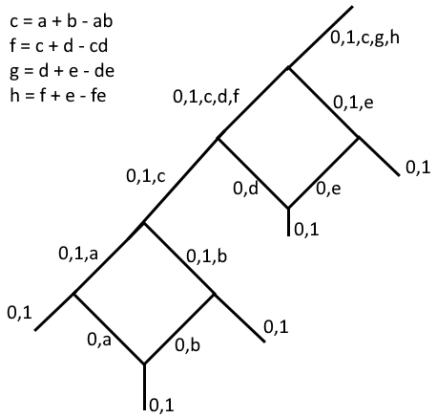


Figure 3.3: Example of a network with more than one reticulation and the options for γ_e on each edge

$x \leq g$.Therefore it will always satisfy the final constraint.
This means it is not necessary in the formulation to specify that this constraint is only used for edges above tree vertices that start cycles.

3.3.3. ILP formulation for networks with multiple reticulations

For one reticulation there was at most one edge for which the γ-function could not be calculated using the maximum of the edges below it. For m reticulations the number of these edges is bounded by $2m-1$. Unfortunately it is not possible to use the same construction as for networks with one reticulation.
The reason that it was possible to do so with one reticulation, is because the edge only had a single possible value in the interval $(0,1)$. This value could then be calculated in advance and used in the constraints with a trick of M being a large number that controlled whether this constraint would have to be used.
When there are multiple reticulations, this is no longer one possible value in the interval $(0,1)$, but multiple possible values. For example, in figure 3.3, the highest edge now has three values in the interval $(0,1)$ that it can attain. Therefore it no longer works by just having the one constraint for a value in the interval $(0,1)$.

For networks with multiple reticulations other solutions need to be found. This thesis will not go into depth about these solutions, but in appendix A two possible solutions are introduced and partially worked out.
The first solution works by bounding the Network-PD function. Therefore by maximizing those bounds, an approximation for the optimal subset for Max-Network-PD can be found.
The second solution uses the same basis as the approach for a network with one reticulation, but it forces edges below reticulations to be on the PD path or not. That reduces the number of options per edge above that reticulation. This might only be possible for level-1 networks however. This construction hopefully makes it possible to calculate the γ value for every edge using linear constraints. This will, however, need the ILP to be run 2^m times, if there are m reticulations. Once for every combination of fixed reticulations.

4

Conclusion and discussion

4.1. Conclusion

The goal in this thesis was to find an ILP formulation for the problem: For a given phylogenetic network $\mathcal{N}$ and leaf set X, find subset S of X with cardinality k, such that S has maximum phylogenetic diversity. This problem has been split into two problems. The Max-AllPath-PD problem and Max-Network-PD problem.
For Max-AllPath-PD an ILP formulation has been found. It is based on two sets of decision variables, that check whether vertices and edges are on the path between one of the leaves in set S and the root node. This path was called the PD path.
The ILP was first formulated for tree networks, but ended up working for all phylogenetic networks.
For Max-Network-PD there were more problems. Max-Network-PD on trees would result in the same problem as Max-AllPath-PD on trees, so that would not get any new information, but Max-Network-PD on networks did give a new ILP. Since $\gamma(S,e)$ can not be directly translated to decision variables with linear dependence, it was necessary to look at the possible values this $\gamma(S,e)$ could attain at each edge. To be able to do that, first a network with only one reticulation was considered. Here the different options for $\gamma(S,e)$ only were a problem for tree vertices. Tree vertices that had at most one of their outgoing edges leading into a reticulation, could be described using the maximum. The maximum still is not a linear function but using three linear constraints, the maximum can be rewritten in a linear way. Only tree vertices where both outgoing edges led to a reticulation were a problem. Since this is only possible if both edges lead to the same reticulation in a network with one reticulation, the number of possible values of $\gamma(S,e)$ decreased, leaving it with three options. Therefore it was possible to rewrite it to a set of linear constraints for the edge above that tree vertex.

4.2. Recommendations for future research

Max-Network-PD for networks with more than one reticulations has not been investigated properly and is therefore a good place to start further research. Two ideas are stated within this thesis. The first will continue from the basis used with one reticulation. By fixing the reticulations, the number of possible values for all the $\gamma(S,e)$ decreases drastically. Likely this makes it possible to formulate it as an ILP for level-1 networks. It is not certain that it will work, since not all the possible ways of cycles leading into each other have been properly investigated. Besides that, the ILP would have to be run multiple times, since all combinations of fixed reticulations have to be run to check which result is actually the maximum.

The second idea takes another route. By bounding the Network-PD function from below and above, it would make two new optimization problems. The ILP for the bound from above is already worked out, but the ILP for the bound for below has not been formulated. If this would be possible, it still does not solve the original Max-Network-PD problem for certain. That is because a different set S might maximize the upper bound or lower bound. Therefore the new problem would have to combine the two bounds whenever different sets S are the result.

Due to lack of time and a focus on other parts of the research, some parts have been left out. The first and probably most important part is that the ILP's have not been tested. Mainly for the Max-Network-PD ILP's it might be useful to test it using ILP solvers and extreme examples to test any possible case.

Something else that has not been taken into account is the biological background of the problem. If the goal is to preserve diversity in species that would be able to get new descendants with hybridization, it might be desirable to have two species that have a maximum diversity with respect to each other, such that they are compatible to have descendants.

A last point of interest that has not been looked at is the possibility of different phylogenetic score functions. This thesis has looked at AllPath-PD, which had a direct correlation to Faith's approach for trees in [1] and Network-PD, which was more realistic when looking at networks with reticulations. New functions can of course be made that might look at the species in a different way, since $p(e)$ and therefore $\gamma(S,e)$ could be very hard to determine for each edge. The weights for each edge might not be available either, so in that case another function must be used.

Bibliography

[1] Faith, D. (1992). Conservation evaluation and phylogenetic diversity. *Biological conservation, vol.* 61 (Issue 1), 1-10. https://doi.org/10.1016/0006-3207(92)91201-3

[2] Bordewich, M., Semple, C., Wicke, K. (2021). On the complexity of optimising variants of phylogenetic diversity on phylogenetic networks. https://arxiv.org/abs/2107.07834

Cover image by Robert Spencer.
https://www.newcivilengineer.com/latest/infrastructure-has-the-potential-to-enhance-biodiversity-15-04-2021/

A

Appendices

A.1. Approaching Network-PD with bounds

Since Network-PD has non-linear aspects, it might be beneficial to find functions that bound Network-PD from above and below. Then by combining the ILP's of those two functions an approximation for the optimal subset can be found for the Max-Network-PD problem. It is still possible that it is not the optimal subset, since the functions might have different sets S as their optimal set.
In [2] there are two functions that bound Network-PD. These functions are:

$$\text{MaxWeightTree-PD}_{\mathcal{N}}(S) = \max_{T \in \mathrm{T}_S(\mathcal{N})} \sum_{e \in T} w(e)$$

and:

$$\text{MinWeightTree-PD}_{\mathcal{N}}(S) = \min_{T \in \mathrm{T}_S(\mathcal{N})} \sum_{e \in T} w(e)$$

These functions consider the set of phylogenetic subtrees $\mathrm{T}_S(\mathcal{N})$ that have vertices in S as their only leaves. Then on the subtree with maximum or minimum weight respectively AllPath-PD is used. Proof that Network-PD is bounded by these functions can be found in [2].

A.1.1. ILP representation of MaxWeightTree-PD on networks

Let $\mathcal{N}$ be a rooted binary phylogenetic network with vertex set V, corresponding and set E for its edges. Here the weights of the edges are written as $w(e)$. Each edge can be seen as an ordered pair of 2 vertices, which are its starting and endpoint, $e = (u, v), \{u, v\} \in V$. We can formulate an ILP for maximizing the All-Path-PD score of a subset of species with cardinality at most k:

$$\begin{aligned}
&\text{Maximize} && \sum_{e \in E} y_e w(e) \\
&\text{Subject to} && \sum_{v \in V} l_v x_v \leq k \\
& && \sum_{u \in V} y_{(u,v)} = x_v && \forall u, v \in V \\
& && x_u \geq y_{(u,v)} && \forall u, v \in V \\
& && \sum_{u \in V} y_{(v,u)} \geq x_v(1 - l_v) && \forall v \in V \\
&\text{Decision variables} && y_e = \begin{cases} 1 & e \text{ an edge on the PD path} \\ 0 & e \text{ an edge not on the PD path} \end{cases} \\
& && x_v = \begin{cases} 1 & v \text{ a node on the PD path} \\ 0 & v \text{ a node not on the PD path} \end{cases} \\
&\text{Parameters} && l_v = \begin{cases} 1 & v \text{ is a leaf} \\ 0 & v \text{ is not a leaf} \end{cases}
\end{aligned} \tag{A.1}$$

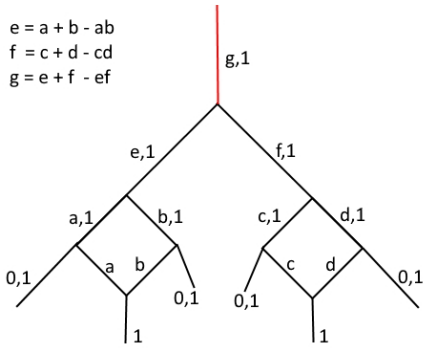


Figure A.1: Example of a network with more than one reticulation and the options for $\gamma(S,e)$ on each edge. Both reticulations are fixed. Therefore this network would have to run the ILP four times with different fixed values.

The ILP formulation of the MaxWeightTree-PD function is very similar to the ILP formulation of AllPaths-PD, since it only has one constraint changed. The second constraint now adds up y_e of all incoming edges. This way only one edge will be used for reticulations if the node is on the PD path, or no edges when it is not on the path. For tree vertices the constraint still works in the same way, since there is only one incoming edge.

A.1.2. Using the bounds for Network-PD

Finding an ILP representation for MinWeightTree-PD is not as easy as for MaxWeightTree-PD. This is because the problem has a minimization within the maximization. Therefore this idea has not been worked out in this thesis any further.
The upper bound that comes from MaxWeightTree-PD can still be used, but it would still be possible that the optimal set for that function would be the least optimal set for MinWeightTree-PD. Therefore the actual value for Network-PD(S) with that set is not necessarily the maximum score for Network-PD.
If an ILP formulation for MinWeightTree-PD were to be found, it would still not have certainty to select the best set S for Network-PD. By using a combination of MinWeightTree-PD and MaxWeightTree-PD, however, a set can be selected that will have a decently high PD score, depending on how these two bounds are combined.

A.2. Max-Network-PD by fixing reticulations

The method that was used for Max-Network-PD on networks with 1 reticulation, explained in section 3.2, does not work for multiple reticulations. It might be possible to force some reticulations to be used. In that case the number of options decreases, which might make it possible to express the γ value with linear constraints again.
This works by forcing some leaf below the reticulation to be in the set S. Therefore $\gamma(S,f)$ of the outgoing edge from that reticulation will be equal to 1. Then the incoming edges for that reticulations will have fixed values $\gamma(S,e)$ as well. This makes it such that the number of options for edges above these fixed edges decrease as well. Due to lack of time, not all options have been explored, but as an example, look at figure A.1. Here the number of options have decreased enough. Therefore the γ value of the red edge can be found using linear constraints.
The downside of this approach is that the ILP has to be run multiple times. For each reticulation the ILP has to be run once when the reticulation has at least one leaf below it in the set S and it has to be run once when no leaves below it are in set S, so when the outgoing edge has $\gamma(S,f)=0$. For n reticulations, the ILP has to be run 2^n times.