Technical Report CUCS-033-93

Minimal Path Length of Trees with Known Fringe

Roberto De Prisco* **Giuseppe Parlati***

Giuseppe Persiano†

Abstract

In this paper we continue the study of the path length of trees with known fringe as initiated by [1] and [2]. We compute the path length of the minimal tree with given number of leaves N and fringe Δ for the case $\Delta \geq N/2$. This complements the result of [2] that studied the case $\Delta \leq N/2$. Our methods also yields a linear time algorithm for constructing the minimal tree when $\Delta \geq N/2$.

1 Introduction

The path length of a tree is the sum of the length of all root-leaf paths and it is an important measure of efficiency. Given the number of leaves N, it is well known that the path length of an extended binary tree is $\Theta(N \log N)$ in the best case and $\Theta(N^2)$ in the worst case.

Because of this large gap, it is an important problem to study the path length of a binary tree when additional information on the tree is available or the tree is of some special form (see, for example [4, 5]).

Klein and Wood [1] were the first to consider the case in which, besides the number N of nodes, the *fringe* Δ (i.e., the difference between the longest and the shortest root-leaf path) is known. They gave an upper bound that, when $\Delta \leq \sqrt{N}$, could be met up to a factor proportional to N.

De Santis and Persiano [3] improved on this result by giving an upper bound achievable for infinitely many values of N and Δ. Subsequently [2], they started the study of the minimal path length for given N and Δ. More precisely, they gave an expression of the minimal path length for the case $\Delta \leq N/2$.

In this paper we extend their result providing a closed formula of the minimal path length for the case $\Delta \geq N/2$. Our techniques are conceptually different from those of [2] and they enable us to construct the minimal tree in linear time.

*Department of Computer Science, Columbia University, New York, N.Y. 10027.

†Dipartimento di Matematica, Università di Catania, Catania, Italy. Part of this work was done while at Aiken Comp. Lab., Harvard University, Cambridge, MA 02138.

2 Preliminaries

In this section we give the definitions that we need to formally present our results.

An *extended* binary tree T is a rooted binary tree where each node has zero or two children. Nodes without children are *leaves* and nodes with two children are *internal*. We denote the number of leaves of a tree T with $\mathcal{N}(T)$. Throughout this paper we refer to an extended binary tree simply as a tree and we will consider only trees with at least two leaves. The *level* of a node in a tree T is defined as the length of the unique path from the root to that node. Let T be a tree with leaves at levels $l_1, ..., l_{\mathcal{N}(T)}$. The *path length* $\mathtt{PL}(T)$ of T is

$$\mathtt{PL}(T) = \sum_{i=1}^{\mathcal{N}(T)} l_i.$$

Instead, the *fringe* $\mathcal{D}(T)$ of T is the difference between the longest and the shortest root-leaf paths, that is

$$\mathcal{D}(T) = \mathtt{Ml}(T) - \mathtt{ml}(T)$$

where $\mathtt{Ml}(T)$ and $\mathtt{ml}(T)$ are the maximum and the minimum of $\{l_1, ..., l_{\mathcal{N}(T)}\}$, respectively.

Definition 1 *The length vector of a tree T is the vector $\underline{n}(T) = (n_1, n_2, \cdots, n_k)$ where n_i is the number of leaves at level i in T and $k = \mathtt{Ml}(T)$.*

We say that two trees T_1 and T_2 are *isomorphic* if and only if $\underline{n}(T_1) = \underline{n}(T_2)$. For our purposes for each set of isomorphic trees we will focus our attention on one selected tree. In particular, our choice is to consider only the unique tree which has the following property: for each internal node u, and for each pair of leaves v and z in the left and right subtree of u respectively, the level of v is less or equal to the level of z. Roughly speaking, such a tree has at each level all the leaves on the left and all the internal nodes on the right. The notation $\underline{n} = (n_1, \cdots, n_{i-1}, z^{i-j}, n_j, \cdots, n_k)$ means that $n_i = n_{i+1} = ... = n_{j-1} = z$.

Definition 2 *$\mathcal{C}(N, \Delta)$ is the set of trees with N leaves and fringe Δ, that is*

$$\mathcal{C}(N, \Delta) = \{T | \mathcal{N}(T) = N \text{ and } \mathcal{D}(T) = \Delta\}$$

A tree in $\mathcal{C}(N, \Delta)$ is called an (N, Δ)-tree. We will say that a pair of integers (N,Δ) is *admissible* if $N \geq 2$, $1 < \Delta < N - 2$ and there exists at least one (N, Δ)-tree. For technical reasons, we have chosen not to consider the cases $\Delta = 0, 1, N - 2$. We remark, though, that for such values of Δ there exists a unique tree and thus the problem of upper and lower bounding its path length is trivial.

Given a set S of trees, a tree T of S is said *minimal for S* if and only if it has the smallest path length among the trees of S. When S is clear from the context, we will just say minimal tree instead of minimal tree for S.

Notice that for a (N, Δ)-tree with length vector $\underline{n}$ have that $\sum_i^k n_i = N$, $n_i = 0$ for $i = 1, 2, ..., k - \Delta - 1$, $n_{k-\Delta} > 0$ and $n_k > 0$.

A useful result about binary trees is the *Kraft equality* (see [6]): for any tree T with leaves at levels $l_1, ..., l_{\mathcal{N}(T)}$, we have that $\sum_{i=1}^{\mathcal{N}(T)} 2^{-l_i} = 1$.

3 Constructing the minimal tree

In order to study the minimal tree for the class $\mathcal{C}(N,\Delta)$ we define a partition of this set based on the value of $\mathtt{ml}(T)$ and we construct the minimal tree for each subset of the partition. Then, the minimal tree in $\mathcal{C}(N,\Delta)$ is obtained by comparing the minimal trees of the subclasses.
Define $L_{min}(N,\Delta) = \lceil \log(N + 2^{\Delta} - 1)\rceil - \Delta$ and $L_{max}(N,\Delta) = \lfloor \log(N-\Delta)\rfloor$ (throughout this paper all logarithms are base 2). Then the following lemma holds.

Lemma 1 *For any (N,Δ)-tree T we have that*

$$L_{min}(N,\Delta) \leq \mathtt{ml}(T) \leq L_{max}(N,\Delta).$$

Proof. Consider the trees with fringe Δ and minimal leaf level L. Any such tree has no leaves on levels $1, 2, \cdots, L-1$ and at least one leaf on level L. Thus, the tree with fringe Δ, minimal leaf level L, and the greatest number of leaves has exactly $1 + (2^L - 1)2^{\Delta}$ leaves. On the other hand the tree with fringe Δ, minimal leaf level L, and the smallest number of leaves has exactly $2^L + \Delta$ leaves. Hence for any tree with fringe Δ, minimal leaf level L and with N leaves, we have that $2^L + \Delta \leq N \leq 1 + (2^L - 1)2^{\Delta}$. These two inequalities prove the lemma. ■

A triplet of integers (N,Δ,L) is *admissible* if (N,Δ) is admissible and $L_{min}(N,\Delta) \leq L \leq L_{max}(N,\Delta)$.

Definition 3 *For admissible (N,Δ,L), we define the subset $\mathcal{C}(N,\Delta,L)$ of $\mathcal{C}(N,\Delta)$ as*

$$\mathcal{C}(N,\Delta,L) = \{T \in \mathcal{C}(N,\Delta) | \mathtt{ml}(T) = L\}.$$

It is immediate to see that the sets $\mathcal{C}(N,\Delta,L)$ constitute a partition of $\mathcal{C}(N,\Delta)$. A tree in $\mathcal{C}(N,\Delta,L)$ is called a (N,Δ,L)-tree.
Now we define a particular tree that will be useful to study the minimal tree for $\mathcal{C}(N,\Delta,L)$.

Definition 4 *For $N \geq 2^L$, define the skeleton tree $S(N,L)$ as the tree described by the length vector*

$$\underline{n}(S(N,L)) = (0^{L-1}, 2^L - 1, 1^{N-2^L-1}, 2).$$

Notice that the skeleton $S(N,L)$ has N leaves and fringe $\mathcal{D}(S(N,L)) = N - 2^L$.

3.1 Minimality in $\mathcal{C}(N,\Delta,L)$

In this section we provide an algorithm that constructs the minimal tree for $\mathcal{C}(N,\Delta,L)$.
Before going any further, we introduce two operations on a tree that will be useful to describe the algorithm. A node u of T is called a *bush* if both its children are leaves. A *cut* operation **cut** (u,T) on a bush u of a tree T deletes the two leaves which are children of u and make node u leaf. The tree T' obtained by performing a cut of a bush u at level ℓ_u in the tree T has path length

$$\mathtt{PL}(T') = \mathtt{PL}(T) - \ell_u - 2.$$

An *insert* operation **ins** (v,T) on a leaf v of tree T makes v internal and inserts two leaves as children of v. The tree T' obtained by performing an insert on a leaf v at level ℓ_v in the tree T has path length

$$\mathtt{PL}(T') = \mathtt{PL}(T) + \ell_v + 2$$

Now, we are ready to describe the algorithm `Min-L` that, given an admissible triplet (N, Δ, L), constructs the minimal tree for $\mathcal{C}(N, \Delta, L)$.
Informally speaking, the algorithm starts from the skeleton $S(N, L)$. Recall that this tree has $2^L - 1$ leaves at level L, exactly one leaf on levels $L+1, L+2, ..., L+\Delta-1$ and two leaves on level $L+\Delta$. The fringe of the skeleton $S(N, L)$ is $N - 2^L$; notice that for any Δ such that (N, Δ, L) is admissible, the fringe of the skeleton is at least Δ. The algorithm `Min-L` performs exactly $N - \Delta - 2^L$ iterations and in each iteration it performs one cut and one insert in such a way that the fringe decreases by one and the path length decreases as much as possible. This means that the cut must be performed on the deepest bush (notice that the fringe decreases by one upon each cut) and the insert must be performed on the highest level that has at least one leaf, taking in account that the insert operation must not modify the fringe (i.e., there must be at least one leaf on level L). Since the number of leaves is kept constant (the number of cut and the number of insert are the same), at least one leaf is left on level L and the fringe is decreased from $N - 2^L$ to Δ, the algorithm returns an (N, Δ, L)-tree. Moreover, as we will see in the following this tree is minimal for $\mathcal{C}(N, \Delta, L)$ since each step of the construction is performed in such a way that the contribution to the path length is minimized. Figure 1 illustrates the first step of the algorithm and the final result.

```
Min-L(N, Δ, L)

if ( (N > (2^L − 1) · 2^Δ) or (N < 2^L + Δ) )
    return ( Input Error );
endif
T ← S(N, L);
for (i = 1 to N − 2^L − Δ) do
    u ← deepest bush ; //at level N − 2^L + L − i//
    cut (u, T);
    if ( n_L(T) > 1 )
        j ← L;
    else
        j ← smallest integer z > L s.t. n_z(T) > 0;
    endif
    v ← rightmost leaf at level j;
    ins (v, T);
endfor
return (T);
```

Given an admissible triplet (N, Δ, L), we say that a (N, Δ, L)-tree T is of *type 1* if there exist an integer h such that the length vector of T is $(0^{L-1}, 1, 0^h, a, b, 1^{\Delta-h-3}, 2)$. The values of a and b are uniquely determined by the constraint that the number of leaves is N and by the Kraft equality. We say that T is of *type 2* if its length vector is $(0^{L-1}, a, b, 1^{\Delta-2}, 2)$ and of *type 3* if its length vector is $(0^{L-1}, 1, 0^{\Delta-2}, a, b)$.

Let $\mathcal{T}_L(N, \Delta)$ be the tree output by `Min-L` on input N, Δ, L. In the following we will just write $\mathcal{T}_L$ instead of $\mathcal{T}_L(N, \Delta)$.

Lemma 2 *For any admissible triplet (N, Δ, L), $\mathcal{T}_L$ belongs to $\mathcal{C}(N, \Delta, L)$ and it is of type 1,2 or 3. Moreover, no other tree in $\mathcal{C}(N, \Delta, L)$ is of type 1,2 or 3.*

Proof. It is easy to see that $\mathcal{T}_L$ is a (N, Δ, L)-tree and it is of type 1,2 or 3. Hence we have to show that no other tree in $\mathcal{C}(N, \Delta, L)$ is of type 1,2 or 3.

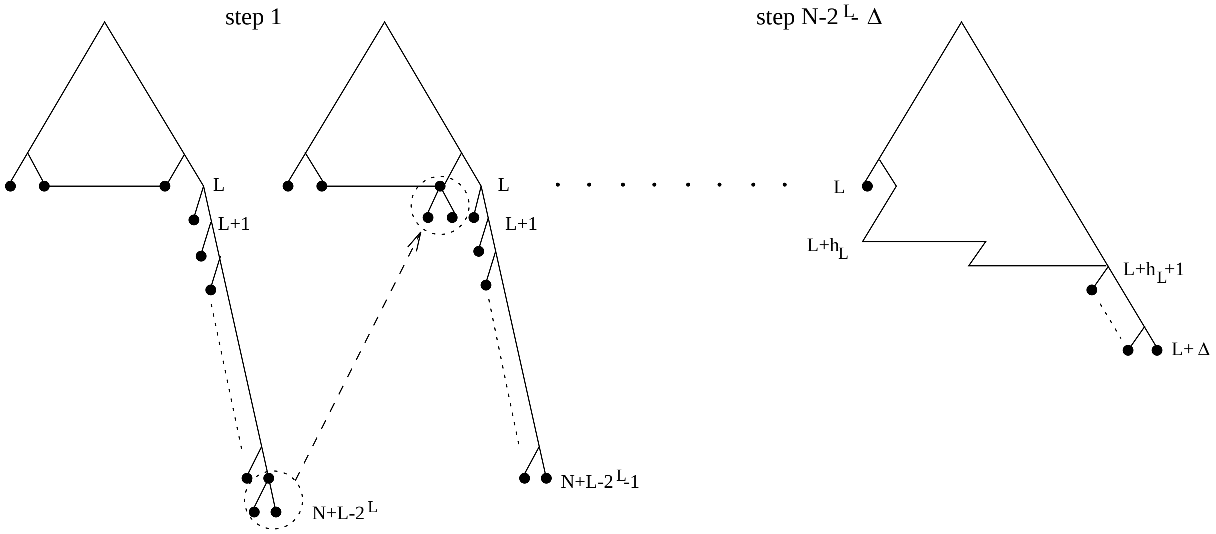

Figure 1: The first step and the output of algorithm `Min-L`

Let us consider the case when $\mathcal{T}_L$ is of type 1; i.e., there exist a, b and h such that $\underline{n}(\mathcal{T}_L) = (0^{L-1}, 1, 0^h, a, b, 1^{\Delta-h-3}, 2)$. Then we prove that no other tree T can be of type 1. When $\mathcal{T}_L$ is of type 2 or 3, the proofs are similar.

For sake of contradiction suppose that there exists a (N, Δ, L)-tree T different from $\mathcal{T}_L$ that is of type 1; i.e., there exist a', b' and h' such that $\underline{n}(T) = (0^{L-1}, 1, 0^{h'}, a', b', 1^{\Delta-h'-3}, 2)$ with $h \neq h'$ (if $h = h'$ then, by Kraft equality, $a' = a$ and $b' = b$ and thus $T = \mathcal{T}_L$). We show that T cannot be in $\mathcal{C}(N, \Delta, L)$.

Suppose that $h' < h$ and let $c = h - h'$. When $h' > h$ the proof is similar. We have that $\mathcal{N}(\mathcal{T}_L) \geq (2^L - 1)2^{h+1} + \Delta - h - 1$ and $\mathcal{N}(T) \leq (2^L - 1)2^{h'+1} + \Delta - h'$. Hence, $\mathcal{N}(\mathcal{T}_L) - \mathcal{N}(T) \geq (2^L - 1)2^{h'+1}(2^c - 1) - c - 1$ which is always greater than zero. Thus T has more than N leaves that is a contradiction. ■

Lemma 3 *For all admissible triplets* (N, Δ, L)*,* $\mathcal{T}_L$ *is minimal for* $\mathcal{C}(N, \Delta, L)$*.*

Proof. By Lemma 2, $\mathcal{T}_L$ belongs to $\mathcal{C}(N, \Delta, L)$. Let $T \in \mathcal{C}(N, \Delta, L)$ be a tree different from $\mathcal{T}_L$. We show that T has not the minimal path length.

By Lemma 2, the tree T cannot be of type 1,2 or 3. Hence, in the tree T there exists a leaf u and a bush v such that, denoted by l_u and l_v their levels, we have that $l_u < l_v$. Then, by performing a **cut** (v, T) and a **ins** (u, T), we obtain a tree $T' \in \mathcal{C}(N, \Delta, L)$ whose path length is

$$\mathtt{PL}(T') = \mathtt{PL}(T) - l_v - 2 + l_u + 2 < \mathtt{PL}(T).$$

Thus T can not have minimal path length. ■

Let us define the foliage-height of $\mathcal{T}_L$. The foliage-height will be fundamental in deriving the bound. The foliage is the set of nodes inserted into $\mathcal{T}_L$ during the execution of `Min-L`. Since the algorithm performs an insertion on the highest level available we have that the leaves of the foliage will be placed on at most two consecutive levels. We denote by H the highest of these two level. The integer $h_L = H - L$ is called the *foliage-height*; h_L is a function of N, Δ and L, but we will refer to it with the notation h_L emphasizing the dependence from L.

In the following we study some properties of $\mathcal{T}_L$ that will be useful in deriving the lower bound.

Lemma 4 *For each k, $0 \leq k < \Delta$, the level $L+k$ of the tree $\mathcal{T}_L$ has at most $2^{L+k} - 2^k - 1$ leaves.*

Proof. The tree $\mathcal{T}_L$, has at least on leaf on level L. Hence, at level $L+k$ we can have at most $(2^L - 1)2^k$ nodes. Moreover, also deeper levels have leaves, then at least one of the nodes at level $L+k$ must be an internal node. ■

For each k, $0 \leq k < \Delta$, we denote by $\mathcal{F}(k)$ be the number of steps performed by the algorithm `Min-L`, after which insertions on levels $L, L+1, ..., L+k$ are not possible. In other words $\mathcal{F}(k)$ is the number of iterations that the algorithm `Min-L` performs until it "fills" the level $L+k$ of $\mathcal{T}_L$. Notice that level L is filled if it has only one leaf, instead levels $L+k$, $k = 0, 1, ..., \Delta - 1$, are filled if they have no leaves. Moreover, we set $\mathcal{F}(-1) = 0$ since the level $L-1$ is already filled.

Notice that when level $L+k$ is filled the foliage-height is $k+1$.

Lemma 5 *For any admissible triplet (N, Δ, L) and for k such that $-1 \leq k < \Delta$ we have that*

$$\mathcal{F}(k) = (2^L - 1) \cdot (2^{k+1} - 1) - k - 1$$

Proof. Observe that the level $L+k$ is filled when the algorithm `Min-L` has performed an ins operation for each but one leaf on level L and for each leaf of levels between $L+1$ and $L+k$. By Lemma 4 we have that

$$\mathcal{F}(k) = \sum_{i=0}^{k} (2^{L+i} - 2^i - 1) = (2^L - 1) \cdot (2^{k+1} - 1) - k - 1.$$

■

Lemma 6 *For any admissible triplet (N, Δ, L), h_L is the greatest integer z such that $2^{L+z} - 2^z - z < N - \Delta$.*

Proof. Let us recall that `Min-L` performs $N - \Delta - 2^L$ insertions. By definitions of h_L and $\mathcal{F}(k)$ we have that

$$\mathcal{F}(h_L - 1) \leq N - \Delta - 2^L < \mathcal{F}(h_L).$$

Thus

$$2^{L+h_L} - 2^{h_L} - h_L + 1 \leq N - \Delta < 2^{L+h_L+1} - 2^{h_L+1} - h_L.$$

■

Lemma 7 *For all $L_{min}(N, \Delta) \leq \ell < L_{max}(N, \Delta)$ we have that $h_{\ell+1}$ is either $h_\ell - 1$ or $h_\ell - 2$.*

Proof. From Lemma 6 it follows

$$2^{h_\ell}(2^\ell - 1) - h_\ell < N - \Delta \leq 2^{h_\ell+1}(2^\ell - 1) - h_\ell - 1 \tag{1}$$

$$2^{h_{\ell+1}}(2^{\ell+1} - 1) - h_{\ell+1} < N - \Delta \leq 2^{h_{\ell+1}+1}(2^{\ell+1} - 1) - h_{\ell+1} - 1 \tag{2}$$

First of all observe that, for all constants $A > 1$, the function $A2^x - x$ is increasing in x. Suppose for sake of contradiction that $h_{\ell+1} \geq h_\ell$. Then by (2) we have that

$$\begin{aligned}
N - \Delta &> 2^{h_{\ell+1}}(2^{\ell+1} - 1) - h_{\ell+1} \\
&\quad (\text{since } h_{\ell+1} \geq h_\ell) \\
&> 2^{h_\ell}(2^{\ell+1} - 1) - h_\ell \\
&\quad (\text{since } -2^{h_\ell} - 1 < 0) \\
&> 2^{h_\ell+1}(2^\ell - 1) - h_\ell - 1 \\
&\quad (\text{by } (1)) \\
&\geq N - \Delta.
\end{aligned}$$

Contradiction.
Second, again for sake of contradiction, suppose that $h_{\ell+1} < h_\ell - 2$ (hence $h_{\ell+1} + 1 \leq h_\ell - 2$) then by (2) we have that

$$\begin{aligned}
N - \Delta &\leq 2^{h_{\ell+1}+1}(2^{\ell+1} - 1) - (h_{\ell+1} + 1) \\
&\quad (\text{since } h_{\ell+1} + 1 \leq h_\ell - 2) \\
&< 2^{h_\ell-2}(2^{\ell+1} - 1) - (h_\ell - 2) \\
&\quad (\text{since } 2^{\ell+h_\ell} - 3 \cdot 2^{h_\ell-1} - 2 > 0) \\
&< 2^{h_\ell}(2^\ell - 1) - h_\ell \\
&\quad (\text{by } (1)) \\
&< N - \Delta
\end{aligned}$$

Contradiction. ■

Lemma 8 *The path length of the minimal tree in $\mathcal{C}(N, \Delta, L)$ is*

$$\mathtt{PL}(\mathcal{T}_L) = N(L + h_L + 2) - 2^{h_L+1}(2^L - 1) - \Delta - 2 + \frac{(\Delta - h_L)(\Delta - h_L + 1)}{2}.$$

Proof. We prove the lemma for the case in which $\mathcal{T}_L$ is of type 1. The other cases are similar.

Let x be the number of nodes at level $L + h_L$ of the tree $\mathcal{T}_L$ having exactly two leaves as children. It is easy to see that the number of leaves is $N = 2^{h_L}(2^L - 1) - h_L + \Delta + x + 1$. Hence we get $x = N - 2^{h_L}(2^L - 1) + h_L - \Delta - 1$. Observe that $\mathcal{T}_L$ has one leaf at level L, $(2^{h_L}(2^L - 1) - x - 1)$ leaves at level $L + h_L$, $2x + 1$ leaves at level $L + h_L + 1$, one leaf on levels $L + h_L + 2, \ldots, L + \Delta - 1$ and two leaves on level $L + \Delta$. Hence we have that

$$\mathtt{PL}(\mathcal{T}_L) = L + (L + h_L)(2^{h_L}(2^L - 1) - x - 1) + (L + h_L + 1)2x + \sum_{i=L+h_L+1}^{L+\Delta} i + L + \Delta.$$

By simple algebraic manipulations we get the lemma.

■

The following lemmas provide the value of h_L as a function of N, Δ for the cases $L = 1, 2$. These values will be used in Section 4 to derive our lower bound.

Lemma 9 *Let (N, Δ) be such that $(N, \Delta, 1)$ is admissible. Then the foliage-height of $\mathcal{T}_1$ is*

$$h_1 = \lfloor \log(N - \Delta + \lfloor \log(N - \Delta) \rfloor) \rfloor .$$

Proof. Denote by d the number $N - \Delta - 2$ of iterations performed by `Min-L` when $L = 1$. Moreover, let k be the unique integer such that $2^k \le d + 2 < 2^{k+1}$. We study the function

$$\phi(x) = \lfloor \log(x + 2 + \lfloor \log(x + 2) \rfloor) \rfloor$$

and show that $h_1 = \phi(d)$ thus proving the lemma. Since $\lfloor \log(d + 2) \rfloor = k$, we have that $\phi(d) = \lfloor \log(d + 2 + k) \rfloor$. It is easy to see that $2^k < 2^k + k \le d + 2 + k < 2^{k+1} + k < 2^{k+2} - 1$. Hence $\phi(d)$ is equal to k or $k + 1$. In particular, we have that

$$\phi(d) = \begin{cases} k, & \text{if } 2^k < d + 2 + k < 2^{k+1}; \\ \\ k + 1, & \text{if } 2^{k+1} \le d + 2 + k < 2^{k+2} - 1. \end{cases}$$

By simple algebraic manipulations, and recalling that $\mathcal{F}(x) = 2^{x+1} - (x + 1) - 1$ we have that

$$\phi(d) = \begin{cases} k, & \text{if } \mathcal{F}(k - 1) \le d < \mathcal{F}(k); \\ \\ k + 1, & \text{if } \mathcal{F}(k) \le d < \mathcal{F}(k + 1). \end{cases}$$

Since d is the number of iterations performed by `Min-L`, by definition of $\mathcal{F}(k)$ we conclude that $h_1 = \phi(d)$. ■

Lemma 10 *Let (N, Δ) be such that $(N, \Delta, 2)$ is admissible. Then the foliage-height of $\mathcal{T}_2$ is*

$$h_2 = \left\lfloor \log \frac{N - \Delta - 2 + \lfloor \log(N - \Delta - 2) \rfloor}{3} \right\rfloor .$$

Proof. Denote by d the number $N - \Delta - 4$ of iterations performed by `Min-L` when $L = 2$. We study the function

$$\phi(x) = \left\lfloor \log \frac{x + 2 + \lfloor \log(x + 2) \rfloor}{3} \right\rfloor$$

and show that $\phi(d) = h_2$.

Assume $d \ge 10$ (for $d < 10$ the Lemma can be proved by inspection). Let k be the unique integer such that $3 \cdot 2^k \le d + 2 < 3 \cdot 2^{k+1}$. As $d \ge 10$ we have that $k \ge 2$. By the definition of $\mathcal{F}(\cdot)$, we have that $\mathcal{F}(k - 1) < 3 \cdot 2^k$ and $3 \cdot 2^{k+1} < \mathcal{F}(k + 1)$. Moreover $2^{k+2} < \mathcal{F}(k) < 3 \cdot 2^{k+1}$.

We distinguish between three possible cases in according to the value of $d + 2$.

CASE 1. $3 \cdot 2^k \le d + 2 < 2^{k+2}$. In this case we have that $\lfloor \log(d + 2) \rfloor = k + 1$. Hence $\phi(d) = \left\lfloor \log \frac{d+3+k}{3} \right\rfloor$. A simple algebra shows that $2^k \le \frac{d+3+k}{3} < 2^{k+1}$. Hence $\phi(d) = k$. On the other hand we have that in this case $\mathcal{F}(k - 1) \le d < \mathcal{F}(k)$, that means $h_2 = k = \phi(d)$.

CASE 2. $2^{k+2} \leq d+2 < \mathcal{F}(k)+2$. In this case we have that $\lfloor \log(d+2) \rfloor = k+2$. Hence $\phi(d) = \left\lfloor \log \frac{d+4+k}{3} \right\rfloor$. Simple algebra shows that $2^k \leq \frac{d+3+k}{3} < 2^{k+1}$. Hence $\phi(d) = k = \phi(d)$. Again in this case we have that $\mathcal{F}(k-1) \leq d < \mathcal{F}(k)$, that is $h_2 = k$.

CASE 3. $\mathcal{F}(k)+2 \leq d+2 < 3 \cdot 2^{k+1}$. In this case we have that $\lfloor \log(d+2) \rfloor = k+2$. Hence $\phi(d) = \left\lfloor \log \frac{d+4+k}{3} \right\rfloor$. Simple algebra shows that $2^{k+1} \leq \frac{d+3+k}{3} < 2^{k+2}$. Hence $\phi(d) = k+1$. On the other hand we have that in this case $\mathcal{F}(k) \leq d < \mathcal{F}(k+1)$, that means $h_2 = k+1 = \phi(d)$.

This proves that $\phi(d) = h_2$ and hence the lemma. ∎

3.2 Minimality in $\mathcal{C}(N, \Delta)$

Since the value of L lies between $L_{min}(N, \Delta)$ and $L_{max}(N, \Delta)$ the minimal path length among all the trees in $\mathcal{C}(N, \Delta)$ is given by

$$\min_{L_{min}(N,\Delta) \leq L \leq L_{max}(N,\Delta)} \mathtt{PL}(\mathcal{T}_L).$$

This enable us to obtain the minimal tree in an algorithmic fashion. This algorithmic construction of the minimal tree can be also used for the maximal tree obtaining an algorithm similar to the one of [7]. Notice that the range of variation of L is $O(\log(N - \Delta))$.

4 The Lower Bound

In this section we analyze $\mathtt{PL}(\mathcal{T}_L)$ as a function of L. We show that, when $\Delta \geq N/2$, $\mathtt{PL}(\mathcal{T}_L)$ is an increasing function of L and thus the minimum is obtained for $L_{min}(N, \Delta)$. However, if $\Delta \geq N/2$ then $L_{min}(N, \Delta) = 1$. By plugging in the value $L = 1$ and the expression for h_1 in the formula for the path length given by Lemma 8 we obtain our lower bound.

Next lemma proves that when $h_L = h_{L+1} + 2$ then $\mathcal{T}_{L+1}$ has greater path length than $\mathcal{T}_L$. The proof for the case for $h_L = h_{L+1} + 1$ requires more care.

Lemma 11 *For each admissible (N, Δ, L) such that $\Delta \geq N/2$, $L_{min}(N, \Delta) \leq L \leq L_{max}(N, \Delta) - 1$, and $h_L = h_{L+1} + 2$ we have that $\mathtt{PL}(\mathcal{T}_L) \leq \mathtt{PL}(\mathcal{T}_{L+1})$.*

Proof. Using Lemma 8 and the fact that $h_L = h_{L+1} + 2$ we have

$$\mathtt{PL}(\mathcal{T}_{L+1}) - \mathtt{PL}(\mathcal{T}_L) = 2\Delta - N + 2^{h_L + L} + 3 - 2^{h_L} - 2^{h_L - 1} - 2h_L.$$

It is easy to see that $2^{h_L+L} + 3 - 2^{h_L} - 2^{h_L-1} - 2h_L$ is always non negative. Therefore the above difference is positive since $\Delta \geq N/2$. ∎

We now undertake the study of the case $h_L = h_{L+1} + 1$. First we consider the case $L \geq 2$.

Lemma 12 *For each admissible (N, Δ, L) such that $\Delta \geq N/2$, $2 \leq L \leq L_{max}(N, \Delta) - 1$, and $h_L = h_{L+1} + 1$ we have that $\mathtt{PL}(\mathcal{T}_L) \leq \mathtt{PL}(\mathcal{T}_{L+1})$.*

Proof. Using Lemma 8 and the fact that $h_L = h_{L+1} + 1$ we have

$$\mathtt{PL}(\mathcal{T}_{L+1}) - \mathtt{PL}(\mathcal{T}_L) = \Delta + 1 - 2^{h_L} - h_L.$$

Since h_L is a decreasing function of L, the above difference is increasing with L. Thus $\mathtt{PL}(\mathcal{T}_{L+1}) - \mathtt{PL}(\mathcal{T}_L) \geq \Delta + 1 - 2^{h_2} - h_2 \geq N/2 + 1 - 2^{h_2} - h_2$. Using the expression for h_2 (see Lemma 10), we have that

$$\begin{aligned} 2^{h_2} + h_2 &\leq \frac{N - \Delta - 2 + \lfloor \log(N - \Delta - 2) \rfloor}{3} + \log \frac{N - \Delta - 2 + \lfloor \log(N - \Delta - 2) \rfloor}{3} \\ &\leq \frac{N}{6} + \frac{4}{3} \log(\frac{N}{2} - 2) + \frac{1}{3} - \log 3. \end{aligned}$$

Hence $\mathtt{PL}(\mathcal{T}_{L+1}) - \mathtt{PL}(\mathcal{T}_L) \geq \frac{N}{6} - \frac{4}{3}\log(\frac{N}{2} - 2) - \frac{2}{3} + \log 3$ that is positive for $N \geq 6$. Observe, though, that if $(N, \Delta, 2)$ is admissible then N is at least 6. ■

All it is left to prove is that $\mathtt{PL}(\mathcal{T}_1) \leq \mathtt{PL}(\mathcal{T}_2)$ when $h_1 = h_2 + 1$. We start by studying the relation between h_1 and h_2.

Lemma 13 *For each N and Δ such that $(N, \Delta, 1)$ is admissible we have that*

$$h_1 = \begin{cases} k, & \text{if } 0 \leq c < 2^k - k; \\ k+1, & \text{if } 2^k - k \leq c < 2^k; \end{cases}$$

where k and c are integers such that $N - \Delta = 2^k + c$ and $0 \leq c < 2^k$.

Proof. From Lemma 9 we have that

$$h_1 = \lfloor \log(N - \Delta + \lfloor \log(N - \Delta) \rfloor) \rfloor = \lfloor \log(2^k + c + \lfloor \log(2^k + c) \rfloor) \rfloor = \lfloor \log(2^k + c + k) \rfloor.$$

Observing that $2^k < 2^k + c + k$ and $2^k + c + k < 2^{k+2}$ we have that

$$h_1 = \begin{cases} k, & \text{if } 2^k < 2^k + c + k < 2^{k+1}; \\ k+1, & \text{if } 2^{k+1} \leq 2^k + c + k < 2^{k+2}. \end{cases}$$

Whence the lemma. ■

Lemma 14 *For each N and Δ such that $(N, \Delta, 2)$ is admissible we have that*

$$h_2 = \begin{cases} k-2, & \text{if } 0 \leq c \leq 2^{k-1} - k + 1; \\ k-1, & \text{if } 2^{k-1} - k + 2 \leq c < 2^k; \end{cases}$$

where k and c are integers such that $N - \Delta = 2^k + c$ and $0 \leq c < 2^k$.

Proof. Since $(N, \Delta, 2)$ is admissible, then $N - \Delta \geq 4$ (see the expression for $L_{max}(N, \Delta)$). This implies that $k \geq 2$. Using the expression for h_2 we can write

$$h_2 = \left\lfloor \log \frac{2^k + c - 2 + \lfloor \log(2^k + c - 2) \rfloor}{3} \right\rfloor.$$

But

$$\lfloor \log(2^k + c - 2) \rfloor = \begin{cases} k-1, & \text{if } c = 0, 1; \\ \\ k, & \text{otherwise.} \end{cases}$$

Thus one has

$$h_2 = \begin{cases} \lfloor \log \frac{2^k+k+c-3}{3} \rfloor, & \text{if } c = 0, 1; \\ \\ \lfloor \log \frac{2^k+k+c-2}{3} \rfloor, & \text{otherwise.} \end{cases}$$

Let us start by considering the case $c = 0, 1$. Then we have that, since $k \geq 2$,

$$2^{k-2} \leq \frac{2^k + k + c - 3}{3} < 2^{k-1}$$

whence $h_2 = k - 2$.

Now consider the case $2 \leq c \leq 2^{k-1} - k + 1$. Then we have,

$$2^{k-2} \leq \frac{2^k + k + c - 2}{3} < 2^{k-1}$$

which implies $h_2 = k - 2$.

Finally, consider the case $2^{k-1} - k + 2 \leq c \leq 2^k$. Then we have,

$$2^{k-1} \leq \frac{2^k + k + c - 2}{3} < 2^k.$$

Thus, $h_2 = k - 1$. ■

Lemma 15 *For any N, Δ such that $\Delta \geq N/2$, both $(N, \Delta, 1)$ and $(N, \Delta, 2)$ are admissible, and $h_1 = h_2 + 1$, we have that $\mathtt{PL}(\mathcal{T}_1) \leq \mathtt{PL}(\mathcal{T}_2)$.*

Proof. By Lemma 8 we have that

$$\mathtt{PL}(\mathcal{T}_2) - \mathtt{PL}(\mathcal{T}_1) = \Delta + 1 - 2^{h_1} - h_1.$$

Let k and c be the integers such that $N - \Delta = 2^k + c$, with $0 \leq c < 2^k$. Notice that, since $(N, \Delta, 2)$ is admissible, then $N - \Delta \geq 4$ and thus $k \geq 2$. By Lemma 13 and 14 we have that $h_1 - h_2 = 1$ if and only if $2^{k-1} - k + 2 \leq c \leq 2^k - k - 1$. In this case, $h_1 = k$ and thus

$$\begin{aligned} \mathtt{PL}(\mathcal{T}_2) - \mathtt{PL}(\mathcal{T}_1) &= \Delta + 1 - 2^k - k \\ &= \Delta + 1 - N + \Delta + c - k \\ &\geq 2\Delta - N + 2 + 2^{k-1} - 2k \\ &\geq 2\Delta - N, \end{aligned}$$

that is non negative. ■

Lemma 16 *Let (N, Δ) be such that $\Delta \geq N/2$. Then for all (N, Δ)-tree T we have $\mathtt{PL}(T) \geq \mathtt{PL}(\mathcal{T}_1)$.*

Proof. First observe that if $\Delta \geq N/2$ then $L_{min}(N, \Delta) = 1$ and thus $\mathcal{T}_1$ exists. Now let $L \geq 1$ be such that T is a (N, Δ, L)-tree. Then we have $\mathtt{PL}(T) \geq \mathtt{PL}(\mathcal{T}_L)$ and, by Lemmas 11,12 and 15, we have that $\mathtt{PL}(\mathcal{T}_L) \geq \mathtt{PL}(\mathcal{T}_1)$. ■

The above lemma gives a simple linear time algorithm for constructing the minimal tree for given N and $\Delta \geq N/2$. The algorithm consists in running `Min-L` on input N, Δ and $L = 1$.

Finally, we are ready to state our lower bound.

Theorem 1 *For any admissible pair (N, Δ) such that $\Delta \geq N/2$, the path length of the minimal (N, Δ)-tree is*

$$N(h+3) - 2^{h+1} - \Delta - 2 + \frac{(\Delta - h)(\Delta - h + 1)}{2},$$

where $h = \lfloor \log(N - \Delta + \lfloor \log(N - \Delta) \rfloor) \rfloor$.

Proof. By the previous lemma, plug in $L = 1$ and the expression for h_1 in the formula for the path length given by Lemma 8. ∎

5 Conclusions and open problems

In this paper we have closed the problem of studying the minimal path length of trees of given fringe. The case of the maximal tree is still open. We suspect that techniques similar to those developed in this paper might be useful also for the study of the maximal tree. Also it would be interesting to study the average path length of (N, Δ)-trees.

Acknowledgements. We would like to thank Alfredo De Santis for many useful discussions and suggestions. The first two authors thank Zvi Galil for his support at Columbia University.

References

[1] R. Klein and D. Wood, On the Path Length of Binary Trees, *Journal of ACM*, Vol.36, n.2, Apr 1989, pp. 280–289.

[2] A. De Santis and G. Persiano, Tight Upper and Lower Bounds on the Path Length of Binary Trees, *SIAM Journal on Computing*, Oct 1993, to appear.

[3] A. De Santis and G. Persiano, Tight Bounds on the Path Length of Binary Trees, in Proceedings of *8th Annual Symposium on Theoretical Aspects of Computer Science (STACS '91)*, Ed. C. Choffrut e M. Jantzen, vol. 480 of "Lecture Notes in Computer Science", Springer-Verlag, pp. 478–487, February 1991.

[4] R. Klein and D. Wood, A tight upper bound for the path length of AVL trees, *Theoretical Computer Science*, Vol. 72, 1990, pp.251–264

[5] J. Nievergelt and C.K. Wong, Upper bounds for the total path length of binary trees, *Journal of ACM*, Vol.20, n.1, Jan 1973, pp. 1–6.

[6] R.W. Hamming, *Coding and Information Theory*, Prentice-Hall, Inc., Englewood Cliffs, N.J., 1980.

[7] H. Cameron and D. Wood, The Maximal Path Length of Binary Trees, manuscript.