An Algorithmic Approach to Generating A Specific Infinite Binary Tree

William Wodrich, Aaliyah Fiala, Dr. Daniel Rockwell

Oregon State University

Abstract

Our interactive presentation shows the expansion of a simple idea into an infinitely complex pattern. Interacting with our research using various applications and graphical displays, will give you an in depth view of concepts such as infinity, binary trees, and convergence. When you are listening to our presentation, you are not just a listener, but can easily become a researcher, discovering patterns in these intricate mathematical structures.

This research project takes a new approach to the process of generating an infinite binary tree. In our process of searching for more efficient algorithms, we have discovered numerous connections just by playing around, and finding ways to display our data. If you would like to gain insight and assist in the uncovering of connections and patterns, come and interact with our binary trees!

Background

What is a binary word?

A binary word is simply a sequence of 0's and 1's. Combinations of these binary words of length n in lexicographical order forms a binary list. Two binary words are balanced if the sum of 1âĂŹs in each word has a difference of less than or equal to 1.

What is a balanced word list?

A balanced word list means that every binary word in the list is balanced with each other word. For example,

0 0
0 1
1 0

forms a balanced binary list, but

0 0
0 1
1 0
1 1

is not balanced since 0 0 and 1 1 have a difference of two.

How words are generated for our set?

We create the tree by looking at all possible balanced lists for any given n. Our tree begins with the simplest binary list

0
1

Note that this is a balanced list for the case n = 1. For the next level, n = 2, all possible words are

0 0
0 1
1 0
1 1

Since this is clearly an imbalanced list, a split occurs and there exists two possible balanced lists, namely

0 0
0 1
1 0

and

0 1
1 0
1 1

What is a Keyword?

A word w is a keyword of a word list A if w is a word in A and w followed by a 0 is balanced with all words in the word list A. Additionally, w followed by a 1 is balanced with all words in A. For example, the word list

0 0
0 1
1 0

the words 0 0 and 1 0 are keywords. This is because 0 0 0 , 0 0 1 , 1 0 0 , and 1 0 1 are all balanced with the word list above. However, the word 0 1 is not a keyword because the word when it is followed by a 1, giving 0 1 1 , is not balanced with the word 0 0 which is in the word list.

Fully Expanded Binary Word List Tree to Depth 12

Conclusions

At least one split will occur in every level n, $\forall n \in N$.

The occurrence of a split in every branch of the tree implies that our binary word tree will never stop growing in complexity. We were able to make the argument by noticing the recurring pattern in the leftmost branch. Through induction we were able to show that the leftmost branch will always split, and thus the tree will always split at least once in for every n.

Any two unique keywords w and x s.t. $w = w_1 w_2 \ldots w_n$, $x = x_1 x_2 \ldots x_n$ in a binary word list must be of the form $w_i = x_i;\ i \in \{2, 3, \ldots, n\}$. Subsequently, any word list has at most two keywords.

Using this definition of a keyword we were able to prove that all keywords are of the same form. Through contradiction, we were able to show that any word list will have at most two keywords. This is useful because we have identified the pattern that if a binary word list has two keywords, a split will occur on the next iteration, otherwise it will only have one child list.

If a word list has two keywords $0w$ and $1w$ then w is a palindrome.

This is useful because it tells us that every key word, with its last binary digit detached, is in the form of a palindrome. Subsequently, this implies that every exclusion is a palindrome. We were able to then create a new algorithm for generating the binary word tree by searching for all palindromes, which is equivalent to all possible exclusions, and iteratively building the tree using this list.

Future Endeavors

The totient function is also commonly known as Euler's phi. Euler's phi defines a function of a positive integer n, where Euler phi of n counts the number of positive integers up to n that are relatively prime to that n . This beautifully simple function has been found to be generated in several ways, as it is a powerful function whose pattern defines several things in mathematics, some that are not even expected. We have found that the number of splits in the tree converge to the values of Euler Phi of n where n is the nth level of the tree.

Oregon State University