Results on the K-Equitable Labeling of Complete Binary Trees

A Thesis Presented to

The Faculty of the Mathematics Program

California State University Channel Islands

In Partial Fulfillment

of the Requirements for the Degree

Masters of Science

by

Patrick Snyder

May 2012

Signature page for the Masters in Mathematics Thesis of Patrick Snyder

APPROVED FOR THE MATHEMATICS PROGRAM

Cynthia L. Wyels 5/18/12

Dr. Cynthia Wyels, Thesis Advisor — Date

5/18/12

Dr. Jorge Garcia, Thesis Committee — Date

APPROVED FOR THE UNIVERSITY

5-21-12

Dr. Gary A. Berg, AVP Extended University — Date

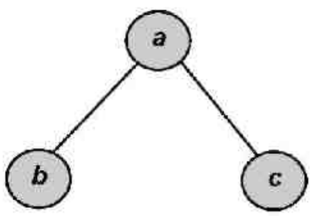
Table of Contents

Introduction

When it comes to graphs, there have always been questions about different ways to label the vertices and edges, and the properties that a given method will have. These questions have led to many different results regarding a myriad of different types of graphs. Some of the more popular derivations are in regards to radio labeling, network flows, minimum and maximum matching algorithms and other optimization problems that can have real-world applications to things like traffic control, infrastructure, logistics, radio frequency assignment, computer networks, micro-chip architecture and many, many more. As is the case with many areas of mathematics, many of these properties and methods were explored well before there were any known applications. Such is sometimes the nature of mathematics research. This thesis will cover some topics that have been developed around the graceful and k-equitable labeling of graphs and trees (focusing more on trees, specifically binary trees) and their relationship to other well-known problems including the n-queens problems. First, we must start with some definitions and examples. Please note that this thesis will only deal with standard graphs (graphs that do not contain multiple edges between any two given vertices).

Definitions and Terminology:

A **graph** is a 2-list (V, E) where V is a non-empty, finite set and E is a set of unordered pairs of V. The set V is the vertex set and the set E is the edge set. The elements of V are called vertices and the elements of E are called edges.



$V = \{a,b,c\}$

$E = \{(a,b),(a,c)\}$

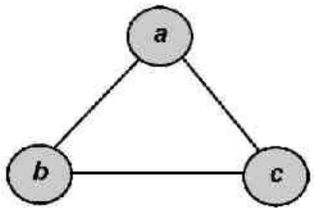
Figure 1

A vertex and an edge are **incident** provided that the vertex belongs to the edge (such as vertex a and edge (a,b) in Figure 1).

The **degree** of a vertex is the number of edges that are incident to that vertex. For example, in Figure 1 vertices b and c are of degree 1 while vertex a is of degree 2.

Two vertices are **adjacent** provided they are both contained in a single edge, such as vertices a and b in Figure 1 which are contained in edge (a,b).

An **acyclic** graph does not contain any sub-graph that is a cycle. The graph in Figure 1 is acyclic while the graph in Figure 2 is not.



$V = \{a, b, c\}$

$E = \{(a, b), (b, c), (a, c)\}$

Figure 2

A **tree** is a connected, acyclic graph. (See Figures [illegible] and 3.)

A **leaf** of a tree is a vertex of degree 1, such as vertices c, d and e in Figure 4.

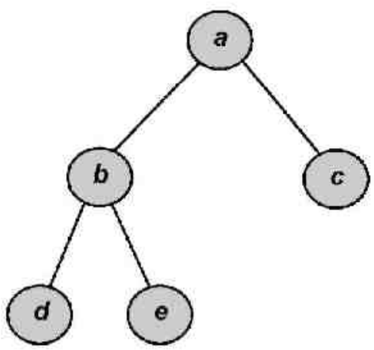


Figure 3

The **level** of a vertex v within a tree is the number of edges in the shortest path between v and the root vertex r of the tree. The level of r is by definition 0.

A **vertex labeling** of a graph G is a function $f: V(G) \rightarrow N$ that maps each vertex of G to an integer. Figure 4 shows a labeling f for the graph in Figure 3 for which $f(a) = 1$, $f(b) = 2$, $f(c) = 3$, $f(d) = 4$ and $f(e) = 5$.

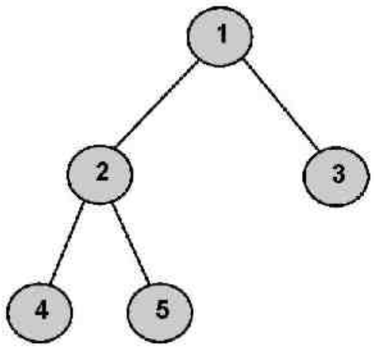


Figure 4

What is Graceful Labeling of a Graph?

For any undirected connected graph G with at least one vertex a graceful labeling is a labeling that satisfies the following conditions:

i) Each vertex of the graph is labeled by $f: V(G) \rightarrow \{0,\ldots,e\}$ where e is the number of edges in G.

ii) $f(v) = f(w)$ if and only if $v = w$.

iii) f induces a labeling of the edges of G such that edge $\{w,v\}$ receives the label $|f(w) - f(v)|$.

iv) Each label value from 1 to e is assigned to exactly one edge in G.

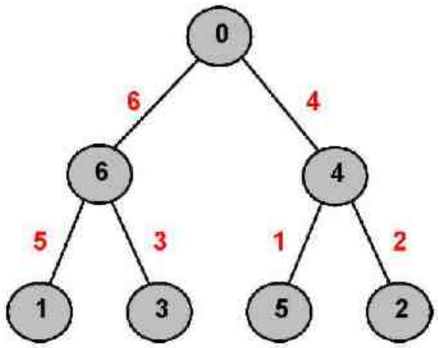


Figure 6: A gracefully labeled tree.

The Ringel-Kotzig conjecture states that all trees can be gracefully labeled. This is a result of the original conjecture by Ringel in 1963 that essentially claimed that every complete graph (K_n) can be decomposed into isomorphisms of an arbitrary tree. Many papers have been written that attempt to expand the knowledge surrounding the graceful labeling of trees and graphs with examples that show that many specific types, or classes of graphs or trees can be gracefully labeled. Although there has been much work on this topic, the Ringel-Kotzig conjecture has not yet been proven, and so remains a significant open problem in graph (and tree) labeling. The focus of this paper is a more general version of labeling called k-equitable labeling.

What is k-Equitable Labeling?

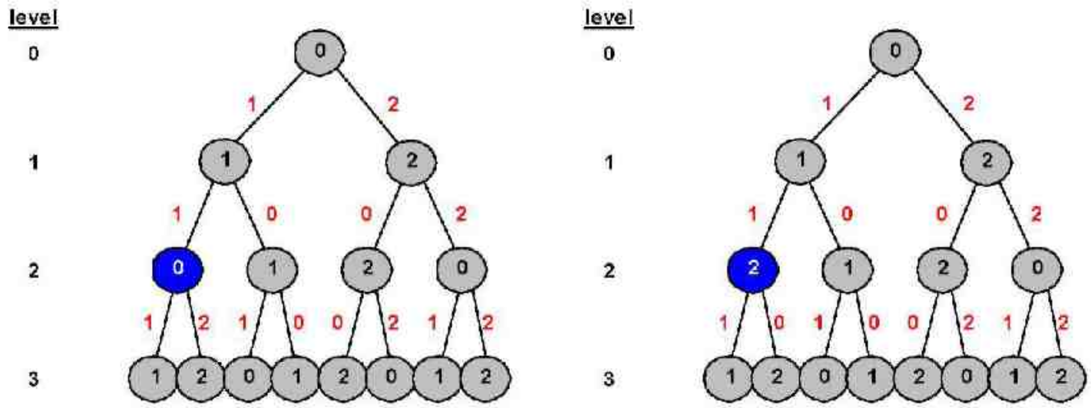
For any undirected connected graph G with at least one vertex, a k-equitable labeling is a labeling such that the following conditions are satisfied:

i) Each vertex in G is labeled with $f: v(G) \rightarrow \{0,\ldots,k\text{-}1\}$.

ii) f induces a labeling of the edges of G such that edge (w,v) receives the label $|f(w) - f(v)|$.

iii) The count of vertices and edges with values 0 to (k-1) are equitable. That is, the absolute value of the maximum difference between the counts of values of any edge or vertex label is 1 for all vertex and edge values 0 to (k-1).

Since a tree is a type of graph, a k-equitably labeled tree will satisfy these same conditions. Looking at Figure 7, we see that the trees are labeled with the values 0,1, and 2. The edges are labeled in accordance with condition *ii* above. Condition *iii* is satisfied in the tree on the left as we can see in the table beneath the tree shows that all values 0 to 2 are applied as a vertex label exactly five times and the resulting edge labels 1 and 2 are each applied five times and 0 is applied 4. This is in contrast to the tree on the right that shows the potential effect of changing a vertex label. The result is the vertex label 2 appearing six times while the label 0 appears only 4 times. Therefore the tree on the right is no longer 3-equitably labeled.



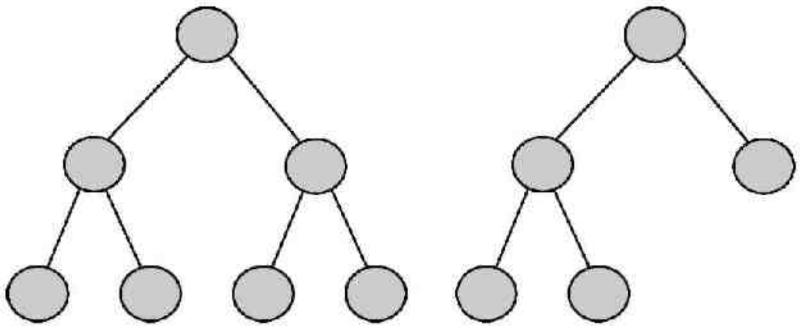
Value	0	1	2
Vertex Ct	5	5	5
Edge Ct	4	5	5

Value	0	1	2
Vertex Ct	4	5	6
Edge Ct	5	5	4

Figure 7: A 3-equitably labeled tree and a non-equitably labeled variation.

What is a Complete Binary Tree?

A binary tree is a connected tree in which each vertex has either two leaves or zero leaves. A complete binary tree is a binary tree with the added condition that only vertices in the final (let's call it n^{th}) level have no leaves and thus are of degree 1 while any vertices in levels 0 to n-1 have 2 leaves each and are of degree 3 except the root, which is of degree 2.



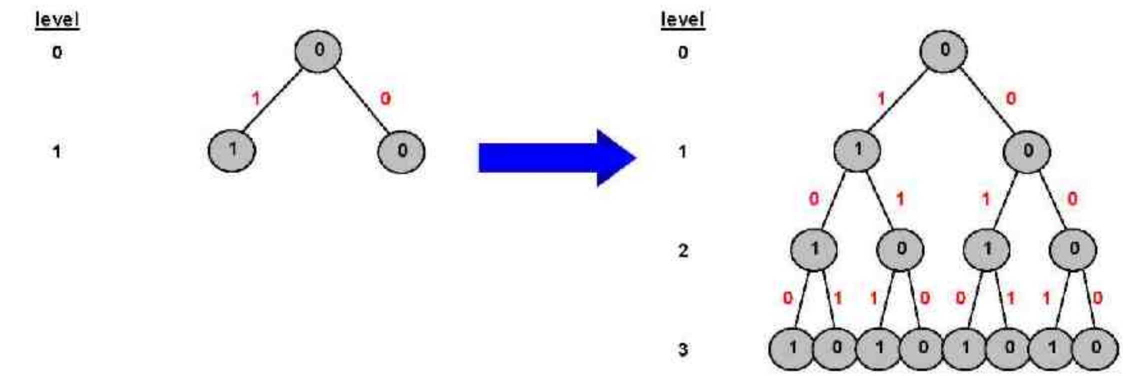
Complete Binary Tree **Incomplete Binary Tree**

Figure 8

Observations:

- A complete binary tree with n levels will have $2^{n+1} - 1$ vertices.
- A complete binary tree with n levels will have $2^{n+1} - 2$ edges.

All binary trees are 2-equitable, as may be demonstrated using the algorithm described next. First, the root vertex is labeled either 0 or 1. For $n = 0$, the case is trivial. Otherwise, the vertices adjacent to the root are then labeled 0 and 1 respectively.. For $n = 1$, the tree is 2-equitable because it either has two vertices labeled 0 (and one vertex labeled 1) or two vertices labeled 1 (and one vertex labeled 0). In either case, we end with one edge labeled 0 and one edge labeled 1. To label vertices in level n for $n > 1$, one vertex in level n adjacent to any given vertex in level n-1 is labeled 0 and the other is labeled 1. The binary property of the tree ensures that this labeling will always be equitable (See Figure 9).



Value	0	1
Vertex C	2	1
Edge Ct	1	1

Value	0	1
Vertex Ct	8	7
Edge Ct	7	7

Figure 9: 2-equitably labeled trees for $n = 1$ and $n = 3$.

Similarly, all complete binary trees are 3-equitable. A k-equitable labeling is produced by labeling the vertices 0,1,2,0,1,2,0,1,2,… as ordered from top to bottom and left to right (See Figure 7). The vertex labels are clearly equitably distributed by this labeling. The resulting algorithm is such that a vertex in level n labeled 0 be adjacent to vertices in level n+1 labeled 1 and 2. Similarly, n-level vertices labeled 1 will be adjacent to level n+1 vertices labeled 0 and 1 while n-level vertices labeled 2 will be adjacent to level n+1 vertices labeled 2 and 0. These vertex labels induces edge values of {1,2},{1,0}, and {0,2} respectively which by definition maintains the equitability of the edge values (See Table 10). In fact, David Speyer and Zsuzsanna Szanizlo proved that all trees are 3-equitable in 1999 [3].

Level n vertex label	Adjacent level $n+1$ vertex label		Induced edge labels	
	1st	2nd	1st	2nd
0	1	2	1	2
1	0	1	1	0
2	2	0	0	2

Table 10

Problem

While there is a substantial number of examples of many different types of k-equitable graphs in general, there is much that is as of yet unknown. There are few theorems that generalize the k-equitable labeling of anything but very specific types of trees or graphs. While complete binary trees certainly fit into the category of a very specialized type of tree, anything we can do to expand the knowledge in this area is meaningful. My interest in this topic came after discussions with my advisor Dr. Wyels, who had done some interesting work in this area using a modified n-queens type of solution. I decided to try to expand this base of solutions by developing algorithms that can take a k-equitable tree and modify it in such a way as to create a tree that is m-equitable for some value $m \neq k$. I found some interesting results that also tie into the work of another student quite well, all of which will be discussed in the following results. Given the importance and applications of binary trees in combinatorics and other branches of mathematics, we can only benefit by expanding what we know about them and their labeling.

Results

Theorem 1 (Doubling Algorithm):

Let T_1 be a complete binary tree with n levels such that

a) T_1 is k-equitably labeled by the labeling function $f_1(v)$,

b) the last level of T_1 is itself k-equitably labeled by $f_1(v)$, and

c) 2^{n+1} mod $2k$ is not greater than k.

Then a $2k$-equitably labeled, $(n+1)$-level complete binary tree (T_2) can be constructed from T_1.

Proof: We specify a labeling f_2 of the first n levels of an $(n+1)$-level complete binary tree and argue that f_2 may be extended to create a $2k$-equitable labeling of the entire tree.

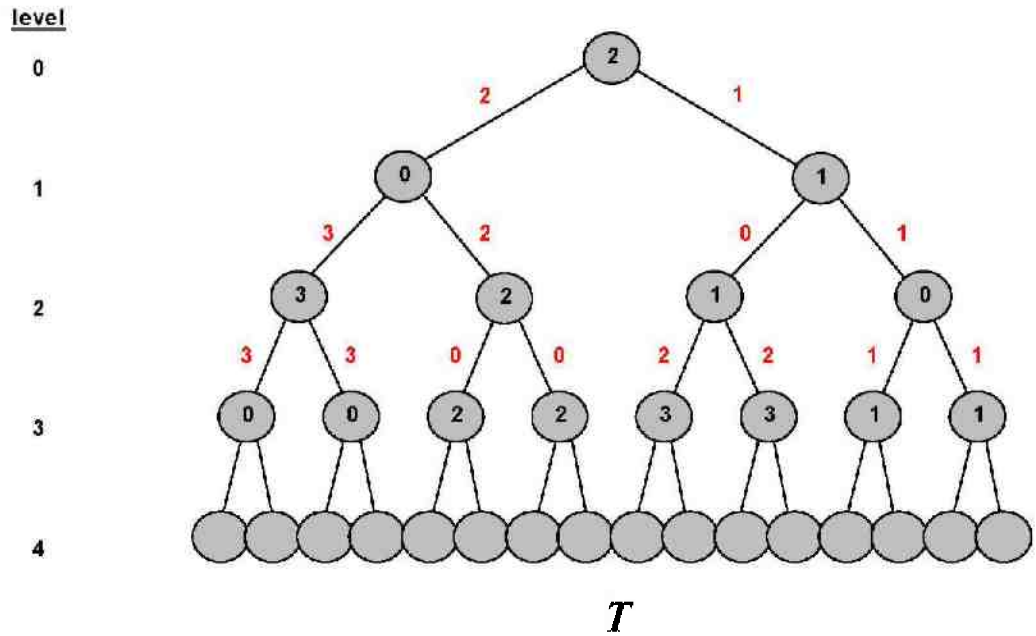
Identify the vertices of T_1 with the first n levels of an $(n+1)$-level tree T_2. Define f_2: $V(T_2) \rightarrow Z$ by $f_2(v) = 2f_1(v)$ for all vertices v in levels 0 to n. Let w be a vertex in level $n+1$. We need to define $f_2(w)$. Use v_w to denote the vertex in level

n that is adjacent to w. Specify $f_2(w) = (2k\text{-}1)\text{-}f_2(v_w)$ until each value of $f_2(w)$ has occurred $\left\lfloor \frac{2^{n+1}}{k} \right\rfloor$ times. The remaining (2^{n-1} mod $2k$) vertices are then labeled by identifying all possible labels that would induce an edge labeled with the value $2k$-1. Select one of these possibilities (if one exists) to be a label value, then repeat on the remaining unlabeled vertices for $2k$-3, $2k$-5, … or until no unlabeled vertices remain.

Since T_1 is k-equitably labeled, doubling the values of all the vertex labels in T_1 will also double all of the edge values in T_1. Thus T_2 is equitably labeled with all even values from 0 to $2k$-2. Since the vertices on level n are equitably labeled with these values, f_2 will result in an (n+1)st level that contains all of the odd values from 1 to $2k$-1 on the vertices. Edges between vertices in the nth and (n+1)st levels will be assigned values that take the same odd values from 1 to $2k$-1; these values will be equitably distributed as a result of the algorithm. Since there are 2^{n-1} vertices in the (n+1)st level of the tree, and there are 2^{n+1}-1 vertices in a complete binary tree with n levels, we know that creating an (n+1)st level with an equitable number of the odd values from 1 to $2k$-1 will result in a complete binary tree that has n+1 levels and is $2k$-equitable.

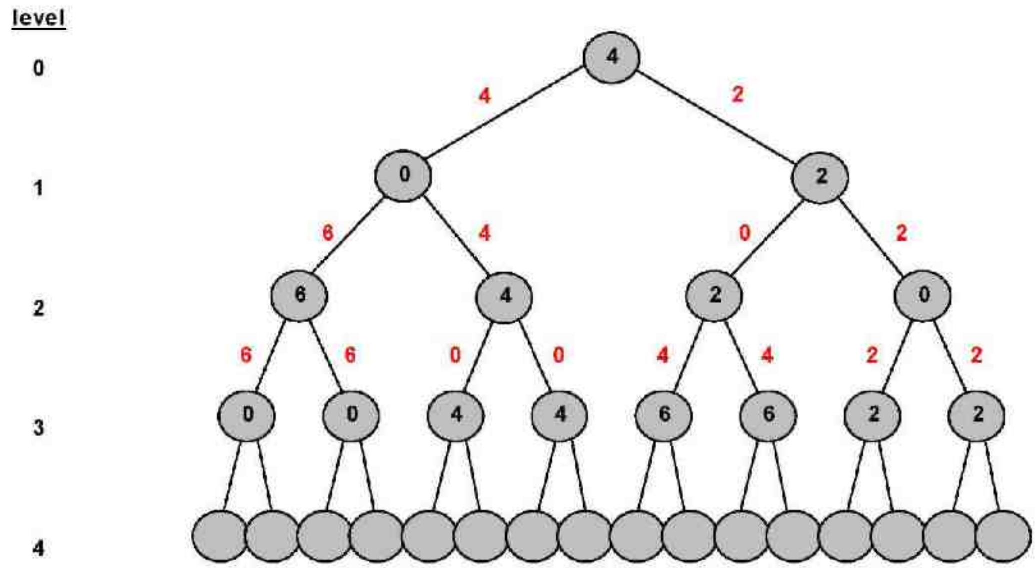
Example:

We start with T which is 4-equitable with n = 3 and has all the necessary conditions satisfied.



Value	0	1	2	3
Vertex Ct	4	4	4	3
Edge Ct	3	4	4	3
Level *n* Vertex Ct	2	2	2	2

Doubling all of the vertex labels of T results in the following partial T' which is equitably labeled (both in its entirety and in the 3rd level) with 0,2,4, and 6.



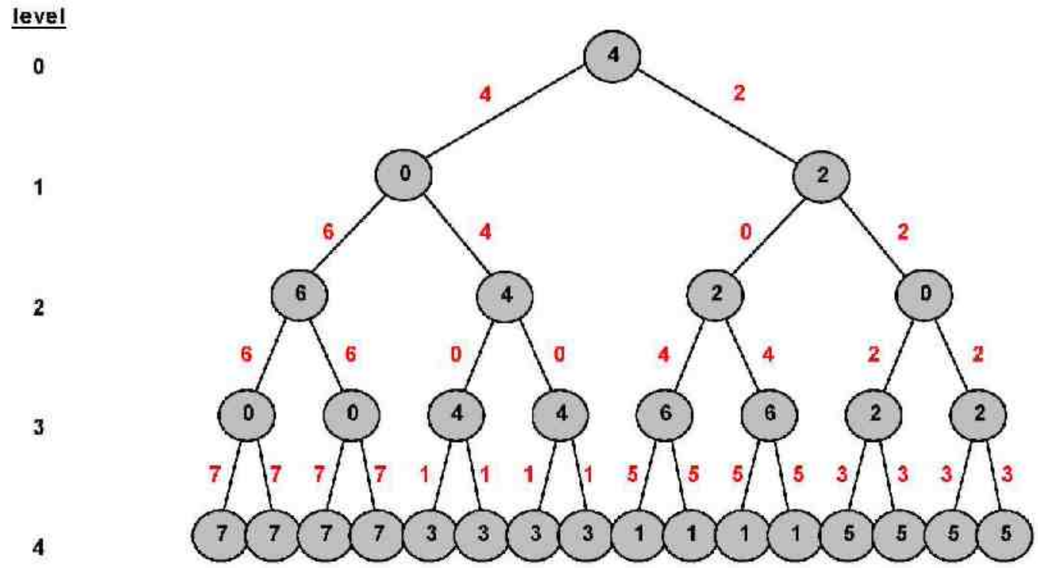
T' (partial)

Value	0	1	2	3	4	5	6	7
Vertex Ct	4	0	4	0	4	0	3	0
Edge Ct	3	0	4	0	4	0	3	0
Level n Vertex Ct	2	0	2	0	2	0	2	0

We then fill level n+1 according to our described algorithm, which in this case results in the adjacency label mapping:

Level n	is adjacent to	Level $n+1$
0	$\rightarrow$	7
2	$\rightarrow$	5
4	$\rightarrow$	3
6	$\rightarrow$	1

This results in T' as shown below which is shown by its table to be 8-equitable with $n+1 = 4$ levels.

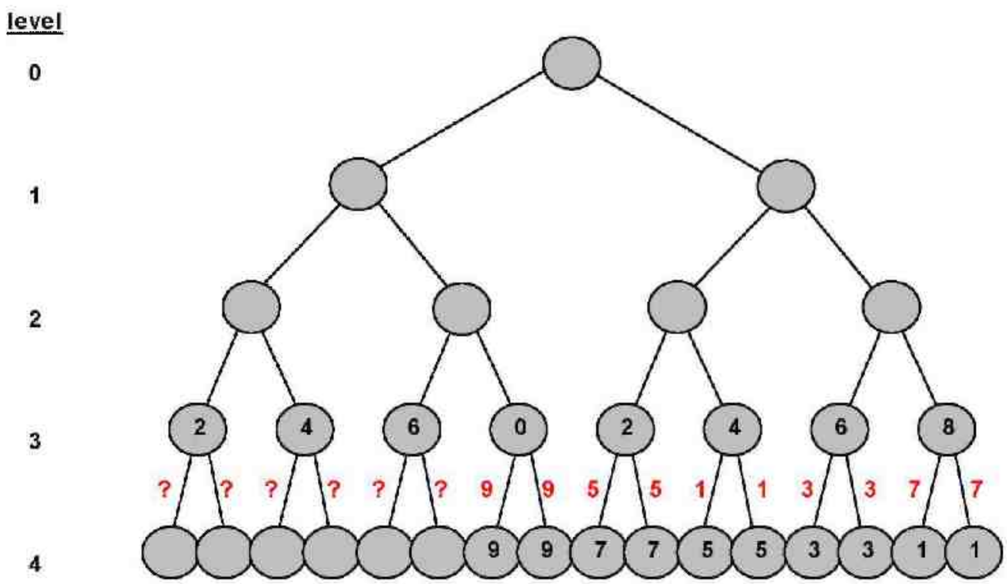


T'

Value	0	1	2	3	4	5	6	7
Vertex Ct	4	4	4	4	4	4	3	4
Edge Ct	3	4	4	4	4	4	3	4
Level n Vertex Ct	2	0	2	0	2	0	2	0

In this example, $n = 3$ and $k = 4$. Therefore, when we check condition (c), $2^{n+1} \bmod 2k = 2^4 \bmod 8 = 0$ is clearly not greater than 4. But what happens if it is? That is to say, why is the condition important? If $2^{n+1} \bmod 2k$ is equal to or greater than k, it would require that the second part of the algorithm create at least one edge labeled with each odd value from 1 to $2k$-1. Depending on the values of the nth level vertices adjacent to (n+1)st level vertices that remain to be labeled in this step, this may not be possible.

For example, if $n = 3$ and $k = 5$, then $2^k \bmod 10 = 6$. If 2, 4, and 6 are the values of the 3rd-level vertices adjacent to 4th level vertices that remain to be labeled, then there would be know way to label those vertices with the odd values from 1 to 9 in such a way as to induce an edge labeled 9. This means that we would not be able to equitably label that remaining portion of the 4th level since there are only 4 odd labels from which to choose and 6 total edges that need to be labeled equitably with all of those odd values as shown below.



Theorem 2 (Extended Doubling Algorithm):

Let T_2 be a complete binary tree with n+1 levels such that

a) T_2 is 2k-equitably labeled by the labeling function $f_2(v)$,

b) the last level of T_2 is itself equitably labeled with the odd values from 1 to 2k-1 by $f_2(v)$, and

c) 2^{n+2} mod 4k is not greater than 2k.

Then a 4k-equitably labeled, (n+2)-level complete binary tree (T_3) can be constructed from T_2.

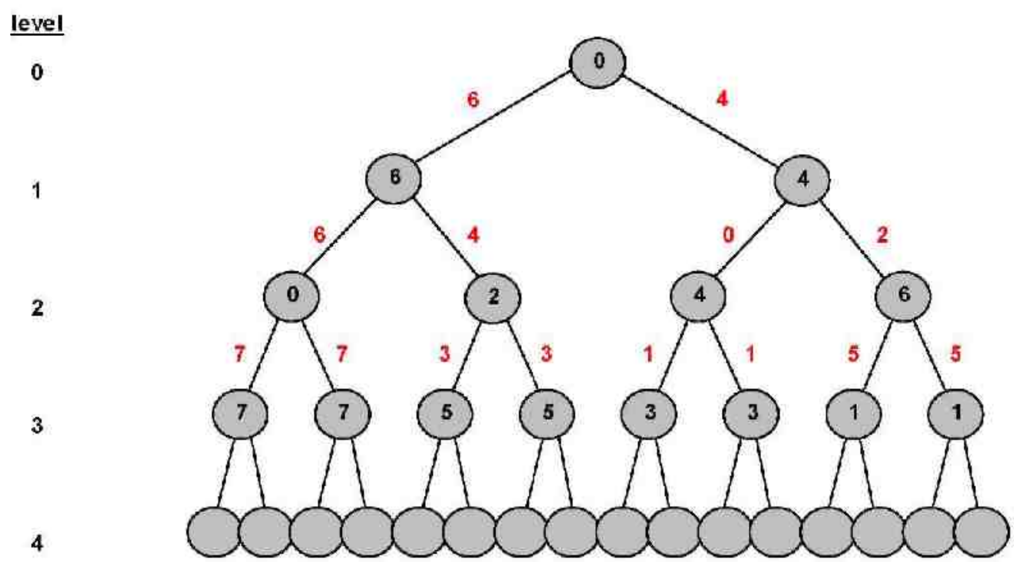
Proof: We specify a labeling f_3 of the first n+1 levels of an (n+2)-level complete binary tree and argue that f_3 may be extended to create a 4k-equitable labeling of the entire tree.

Identify the vertices of T_2 with the first n+1 levels of an (n+2)-level tree T_3. Define $f_3: V(T_3) \rightarrow Z$ by $f_3(v) = 2f_2(v)+1$ for all vertices v in levels 0 to n+1. Let u, w be vertices in level n+2 adjacent to the same vertex in level n+1. We need to define $f_3(u)$ and $f_3(w)$. Use $v_{u,w}$ to denote the vertex in level (n+1) that is adjacent to u and w. Specify $f_3(u) = (4k\text{-}1)\text{-}f_3(v_{u,w})$ while $f_3(w) = (4k\text{-}3)\text{-}f_3(v_{u,w})$ until each value of $f_3(w)$ has occurred $\left\lfloor \frac{2^{n-1}}{k} \right\rfloor$ times. The remaining (2^{n-2} mod 4k) vertices are then labeled by identifying all possible labels that would induce an edge labeled with the value 4k-1. Select one of these possibilities (if one exists) to be a label value, then repeat on the remaining unlabeled vertices for $4k$-3, $4k$-5, … or until no unlabeled vertices remain.

As defined, T_2 is a $2k$-equitably labeled tree with n+1 levels and is equitably labeled on the (n+1)st level with all of the odd values from 1 to $2k$-1. By doubling the vertex labels in T_2 and adding 1, we create a new tree whose vertices are equitably labeled with all odd values 1 to $4k$-1 and such that the (n+1)st level is equitably labeled with every other odd value from 3 to $4k$-1. All of the edge values within the tree were simply doubled by this step so the edges are now equitably labeled with all of the even values from 0 to $4k$-2. f_3 will result in an (n+2)nd level that contains all the even values between 0 and $4k$-2 on the vertices. Edges between vertices in the (n+1)st and (n+2)nd levels will be assigned values that take the odd values from 1 to $4k$-1; these values will be equitably distributed as a result of the algorithm. Therefore, the new tree (T_3) is a $4k$-equitable complete binary tree with n+2 total levels.

Example:

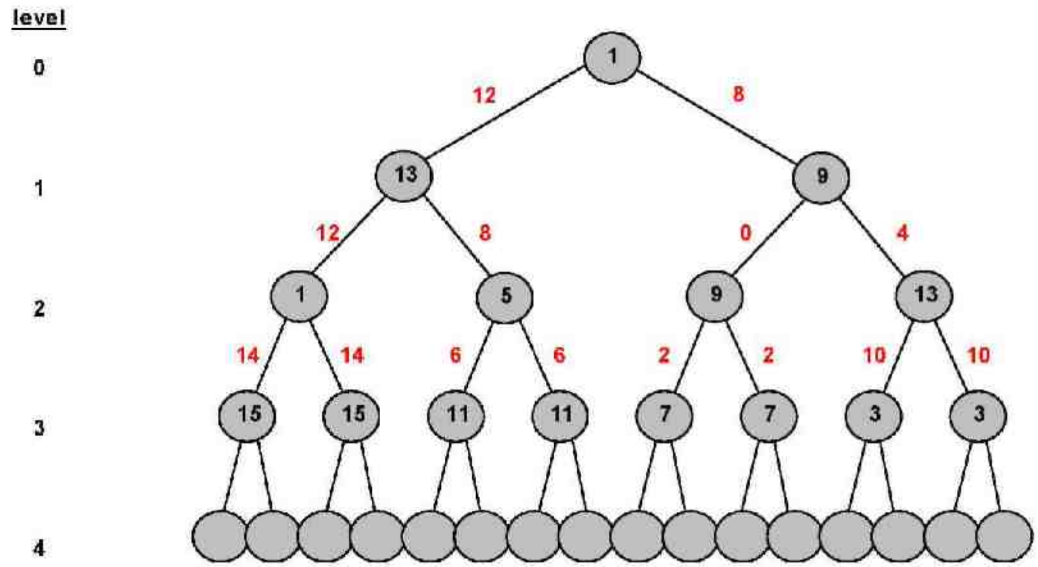
We start with T which is 8-equitable with $n = 3$ and has all the necessary conditions satisfied.



T

Value	0	1	2	3	4	5	6	7
Vertex Ct	2	2	1	2	2	2	2	2
Edge Ct	1	2	1	2	2	2	2	2
Level n Vertex Ct	0	2	0	2	0	2	0	2

Doubling all of the vertex labels of T and adding 1 results in the following partial T whose vertices are equitably labeled with the odd values from 1 to 15 while the vertices in level n are equitable labeled with every other odd value from 3 to 15 and the edges are equitably labeled with the even values from 0 to 14.



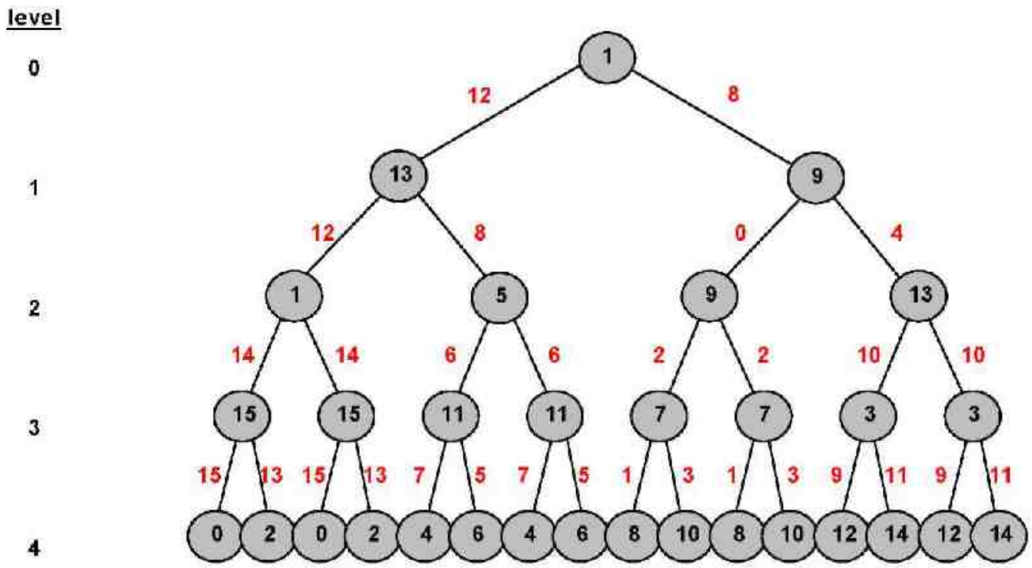
T^3 (partial)

Value	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Vertex Ct	0	2	0	2	0	1	0	2	0	2	0	2	0	2	0	2
Edge Ct	1	0	2	0	1	0	2	0	2	0	2	0	2	0	2	0
Level n Vertex Ct	0	0	0	2	0	0	0	2	0	0	0	2	0	0	0	2

We then fill level n+1 according to our described algorithm, which in this case results in the adjacency label mapping:

Level n	is adjacent to	Level $n+1$	Level $n+1$
3	$\rightarrow$	12	14
7	$\rightarrow$	8	10
11	$\rightarrow$	4	6
15	$\rightarrow$	0	2

This results in the tree T^4 as shown below, which is shown by its table to be 16-equitable with n+1 = 4 levels.



Value	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Vertex Ct	2	2	2	2	2	1	2	2	2	2	2	2	2	2	2	2
Edge Ct	1	2	2	2	1	2	2	2	2	2	2	2	2	2	2	2
Level *n* Vertex Ct	0	0	0	2	0	0	0	2	0	0	0	2	0	0	0	2

Other Results:

To lead into the next result, we must first describe what is known as an "n-queens" problem. There are several variations of this problem. The version of interest in this thesis asks whether it is possible to place n queens on an n x n chessboard in such a way that they are non-attacking. Relevance to my topic comes when we consider a specialized version of this problem.

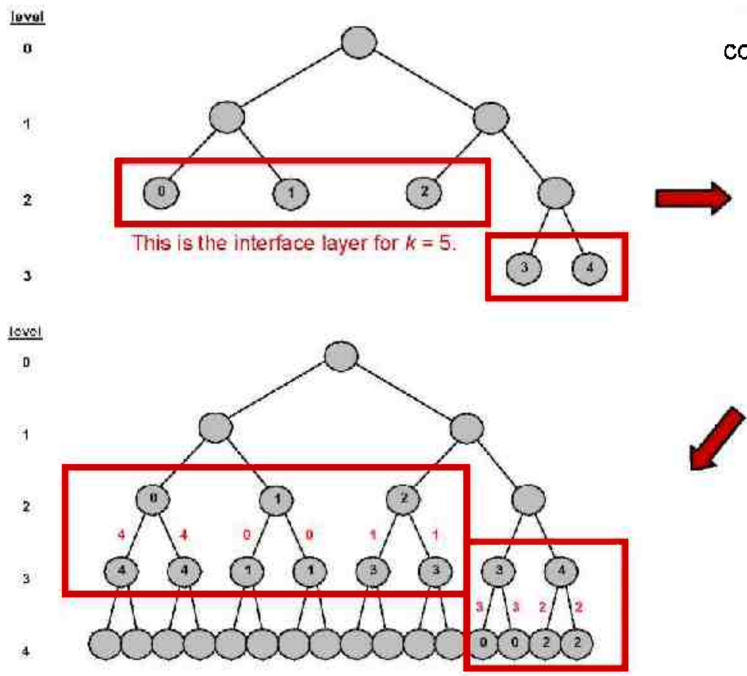
First we need to define a special type of matrix. A symmetric Toeplitz matrix is an n x n matrix that has constant negative diagonals with the further restriction that entry $(i_1, j_1) = (i_1, j_2)$ whenever $|i_1 - j_1| = |i_2 - j_2|$ (See Figure 11).

Let us then consider the n-queens problem on an n x n symmetric Toeplitz matrix instead of the standard chessboard. Let's also replace the usual diagonal restriction by specifying that queens may attack other queens if they occupy squares with the same number value in the matrix. Then it has been shown that n queens can be placed on this specialized chessboard for values of $n \equiv 0, 1 \bmod 4$, and that there is no solution when $n \equiv 2, 3 \bmod 4$ [4].

0	1	2	3	4
1	0	1	2	3
2	1	0	1	2
3	2	1	0	1
4	3	2	1	0

Figure 11: A 5×5 symmetric Toeplitz matrix with 5-queens solution highlighted and negative diagonal boxed.

Jennifer Russell is a Master's student at California State University Channel Islands who is currently working to apply solutions to the n-queens problem on a symmetric Toeplitz matrix to the equitable labeling of binary trees. The idea is that such a solution provides an algorithm for taking a tree that is equitably labeled above what she calls the interface layer, and using the solution to continue the labeling indefinitely for any number of levels of the tree. The interface layer is split between at most two levels of the tree and represents a partition of the tree into a "top" and "bottom." This layer also contains exactly one each of the label values 0 to k-1 on the vertices. Figure 12 shows the resulting labeling algorithm from the solution shown in Figure 11, the interface layer for $k = 5$, and the application of the algorithm to continue the labeling.



Permutation to continue equitable labeling	Induced edge values
$0 \rightarrow 4$	4
$1 \rightarrow 1$	0
$2 \rightarrow 3$	1
$3 \rightarrow 0$	3
$4 \rightarrow 2$	2

Figure 12

So far, Ms. Russell has been able to find tops (parts of the tree above the interface layer) that are k-equitable for many different values of k. The results of her process are equitably labeled binary trees that are not necessarily complete in and of themselves. It is conjectured that under most circumstances, these trees can be completed; an algorithm has not yet been developed. In either case, the tree that is created through this process has a familiar condition: the leaves of the tree are k-equitably labeled.

Ms. Russell has shown for specific values and conjectured in general that given an appropriate interface layer, there can be found a top to the tree that satisfies the requirements for a k-equitable labeling. From there the Toeplitz solution can be applied to continue the labeling for any number of levels. It is important to note that the results of her process are equitably labeled binary trees that are not necessarily complete. However, with the correct choice of interface layer, it is conjectured that these trees can be completed. Since the Toeplitz solution has been shown to exist if and only if the dimensions of the chessboard are congruent to 0, 1 mod 4, Ms. Russell's work as of the writing of this thesis has resulted in Conjecture 3 below:

Conjecture 3 (Toeplitz-based k-equitable labeling for $k \equiv 0$, 1 mod 4)

Let T_4 be a complete binary tree with n levels for n sufficiently large. Then there exists a Toeplitz solution-based k-equitable labeling of T_4 for all values of k such that $k \equiv 0,1 \bmod 4$.

The next theorem is inspired by this conjecture. This is due to the fact that a binary tree labeled in accordance with Conjecture 3 will have a condition somewhat analogous to those described in Theorem 1.

Theorem 4 (Doubling Toeplitz based solutions):

Let T_4 be an n-level binary tree such that

a) T_4 is k-equitably labeled using a Toeplitz-based solution as described in Conjecture 3 (prior to being completed by any non-Toeplitz algorithm), and

b) 2^{n+2} mod $2k$ is not greater than $2k$.

Then a $2k$-equitably labeled, $(n+1)$-level complete binary tree (T_5) can be constructed from T_4.

Proof: We first note that any k-equitable complete binary tree labeled in accordance with Conjecture 3 results in an interface-like layer (spanning levels n-1 and n) that is also k-equitably labeled.

Then by Theorem 1, and the same idea for completion as in Conjecture 3, we can create T_5 where T_5 is a $2k$-equitably labeled, $(n+1)$-level complete binary tree.

Conjecture 5 is a direct result of Conjecture 3 and Theorem 4. This is because any value congruent to 1 mod 4 can be doubled to a value congruent to 2 mod 4.

Conjecture 5 (Toeplitz-based k-equitable labeling for $k \equiv 0,1 \bmod 4$ and every other value of $k \equiv 2 \bmod 4$):

> Let T be a complete binary tree with n levels for n sufficiently large. Then there exists a Toeplitz solution-based k-equitable labeling of T for all values of k such that $k \equiv 0, 1 \bmod 4$ as well as every other value of k such that $k \equiv 2 \bmod 4$.

Proof: Conjecture 3 implies this is true for $k \equiv 0, 1 \bmod 4$. For $k \equiv 2 \bmod 4$, we simply apply Conjecture 3 for any $k \equiv 1 \bmod 4$ to the first n-1 levels of T and then apply Theorem 4 to label T such that T is k-equitable for $k \equiv 2 \bmod 4$. Since every other value of $k \equiv 2 \bmod 4$ can be reached multiplicatively by doubling some value congruent to 1 mod 4 (the other values of k congruent to 2 mod 4 are the double of a value congruent to 3 mod 4), we can do this for every value of $k \equiv 2 \bmod 4$ where $\frac{k}{2} \equiv 1 \bmod 4$.

Areas for Continued Research

The logical next step of this work would be to develop a proof that complete binary trees are k-equitable for all $k \equiv 3 \bmod 4$ using another operation-based algorithm and possible Toeplitz solutions. Depending on the characteristics of the resulting trees, this could imply that all complete binary trees are k-equitable for all k. Ideally, we would then be able to generalize the results to more types of trees, and eventually to all trees. If all trees are found to be k-equitable, then the Ringel-Kötzig Conjecture is directly implied. Therefore, it may be prudent to focus on the Ringel-Kötzig Conjecture first, as the special case should be less cumbersome to prove than the more general k-equitability problem, although even the special case has eluded proof for almost 40 years. In lieu of the most general results, one can always expand the base of information regarding tree labeling for other families of trees, or graphs.

References

[1]Mazur, David R. *Combinatorics: A Guided Tour.* U.S.A.: The Mathematical Association of America, 2010. Print.

[2] Bloom, G. S. (1979), *A Chronology of the Ringel-Kotzig Conjecture and the Continuing Quest to Call All Trees Graceful.* Annals of the New York Academy of Sciences, 328: 32–51. doi: 10.1111/j.1749-6632.1979.tb17766.x

[3] Speyer, David, and Zsuzsanna Szaniszló. "Every Tree is 3-Equitable." *Discrete Mathematics*, 220.1-3 (2000): 283.

[4] Szaniszló, Zsuzsanna, Maggy Tomova, and Cindy Wyels. "The *N*-Queens Problem on a Symmetric Toeplitz Matrix." *Discrete Mathematics*, 309.4 (2009): 969-974.

[5] Russell, Jennifer. "The *k*-Equitability of Complete Binary Trees." California State University Channel Islands, Camarillo.