Every Complete Binary Tree Is Prime

Joseph Chang
Palo Alto High School
50 Embarcadero Road
Palo Alto, CA 94301, USA

ABSTRACT

A graph with a vertex set V is said to have a prime labeling if its vertices can be labeled with distinct integers $1, 2, \cdots, |V|$ such that for every edge $\{x, y\}$, the labels assigned to x and y are relatively prime. A tree is prime if it has at least one prime labeling. Around 1980, Entringer conjectured that every tree is prime. After three decades, this conjecture remains open. Nevertheless, a few special classes of trees, specifically paths, stars, caterpillars, spiders, and small trees, have been shown to be prime. Among different types of trees, binary trees are probably the most frequently used in computer science. Fu and Huang showed that every perfect binary tree of order $2^d - 1$ is prime. Although Fu and Huang ambiguously called perfect binary trees as complete binary trees in their paper, it has been verified that they only proved that perfect binary trees are prime. In this paper, the author looked beyond perfect binary trees and devised a two-step method to prove that every complete binary tree is prime.

First, for the case of $2k - 1$ vertices, a near prime labeling was constructed such that the co-prime requirement was satisfied for every edge, except possibly for the edges between right leaves and their parents. In order to successfully construct a prime labeling without co-prime violations, the original prime labeling problem was transformed into a complete (co-prime) matching problem between the right leaves and their parents. By applying Hall's Theorem, we proved that a complete (co-prime) matching exists for the right leaves and their parents, thus proving that a prime labeling exists for every complete binary tree with $2k - 1$ vertices. Second, for the case of $2k$ vertices, we applied Bertrand-Chebyshev Theorem and proved that a three-way child swap could be performed to construct a prime labeling for $2k$ vertices based on the prime labeling for $2k - 1$ vertices, thus completing the proof that every complete binary tree of any number of vertices is prime.

Our proof that all complete binary trees are prime broadens the coverage of the tree classes that are known to be prime and propels the research one step closer to prove Entringer's conjecture.

Keywords:

prime labeling, Entringer's conjecture, complete binary tree

1. INTRODUCTION

Prime labeling was first introduced by Tout, Dabboucy, and Howalla [8]. Given a graph $G = (V, E)$ comprising a set V of vertices together with a set E of edges, a bijection $f : V \rightarrow \{1, 2, \cdots, |V|\}$ is called a prime labeling for G if for every edge $e = \{u, v\} \in E$, the greatest common factor of their labels $\gcd(f(u), f(v)) = 1$. A graph that admits a prime labeling is called a prime graph. Prime graphs have been studied by many researchers [3, 5, 9]. Around 1980, Entringer conjectured that every tree is prime. In 2007, Pikhurko proved that this conjecture is true for small trees with less than 51 vertices [6]. In addition, a few special classes of trees such as paths, stars, caterpillars, and spiders have been shown to be prime. Nevertheless, after 30 years, the conjecture remains open. Gallian's survey paper in 2013 [3] contains a summary on recent progress made on Entringer's conjecture.

Among different classes of trees, binary trees are probably the most frequently used in computer science. There are many types of binary trees. A *perfect* binary tree is a binary tree in which every parent has two children and all leaves are at the same depth [11]. As shown in Figure 1(a), a perfect binary tree of d levels has exactly $2^d - 1$ vertices, and all its internal vertices must have two children. As a comparison, a *complete* binary tree is a binary tree in which every level, except possibly the deepest level, is completely filled, and at depth d, the height of the tree, all vertices must be as far left as possible [1]. A complete binary tree of 13 vertices is shown in Figure 1(b) as an example. Because of the shape property, any complete binary tree can be conveniently stored in an array without storing any pointers. In addition, a binary heap, an important computer data structure, is stored in a complete binary tree. Note that a perfect binary tree is a special case of a complete binary tree.

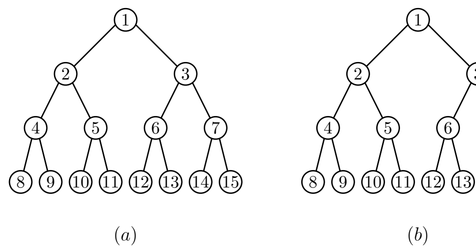

Fig. 1. (a) A perfect binary tree of four levels. (b) A complete binary tree of 13 vertices.

Fu and Huang [2] have proved that every perfect binary tree of order $2^d - 1$ ($d \geq 1$) is prime. Although Fu and Huang ambiguously called perfect binary trees as complete binary trees in their paper, we have confirmed that they only proved that perfect binary trees are prime. Their technique is to label all vertices from 2 to $|V|+1$ based on in-order traversal starting from the root of the tree. Then the label of the final vertex ($|V|+1$) is replaced by 1. It is easy to show that all leaf vertices are labeled with even numbers or 1, and all parent vertices are labeled with odd numbers. The difference between any (odd-labeled) parent and its even-labeled child is 1, and the difference between any (odd-labeled) parent and its odd-labeled child is a power of 2. With this labeling, every edge in a perfect binary tree connects a pair of coprime vertices, which proves that every perfect binary tree has a prime labeling. Figure 2(a) shows an example of Fu's labeling for a 15-vertex perfect binary tree. However, Fu's algorithm will not always produce a prime labeling for a complete binary tree. For example, as shown in Figure 2(b), Fu's algorithm labels the three vertices at the top of a 8-vertex complete binary tree with 4, 6, and 8, which are not coprime.

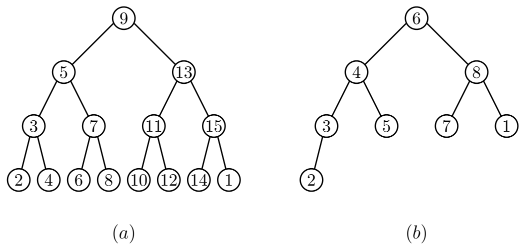

Fig. 2. (a) Fu's prime labeling for a perfect binary tree of four levels. (b) Fu's prime labeling algorithm not applicable to a 8-vertex complete binary tree.

In this paper, the author looks beyond perfect binary trees and shows that every complete binary tree is prime. Since a perfect binary tree is a special case of a complete binary tree, we move closer to prove Entringer's conjecture that every tree is prime.

2. PRIME LABELING OF COMPLETE BINARY TREES

2.1 Definitions

Let $v_{p,q}$ be the q^{th} vertex (from the left) of the p^{th} level (from the top) in a complete binary tree of n vertices. In this paper, we also say $v_{p,q}$ is at location (p,q). Note that for the vertex at location (p,q), its parent (if existent) is at location $(p-1, \lceil \frac{q}{2} \rceil)$, and its left and right children (if existent) are at locations $(p+1, 2q-1)$ and $(p+1, 2q)$, respectively.

Define the *index* of a vertex as a one-to-one and onto mapping x from the vertex set onto $\{1, \cdots, |V|\}$ as follows:

$$x(v_{p,q}) = 2^{p-1} + q - 1, p \in \{1, \cdots, d\}, q \in \{1, \cdots, 2^{p-1}\} \quad (1)$$

As illustrated in Figure 1(b), the index of a vertex is in fact its position in the level-order traversal sequence in which every vertex on a level is visited from left to right before descending to a lower level. It can be easily verified that, in a complete binary tree, the i^{th} vertex (except the root) has the parent with the index $\lfloor \frac{i}{2} \rfloor$. If the i^{th} vertex itself is also a parent, its left child has the index $2i$, and right child has the index $2i+1$. Thus, the index of a left child is always even, and the index of a right child is always odd. Conversely, every vertex with an even index is a left child.

Define a *leading* parent as a parent that is also a left child or the root. Since the index of a left child is always even, the index of a leading parent must be even or 1. For example, in the complete binary tree shown in Figure 1(b), vertices with indices 1, 2, 4, and 6 are leading parents.

In the next section, it will be shown that a prime labeling can be constructed in a two-step procedure for any complete binary tree with an odd number of vertices. Thereafter, it will be shown that a prime labeling for any complete binary tree with an even number of vertices can be constructed with one additional step.

2.2 The Prime Labeling Procedure for Complete Binary Trees with $2k-1$ Vertices

Starting from an indexed complete binary tree with $2k-1$ vertices, a prime labeling can be constructed in two steps. The first step is to bubble leading parents down their right branches. The goal of this step is to create a *near* prime labeling for a complete binary tree of $2k-1$ vertices. After the bubbling down step, it is guaranteed that the coprime requirement is satisfied for every edge, except possibly for the edges between right leaves and their parents.

The following describes the bubbling down step in details. In this bubbling down step, each and every leading parent displaces its right child recursively until the leading parent becomes a (right) leaf. During this process, all displaced right descendants are promoted one level up. For example, the leading parent with index 1 bubbles down the path $1 \rightarrow 3 \rightarrow 7$ in Figure 3(a) and becomes a right leaf in Figure 3(b), while 3 and 7 are promoted one level up. The leading parent with index 2 bubbles down the path $2 \rightarrow 5 \rightarrow 11$ in Figure 3(a) and becomes a right leaf in Figure 3(b), while 5 and 11 are promoted one level up.

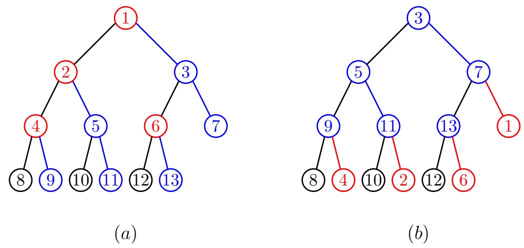

Fig. 3. A complete binary tree with $2k-1$ vertices (a) before the bubbling down step and (b) after the bubbling down step.

In a complete binary tree with $2k-1$ vertices, every parent has exactly one right child, and every right child has exactly one parent. In other words, the pairing between every parent and its right child is a one-to-one correspondence. Thus, every leading parent bubbles down to a unique right leaf, and every right leaf was a unique leading parent before bubbling down.

After bubbling down, a new parent whose location is (p,q) must have been promoted from location $(p+1, 2q)$, while every left leaf remains at its original location and every right leaf was a leading parent before bubbling down. Define the *label* of a location as its index after the bubbling down step. For the parent at location (p,q), its label is the index of the vertex that used to be at location $(p+1, 2q)$. That is, the label for a new parent at location (p,q) is given by

$$\ell(p,q) = x(v_{p+1,2q}) = 2^p + 2q - 1. \quad (2)$$

From (2), it is clear that the labels for all the new parents after the bubbling down step are odd numbers.

Next, it will be proved that, after bubbling down, any parent and its left child are coprime. Let the parent's location be (p, q). Then its left child is at location $(p+1, 2q-1)$. A left child is either a parent itself or a left leaf. First, consider the case where the left child is also a parent. Substituting location $(p+1, 2q-1)$ into (2), we have

$$\ell(p+1, 2q-1) = 2^{p+1} + 2(2q-1) - 1 = 2\ell(p, q) - 1. \quad (3)$$

Applying Euclidean algorithm, we have $\gcd(\ell(p, q), 2\ell(p, q) - 1) = 1$. Therefore, a parent and its left child who is also a parent are coprime. Next, consider the case where the left child is a left leaf. Since every left leaf remains at its original location, substituting location $(p+1, 2q-1)$ into (1), we have

$$\ell(p+1, 2q-1) = x(v_{p+1,2q-1}) = \ell(p, q) - 1. \quad (4)$$

In other words, a left leaf's label is always one less than its parent's. Applying Euclidean algorithm, we have $\gcd(\ell(p, q), \ell(p, q)-1) = 1$. Hence, a left leaf and its parent are coprime. Combining both cases, we assert that any parent and its left child are always coprime after bubbling down.

If every parent and its right child were also coprime, then a prime labeling would exists for the complete binary tree with $2k - 1$ vertices. Let the parent's location be (p, q). Then its right child is at location $(p+1, 2q)$. A right child is either a parent itself or a right leaf. Consider the case where the right child is a parent. Substituting location $(p+1, 2q)$ into (2), we have

$$\ell(p+1, 2q) = 2^{p+1} + 2(2q) - 1 = 2\ell(p, q) + 1. \quad (5)$$

Applying Euclidean algorithm, we have $\gcd(\ell(p, q), 2\ell(p, q) + 1) = 1$. Therefore, a parent and its right child that is also a parent are coprime.

In the case where the right child is a right leaf, the right leaf and its parent are not guaranteed to be coprime after bubbling down, as shown in Figure 4. Hence, this case is more complicated and requires the second step of the procedure to be described in the following. Consider a complete binary tree with $2k - 1$ vertices. Let L_k be the set of the labels of all the right leaves, and P_k be the set of the labels of the parents of all the right leaves. For $k = 1$, the complete binary tree degenerates into a single vertex with zero edge, which is prime trivially. For $k = 2$, the complete binary tree consists of two edges and three nodes labeled 1, 2, and 3, which is prime easily. When $k \geq 3$ and k is odd, $P_k = \{k, k+2, \cdots, 2k-1\}$ and $L_k = \{1, 2, 4, \cdots, k-1\}$. When $k \geq 3$ and k is even, $P_k = \{k+1, k+3, \cdots, 2k-1\}$ and $L_k = \{1, 2, 4, \cdots, k-2\}$. In both cases, $|P_k| = |L_k| = \lceil \frac{k}{2} \rceil$. The goal is to match each right leaf $r \in L_k$ with a unique parent $p \in P_k$ such that $\gcd(p, c) = 1$. In other words, the original prime labeling problem has been converted into a matching problem.

We need to introduce the Hall's (marriage) theorem. In 1935, Hall [4] answered the following question, also known as the marriage problem: for a finite set of girls, each of whom knows several boys, under what conditions can all the girls marry the boys in such a way that each girl marries a boy she knows? This problem can be represented graphically by taking H to be the bipartite graph [10] in which the vertex set is divided into two disjoint sets X and Y which correspond to the girls and boys, respectively, and each edge joins a girl to a boy she knows. A **matching** M in H is a subset of edges such that no two edges in M are incident to the same vertex. A complete matching of X into Y is a matching such that every vertex in X has an edge incident from it. The marriage problem can then be restated as follows: given a bipartite graph $H = H(X, Y)$, under what conditions does a complete matching from X into Y exist? For a set $A \subseteq X$, define the set $R(A)$ to be the vertices in Y that are adjacent to at least one vertex in A. Hall proved the sufficient and necessary conditions for the existence of a complete matching.

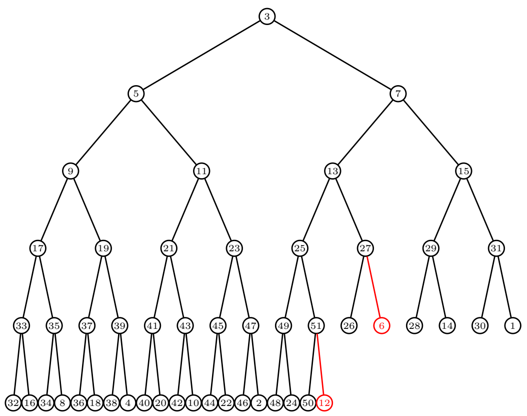

Fig. 4. A complete binary tree with 51 vertices after bubbling down. There are two coprime violations highlighted in red. The first violation is between labels 27 and 6. The second violation is between labels 51 and 12.

THEOREM 1. *Let H be a bipartite graph with parts X and Y. There is a complete matching of X into Y if and only if $|A| \leq |R(A)|$ for every $A \subseteq X$.*

For our specific matching problem, $X = L_k$ and $Y = P_k$. An element $x \in X$ and an element $y \in Y$ are adjacent to each other if $\gcd(x, y) = 1$. Consider a subset $A \subseteq L_k$. If A contains any pure power of 2, then $R(A) = P_k$ since every member of P_k is an odd number and coprime to any pure power of 2. Hence, the sufficient condition $|R(A)| = |P_k| = |L_k| \geq |A|$ is true, and the equality holds only when $A = L_k$.

Now consider the case where A does not contain any pure power of 2. For any element $a_i \in A$, the prime factorization of $a_i = 2^{y_i} \times \prod_j q_{ij}^{s_{ij}}$ where q_{ij} are odd prime factors of a_i, $3 \leq q_{ij} \leq \frac{1}{2} \max L_k$. For any element $b_i \in L_k$ which has the same prime factors as a_i, let $b_i = 2^{y'_i} \prod_j q_{ij}^{s'_{ij}}$ and let $B_i = \{b_i\}$ be the set of all such b_i. It is trivial that $a_i \in B_i$ and $A \subseteq \bigcup_i B_i$. By definition, $R(\{a_i\}) = \{p \mid p \in P_k, q_{ij} \nmid p, \forall j\}$. Note that $R(B_i) = \{p \mid p \in P_k, q_{ij} \nmid p, \forall j\} = R(\{a_i\})$. Consequently, $R(\bigcup_i B_i) = R(A)$.

Let $\ell = |L_k| = |P_k| = \lceil \frac{k}{2} \rceil$. Note that the elements of L_k are $(\ell - 1)$ consecutive even numbers and 1. For these $(\ell - 1)$ consecutive even numbers, there are at most $\lfloor (\ell - 1) \prod_j \frac{1}{q_{ij}} \rfloor$ elements in L_k that have the same prime factors as a_i. That is, $|B_i| \leq \lfloor (\ell - 1) \prod_j \frac{1}{q_{ij}} \rfloor$. Consider the union of all such sets B_i and apply the De Morgan's Law, and we have

$$|A| \leq |\cup_i B_i| = \left|\overline{\cap_i \overline{B_i}}\right| \leq \left\lfloor (\ell - 1)(1 - \prod_i (1 - \prod_j \frac{1}{q_{ij}})) \right\rfloor. \quad (6)$$

Recall that $R(B_i) = R(\{a_i\}) = \{p \mid p \in P_k, q_{ij} \nmid p, \forall j\}$. Similar to the derivation of the Euler's totient function, there are at least $\lceil (\ell - 1) \prod_j (1 - \frac{1}{q_{ij}}) \rceil$ elements in P_k that are coprime to a_i. Consider the union of all such elements and apply the De

Morgan's Law, and we have

$$\begin{aligned} |R(A)| &= |\cup_i R(B_i)| = \left|\overline{\cap_i \overline{R(B_i)}}\right| \\ &\geq \left\lceil (\ell-1)(1-\prod_i (1-\prod_j \tfrac{g_{ij}-1}{g_{ij}})) \right\rceil. \end{aligned} \quad (7)$$

Since $g_{ij} \geq 3, \forall i, j$, $\frac{g_{ij}-1}{g_{ij}} > \frac{1}{g_{ij}}$. Hence, the lower bound of $|R(A)|$ is always greater than the upper bound of $|A|$. In other words, $|R(A)| > |A|$ if A does not contain any pure power of 2. Combined the results from both cases, $|R(A)| \geq |A|$ for any subset $A \subseteq L_k$, and $|R(A)| > |A|$ for any non-empty proper subset $A \subset L_k$. By Hall's Theorem, there is a complete matching of L_k into P_k. That is, a complete coprime matching can always be found for the set of right leaves and the set of their parents. Therefore, a prime labeling exists for a complete binary tree with $2k-1$ vertices. As shown in Figure 5, after completely matching the right leaves and their parents, a prime labeling is constructed for the complete binary tree with 51 vertices.

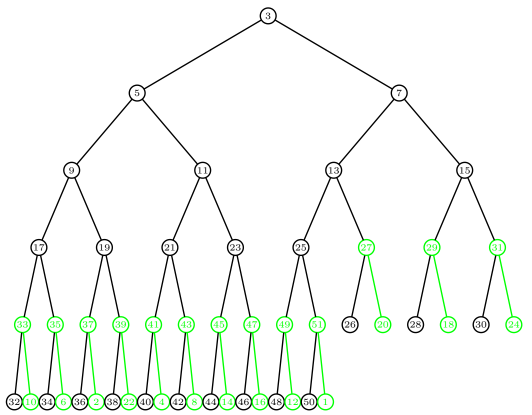

Fig. 5. A complete binary tree with 51 vertices after completely matching the right leaves and their parents. A prime labeling exists for any complete binary tree with $2k-1$ vertices. Note that, this prime labeling contains a special pair $(47, 16)$.

2.3 The Prime Labeling Procedure for Complete Binary Trees with $2k$ Vertices

Recall that $|R(A)| > |A|$ for any non-empty proper subset $A \subset L_k$. Let $r^* \in L_k$ and $p^* \in R(\{r^*\})$, so $\gcd(p^*, r^*) = 1$. Let $L = L_k \backslash \{r^*\}$ and $P = P_k \backslash \{p^*\}$. Then for any subset $A \subseteq L$, A is a proper subset of L_k. It follows that $|R(A)| > |A|$ or equivalently $|R(A)| \geq |A| + 1$. Since p^* may or may not be in $R(A)$, it follows that $|R(A) \backslash \{p^*\}| \geq |R(A)| - 1 \geq |A|$, for any $A \subset L_k$. By Hall's Theorem, there is a complete matching of $L = L_k \setminus \{r^*\}$ into $P = P_k \setminus \{p^*\}$. Since $\gcd(p^*, r^*) = 1$, a complete matching of L_k into P_k with (p^*, r^*) paired together is constructed. The significance of this fact is that there is great freedom when choosing the preferred coprime pair (p^*, r^*).
By Bertrand-Chebyshev Theorem [7], for any integer $t > 3$, there is always at least one prime p such that $t < p < 2t - 2$. Let $t = k + 1$. Then there is always at least one prime p such that $k + 1 < p < 2k, \forall k \geq 3$. In other words, there is always at least one prime number in P_k. Let p^* be the largest prime in P_k. Let $r^* = 2^z$ be the largest pure power of 2 in L_k. Clearly, $\gcd(p^*, 2^z) = 1$. Therefore, when a prime labeling is constructed for a complete binary tree with $2k-1$ vertices, the special pairing $(p^*, 2^z)$ can always be made mandatory.

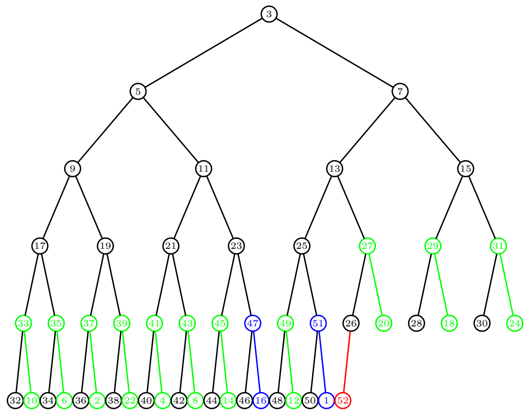

Fig. 6. The only difference between complete binary trees with 51 and 52 vertices is the extra vertex 52 and the edge connected to its parent 26.

Compare the complete binary trees of $2k-1$ versus $2k$ vertices. Their only difference is that the complete binary tree with $2k$ vertices has an extra vertex $2k$ as the left leaf child of the vertex k, as shown in Figure 6. Let us temporarily remove the vertex $2k$ and construct a prime labeling for the resulting $2k-1$ tree with the special pairing $(p^*, 2^z)$. Let p_1 be the parent of 1 in our prime labeling for the $2k-1$ tree. Now re-attach the left leaf $2k$ back to the vertex k. The only coprime violation is the pair $(k, 2k)$, as shown in red in Figure 6. Consider the following three parent-child pairs: $(k, 2k)$, $(p^*, 2^z)$, and $(p_1, 1)$. If a three-way child swap is performed to form the following three pairs: $(k, 1)$, $(p^*, 2k)$, and $(p_1, 2^z)$, we will establish three coprime parent-child pairs while maintaining a valid prime labeling for the rest of the tree. In other words, a prime labeling for a complete binary tree with $2k$ vertices can be constructed from the prime labeling for a complete binary tree with $2k-1$ vertices with a three-way child swap, which is demonstrated in Figure 7.
In summary, a complete binary tree with either $2k-1$ or $2k$ vertices has a prime labeling. Thus, we assert that every complete binary tree is prime.

3. CONCLUSION

Entringer's conjecture that every tree has a prime labeling remains unsolved after three decades. Mathematicians were able to prove that several classes of trees such as paths, stars, caterpillars, spiders, small trees, and perfect binary trees are prime. To contribute to the effort of proving Entringer's conjecture, we looked beyond those solved trees and studied the class of complete binary trees.
Using a novel method, we showed that every complete binary tree is prime. Our proof of the prime labeling for complete binary trees shed new light on Entringer's conjecture by expanding the coverage of the types of trees and paved the way for further prime labeling researches.

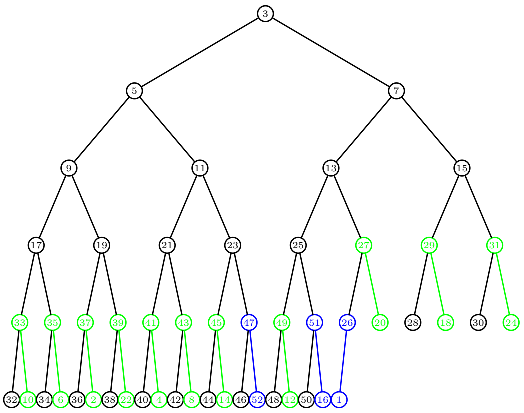

Fig. 7. After performing three-way child swap, the three new parent-child pairs $(26, 1), (47, 52)$, and $(51, 16)$ are all coprime pairs. The coprime violation $(26, 52)$ in Figure 6 is removed, and a prime labeling for the complete binary tree with 52 vertices is constructed.

4. REFERENCES

[1] P. E. Black. *Complete binary tree*. U.S. National Institute of Standards and Technology, 2011. http://xlinux.nist.gov/dads.

[2] H. L. Fu and K. C. Huang. On prime labelling. *Discrete Math.*, 127:181–186, 1994.

[3] J. A. Gallian. A dynamic survey of graph labeling. *The Electronic Journal of Combinatorics*, 16, 2013.

[4] P. Hall. On representatives of subsets. *J. London Math. Soc.*, 10:26–30, 1935.

[5] P. Haxell, O. Pikhurko, and A. Taraz. Primality of trees. *J. Combinatorics*, 2:481–500, 2011.

[6] O. Pikhurko. Trees are almost prime. *Discrete Math.*, 307:1455–1462, 2007.

[7] Sondow and Weisstein. Bertrand's postulate. From MathWorld – A Wolfram Web Resource. http://mathworld.wolfram.com/BertrandsPostulate.html.

[8] A. Tout, A.N. Dabboucy, and K. Howalla. Prime labeling of graphs. *Nat. Acad. Sci. Letters*, 11:365–368, 1982.

[9] S. K. Vaidya and U. M. Prajapati. Some new results on prime graphs. *J. Discrete Math.*, 2:99–104, 2012.

[10] Weisstein. Bipartite graph. From MathWorld – A Wolfram Web Resource. http://mathworld.wolfram.com/BipartiteGraph.html.

[11] Y. Zou and P. E. Black. *Perfect binary tree*. U.S. National Institute of Standards and Technology, 2008. http://xlinux.nist.gov/dads.