Finding Antimagic Labelings of Trees by Evolutionary Search

Luke Branson
Algorithmic Evolution Lab
Department of Computer Science
University of Minnesota Duluth
Duluth, MN, USA

Andrew M. Sutton
Algorithmic Evolution Lab
Department of Computer Science
University of Minnesota Duluth
Duluth, MN, USA

Xiankun Yan
Optimisation and Logistics Group
School of Computer Science
University of Adelaide
Adelaide, SA, Australia

ABSTRACT

Randomized search heuristics can sometimes be effective verifiers for combinatorial conjectures. In this paper, we demonstrate how a simple evolutionary algorithm can be used to confirm the *antimagic tree conjecture* for all trees up to order 25. This conjecture, which has been open for over thirty years, is that every tree except K_2 has an *antimagic labeling*: a bijective edge labeling such that the sum of labels assigned to edges incident to a vertex v is unique for all vertices $v \in V$. Moreover, we formally prove that that simple evolutionary algorithms are guaranteed to find antimagic labelings in expected polynomial time on trees of any order for certain restricted classes (paths, combs, uniform caterpillars, uniform spiders and perfect binary trees).

CCS CONCEPTS

• **Theory of computation → Theory of randomized search heuristics**.

KEYWORDS

Combinatorial optimization, graph labeling, runtime analysis

ACM Reference Format:
Luke Branson, Andrew M. Sutton, and Xiankun Yan. 2023. Finding Antimagic Labelings of Trees by Evolutionary Search. In *Proceedings of the 17th ACM/SIGEVO Conference on Foundations of Genetic Algorithms (FOGA '23), August 30-September 1, 2023, Potsdam, Germany.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3594805.3607133

1 INTRODUCTION

Computational verification of conjectures is an important application in combinatorial optimization [7]. In some cases, this work can be done effectively by randomized search heuristics [2, 5, 11, 15, 21], especially when the underlying search space is relatively smooth.

Let $G = (V, E)$ be an undirected graph with $|V| = n$ vertices and $|E| = m$ edges. An *edge labeling* of G is a bijection $\ell\colon E \to \{1, 2, \ldots, m\}$. For $v \in V$, denote as $E(v)$ the set of edges incident to v. An edge labeling ℓ induces the following weight sequence on the vertices of G.

$$\phi_\ell(v) = \sum_{e \in E(v)} \ell(e).$$

An *antimagic labeling* of G is an edge labeling ℓ in which $\phi_\ell(u) \neq \phi_\ell(v)$ for any two distinct $u, v \in V$. See Figure 1 for an example.

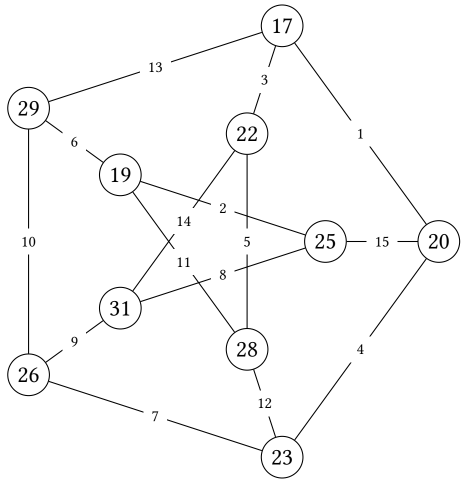


Figure 1: Antimagic labeling of the Petersen graph.

Many classes of graphs are known to have antimagic labelings, for example, all paths and complete graphs with $n > 2$, all cycles and wheels [13], caterpillars [19], regular graphs [8], graphs with maximum degree at least $n - 2$ [3], all complete partite graphs (except K_2) [3], and many others [4, 12].

Despite this, the following conjecture has remained open for over thirty years.

Conjecture 1.1 (Hartsfield and Ringel [13]). *Every tree other than K_2 is antimagic.*

The goal of this paper is to take a computational perspective for investigating this conjecture using relatively simple evolutionary algorithms. We first formally prove that for certain restricted classes of trees: paths, combs, uniform caterpillars and spiders, and perfect binary trees, a simple evolutionary algorithm augmented with a "guide" function always finds antimagic labelings in expected polynomial time. This guide function serves the purpose of imposing extra structure on the intricate labeling search landscape in order to support our proofs. However, we conjecture that the guide function is not required for polynomial runtime on these instances.

We then experimentally demonstrate that simple evolutionary search performs remarkably well for finding antimagic labelings of general trees up to a finite order. We employ this to confirm Conjecture 1.1 for every tree up to order $n = 25$. We empirically investigate a number of different mutation operations and observe efficient scaling behavior on all trees.

1.1 Background

Verification of the related *graceful tree conjecture* on trees up to order $n = 27$ was carried out by Aldred and McKay [2] who used an algorithm of Wright et al. [25] to systematically generate every tree of a particular order, and then applied local search on label permutations of each tree using a swap operator and tabu list to optimize an objective based on the number of conflicts. Later, Horton [15] and Fang [11] extended this verification to trees of up to 29 and 35 vertices, respectively, using combinations of greedy backtracking, parallelization, and stochastic local search with a tabu list and Metropolis-like acceptance condition.

Other randomized search heuristics such as ant colony optimization [20] and genetic algorithms [10] have also been investigated in the context of graceful labelings. Recently, Branson and Sutton [6] proved polynomial runtime guarantees for the (1+1) EA finding graceful labelings for all paths, stars and complete bipartite graphs $K_{c,n}$ where $c = O(1)$. The problem of determining whether a graph is antimagic is in some sense much less constrained than finding a graceful labeling, and therefore we would expect it to be more amenable to randomized search heuristic methods.

Antimagic labelings of graphs are important in graph theory, but also have real-world applications in networks and telecommunication [1], scheduling, and unbalanced loading problems [9].

2 EVOLVING ANTIMAGIC LABELINGS

Let $T = (V, E)$ be a tree on $|V| = n$ vertices. We impose an arbitrary ordering on the edges $\{e_1, e_2, \ldots, e_{n-1}\}$ and represent an edge labeling of T as a permutation x of $(1, 2, \ldots, n-1)$, where $x[i]$ corresponds to the label assigned to edge e_i. Given a labeling x, let

$$d(x) := |\{w \mid \exists v \in V \text{ s.t. } \phi_x(v) = w\}| \tag{1}$$

be the size of the partition of vertex weights induced by x and define the fitness function to be

$$f(x) := n - d(x). \tag{2}$$

A labeling x is an antimagic labeling of T if and only if $f(x) = 0$. Therefore, we reduce the problem of finding an antimagic labeling of T to the problem of minimizing f over permutations of length $n - 1$.

To minimize f, we consider the sequence-based (1+1) EA introduced by Scharnow, Tinnefeld and Wegener [22]. Given a tree T on n vertices, the (1+1) EA, shown in Algorithm 1, is initialized with a random permutation of $(1, 2, \ldots, n-1)$ and in each iteration produces an offspring by a permutation mutation operation. If the fitness value of the offspring is no worse than the current point, it replaces the current point.

A given sequence is mutated by performing a number of local permutation operations. In this paper, we consider the *exchange* and *reverse* [22] operations. In particular, *exchange*$_{i,j}(x)$ exchanges the elements at positions i and j in x, while *reverse*$_{i,j}(x)$ reverses the subsequence between i and j in x. Mutation is performed in lines 5–8 in Algorithm 1, which uses the local permutation operation provided as input to the algorithm. The mutation is global in the sense that there is a nonzero probability to escape local optima. This is achieved by applying several random operations in each iteration, the number of which is drawn from a Poisson distribution with $\lambda = 1$. The justification for this is that the number of local

Algorithm 1: (1+1) EA

```
input: A tree $T = (V, E)$ on $n$ vertices and a local
       permutation operation $M$
1 $x \leftarrow (1, 2, \ldots, n-1)$
2 Randomly shuffle $x$
3 while $f(x) > 0$ do
4     $y \leftarrow x$
5     Choose $s$ from a Poisson distribution with $\lambda = 1$
6     repeat $s$ times
7         Choose $i, j \in \{1, \ldots, n-1\}$ uniformly at random
8         Apply operation $M_{i,j}(y)$
9     if $f(y) \leq f(x)$ then $x \leftarrow y$
```

operations is asymptotically distributed the same as the number of individual bits flipped during the mutation step of the binary (1+1) EA [22].

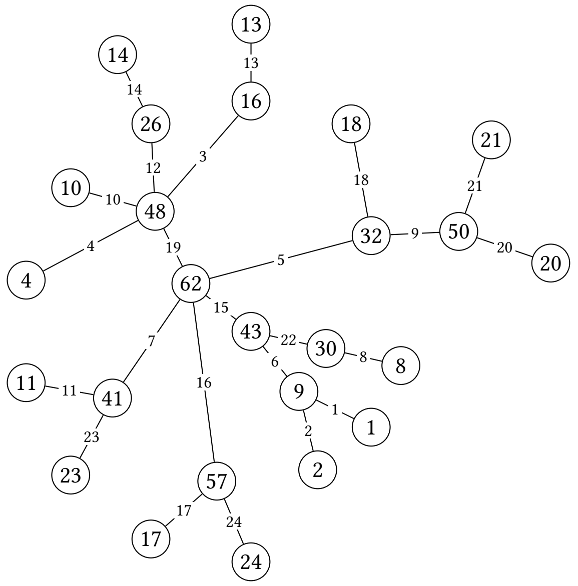


Figure 2: Antimagic labeling for a tree of order 25 found by the (1+1) EA using exchange mutation.

3 CLASSES OF TREES WITH POLYNOMIAL TIME GUARANTEES

To get a better idea of the runtime behavior of evolutionary algorithms on trees, we focus in this section on several subclasses of trees that are known to be antimagic, and prove that a simple EA can find an antimagic labeling in expected polynomial time.

3.1 Paths

Define $P_n = (V, E)$ as the path graph on n vertices, i.e., $V = \{v_1, v_2, \ldots, v_n\}$ where $E = \{v_i v_{i+1} \mid 1 \leq i < n\}$. Every path has an antimagic labeling [13].

	local optima			
n	reverse	exchange	global optima	total points
3	0	0	1	2
4	0	0	2	6
5	0	0	5	24
6	2	2	16	120
7	4	4	91	720
8	56	186	352	5040
9	112	1022	1909	40320
10	1168	7642	10892	362880
11	4536	50608	72243	3628800
12	34876	407210	518168	39916800
13	196990	3337240	4087887	479001600
14	1508118	30023582	34742636	6227020800
15	10615628	284168846	319233359	87178291200

Table 1: Exhaustive count of local and global optima on P_n for $n \leq 15$ for reverse and exchange operators.

3.1.1 *Local Optima.*

Initial experiments with the (1+1) EA on paths exhibited surprisingly fast convergence to optimal solutions, which suggested that local optima are sparse and/or easy to escape. Here we define a reverse (respectively, exchange) local optimum as a sequence x for which $f(x) > 0$, and for all $1 \leq i < j < n$, every y constructed by reversing the segment from i to j (respectively, exchanging the positions i and j) does not improve the fitness, i.e. $f(y) \geq f(x)$. Figure 3 illustrates some local optima for the reverse operator. A global optimum is any point x where $f(x) = 0$, as this would correspond to an antimagic labeling.

For small paths, it is possible to exhaustively enumerate the entire search space in order to examine the structure of the landscape. Table 1 presents the exhaustive enumeration of local optima for both the reverse and exchange operator on all paths P_n with $n \leq 15$. A few interesting characteristics worth noting is that, though the count of local optima is increasing very fast with n, it is also true that, for these small paths, there are more global optima than local optima, suggesting that the local optima may not in general pose a significant problem for local methods like the (1+1) EA. Empirically, the density of global optima is shrinking with n and the density of exchange optima seems to increase to approach the density of global optima, while the density of reverse local optima remains relatively low. This is illustrated in Figure 4.

3.1.2 *Local optima in a restricted neighborhood.*

The structure of the neighborhoods that arise from the reverse and exchange mutation operators on this problem poses a challenge to existence proofs of local optima. Restricting the mutation operation to only exchange or reverse *adjacent* elements in the permutation yields a reduced neighborhood. This coincides to the mutation operations **reverse**$(i, i+1)$ or **exchange** $(i, i+1)$ for $i \in \{1, \ldots, m-1\}$. This operator has been called *adjacent pairwise exchange* or *swap* [23]. It has been investigated in terms of minimal transformation distance for permutations [17] and corresponds to the basic exchange operation used in bubble sort [18]. For this restricted neighborhood, we prove the existence of local optima as follows.

Lemma 3.1. *On P_n where $n \geq 9$ ($m \geq 8$) is odd and $\lfloor \frac{m}{4} \rfloor = \lceil \frac{m}{4} \rceil$, the sequence*

$$\hat{x} := \left(1, m, \tfrac{m}{2}+1, 2, \tfrac{m}{2}+2, 3, \tfrac{m}{2}+3, \ldots, \tfrac{m}{4}, \tfrac{3m}{4}, \tfrac{m}{4}+1, \ldots, m-1, \tfrac{m}{2}\right)$$

is a (nonstrict) local optimum if only adjacent positions are selected to do the reverse (exchange) operation.

Proof. Let $v_1, v_2, \ldots, v_n$ denote the sequence of adjacent vertices in P_n. The vertex weights produced by the local optimum $\hat{x}$ are given by the sequence

$$1, m+1, \frac{3m}{2}+1, \frac{m}{2}+3, \frac{m}{2}+4, \ldots, m, m+1, \ldots, \frac{3m}{2}-1, \frac{m}{2}.$$

Since $m \geq 8$, it holds that $\frac{m}{2}+3 < m+1 < \frac{3m}{2}-1$. Note that all induced vertex weights are unique except for $m+1$, which appears twice.

We show that no single swap between adjacent positions can create a sequence y such that $f(y) < f(\hat{x})$. First of all, we note that the fitness of $\hat{x}$ does not decrease while the positions of edge labels $\{1, m, \frac{3m}{4}, \frac{m}{4}+1\}$ do not change. Therefore, four possible cases exist for the operation on those positions discussed below, obtaining an offspring y of $\hat{x}$.

Case (1). Changing the position of the edge label 1. As the edge label 1 is in the first position, it can only be swapped with the second position with the edge label m. However, the vertex weight $m+1$ is still in the second position of the induced vertex sequence and appears twice.

Case (2). Changing the position of the edge label m. When it is swapped with the edge label 1, the vertex weight $m+1$ still appears twice as the discussion in **Case (1)**. Now we consider swapping it with its adjacent edge label $\frac{m}{2}+1$. After the operation, the new vertex weights produced by the offspring are given by the sequence

$$1, \frac{m}{2}+2, \frac{3m}{2}+1, m+2, \frac{m}{2}+3, \frac{m}{2}+4, \ldots, \frac{3m}{2}-1, \frac{m}{2}.$$

Since $m \geq 8$, it yields that $\frac{m}{2}+3 < m+2 < \frac{3m}{2}-1$. Therefore, the vertex weight $m+2$ appears twice.

Case (3). Changing the position of the edge label $\frac{3m}{4}$. We notice that the vertex weight $m+1$ still appears twice when it was swapped with the edge label $\frac{m}{4}+1$. Then the edge label $\frac{m}{4}$ is considered. After the operation, it has new vertex weights induced by y, which is given as

$$1, m+1, \frac{3m}{2}+1, \frac{m}{2}+3, \ldots, m-2, \frac{3m}{2}-1, m, \frac{m}{2}+1, m+2, \ldots, \frac{3m}{2}-1, \frac{m}{2}.$$

Clearly, the vertex weight $\frac{3m}{2}-1$ appears twice.

Case (4). Changing the position of the edge label $\frac{m}{4}+1$. As mentioned in **Case (3)**, the vertex weight $m+1$ still appears twice while it was swapped with the edge label $\frac{3m}{4}$. Now, we consider the edge label $\frac{3m}{4}+1$. After the operation, the new vertex weights are

$$1, m+1, \frac{3m}{2}+1, \frac{m}{2}+3, \ldots, m, \frac{3m}{2}+1, m+2, \frac{m}{2}+3, m+4, \ldots, \frac{3m}{2}-1, \frac{m}{2}.$$

We can find that the vertex weight $\frac{3m}{2}+3$ appears twice. □

Lemma 3.2. *On P_n where $n \geq 8$ ($m \geq 7$) is even, the sequence*

$$\hat{x} := \left(\tfrac{m+1}{2}, 1, \tfrac{m+1}{2}+1, 2, \ldots, \tfrac{m-3}{2}-1, m-2, m, \tfrac{m-3}{2}, m-1, \tfrac{m-1}{2}\right)$$

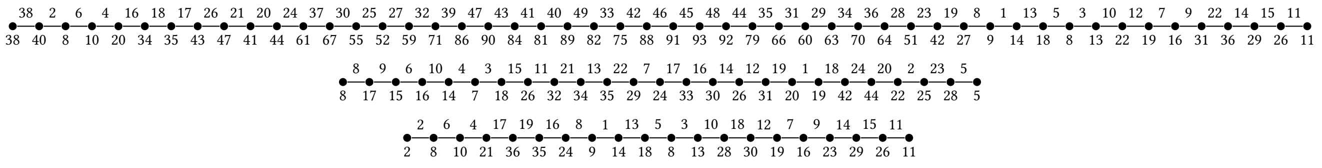


Figure 3: Various path local optima for reverse neighborhood.

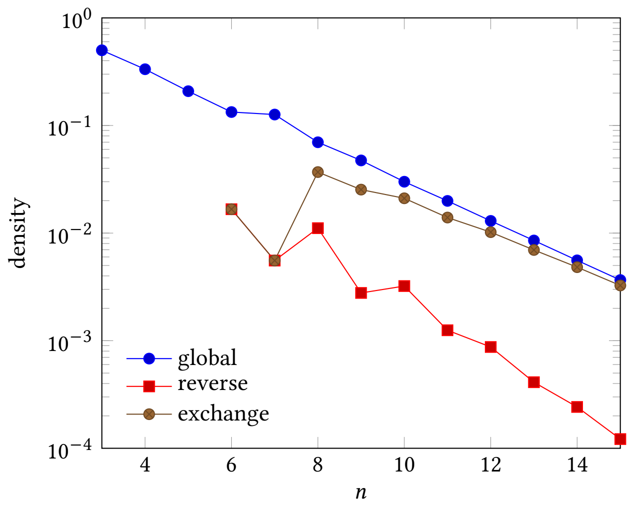


Figure 4: Density of optima on small path problems $n \leq 15$.

is a (nonstrict) local optimum if only adjacent positions are selected to do the reverse (exchange) operation.

Proof. Let $v_1, v_2, \ldots, v_n$ denote the sequence of adjacent vertices in P_n. The vertex weights of P_n induced by the local optimum $\hat{x}$ are given by the sequence

$$\frac{m+1}{2}, \frac{m+3}{2}, \ldots, \frac{3m-9}{2}, 2m-2, \frac{3m-3}{2}, \frac{3m-5}{2}, \frac{3m-3}{2}, \frac{m-1}{2}.$$

Since $m \geq 7$, it holds that $\frac{3m-9}{2} > \frac{m+1}{2}$. We note that only the vertex weight $\frac{3m-3}{2}$ appears twice.

As discussed in Lemma 3.1, the fitness of $\hat{x}$ does not decrease while the positions of edge label $\{m, \frac{m-3}{2}, m-1, \frac{m-1}{2}\}$ do not change. Therefore, four possible cases exist for those edge labels discussed below, obtaining an offspring y of $\hat{x}$ by the adjacent operation.

Case (1). Changing the position of the edge label m. First, we consider swapping it with the edge label $m-2$. After swapping, the sequence of the vertex weights of y becomes

$$\frac{m+1}{2}, \frac{m+3}{2}, \ldots, \frac{3m-5}{2}, 2m-2, \frac{3m-7}{2}, \frac{3m-5}{2}, \frac{3m-3}{2}, \frac{m-1}{2}.$$

The induced vertex weight $\frac{3m-5}{2}$ appears twice exactly. Then, if it is swapped with edge label $\frac{m-3}{2}$, the vertex weight $\frac{3m-3}{2}$ will still appear twice.

Case (2). Changing the position of the edge label $\frac{m-3}{2}$. As we discussed in **Case (1)**, the vertex weight $\frac{3m-3}{2}$ still appears twice while it is swapped with the edge label m. Now, we consider the next adjacent edge label $m-1$. We obtain the new vertex weight sequence produced by y is

$$\frac{m+1}{2}, \frac{m+3}{2}, \ldots, \frac{3m-9}{2}, 2m-2, 2m-1, \frac{3m-5}{2}, m-2, \frac{m-1}{2}.$$

Since $m \geq 7$, it yields $\frac{m+1}{2} < m-2 < \frac{3m-9}{2}$. Therefore, the vertex weight $m-2$ appears twice.

Case (3). Changing the position of the edge label $m-1$. In the previous case, we demonstrated that the vertex weight $m-2$ occurs twice upon being exchanged with the edge label $\frac{m-3}{2}$. Additionally, the vertex weight $\frac{3m-3}{2}$ persists in appearing twice even when it is swapped with $\frac{m-1}{2}$.

Case (4). Changing the position of the edge label $\frac{m-1}{2}$. Given that the edge label $\frac{m-1}{2}$ is situated in the final position, it can solely be interchanged with its adjacent edge label $m-1$ by the operation. Nevertheless, the vertex weight $\frac{3m-3}{2}$ continues to occur twice. □

3.1.3 *Polynomial time guarantee.*

We conjecture that the (1+1) EA using the (unrestricted) reverse operation runs in expected polynomial time on all P_n. We base this conjecture on the observation (e.g., Figure 4) that local optima tend to be sparse and easy to escape. Nevertheless, we are interested in obtaining a rigorous proof of polynomial runtime. In order to facilitate this, we introduce the following auxiliary fitness function that guides the EA out of local optima.

$$g(x) = \sum_{i=1}^{n-1} |x_i - x_{i-1}|, \tag{3}$$

where we define $x_0 = 0$. In effect, this function inscribes extra structure on the search space that facilitates proving runtime guarantees on certain trees. Later, in Section 4.2, we empirically demonstrate that the (1+1) EA often performs better than the version that uses this auxiliary function, likely due to the fact that it would not be slowed down by the extra structure.

This auxiliary function is only useful if it can be used to guide us out of local optima. We will use the following definition.

Definition 3.3. Let G be a graph. We say that g is a *guide* for f on G if the global minimum of g is either a global minimum of f or is in the (symmetric) reverse neighborhood of a global minimum for f.

Note that this property depends on how we associate the elements of the permutation to the edges of the graph. This is a shortcoming of our approach, however, we point out that in all our proofs, the association is somewhat natural.

Using the auxiliary function defined in Equation (3), we modify Algorithm 1 to maintain a population of size two: one that keeps the best-so-far f-value as defined in Equation (2), and one that keeps the best-so-far g-value as defined in Equation (3). In each iteration, an offspring is created by selecting one of the parents uniformly at random for mutation. The parents are replaced by the offspring only if the corresponding function is not worse. The modified EA is listed in Algorithm 2.

Algorithm 2: (2+1) $\text{EA}_{\text{2-obj}}$

```
input: A tree $T = (V, E)$ on $n$ vertices
1 $x \leftarrow (1, 2, \ldots, n-1)$
2 Randomly shuffle $x$
3 $x' \leftarrow x$
4 while $f(x) > 0$ do
5     Select one of $\{x, x'\}$ uniformly at random and store in $y$
6     mutate($y$)
7     if $f(y) \le f(x)$ then $x \leftarrow y$
8     if $g(y) \le g(x')$ then $x' \leftarrow y$
```

Algorithm 2 can be interpreted as a simple bi-objective evolutionary algorithm that only maintains a constant-size subset of the Pareto front defined by the two objective functions.

We now prove that the expected time until (2+1) $\text{EA}_{\text{2-obj}}$ finds an antimagic labeling of P_n for any $n > 2$ is bounded by a polynomial. We adopt the usual notational convention for integral intervals as follows. For any $a, b \in \mathbb{N}$ with $a < b$, denote as $[a\,..\,b] := [a, b] \cap \mathbb{N}$ and $[a] := [1\,..\,a]$. The following two lemmas will be useful here and in upcoming sections.

Lemma 3.4. *Let $(x_1, x_2, \ldots, x_{n-1})$ be a permutation on the elements of $[n-1]$ not equivalent to the identity permutation. Consider the extension $(x_0 := 0, x_1, x_2, \ldots, x_{n-1})$, and let $i := \min\{0 \le i < n-1 \mid |x_i - x_{i+1}| > 1\}$. Then at least one of the following must be true.*

(1) $x_{n-1} < n-1$
(2) *there must exist an x_j with $j > i+1$ such that $x_i < x_j < x_{i+1} < x_{j+1}$.*

Proof. If $x_{n-1} < n-1$ then we are done. Otherwise, suppose $x_{n-1} = n-1$. Define the sets $A := [n-1] \setminus ([x_i] \cup [x_{i+1}\,..\,n-1])$ and $B := [n-1] \setminus [x_{i+1}]$.

It is clear that the choice of i ensures that $x_k = k$ for all $0 \le k \le i$. Thus it holds that $x_i < x_{i+1}$, and since $|x_i - x_{i+1}| > 1$, A must be nonempty. Moreover, since $x_{i+1} < n-1$ (otherwise x is the identity permutation), it holds that B is nonempty.

Since A and B partition the remaining labels, there must be a pair of labels that are adjacent and one is in A one is in B. Furthermore, since $x_{n-1} = (n-1) \in B$, there must be an adjacent pair x_j, x_{j+1} with $x_j \in A$ and $x_{j+1} \in B$, which completes the proof of this case. □

Lemma 3.5. *If g is a guide (as in the sense of Definition 3.3) for f in a graph G, then the (2+1) $\text{EA}_{\text{2-obj}}$ (Algorithm 2) using the reverse operation finds an antimagic labeling of G in $O(n^4)$ time in expectation.*

Proof. Let $(F_t)_{t\in\mathbb{N}}$ and $(F'_t)_{t\in\mathbb{N}}$ be the values of $f(x)$ and $g(x')$ in iteration t. The run time of the (2+1) $\text{EA}_{\text{2-obj}}$ is thus $T = \min\{t \in \mathbb{N} \mid F_t = 0\}$. If at least one of x and x' is not a local optimum for f, then $F_{t+1} < F_t$ with probability at least $\frac{1}{en(n-1)}$. This is because the probability to select a non-local optimum parent is at least $1/2$, the probability to perform a single reverse operation is $1/e$, and the probability of selecting an improving reverse neighbor is at least $2/(n(n-1))$. If both x and x' are local optima for f, then since g is a guide for f, x' is not the minimum of g, which coincides with the identity permutation.

Thus, assume x' is not the identity permutation. By Lemma 3.4, it holds that either $x'_{n-1} < n-1$, or there exists indices $i < j$ such that $x'_i < x'_j < x'_{i+1} < x'_{j+1}$. In the first case, suppose that $x'_k = n-1$ for some $k < n-1$. Then the operation that reverses the segment from k to $n-1$ results in an offspring y with

$$g(y) = g(x') - |x'_{k-1} - (n-1)| + |x'_{k-1} - x'_{n-1}| < g(x'),$$

since $x'_{n-1} < n-1$.

Otherwise, suppose $x'_{n-1} = n-1$. Then the operation that reverses the segment from $i+1$ to j in x' results in an offspring y with

$$\begin{aligned} g(y) = g(x') &- \left(|x'_i - x'_{i+1}| + |x'_j - x'_{j+1}|\right) \\ &+ \left(|x'_i - x'_j| + |x'_{i+1} - x'_{j+1}|\right) \\ < g(x'). \end{aligned}$$

In either case, this operation occurs when x' is selected for mutation with probability $1/2$, and exactly the correct reverse operation is selected with probability $\frac{2}{en(n-1)}$.

We define the potential

$$H_t = \begin{cases} 0 & \text{if } F_t = 0, \\ F_t + F'_t & \text{otherwise.} \end{cases}$$

Since $H_1 \le n + \binom{n}{2} = \frac{n(n+1)}{2}$ and at least one of the events in $\{F_{t+1} < F_t\} \cup \{F'_{t+1} < F'_t\}$ occurs with probability $\Omega(n^{-2})$, the hitting time to zero of H_t, and thus, the running time of the (2+1) $\text{EA}_{\text{2-obj}}$, is at most $O(n^4)$ by the Additive Drift Theorem [14]. □

Theorem 3.6. *The (2+1) $\text{EA}_{\text{2-obj}}$ (Algorithm 2) using the reverse operation finds an antimagic labeling of P_n for $n > 2$ in $O(n^4)$ time in expectation.*

Proof. It suffices to show that g is a guide for f on P_n. Let z be the identity permutation $(1, 2, \ldots, n-1)$. The vertex weights of the edge labeling induced by z are

$$\{\phi_z(v_i) \mid i \in [n]\} = \{2k-1 \mid k \in [n-1]\} \cup \{n-1\}.$$

If n is odd, then $n-1$ is even, and these weights are distinct so $f(z) = 0$. If n is even, then $f(z) = 1$ since $2k-1 = n-1$ when $k = n/2$. Consider the permutation z' obtained from exchanging the elements in position $n-2$ and $n-1$ in z. The set of vertex weights $\{\phi_{z'}(v_i) \mid i \in [n]\}$ of the edge labeling induced by z' is

$$\{2k-1 \mid k \in [n-3]\} \cup \{2(n-2)\} \cup \{2(n-1)-1\} \cup \{n-2\}.$$

The only even weights are $2(n-2)$ and $n-2$, and these are not equal for $n > 2$. Therefore, the weights are distinct and $f(z') = 0$. Since z' can be reached from z by a reverse operation on the last two indices, g is a guide for f on P_n and the claim follows from Lemma 3.5. □

3.2 Combs

We now show that the technique for proving a polynomial runtime guarantee on paths used in Section 3.1 can be generalized to other interesting classes of trees.

A comb is a graph on $n = 2\ell$ vertices obtained by joining a single pendant edge to each vertex of a path of length ℓ (see Figure 5). A comb is a type of *caterpillar tree*: that is, a tree that results in a path when all the leaves are deleted. Lozano et al. [19] proved that all caterpillar trees (and hence combs) are antimagic. Formally, we define the comb C_n for n even as the graph $C_n = (V, E)$ with $V = \{v_1, v_2, \ldots, v_{n/2}, u_1, u_2, \ldots, u_{n/2}\}$, $E = \{v_i v_{i+1} \mid 1 \le i < n/2\} \cup \{v_i u_i \mid 1 \le i \le n/2\}$. The v_i vertices form a $P_{n/2}$ subgraph, which we call the *spine*, and each u_i vertex has degree one.

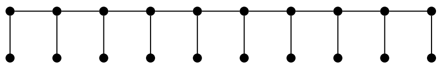


Figure 5: A comb graph on $n = 20$ vertices.

We identify a permutation x of $[n-1]$ with the edges of C_n as follows. For $1 \le i \le n/2$, the element x_i corresponds to the label of the pendant edge $v_i u_i$. For $n/2 < i < n$, the element x_i corresponds to the label of the spine edge $v_{(i-n/2)} v_{(i-n/2+1)}$.

Theorem 3.7. *The (2+1) EA$_{2\text{-obj}}$ (Algorithm 2) using the reverse operation finds an antimagic labeling of a comb on n vertices in $O(n^4)$ time in expectation.*

Proof. Similar to the proof of Theorem 3.6, we show that progress is always possible in at least one of the objectives. Since g by itself is invariant to the graph type, the arguments about g are the same, and thus there is always a reverse operation that improves either f or g (or both).

It remains to prove that the identity permutation $z = (1, 2, \ldots, n-1)$ is not a local optimum for f on C_n. Let z' be the permutation obtained by applying the reverse operation on the subsequence of z from index $n/2 + 1$ to $n - 1$, that is,

$$z' = (1, 2, \ldots, n/2, n-1, n-2, \ldots, n/2+1)$$

We argue that $f(z') = 0$. In particular, the weights for all u_i under z' are $\{1, 2, \ldots, n/2\}$.

The vertex weights for v_1 and $v_{n/2}$ are n and $n+1$, respectively. The remaining vertex weights along the spine path are $2n - (k-1)$ for $1 < k < n/2$. Thus the n vertices have weights

$$\{\phi_{z'}(v_i) \mid i \in [n]\} = \{1, 2, \ldots, n/2\} \cup \{n, n+1\} \cup \bigcup_{k=1}^{n/2-2} \left\{1 + \frac{3n}{2} + k\right\}$$

which must all be unique. Therefore $f(z') = 0$. The run time bound then follows from Lemma 3.5. □

3.3 p-Uniform Caterpillar Trees

A caterpillar tree is *p-uniform* if it is a path of length k to which each vertex is joined $p > 0$ pendants (the comb graph is thus a 1-uniform caterpillar tree). Let $C_{n,k,p} = (V, E)$ be a caterpillar tree where

$$V = \{v_i \mid i \in [k]\} \cup \left\{u_{i,j} \mid i \in [k], j \in [p]\right\}.$$

Figure 6 illustrates an example for $p = 3$. For each $i \in [k-1]$ and each $j \in [p]$, we associate the edge label $x_{(i-1)p+j}$ to the edge $v_i u_{i,j}$ and for each $i \in [k-1]$ the edge label x_{kp+i} to the edge $v_i v_{i+1}$.

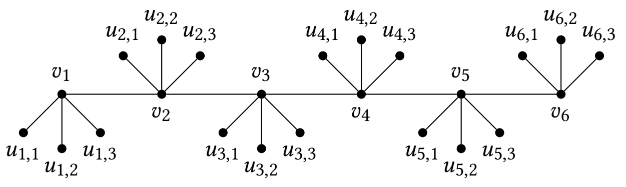


Figure 6: A 3-uniform caterpillar tree.

Theorem 3.8. *The (2+1) EA$_{2\text{-obj}}$ (Algorithm 2) using the reverse operation finds an antimagic labeling of a p-uniform caterpillar tree on n vertices in $O(n^4)$ time in expectation.*

Proof. Again we argue that progress is always possible in at least one objective. If both x and x' are f-local optima, then as long as x' is not the identity permutation, x' is not a g-local optimum (by Lemma 3.4).

Now suppose $x' = z$ where z is the identity permutation. If $f(z) = 0$, then we are done since x' was accepted, it also has the lowest f-value, and no point x would have a larger f-value. Otherwise, suppose $f(z) > 0$.

Consider the vertex weights under the edge labeling induced by z. Then the following must be true.

(1) for all i, j, $\phi_z(u_{i,j}) = (i-1)p + j < kp + p$.
(2) $\phi_z(v_1) = kp + \frac{p(p+1)}{2} + 1 < 2kp + \frac{p(p+1)}{2} + 1$
(3) for all $i \in [2\,..\,k-1]$, $\phi_z(v_i) = 2kp + \frac{p(p+1)}{2} + 1 + (i-1)(p^2+2)$
(4) $\phi_z(v_k) = kp + \frac{p(p+1)}{2} + (k-1)(p^2+1)$

Therefore, we have for all i, j that $\phi_z(u_{i,j}) < \phi_z(v_1) < \phi_z(v_k)$. Note that $\phi_z(v_2) - \phi_z(v_1) = p^2 + 2 + kp$ and for all $i \in [2\,..\,k-2]$ we have $\phi_z(v_{i+1}) - \phi_z(v_i) = p^2 + 2$. Thus for all $i \in [k-2]$ it always holds that

$$\phi_z(v_{i+1}) - \phi_z(v_i) \ge p^2 + 2.$$

As we assume $f(z) > 1$, there must be some $i' < k$ such that $\phi_z(v_{i'}) = \phi_z(v_k)$. Solving for i',

$$i' = \frac{\ell(p^2 - p + 1)}{p^2 + 2}.$$

Since i' must be unique, it follows that this is the only nonunique weight pair. Consider the reverse operation that swaps neighboring positions $k(p+1) - 2$ and $k(p+1) - 1$ to produce z'. This reduces the weight of v_k by one and increases the weight of v_{k-2} by one, changing no other weights, i.e.,

$$\phi_{z'}(v) = \begin{cases} \phi_z(v) - 1 & \text{if } v = v_k, \\ \phi_z(v) + 1 & \text{if } v = v_{k-2}, \\ \phi_z(v) & \text{otherwise.} \end{cases}$$

Since the spine weights differ by at least p^2+2, the resulting weights must be unique and thus $f(z') = 0$. □

3.4 Uniform Spider Trees

A *spider* is a tree that has at most one vertex of degree greater than two [12], which is called the center vertex. Every path from the center vertex to a pendant vertex is called a *leg* of the spider. A *uniform spider* is a spider with all legs the same length [24].

Let $S(p,k) = (V,E)$ denote a uniform spider with p legs of length k each, i.e.,

$$V = \{v\} \cup \{u_{i,j} \mid i \in [p], j \in [k]\}],$$

where v is the center vertex and $u_{i,j}$ is the jth vertex on the ith leg, where $u_{i,1}$ is an endpoint. The graph $S(p,1)$ is equivalent to the p-star graph $K_{1,p}$.

For each $i \in [p]$ and each $j \in [k-1]$ we associate the edge label $x_{(j-1)p+i}$ to the edge $u_{i,j}u_{i,j+1}$ and for each $i \in [p]$ the edge label $x_{(k-1)p+i}$ to the edge $u_{i,k}v$

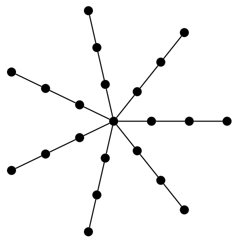


Figure 7: The uniform spider tree $S(7,3)$.

Theorem 3.9. *When $p > 1$, the (2+1) EA$_{2\text{-}obj}$ (Algorithm 2) using the reverse operation finds an antimagic labeling of a uniform spider $S(p,k)$ on $n = pk$ vertices in $O(n^4)$ time in expectation.*

Proof. Similar to before we argue that there is always progress in one of the objectives. If both x and x' are f-local optima and x' is not the identity permuation, then x' is not a g-local optimum (by Lemma 3.4)

Now suppose $x' = z$ where z is the identity permuation. The following must be true about the vertex weights under the edge labeling induced by z.

(1) for all $i \in [p]$, $\phi_z(u_{i,1}) = i$
(2) for all $i \in [p]$ and $j \in [2..k]$, then $\phi_z(u_{i,j}) = (2j-3)p + 2i$
(3) $\phi_z(v) = (k-1)p^2 + \frac{p(p+1)}{2}$

Therefore, for all $i \in [p]$ and $j \in [2..k]$ we have that $\phi_z(u_{i,1}) < \phi_z(u_{i,j}) < \phi_z(v)$. Note that $\phi_z(v)$ is the sum of the p largest edge labels, so this will never be equal to any other vertex weights for $p > 1$. Also for any $i \neq i'$ and $j < j'$ we have $u_{i,j} < u_{i',j'}$ and for any $i < i'$ we have $u_{i,j} < u_{i',j}$.

Therefore $f(z) = 0$ and we are done. □

3.5 Perfect Binary Trees

A *perfect binary tree* is a complete depth d binary tree on exactly $2^{d+1} - 1$ vertices, i.e., all leaves are uniformly at distance d from the root. We associate the edge labels as follows. The first 2^d edges correspond to the edges farthest from the root (incident to the leaves) from left to right. The next 2^{d-1} in the next level up, and so on (see Figure 8).

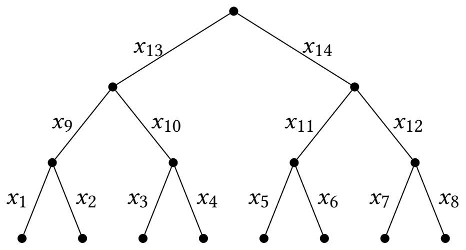


Figure 8: A perfect binary tree of depth 3.

Theorem 3.10. *The (2+1) EA$_{2\text{-}obj}$ (Algorithm 2) using the reverse operation finds an antimagic labeling of a perfect binary tree of depth $d > 0$ and $n = 2^{d+1} - 1$ vertices in $O(n^4)$ time in expectation.*

Proof. Let T be a perfect binary tree of depth d. Every nonroot internal node is incident to exactly three edges labeled by elements x_k, x_{k+1} (where k is odd) and $x_{2^d + \frac{k+1}{2}}$. The root is incident to the two edges labeled by the elements $x_{2^{d+1}-3}$ and $x_{2^{d+1}-3}$. The leaves are each incident to a single edge.

Let $z = (1, 2, \ldots, 2^{d+1} - 2)$ be the identity permutation on the edges. The vertex weights of the leaves under this edge labeling is the set $\{1, 2, \ldots, 2^d\}$ and are pairwise distinct. Let v be a nonroot internal node. Since $z_k = k$ for all $k \in \{1, \ldots, n-1\}$, we have

$$\phi_z(v) = k + (k+1) + 2^d + \frac{k+1}{2} = 2^d + \frac{5k+3}{2}.$$

Thus the vertex weights of nonroot internal nodes are all strictly larger than the weights on the leaves, and also must be pairwise distinct.

Finally, let r be the root of T. We have

$$\phi_z(r) = 2^{d+1} - 3 + 2^{d+1} - 2 == 2n - 3.$$

Since the set of $T - \{r\}$ have unique vertex weights, and the largest leaf weight is $2^d = (n+1)/2 < 2n-3$, the only collision could occur with the root r and some nonroot internal node v.

Suppose $\phi_z(r) = \phi_z(v)$ for $v \neq r$. Then

$$\frac{n+1}{2} + \frac{5k+3}{2} = 2n - 3.$$

Solving for k, we find the left child edge of v must have index $3n/5 - 2$. If n is not a multiple of 5, then v cannot exist (as k must be an integer).

If n is a multiple of 5, then $(2^{d+1} - 1)/5$ is an integer, which is only possible if $d \equiv 3 \pmod 4$.

Thus, if d is not congruent to $3 \pmod 4$, z is an antimagic labeling for T. Otherwise, there is exactly on repeated vertex weight: at the root of T and in index $3n/5 - 2$. The (adjacent) reverse operation that exchanges $3n/5 - 1$ and $3n/5$ would change the vertex weights on which it is incident by exactly one. The closest vertex weights of nonroot internal vertices differ by $\frac{5(k+2)+3}{2} - \frac{5k+3}{2} = 5$ so this swap introduces no new violations, but removes the violation with the root. The claim follows by Lemma 3.5. □

4 COMPUTATIONAL RESULTS ON TREES

In order to confirm Conjecture 1.1 for all trees up to order 25, we use the exhaustive unrooted tree generation procedure of Wright

n	avg. (sec)	max. (sec)	total (sec)	# trees
3	0.00083	0.00289	0.00827	1
4	0.00064	0.00243	0.01284	2
5	0.00076	0.00229	0.02283	3
6	0.00082	0.00463	0.04938	6
7	0.00083	0.00578	0.09168	11
8	0.00094	0.00664	0.21534	23
9	0.00084	0.00639	0.39713	47
10	0.00088	0.00984	0.93479	106
11	0.00093	0.01492	2.19716	235
12	0.00096	0.00993	5.30864	551
13	0.00100	0.01159	13.02400	1301
14	0.00095	0.01119	29.99961	3159
15	0.00077	0.01370	59.96971	7741
16	0.00079	0.01401	152.40342	19320
17	0.00081	0.02418	392.47724	48629
18	0.00086	0.02501	1060.25184	123867
19	0.00088	0.02531	2811.65187	317955
20	0.00092	0.02520	7584.64095	823065
21	0.00096	0.03307	20628.32180	2144505
22	0.00101	0.02930	57012.50105	5623756
23	0.00121	0.03276	179885.20517	14828074
24	0.00108	0.14744	423725.30242	39299897
25	0.00109	0.22096	1137836.63134	104636890

Table 2: Wall-clock time statistics for all trees of order $n \leq 25$.

n	# superficial	# trees	% superficial
3	1	1	100
4	1	2	50
5	1	3	33.33333
6	1	6	16.66667
7	2	11	18.18182
8	2	23	8.69565
9	4	47	8.51064
10	2	106	1.88679
11	4	235	1.70213
12	11	551	1.99637
13	7	1301	0.53805
14	16	3159	0.50649
15	12	7741	0.15502
16	14	19320	0.07246
17	30	48629	0.06169
18	45	123867	0.03633
19	61	317955	0.01919
20	99	823065	0.01203
21	134	2144505	0.00625
22	210	5623756	0.00373
23	313	14828074	0.00211

Table 3: Count of trees with an antimagic labeling already in all ten initial iterations.

et al. [25]. For each $n \in \{3, 4, \dots, 25\}$, we generated all 167,879,144 trees of order n and ran the (1+1) EA (Algorithm 1) using the exchange operator for mutation. We ran the (1+1) EA 10 times on each tree, and the (1+1) EA successfully found an antimagic labeling in every single run on every single tree[1].

These results were generated on a heterogeneous high-performance Linux cluster with 128 core AMD compute nodes. Table 2 reports the average, maximum and total wall-clock time over all runs for each n. A visual representation of this is illustrated on a logarithmic plot in Figure 9d.

The (1+1) EA always finds an antimagic labeling in a fraction of a second. The computational bottleneck comes from the number of trees to check of each order, which is reported in the final column of the table (see also [16]). The total computation time is roughly 509 CPU hours.

4.1 Superficial trees

For some trees, every valid edge labeling is already an antimagic labeling. An obvious example is the star graph $K_{1,n-1}$: the bipartite graph with a single hub node v_1 connected to $n-1$ spoke nodes $\{v_2, \dots, v_n\}$, i.e., $E = \{v_1 v_i \mid i \in [2..n]\}$. In any edge labeling x of $K_{1,n-1}$, the hub node has weight $\phi_x(v_1) = n(n-1)/2$, and the weights at the $n-1$ spoke vertices are $\{\phi_x(v_i) \mid i \in [2..n]\} = [n-1]$. Therefore, $|\{\phi_x(v_i) \mid i \in [n]\}| = n$ for all edge labelings x, and we would expect at least one tree per order to require no search as the initial search point already corresponds to an antimagic labeling. We call such trees *superficial*.

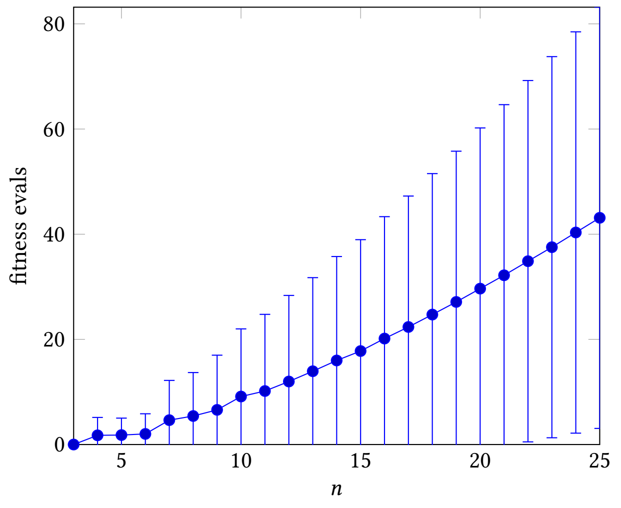
To explore how often this occurs in trees up to order 23, we report the count of trees for which the initial labeling was antimagic in all ten runs of the (1+1) EA in Table 3 together with the proportion of trees with of each order for which we observed this effect. Identifying these trees might be a good starting point for determining what kinds of trees other than $K_{1,n-1}$ are superficial.

4.2 (1+1) EA vs. (2+1) EA$_{\text{2-obj}}$

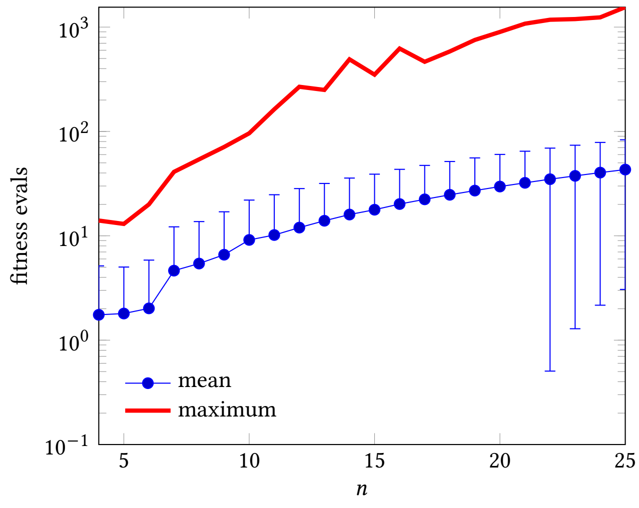
In Section 3 we proved that the (2+1) EA$_{\text{2-obj}}$ has a polynomial-time run time guarantee on certain classes of tree. We employed the (2+1) EA$_{\text{2-obj}}$ to manage the problem of complicated local optima in the space. Nevertheless, we conjecture that due to the sparsity and tractability of escaping these optima, the running time of the (1+1) EA is not generally worse than that of the (2+1) EA$_{\text{2-obj}}$. Moreover, the polynomial bounds proved in Section 3 are likely far from tight, as we pessimistically assume there is only one improving move at any time. To better understand the true run time character of the (1+1) EA and (2+1) EA$_{\text{2-obj}}$, and also investigate the difference between the reverse, exchange and adjacent exchange operations, we measure the number of fitness evaluations required to find antimagic labelings on the various tree classes introduced in Section 3.

For these experiments, for each graph type and size, except where otherwise noted we ran each algorithm for 100 trials and measured the average and standard deviation of the runtime over these trials. The runtime for path graphs of up to size n is plotted in Figure 10. In Figure 11, we plot the runtime for combs of order 6 to 48.

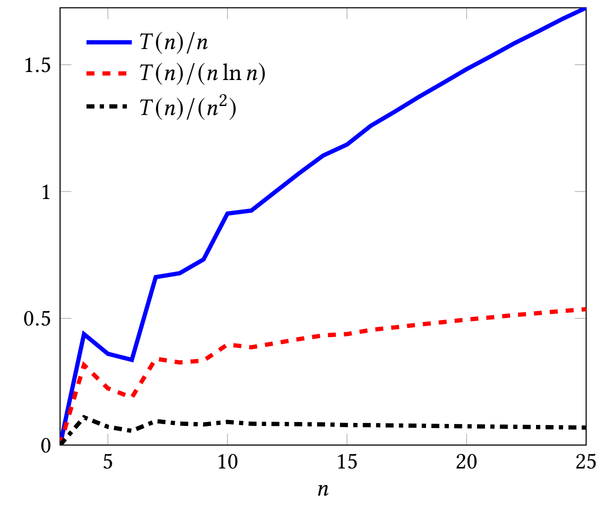
[1]All antimagic tree labelings are available in an online database at https://doi.org/10.5281/zenodo.8146165.



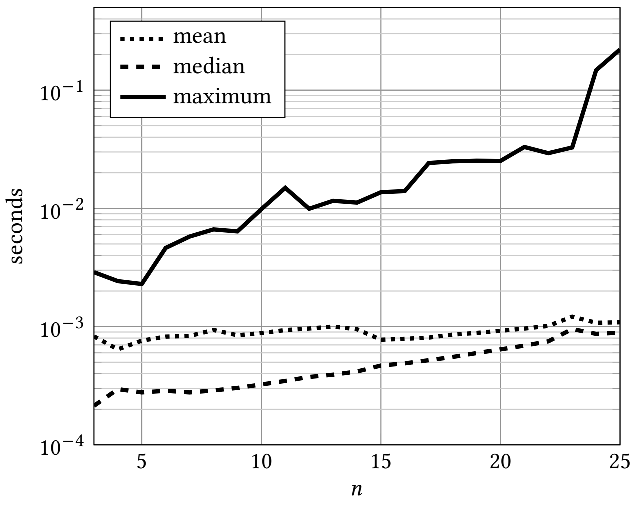
(a) Mean running time of (1+1) EA on all trees. Error bars denote standard deviation.



(b) Logarithmic plot of mean and maximum running time of (1+1) EA over all trees. Error bars denote standard deviation.



(c) Average run times $T(n)$ divided by various functions of n.



(d) Wall-clock time statistics as a function of n.

Figure 9: Run time results of the (1+1) EA on all trees of order $n \leq 25$.

Three interesting properties can be found in these figures. First, the (1+1) EA tends to perform slightly better than the (2+1) $\text{EA}_{\text{2-obj}}$, suggesting that the auxiliary function introduced, though it effectively eliminates local optima, also slows the optimization process down. Second, the reverse operator performs best on paths, which is not too surprising, but we find the opposite to be true on combs. Third, the pairwise adjacent exchange operation exhibits slightly superior performance compared to the exchange operation on combs, but on paths, the opposite is true.

In Figure 12 we plot the data for 3-uniform caterpillars, varying the path length from 3 to 49, yielding trees of order $n = 12, 16, 20, \ldots, 196$. For p-uniform spiders, we vary the leg length from 3 to 9. In Figure 13 we plot the results for 3-uniform spiders, resulting in tree sizes of $n = 10, 13, 16, \ldots, 28$ and in Figure 14 we plot the data for 4-uniform spiders, resulting in tree size of $n = 13, 17, 21, \ldots, 37$.

Finally, for perfect binary trees, we vary the depth from 1 to 8 resulting in tree sizes from $n = 3, \ldots, 511$. We note that due to the larger n values required, we only ran 50 trials each for each algorithm on each tree. The reverse mutation performs very poorly compared to exchange in this case, most likely because the reverse operation cannot exploit graph structure in the same way it can on paths. The exchange operator performs exceptionally well for both the (1+1) EA and the (2+1) $\text{EA}_{\text{2-obj}}$ on perfect binary trees.

5 CONCLUSION

In this paper, we have investigated simple evolutionary search strategies applied to quickly searching for antimagic labelings of trees. Using this approach, we have confirmed the antimagic tree conjecture up to trees of order 25. For certain tree classes with

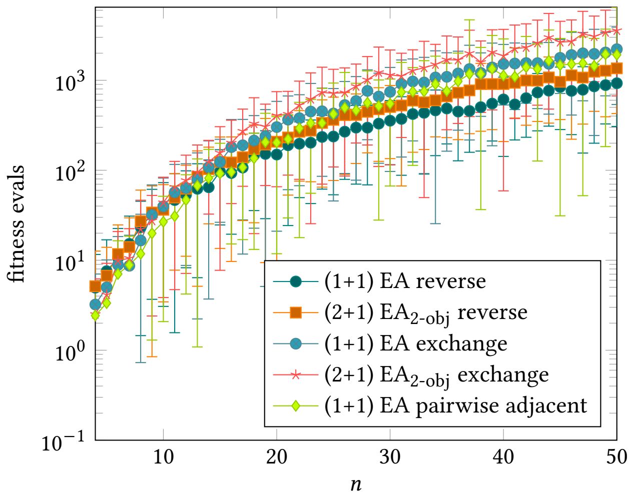


Figure 10: Empirical run time of the (1+1) EA and (2+1) EA$_{2\text{-obj}}$ on paths P_n using reverse and exchange mutation. Error bars denote standard deviation.

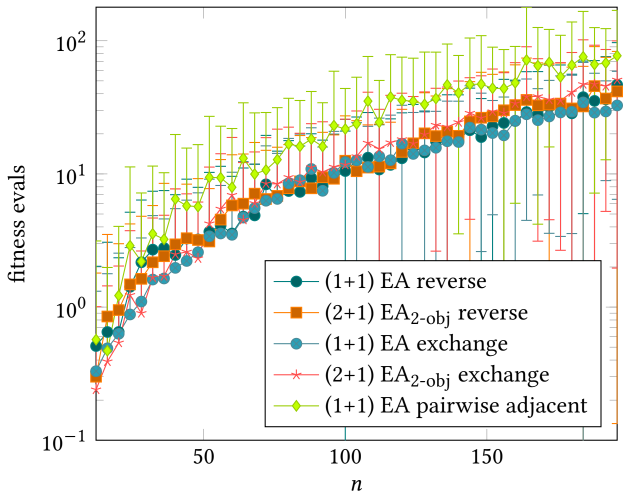


Figure 12: Empirical run time of the (1+1) EA and (2+1) EA$_{2\text{-obj}}$ on 3-uniform caterpillars using reverse and exchange mutation. Error bars denote standard deviation.

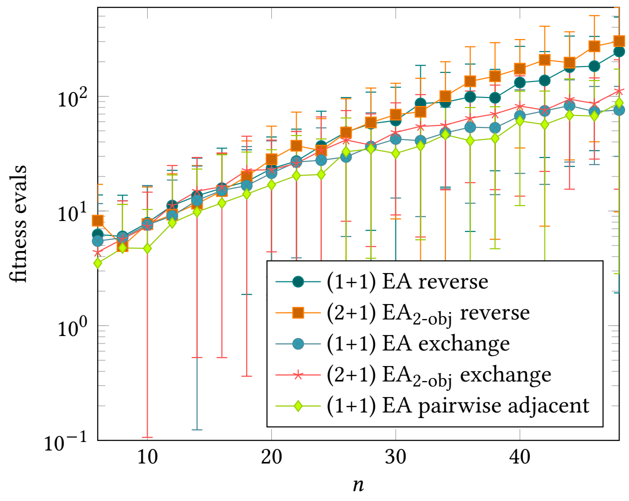


Figure 11: Empirical run time of the (1+1) EA and (2+1) EA$_{2\text{-obj}}$ on combs C_n using reverse and exchange mutation. Error bars denote standard deviation.

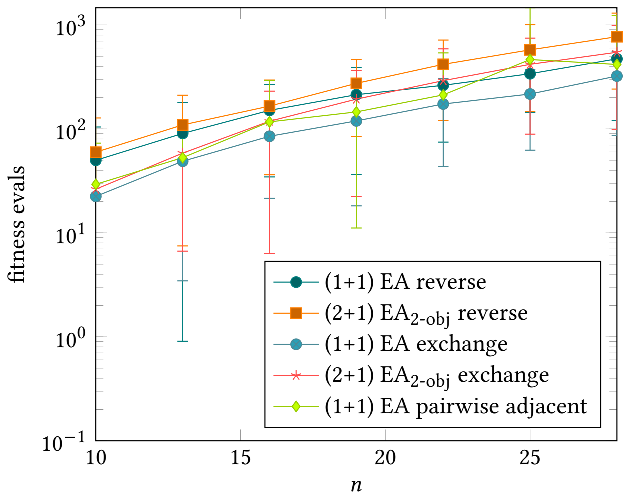


Figure 13: Empirical run time of the (1+1) EA and (2+1) EA$_{2\text{-obj}}$ on 3-uniform spiders using reverse and exchange mutation. Error bars denote standard deviation.

relatively elementary structure, we have proved rigorous run time bounds that guarantee polynomial time efficiency in expectation.

There are many directions for future work that remain open. An obvious direction is to continue to investigate the conjecture for higher order trees. Similar to work on the related graceful tree conjecture, it might be reasonable to explore hybrid methods that perform more structured search on the trees in order to push the limit to higher n. From a theoretical perspective, we would also anticipate tighter upper bounds on the running time of the (2+1) EA$_{2\text{-obj}}$ on paths, combs and caterpillar trees, as well as broadening the results to other classes of tree.

ACKNOWLEDGEMENTS

This work was supported by the National Science Foundation under grant 2144080. The authors acknowledge the Minnesota Supercomputing Institute (MSI) at the University of Minnesota for providing resources that contributed to the research results reported within this paper. http://www.msi.umn.edu

REFERENCES

[1] Hafiz Usman Afzal, Ahmed Alamer, and Muhammad Javaid. 2022. Computing Antimagic Labeling of Lattically Designed Symmetric Networks. *IEEE Access* 10 (2022), 32394–32405. https://doi.org/10.1109/ACCESS.2022.3160715

[2] R. E. L. Aldred and Brendan D. McKay. 1998. Graceful and harmonious labellings of trees. *Bulletin of the Institute of Combinatorics and its Applications* 23 (1998),

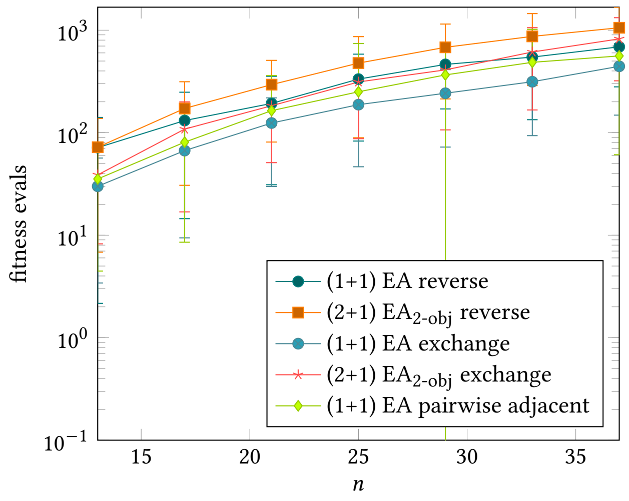


Figure 14: Empirical run time of the (1+1) EA and (2+1) $EA_{2\text{-obj}}$ on 4-uniform spiders using reverse and exchange mutation. Error bars denote standard deviation.

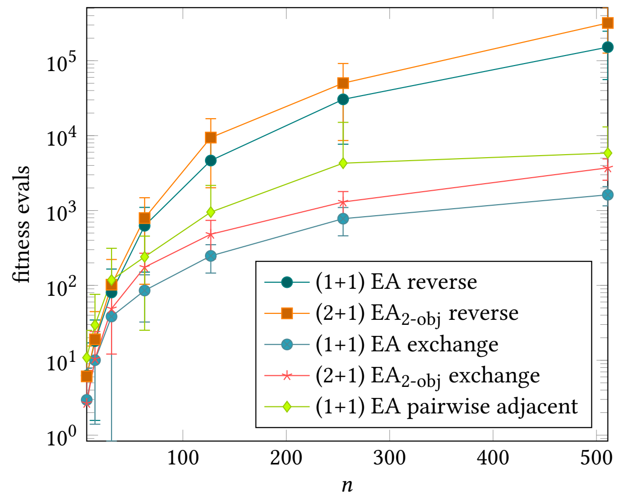


Figure 15: Empirical run time of the (1+1) EA and (2+1) $EA_{2\text{-obj}}$ on perfect binary trees using reverse and exchange mutation. Error bars denote standard deviation.

69–72.
[3] N. Alon, G. Kaplan, A. Lev, Y. Roditty, and R. Yuster. 2004. Dense Graphs Are Antimagic. *J. Graph Theory* 47, 4 (dec 2004), 297–309.
[4] Martin Bača, Mirka Miller, Joe Ryan, and Andrea Semaničová-Feňovčíková. 2019. *Magic and Antimagic Graphs: Attributes, Observations and Challenges in Graph Labelings.* Springer.
[5] Nana Cabo Bizet, Cesar Damian, Oscar Loaiza-Brito, Damián Kaloni Mayorga Peña, and J. A. Montañez-Barrera. 2020. Testing Swampland Conjectures with Machine Learning. https://doi.org/10.48550/ARXIV.2006.07290
[6] Luke Branson and Andrew M. Sutton. 2022. Evolving Labelings of Graceful Graphs. In *Proceedings of the Genetic and Evolutionary Computation Conference* (Boston, Massachusetts) *(GECCO '22)*. Association for Computing Machinery, New York, NY, USA, 195–203. https://doi.org/10.1145/3512290.3528855
[7] Curtis Bright, Dragomir Ž. Đoković, Ilias Kotsireas, and Vijay Ganesh. 2019. The SAT+CAS method for combinatorial search with applications to best matrices. *Annals of Mathematics and Artificial Intelligence* 87, 4 (01 Dec 2019), 321–342. https://doi.org/10.1007/s10472-019-09681-3
[8] Feihuang Chang, Yu-Chang Liang, Zhishi Pan, and Xuding Zhu. 2016. Antimagic Labeling of Regular Graphs. *Journal of Graph Theory* 82, 4 (2016), 339–349. https://doi.org/10.1002/jgt.21905 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/jgt.21905
[9] Jacqueline W. Daykin, Costas S. Illopolous, Mirka Miller, and Oudone Phanalasy. 2015. Antimagicness of Generalized Corona and Snowflake Graphs. *Mathematics in Computer Science* 9 (2015), 105–111. https://doi.org/10.1007/s11786-014-0213-x
[10] Jon Ernstberger and Andy D. Perkins. 2013. *A Metaheuristic Search Technique for Graceful Labels of Graphs.* Technical Report. LaGrange College. Presented at the 44th annual Southeastern International Conference on Combinatorics, Graph Theory, and Computing.
[11] Wenjie Fang. 2010. A Computational Approach to the Graceful Tree Conjecture. *CoRR* abs/1003.3045 (2010). arXiv:1003.3045 http://arxiv.org/abs/1003.3045
[12] Joseph A. Gallian. 2022. A dynamic survey of graph labeling. *Electronic Journal of Combinatorics* (2022). Dynamic Surveys #DS6.
[13] Nora Hartsfield and Gerhard Ringel. 1990. *Pearls in Graph Theory.* Academic Press, Boston.
[14] Jun He and Xin Yao. 2004. A study of drift analysis for estimating computation time of evolutionary algorithms. *Nat. Comput.* 3, 1 (2004), 21–35. https://doi.org/10.1023/B:NACO.0000023417.31393.c7
[15] Michael Horton. 2003. *Graceful Trees: Statistics and Algorithms.* Bachelor's Thesis. University of Tasmania.
[16] OEIS Foundation Inc. 2023. Entry A000055 in The On-Line Encyclopedia of Integer Sequences. https://oeis.org/A000055.
[17] Mark Jerrum. 1985. Finding minimum-length generator sequences. *Theoretical Computer Science* 36 (1985), 265–289.
[18] Donald Knuth. 1998. *The Art of Computer Programming, Volume 3: Sorting and Searching.* Addison-Wesley.
[19] Antoni Lozano, Mercè Mora, Carlos Seara, and Joaqíin Tey. 2021. Caterpillars are Antimagic. *Mediterranean Journal of Mathematics* 18, 39 (2021).
[20] Houra Mahmoudzadeh and Kourosh Eshghi. 2007. A Metaheuristic Approach to the Graceful Labeling Problem of Graphs. In *2007 IEEE Swarm Intelligence Symposium.* 84–91. https://doi.org/10.1109/SIS.2007.368030
[21] M. Saqib Nawaz, M. Zohaib Nawaz, Osman Hasan, Philippe Fournier-Viger, and Meng Sun. 2021. An evolutionary/heuristic-based proof searching framework for interactive theorem prover. *Applied Soft Computing* 104 (2021), 107200. https://doi.org/10.1016/j.asoc.2021.107200
[22] Jens Scharnow, Karsten Tinnefeld, and Ingo Wegener. 2004. The analysis of evolutionary algorithms on sorting and shortest paths problems. *J. Math. Model. Algorithms* 3, 4 (2004), 349–366. https://doi.org/10.1007/s10852-005-2584-0
[23] Tommaso Schiavinotto and Thomas Stützle. 2007. A review of metrics on permutations for search landscape analysis. *Computers and Operations Research* 34 (2007), 3143–3153. https://doi.org/10.1016/j.cor.2005.11.022
[24] Pramod Shinde, Samina Boxwala, Aditi Phadke, Nilesh Mundlik, and Vikas Jadhav. 2022. Product Dimension of Uniform Spider. *Palestine Journal of Mathematics* 11, 4 (2022), 276–281.
[25] Robert Alan Wright, Bruce Richmond, Andrew Odlyzko, and Brendan D. McKay. 1986. Constant Time Generation of Free Trees. *SIAM J. Comput.* 15, 2 (1986), 540–548. https://doi.org/10.1137/0215039