Eastern Kentucky University

Encompass

January 2018

Results on the Gold Grabbing Game

Stephen Acampa
Eastern Kentucky University

Follow this and additional works at: https://encompass.eku.edu/etd

Part of the Mathematics Commons

RESULTS ON THE GOLD GRABBING GAME
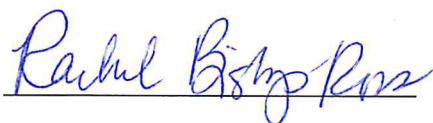
By:

STEPHEN ACAMPA

THESIS APPROVED:

Chair, Advisory Committee

Member, Advisory Committee

Member, Advisory Committee

Dean, Graduate School

STATEMENT OF PERMISSION TO USE

In presenting this thesis in partial fulfillment of the requirements for a Master's degree at Eastern Kentucky University, I agree that the Library shall make it available to borrowers under rules of the Library. Brief quotations from this thesis are allowable without special permission, provided that accurate acknowledgment of the source is made. Permission for extensive quotation from or reproduction of this thesis may be granted by my major professor, or in her absence, by the Head of Interlibrary Services when, in the opinion of either, the proposed use of the material is for scholarly purposes. Any copying or use of the material in this thesis for financial gain shall not be allowed without my written permission.

Signature:

Date:

4/9/18

RESULTS ON THE GOLD GRABBING GAME

BY

STEPHEN ACAMPA

Submitted to the Faculty of the Graduate School of

Eastern Kentucky University

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

2018

Abstract

In this paper, we will contribute to research on a Graph Theory problem known as the Gold Grabbing Game. The game consists of two players and a tree in which each vertex has a positive integer value of gold. Players take turns removing leaves from the tree and deleting the associated edge until the graph is entirely empty. A winning condition is acquiring at least half of the total gold. Existing research shows that for a tree with an even number of vertices, Player 1 can always win.

It can also be shown via simple examples that for a tree with an odd number of vertices, the game board may favor Player 1 or Player 2, depending on the configuration of the tree, the integer values at a given vertex, or both. We will expand on the reason for Player 1's advantage on even trees and attempt to clarify the winning strategy, while also expanding on the case of an odd tree and various winning scenarios for Player 1 or 2.

Table of Contents

List of Figures

1 Introduction to Graph Theory

In this section, we will briefly describe the field of graph theory and introduce some basic terminology, which will be used throughout the remainder of the work. Broadly put, graph theory utilizes the study of graphs which model relationships between objects. In this context, "graph" does not refer to a bar or pie chart that represents data visually but rather a collection of vertices and edges where visual representation has little to no impact. An example of a graph is shown in Figure 1 below:
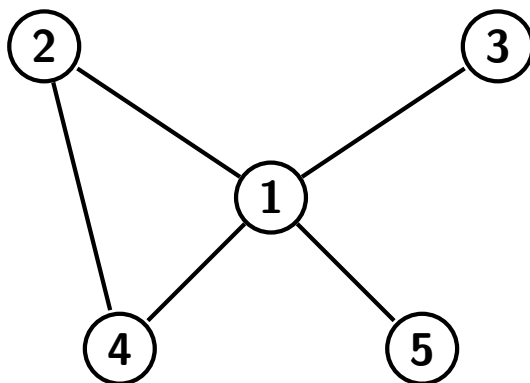
Figure 1: Example Graph

Each circle is called a vertex (sometimes called a *node* depending on book or resource) and is the most fundamental unit of a graph. Vertices are connected by edges. Each edge represents some relationship between the two. Two vertices are said to be adjacent if an edge exists between them. For clarity throughout the rest of the work, several definitions pertaining to graph theory and our main topic will be given below.

Definition. A *graph* G is an ordered pair (V, E) consisting of a vertex set $V(G)$, an edge set $E(G)$, and an incidence relation that associates each edge with two vertices.

Definition. A *path* is a sequence of edges taken when traveling from one vertex to another.

Definition. A graph G is said to be *connected* if for any two vertices $u, v \in G$, there exists a path from u to v.

Definition. The *degree* of a vertex is the number of edges connected to it.

Definition. A *cycle* is a finite path that begins and ends at the same vertex without traveling through any other edge or vertex more than once.

Definition. A *complete* graph K_n is a graph with n vertices where for all pairs of vertices $v_i, v_j, i \neq j$ and $1 \leq i, j \leq n$, v_i and v_j are adjacent.

Definition. A *tree* is a connected graph that contains no cycles.

Definition. For any tree T, a *leaf* is a vertex with degree 1.

2 A Graph-Sharing Game and Variations

2.1 A Graph-Sharing Game

With the previous definitions in mind, we can now understand the construction and rules of a graph-sharing game, and eventually discuss additional constraints which make the game more interesting. The most basic premise of the game is as follows:

1. A connected graph serves as the game board.

2. Every vertex has a positive integer value associated with it (how much "gold" it is worth).

3. Alice and Bob take turns making moves, with Alice playing first.

4. Players make moves by selecting a vertex and collecting its gold value.

5. After a vertex is selected, it is deleted from the graph, along with associated edges.

6. The only restriction on moves at any given point is that a vertex cannot be selected if its deletion will disconnect the graph.

7. The player who secures at least half the gold wins.

The phrasing of the win condition for either Alice or Bob is important here. Essentially, a player wins anytime their opponent does not have more gold than they do. This means that in the case of an exact tie, where Alice's gold haul is equal to Bob's, we can say that both players are winners. This distinction is made only to simplify our results on various types of graphs and what constitutes a winning condition.

Now, the restriction that a player may not disconnect the game board is what introduces an element of strategy to the game. There is not a set order in which

moves must be played, but this rule does introduce the idea that certain vertices must be collected before others. Consider Figure 2 below.
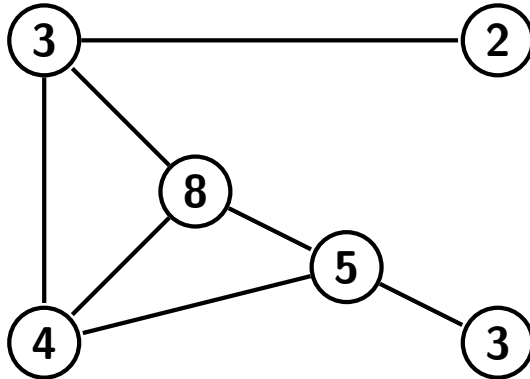
Figure 2: A Simple Game

Here, Alice's first move can be any vertex except the top-left vertex, or the vertex of value five. Taking the top-left vertex disconnects the two vertex in the top-right, and taking the five will disconnect the three in the bottom-right corner. This particular game is short enough that we could reasonably predict who will win and how. Alice can take the eight vertex on her first move and in fact, this alone secures her victory. This is because there are 25 "points" on the graph and thus only 13 are needed to win. Alice will get to collect three vertices and after collecting the eight, any two other vertices have a total sum of at least five. So on this graph, Alice wins.

2.2 Similar Games and Variations

In 2011, Piotr Micek and Bartosz Walczak published "A Graph-Grabbing Game" [4], in which they analyzed multiple variations on this game. One of the primary focuses of the paper was the parity of vertices on graph-sharing games and how they affected Alice or Bob. In particular, it was noted that, on trees, when there are an even number of vertices, Alice will have a distinct advantage regardless of gold placement, while having an odd number of vertices can favor either Alice or Bob, depending solely on the gold values. They made a fairly significant conjecture that

Alice can win on any even tree. They went on to prove this is true for at least all subdivided stars. A subdivided star is a tree with at most one vertex of degree greater than 2. It was also proven that Alice can secure at least $\frac{1}{4}$ of the gold on any even tree.

Graph-sharing games on any simple, connected graph can become very complicated, even for relatively small sizes of vertices. Depending on the density of edges, selecting any one vertex may reveal one, a few, or several more valid moves. A paper published in 2012 demonstrated that for general graphs, deciding whether Alice has a winning strategy is PSPACE-complete [2].

Most general graphs can be ambiguous as they get larger about who has an advantage and when, with at least one exception. Consider playing this game on K_n. No matter how large n gets, the outcome is predetermined. We can formalize this in a quick theorem.

Theorem 2.1. *Alice wins on any K_n.*

Proof. The proof is direct. In K_n, every vertex is a valid move at every stage. That is, Alice will never be restricted on which vertex she selects. Thus it suffices to collect the largest vertex at every turn, and since she always plays first, Bob will never be able to secure a gold amount greater than Alice. □

The K_n case is interesting because it is true regardless of whether n is even or odd. In nearly any other circumstance, parity contributes to whether Alice or Bob should expect to have an advantage, but on a complete graph, this is not true. This is because Alice's moves are never restricted. In fact, any graph where Alice's moves are not restricted will necessarily favor her. It may be of interest to look at the family of graphs of K_n with one edge removed, or perhaps a small handful of edges. On K_3, removing one edge results in a path of length three which we will see does not guarantee good outcomes for Alice. However, as n grows larger, removing a single

5

edge from a complete graph does little to affect its connectivity. There may be an interesting relation between the size of n and the number of edges which can be removed from K_n while still guaranteeing Alice can always win.

Another variation on the game has a nearly-identical rule set, except that rather than forcing the remaining subgraph to be connected after a move is made, the deleted vertices must remain connected at every step. Then, any vertex is a valid first move, and any subsequent move must be a vertex which was adjacent to a previously-deleted vertex. A paper published in 2014 showed that in this game, odd graphs which contain no subdivisions of K_n allow Alice to secure some proportion c_n of the gold [3]. However, in switching to even graphs or removing the forbidden subdivision, this proportion guarantee no longer holds and Alice's wins can tend to 0.

Some versions of either game also allow for vertices of zero value, which can force a player to waste a turn (or turns) not collecting any gold. Yet another variation involves "pizza" graphs which are simply cycles of any length, where Alice first takes a slice and then Bob chooses one of the two slices adjacent to the piece Alice took. This particular example will lead to an interesting result, as the first slice Alice takes "cuts" the pizza and then creates a path, where a valid move is only either endpoint.

We will primarily focus on the rule set given in the original definition. Furthermore, we will eventually focus our efforts on trees. The first thing we will investigate is whether Alice or Bob can identify a simple enough strategy or algorithm to maximize their winnings.

3 On Trees and Parity

3.1 Greedy Algorithms

Perhaps the simplest algorithm, but rarely the best, is a "greedy" algorithm. A greedy algorithm takes the largest valid vertex every turn without any forethought. In the first game board that we introduced, greedy grabs worked for both Alice and Bob to maximize their winnings, though Alice ultimately won. We can see from Figure 3 that such an algorithm is not generally effective:
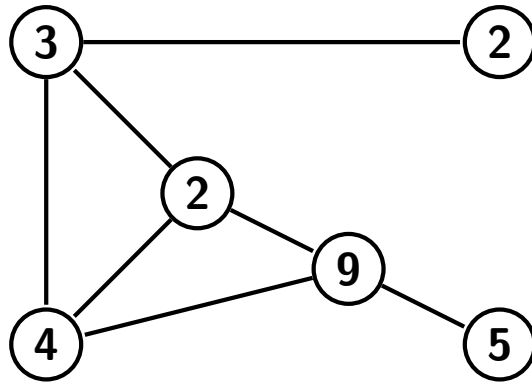
Figure 3: Greedy is not good

Here, Alice's greediest grab on her first turn will be to take the five in the bottom-right. However, taking this vertex will allow Bob to take the nine on his first turn, which will ultimately guarantee him a win. However, if Alice instead chooses the four vertex, she will win.

Note that this graph is identical in structure to Figure 2, which previously supported a greedy approach for Alice. As one can see from comparing this example, with only a few vertex values changed, the fact that being greedy worked out for Alice was mostly coincidental and is not reliable as a general strategy.

Restricting our game boards to trees allows us some certainty about each move. On a tree, only a leaf is a valid move at any given turn, and removing said leaf will either reveal precisely one new leaf or none. In this way, it might seem that Alice

can simply wisely avoid giving Bob better leaves and maximize her winnings in this way. One possible intuitive idea from here is that perhaps being blindly greedy is not so smart for Alice, but maybe with just a little forethought, her strategy can be amended until it is satisfactory. What if Alice always makes the greediest grab she can which will not reveal a vertex of greater value? Unfortunately, this also does not hold, as we see from Figure 4 below:
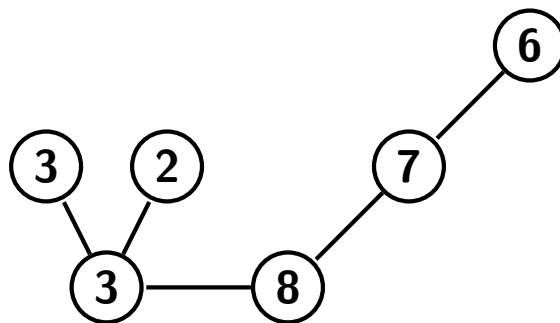

Figure 4: Careful greed is still not good

Here, if Alice follows this careful greed strategy, she will begin with the leaf of value three, as it does not reveal a vertex of greater value. This allows Bob to take the leaf of value six. Now, Alice could take the seven in response but doing so will reveal the eight, which is of greater value. So she instead takes the two, and Bob claims the seven. Now, Alice can take the eight, but this gives her $3 + 2 + 8 = 13$ and gives Bob $6 + 7 + 3 = 16$. Alice could have won on this tree by taking the six vertex. Even if Bob took the seven, Alice would have also claimed the eight and ultimately the win.

Alice's strategy here was highly dependent on the configuration of the game board itself, not some set of rules to follow which works generally. Any attempt to construct an algorithm for Alice has been met with a simple enough counterexample where the algorithm does not provide a maximal score. Despite this, Alice does seem to fare extremely well on trees with an even number of vertices. There are two chief reasons for this, both of which we will investigate.

3.2 Alice and Outlier Vertices

Intuitively, it may seem that Alice would prefer odd trees to even trees. This allows her to select one more vertex than Bob, which is one more opportunity to amass more gold. However, we will conclude that quite the opposite is true. Even trees are much better for Alice. First, let $g(v)$ be the gold value associated with a vertex v of a tree T. Now, we claim and then prove that the existence of a large outlier value of $g(v)$ always benefits Alice. This is because she can guarantee that she claims it.

Theorem 3.1. *On any even tree, Alice can select any arbitrary vertex v and guarantee she gets to claim its gold value.*

Proof. We use induction to prove this. First, if T is a tree of $n = 2$ vertices, then the result clearly holds, as Alice simply takes whichever vertex she desires. Now, suppose the result holds for any tree of $2k - 2$ vertices, and let T be a tree of $2k$ vertices. Let v be the vertex Alice wishes to claim. If v is a leaf, we are done. If v is not a leaf, Alice instead simply selects any leaf which does not reveal v. Then Bob cannot claim v on his first move, and after both players have taken their turn, Alice gets to move on a tree with $2k - 2$ vertices. Then by the inductive hypothesis, Alice claims v. $\square$

We can examine this result more directly. Consider Figure 5.
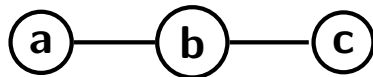

Figure 5: A path of length 3

Here, vertices a and c are the only valid moves, and taking either of these reveals b. This path is the only structure in which it is impossible to avoid revealing b. The addition of any single vertex guarantees that one can take some leaf which does not allow an opponent to claim it. Thus, whoever moves when there are an odd number

9

of vertices remaining can eventually be forced into giving up its value. On even trees, this is Bob. So if $g(b)$ is worth significantly more than other vertices, Alice can claim it and boost her score considerably. This also yields an interesting corollary.

Corollary 3.2. *Let T be an even tree with some vertex v such that $g(v) \geq g(T - v)$. Then Alice will win on T.*

Proof. This is a direct result of Theorem 3.1. If such a vertex v exists, Alice can guarantee she claims it no matter how the rest of the gold is distributed. Since she can claim $g(v)$ which is at least half the gold on T, she must win. $\square$

3.3   Alice and Structured Trees

In addition to large-value vertices, Alice also has a distinct advantage on many subtypes of trees. The simplest tree structure which we can examine is a path. In fact, we will see that Alice wins on any even path.

Theorem 3.3. *Let T be an even path. Then Alice can win on T.*

Proof. The proof is direct. Two-color the vertices, say black and white. Either the black or white vertices must contain at least half the gold. Since T is even, each endpoint will have its own distinct color. Alice should simply select the vertex whose color has at least half the gold. Then Bob can only respond with a vertex of the opposite color. Repeat this process until all vertices have been selected - Alice will collect all vertices of her chosen color and thus, at least half the gold. $\square$

Recall that it has also been proven that Alice wins on even subdivided stars [4]. This includes regular stars, where there is some core vertex v and every other vertex is a leaf that is adjacent to v. In this case, a greedy grab always works for Alice, as she can continuously grab the largest ray available and leave Bob with only a ray of less value. The core will not be accessible until the second-to-last ray is taken, which

Bob will always take due to parity. So Alice will have the option of the core if it is more valuable, or the final ray instead if that is the better move.

It seems that whenever an even tree has a simple enough structure, it favors Alice. We can demonstrate that in addition to the aforementioned subtrees, *caterpillars* also guarantee Alice a win condition. A *caterpillar* is a tree C which consists of a path, or "spine", and "legs" which are adjacent to some vertex along the spine. See Figure 6 for an example.
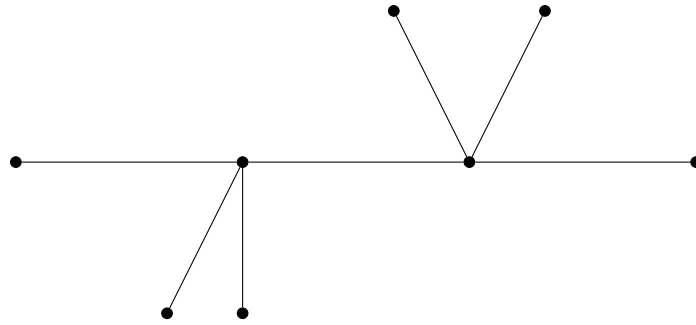
Figure 6: A caterpillar

We call the left-most vertex along the spine the *tail* and the right-most vertex along the spine the *head*. For the purposes of our definition, we say that no legs can be attached to the tail or head. The vertices can simply be oriented such that one vertex is considered to be part of the spine. For example, consider Figure 7, which is identical to the previous caterpillar.

Figure 7: The same caterpillar, with the tail rotated

This caterpillar has simply been drawn slightly differently; it is isomorphic to the previous graph. A simple path can technically also be considered a caterpillar with

11

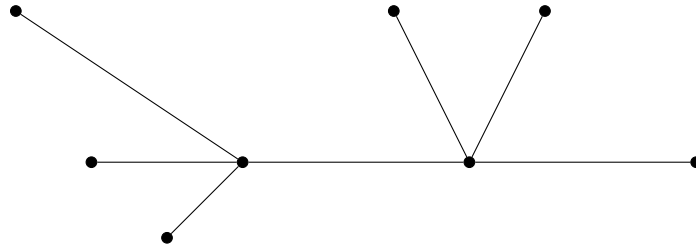no legs, so we will ensure to only refer to a tree as a caterpillar when there is at least one leg. Then, the smallest possible caterpillar has four vertices, as seen in Figure 8.
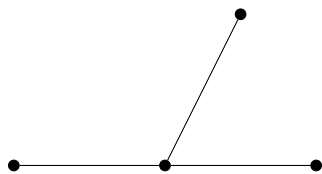

Figure 8: The smallest caterpillar

Note that this particular caterpillar is also a star, and thus Alice will win on it. We will now show that Alice wins on any even caterpillar. To do so, we make use of two lemmas.

Lemma 3.4. *Alice wins on any even caterpillar with precisely one leg.*

Proof. We prove this by induction. As previously demonstrated, this is true for a caterpillar of 4 vertices. Assume the result holds for any caterpillar of $2k - 2$ vertices, $k \geq 3$, and let C be a caterpillar of $2k$ vertices. First note that if the leg is the vertex of greatest value, Alice can simply take it and Bob can only counter with a vertex of equal or lesser value. Then by the inductive hypothesis, Alice wins.

Since there is precisely one leg, it must be that the spine has odd length. Two-color the vertices black and white. Since the spine has odd length, both the head and tail will share the same color. Without loss of generality, assume the head and tail are both white. The leg may be white or black depending on which vertex it is affixed to, but we will show its color is mostly irrelevant, as there are only two important cases: The white vertices contain at least half the gold, or the black vertices do.

Case 1: Suppose the black vertices contain at least half the gold. Then Alice should take the leg, regardless of its color. Bob can then choose only the head or tail, both of which are white, and Alice takes all black vertices, resulting in a win.

Case 2: Suppose the white vertices contain at least half the gold. Then Alice should simply take the heaviest vertex. If this is the leg, then as noted previously,

12

Alice wins. If the heaviest vertex is the head or tail, it is true that Alice may potentially reveal a vertex of greater value. However, if Bob chooses the black vertex Alice revealed, she can simply take the subsequently revealed white vertex and play along that component until he is forced to take either the leg or other vertex initially available on C. Either way, Alice wins. Since in either case, Alice does not lose, then by the inductive hypothesis, it must be that Alice wins on any even caterpillar with precisely one leg. □

We now introduce and prove one more lemma, which when combined with the previous, gives us the desired result.

Lemma 3.5. *If Alice wins on an even caterpillar, then she also wins on a caterpillar formed by adding two more legs to it.*

Proof. Suppose Alice wins on some even caterpillar, and let C be a caterpillar that has been constructed by adding two legs to it. First, if Alice's optimal move is a leg, then we are done, because if the two new legs are less heavy than the original ones, her optimal move and Bob's optimal response remain unchanged. However, if either or both of the legs are heavier, Alice's margin can only improve after grabbing a heavier leg, so after her move and Bob's response, she will still win.

Now, if Alice's optimal move was along the spine but a new optimal response for Bob was introduced, Alice can simply choose between her original move if Bob's new optimal response is still not good enough to secure victory, or she can take his new response and leave Bob only with a suboptimal move. In any case, she wins. □

Now, we show the previous two lemmas make it impossible to construct an even caterpillar on which Alice does not win.

Theorem 3.6. *Alice wins on any caterpillar with even vertices.*

Proof. We use induction and the previous two lemmas. Suppose Alice wins on any caterpillar of $2k - 2$ vertices. Let C be a caterpillar of $2k$ vertices. If C has precisely

one leg, then by Lemma 3.4, we know Alice wins. So suppose C has at least two legs. If C has two or more legs, then C can be constructed by first constructing a caterpillar of $2k - 2$ vertices and then adding two legs to it. By lemma 3.5, Alice wins on C. So by induction, Alice wins on any even caterpillar. □

Given previous results, this is not surprising. Alice does well when she can tightly control Bob's responses to her moves, and on a caterpillar, every move becomes a binary choice: take a leg, or take a vertex along the spine. Alice can easily examine Bob's options and ensure she never gives him an advantage. Theorem 3.6 also yields an interesting corollary.

Corollary 3.7. *Alice wins on any tree with precisely six vertices.*

Proof. There are only six non-isomorphic tree structures that have six vertices [1]. One is a path, and one is a star, which we now know Alice always wins on. The remaining four are listed as Figures 9 - 12 below.
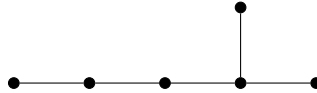
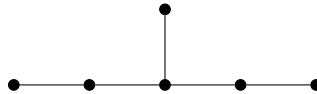Figure 9: First non-isomorphic tree of order 6

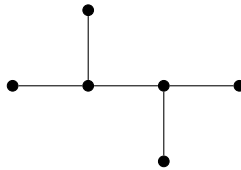Figure 10: Second non-isomorphic tree of order 6

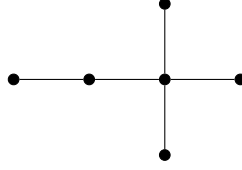Figure 11: Third non-isomorphic tree of order 6

Figure 12: Fourth non-isomorphic tree of order 6

Notice that each of these are caterpillars. Then by Theorem 3.6, Alice wins on any of these. So Alice wins on any tree of six vertices. □

One could technically make this same argument for trees of eight vertices. There are 23 non-isomorphic tree structures with eight vertices, all of which are a path, caterpillar, star, or subdivided star. However, this process can eventually become tiring. These results have thus far not shown any case in which Alice does not win on an even tree. We conjecture that this is because no such tree exists. That is, it seems to be that Alice can secure at least half the gold on any even tree.

Several efforts have been made to prove this. In 2012, Deborah E. Seacrest and Tyler Seacrest published a paper in which they finally proved the conjecture [5]. However, the proof is non-constructive. The argument used relies on the the notion of a "rooted" tree, wherein one vertex serves as the root and must be chosen last. A lemma was then stated and proved which showed that Alice's optimal score on a non-rooted tree is at least as good as her optimal score on a rooted tree. From there, the idea is that either Alice or Bob must be able to secure at least half the gold on any even tree, and Alice can choose whether to play as herself or as "Bob" if that is the better option. With this idea in mind, she can always win.

This proof unfortunately does not provide an actual strategy for Alice. Furthermore, it implies that since Alice may have to "switch" roles with Bob in some cases, there are some even trees which favor Bob if Alice doesn't make this clever swap. We will attempt to describe what such a tree must look like. We use a combination of induction and known results to investigate what structure and gold placement this

tree must have. In order to more concisely describe a series of moves and responses that Alice and Bob might make, we first introduce some new definitions.

Definition. Suppose there are two leaves v and u on some tree T such that if a player takes v, it allows their opponent to take u. Furthermore, suppose $g(u) > g(v)$. Then, call v a *sacrifice* and call u a *bounty*.

Note that in a sequence of vertices along some path component of a tree, it is possible for any given vertex to be both a sacrifice and a bounty. Using previous results, we now attempt to construct an even tree on which Alice cannot secure at least half the gold. That is, Bob can ensure his gold total is strictly greater than Alice's, assuming both players play optimally. In fact, we attempt to create the smallest such tree that exists.

First, suppose Alice wins on any even tree of $2k - 2$ vertices or fewer. As we have already seen, this is true for up to least 8 vertices [1]. Let T be a tree of $2k$ vertices, and suppose Alice does not win on T. Suppose T has n leaves, and for $1 \leq i \leq n$, let v_i be Alice's first move on T. Now, Bob may have $n - 1$ leaves to choose from as his first move, or if Alice's first move reveals a new leaf, then Bob also has n options. Either way, we can show that T must have several constraints.

Note that after Alice and Bob have both made their first turn, Alice makes a move on a tree of $2k - 2$ vertices and thus, will claim at least half of the remaining gold. Then it must be that Bob can somehow guarantee a large enough gold advantage on his first move to win on T even though Alice will win half of the remainder after their first moves. Now, we can claim the following things must all be simultaneously true of T:

1. T has 10 or more vertices.

2. The leaf of greatest value, call it v, must be a sacrifice.

3. T contains no vertex u such that $g(u) \geq g(T - u)$.

4. T has 3 or more leaves.

5. T is not a path, star, subdivided star, or caterpillar.

The justification for each is as follows:

1. Alice wins on even trees of 8 or fewer vertices.

2. If v is not a sacrifice, Alice can claim it and since it is of greatest value, Bob cannot secure any gold advantage on his first turn. Then Alice wins on the resulting tree, so she wins on T.

3. By Theorem 3.1, Alice can claim u and win if such a vertex exists.

4. If T has two leaves, it is a path of even length, which Alice always wins on.

5. Alice has been shown to always win on any tree of these structures.

We now attempt to further constrain T. Let u_i be Bob's optimal move after Alice takes v_i. Note that it's possible for $u_i = u_j$ for $i \neq j$; these are not necessarily distinct moves for Bob but simply his best response to the first leaf on T that Alice chooses. After Alice and Bob have taken v_i and u_i, call the resulting tree T_i. Then, by our assumptions, Alice will secure at least half the gold on all T_i, $1 \leq i \leq n$. Define m_i to be the margin by which Alice wins on T_i. Recall that since Alice considers tying a win, it is possible to get that $m_i = 0$. Define $m = \min\{m_i\}$. That is, m is the smallest margin by which Alice has an advantage against Bob after they have both taken their first turn.

Now, since we assume Bob wins on T no matter how well Alice plays, we can make claims about the results of various moves. Firstly, we know that for any v_i Alice takes, it must be that Bob's optimal response u_i is better by at least $m + 1$. That is, it must be that $u_i - v_i > m_i \geq m$. Let v_n be the leaf on T of greatest value. Then if Alice takes v_n, any other v_i, $1 \leq i < n$ will not secure Bob an advantage and

17

so his optimal response u_n must be the bounty revealed by Alice's sacrifice, and we have that $u_n - v_n > m_n \geq m$.

Then, for any $v_i \neq v_n$, one of two circumstances must be true:

1. v_i is also a sacrifice as above, where the bounty secures Bob a victory.

2. $v_i < v_k$ for some other leaf v_k which also serves as Bob's u_i.

Altogether, these restrictions demonstrate that it is very difficult for Bob to get even close to half the gold. If we arbitrarily let all vertices have gold value n then clearly the result will be a tie no matter how the vertices are selected, but otherwise, Alice can do a good job of preventing Bob's responses from becoming too good. The fact that at least a small handful of vertices must exist whose values differ by at least a margin of $m + 1$ also certainly make Alice's life easier. We know she can tightly control Bob's responses to her moves and while it may be possible for him to claim a vertex or two of significant value, we also know there must be more that Alice can also claim.

The difficulty is in narrowing the bounds of each of these outcomes enough that we can eventually identify where Alice must use the "player-swapping" technique to win. If T is small enough or simple enough, she can win on her own, and as T grows larger, it becomes more of a computational complexity problem than anything else.

4   Odd Trees and Conclusion

4.1   A Remark On Odd Trees

A significant portion of the results we have examined thus far has been centered on even trees. As it turns out, odd trees can be constructed to favor Alice or Bob, and the result is a factor of the gold placement, not the configuration of the tree. In fact, we can demonstrate this with a quick theorem.

Theorem 4.1. *Let T be any odd tree where only the vertices and edges have been constructed. Then, through intentional placement of the gold values, it is possible to make T a tree on which either Alice or Bob wins.*

Proof. Let T be any odd tree whose gold values have not yet been placed. To ensure Alice wins on T, let each vertex have gold value n. Then, since Alice selects one more vertex than Bob, she will end with n more gold than Bob.

Alternatively, to ensure Bob wins, let every vertex except for one non-leaf have gold value 1. For this last vertex, let it have gold value 5. Then, by Theorem 3.1, Bob will claim the 5 and win. □

While this proof is a rather simple example, it does demonstrate that odd trees cannot be said to have the same strict properties as even trees. On any even path, gold cannot be distributed in a way that Alice does not win. But on any odd path, the gold can be distributed in infinitely many ways which grant Alice or Bob a victory. This is also true of stars, subdivided stars, caterpillars, or trees in general where Alice would have a guaranteed victory if there were an even number of vertices.

We can at the least say that if the gold values are distributed randomly, without any intention of trying to help either player, that Bob will most likely win. In fact, one known result of the previous work is that Bob will secure at least half the remaining gold on any odd tree after Alice makes her first move. Thus, Alice's first move must

be significantly advantageous to make up for the remainder of the game where Bob is favored. This will generally only be when T has a leaf whose gold value is significantly higher than the remaining vertices.

It may be of interest to investigate the case of odd trees from a probability perspective instead. Since structure has been shown to not be a deciding factor, letting some amount of gold be randomly distributed among the vertices of an odd tree could yield some stricter bounds on Alice's best possible proportion or Bob's percent chance to win. Much of the existing research on these games focuses either on the "taken" variant where deleted vertices must remain connected, or general graphs, or both. In fact, it seems that when one considers the taken game, odd trees tend to have the more interesting properties and results, contrasted with the "remaining" game we examined, in which everything relates to even trees. It could then be of interest to also conduct more research on the taken game on even graphs from a probability perspective as well.

4.2   Conclusion

While there are many other interesting results that could be investigated on this topic, we have made significant progress on perhaps the most rigorous topic of even trees. We categorized several types of trees which Alice will always win on and showed that she has an advantage regardless of structure due to her control over certain vertices. We also proved that Alice can always win on any even caterpillar and found that identifying her strategy on any sufficiently large or complex even tree is significantly more complicated than one might think. In fact, attempts to construct a better proof that Alice always wins on even trees resulted only in tight constraints. Further investigation may ultimately lead to new properties which do provide a more constructive proof of the conjecture or a general strategy which will reliably work for Alice.

Bibliography

[1] *The On-Line Encyclopedia of Integer Sequences*, accessed April 4, 2018. `http://oeis.org/A000055`.

[2] J. Cibulka, J. Kyncl, V. Meszaros, R. Stolar, and P. Valtr. Graph sharing games: Complexity and connectivity. *Theoretical Computer Science*, 494:49–62, 7 2013.

[3] Adam Ggol, Piotr Micek, and Bartosz Walczak. Graph sharing game and the structure of weighted graphs with a forbidden subdivision. *Journal of Graph Theory*, 11 2014.

[4] Piotr Micek and Bartosz Walczak. A graph-grabbing game. *Combinatorics, Probability and Computing*, 20(4):623–629, 3 2011.

[5] Deborah Seacrest and Tyler Seacrest. Grabbing the gold. *Discrete Mathematics*, 312(10):1804–1806, 5 2012.