5-2013

Peg Solitaire on Trees with Diameter Four

Clayton A. Walvoort
East Tennessee State University

Peg Solitaire on Trees with Diameter Four

A thesis

presented to

the faculty of the Department of Mathematics

East Tennessee State University

In partial fulfillment

of the requirements for the degree

Master of Science in Mathematical Sciences

---

by

Clayton Walvoort

May 2013

Robert A. Beeler, Ph.D., Chair

Robert B. Gardner, Ph.D.

Anant P. Godbole, Ph.D.

Teresa W. Haynes, Ph.D.

Keywords: graph theory, peg solitaire, games on graphs, combinatorial games

ABSTRACT

Peg Solitaire on Trees with Diameter Four

by

Clayton Walvoort

In a paper by Beeler and Hoilman, the traditional game of peg solitaire is generalized to graphs in the combinatorial sense. One of the important open problems in this paper was to classify solvable trees. In this thesis, we will give necessary and sufficient conditions for the solvability for all trees with diameter four. We also give the maximum number of pegs that can be left on such a graph under the restriction that we jump whenever possible.

ACKNOWLEDGMENTS

I would like to thank my advisor, Dr. Robert Beeler, for his guidance and support through my graduate studies. I would also like to thank my finacee, Katie Smith, and my family and friends for their support. Thanks also go to Dr. Teresa Haynes, Dr. Anant Godbole, and Dr. Bob Gardner.

TABLE OF CONTENTS

ABSTRACT 2

ACKNOWLEDGMENTS 4

LIST OF TABLES 6

LIST OF FIGURES 8

1 INTRODUCTION 9

2 LITERATURE REVIEW 15

2.1 Combinatorial Games 15

2.2 Traditional Peg Solitaire 19

2.3 Peg Solitaire on Graphs 27

3 THE (n, t)-STARS 29

4 PACKAGES AND PURGES 35

5 GENERALIZATION 40

6 FOOL'S SOLITAIRE 52

7 OPEN PROBLEMS 56

BIBLIOGRAPHY 57

APPENDIX: Tables 60

VITA 62

LIST OF TABLES

1 Data for trees with order $n(T) \leq 12$. 49

2 Number of trees with order $n(T)$ that have maximum degree $\Delta(T)$. . 50

3 Terminal states of $K_{1,2}(1;2,1)$ with $S = \{x, y_{1,2}\}$. 60

4 Terminal states of $K_{1,3}(1;2,1,1)$ with $S = \{x, y_{1,2}\}$. 60

5 Terminal states of $K_{1,2}(1;1,1)$ with $S = \{x\}$. 60

6 Terminal states of $K_{1,3}(1;1,1,1)$ with $S = \{x\}$. 61

LIST OF FIGURES

1 A Typical Jump in Peg Solitaire 9

2 The $K_{1,3}$ 11

3 The Double Star - $DS(4, 3)$. 13

4 The $K_{1,3}(4; 3, 2, 2)$. 14

5 *Madame la Princesse de Soubise joüant au jeu de Solitaire* by Claude-Auguste Berey, 1697. 20

6 The English Board 21

7 The European Board 22

8 The Wiegleb Board 22

9 The Asymmetrical English Board 23

10 The Diamond Board 23

11 The Triangular Board 24

12 The 4-claw Graph 29

13 The $(3, 4)$-star Graph 31

14 The $(4, 2)$-star graph. 32

15 The Wishbone(X) Purge 35

16 The Wishbone(x) Purge 36

17 The Trident(X) Purge 36

18 The Trident(x) Purge 37

19 The Spider(x) Purge 37

20 The Spider(N) Purge 38

21 The Spider(Y) Purge 38

22 The Double Star Purge 39

1 INTRODUCTION

Peg solitaire is a table game which traditionally begins with “pegs” in every space except for one which is left empty (i.e., a “hole”). If in some row or column two adjacent pegs are next to a hole (as in Figure 1), then the peg in x can jump over the peg in y into the hole in z. In [4], peg solitaire is generalized to graphs.

A graph, $G = (V, E)$, is a set of vertices V and a set of edges E. Because of the restrictions of peg solitaire, we will assume that all graphs are finite undirected graphs with no loops or multiple edges. In particular, we will *always* assume that graphs are connected. If there are pegs in vertices x and y and a hole in z, then we allow x to *jump* over y into z, provided that $xy, yz \in E$. We will modify the notation used in [2] and a jump from x over y into z will be denoted $x \cdot \overrightarrow{y} \cdot z$.

Some basic graph theory definitions will be given. The number of vertices in a graph G is the *order* of G. The number of edges is the *size* of G. If uv is an edge of G, then u and v are *adjacent vertices*. The vertex u and the edge uv are said to be *incident* with each other. The *degree of a vertex* in v in G is the number of edges in G that are incident to v. A vertex of degree one is called a *pendant* or a *leaf*. The largest degree among the vertices of G is called the *maximum degree* of G and is

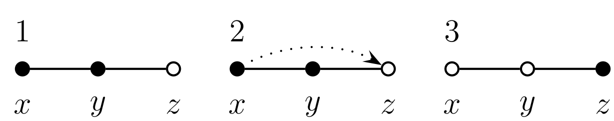

Figure 1: A Typical Jump in Peg Solitaire

denoted $\Delta(G)$.

For two vertices u and v of G, a *u-v walk*, W, in G is a sequence of vertices in G, beginning with u and ending at v such that the consecutive vertices in W are adjacent in G. A walk in a graph G in which no vertex is repeated is called a *path*. Two vertices u and v are *connected* if G contains a u-v path. The graph G itself is connected if G contains a u-v walk for every two vertices u and v of G.

The *distance* from a vertex u to a vertex v in a connected graph G is the minimum of the lengths of u-v paths in G. The *eccentricity* of a vertex v in a connected graph G is the distance between v and a vertex farthest from v in G. The *diameter* of G is the greatest eccentricity among the vertices of G [10].

For an integer $n \geq 1$, the *path* P_n is a graph of order n and size $n-1$ whose vertices can be labeled $v_1, \ldots, v_n$ and whose edges are $v_i v_{i+1}$ for $i = 1, \ldots, n-1$. For an integer $n \geq 3$, the *cycle* C_n is a graph of order n and size n whose vertices can be labeled by $v_1, \ldots, v_n$ and whose edges are $v_1 v_n$ and $v_i v_{i+1}$ for $i = 1, \ldots, n-1$. A *tree* is a connected graph that contains no cycles. A *complete graph* contains all possible edges such that every two distinct vertices are adjacent.

A graph G is *bipartite* if $V(G)$ can be partitioned into two sets U and W so that if $uw \in E(G)$, then $u \in U$ and $w \in W$. A graph is a *complete bipartite graph* if $V(G)$ can be partitioned so that uw is an edge of G if and only if $u \in U$ and $w \in W$. The complete bipartite graph is denoted $K_{s,t}$, where $|U| = s$ and $|W| = t$. The complete bipartite graph $K_{1,t}$ is called a *star* (see Figure 2).

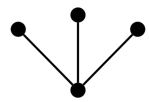

Figure 2: The $K_{1,3}$

A set of vertices U in a graph G is *independent* if no two vertices in U are adjacent. The maximum number of vertices in an independent set of vertices of G is called the *vertex independence number*, or more simply, the *independence number* of G. This is denoted $\alpha(G)$.

A graph H is a *subgraph* of a graph G if $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$. If $V(H) = V(G)$, then H is a *spanning subgraph* of G. A *spanning tree* of a graph G is a spanning subgraph of G that is a tree.

The *Cartesian product* G of two graphs G_1 and G_2 is commonly denoted by $G_1 \Box G_2$. This has vertex set $V(G) = V(G_1) \times V(G_2)$. Two distinct vertices (u, v) and (x, y) of $G_1 \Box G_2$ are adjacent if either $u = x$ and $vy \in E(G_2)$ or $v = y$ and $ux \in E(G_1)$. A convenient way to draw $G_1 \Box G_2$ is to first place a copy of G_2 at each vertex of G_1 and then join corresponding vertices of G_2 in those copies of G_2 placed at adjacent vertices of G_1.

Two graphs G and H are *isomorphic* if there exists a bijective function $\phi : V(G) \to V(H)$ such that two vertices u and v are adjacent in G if and only if $\phi(u)$ and $\phi(v)$ are adjacent in H. The function ϕ is called an *isomorphism* from G to H. An *auto-*

morphism of a graph G is an isomorphism from G to itself. Thus, an automorphism of G is a permutation of $V(G)$ that preserves adjacency and non-adjacency.

The following definitions are specific to playing peg solitaire on graphs. A *terminal state* $T \subset V$ is a set of vertices that have pegs at the end of the game. A terminal state T is associated with *starting state* S if T can be obtained from S by a series of jumps. Unless otherwise noted, we will assume that S consists of a single vertex. A graph G is *solvable* if there exists some vertex s so that, starting with a hole in s, there exists an associated terminal state consisting of a single peg. A graph G is *freely solvable* if for all vertices s so that, starting with a hole in s, there exists an associated terminal state consisting of a single peg. It is not always possible to solve a graph. A graph G is *k-solvable* if there exists some vertex s so that, starting with a hole in s, there exists an associated terminal state consisting of k nonadjacent pegs. In particular, a graph is *distance 2-solvable* if there exists some vertex s so that, starting with a hole in s, there exists an associated terminal state consisting of two pegs that are distance 2 apart. The preceding definitions are from [4]. For more information on traditional peg solitaire, refer to [2, 8, 9, 12].

In [4, 5], the solvability of several families of graphs was determined. One of the more important open problems in [4] was to classify the solvability of trees. We note that any connected graph has a spanning tree [10]. Also, if G is a spanning subgraph of H and G is k-solvable, then H is at worst k-solvable [4]. Thus, an important step in determining which graphs are solvable is to determine what trees are solvable.

However, there exists solvable graphs that do not have a solvable tree as a spanning subgraph. In this thesis, we seek to expand these results by considering trees of a fixed diameter. We note that the only tree of diameter one is the P_2, which is trivially freely solvable [4]. The trees of diameter two are precisely the stars with n arms. The solvability of stars is given in the following proposition.

Proposition 1.1 [4] *The star $K_{1,n}$ is $(n-1)$-solvable.*

The trees of diameter three are precisely the double stars. The *double star* consists of two adjacent vertices u_ℓ and u_r. The vertex u_ℓ is adjacent to L pendant vertices denoted $\ell_1, \ldots, \ell_L$. Similarly, u_r is adjacent to R pendant vertices denoted $r_1, \ldots, r_R$. Without loss of generality, assume that $L \geq R$. The double star with parameters L and R is denoted $DS(L, R)$ (see Figure 3). The solvability of double stars is given below.

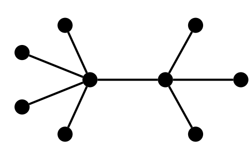

Figure 3: The Double Star - $DS(4, 3)$.

Proposition 1.2 [5] *The double star $DS(L, R)$ is freely solvable iff $L = R$ and $R \neq 1$; $DS(L, R)$ is solvable iff $L \leq R + 1$; $DS(L, R)$ is distance 2-solvable iff $L = R + 2$; $DS(L, R)$ is $(L - R)$-solvable in all other cases.*

As shown in Proposition 1.1 and Proposition 1.2, the solvability of trees of diameter three or less has been established. Therefore, the natural next step in this classification is to determine the solvability of trees of diameter four. We note that any tree of diameter four can be obtained by appending pendant vertices to the existing vertices of $K_{1,n}$. Label the center of the star as x and its arms as $y_1, \dots, y_n$. Suppose that we append c pendant vertices to x, namely $x_1, \dots, x_c$. We also append a_i pendant vertices to y_i, namely $y_{i,1}, \dots, y_{i,a_i}$ for $i = 1, \dots, n$. We denote the resulting graph as $K_{1,n}(c; a_1, \dots, a_n)$. An example is shown in Figure 4. For convenience of notation, we will denote the set of vertices $X = \{x_1, \dots, x_c\}$ and $Y_i = \{y_{i,1}, \dots, y_{i,a_i}\}$ for $i = 1, \dots, n$. Such sets of vertices will be called *clusters* and the original vertices of the $K_{1,n}$ will be called *support vertices* with $N = \{y_i, \dots, y_n\}$. We will assume, without loss of generality, that $a_1 \geq \ldots \geq a_n \geq 1$ and denote $\sum_{i=1}^{n} a_i = s$. This ensures that each tree of diameter four has a unique parametrization under this notation.

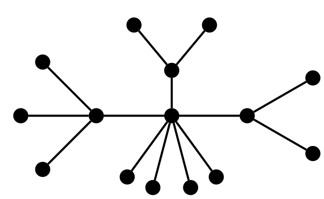

Figure 4: The $K_{1,3}(4; 3, 2, 2)$.

2 LITERATURE REVIEW

In this chapter, we will give a brief overview of combinatorial games. Then, the traditional game of Peg Solitaire will be described, as well as several of its variations. Lastly, some motivation from [2] and [9] will be used to describe methods that will be helpful in playing the game.

2.1 Combinatorial Games

From [14], the main question is what are combinatorial games? Roughly speaking, the family of *combinatorial games* consists of the two-player games with perfect information (no hidden information as in some card games), no chance moves (no dice), and outcome restricted to (lose, win), (tie, tie), (draw, draw) for the two players who move alternately. *Tie* is an end position such as in tic-tac-toe, where no player wins. However, a *draw* is any position from which both players have non-losing moves, but neither can force a win. Both the easy game of NIM and the seemingly difficult game of chess are examples of combinatorial games.

Games are intriguing for several reasons. There are applications or connections to various disciplines, such as logic, graph theory, online algorithms, and biology. Fraenkel [14] also gives other reasons for why games are intriguing:

> Perhaps the urge to play games is rooted in our primal beastly instincts; the desire to corner, torture, or at least dominate our peers. A common

> expression of these vile cravings is found in the passions roused by local, national, and international tournaments. An intellectually refined version of these dark desires, well hidden beneath the facade of scientific research, is the consuming drive "to beat them all," to be more clever than the most clever.... Reaching this goal is particularly satisfying and sweet in the context of combinatorial games, in view of their inherent high complexity. (Fraenkel 2)

With a slant towards artificial intelligence, Pearl wrote in [20] that games:

> ... offer a perfect laboratory for studying complex problem solving methodologies.... in areas such as business, government, scientific, legal, and others.... Last, but not least, games possess addictive entertaining qualities of a very general appeal. That helps maintain a steady influx of research talents into the field and renders games a convenient media for communicating powerful ideas about general methods of strategic planning. (Pearl 221)

Fraenkel classifies games in two ways: playgames and mathgames. *Playgames* can be thought of as games that can be bought at the store. Playgames are challenging enough that people will purchase them and play them (such as peg solitaire). *Mathgames* are games that mathematicians study. Mathgames are challenging for mathematicians or other scientists to play with and ponder about, but not necessarily to "the man on the street." Mathgames may take considerable time to solve, which may be unappealing to non-mathematicians. As more games are studied by mathematicians, the overlap between playgames and mathgames is growing larger.

As an introduction to combinatorial games, we will first examine the game of NIM. NIM is an excellent game to start with because it has simple rules, one player must win, and it is easy to model mathematically. The general game of NIM is played by two players and consists of at least three piles of stones. The players alternate turns removing any number of stones from a single pile. The winner is the player who removes the last remaining stone or stones. Alternate versions of the game can be played such that the loser is the player who removes the last remaining stone or stones. The winning strategy, as described in [13], is for one to write the number of stones in each pile as a binary digit. Then, the place values of the binary digits are added modulo 2 in each column, producing another binary digit. The digits are not carried to the next place value. If the sum is zero after player 1 moves, then it is impossible for player 2 to win with the next move. This is because player 2 must remove a stone, hence making the sum non-zero. Therefore, the winning strategy is for the sum of the piles to be zero after every move you make.

In [1], a different game is played on a graph, called *Cops and Robbers*. This game involves two players, one being the *Cop* C and the other being the *Robber* R. The game begins by first C and then R occupying distinct vertices of a finite connected undirected graph G. Then, C and R alternate moving along the edges of G to other vertices. The objective of the game is for C to occupy the same vertex as R, essentially "catching" R. If C catches R, then C wins. However, if R can prevent C from ever catching him, then R wins. The family of graphs where C has a winning strategy are denoted by $\mathcal{C}$. Likewise, $\mathcal{R}$ denotes the family of graphs in which R has a winning strategy. Some families of graphs have been classified. For example, trees are in $\mathcal{C}$, while cycles and regular non-complete graphs are in $\mathcal{R}$. In [1], many more families of graphs are classified in great detail.

There are several variations of the game. For example, there can be multiple cops with a single robber, a single cop with multiple robbers, or multiple cops with multiple robbers. There are other versions as well. In the *active* version, the robber must move whenever it is his turn. In the *passive* version, the robber may choose to remain stationary. However, it is assumed the cop always moves in both versions. In [1], the passive version is considered because it is more realistic. It is also noted that changing the characteristics of a game, such as the number of cops or robbers, or from active to passive, can change whether the graph is in $\mathcal{C}$ or $\mathcal{R}$.

There are too many combinatorial games to list, but a large compilation of works can be found in [14]. There are many different types of combinatorial games, which is why they appeal to so many individuals. As more complex problems are discovered daily, the number of combinatorial games will continue to grow.

2.2 Traditional Peg Solitaire

The origins of peg solitaire are not known for certain. However, John Beasley and John Maltby have done extensive research trying to determine the genealogy of the game [2]. The most common legend is that it was created by a French nobleman while he was incarcerated in the Bastille during the seventeenth century [2]. This legend would explain one of the game's less common names, *solo noble*. However, there is no hard evidence to support this myth.

Beasley also gives other possible origins of the game in [2] such as American Indians playing the game with their arrows after returning from a hunt, or that it was a German nun's game. Some even suggest the game has roots in China, Chaldaea, or ancient Rome. However, the earliest known evidence is the engraving *Madame la Princesse de Soubise joüant au jeu de Solitaire* by Claude-Auguste Berey, which is dated 1697. The picture is of Anne de Rohan-Chabot, Princess of Soubise, who is seated with the game by her side as shown in Figure 5. It is also a legend that the game was invented by Pelisson, a French mathematician, to entertain Louis XIV [23].

 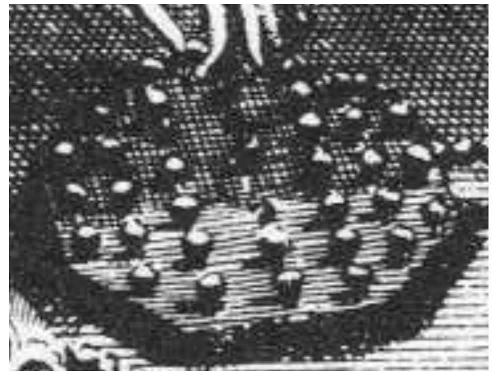

Figure 5: *Madame la Princesse de Soubise joüant au jeu de Solitaire* by Claude-Auguste Berey, 1697.

The earliest written description of the game is given in a paper by Leibniz in 1710, which is quoted in [2] as follows:

> Not so very long ago there became widespread an excellent kind of game, called *Solitaire*, where I play on my own, but as with a friend as witness and referee to see that I play correctly. A board is filled with stones set in holes, which are removed in turn, but none (except the first, which may be chosen for removal at will) can be removed unless you are able to jump another stone across it into an adjacent empty place, when it is captured as in Draughts. He who removes all the stones right to the end according

to this rule, wins; but he who is compelled to leave more than one stone still on the board, yields the palm. (Beasley xii)

Despite evidence documenting the game to the late 17th century, both Beasley and Maltby [2] agree that similar games were played much earlier due to the simplicity of the rules.

There are several variations of boards on which the game can be played. The most common is called the English board and is shown in Figure 6. The board is made of wood and has recessed impressions, called holes, which are filled with glass marbles or small stones, called pegs or "men." The English board has thirty-three holes which are arranged symmetrically in the shape of a cross.

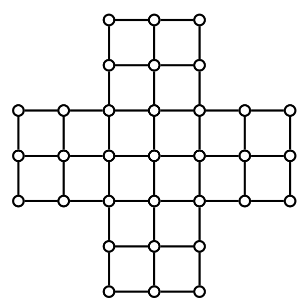

Figure 6: The English Board

Another variation of the game is called the European or the Continental board, as shown in Figure 7. This board is made by adding four additional holes to the English board, for a total of thirty-seven holes. These additional hole are located at the "inner corners" of the English board, giving the board a perceived circular shape.

This is the board the Princess of Soubise is playing in Berey's engraving (see Figure 5).

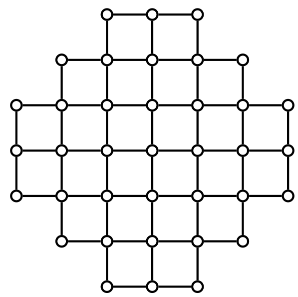

Figure 7: The European Board

There is also the board described by J. C. Wiegleb from [24], which has forty-five holes and is a larger version of the symmetrical English board. This is illustrated in Figure 8.

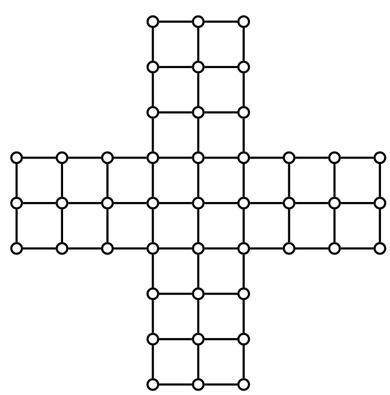

Figure 8: The Wiegleb Board

The asymmetrical English board, Figure 9, was given by George Bell [24] in the twentieth century and has thirty-nine holes, with two adjacent sides of the cross being longer than the other two.

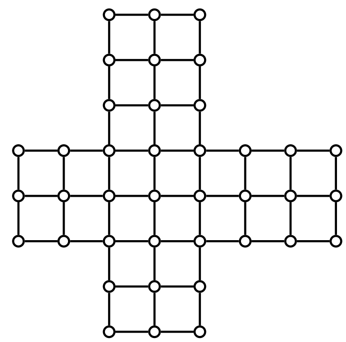

Figure 9: The Asymmetrical English Board

The Diamond board, Figure 10, has forty-one holes and is made by adding four more holes to the European board, located at the tips of the sides of the original English board.

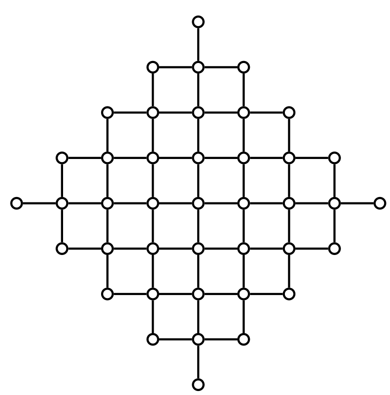

Figure 10: The Diamond Board

There is also the Triangular board, Figure 11, which consists of fifteen holes arranged in a triangular shape. This board is commonly found on the dining tables of the restaurant Cracker Barrel and is referred to as the "peg game." Also, in this configuration, diagonal jumps are allowed, unlike the previous boards where only vertical and horizontal moves are allowed.

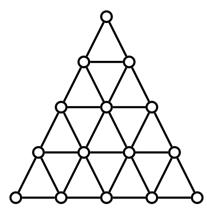

Figure 11: The Triangular Board

There are obviously many variations of boards on which peg solitaire can be played. Only a few of the most commonly used boards have been listed above from [24]. In [2], Beasley goes into great detail on how to solve some of the boards. Beasley even explores playing the game on a three dimensional board.

Another work that is certainly worth noting is *Winning Ways for your Mathematical Plays*, by Berlekamp, Conway, and Guy [9]. This publication deals mainly with playing the game on traditional boards. In this work, the major idea is the use of purges to solve the boards instead of using "brute force" methods to play the game. A *package* is a collection of vertices which satisfy a specific configuration of pegs and holes such that a predetermined sequence of jumps will preserve the locations of certain pegs and holes and remove the remaining pegs. When a package is used to remove pegs, it is called a *purge*. The pegs and holes which are restored to their original locations are called the *catalyst*. This idea of purges will prove to be very useful in this thesis.

From [11], Bruijn explains in detail the relationship of the English board and a finite field. This paper also gives another variation of the game, demanding that the remaining peg should end in the hole in the center of the board. They also define four elements, as well as addition and multiplication operators, such that the pegs and holes can be thought of as elements of a finite field. The pegs and holes are also given coordinates in the ordinary Cartesian plane, which is easily done on the English board. It is also mentions that the shape of the board plays no essential role in the consideration of each hole's given coordinates and the game can even be considered in more than two dimensions. Operations are then performed on the elements which are analogous to making the moves on the board. The location of the final peg can be found by this method.

In [17], Hentzel uses the same approach as Bruijn [11], except Hentzel plays the game on the triangular board. Also, Hentzel has to only define a commutative group and its addition operator, instead of a finite field with two operations. Hentzel then defines the *parity* of the game, which does not change throughout the game. This is used to classify some games as being not solvable without making any jumps. However, this does not necessarily classify a game as being solvable. Hentzel's method can be used on any hexagonal type board, or hexagonal array.

In [18], playing the game on the English board is approached from the artificial intelligence point of view. Variables are assigned to represent the pegs and holes, then operations are performed to simulate jumps being made and pegs being removed.

Also in [18], the variation of the game known as fool's solitaire is explored. The method of finding the optimal number of pegs that can remain at the end of the game is done by an exhaustive computer search. The downside to this method is that it can take a very long time to calculate every possible terminal state. One way that was mentioned to reduce the number of terminal states that need to be checked is by only checking unique terminal states. Since the board is symmetrical, one terminal state can be thought of as four terminal states simply by rotating the board. Another way to ensure that an optimal solution has been found is to seek a terminal state after performing only one move. If that cannot be done, then attempt finding a terminal state after performing only two moves, and so on. This variation will also be discussed in detail in this thesis.

Another interesting variation is noted in [18], called Long-hop solitaire. It is called a *hop* when the same peg is moved consecutively, as if playing checkers. This is another optimization problem by attempting to perform as few hops as possible.

2.3 Peg Solitaire on Graphs

In [4], Beeler and Hoilman generalize the game to arbitrary boards which are treated as graphs in the combinatorial sense. This transition from a physical board to an arbitrarily drawn board now allows us to contemplate the idea of playing all possible games of peg solitaire. The transition also allows for 'L'-shaped jumps, which are not allowed by the traditional rules.

In [4], Beeler and Hoilman give necessary and sufficient conditions for the solvability of several well-known families of graphs. Some of these families include stars, paths, cycles, complete graphs, and complete bipartite graphs. They also show that the Cartesian product of two solvable graphs is also solvable. In [5], Beeler and Hoilman extend this idea by finding the necessary and sufficient conditions for solvability of the windmill and double star graphs.

A new way of finding solvable graphs is given by Beeler, Gray, and Hoilman in [3]. This approach begins with a single peg and a single hole. Pegs and holes are then added in a manner that the game is being played "backwards" in such a way that the product is guaranteed to be solvable when the game is reversed.

Then, in [7], Beeler and Rodriguez explore the variation of the game in which the player wants to leave the maximum number of pegs that can remain at the end of the game, under the restriction that a jump is made whenever possible. This variation is called fool's solitaire and begins to answer an open problem in [4]. In [7], Beeler and Rodriguez find an upper bound for the fool's solitaire number of a graph. However, they only conjecture a lower bound. Another key idea presented in [7] is the *dual* of a configuration. The dual of a configuration is obtained by reversing the roles of pegs and holes in a configuration. The dual of a configuration will be very important in this thesis. Therefore, a more proper definition will be given.

The *dual* of a peg configuration T, denoted T', is the state resulting from reversing the roles of pegs and holes [7]. The relationship between the dual of a configuration and whether it is a valid terminal state is given below.

Proposition 2.1 [4, 7] *Suppose that S is a starting state of G with associated terminal state T. Let S' and T' be the duals of S and T, respectively. It follows that T' is a starting state of G with associated terminal state S'.*

This is only a small sample of the works that have been done pertaining to peg solitaire. Many of the publications primarily deal the traditional boards, as Beeler and Hoilman only began work on arbitrary boards in 2010. However, as research continues into arbitrary boards, more will be known about the solvability of these graphs. In the future, it may be known what percentage of graphs are solvable and equivalently, what percentage are not solvable.

3 THE (n, t)-STARS

Before considering the most general case, we first restrict our attention to several special cases. The star $K_{1,3}$ is sometimes referred to as a *claw* [10] and is also a complete bipartite graph. The claw is commonly represented as shown in Figure 2.

Since $n = 3$ with the claw, the claw is at best 2-solvable by Proposition 1.1. The *n-claw* graph is defined to be a graph that contains n claws that all share a common vertex as shown in Figure 12. This graph is isomorphic to $K_{1,n}(0; 2, \ldots, 2)$.

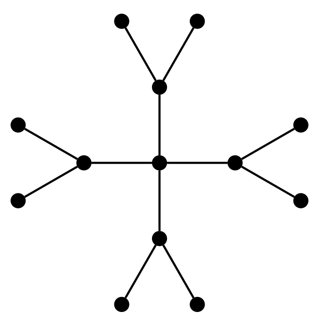

Figure 12: The 4-claw Graph

Proposition 3.1 *The n-claw graph is $(n + 1)$-solvable.*

Proof. For the n-claw to be solved, the leaves, $y_{i,j}$, must be removed. Notice that for $y_{i,j}$ to be removed, the move $y_{i,j}\cdot\overrightarrow{y_i}\cdot x$ must be made. This removes y_i from the game. Every support vertex can remove only one leaf. However, there are $s = 2n$ leaves and n of them will be eliminated by a support vertex. Since the final jump will put a peg in x, there will be $2n - n + 1$ pegs left and no moves are available. Therefore, the n-claw graph is $(n + 1)$-solvable.

To show sufficiency, begin with the initial hole in x and move $y_{i,1}\cdot\overrightarrow{y_i}\cdot x$, $y_{i+1}\cdot\overrightarrow{x}\cdot y_i$, $y_{n-i+1,1}\cdot\overrightarrow{y_{n-i+1}}\cdot x$, and $y_i\cdot\overrightarrow{x}\cdot y_{i+1}$ for $1 \leq i \leq z = \lceil\frac{n-2}{2}\rceil$. If n is odd, there is only one cluster, Y_{z+1}, that has pegs in both leaves and a peg in its support vertex. So make the final move $y_{z+1,1}\cdot\overrightarrow{y_{z+1}}\cdot x$. These moves will remove one peg from every cluster, as well as remove the pegs from every anchor. Since there is a peg remaining in x and one in each cluster, the n-claw is $(n+1)$-solvable when n is odd and the initial hole in x. If n is even, there will be two clusters that have pegs in both leaves and a peg in the respective support vertices, the clusters Y_{z+1} and Y_{z+2}. Now move $y_{z+1,1}\cdot\overrightarrow{y_{z+1}}\cdot x$, $y_{z+2}\cdot\overrightarrow{x}\cdot y_{z+1}$, and $y_{z+1,2}\cdot\overrightarrow{y_{z+1}}\cdot x$. This results in all pegs being removed from the cluster Y_{z+1}, but two pegs being in the cluster Y_{z+2}, as well as removing the remaining support vertices. Since there is a peg in x, the n-claw is also $(n+1)$-solvable when n is even and the initial hole in x.

Now begin with the initial hole in Y_i, say $y_{1,1}$. Notice that moving $y_{1,2}\cdot\overrightarrow{y_1}\cdot y_{1,1}$ yields the same configuration as if the initial hole where in x and making the first move as described above. Therefore, make the move $x\cdot\overrightarrow{y_1}\cdot y_{1,1}$ instead. Notice that this removes a peg from a support vertex and adds a peg to a leaf. Since pegs in support vertices are used to remove the pegs in the leaves, making this move is the exact opposite of what we are trying to accomplish and will result in leaving additional pegs when there are no moves possible. Hence, the previous move $y_{1,2}\cdot\overrightarrow{y_1}\cdot y_{1,1}$, must be made. Therefore, the n-claw is $(n+1)$-solvable when the initial hole in Y.

Suppose the initial hole is in N, say y_1. The move $y_2\cdot\overrightarrow{x}\cdot y_1$ is forced, up to automorphism on the vertices. This produces the same configuration as if the initial hole is in Y and the move $x \cdot \overrightarrow{y_1} \cdot y_{1,1}$ was made. As described above, this is not beneficial. Hence, there will be at least $n+1$ pegs remaining when no more moves are available. Therefore, the n-claw is $(n+1)$-solvable. ■

We will now generalize Proposition 3.1 to graphs with more pendant edges. First, define the *t-star* to be the graph consisting of t leaves joined to a center vertex , as defined in [19]. Notice the t-star notation is simply referring to stars with t arms, also denoted $K_{1,t}$. Therefore, the claw in Figure 2 is also known as a 3-star. Now we will define the (n,t)*-star* to be a collection of n t-stars that share one common vertex as shown in Figure 13. Note that the (n,t)-star is isomorphic to $K_{1,n}(0; t-1, \ldots, t-1)$.

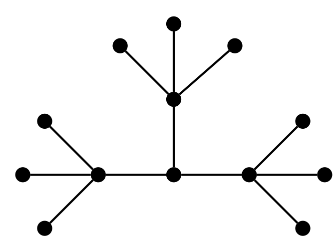

Figure 13: The $(3,4)$-star Graph

Proposition 3.2 *The* (n,t)*-star graph is* $(nt-2n+1)$*-solvable.*

Proof. Using the same techniques used in Proposition 3.1, at most n leaves can be removed. Since $s = n(t-1)$, there will be $n(t-1) - n + 1$ pegs left when no further moves can be made. Hence, the (n,t)-star graph is $(nt-2n+1)$-solvable.

Notice that the same techniques used in Proposition 3.1 can be used to show sufficiency as well. However, there will now be $t-2$ pegs remaining in every cluster, as well as a peg in x. Hence, $n(t-2)+1$ pegs will remain when no more moves are available. Therefore, the (n,t)-star graph is at best $(nt-2n+1)$-solvable. ■

Notice that we can derive Proposition 3.1 from Proposition 3.2 by letting $t=3$. Let us now consider (n,t)-stars when $t=2$, as shown in Figure 14. These graphs are also known as *subdivided star* graphs [10]. We will refer to the combination of a support vertex and a single pendant as a leg. Using Proposition 3.2, when $t=2$, the $(n,2)$-star graph should be solvable.

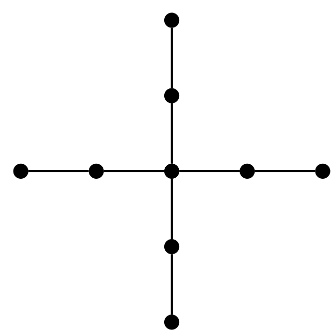

Figure 14: The $(4,2)$-star graph.

Proposition 3.3 *The $(n,2)$-star is solvable if and only if the initial hole is in the center vertex and $n \neq 2$. Otherwise, the $(n,2)$-star is distance 2-solvable.*

Proof. To show it is necessary that the initial hole must be in the center vertex for the $(n,2)$-star to be solvable, first assume the initial hole is not in the center vertex.

Suppose the initial hole is in a support vertex, say y_1. Up to automorphism on the vertices, the moves $y_2 \cdot \overrightarrow{x} \cdot y_1$ and $y_{1,1} \cdot \overrightarrow{y_1} \cdot x$ are forced, which clears the first leg. By ignoring the empty leg, we now have an $(n-1, 2)$-star with the initial hole in y_2. Since we began the game with the initial hole in a support vertex, it is not beneficial to jump into a support vertex that has an empty leaf. Doing so creates two leaves that have empty support vertices, which is not the objective. Therefore, the legs are removed one at a time until the $(2, 2)$-star remains. Since the $(2, 2)$-star is isomorphic to the P_5, the $(n, 2)$-star with the initial hole in a support vertex is distance 2-solvable by [4].

Now suppose the initial hole is in $y_{1,1}$. The moves $x \cdot \overrightarrow{y_1} \cdot y_{1,1}$ and $y_{2,1} \cdot \overrightarrow{y_2} \cdot x$ are forced by symmetry and clears the second leg. By ignoring the empty leg, this is identical to placing the initial hole in y_1, which is distance 2-solvable by above.

For sufficiency, let the initial hole be in x. Up to automorphism, the moves $y_{n,1} \cdot \overrightarrow{y_n} \cdot x$, $y_{n-1} \cdot \overrightarrow{x} \cdot y_n$, and $y_{1,1} \cdot \overrightarrow{y_1} \cdot x$ are forced. Let $n \geq 4$. If n is even, then let $n = 2k, k \in \mathbb{N}$. Similarly, if n is odd, then let $n = 2k + 1$. Now, for $\ell = 1, \ldots, k - 1$, move $y_{2\ell} \cdot \overrightarrow{x} \cdot y_1$, $y_{2\ell+1,1} \cdot \overrightarrow{y_{2\ell+1}} \cdot x$, $y_1 \cdot \overrightarrow{x} \cdot y_{2\ell}$, and $y_{2\ell,1} \cdot \overrightarrow{y_{2\ell}} \cdot x$. Next, if $n = 2k$, then moving $y_{n-2} \cdot \overrightarrow{x} \cdot y_{n-1}$, $y_{n-1,1} \cdot \overrightarrow{y_{n-1}} \cdot x$, $y_n \cdot \overrightarrow{x} \cdot y_{n-2}$, and $y_{n-2,1} \cdot \overrightarrow{y_{n-2}} \cdot x$ places the final peg in x. However, if $n = 2k + 1$, then moving $y_n \cdot \overrightarrow{x} \cdot y_{n-1}$ and $y_{n-1,1} \cdot \overrightarrow{y_{n-1}} \cdot x$ positions the final peg in x.

If $n = 3$, then make the first set of moves and the last set of moves since it is not necessary to remove pairs of legs. This results with the final peg in x. If $n = 2$, then the $(n, 2)$-star is isomorphic to P_5, which is distance 2-solvable [4]. However, if

$n = 1$, then the $(n, 2)$-star is isomorphic to P_3, which is solvable with the initial hole in x or $y_{1,1}$ [4]. However, by automorphism, $x \cong y_{1,1}$ for this arrangement, so it can be assumed that the initial hole is always in x. ■

Also, note that the final peg of a solvable $(n, 2)$-star is always in x. Taking note of where the final peg is located, as well as the initial hole, will prove to be useful later in this paper.

In an effort to streamline the next result, we will now define four useful packages along with the associated purges.

The *wishbone(X)* package will consist of a $K_{1,2}(1;1,1)$ with a hole in x_1. The wishbone(X) purge will be $y_1 \cdot \overrightarrow{x} \cdot x_1$, $y_{2,1} \cdot \overrightarrow{y_2} \cdot x$, $x_1 \cdot \overrightarrow{x} \cdot y_1$, and $y_{1,1} \cdot \overrightarrow{y_1} \cdot x$. This results in the removal of pegs from y_1, $y_{1,1}$, y_2, and $y_{2,1}$. The catalyst is x and x_1. The wishbone(X) purge is shown in Figure 15.

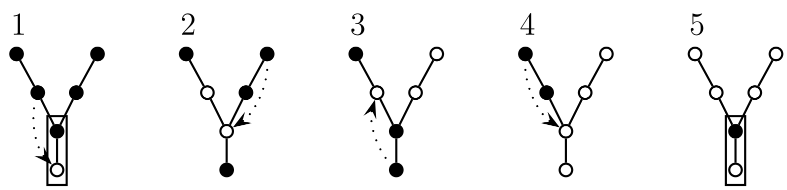

Figure 15: The Wishbone(X) Purge

We will also define the *wishbone(x)* package, with the hole in x. The associated purge will be $y_{2,1} \cdot \overrightarrow{y_2} \cdot x$, $x_1 \cdot \overrightarrow{x} \cdot y_2$, $y_{1,1} \cdot \overrightarrow{y_1} \cdot x$, and $y_2 \cdot \overrightarrow{x} \cdot x_1$. Again, this removes y_1, $y_{1,1}$, y_2, and $y_{2,1}$, while the catalyst is x and x_1. The wishbone(x) purge is shown in Figure 16. Note that the wishbone(X) and wishbone(x) purge both remove two legs. Also notice that the letter in parenthesis denotes the location of the hole in the catalyst.

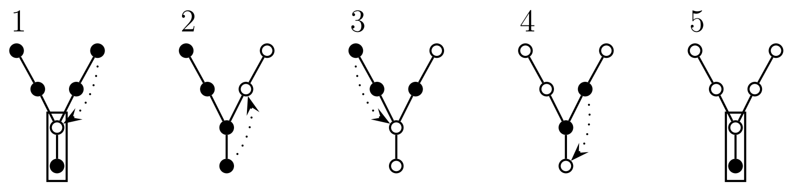

Figure 16: The Wishbone(x) Purge

The *trident(X)* package will consist of a $K_{1,3}(1;1,1,1)$ with a hole in x_1. The trident(X) purge will be $y_3 \cdot \overrightarrow{x} \cdot x_1$, $y_{2,1} \cdot \overrightarrow{y_2} \cdot x$, $y_1 \cdot \overrightarrow{x} \cdot y_3$, $y_{3,1} \cdot \overrightarrow{y_3} \cdot x$, $x_1 \cdot \overrightarrow{x} \cdot y_1$, and $y_{1,1} \cdot \overrightarrow{y_1} \cdot x$. The result is the removal of pegs from y_1, $y_{1,1}$, y_2, $y_{2,1}$, y_3, and $y_{3,1}$, while x and x_1 are the catalyst. The trident(X) purge is shown in Figure 17.

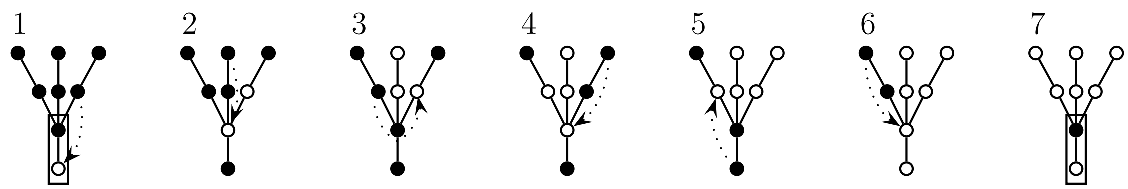

Figure 17: The Trident(X) Purge

We will also define the *trident(x)* package, with the hole in x. The associated purge will be $y_{3,1} \cdot \overrightarrow{y_3} \cdot x$, $x_1 \cdot \overrightarrow{x} \cdot y_3$, $y_{2,1} \cdot \overrightarrow{y_2} \cdot x$, $y_3 \cdot \overrightarrow{x} \cdot y_2$, $y_{1,1} \cdot \overrightarrow{y_1} \cdot x$, and $y_2 \cdot \overrightarrow{x} \cdot x_1$. Again, this removes y_1, $y_{1,1}$, y_2, $y_{2,1}$, y_3, and $y_{3,1}$, while the catalyst is x and x_1. The trident(x) purge is shown in Figure 18. Note that the trident(X) and trident(x) purge both remove three legs.

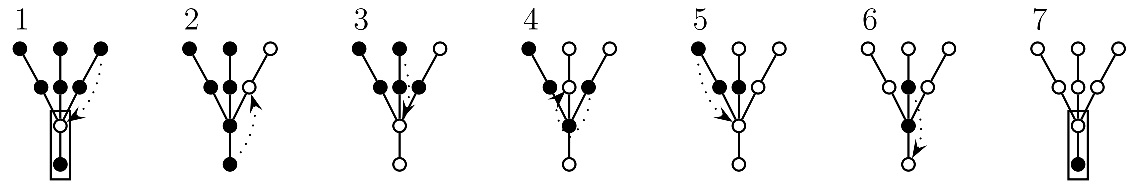

Figure 18: The Trident(x) Purge

The *spider(x)* package will consist of a $K_{1,3}(2;1,1,1)$ with the hole in x. The spider(x) purge will be $y_{1,1}\cdot\overrightarrow{y_1}\cdot x$, $x_1\cdot\overrightarrow{x}\cdot y_1$, $y_{2,2}\cdot\overrightarrow{y_2}\cdot x$, $x\cdot\overrightarrow{y_1}\cdot y_{1,1}$, $y_{3,1}\cdot\overrightarrow{y_3}\cdot x$, and $x_2\cdot\overrightarrow{x}\cdot y_1$. The results is the removal of pegs from x_1, x_2, y_2, $y_{2,1}$, y_3, and $y_{3,1}$, while x, y_1 and $y_{1,1}$ are the catalyst. The spider(x) purge is shown in Figure 19.

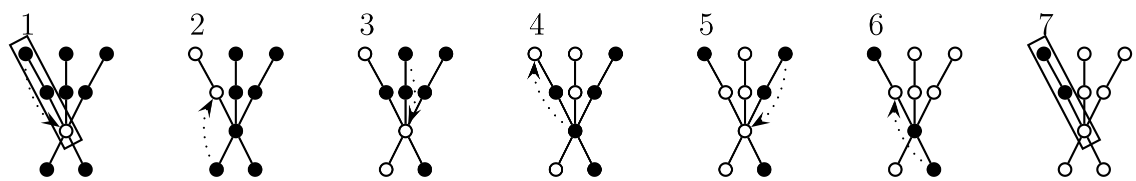

Figure 19: The Spider(x) Purge

Define the *spider(N)* package, with a hole in y_1. The associated purge will be $x_1\cdot\overrightarrow{x}\cdot y_1$, $y_{1,1}\cdot\overrightarrow{y_1}\cdot x$, $x_2\cdot\overrightarrow{x}\cdot y_1$, $y_{2,1}\cdot\overrightarrow{y_2}\cdot x$, $x\cdot\overrightarrow{y_1}\cdot y_{1,1}$, and $y_{3,1}\cdot\overrightarrow{y_3}\cdot x$. Again, this removes x_1, x_2, y_2, $y_{2,1}$, y_3, and $y_{3,1}$, while x, y_1 and $y_{1,1}$ are the catalyst. The spider(N) purge is shown in Figure 20.

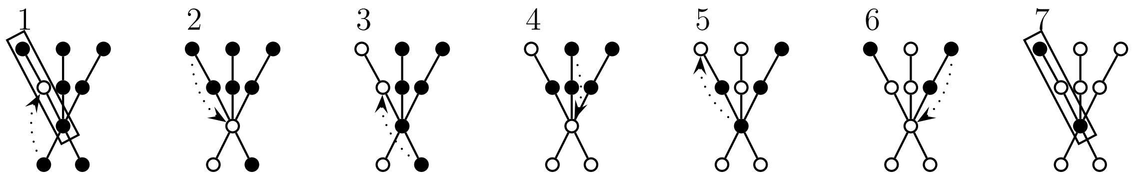

Figure 20: The Spider(N) Purge

We will also define the *spider(Y)* package, with the hole in $y_{1,1}$. The associated purge will be $x\cdot\overrightarrow{y_1}\cdot y_{1,1}$, $y_{2,1}\cdot\overrightarrow{y_2}\cdot x$, $x_1\cdot\overrightarrow{x}\cdot y_1$, $y_{1,1}\cdot\overrightarrow{y_1}\cdot x$, $x_2\cdot\overrightarrow{x}\cdot y_1$, and $y_{3,1}\cdot\overrightarrow{y_3}\cdot x$. Again, this removes x_1, x_2, y_2, $y_{2,1}$, y_3, and $y_{3,1}$, while x, y_1 and $y_{1,1}$ are the catalyst. The spider(Y) purge is shown in Figure 21. Note that the spider(x), spider(N), and spider(Y) purges remove two pegs from X and remove both of the pegs from two legs.

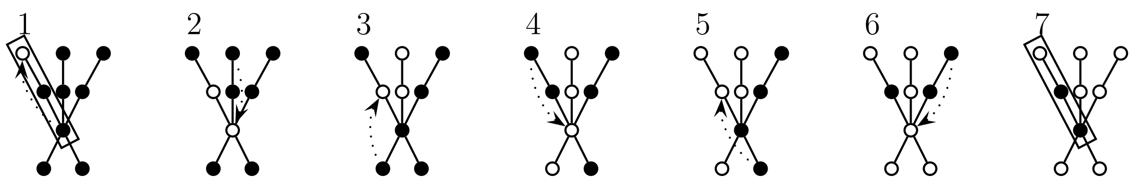

Figure 21: The Spider(Y) Purge

We will also borrow the *2-purge* from [9]. This is a $K_{1,1}(1;1)$ where $y_{1,1}$ and x_1 are removed while x and y_1 are the catalyst. Notice that the 2-purge is the motivation behind solving double stars. Therefore, we will refer to this as the *double star* purge. The double star purge is shown in Figure 22. It will be useful to define a notation for a repeated 2-purge among automorphic vertices. Denote the double star purge as

$\mathcal{DS}(A, B, d)$, where A is the cluster with a peg in its support vertex, B is the cluster with a hole in its support vertex and d is the number of pegs to be removed from each cluster.

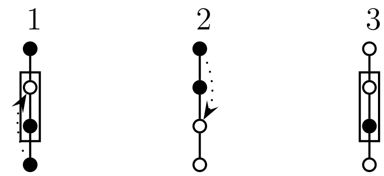

Figure 22: The Double Star Purge

5 GENERALIZATION

We will now proceed with our main result. Namely, we will provide the necessary and sufficient conditions for the solvability of all trees of diameter four. The strategy for solving $K_{1,n}(c; a_1, \ldots, a_n)$ will be to begin by performing double star purges. Therefore, we introduce a new parameter, $k = c - s + n$, where $s = \sum_{i=1}^{n} a_i$. This gives the number of pegs remaining in X after $a_i - 1$ pegs have been removed from Y_i for $1 \leq i \leq n$ using double star purges. We begin with the case where at least one of the $a_i \geq 2$.

Theorem 5.1 *The conditions for solvability of $K_{1,n}(c; a_1, ..., a_n)$ where $a_1 \geq 2$ are as follows:*

(i) The graph $K_{1,n}(c; a_1, \ldots, a_n)$ is solvable iff $0 \leq k \leq n + 1$.

(ii) The graph $K_{1,n}(c; a_1, \ldots, a_n)$ is freely solvable iff $1 \leq k \leq n$.

(iii) The graph $K_{1,n}(c; a_1, \ldots, a_n)$ is distance 2-solvable iff $k \in \{-1, n + 2\}$.

(iv) The graph $K_{1,n}(c; a_1, \ldots, a_n)$ is $(1 - k)$-solvable if $k \leq -1$. The graph $K_{1,n}(c; a_1, \ldots, a_n)$ is $(k - n)$-solvable if $k \geq n + 2$.

Proof. To show necessity, let $k < 0$. In this case, $c - s + n < 0$. It is necessary to remove all pegs in Y_i. To remove a peg from Y_i, there must first be a peg in y_i. The only moves that accomplish this are $x_p \cdot \overrightarrow{x} \cdot y_i$ and $y_j \cdot \overrightarrow{x} \cdot y_i$ for $1 \leq p \leq c$ and

$j \neq i$. Therefore, the double star purges $\mathcal{DS}(Y_i, X, d)$ and $\mathcal{DS}(Y_i, N, d)$ are necessary to remove pegs in Y_i. Notice this is analogous to $K_{1,1}(s; n+c)$, but $s \geq n+c+1$. Therefore, the initial hole must be in y_i [4]. However, all of the elements of N are in the other side of the double star. Thus, this is not solvable. Since the double star purges are necessary and the initial hole is in N, at least $s-n-c$ pegs will remain in Y_i after the double star purges [4]. Also, there was a peg in x at the beginning of the game and there will be a peg in x when the last move of the final double star purge is made. Therefore, this peg must be added as well, meaning there will be at least $s-n-c+1 = 1-k$, pegs remaining when $k < 0$. Thus the graph is at best $(1-k)$-solvable.

Now let $k \geq n+2$ which implies $s \leq c-2$. Similar to above, $\mathcal{DS}(X, Y_i, d)$ is the only way to remove pegs from X. Thus, we leave at least $c-s = k-n$ pegs. Hence the graph is at best $(k-n)$-solvable.

To show sufficiency, we simply give an algorithm to solve $K_{1,n}(c; a_1, \ldots, a_n)$ for $0 \leq k \leq n+1$. We begin with the hole in x and perform $\mathcal{DS}(Y_{n-i+1}, X, a_{n-i+1}-1)$ for $i = 1, ..., n$. Without loss of generality, the last peg in Y_i is in $y_{i,1}$. This will remove $s-n$ pegs from X.

If $k=0$ and $n=2$, then begin by performing one less double star purge such that the "extra pegs" are in $y_{1,2}$ and x_c. Now move $y_{2,1} \cdot \overrightarrow{y_2} \cdot x$, $x_c \cdot \overrightarrow{x} \cdot y_2$, $y_{1,2} \cdot \overrightarrow{y_1} \cdot x$, $y_2 \cdot \overrightarrow{x} \cdot y_1$, and $y_{1,1} \cdot \overrightarrow{y_1} \cdot x$ to solve with the final peg in x. If $k=0$ and $n \geq 3$, then eliminate the remaining legs by using a combination of wishbone and trident purges with x and y_1

as the catalyst. Finally, move $y_{1,1} \cdot \overrightarrow{y_1} \cdot x$ to solve. It will later be shown that the case where $k = 0$ is not freely solvable.

If $k = 1$ and $n = 2$ or $n = 3$, then double star purges will reduce the graph to $K_{1,2}(1; 2, 1)$ or $K_{1,3}(1; 2, 1, 1)$, respectively, with holes in x and $y_{1,2}$. If $n \geq 4$, then use wishbone and trident purges to reduce the graph to $K_{1,2}(1; 2, 1)$ with holes in x and $y_{1,2}$. It can be checked using [6] that both configurations can be solved with the final peg located in any vertex. Ergo, $K_{1,n}(c; a_1, \ldots, a_n)$ is freely solvable when $k = 1$ by Proposition 2.1.

If $2 \leq k \leq n - 1$ and k is odd, then perform the double star purges as above. Next, perform the spider(x) purge $\frac{k-1}{2}$ times using x, y_1 and $y_{1,1}$ as the catalyst. The graph has been reduced to the case of $k = 1$ with holes in x and $y_{1,2}$. As shown above, the final peg can now be located anywhere. Hence, the graph is freely solvable by Proposition 2.1. If k is even, then begin the game with the initial hole in y_1, and make the move $x_c \cdot \overrightarrow{x} \cdot y_1$. Ignoring x_c reduces this to the case when k is odd, with the initial hole in x. Thus $K_{1,n}(c; a_1, \ldots, a_n)$ is freely solvable when $2 \leq k \leq n - 1$.

If $k = n$, then instead begin with the initial hole in y_1. Move $x_c \cdot \overrightarrow{x} \cdot y_1$ and ignore x_c. This reduces to the case of $k = n - 1$ with the initial hole in x, which can have the final peg in any location. Thus $K_{1,n}(c; a_1, \ldots, a_n)$ is freely solvable when $k = n$.

If $k = n + 1$, then instead begin with the initial hole in y_1 and move $x_c \cdot \overrightarrow{x} \cdot y_1$. Ignoring the hole in x_c, this reduces this to the case when $k = n$ and the initial hole is in x, which we know to be solvable. It will later be shown that the case where

$k = n + 1$ is not freely solvable.

Throughout the proof, it has been shown that the conditions given for $K_{1,n}(c; a_1, \ldots, a_n)$ to be freely solvable are sufficient. We now show that these conditions are necessary. For $K_{1,n}(c; a_1, \ldots, a_n)$ and $k = 0$, consider $K_{1,2}(1; 2, 1)$, which is not solvable if the initial hole is in X or N. If $k = n + 1$, consider $K_{1,2}(4; 2, 1)$, which is not solvable if the initial hole is in x or Y_i. These can be verified using an exhaustive computer search [6]. We will now show that any $K_{1,n}(c; a_1, \ldots, a_n)$ where $a_1 \geq 2$ and $k = 0$ or $k = n + 1$ reduces to $K_{1,2}(1; 2, 1)$ or $K_{1,2}(4; 2, 1)$, respectively.

Note that when a double star purge is performed, k is not changed. This is because a peg is taken from each of X and Y_i, thus reducing c and s by one. So we can append one "extra" vertex to X and one "extra" vertex to one Y_i and k will not change. Also, $\mathcal{DS}(Y_1, X, 1)$ will remove the recently added pegs. Ignoring these now empty vertices will result in obtaining the original $K_{1,n}(c; a_1, \ldots, a_n)$. The double star purges are necessary, as shown earlier. Thus, as many as desired of these "extra" vertices can be added in pairs and the new tree will reduce to the original. We can append the set of vertices $\{x_{c+1}, y_{n+1}, y_{n+1,1}, y_{n+1,2}\}$ without changing k. Further, $y_{n+1,1} \cdot \overrightarrow{y_{n+1}} \cdot x$, $x_c \cdot \overrightarrow{x} \cdot y_{n+1}$, $y_{n+1,2} \cdot \overrightarrow{y_{n+1}} \cdot x$, and $x_{c+1} \cdot \overrightarrow{x} \cdot x_c$ will remove the newly added vertices. This sequence of moves is analogous to a double star purge, which we have argued is necessary. By using combinations of these two "addition" methods, any diameter four tree with $k = 0$ or $k = n + 1$ can be constructed from $K_{1,2}(1; 2, 1)$ and $K_{1,2}(4; 2, 1)$, respectively. Therefore, all such trees must reduce to either $K_{1,2}(1; 2, 1)$

and $K_{1,2}(4; 2, 1)$, which are not freely solvable.

If the above algorithm is used on a diameter four tree with $k \geq n + 2$, then the remaining $k - n$ pegs will be in X. In particular, if $k = n + 2$, then this results in two pegs that are distance 2 apart. Hence, $K_{1,n}(c; a_1, \ldots, a_n)$ is distance 2-solvable when $k = n + 2$.

If $k \leq -1$, then a different technique is required. Again, we begin with the hole in x and use the double star purges as described above. After the purges are performed, there are no pegs in X, there are $s - c$ pegs left in the Y_i, and n pegs remaining in the support vertices. We now remove the remaining support vertices using a combination of wishbone and trident purges with x and y_1 as the catalyst. This removes an additional $2n - 2$ pegs. After the final move of $y_{1,1} \cdot \overrightarrow{y_1} \cdot x$, there are $s - c + n - 1 - (2n - 2) = 1 - k$ pegs remaining. Further, we have $-k$ pegs in Y_i and one peg in x. In particular, if $k = -1$, then we have two pegs that are distance 2 apart. Hence, $K_{1,n}(c; a_1, \ldots, a_n)$ is distance 2-solvable when $k = -1$. ■

We now deal with the case when all of the $a_i = 1$. We note that in this case, $k = c \geq 0$. For this reason, we give our conditions in terms of c.

Theorem 5.2 *The conditions for solvability of $K_{1,n}(c; 1, ..., 1)$ are as follows:*

(i) The graph $K_{1,2t}(c; 1, \ldots, 1)$ is solvable iff $0 \leq c \leq 2t$ and $(t, c) \neq (1, 0)$. The graph $K_{1,2t+1}(c; 1, \ldots, 1)$ is solvable iff $0 \leq c \leq 2t + 2$.

(ii) The graph $K_{1,n}(c; 1, \ldots, 1)$ is freely solvable iff $1 \leq c \leq n - 1$.

(iii) The graph $K_{1,2t}(c;1,\ldots,1)$ *is distance 2-solvable iff* $c = 2t+1$ *or* $(t,c) = (1,0)$. *The graph* $K_{1,2t+1}(c;1,\ldots,1)$ *is distance 2-solvable iff* $c = 2t+3$.

(iv) The graph $K_{1,2t}(c;1,\ldots,1)$ *is* $(c-2t+1)$*-solvable if* $c \geq 2t+1$. *The graph* $K_{1,2t+1}(c;1,\ldots,1)$ *is* $(c-2t-1)$*-solvable if* $c \geq 2t+3$.

Proof. Using the argument from Theorem 5.1, $K_{1,n}(c;1,...,1)$ is at best $(c-n)$-solvable when $c \geq n+2$. If $n = 2$ and $c = 0$, then the graph is the path on five vertices, which is distance 2-solvable [4]. The additional necessary conditions will be discussed later.

If $c = 0$ and $n \geq 3$, then begin with the initial hole in x. Remove $n-1$ legs using a combination of wishbone and trident purges, as in Theorem 5.1. Finally, use $y_{1,1} \cdot \overrightarrow{y_1} \cdot x$ to solve the graph.

If $1 \leq c \leq n-1$ and c is odd, then start with the initial hole in x. Perform $\frac{c-1}{2}$ spider(x) purges, then use wishbone and trident purges to reduce the graph to $K_{1,2}(1;1,1)$ or $K_{1,3}(1;1,1,1)$ with the hole in x. It can be checked using [6] that $K_{1,2}(1;1,1)$ and $K_{1,3}(1;1,1,1)$ with the hole in x can have the final peg located anywhere except x. Thus, $K_{1,n}(c;1,\ldots,1)$ is freely solvable when $1 \leq c \leq n-1$ and c is odd by Proposition 2.1.

If $1 \leq c \leq n-1$ and c is even, then start with the initial hole in y_1, and make the move $x_c \cdot \overrightarrow{x} \cdot y_1$. Ignoring x_c reduces this to the case when c is odd with the initial hole in x. We have shown this case can have the final peg anywhere except x. If the initial hole is in x, then use spider(x) purges to reduce to the case where $c = 0$. Therefore, $K_{1,n}(c; 1, \ldots, 1)$ is freely solvable when $1 \leq c \leq n - 1$ and c is even.

If $c = n$, then let the initial hole be in y_1 and use the spider(N) purge $\lceil \frac{c}{2} \rceil - 1$ times. Ignoring the vertices cleared by the spider(N) purges reduces this to $K_{1,1}(1; 1)$ or $K_{1,2}(2; 1, 1)$, depending on whether n is odd or even, respectively. Note that both $K_{1,1}(1; 1)$ and $K_{1,2}(2; 1, 1)$ have the hole in y_1. For $K_{1,1}(1; 1)$, move $x_1 \cdot \overrightarrow{x} \cdot y_1$ and $y_{1,1} \cdot \overrightarrow{y_1} \cdot x$ to solve. For $K_{1,2}(2; 1, 1)$, move $y_2 \cdot \overrightarrow{x} \cdot y_1$, $y_{1,1} \cdot \overrightarrow{y_1} \cdot x$, $x_2 \cdot \overrightarrow{x} \cdot y_2$, $y_{2,1} \cdot \overrightarrow{y_2} \cdot x$, and $x_1 \cdot \overrightarrow{x} \cdot y_1$ to solve.

If $c = n + 1$ and n is odd, then start with the initial hole in y_1 and perform the spider(N) purge $\frac{n-1}{2}$ times. This reduces to $K_{1,1}(2; 1)$ with the hole in y_1, which is a solvable double star with the final peg in y_1. However, when n is even, spider(N) purges will reduce the game down to $K_{1,2}(3; 1, 1)$, which is distance 2-solvable. We will show that $K_{1,2t}(2t+1; 1, ..., 1)$ will reduce to $K_{1,2}(3; 1, 1)$ because spider(N) purges are necessary. Begin with the initial hole in y_{2t}. If we make the initial jump $y_1 \cdot \overrightarrow{x} \cdot y_{2t}$, then $y_{2t,1} \cdot \overrightarrow{y_{2t}} \cdot x$ is forced. If we think of $y_{2t} \in X$, then this is essentially $K_{1,2t-1}(2t+1; 1, ..., 1)$ with a hole in y_1, which is unsolvable. Thus, up to automorphism, the first two moves are $x_1 \cdot \overrightarrow{x} \cdot y_1$ followed by $y_{1,1} \cdot \overrightarrow{y_1} \cdot x$. As before, the move $y_2 \cdot \overrightarrow{x} \cdot y_1$ will lead to an unsolvable graph. Therefore, $x_2 \cdot \overrightarrow{x} \cdot y_1$ and $y_{2,1} \cdot \overrightarrow{y_2} \cdot x$ are forced. Again, $y_1 \cdot \overrightarrow{x} \cdot y_2$ will lead to an unsolvable graph. Thus $x \cdot \overrightarrow{y_1} \cdot y_{1,1}$ and $y_{3,1} \cdot \overrightarrow{y_3} \cdot x$ are forced. This concludes the exact moves of the spider(N) purge. Since the spider purges are necessary, if the initial hole is not in y_1, then the graph will reduce to a case that is not solvable by a similar argument.

It has been shown that the conditions provided for $K_{1,2t}(c; 1, \ldots, 1)$ and $K_{1,2t+1}(c; 1, \ldots, 1)$ to be freely solvable are sufficient. We now show that these conditions are necessary. For $K_{1,n}(c; 1, \ldots, 1)$ with $c = 0$, assume the initial hole is in y_1. Up to automorphism, the moves $y_2 \cdot \overrightarrow{x} \cdot y_1$ and $y_{1,1} \cdot \overrightarrow{y_1} \cdot x$ are forced, which clears the first leg. Therefore, the legs must be removed one at a time until the path on five vertices remains, which is distance 2-solvable [4].

If $n = 2t$ and $c = n$, then we have shown that spider(N) or spider(x) purges are necessary. Hence, up to automorphism, the initial hole must be in y_1, x, or X to be solvable. If $n = 2t+1$ and $c = n+1$, then spider(N) purges are necessary. Therefore, up to automorphism on the vertices, the initial hole must be in y_1 or X.

Consider $K_{1,2t+1}(c; 1, ..., 1)$, where $c \geq 2t+3$. If the hole is in y_1, then t spider(N) purges reduce the graph to $K_{1,1}(c - 2t; 1)$ with the hole in y_1. After the moves $x_{c-2t} \cdot \overrightarrow{x} \cdot y_1$, $y_{1,1} \cdot \overrightarrow{y_1} \cdot x$, and $x_{c-2t-1} \cdot \overrightarrow{x} \cdot y_1$, there are $c - 2t - 2$ pegs in X and 1 peg in y_1. In particular, if $c = 2t + 3$, then the graph is distance 2-solvable.

Similarly, for $K_{1,2t}(c; 1, ..., 1)$, where $c \geq 2t + 1$, the $t - 1$ spider(N) purges will reduce the graph to $K_{1,2}(c - 2t + 2; 1, 1)$ with a hole in y_1. After making the jumps $x_{c-2t+2} \cdot \overrightarrow{x} \cdot y_1$, $y_{2,1} \cdot \overrightarrow{y_2} \cdot x$, $x_{c-2t+1} \cdot \overrightarrow{x} \cdot y_2$, $y_{1,1} \cdot \overrightarrow{y_1} \cdot x$, and $y_2 \cdot \overrightarrow{x} \cdot x_{c-2t+1}$, there are $c - 2t + 1$

pegs in X. In particular, if $c = 2t + 1$, then the graph is distance 2-solvable. ■

Knowing the necessary and sufficient conditions for the solvability of diameter four trees also leads to the following result that applies to all trees.

Theorem 5.3 *Let T be a tree with maximum degree $\Delta(T)$, $n(T)$ vertices, and c leaves adjacent to the vertex of maximum degree. If $\Delta(T) \geq n(T) - c + 1$, then T is not solvable.*

Proof. As shown in [4, 5], the above bound holds for trees of diameter three or less. Hence, we may assume that T is a tree with a diameter of at least four. Choose x to be a vertex of maximum degree. Denote the cluster of pendants adjacent to x as X such that $|X| = c$. In order for T to be solvable, every peg must be removed from X. Therefore, the upper bound for solvability of diameter four trees, $k \leq n + 1$, which implies $c \leq s + 1$, can be used to determine when T is not solvable. If $c > s + 1$, then c is too large and thus the terminal state will have multiple pegs remaining in X.

In a tree of diameter four, there are $s + n$ vertices remaining when x and X are excluded. In the general case, there are $n(T) - c - 1$ vertices remaining. Since T is being treated as a diameter four tree, $s + n = n(T) - c - 1$. The upper bound for diameter four trees can be now be manipulated to become $s + 1 = n(T) - c - n$.

Also, $n = \Delta(T) - c$. Thus, $s + 1 = n(T) - \Delta(T)$. Hence, if $c > s + 1$, then $c > n(T) - \Delta(T)$. This implies $\Delta(T) > n(T) - c$. Therefore, $\Delta(T) \geq n(T) - c + 1$

implies that T is not solvable. ■

Notice that there are solvable trees with $\Delta(T) = n(T) - c$. One example is $DS(c+1, c)$. Therefore, the above bound is sharp.

Using this new bound, we can now find the minimum number of trees with order $n(T)$ that are not solvable. Using the On-Line Encyclopedia of Integer Sequences [21], Table 1 gives the number of non-isomorphic trees with order $n(T) \leq 12$. Table 1 also gives the number of trees with $n(T) \leq 12$ that are not solvable just by analyzing the largest degree vertex of a tree. Note that this is may not be the total number of trees that are not solvable for a given $n(T)$.

$n(T)$	# of trees	# not solvable	%
1	1	0	0
2	1	0	0
3	1	0	0
4	2	1	50
5	3	1	33.33
6	6	2	33.33
7	11	2	18.18
8	23	4	17.39
9	47	5	10.64
10	106	9	8.49
11	235	11	4.68
12	551	21	3.81

Table 1: Data for trees with order $n(T) \leq 12$.

Notice that the percentage of trees that can be classified as being not solvable by Theorem 5.3 decreases as $n(T)$ grows larger. Hence, an assumption could be that a greater percentage of trees are solvable as $n(T)$ increases. This evidence would help support a conjecture made by Beeler, Gray, and Hoilman in [3]. The conjecture states, for $n \geq 9$, at least half of all non-isomorphic trees of order n are solvable.

Another interesting pattern was found while interpreting the data. Table 2 gives the number of trees with order $n(T)$ that have maximum degree $\Delta(T)$.

$\Delta(T)$ \ $n(T)$	1	2	3	4	5	6	7	8	9	10	11	12
0	1	0	0	0	0	0	0	0	0	0	0	0
1		1	0	0	0	0	0	0	0	0	0	0
2			1	1	1	1	1	1	1	1	1	1
3				1	1	3	5	10	17	36	65	134
4					1	1	3	7	17	38	93	220
5						1	1	3	7	19	45	118
6							1	1	3	7	19	47
7								1	1	3	7	19
8									1	1	3	7
9										1	1	3
10											1	1
11												1

Table 2: Number of trees with order $n(T)$ that have maximum degree $\Delta(T)$.

The pattern along the main diagonal simply shows that there is only one star, $K_{1,n}$ with n vertices. However, other patterns develop for larger n. Notice the bottom four entries in the last five columns are 7, 3, 1, 1 when read from top to bottom. It also appears that 19 joins this pattern as well, beginning with $n(T) = 10$. It seems that when first number above the repeating sequence increases by two, the resulting number becomes part of the repeating sequence. Searching [21] results in three sequences, [22, 16, 15], all containing the sequence 1, 1, 3, 7, 19, 47. However, all three differ have different elements appearing next. Analyzing trees with order $n(T) = 13$, will either narrow the list down to one sequence, or eliminate all three.

Also, if some pattern can be found which identifies how many trees with maximum degree $\Delta(T)$ have a specific number of pendants c, Table 1 can be extended without the need for examining each tree individually.

6 FOOL'S SOLITAIRE

Fool's solitaire is a variation of peg solitaire where the goal is to have the maximum number of pegs possible remaining at the end of the game under the caveat that the player jumps whenever possible. The *fool's solitaire number* of a graph G, denoted $Fs(G)$, is the cardinality of the largest terminal state T that is associated with a starting state consisting of a single hole. Similarly, a terminal state T is a *fool's solitaire solution* if the cardinality of T is equal to $Fs(G)$ [7]. The fool's solitaire number for stars is $Fs(K_{1,n}) = n$ [7] and double stars is $Fs(K_{1,1}(c; a_1)) = c + a_1$ [5].

In [7], it is conjectured that for all connected graphs G, $Fs(G) \geq \alpha(G) - 1$. However, it can be checked using [6] that $K_{1,3}(0; 2, 2, 2)$ violates this conjecture, because $Fs(K_{1,3}(0; 2, 2, 2)) = 5 = \alpha(K_{1,3}(0; 2, 2, 2)) - 2$. This example is far from unique. In fact, the diameter four trees provide an infinite class of counterexamples to the above conjecture. For this reason, we are motivated to find $Fs(K_{1,n}(c; a_1, \ldots, a_n))$.

We begin with some observations about the maximum independent set A for $G = K_{1,n}(c; a_1, \ldots, a_n)$. First note that A will contain each of the Y_i where $a_i \geq 2$. If $c = 0$, then $x \in A$. If $c = 1$, then either x or x_1 will be in A. In this case, we choose that $x_1 \in A$ for the purpose of the fool's solitaire problem. If $c \geq 2$, then $X \subset A$. If $a_i = 1$ and $c = 0$, then $y_{i,1} \in A$, but $y_i \notin A$. However, if $a_i = 1$ and $c \geq 1$, then we have a choice whether to include $y_{i,1}$ or y_i into A. In any case, $\alpha(G) = s + c + 1$ when $c = 0$ and $\alpha(G) = s + c$ when $c \geq 1$. These cases will be instrumental in proving the following theorem.

Theorem 6.1 *Consider the diameter four tree $G = K_{1,n}(c; a_1, \ldots, a_n)$, where $a_i \geq 2$ for $1 \leq i \leq n - \ell$ and $a_i = 1$ for $n - \ell + 1 \leq i \leq n$.*

(i) If $c = 0$ and $\ell = 0$, then $Fs(G) = s + c - \left\lfloor \frac{n}{3} \right\rfloor$.

(ii) If $c \geq 1$ and $\ell = 0$, then $Fs(G) = s + c - \left\lfloor \frac{n+1}{3} \right\rfloor$.

(iii) If $\ell \geq 1$, then $Fs(G) = s + c - \left\lfloor \frac{n-2m+1}{3} \right\rfloor$, where $m = \min\{\ell, \left\lfloor \frac{n}{2} \right\rfloor\}$.

Proof. First, consider the case where $c = 0$ and $\ell = 0$. As noted above, the maximum independent set is $T = Y_1 \cup \cdots \cup Y_n \cup \{x\}$. The dual of this configuration is $T' = \{y_1, ..., y_n\}$. This has $n \geq 2$ pegs, none of which are adjacent. Hence, we can not obtain the upper bound of $\alpha(G) = s + c$ pegs. Thus, some pegs must be removed from the maximum independent set to obtain the fool's solitaire solution. Equivalently, some pegs must be added to the dual of the maximum independent set in order to obtain a solvable configuration. We will determine the minimum number of pegs that need to be added to the dual.

Up to automorphism, there are two places where we can add a peg to the dual, namely to x or to one of the Y_i. If we add a peg to x, then we can remove one peg from N with the move $x \cdot \overrightarrow{y_1} \cdot y_{1,1}$. Hence, this will not solve the dual. Adding an additional peg to Y_2 will remove an additional three pegs from N using the moves $y_{2,1} \cdot \overrightarrow{y_2} \cdot x$, $y_3 \cdot \overrightarrow{x} \cdot y_1$, $y_{1,1} \cdot \overrightarrow{y_1} \cdot x$, and $x \cdot \overrightarrow{y_4} \cdot y_{4,1}$. However, if a peg is added to Y_1 rather than x, then we can remove two pegs using the moves $y_{1,1}\cdot\overrightarrow{y_1}\cdot x$ and $x\cdot\overrightarrow{y_2}\cdot y_{2,1}$. Adding an additional peg to Y_3 will remove an additional three pegs using a similar sequence as above. Thus, it is more efficient to add a peg to one of the Y_i than it is to add a peg to x. Similarly, adding two or more pegs to a single Y_i is not advantageous, as this would deny us to ability to jump its corresponding support vertex.

Thus, if $n = 3t + r$, where $t, r \in \mathbb{Z}$ and $0 \leq r \leq 2$, then we must add at least $t+1$ pegs to the dual. Equivalently, the fool's solitaire number is at most $s + c - t$, where $t = \left\lfloor \frac{n}{3} \right\rfloor$. To show equality, it is sufficient to provide the dual of the fool's solitaire solution and the sequence of moves that will reduce this to a single peg. We claim that $T' = \{y_1, ..., y_n, y_{1,1}, y_{3i,1} : i \leq t\}$. Begin with the moves $y_{1,1}\cdot\overrightarrow{y_1}\cdot x$ and $x\cdot\overrightarrow{y_2}\cdot y_{2,1}$. For $i = 1, ..., t-1$, make the sequence of jumps $y_{3i,1}\cdot\overrightarrow{y_{3i}}\cdot x$, $y_{3i+1}\cdot\overrightarrow{x}\cdot y_{3i-1}$, $y_{3i-1,1}\cdot\overrightarrow{y_{3i-1}}\cdot x$, and $x\cdot\overrightarrow{y_{3i+2}}\cdot y_{3i+2,1}$. If $r = 0$, then we replace $x\cdot\overrightarrow{y_{3t-1}}\cdot y_{3t-1,1}$ with $y_{3t-1}\cdot\overrightarrow{x}\cdot y_{3t-2}$ and make the additional jumps $y_{3t,1}\cdot\overrightarrow{y_{3t}}\cdot x$ and $y_{3t-2}\cdot\overrightarrow{x}\cdot y_{3t}$. If $r = 1$, then we make the additional jumps $y_{3t,1}\cdot\overrightarrow{y_{3t}}\cdot x$, $y_{3t+1}\cdot\overrightarrow{x}\cdot y_{3t-1}$, and $y_{3t-1,1}\cdot\overrightarrow{y_{3t-1}}\cdot x$. If $r = 2$, then we make the additional jumps $y_{3t,1}\cdot\overrightarrow{y_{3t}}\cdot x$, $y_{3t+1}\cdot\overrightarrow{x}\cdot y_{3t-1}$, $y_{3t-1,1}\cdot\overrightarrow{y_{3t-1}}\cdot x$, and $x\cdot\overrightarrow{y_{3t+2}}\cdot y_{3t+2,1}$. Thus, $Fs(G) = s + c - t$, where $t = \left\lfloor \frac{n}{3} \right\rfloor$.

By a similar argument, if $c \geq 1$ and $\ell = 0$, then $T' = \{y_1, ..., y_n, x, y_{3i-1,1} : i \leq t\}$. If $n \equiv 2 \pmod 3$, then we also include $y_{n,1}$. In any case, we begin by making the jump $x\cdot\overrightarrow{y_1}\cdot y_{1,1}$. The current configuration of pegs is the same as in the previous case after the first two moves had been made. It follows that $Fs(K_{1,n}(c; a_1, ..., a_n)) = s + c - \left\lfloor \frac{n+1}{3} \right\rfloor$, where $c \geq 1$.

We now consider the case where $\ell \geq 1$. If $c = 0$, then the maximum independent set is $Y_1 \cup \cdots \cup Y_n \cup \{x\}$. As before, the dual of this configuration has no adjacent pegs. Hence, it is necessary to add at least one peg to the dual. We claim that adding x to the dual is the best choice. The reason is that we want to replace y_i with $y_{i,1}$ in the dual for all $i \geq m = \min\{\ell, \lfloor \frac{n}{2} \rfloor\}$. If x is in the fool's solitaire solution, then this is not possible because the fool's solitaire solution must be an independent set. Further, this will allow us to "exchange" pegs in Y_i with pegs in N, where $i \geq n - m$. Since x will be in the dual of our fool's solitaire solution, the method described here will also work when $c \geq 1$. Thus, in both cases we claim that $T' = \{y_1, ..., y_{n-m}, y_{n-m+1,1}, ..., y_{n,1}, x, y_{m+3i-1,1} : i = 1, ..., \lfloor \frac{n-2m}{3} \rfloor\}$. If $n - 2m > 0$ and $n - 2m \equiv 2 \pmod 3$, then we also include $y_{n-m,1}$ in the dual. We remove pegs from the dual using the moves $y_i \cdot \overrightarrow{x} \cdot y_{n-m+i}$ and $y_{n-m+i,1} \cdot \overrightarrow{y_{n-m+i}} \cdot x$ for $i = 1, ..., m$. Note that this is the same as the initial configuration in the case where $c \geq 1$ and $\ell = 1$. Thus, $Fs(G) = s + c - \lfloor \frac{n-2m+1}{3} \rfloor$. ■

7 OPEN PROBLEMS

We end our discussion by giving several open problems as a basis for future research.

What are the conditions for the solvability of trees with diameter five? Once the conditions for trees with diameter five are found, can we use the known conditions for trees with diameter less than six to predict conditions for trees of diameter six?

What percentage of trees are not solvable by Theorem 5.3? Can Theorem 5.3 be generalized to describe all graphs? Are there conditions on the degrees of the neighbors of the largest degree vertex that will provide better bounds for the non-solvability of a tree?

What other graphs have $F_s(G) < \alpha(G) - 1$? How far can $F_s(G)$ differ from $\alpha(G)$?

BIBLIOGRAPHY

[1] Martin Aigner and Michael Fromme. A game of cops and robbers. *Discrete Appl. Math.*, 8(1):1–11, 1984.

[2] John D. Beasley. *The Ins and Outs of Peg Solitaire*, volume 2 of *Recreations in Mathematics*. Oxford University Press, Eynsham, 1985.

[3] Robert A. Beeler, Aaron D. Gray, and D. Paul Hoilman. Constructing solvable graphs in peg solitaire. *Bull. Inst. Combin. Appl.*, 66:89–96, 2012.

[4] Robert A. Beeler and D. Paul Hoilman. Peg solitaire on graphs. *Discrete Math.*, 311(20):2198–2202, 2011.

[5] Robert A. Beeler and D. Paul Hoilman. Peg solitaire on the windmill and the double star. *Australas. J. Combin.*, 52:127–134, 2012.

[6] Robert A. Beeler and Hal Norwood. Peg solitaire on graphs solver applet. http://faculty.etsu.edu/BEELERR/solitaire/.

[7] Robert A. Beeler and Tony K. Rodriguez. Fool's solitaire on graphs. To appear in *Involve*.

[8] George I. Bell. Solving triangular peg solitaire. *J. Integer Seq.*, 11(4):Article 08.4.8, 22, 2008.

[9] Elwyn R. Berlekamp, John H. Conway, and Richard K. Guy. *Winning Ways for Your Mathematical Plays. vol. 2.* A K Peters Ltd., Natick, MA, second edition, 2003.

[10] Gary Chartrand, Linda Lesniak, and Ping Zhang. *Graphs & Digraphs.* CRC Press, Boca Raton, FL, fifth edition, 2011.

[11] Nicolaas G. de Bruijn. A Solitaire game and its relation to a finite field. *J. of Recreational Math*, 5(2):133–137, 1972.

[12] Antoine Deza and Shmuel Onn. Solitaire lattices. *Graphs Combin.*, 18(2):227–243, 2002.

[13] Lindsay Anne Erickson. *The game of Nim on graphs.* ProQuest LLC, Ann Arbor, MI, 2011. Thesis (Ph.D.)–North Dakota State University.

[14] Aviezri S. Fraenkel. Combinatorial games: selected bibliography with a succinct gourmet introduction. *Electron. J. Combin.*, 1:Dynamic Survey 2, 45 pp. (electronic), 1994.

[15] Paul D. Hanna. The On-Line Encyclopedia of Integer Sequences. http://oeis.org/A179467A179467, cited March 2013.

[16] Ron H. Hardin. The On-Line Encyclopedia of Integer Sequences. http://oeis.org/A146810A146810, cited March 2013.

[17] Irvin Roy Hentzel. Triangular puzzle peg. *J. of Recreational Math*, 6(4):280–283, 1973.

[18] Christopher Jefferson, Angela Miguel, Ian Miguel, and S. Armagan Tarim. Modelling and solving English Peg Solitaire. *Comput. Oper. Res.*, 33(10):2935–2959, 2006.

[19] Colin McDiarmid. On graphs with few disjoint t-star minors. *European J. Combin.*, 32(8):1394–1406, 2011.

[20] Judea Pearl. *Heuristics: Intelligent Search Strategies for Computer Problem Solving*. Artificial Intelligence Collection. Addison-Wesley Publishing Co., Reading, MA, USA, 1984.

[21] Neil J. A. Sloane. The On-Line Encyclopedia of Integer Sequences. http://oeis.org/A000055A000055, cited January 2013.

[22] Neil J. A. Sloane. The On-Line Encyclopedia of Integer Sequences. http://oeis.org/A001372A001372, cited March 2013.

[23] Marie-Jo Waeber. Marble or french solitaire. http://www.archimedes-lab.org/game_marble/solitaire.html, cited February 2013.

[24] Damian Yerrick. Peg solitaire. http://en.wikipedia.org/wiki/Peg_solitaire, cited February 2013.

APPENDIX

Tables

1	2	3	4	Final Peg
$y_{1,1}\cdot\overrightarrow{y_1}\cdot x$	$x_1\cdot\overrightarrow{x}\cdot y_1$	$y_{2,1}\cdot\overrightarrow{y_2}\cdot x$	$y_1\cdot\overrightarrow{x}\cdot x_1$	x_1
$y_{2,1}\cdot\overrightarrow{y_2}\cdot x$	$y_{1,1}\cdot\overrightarrow{y_1}\cdot y_{1,2}$	$x_1\cdot\overrightarrow{x}\cdot y_1$	$y_{1,2}\cdot\overrightarrow{y_1}\cdot x$	x
$y_{2,1}\cdot\overrightarrow{y_2}\cdot x$	$x_1\cdot\overrightarrow{x}\cdot y_2$	$y_{1,1}\cdot\overrightarrow{y_1}\cdot x$	$y_2\cdot\overrightarrow{x}\cdot y_1$	y_1
$y_{1,1}\cdot\overrightarrow{y_1}\cdot x$	$x_1\cdot\overrightarrow{x}\cdot y_1$	$y_{2,1}\cdot\overrightarrow{y_2}\cdot x$	$y_1\cdot\overrightarrow{x}\cdot y_2$	y_2
$y_{2,1}\cdot\overrightarrow{y_2}\cdot x$	$y_{1,1}\cdot\overrightarrow{y_1}\cdot y_{1,2}$	$x_1\cdot\overrightarrow{x}\cdot y_1$	$y_{1,2}\cdot\overrightarrow{y_1}\cdot y_{1,1}$	$y_{1,1}$
$y_{1,1}\cdot\overrightarrow{y_1}\cdot x$	$x_1\cdot\overrightarrow{x}\cdot y_1$	$y_{2,1}\cdot\overrightarrow{y_2}\cdot x$	$x\cdot\overrightarrow{y_1}\cdot y_{1,2}$	$y_{1,2}$
$y_{2,1}\cdot\overrightarrow{y_2}\cdot x$	$x_1\cdot\overrightarrow{x}\cdot y_2$	$y_{1,1}\cdot\overrightarrow{y_1}\cdot x$	$x\cdot\overrightarrow{y_2}\cdot y_{2,1}$	$y_{2,1}$

Table 3: Terminal states of $K_{1,2}(1;2,1)$ with $S=\{x,y_{1,2}\}$.

1	2	3	4	5	6	Final Peg
$y_{3,1}\cdot\overrightarrow{y_3}\cdot x$	$y_1\cdot\overrightarrow{x}\cdot y_3$	$y_{2,1}\cdot\overrightarrow{y_2}\cdot x$	$x_1\cdot\overrightarrow{x}\cdot y_1$	$y_{1,1}\cdot\overrightarrow{y_1}\cdot x$	$y_3\cdot\overrightarrow{x}\cdot x_1$	x_1
$y_{2,1}\cdot\overrightarrow{y_2}\cdot x$	$y_{1,1}\cdot\overrightarrow{y_1}\cdot y_{1,2}$	$x_1\cdot\overrightarrow{x}\cdot y_2$	$y_{3,1}\cdot\overrightarrow{y_3}\cdot x$	$y_2\cdot\overrightarrow{x}\cdot y_1$	$y_{1,2}\cdot\overrightarrow{y_1}\cdot x$	x
$y_{3,1}\cdot\overrightarrow{y_3}\cdot x$	$y_1\cdot\overrightarrow{x}\cdot y_3$	$y_{2,1}\cdot\overrightarrow{y_2}\cdot x$	$x_1\cdot\overrightarrow{x}\cdot y_1$	$y_{1,1}\cdot\overrightarrow{y_1}\cdot x$	$y_3\cdot\overrightarrow{x}\cdot y_1$	y_1
$y_{3,1}\cdot\overrightarrow{y_3}\cdot x$	$y_1\cdot\overrightarrow{x}\cdot y_3$	$y_{2,1}\cdot\overrightarrow{y_2}\cdot x$	$x_1\cdot\overrightarrow{x}\cdot y_1$	$y_{1,1}\cdot\overrightarrow{y_1}\cdot x$	$y_3\cdot\overrightarrow{x}\cdot y_2$	y_2
$y_{1,1}\cdot\overrightarrow{y_1}\cdot x$	$x_1\cdot\overrightarrow{x}\cdot y_1$	$y_{2,1}\cdot\overrightarrow{y_2}\cdot x$	$y_1\cdot\overrightarrow{x}\cdot y_2$	$y_{3,1}\cdot\overrightarrow{y_3}\cdot x$	$y_2\cdot\overrightarrow{x}\cdot y_3$	y_3
$y_{2,1}\cdot\overrightarrow{y_2}\cdot x$	$y_{1,1}\cdot\overrightarrow{y_1}\cdot y_{1,2}$	$x_1\cdot\overrightarrow{x}\cdot y_2$	$y_{3,1}\cdot\overrightarrow{y_3}\cdot x$	$y_2\cdot\overrightarrow{x}\cdot y_1$	$y_{1,2}\cdot\overrightarrow{y_1}\cdot y_{1,1}$	$y_{1,1}$
$y_{2,1}\cdot\overrightarrow{y_2}\cdot x$	$x_1\cdot\overrightarrow{x}\cdot y_2$	$y_{1,1}\cdot\overrightarrow{y_1}\cdot x$	$y_2\cdot\overrightarrow{x}\cdot y_1$	$y_{3,1}\cdot\overrightarrow{y_3}\cdot x$	$x\cdot\overrightarrow{y_1}\cdot y_{1,2}$	$y_{1,2}$
$y_{1,1}\cdot\overrightarrow{y_1}\cdot x$	$x_1\cdot\overrightarrow{x}\cdot y_1$	$y_{2,1}\cdot\overrightarrow{y_2}\cdot x$	$y_1\cdot\overrightarrow{x}\cdot y_2$	$y_{3,1}\cdot\overrightarrow{y_3}\cdot x$	$x\cdot\overrightarrow{y_2}\cdot y_{2,1}$	$y_{2,1}$
$y_{3,1}\cdot\overrightarrow{y_3}\cdot x$	$y_1\cdot\overrightarrow{x}\cdot y_3$	$y_{2,1}\cdot\overrightarrow{y_2}\cdot x$	$x_1\cdot\overrightarrow{x}\cdot y_1$	$y_{1,1}\cdot\overrightarrow{y_1}\cdot x$	$x\cdot\overrightarrow{y_3}\cdot y_{3,1}$	$y_{3,1}$

Table 4: Terminal states of $K_{1,3}(1;2,1,1)$ with $S=\{x,y_{1,2}\}$.

1	2	3	4	Final Peg
$y_{2,1}\cdot\overrightarrow{y_2}\cdot x$	$x_1\cdot\overrightarrow{x}\cdot y_2$	$y_{1,1}\cdot\overrightarrow{y_1}\cdot x$	$y_2\cdot\overrightarrow{x}\cdot x_1$	x_1
$y_{2,1}\cdot\overrightarrow{y_2}\cdot x$	$x_1\cdot\overrightarrow{x}\cdot y_2$	$y_{1,1}\cdot\overrightarrow{y_1}\cdot x$	$y_2\cdot\overrightarrow{x}\cdot y_1$	y_1
$y_{1,1}\cdot\overrightarrow{y_1}\cdot x$	$x_1\cdot\overrightarrow{x}\cdot y_1$	$y_{2,1}\cdot\overrightarrow{y_2}\cdot x$	$y_1\cdot\overrightarrow{x}\cdot y_2$	y_2
$y_{1,1}\cdot\overrightarrow{y_1}\cdot x$	$x_1\cdot\overrightarrow{x}\cdot y_1$	$y_{2,1}\cdot\overrightarrow{y_2}\cdot x$	$x\cdot\overrightarrow{y_1}\cdot y_{1,1}$	$y_{1,1}$
$y_{2,1}\cdot\overrightarrow{y_2}\cdot x$	$x_1\cdot\overrightarrow{x}\cdot y_2$	$y_{1,1}\cdot\overrightarrow{y_1}\cdot x$	$x\cdot\overrightarrow{y_2}\cdot y_{2,1}$	$y_{2,1}$

Table 5: Terminal states of $K_{1,2}(1;1,1)$ with $S=\{x\}$.

1	2	3	4	5	6	Final Peg
$y_{1,1}\cdot\overrightarrow{y_1}\cdot x$	$x_1\cdot\overrightarrow{x}\cdot y_1$	$y_{3,1}\cdot\overrightarrow{y_3}\cdot x$	$y_1\cdot\overrightarrow{x}\cdot y_3$	$y_{2,1}\cdot\overrightarrow{y_2}\cdot x$	$y_3\cdot\overrightarrow{x}\cdot x_1$	x_1
$y_{1,1}\cdot\overrightarrow{y_1}\cdot x$	$x_1\cdot\overrightarrow{x}\cdot y_1$	$y_{3,1}\cdot\overrightarrow{y_3}\cdot x$	$y_1\cdot\overrightarrow{x}\cdot x_1$	$y_{2,1}\cdot\overrightarrow{y_2}\cdot x$	$x_1\cdot\overrightarrow{x}\cdot y_1$	y_1
$y_{1,1}\cdot\overrightarrow{y_1}\cdot x$	$x_1\cdot\overrightarrow{x}\cdot y_1$	$y_{3,1}\cdot\overrightarrow{y_3}\cdot x$	$y_1\cdot\overrightarrow{x}\cdot x_1$	$y_{2,1}\cdot\overrightarrow{y_2}\cdot x$	$x_1\cdot\overrightarrow{x}\cdot y_2$	y_2
$y_{1,1}\cdot\overrightarrow{y_1}\cdot x$	$x_1\cdot\overrightarrow{x}\cdot y_1$	$y_{3,1}\cdot\overrightarrow{y_3}\cdot x$	$y_1\cdot\overrightarrow{x}\cdot x_1$	$y_{2,1}\cdot\overrightarrow{y_2}\cdot x$	$x_1\cdot\overrightarrow{x}\cdot y_3$	y_3
$y_{1,1}\cdot\overrightarrow{y_1}\cdot x$	$y_3\cdot\overrightarrow{x}\cdot y_1$	$y_{2,1}\cdot\overrightarrow{y_2}\cdot x$	$x_1\cdot\overrightarrow{x}\cdot y_3$	$y_{3,1}\cdot\overrightarrow{y_3}\cdot x$	$x\cdot\overrightarrow{y_1}\cdot y_{1,1}$	$y_{1,1}$
$y_{2,1}\cdot\overrightarrow{y_2}\cdot x$	$y_1\cdot\overrightarrow{x}\cdot y_2$	$y_{3,1}\cdot\overrightarrow{y_3}\cdot x$	$x_1\cdot\overrightarrow{x}\cdot y_1$	$y_{1,1}\cdot\overrightarrow{y_1}\cdot x$	$x\cdot\overrightarrow{y_2}\cdot y_{2,1}$	$y_{2,1}$
$y_{1,1}\cdot\overrightarrow{y_1}\cdot x$	$x_1\cdot\overrightarrow{x}\cdot y_1$	$y_{3,1}\cdot\overrightarrow{y_3}\cdot x$	$y_1\cdot\overrightarrow{x}\cdot y_3$	$y_{2,1}\cdot\overrightarrow{y_2}\cdot x$	$x\cdot\overrightarrow{y_3}\cdot y_{3,1}$	$y_{3,1}$

Table 6: Terminal states of $K_{1,3}(1;1,1,1)$ with $S = \{x\}$.

VITA

CLAYTON WALVOORT

Education:	B.S. Mathematics, Ferrum College, Ferrum, Virginia, 2011
	M.S. Mathematics, East Tennessee State University, Johnson City, Tennessee, 2013
Professional Experience:	Undergraduate Assistant, Ferrum College, Ferrum, Virginia, 2010-2011
	Mathematics Instructor, Patrick Henry CC UBMS, Martinsville, Virginia, 2011
	Graduate Assistant, East Tennessee State University, Johnson City, Tennessee 2011–present
	GK12 Fellow, East Tennessee State University, Johnson City, Tennessee 2011–present
Publications:	Robert A. Beeler and Clayton A. Walvoort "Peg Solitaire on Trees with Diameter Four," in preparation.
	Robert A. Beeler and Clayton A. Walvoort "Peg Solitaire using Purges," in preparation.