5-2021

Peg Solitaire on Graphs In Which We Allow Merging and Jumping

Amanda L. McKinney
East Tennessee State University

Peg Solitaire on Graphs In Which We Allow Merging and Jumping

Amanda L. McKinney
East Tennessee State University

April 15, 2021

ABSTRACT

Peg solitaire is a game in which pegs are placed in every hole but one and the player jumps over pegs along rows or columns to remove them. Usually, the goal of the player is to leave only one peg. In a 2011 paper, this game is generalized to graphs. In this thesis, we consider a variation of peg solitaire on graphs in which pegs can be removed either by jumping them or merging them together. To motivate this, we survey some of the previous papers in the literature. We then determine the solvability of several classes of graphs including stars and double stars, caterpillars, trees of small diameter, particularly four and five, and articulated caterpillars. We conclude this thesis with several open problems related to this study.

Contents

List of Figures

ACKNOWLEDGEMENTS

I would like to thank my thesis advisor, Dr. Robert Beeler for his invaluable contributions over the course of this project, without whom, this would not have been possible. Special thanks to Dr. Rodney Keaton, for serving as a reader and supporting both this project and its presentation. I would like to extend my severe thanks to the East Tennessee State University Honors College, particularly my program coordinator Dr. Karen Kornweibel, for providing me with an enriched undergraduate experience and the opportunity to write this honors thesis.

I would also like to take this opportunity to thank my fiancé, Justin Kilby, for his profound belief in my abilities, and for being patient as this chapter ends and new ones begin together. I would also like to thank my parents, Robin and Jan McKinney. Mom and Dad, I have no words to acknowledge the sacrifices you made and the dreams you had to let go, just to give me a shot at achieving mine; thank you both. To my sister Cas McKinney, you mean the world to me - here's to us chasing our destinies. Finally, thank you to Mercedes Forgey; your friendship has rewarded me with love and understanding, and as we embark on new adventures, we will always have each other.

Please know that all of your guidance and support has not gone unnoticed.

Chapter 1

INTRODUCTION

Peg solitaire is a game that is familiar to most people - it is the wooden peg game that you might find at a popular restaurant. However, few people realize that this game dates back to at least 1697, as evidenced by the engraving in Figure 1.1, when it was played with stones that were set in holes [2]. In fact, Beasley uses a quote from Gottfried Leibniz from 1710 about peg solitaire: "Not so very long ago, there became widespread an excellent kind of game, called solitaire, where I play on my own, but as it with a friend as witness and referee to see that I play correctly."

Today, this classic game consists of a game board with pegs. Traditionally, the player begins with a single hole in the game board and the player jumps pegs in order to remove them one at a time until there are no possible moves remaining, with the goal of only leaving one peg. A jump moves one peg from its position over another peg into an empty hole, removing the peg in the middle with the move. The game is solved if there is only one peg remaining, and we call this a solved state for the game. Similarly, if a game board can reach a solved state, then it is called solvable. Some common boards include the English cross and the European board. These, along with the triangular board are illustrated in Figure 1.2. Readers are most likely to be familiar with the fifteen hole triangular board featured on the tables of Cracker Barrel restaurants. For more information on traditional peg solitaire, the reader is referred to [2, 15].

In 2011, Beeler and Hoilman generalized peg solitaire to graphs in the combinatorial sense. Our graph theoretical terminology and notation will be consistent with Buckley and Lewinter [16]. A *graph G* is a set of *vertices* together with a set of *edges*. An example of a graph on five vertices is given

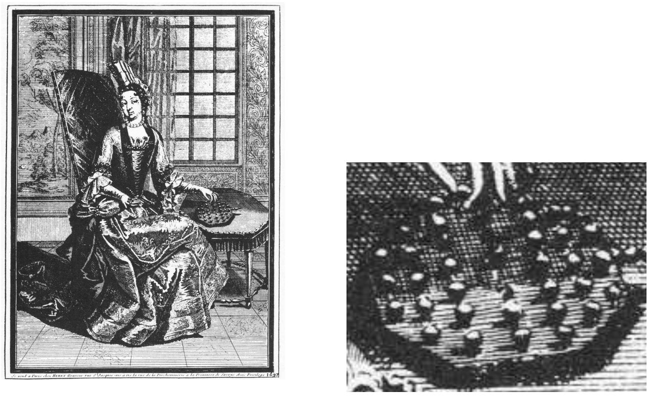

Figure 1.1: *Madame la Princesse de Soubise joüant au jeu de Solitaire* by Claude-Auguste Berey, 1697

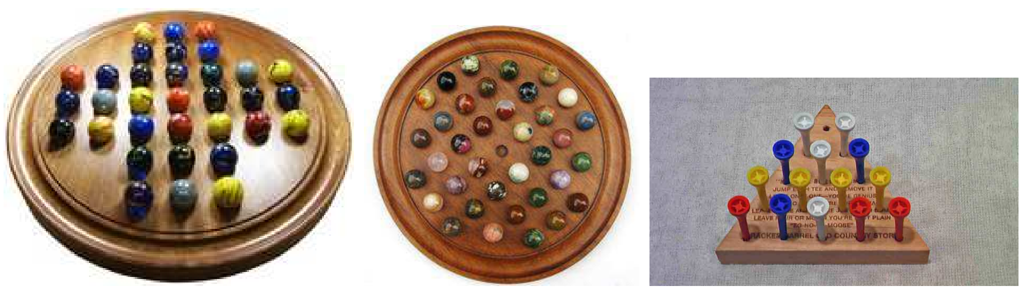

Figure 1.2: The English cross, the European, and the triangular game boards

in Figure 1.3. The vertex set of G is denoted $V(G)$. The edge set of G is denoted $E(G)$. We will assume that both $V(G)$ and $E(G)$ are nonempty and finite. Each edge in G will have a pair of distinct vertices called *endpoints*. Given vertices $u, v \in V(G)$, a $u - v$ *path* is an alternating sequence of vertices and edges that begins at u and ends at v such that no vertex is listed twice in this sequence. If for every $u, v \in V(G)$ there is a $u - v$ path, then we say that G is *connected*. We will assume that all graphs are connected.

In peg solitaire on graphs, 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 $xy, yz \in E(G)$. Such a jump will be denoted $x \cdot \overrightarrow{y} \cdot z$. An example is given in Figure 1.4. In our figures, a hollow vertex represents a hole, while a solid vertex represents

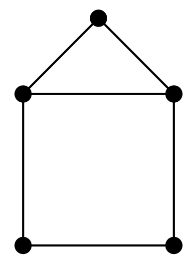

Figure 1.3: An example of a graph, G

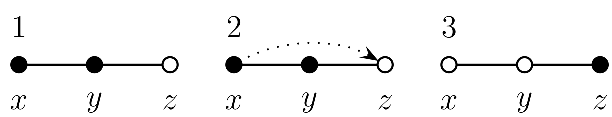

Figure 1.4: A typical jump in peg solitaire, $x \cdot \overrightarrow{y} \cdot z$

a peg. In particular, playing the game on graphs allows for L-shaped jumps, which are usually not allowed in the traditional game. In the next chapter, we will discuss more results on peg solitaire on graphs as well as its variants.

In order to facilitate our discussion, it is useful to introduce basic concepts from graph theory. Our terminology and notation will be consistent with Buckley and Lewinter [16]. Two vertices are *adjacent* if there is an edge between them. A graph H is a *subgraph* of a graph G if $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$. If $V(H) = V(G)$ and $E(H) \subseteq E(G)$, then H is a *spanning subgraph* of G. For a graph $G = (V, E)$, let A be a subset of the vertex set V. The *subgraph of G induced by A*, denoted G_A, is the graph with vertex set A and $E(G_A) = \{xy : x, y \in A, xy \in E(G)\}$. Such graphs are called *vertex induced subgraphs*.

Note, the types of graphs that we are interested in for the sake of this thesis are finite and contain no loops or multiple edges. In other words, there is a maximum of one edge between any two vertices and an edge must be between two distinct vertices. There are many families of graphs, each with their own properties. Much of our time will be spent with a family of graphs known as trees. A *tree* is a type of graph that is connected and acyclic. Trees contain no cycles, or a closed path within a graph such that you can begin at one vertex and traverse edges no more than once to return to the original vertex, and are thus called acyclic. Trees can take on named shapes such as a star, double star, caterpillar, or path, just to name a few. The specific definitions of such graphs will be discussed later. The following are useful and well-known properties of trees.

Proposition 1.0.1 *Trees have the following properties.*

(i) A nontrivial tree contains branches and at least two leaves.

(ii) Given two vertices, x and y, of any tree, T, there is a unique $x - y$ path.

(iii) The deletion of any edge disconnects the graph.

We are interested in exploring trees since every connected graph has a spanning tree. If we can explore what trees are solvable, then other graphs will be implied to be solvable by the following proposition.

Proposition 1.0.2 (Inheritance Principle [9]) *If H is a solvable spanning subgraph of G, then G is solvable.*

There are some other graph theory terms that will prove useful for this research. The *degree* of a vertex is how many edges meet at the vertex and can be classified as even or odd. The *distance* between vertices u and v is the number of edges on the shortest path from u to v. The *diameter* of a graph is the greatest distance between any pair of vertices. A *complete graph* is one in which every vertex is adjacent to all other vertices. The complete graph on n vertices is denoted K_n. A *bipartite graph* is composed of two sets of vertices X and Y such that if $uv \in E$, then $u \in X$ and $v \in Y$. A *complete bipartite graph* is a special kind of bipartite graph where every vertex of the first set is connected to every vertex of the second set. The complete bipartite graph with partite sets of cardinality $|X| = n$ and $|Y| = m$ is denoted $K_{n,m}$. In particular, the complete bipartite graph $K_{1,n}$ is called a *star*. With these definitions in mind, we can explore the current research that has been conducted about peg solitaire on graphs in the next chapter.

Chapter 2

LITERATURE REVIEW

2.1 Peg Solitaire

Despite its long history, peg solitaire is a game that has not had a deep mathematical connection until recently. In 2011, Beeler and Hoilman generalized the ideas behind the table game of peg solitaire (such as the infamous triangle peg board) to any arbitrary board [9]. These boards can be thought of as connected graphs for purposes of analyzing and research, along with general play of the game. This generalization allows us to label the vertices as the holes of the board, which are solid if they contain a peg and hollow otherwise, and allow moves to occur between any adjacent vertices, despite variations that occur in the depiction of the graph. The edges represent locations for possible moves to occur since they determine adjacencies between pegs.

Since the original 2011 paper [9], Beeler and other mathematicians, have expanded the concepts of this game and transposed it to a graph theoretical background so that it can be more easily studied on shapes other than the traditional boards such as the English cross, the European board, and the triangular board. In 2011, Beeler and Hoilman wrote about peg solitaire on graphs, where vertices are the holes of the board and edges represent adjacencies between the holes [9]. They write that the main difference between a traditional game board and peg solitaire on graphs is the allowance of L-shaped moves. There have been a number of papers in which the solvability of various classes of graphs is discussed. In this section, we survey those results that will be pertinent to this paper.

First, some basic definitions. A *starting state* S is the set of vertices that have a hole at the beginning of the game. Unless otherwise noted, S will consist of a single vertex. A *terminal state* T is the set of vertices that have a peg at the end of the game. A terminal state T is associated with a starting state S if T can be obtained from S via a sequence of legal moves. A *solvable graph* is one in which for some configuration of pegs occupying all but one vertex, some sequence of jumps leaves a single peg [21]. A graph G is *freely solvable* if for all vertices s so that, starting with $S = \{s\}$, then there exists an associated terminal state consisting of a single peg. A graph G is *k-solvable* if there exists some vertex s so that, starting with $S = \{s\}$, then there exists an associated minimal 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 $S = \{s\}$, then there exists an associated terminal state consisting of two pegs that are distance 2 apart.

With peg solitaire being applied to graphs, the next logical step is to look at the gameplay on families of graphs. Beeler and Hoilman studied the solvability of several classes of graphs, including complete graphs for $n \geq 2$, complete bipartite graphs, and the Cartesian products of two solvable graphs [9]. The *Cartesian product* of two graphs, $G \times H$ with vertex sets $u_1, u_2, ..., u_m$ and $v_1, v_2, ..., v_n$, respectively, is the graph with the vertex set consisting of mn vertices labeled (u_i, v_j), where $1 \leq i \leq m$ and $1 \leq j \leq n$. Further, two vertices in $G \times H$ are adjacent if either (i) $i = h$ and v_j is adjacent to v_k in H, or (ii), $j = k$ and u_i is adjacent to u_h in H. de Wiljes and Kreh have also written on the Cartesian products of graphs and peg solitaire. For example, it is known that star graphs are not solvable in traditional peg solitaire; however, these authors show that the Cartesian product of two stars on at least two vertices is solvable [25].

Importantly, Beeler and Hoilman noted that the star $K_{1,n}$ is $(n-1)$-solvable, which will be a key note later on when we explore stars with other variations [9]. Paths, P_n, prove to be deceivingly tricky: despite the simplicity of this graph, defining the possibilities of a solved state for a path depends on the parity of n.

Proposition 2.1.1 *[9] The path, P_n, is freely solvable if and only if $n = 2$; P_n is solvable if and only if n is even or $n = 3$; P_n is distance 2-solvable in all other cases.*

They also note that complete graphs for $n \geq 2$ and complete bipartite graphs are freely solvable [9]. Allowing for additional moves would result in additional graphs being freely solvable in these variants. We leave such variants as possible avenues for future research.

After the initial generalization of the peg solitaire game to graphs, there were more published proofs for families of graphs and specific cases of graphs that are considered solvable. In 2012, Beeler and Hoilman prove that the windmill and certain types of double stars are solvable with the traditional jump move [10]. A *double star* is a type of tree that has two star subgraphs with an independent number of pendants attached to each center and whose centers are adjacent along a spine. An example of a double star is shown in Figure 2.1. Since double stars are trees of diameter three, there are several ways to isolate pegs, which means that more pendants must be "exchanged" across the graph, so they are not all solvable using only jumps. The proof provided in the paper by Beeler and Hoilman shows that the double star $S_{m,n}$ is freely solvable if and only if $m \neq n$ and $S_{m,n}$ is solvable if and only if $m \leq n+1$ [10]. This will likely be a key result as we examine stars and caterpillars.

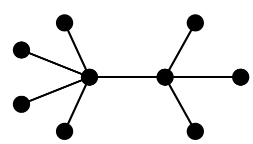

Figure 2.1: The Double Star - $S_{4,3}$

An interesting graph family that has been proven to be freely solvable is Sierpinski graphs. The Sierpinski graph $S(k,n)$, where $(n,k \geq 1)$ is defined on the vertex set $\{0,1,...,k-1\}^n$. Two vertices, $u = (u_1,...,u_n), v = (v_1,...,v_n)$ are adjacent if and only if there exists a $t \in 1,2,...,n$ such that $u_s = v_s, s < t$, $u_t \neq v_t$, and $u_s = v_t$ and $v_s = u_t$ for $s = t+1,...,n$. In a paper by H. Akyar, Çakmak, Torun, and E. Akyar, it is shown that these Sierpinski graphs are freely solvable in traditional peg solitaire [1].

There are also particular methods that can be used so that one can construct a solvable graph. One example of such a method is given by Beeler, Gray, and Hoilman: If G has a universal vertex u and the graph obtained from G by deleting u is connected, then G is freely solvable. If G contains two universal vertices, then G is freely solvable [7]. Another method that is discussed is overlap, which is where for two graphs G and H, we have all of the vertices and edges of both G and H, but some are shared; the overlap can be a basis for a solvable graph [7]. When examining graphs, we will be sure to look for these types of universal vertices and overlap to ensure that cases of graph families are considered as we look for more solvable graphs. That is, if a graph contains these elements, then we will be sure not to be too hasty to ensure that there is not another graph within the same graph family that does not contain these elements. Perhaps, the overlap will be a solvable graph, when, once solved, leaves two solvable portions.

In this project, we will be particularly interested in different types of trees, such as double stars, caterpillars, and trees of small diameter. Relevant previous results from [8, 10, 13] will be referenced as each particular graph is discussed in Section 3.

2.2 Variations

Naturally, there have been a number of variations of graphical peg solitaire. In *fool's solitaire*, the goal is to leave the maximum number of pegs, under the caveat that the player makes a jump whenever possible. The fool's solitaire variant is discussed extensively in [11, 26]. However, the fool's solitaire problem is determined for specific graph families in the papers discussing those graphs (see for example [4, 8, 13, 20]).

Some versions of this game are less concerned with solvability and are more concerned about making game play more exciting for one or more players, as is the case with Beeler and Gray's version of *duotaire* [5]. Another variation is known as the double jump solitaire, where each peg must be jumped twice in order to be removed [6]. A similar problem is studied by Davis et al in [18]. In this paper, each peg has one of two colors. When a peg is jumped by one of the same color, it switches colors. When a peg is jumped by one of the opposite color, it is removed.

Another variation is given by de Wiljes and Kreh in [19], where they examine a game they call *stick solitaire*. In this variation, the player considers sticks along the edges of the graph, rather than pegs in the vertices and tries to remove every sticks but one. This allows the possibility of a graph to be solvable in peg solitaire, but not in stick solitaire, or vice versa.

In some of these variations, the portfolio of game moves has been expanded. One of the more unusual approaches is by Bullington [17]. Bullington [17] considers a variation in which a move (see Figure 2.2) is defined as:

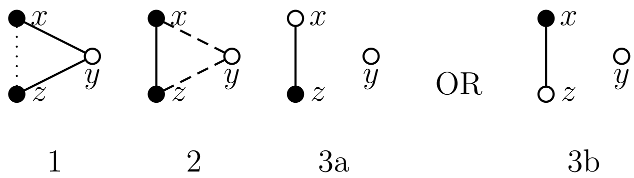

Figure 2.2: A typical move in Bullington's variation

(i) Suppose that 'peg' vertices x and z are adjacent to a 'hole' vertex y.

(ii) Add an edge between x and z if there is not one there already.

(iii) Delete the edges xy and yz.

(iv) Choose either x or z to be a 'hole' vertex.

What makes Bullington's variation so unusual is that the structure of the graph changes as the player progresses through the game.

Another move variation is called an *unjump*. This is best defined as the inverse of a jump. In 2015, Enbergs and Stocker describe what they call reversible peg solitaire on graphs, which allows for both jumps and unjumps, which is simply the reversal of a jump [21]. An example of an unjump is given in Figure 2.3. In the game of reversible peg solitaire, non-stars that contain a vertex of at least three and cycles and paths on n vertices, where n is divisible by two or three are solvable graphs, and all other graphs are not solvable [21]. Unmoves such as the unjump may prove helpful to this project since backstepping may be necessary as we determine the solvability of graph families. They are also helpful in looking at other variations peg solitaire in general.

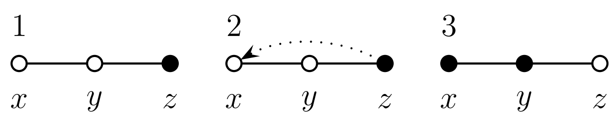

Figure 2.3: A typical unjump in peg solitaire

For the purposes of this thesis, the most significant variation is the merging peg solitaire introduced by Engbers and Weber in 2017 [22]. The merge move is defined by considering vertices x, y, and z with x and y adjacent and y and z adjacent. However, now we start with pegs on vertices x and

z only, and the new move merges those two pegs to a single peg on y [22]. This move will be denoted $(x, z) \to y$. An example of a merge move is given in Figure 2.4. It is worth noting that this move is inspired by the rubbling move introduced by Belford and Sieben [14] as well as moves on the game "Lights out" (see for example [23]).

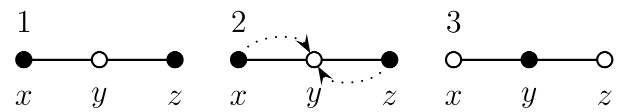

Figure 2.4: An example of a merging move, $(x, z) \to y$

Several results from this paper are given using the following theorem. As the result for paths is useful for several of our results, we include its proof.

Theorem 2.2.1 *[22] In merging peg solitaire:*

(i) The star $K_{1,n}$ is not solvable.

(ii) If $n \geq 2$, then the path P_n is solvable.

(iii) The double star $S_{n,m}$ is solvable if and only if $|n - m| \leq 1$.

Proof. *(ii)* If $n = 2$, then we are done since one hole and one peg remain. Suppose $n > 2$. Label the vertices $v_1, v_2, ..., v_n$ and let the hole be in v_2. For $i = 2, ..., n-1$, we merge v_{i-1} and v_{i+1} into v_i by $(v_{i-1}, v_i) \to v_{i+1}$. This ends the game with a final peg in v_{n-1}. Thus, we successfully solve the graph. ■

With the exception of the paper "Reversible peg solitaire on graphs" by Engbers and Stocker [21], only one type of move has been allowed. Typically, this move has been the jump (see for example [1, 3, 4, 5, 6, 7, 9, 10, 11, 13, 18, 20, 25, 26]). In particular, only merge moves are allowed in "Merging peg solitaire on graphs" by Engbers and Weber [22]. Instead, we will look at how jump and merge moves interact with each other. In other words, we want to determine what combinations of these moves look like in game play.

In this thesis, we study combinations of jumps and merges and try to prove that types of trees, and possibly other families of graphs, are solvable blueprints for this classic game. We also plan to study what implications that the initial choice for the hole has on the outcome of the final state of the graph. This research will be completed primarily via inductive methods of mathematical proofs and will impact future research of this modern idea.

Chapter 3

RESULTS

We are motivated by the comments in the previous section to consider a variation of peg solitaire in which we allow both jump and merge moves. For convenience of exposition, we refer to this as the *jump+merge variant*. In particular, we are interested in how the combination of these moves can help solve graph families. With a portfolio of possible moves in hand, the next step is to research how combinations of moves can affect the solvability of different graph families. In this variation, we see that some graphs are immediately given as solvable.

Proposition 3.0.1 *The following cases are solvable in the jump+merge variant:*

(i) Any distance 2-solvable graph,

(ii) Any solvable graph in the jump-only variant,

(iii) Any solvable graph in the merge-only variant.

Proof. *(i)* Using only jumps, obtain the terminal state of the distance 2-solvable graph. By definition, there are two pegs that are distance two apart. At this point, merge the final two pegs and a solved state has been reached.

(ii) and (iii) In either case, the addition of a second move option will not interfere with the solvability since at worst, both moves are not required to be used in the jump+merge variant. ■

We now consider several families of trees.

3.1 Stars and Double Stars

A star graph, $K_{1,n}$ is a type of tree that has a center vertex, v and n adjacent pendants around the center vertex. A *pendant* is a vertex of degree one. In the case of stars, pendants are only adjacent to the center vertex. These graphs are not solvable with only jumps, since if the hole is in the center vertex, then there are no possible moves [9]. Likewise, if the hole is in any pendant, then there is only one possible jump before there are no adjacent pegs. Likewise, if we allow only merges, then the graph is not solvable. To see this, notice that once there is a peg in the center vertex, then no further merges are possible [22]. In several other variants such as [6, 17, 18, 21] the star is an unsolvable graph. Interestingly, if we allow for merges as well as jumps, then stars are solvable.

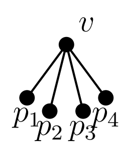

Figure 3.1: The star graph, $K_{1,n}$

Theorem 3.1.1 *Star graphs, $K_{1,n}$, with n pendants are solvable using jumps and merges.*

Proof. First, define the center vertex to be v and label the pendants $p_1, p_2, ..., p_n$.

Case 1 The hole is in v. For $i = 1, ..., \lfloor n/2 \rfloor$, merge $(p_{2i-1}, p_n) \to v$ and then jump $p_{2i} \cdot \overrightarrow{v} \cdot p_n$. If n is even, then this pattern will solve the graph. If n is odd, then merge $(p_n, p_{n-1}) \to v$, and a solved state will be obtained.

Case 2 The hole is in any pendant, say p_1. Then, jump $p_n \cdot \overrightarrow{v} \cdot p_1$. Ignoring the hole in p_n, this is a star with $n-1$ pendants and a hole in the center vertex v, so it is solvable by the above algorithm. ∎

The technique used in the proof of Theorem 3.1.1 can be applied to other graphs as well. To do this, we define a package and a purge based on the above argument. Packages and purges are discussed extensively in [12, 15], but are also used in [4, 13, 20]. When we solve graphs, we often deal with subgraphs using the same sequences of moves. Packages and purges are "short cuts" that allows us to rapidly progress through the solution of a graph without repeating these same moves. A *package* is a subgraph which has a specific configuration of pegs and holes. A *purge* is a sequence of moves (traditionally jumps) which will preserve the locations of certain pegs and holes and remove the remaining pegs on this subgraph. The pegs and holes which are restored to their original locations are called the *catalyst*.

Motivated by the above comments, we define a *Star Purge*. Define a star $K_{1,n}$ with n vertices, where the center vertex is labeled v and the pegs are labeled $p_1, ..., p_n$. If the hole is in v, then merge $(p_1, p_n) \to v$ and then jump $p_2 \cdot \overrightarrow{v} \cdot p_n$. We continue in this fashion using the algorithm defined above, but we can further define a collection of moves, $S_v(X, 2d)$, where X is the set and $2d$ is the number of pegs we remove from X.

If the hole is in a pendant, say p_n, then we jump $p_1 \cdot \overrightarrow{v} \cdot p_n$ and then merge $(p_2, p_n) \to v$. In this case, we denote this as $S_{p_n}(X, 2d)$, where X is the set and $2d$ is the number of pegs we remove from X.

Note, if $K_{1,n}$ has an even number of pegs and the initial hole is in a pendant, then the final peg will be in a pendant, and if $K_{1,n}$ has an odd number of pendants, then the final peg will be in v. Similarly, if $K_{1,n}$ has an even number of pegs and the initial hole is v, then the final peg will be in v, and if $K_{1,n}$ has an odd number of pendants, then the final peg will be in a pendant. This idea will become even more prevalent in some later results. Notice that stars are freely solvable in the jump+merge variant, which is significant since this purge is used to solve other graphs that my or may not be freely solvable.

As an example of this technique, we use the star purge to investigate a special case of caterpillars, the double star, $S_{m,n}$. The double star is a graph with two spinal vertices that are connected by a single edge, like a path, each with an independent number of pendants. We want to use the star purge to prove that double stars are solvable using jumps and merges. If this can be done, then we will have classified a type of caterpillar that is solvable in the jump+merge variant.

Theorem 3.1.2 *The double star $S_{m,n}$ is solvable in the jump+merge variant.*

Proof. Note that $S_{1,1}$ is the path P_4. This was shown to be solvable in

Theorem 2.2.1. This being the case, we will assume that $m \geq n$ and that $m \geq 2$.

First, define the center vertices to be x and y and label the pendants $x_1, x_2, ..., x_m$ and $y_1, y_2, ..., y_n$, respectively. Since the parity of the spinal vertices will affect the outcomes of the star purges, we will examine each possible case for the parity of m and n.

Case 1 m and n are even.

Case 1a Let the hole be in any pendant, say x_1. Complete a star purge on the pegs in $\{x_2, ..., x_m, y\}$ with x_1 (hole) and x (peg) as the catalyst. We are left with a peg in x and a hole in y. Complete a second star purge on the pendants of y with x (peg) and y (hole) as the catalyst to solve.

Case 1b The hole is in either spinal vertex, say x. Complete a star purge on the pegs in $\{x_2, ..., x_m, y\}$ with x (hole) and x_1 (peg) as the catalyst. We are left with a peg in x_1 and holes in x and y. Complete a second star purge on $\{y_3, .., y_n\}$ with y (hole) and y_1 (peg) as the catalyst. This leaves pegs in x_1, y_1, and y_2. Merge $(y_1, y_2) \rightarrow y$ and $(y, x_1) \rightarrow x$ to solve.

Case 2 m and n are odd.

Case 2a Let the hole be in any pendant, say x_1. Jump $x_m \cdot \overrightarrow{x} \cdot x_1$. Complete a star purge on the pendants of x with x (hole) and y (peg) as the catalyst. This leaves us with a peg in y and a hole in x. Jump $y_n \cdot \overrightarrow{y} \cdot x$. Complete another star purge on the pendants of y with y (hole) and x (peg) as the catalyst. We are left with peg in x.

Case 2b The hole is in either spinal vertex, say x. Complete a star purge on the pendants of x with x (hole) and x_1 (peg) as the catalyst. Merge $(x_1, y) \rightarrow x$. We are left with a peg in x and a hole in y. Complete a second star purge on the pendants of y with x (peg) and y (hole) as the catalyst to solve.

Case 3 Exactly one of m and n is odd. Without loss of generality, suppose that m is even and n is odd.

Case 3a Let the hole be in a pendant of x, say x_1. Complete a star purge on the pendants of x with x_1 (hole) and x (peg) as the catalyst. This will leave a peg in one of the pendants of x, say x_2. Jump $x_2 \cdot \overrightarrow{x} \cdot x_1$ and merge $(x_1, y) \rightarrow x$ leaving a peg in x and a hole in y. We then perform a star purge on the pendants of y with x (peg) and y (hole) as the catalyst. This leaves a peg in x, a peg in y_1, and holes elsewhere. The merge $(x, y_1) \rightarrow y$ completes the solution.

Case 3b Suppose the hole is in a pendant of y, say y_1. Complete a star purge on the pendants of y with y_1 (hole) and y (peg). We perform the moves $x{\cdot}\overrightarrow{y}{\cdot}y_1$, $(x_m, x_{m-1}) \to x$, and $(x, y_1) \to y$. If $m = 2$, then the graph is solved. If not, then we perform a star purge with catalyst x (hole) and y (peg) to remove the remaining pegs from the pendants of x.

Case 3c Let the hole be in x. Complete a star purge with x (hole) and x_1 (peg) as the catalyst. This leaves pegs in x_1 and x_2. We perform the moves $(x_2, y) \to x$, $x_1{\cdot}\overrightarrow{x}{\cdot}y$, and $y_n{\cdot}\overrightarrow{y}{\cdot}x$. We are left with pegs in x, y_1,...,y_{n-1} and holes elsewhere. Complete a second star purge on the remaining pendants of y with x (peg) and y (hole) as the catalyst to solve.

Case 3d Let the hole be in y. Complete a star purge to remove pegs from the pendants of y using y (hole) and y_1 (peg) as the catalyst. Merge $(x, y_1) \to y$. We are left with a peg in y and a hole in x. Complete a second star purge on the pendants of x with x (hole) and y (peg) as the catalyst to solve. ■

Notice that the double star is also freely solvable, since the choice of the initial hole does not affect the solvability of the graph. We will see later how the star purge and other purges can be applied to solve other graphs.

3.2 Caterpillars

Stars and double stars are special cases of a family of trees known as caterpillars. A *caterpillar* is obtained from the path on n vertices by appending pendants to the existing vertices of the path. The vertices of the original path, which are called the *spine* of the caterpillar, are labeled x_1,...,x_n in the natural way. We call n the *spine length*. We append a_i pendants to x_i for $1 \leq i \leq n$. The pendants adjacent to x_i will be denoted $x_{i,1}$,...,x_{i,a_i}. Similarly, we define the set $X_i = \{x_{i,1}, ..., x_{i,a_i}\}$. The caterpillar with parameters n, a_1,...,a_n will be denoted $P_n(a_1, ..., a_n)$. An example of a caterpillar is illustrated in Figure 3.2. Without loss of generality, we will assume that for $i \in \{1, n\}$, $a_i \geq 1$. While some literature calls the pendants of the caterpillar *leaves*, we will be referring to them as pendants to remain consistent with our definitions of the star and double star.

In the paper by Beeler, Green, and Harper [8], the solvability for several classes of caterpillars were determined for the jump-only variant of peg solitaire on graphs. These classes include caterpillars of the form $P_n(a_1, ..., 1)$ (i.e., broomsticks), caterpillars of the form $P_n(a_1, ..., 1, a_n)$ (i.e., dumbbells), and caterpillars of the form $P_4(a_1, a_2, a_3, a_4)$.

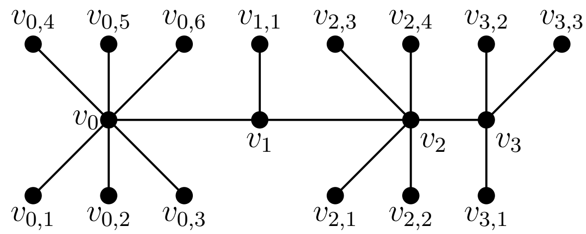

Figure 3.2: The caterpillar $P_4(6, 1, 4, 3)$

In the paper by Engbers and Weber [22], the solvability of several classes of caterpillars is determined for the merge-only variant. These classes include dumbbells (particularly the cases $P_3(a_1, 0, a_3)$ and $P_4(a_1, 0, 0, a_4)$) and caterpillars of the form $P_n(a_1, ..., a_n)$, where $a_1 = t_1$, $a_n = t_{n-1}$, and $a_i = t_i + t_{i-1}$ for $2 \leq i \leq n-1$ and non-negative integers t_1,..,t_{n-1}.

However, determining necessary and sufficient conditions for the solvability of a general caterpillar in either variant is currently an unsolved problem. For this reason, we wish to expand on the results of [8] and [22] by considering the solvability of certain classes of caterpillars in the jump+merge variant.

Theorem 3.2.1 *The caterpillar $T = P_n(a_1, ..., a_n)$ with $n \geq 1$ and $a_i \geq 1$ for $i = 1, ..., n$ is solvable in the jump+merge variant.*

Proof. We prove that the caterpillar $P_n(a_1, ..., a_n)$, where $a_i \geq 1$ for $i = 1, ..., n$, is solvable when the initial hole is in the end of the spine, x_n. In order to do this, proceed by induction on the spine length n.

Let $n = 1$ and begin with the initial hole in x_1. In this case, $P_1(a_1)$ is isomorphic to the star K_{1,a_1}. If a_1 is even, then the star purge $S_{x_1}(X_1, a_1 - 2)$ followed by the merge $(x_{1,u}, x_{1,2}) \to x_1$ solves the graph with final peg in x_1. If a_1 is odd, then the star purge $S_{x_1}(X_1, a_1 - 1)$ solves the graph with the final peg in $x_{1,i}$. So, the claim holds when $n = 1$.

Assume that for some $n \geq 1$ that $P_n(a_1, ..., a_n)$ is solvable with the initial hole in x_n provided that $a_i \geq 1$ for all i.

Now, consider the caterpillar $P_{n+1}(a_1, a_2, ..., a_n, a_{n+1})$ (where $a_i \geq 1$ for all i) and place the initial hole in x_{n+1}.

Case 1 a_{n+1} is even.

Treat x_n as a pendant of x_{n+1}. Use x_n (peg) and x_{n+1} (hole) as a catalyst for star purge $S_{x_{n+1}}(X_{n+1}, a_{n+1})$. We then jump $x_{n,a_n} \cdot \overrightarrow{x_n} \cdot x_{n+1}$. We now treat

x_{n+1} as a pendant of x_n. Ignoring the holes in x_{n,a_n} and X_{n+1}, the resulting graph is $P_n(a_1, ..., a_n)$ with a hole in x_n. So it is solvable by hypothesis.

Case 2 a_{n+1} is odd.

Apply the star purge $S_{x_{n+1}}(X_{n+1}, a_{n+1} - 1)$. Merge $(x_n, x_{n+1,1}) \to x_{n+1}$. Treat x_{n+1} as a pendant of x_n and ignore the holes in x_{n+1} and its pendants. The resulting graph is $P_n(a_1, ..., a_n + 1)$ with a hole in x_n, so it is solvable by hypothesis.

In either case, caterpillars of the form $P_n(a_1, ..., a_n)$, where $a_i \geq 1$ for all i are solvable in the jump+merge variant by the principle of mathematical induction. ∎

Note that Theorem 3.2.1 shows that an infinite class of caterpillars are solvable. However, it does not show that *all* caterpillars are solvable. In particular, this result does not cover caterpillars of the form $P_n(a_1, ..., a_n)$, where $a_i = 0$ for some $i \in \{2, ..., n-1\}$. Our next result seeks to alleviate this discrepancy by considering an infinite class of caterpillars of this form.

Theorem 3.2.2 *The caterpillar* $T = P_n(a_1, ..., a_p, 0, ..., 0, a_n)$ *where* $n \geq 1$, $p \geq 1$, *and* $a_i \geq 1$ *for* $i \in \{1, ..., p, n\}$ *is solvable in the jump+merge variant.*

Proof. Suppose that a_n is even. Begin with the initial hole in x_{n-1}. Using x_{n-1} (hole) and x_n (peg) as the catalyst for our star purge, we remove a_n pegs from the leaves of x_n. We then make merge moves $(x_{n-i+1}, x_{n-i-1}) \to x_{n-i}$ for $i = 1, ..., n-p-1$. We ignore the holes in x_{p+2},...,x_n, and holes in the pendants of x_n. Further, we treat x_{p+1} as one of the leaves of x_p. What remains is the caterpillar $P_p(a_1, ..., a_{p-1}, a_p + 1)$, where $a_i \geq 1$ for $i = 1, ..., p$. Since we have a hole in x_p, this is solvable by Theorem 3.2.1.

Suppose that a_n is odd. Begin with the initial hole in x_n. We use x_n (hole) and x_{n-1} (peg) as the catalyst for our star purge to remove all but one peg from the pendants of x_n. We then merge $(x_{n,1}, x_{n-1}) \to x_n$ followed by $(x_{n-i+1}, x_{n-i-1}) \to x_{n-i}$ for $i = 1, ..., n-p-1$. We ignore the holes in x_{p+2},...,x_n, and holes in the pendants of x_n. Further, we treat x_{p+1} as one of the leaves of x_p. What remains is the caterpillar $P_p(a_1, ..., a_{p-1}, a_p + 1)$, where $a_i \geq 1$ for $i = 1, ..., p$. Since we have a hole in x_p, this is solvable by Theorem 3.2.1. ∎

The following corollary follows immediately from Theorem 3.2.1 and Theorem 3.2.2. This corollary is especially important as it covers several special cases of caterpillars that were previously investigated in [8] and [22].

Corollary 3.2.3 *The following families of caterpillars are solvable in the jump+merge variant:*

(i) *Caterpillars with spine length three, i.e.,* $P_3(a_1, a_2, a_3)$

(ii) *Caterpillars with spine length four, i.e.,* $P_4(a_1, a_2, a_3, a_4)$*.*

(iii) *Caterpillars of the form* $P_n(a_1, 0, ..., 0, 1)$*, i.e., broomsticks.*

(iv) *Caterpillars of the form* $P_n(a_1, 0, ..., 0, a_n)$*, i.e., dumbbells.*

In the proof of Theorem 3.2.1, the parity of a_n determines how we deal with x_{n-1} and its pendants. Specifically, when a_n is even, then the star purge removes all of the pegs from the pendants of x_n. We then jump $x_{n-1}, a_{n-1} \cdot \overrightarrow{x_{n-1}} \cdot x_n$. We ignore $x_{n-1,a_{n-1}}$ and the pendants of x_n. We treat x_n as a pendant of x_{n-1}, allowing us to work with the caterpillar $P_{n-1}(a_1, ..., a_{n-1})$ with a hole in x_{n-1}. In short, the parity of a_{n-1} is not changed. Likewise, if a_n is odd, then the star purge removes all but one peg from the pendants of x_n. We then merge $(x_{n-1}, x_{n,1}) \to x_n$. Ignoring the pendants of x_n and treating x_n as a pendant of x_{n-1} we have the caterpillar $P_{n-1}(a_1, ..., a_{n-2}, a_{n-1}+1)$ with a hole in x_{n-1}. In short, the parity of a_{n-1} is changed. We are motivated by the above discussion to define the *adjusted values* a'_1,...,a'_n for the caterpillar as follows: $a'_n = a_n$ and for $i = 1, ..., n-1$,

$$a'_i = \begin{cases} a_i & a'_{i+1} \equiv 0 \pmod 2 \\ a_i + 1 & a'_{i+1} \equiv 1 \pmod 2. \end{cases}$$

Observe that when a'_1 is even, then our final peg is in x_1. Whereas if a'_1 is odd, then the final peg is in one of the neighbors of x_1. A more subtle observation is that we can change the location of final peg by changing the location of the initial hole. This observation is important because we want a hole in the final spine vertex in order to utilize our earlier result. We will go into more detail in the proof of our next theorem.

Theorem 3.2.4 *Let* p*,* q*, and* n *be positive integers such that* $p+q \leq n$*. The caterpillar* $T = P_n(a_1, ..., a_n)$*, where* $a_i \geq 1$ *for* $i \in \{1, ..., p, n-q+1, .., n\}$ *and* $a_i = 0$ *for* $p+1 \leq i \leq n-q$ *is solvable in the jump+merge variant.*

Proof. If $q = 1$, then the result is covered by Theorem 3.2.2. For this reason, we will assume that $q \geq 2$ for the rest of the proof. Compute the adjusted values for a_{n-q+1},...,a_n as described above.

If a'_{n-q+1} is odd, then begin with the initial hole in x_n. Consider the subgraph induced by x_{n-q+1},...,x_n and their pendants. Solve this $P_q(a_{n-q+1}, ..., a_n)$ subgraph as described in the proof of Theorem 3.2.1. Since a'_{n-q+1} is odd, this results in a hole in x_{n-q+1} and a peg in one of its neighbors, say x_{n-q+2}. We then make the merge moves $(x_{n-q+3-i}, x_{n-q+1-i}) \to x_{n-q+2-i}$ for $i = 1, ..., n-q-p+1$. We ignore the holes in x_{p+2},...,x_n and their pendants and treat x_{p+1} as a pendant of x_p. What remains is the caterpillar $P_p(a_1, ..., a_{p-1}, a_p + 1)$ with a hole in x_p. Thus, it is solvable by Theorem 3.2.1.

Suppose a'_{n-q+1} is even and that $a_n \geq 2$. Begin with the initial hole in x_{n,a_n-1} and jump $x_{n,a_n} \cdot \overrightarrow{x_n} \cdot x_{n,a_n-1}$. Ignoring the hole in x_{n,a_n}, our result is the caterpillar $P_n(a_1, ..., a_{n-1}, a_n - 1)$ with a hole in x_n. Since the parity of a'_n has changed, the parities of a'_{n-q+1},...,a'_{n-1} have each changed from their original values. Thus, this reduces to the case above.

Suppose a'_{n-q+1} is even and that $a_n = 1$. Begin with the initial hole in x_{n-1}. Notice that $q \geq 2$ implies that $a_{n-1} \geq 1$. Jump $x_{n,1} \cdot \overrightarrow{x_n} \cdot x_{n-1}$ and $x_{n-1,a_{n-1}} \cdot \overrightarrow{x_{n-1}} \cdot x_n$. Ignore the holes in $x_{n,1}$ and $x_{n-1,a_{n-1}}$ and treat x_n as a pendant of x_{n-1}. The resulting graph is $P_{n-1}(a_1, ..., a_{n-1})$ with a hole in x_{n-1}. Since a_n (and therefore a'_n) is odd, this removal changes the parity of a'_{n-1} from its initial value. Subsequently, the parities of a'_{n-q+1},...,a'_{n-2} have each changed from their original values. Thus, this reduces to the case where a'_{n-q+1} is odd. ■

Using a similar technique as was used in the proof of Theorem 3.2.4, we can find a bound on how many pegs are left on a general caterpillar. To aid in this, we define some additional terminology. With the exception of the end vertices of the spine x_1 and x_n, which we will discuss later, we place every vertex within either a dessert or a garden, Within the caterpillar, the *desert* D_i is the subgraph induced by x_{d_i},...,$x_{d_i+\ell_i}$, where $a_{d_i} = \cdots = a_{d_i+\ell_i} = 0$, $a_{d_i-1} \geq 1$, and $a_{d_i+\ell_i+1} \geq 1$. Likewise, the *garden* G_j is the subgraph induced by x_{g_j},...,$x_{g_j+\ell_j}$ and their pendants, where $a_{g_j+c-1} \geq 1$ for $c = 1, ..., \ell_j + 1$, $a_{g_j-1} = 0$, and $a_{g_j+\ell_j+1} = 0$. If $a_1 \geq 2$ ($a_n \geq 2$) or $a_2 \geq 2$ ($a_{n-1} \geq 2$), then we include x_1 (x_n) along with its pendants in the garden that contains x_1 (x_n). If $a_1 = 1$ ($a_n = 1$) and $a_2 = 0$ ($a_{n-1} = 0$), then we include x_1 (x_n) and its pendant along with the desert that contains x_2 (x_{n-1}).

In this way, we can describe the caterpillar as an alternating sequence of gardens and deserts. Without loss of generality, there are three possible patterns:

(i) $G_1, D_2, G_3, ..., D_{t-1}, G_t$;

$$\textit{(ii)}\ \ D_1, G_2, D_3, ..., D_{t-1}, G_t;$$

$$\textit{(iii)}\ \ D_1, G_2, D_3, ..., G_{t-1}, D_t.$$

We will assume that the spine vertices in the ith entry (regardless of whether it is a desert or a garden) are x_{i_1},...,$x_{i_{n_i}}$. As in the proofs of Theorem 3.2.1, Theorem 3.2.2, and Theorem 3.2.4, our solution will progress from right to left.

Observe what happens as we move from the desert D_{i+1} into the garden G_i. The final peg on D_{i+1} will be in $x_{(i+1)_2}$. To move into G_i, we merge $(x_{(i+1)_2}, x_{i_{n_i}}) \to x_{(i+1)_1}$. We then treat $x_{(i+1)_1}$ as a pendant of $x_{i_{n_i}}$. Since $x_{i_{n_i}}$ has a hole in it, we have effectively increased the value of $a_{i_{n_i}}$ by one. This suggests that our adjusted values for the parameters of the gardens (other than G_t, in cases (i) and (ii)) will be changed accordingly. As before, we define $a'_{t_{n_t}} = a_{t_{n_t}}$, if we are in case (i) or case (ii). Based on the above discussion, we define $a'_{i_{n_i}} = a_{i_{n_i}} + 1$ for G_i, $i \neq t$. For each garden G_i and $j = 1, ..., n_i - 1$ we define the other adjusted values in the same manner as above, namely

$$a'_{i_j} = \begin{cases} a_{i_j} & a'_{i_{j+1}} \equiv 0 \pmod 2 \\ a_{i_j} + 1 & a'_{i_{j+1}} \equiv 1 \pmod 2. \end{cases}$$

At this point, it behooves us to discuss how we move from the garden G_{i+1} to the desert D_i. On G_{i+1}, we want a hole in $x_{(i+1)_1}$ and a peg in one of its neighbors, say $x_{(i+1)_2}$. This allows us to traverse the desert D_i using the merge moves $(x_{(i+1)_2}, x_{i_{n_i}}) \to x_{(i+1)_1}$, and $(x_{i_{n_i-1}}, x_{(i+1)_1}) \to x_{i_{n_i}}$ followed by $(x_{i_{n_i-j-1}}, x_{i_{n_i-j+1}}) \to x_{i_{n_i-j}}$ for $j = 1, ..., n_{i-2}$. As in the proof of Theorem 3.2.4 and in cases (i) and (ii), we can ensure that the final peg on G_t will be in x_{t_2} by choice of our initial hole. However, we do not have this luxury for subsequent gardens. If a'_{i_1} is odd, then the star purges will leave a single peg in x_{i_2} in G_i. We can then progress down the caterpillar as described above. However, if a'_{i_1} is even for $2 \le i \le t-1$, then our star purges will result in a peg in each of x_{i_2} and $x_{i_1,1}$ on G_i. In order to continue, we choose to leave the peg in $x_{i_1,1}$ and then progress with our solution as we have described. Observe that in case (i), if a'_{1_1} is even, then we can merge the final two pegs in G_1. Thus, in order to determine a bound on the solvability of the caterpillar, we need only know which G_i have even a'_{i_1}. For $2 \le i \le t-1$, we say that G_i is an *even garden* if the adjusted value a'_{i_1} is even. Using the method described above, we will leave one peg in each even garden as well as a final peg in either G_1 or D_1. Based on this, the following theorem is immediate.

Theorem 3.2.5 *Given the caterpillar $P_n(a_1, ..., a_n)$, at most we will leave $1 + m$ pegs on this graph, where m is the number of even gardens.*

There are various ways that the bound in Theorem 3.2.5 can be improved. For example, suppose that we have an even garden G_i such that for $x_{i_j}, x_{i_{j+1}}, x_{i_{j+2}} \in V(G_i)$ the adjusted values satisfy $a'_{i_j} \geq 3$ and $a'_{i_{j+2}} \geq 2$. In this case, we solve the garden G_i as described above until there is a hole in $x_{i_{j+2}}$. At this point, we jump $x_{i_j} \cdot \overrightarrow{x_{i_{j+1}}} \cdot x_{i_{j+2}}$, jump $x_{i_{j+2}, a_{i_{j+2}}} \cdot \overrightarrow{x_{i_{j+2}}} \cdot x_{i_{j+1}}$, and merge $(x_{i_j, a_{i_j}-1}, x_{i_j, a_{i_j}}) \to x_{i_j}$. We then ignore the holes in $x_{i_j, a_{i_j}-1}$, $x_{i_j, a_{i_j}}$, and $x_{i_{j+2}, a_{i_{j+2}}}$. Notice that this changes the parity of $a_{i_{j+2}}$ while leaving the parity of a_{i_j} and $a_{i_{j+1}}$ alone (however, the actual value of a_{i_j} has changed). Thus, the adjusted value of $a'_{i_{j+2}}$ has changed, thereby changing the subsequent adjusted values. This allows us to solve the garden G_i, leaving no pegs behind.

No doubt, there are similar methods that will also allow for minor improvements on Theorem 3.2.5. However, we have no desire to catalog additional methods at this time.

3.3 Trees of Small Diameter

As mentioned previously, trees can take many different named or unnamed forms. One way to discuss unnamed forms of trees is by their diameter, since this gives us an idea to their size and shape, even though a tree of a certain diameter may or may not be an explicit named form. Note that all trees of diameter two are stars and all trees of diameter three are double stars.

We are motivated by the above comments to determine the solvability of all trees of diameter four. Any tree of diameter four can be obtained by appending pendant vertices to the existing vertices of the star $K_{1,n}$. Label the center of the star as x and its arms as $y_1, ..., y_n$. Suppose that we append c pendant vertices to x, namely $x_1, ..., x_c$ and a_i pendant vertices to y_i, namely $y_{i,1}, ..., y_{i,a_i}$ for $i = 1, ..., n$. Note that for $i \neq j$ and for any ℓ and m, the vertices $y_{i,\ell}$, y_i, x, y_j, and $y_{j,m}$ induce a path of length four. Thus, this construction gives all trees of diameter four. The resulting graph will be denoted $K_{1,n}(c; a_1, ..., a_n)$. An example is shown in Figure 3.3.

The necessary and sufficient conditions for the solvability of trees of diameter four are given in a 2015 paper by Beeler and Walvoort [13]. Because their results are pertinent to our discussion, we include them here. Note that

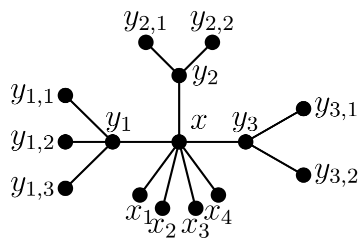

Figure 3.3: The graph $K_{1,3}(4; 3, 2, 2)$

Beeler and Walvoort introduce a new parameter $k = c - \Sigma a_i + n$ to facilitate their discussion.

Theorem 3.3.1 *[13] The conditions for solvability of $K_{1,n}(c; a_1, ..., a_n)$ where $a_1 \geq 2$ are as follows:*

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

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

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

Theorem 3.3.2 *[13]*

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

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

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

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

As we begin to study more complex graphs such as trees of small diameter, we will want to take advantage of the packages and purges introduced in Beeler and Walvoort's paper on trees of diameter four. These purges include the wishbone, the trident, the double star, and two variations of a spider purge.

The *wishbone* package consists of $K_{1,2}(1;1,1)$ with the hole in x. The wishbone purge is $y_{2,1}\cdot\overrightarrow{y_2}\cdot x$, $x_1\cdot\overrightarrow{x}\cdot y_2$, $y_{1,1}\cdot\overrightarrow{y_1}\cdot x$, and $y_2\cdot\overrightarrow{x}\cdot x_1$. The wishbone purge removes two legs, while the catalyst is x (hole) and x_1 (peg).

The *trident* package consists of $K_{1,3}(1;1,1,1)$ with the hole in x and pegs elsewhere. The trident purge is $y_{3,1}\cdot\overrightarrow{y_3}\cdot x$, $x_1\cdot\overrightarrow{x}\cdot y_3$, $y_{2,1}\cdot\overrightarrow{y_2}\cdot x$, $y_3\cdot\overrightarrow{x}\cdot y_2$, $y_{1,1}\cdot\overrightarrow{y_1}\cdot x$, and $y_2\cdot\overrightarrow{x}\cdot x_1$. This purge removes three legs, while the catalyst is x (hole) and x_1 (peg).

The *double star* package will consist of a $S_{d,d}$ with a hole in x. The associated purge to remove d pendants from each side of the double star is accomplished with the moves $y_{1,i}\cdot\overrightarrow{y_1}\cdot x$ and $x_i\cdot\overrightarrow{x}\cdot y_1$ for $i = 1, ..., d$ [10]. This purge is denoted $\mathscr{DS}(Y_1, X, d)$. The catalyst is x (hole) and y_1 (peg).

The *spider(N)* package consists of a $K_{1,3}(2;1,1,1)$ with the hole in y_1 and pegs elsewhere. The spider(N) purge is $x_1\cdot\overrightarrow{x}\cdot y_1$, $y_{1,1}\cdot\overrightarrow{y_1}\cdot x$, $x_2\cdot\overrightarrow{x}\cdot y_1$, $y_{2,1}\cdot\overrightarrow{y_2}\cdot x$, $x\cdot\overrightarrow{y_1}\cdot y_{1,1}$, and $y_{3,1}\cdot\overrightarrow{y_3}\cdot x$. Note that the spider($N$) purge removes two pegs from X and two legs, while y_1 (hole), x (peg) and $y_{1,1}$ (peg) are the catalyst.

The *spider(x)* package has the hole in x. The associated purge is $y_{1,1}\cdot\overrightarrow{y_1}\cdot x$, $x_1\cdot\overrightarrow{x}\cdot y_1$, $y_{2,1}\cdot\overrightarrow{y_2}\cdot x$, $x\cdot\overrightarrow{y_1}\cdot y_{1,1}$, $y_{3,1}\cdot\overrightarrow{y_3}\cdot x$, and $x_2\cdot\overrightarrow{x}\cdot y_1$. The spider($x$) purge removes two pegs from X and two legs, while x (hole), y_1 (peg), and $y_{1,1}$ (peg) are the catalyst.

Illustrations of the graphs used for the wishbone, trident, and spider purges are given in Figure 3.4. For each graph, the catalyst is placed in a box. The hollow vertex in the box represents the initial hole for the purges.

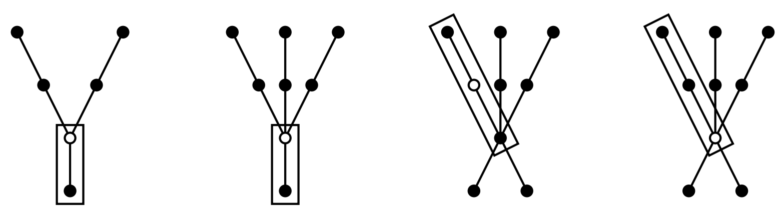

Figure 3.4: The graphs for the wishbone, trident, and spider(N), and spider(x) purges

We are now prepared to show that all trees of diameter four are solvable in the jump+merge variant. Throughout this theorem, we will define several versions of a set S. We perform star purges on this set to complete the solution.

Theorem 3.3.3 *Let $T = K_{1,n}(c; a_1, ..., a_n)$ be a tree of diameter four. Then T is solvable in the jump+merge variant of peg solitaire.*

Proof. Note that $K_{1,2}(0; 1, 1)$ is isomorphic to the path on five vertices. As the solvability of all paths was established in Theorem 2.2.1, we will assume that $T \neq P_5$ for the remainder of the proof.

Begin with the initial hole in x. For $i = 1, ..., n$ perform star purges with catalyst x (hole) and y_i (peg) to remove $2\lfloor a_i/2 \rfloor$ pegs from the pendants of y_i. Note that if a_i is even, then we have removed all of the pegs from $y_{i,1}$,...,y_{i,a_i}. Likewise if a_i is odd, then one peg remains in $y_{i,1}$. Let ℓ be the number of $y_{i,1}$ that still have pegs. If $\ell = 0$, then define the set $S = \{y_1, ..., y_n, x_1, ..., x_c\}$. If $\ell \geq 1$, then reorder the y_i so that there are pegs in $y_{1,1}$,...,$y_{\ell,1}$.

If $\ell = 1$, then jump $y_{1,1} \cdot \overrightarrow{y_1} \cdot x$ and $y_n \cdot \overrightarrow{x} \cdot y_1$. We define the set $S = \{y_1, ..., y_{n-1}, x_1, ..., x_c\}$.

Suppose that $2 \leq \ell \leq n - 1$. Perform trident and wishbone purges with catalyst x (hole) and y_n (peg) to remove all pegs from y_i and $y_{i,1}$ for $i = 1, ..., \ell$. Define the set $S = \{y_{\ell+1}, ..., y_n, x_1, ..., x_c\}$.

Similarly, when $\ell = n$ and $n \geq 3$ we perform trident and wishbone purges with catalyst x (hole) and y_n (peg) to remove all pegs from y_i and $y_{i,1}$ for $i = 1, ..., n-1$. We then jump $y_{n,1} \cdot \overrightarrow{y_n} \cdot x$. If $c = 0$, then the graph is solved with the final peg in x. Otherwise, jump $x_c \cdot \overrightarrow{x} \cdot y_1$ and define $S = \{y_1, x_1, ..., x_{c-1}\}$.

Finally, suppose that $\ell = n$ and $n = 2$. If $c \geq 1$, then we perform a wishbone purge with catalyst x (hole) and x_1 (peg) to remove the pegs from y_1, $y_{1,1}$, y_2, and $y_{2,1}$. We define $S = \{x_1, ..., x_c\}$. If $c = 0$, then either $a_1 \geq 2$ or $a_2 \geq 2$ since $T \neq P_5$. Assume without loss of generality that $a_1 \geq 2$. This means that there are pegs in y_1, $y_{1,1}$, y_2, and $y_{2,1}$ and holes in $y_{1,2}$ and x. To complete the solution, we jump $y_{1,1} \cdot \overrightarrow{y_1} \cdot y_{1,2}$, jump $y_{2,1} \cdot \overrightarrow{y_2} \cdot x$, and merge $(y_{1,2}, x) \to y_1$, completing the solution.

If S is undefined or $|S| = 1$, then the graph is solved. Otherwise, let $w \in S$. Perform star purges on the set $S - \{w\}$ with catalyst x (hole) and w (peg) to remove $2\lfloor |S - \{w\}|/2 \rfloor$ pegs from $S - \{w\}$. If $|S|$ is odd, then we have solved the graph with the final peg in w. If $|S|$ is even, then there is a peg in w and a peg in some element of $S - \{w\}$, say z. In this case, we complete the solution with the merge $(w, z) \to x$.

■

Using our result from Theorem 3.3.3, it is straightforward to show that

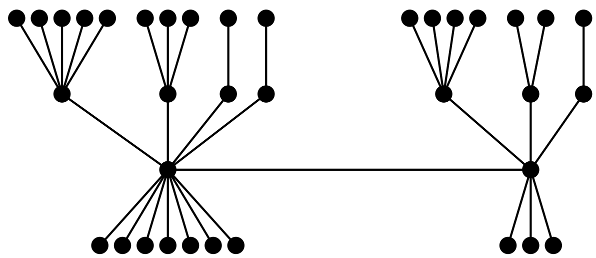

Figure 3.5: The graph $S_{4,3}(7; 5, 3, 1, 1, ; 3; 4, 2, 1)$

trees of diameter five are also solvable. Any tree of diameter five can be obtained by appending leaves to the existing vertices of the double star $S_{r,s}$. We append c_1 leaves to x, namely w_1,...,w_{c_1}. We append c_2 leaves to y, namely z_1,...,z_{c_2}. Similarly, we append a_i leaves to x_i, namely $x_{i,1}$,...,x_{i,a_i}, and b_j leaves to y_j, namely $y_{j,1}$,...,y_{j,b_j}. A diameter five tree with these parameters is denoted $S_{r,s}(c_1; a_1, ..., a_r; c_2; b_1, ..., b_s)$ (see Figure 3.5). Without loss of generality, assume that $a_1 \geq ... \geq a_r \geq 1$ and $b_1 \geq .... \geq b_s \geq 1$. Note that if $r = s = 1$, then this graph is isomorphic to the caterpillar $P_4(a_1, c_1, c_2, b_1)$. As these graphs were shown to be solvable in Theorem 3.2.3, we can assume that $r \geq 2$.

For purposes of exhibition, we define two subsets of the vertex set

$$L = \{x, y, x_1, ..., x_r\} \cup \{w_1, ..., w_{c_1}\} \cup \{x_{1,1}, ..., x_{1,a_1}\} \cup \cdots \cup \{x_{r,1}, ..., x_{r,a_r}\} \quad \text{and}$$

$$R = \{x, y, y_1, ..., y_s\} \cup \{z_1, ..., z_{c_2}\} \cup \{y_{1,1}, ..., y_{1,b_1}\} \cup \cdots \cup \{y_{s,1}, ..., x_{s,b_s}\}.$$

Note that the subgraph induced by L is isomorphic to the tree of diameter four $K_{1,r}(c_1 + 1; a_1, ..., a_r)$. Similarly, when $s = 1$, the subgraph induced by R is isomorphic to the double star S_{c_2+1,b_1}. When $s \geq 2$, the subgraph induced by R is isomorphic to the tree of diameter four $K_{1,s}(c_2 + 1; b_1, ..., b_s)$. We will use our results regarding the solvability of stars, double stars, and trees of diameter four to yield the analogous result on trees of diameter five. This result is especially significant as the solvability of such trees has not been determined in either the (traditional) jump variant or the merge variant of peg solitaire.

Theorem 3.3.4 *The tree of diameter five $S_{r,s}(c_1; a_1, ..., a_r; c_2; b_1, ..., b_s)$ is solvable when merge moves and jump moves are allowed.*

Proof. Note that if $r = s = 1$, then this graph is isomorphic to the caterpillar $P_4(a_1, c_1, c_2, b_1)$. As these graphs were shown to be solvable in Theorem 3.2.3, we will assume for the rest of the proof that $r \geq 2$.

Begin with the initial hole in y and solve G_R as described in Theorem 3.1.2 (when $s = 1$) or Theorem 3.3.3 (when $s \geq 2$). In either case, we can assume the final peg on G_R will be in either y or in y_1. Observe that there is a hole in x.

Case 1 Suppose that the final peg on G_R is in y. Treat y as a pendant of x. As G_L has a hole in x, we solve it as we would $K_{1,r}(c_1 + 1; a_1, ..., a_r)$ (see Theorem 3.3.3).

Case 2 Suppose that the final peg on G_R is in y_1.

If $a_r = 1$, then we jump $x_{r,1} \cdot \overrightarrow{x_r} \cdot x$ and merge $(y_1, x) \to y$. We ignore the holes in x_r and $x_{r,1}$ and treat y as a pendant of x. If $r = 2$, then we solve G_L as we would S_{c_1+1,a_1} with a hole in x (see Theorem 3.1.2). If $r \geq 3$, then we solve G_L as $K_{1,r-1}(c_1 + 1; a_1, ..., a_{r-1})$ with a hole in x (see Theorem 3.3.3).

If $a_1 \geq 3$ and $a_r \geq 2$, then jump $x_{1,a_1} \cdot \overrightarrow{x_1} \cdot x$, merge $(y_1, x) \to y$, and merge $(x_{1,a_1-2}, x_{1,a_1-1}) \to x_1$. We now ignore the holes in x_{1,a_1-2}, x_{1,a_1-1}, and x_{1,a_1} and treat y as a pendant of x. If $a_1 = 3$ and $r = 2$, then we also treat x_1 as a pendant of x and we solve G_L as we would S_{c_1+2,a_2} with a hole in x (see Theorem 3.1.2). If $a_1 = 3$ and $r \geq 3$, then we solve G_L as we would $K_{1,r-1}(c_2 + 2; a_2, ..., a_r)$ with a hole in x (see Theorem 3.3.3). If $a_1 \geq 4$, then we solve G_L as we would $K_{1,r}(c_1 + 1; a_1 - 3, a_2, ..., a_r)$.

Finally, suppose that $a_i = 2$ for $i = 1, ..., r$. We make the following merges $(x_{r-1}, x_r) \to x$, $(y_1, x) \to y$, $(x_{r-1,1}, x_{r-1,2}) \to x_{r-1}$, and $(x_{r,1}, x_{r,2}) \to x_r$. We ignore the holes in $x_{r-1,1}$, $x_{r-1,2}$, $x_{r,1}$, and $x_{r,2}$ and treat x_{r-1}, x_r, and y as pendants of x. If $r = 2$, then we solve G_L as we would K_{1,c_1+3} with a hole in x (see Theorem 3.1.1). If $r = 3$, then we solve G_L as we would S_{c_1+3,a_1} with a hole in x (see Theorem 3.1.2). If $r \geq 4$, then we solve G_L as we would $K_{1,r-2}(c_1 + 3; a_1, ..., a_{r-2})$ (see Theorem 3.3.3).

■

A natural next step may be to consider trees of diameter six. Any tree of diameter six can be obtained by adding pendant vertices to the vertices of a tree of diameter four. Given the notation for a tree of diameter four, it is likely that any notation used for a general tree of diameter six would be complicated and cumbersome. However, it may be interesting and insightful to consider certain classes of trees of diameter six. One such class could be the banana trees studied by de Wiljes and Kreh [20]. However, we leave such considerations as a problem for future research.

3.4 Other Trees

We have explored trees as a graph family that has much to offer to peg solitaire on graphs. We have mentioned that this is due to the fact that every connected graph contains a spanning tree. A great resource that explores these spanning trees on n vertices is Steinbach's *Field Guide to Simple Graphs* [29]. In this book, illustrations are provided for each non-isomorphic tree on n vertices where $n \leq 12$, and we see that there is one on one vertex, one on two vertices, one on three vertices, two on four vertices, three on five vertices, and so on. Using the results that we have obtained thus far and comparing them to the trees in [29], we notice that all trees on eight vertices or less are solvable using the jump and the merge. There are two non-isomorphic trees on nine vertices that we have not shown to be solvable (there are eleven on ten vertices and more than fifty left unsolved on eleven vertices). Upon further study, we notice that there are some commonalities among some of the trees on $n \leq 11$ vertices. One in particular is a caterpillar-like graph that we will now explore.

Consider a graph that looks like a caterpillar with a path as a spine and pendants along the spinal vertices. However, we now allow these pendants to be subdivided so that it looks like a tail with two adjacent vertices rather than just one. We will refer to this caterpillar-like graph as an *articulated caterpillar* and denote it $AP_n(b_1, ..., b_n)$, where b_i is the number of tails on x_i (such graphs are sometimes referred to as *lobster graphs* in the literature). An example of such a graph is given in Figure 3.6. For clarity, we assume that $n \geq 2$ and $b_1 = b_n = 0$. Otherwise, the graph could be transformed into various interpretations via various arrangements of the tails of the endpoints. For example, an articulated caterpillar with $b_1 = 1$ could be interpreted as an articulated caterpillar with no tails on the first three vertices.

Theorem 3.4.1 *The articulated caterpillar, $AP_n(b_1, ..., b_n)$, where $n \geq 2$, $b_1 = b_n = 0$, and $b_i \leq 1$ for all i is solvable in the jump+merge variant.*

Proof. Let t be the number of tails on $AP_n(b_1, ..., b_n)$. We proceed by induction on t.

Let $t = 1$. Then, by the definition of an articulated caterpillar, there are at least three spinal vertices, say $v_1, ..., v_n$, where v_i is the vertex with the

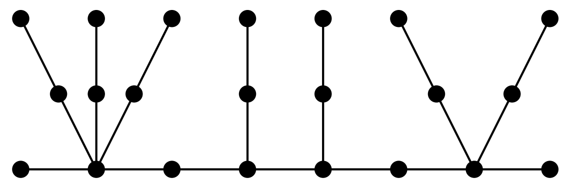

Figure 3.6: The articulated caterpillar $AP_8(0,3,0,1,1,0,2,0)$

tail. Note that $2 \leq i \leq n-1$. Label the vertices of the tail $v_{i,1}$ and $v_{i,2}$. Let the initial hole be in the peg adjacent to the end of the spine, say v_2. Treat the vertices along the spine from $v_1, ..., v_i, v_{i,1}$ as a path and merge along this path. Now, $v_1, ..., v_{i-1}$ are empty, v_i and $v_{i,2}$ have pegs, and $v_{i,1}$ is empty. Now, treat $v_{i,2}, v_{i,1}, v_i, v_{i+1}, ...v_n$ as a path. Note, the hole is still in the vertex adjacent to the end of the path, so it can be solved using merges. So, the claim holds when $t = 1$.

Assume that for some $t \geq 1$ that the articulated caterpillar, $AP_n(b_1, ..., b_n)$, where $n \geq 2$ and $b_1 = b_n = 0$, with t tails is solvable in the jump+merge variant when $b_i \leq 1$.

Now, consider the articulated caterpillar, $AP_n(b_1, ..., b_n)$, where $n \geq 2$, $b_1 = b_n = 0$, and $b_i \leq 1$ for all i with $t+1$ tails. Label the spinal vertices that have tails as $v_{i_1}, v_{i_2}, ..., v_{i_t}, v_{i_{t+1}}$, where $1 < i_1 < i_2 < ... < i_{t+1} < n$. Label the vertices of the i_jth tail as $v_{i_j,1}$ and $v_{i_j,2}$. Place the hole in the vertex adjacent to the end of the spine, say v_2. Treat the vertices $v_1, ..., v_{i_1}, v_{i_1,1}$ as a path and merge along this path so that all vertices to the left of v_{i_1} are empty. We ignore v_1,...,v_{i_1-1} and treat $v_{i_1,1}$ and $v_{i_1,2}$ as spinal vertices on an articulated caterpillar. The resulting graph is $AP_{n-i_1+3}(0, 0, 0, b_{i_1+1}, ..., b_n)$ with a hole in the second spinal vertex, namely $v_{i_1,1}$. As this is an articulated caterpillar with t tails, it is solvable by hypothesis.

Thus, the articulated caterpillar, $AP_n(b_1, b_2, ..., b_n)$, where $n \geq 2$, $b_1 = b_n = 0$, and $b_i \leq 1$ for all i is solvable in the jump+merge variant.

■

We are interested to see what happens to the solvability of $AP_n(b_1, ..., b_n)$ when b_i is unrestricted.

Theorem 3.4.2 *The articulated caterpillar, $AP_n(b_1, , ..., b_n)$, where $n \geq 2$*

and $b_1 = b_n = 0$*, is solvable in the jump+merge variant.*

Proof. As before, we label the spinal vertices as $v_1, ..., v_n$. There is now a set of spinal vertices v_i such that $b_i \geq 1$. We label them $v_{t_1}, v_{t_2}, ...v_{t_j}$. Notice that if $b_i \leq 1$ for all i, then this graph is solvable by Theorem 3.4.1. Thus, we will assume that at least one of the b_{t_k} is at least two. We proceed by induction on j.

If $j = 1$, then let the initial hole be in the spinal vertex adjacent to the end of the spine, say v_2. Perform the merge moves $(v_i, v_{i+2}) \to v_{i+1}$ for $i = 1, ..., t_1 - 2$. Since $b_{t_1} \geq 2$, then we let v_{t_1-1} (peg) and v_{t_1} (hole) act as the catalyst for wishbone and trident purges (see Figure 3.4) to remove the tails of v_{t_1}. We complete our solution with the merges $(v_{t_1+i-2}, v_{t_1+i}) \to v_{t_1+i-1}$ for $i = 1, ..., n - t_1$.

Assume that for some $j \geq 1$ that the articulated caterpillar $AP_n(b_1, ..., b_n)$ with $n \geq 2$, $b_1 = b_n = 0$, and $b_{t_i} \geq 1$ for $i = 1, ..., j$ is solvable.

Consider the articulated caterpillar $AP_n(b_1, ..., b_n)$ with $n \geq 2$, $b_1 = b_n = 0$, and $b_{t_i} \geq 1$ for $i = 1, ..., j + 1$. Begin with the initial hole in v_2. For $i = 1, ..., t_1 - 2$ perform the merge $(v_i, v_{i+2}) \to v_{i+1}$. There are now holes in v_1,...,v_{t_1-2}, v_{t_1} and pegs elsewhere. We now need to consider cases for b_{t_1}.

Case 1 If $b_{t_1} = 1$, then we make the merge $(v_{t_1-1}, v_{t_1,1}) \to v_{t_1}$. We now treat $v_{t_1,1}$ and $v_{t_1,2}$ as spinal vertices on the articulated caterpillar $AP_{n-i_1+3}(0, 0, 0, b_{i_1+1}, ..., b_n)$, where $b_n = 0$, $b_{t_i} \geq 1$ for $i = 1, ..., j$, and the hole is in the second spinal vertex, namely $v_{t_1,1}$. Hence it is solvable by hypothesis.

Case 2 If $b_{t_1} \geq 2$, then we let v_{t_1-1} (peg) and v_{t_1} (hole) act as the catalyst for a combination of wishbone and or trident purges (see Figure 3.4) to remove the pegs from the tails of v_{t_1}. Ignoring the tails of v_{t_1} as well as v_1,...,v_{t_1-2}, we are left with the graph $AP_{n-t_1+1}(0, 0, b_{t_1+1}, ..., b_n)$, where $b_n = 0$ and there is a hole in the second spinal vertex, namely v_{t_1}. Hence it is solvable by hypothesis. ■

At this point, we have only discussed caterpillars and articulated caterpillars separately. Solving a caterpillar with both pendants and tails is a problem that we will leave to future research. For now, we can expand our results by the idea of overlap, which is discussed in [7]. In this way, we can take a solvable caterpillar and overlap it with an articulated caterpillar of the same spinal length to obtain a solvable graph. Though this certainly leaves many graphs unclassified as solvable or unsolvable, this is useful in

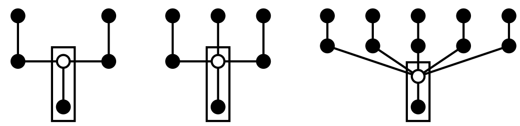

Figure 3.7: P_1, P_2, and $O(P_1, P_2)$

generating solvable graphs from the results that we have obtained.

We can also apply this idea to purges using a technique from [12]. Suppose that P is a package with catalyst $C(P) = (S(P), T(P))$, where $S(P)$ ($T(P)$) is the set of vertices with holes (pegs) in them. It is possible to construct additional purges using overlaps. Suppose that P_1 and P_2 are two packages such that $|S(P_1)| = |S(P_2)|$ and $|T(P_1)| = |T(P_2)|$. Define the overlap of P_1 and P_2 induced by their catalysts to be the graph obtained by identifying $S(P_1) = S(P_2)$ and $T(P_1) = T(P_2)$. This will be denoted $O(P_1, P_2)$ as in Figure 3.7.

Theorem 3.4.3 *[12] Suppose that P_1 and P_2 are packages with catalysts $(S(P_1), T(P_1))$ and $(S(P_2), T(P_2))$, respectively, such that $|S(P_1)| = |S(P_2)|$ and $|T(P_1)| = |T(P_2)|$. The overlap of P_1 and P_2 induced by their catalysts, $P_3 = O(P_1, P_2)$ is a package with catalyst $(S(P_3), T(P_3))$, where $S(P_1) = S(P_2) = S(P_3)$ and $T(P_1) = T(P_2) = T(P_3)$.*

We can also construct new purges by examining the moves within the purge. Namely, if we have a package P_1, then we can perform the moves of its purge. Whenever we have a subgraph with the configuration of the catalyst of a second purge P_2, then we can overlap them as described above. Further details to this process are given in [12].

That is, if we have a large graph with multiple packages as subgraphs, we can overlap them and use multiple purges to reduce the graph. In this way, larger portions of the tree will be emptied quickly to reduce a tree to one that is solvable. Though this is not possible with all trees, this form of overlap is a useful technique to identify trees of diameter greater than five that are also solvable and to generate trees of larger diameter that are also solvable.

Chapter 4

OPEN PROBLEMS AND CONCLUDING REMARKS

This research project has laid a good foundation for some families of trees that are solvable - and even freely solvable in the cases of stars and double stars - but is far from complete. Even with as much research that has been conducted on peg solitaire on graphs, there are still several open problems and variations that deserve future attention that have been mentioned by researchers, or that have been alluded to within this thesis. Some examples of these are finding necessary and sufficient conditions for any arbitrary graph to be solvable, trees of larger diameter, and variations of peg solitaire using new conditions or combinations of moves.

One type of tree that we did not discuss in this project is asters. An *aster* (or *star-like graph*) is obtained from the star $K_{1,n}$ by replacing the ith pendant with a path on p_i edges for $i = 1, ..., n$. Such an aster is denoted $A_n(p_1, ..., p_m)$ (see Figure 4.1). This graph is essentially a star with paths adjacent to the center c rather than single pendants. While we suspect that asters are unsolvable due to the nature of stranding pegs along each path, we leave this problem open for future research. An extension of this idea would be an aster-caterpillar hybrid with only asters, or even with a combination of pendants, tails, and paths adjacent to the spinal vertices. If some asters were proven solvable, then this idea could lead to fascinating results as bounds are determined.

An interesting idea that would serve as a good foundation for future research is the following: Consider peg solitaire played on a solvable graph G. The player is allowed j jumps and m merges, where $j + m = |V(G)| - 1$.

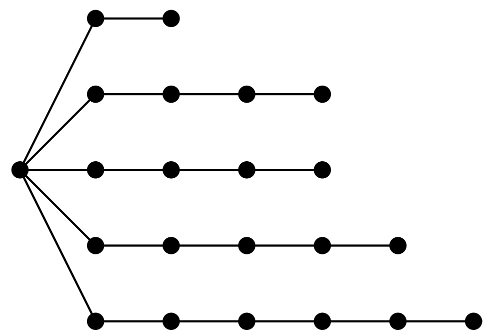

Figure 4.1: The aster $A_5(6, 5, 4, 4, 2)$

For what values of j and m does the graph G remain solvable? Another such problem was presented by Beeler and Hoilman in 2012 [10]: Namely, what role, if any, does the maximum degree of a tree (or in general, a graph) have in its solvability?

One of the most studied variations of peg solitaire is the *fool's solitaire problem* [11, 26]. Namely, what is the maximum number of pegs that can be left under the caveat that the player must make a move whenever possible. Obviously, we could study the fool's solitaire number of graphs when both merging and jumping are allowed.

Beeler and Gray also studied a variation of peg solitaire on graphs known as duotaire [5]. Duotaire on traditional game boards was also studied in [24, 27]. Typically, the first player selects a location for the initial hole in duotaire. The players the alternate making legal moves on the graph. When a player can not make a move, then they lose. There is also a version of duotaire on graphs where one player aims to leave the maximum number of pegs at the end while the other player seeks to leave the minimum number of pegs at the end. When both players play optimally, then the resulting number of pegs is a *competitive graph parameter*. For more information on competitive graph parameters, see [28] among others.

Another aspect of peg solitaire on graphs that deserves more research is exploring the impact of the initial hole. Engbers and Weber begin to explore this concept with merging peg solitaire on graphs. As in jump-only peg solitaire, they define a graph as freely solvable if the starting point of the empty vertex does not change the fact that the graph is solvable [22]. While there are infinite families of graphs that have previously been proven freely solvable, there are many more families to be investigated.

There is also little research that has been conducted exploring the combination of different moves in the game, which is the inspiration for this research project. While we have explored the combination of jumps and merges, there is more that can be explored in this variant, as well as other combinations with new, more complex moves. For example, there is likely more that can be determined involving trees that this project did not allow the time for. While we mentioned the possibility of overlap, allowing any arbitrary combination of pendants and tails or asters, pendants, and tails could yield some challenging yet interesting results that we will leave for future research.

Bibliography

[1] Handan Akyar, Nazlican Çakmak, Nilay Torun, and Emrah Akyar. Peg solitaire game on Sierpinski graphs. *Journal of Discrete Mathematical Sciences and Cryptography*, 0(0):1–10, 2021.

[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 and Aaron D. Gray. Peg solitaire on graphs with seven vertices or less. *Congr. Numer.*, 211:151–159, 2012.

[4] Robert A. Beeler and Aaron D. Gray. Extremal results for peg solitaire on graphs. *Bull. Inst. Combin. Appl.*, 77:30–42, 2016.

[5] Robert A. Beeler and Aaron D. Gray. An introduction to peg duotaire on graphs. *J. Combin. Math. Combin. Comput.*, 104:171–186, 2018.

[6] Robert A. Beeler and Aaron D. Gray. Double jump peg solitaire on graphs. *The Bulletin of the ICA*, 91:80–93, 2021.

[7] Robert A. Beeler, Aaron D. Gray, and D. Paul Hoilman. Constructing solvable graphs in peg solitaire. *The Bulletin of the ICA*, 66:89–96, 2012.

[8] Robert A. Beeler, Hannah Green, and Russell T. Harper. Peg solitaire on caterpillars. *Integers*, 17:Paper No. G1, 14, 2017.

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

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

[11] Robert A. Beeler and Tony K. Rodriguez. Fool's solitaire on graphs. *Involve*, 5(4):473–480, 2012.

[12] Robert A. Beeler and Clayton A. Walvoort. Packages and purges for peg solitaire on graphs. *Congr. Numer.*, 218:33–42, 2013.

[13] Robert A. Beeler and Clayton A. Walvoort. Peg solitaire on trees with diameter four. *Australas. J. Combin.*, 63:321–332, 2015.

[14] Christopher Belford and Nándor Sieben. Rubbling and optimal rubbling of graphs. *Discrete Math.*, 309(10):3436–3446, 2009.

[15] 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.

[16] Fred Buckley and Marty Lewinter. *Introductory Graph Theory with Applications*. Waveland Press, 2003.

[17] Grady D. Bullington. Peg solitaire: "burn two bridges, build one". *Congr. Numer.*, 223:187–191, 2015.

[18] Tara C. Davis, Alexxis De Lamere, Gustavo Sopena, Roberto C. Soto, Sonali Vyas, and Melissa Wong. Peg solitaire in three colors on graphs. *Involve*, 13(5):791–802, 2020.

[19] Jan-Hendrik de Wiljes and Martin Kreh. Stick solitaire on graphs. Submitted August 2020.

[20] Jan-Hendrik de Wiljes and Martin Kreh. Peg solitaire on banana trees. *Bull. Inst. Combin. Appl.*, 90:63–86, 2020.

[21] John Engbers and Christopher Stocker. Reversible peg solitaire on graphs. *Discrete Math.*, 338(11):2014–2019, 2015.

[22] John Engbers and Ryan Weber. Merging peg solitaire on graphs. *Involve*, 11(1):53–66, 2018.

[23] Rudolf Fleischer and Jiajin Yu. A survey of the game "Lights Out!". In *Space-efficient data structures, streams, and algorithms*, volume 8066 of *Lecture Notes in Comput. Sci.*, pages 176–198. Springer, Heidelberg, 2013.

[24] J. P. Grossman. Periodicity in one-dimensional peg duotaire. *Theoret. Comput. Sci.*, 313(3):417–425, 2004. Algorithmic combinatorial game theory.

[25] Martin Kreh and Jan-Hendrik de Wiljes. Peg solitaire on Cartesian products of graphs. *Graphs Combin.*, pages 1–11, 2021.

[26] Sarah Loeb and Jennifer Wise. Fool's solitaire on joins and Cartesian products of graphs. *Discrete Math.*, 338(3):66–71, 2015.

[27] Cristopher Moore and David Eppstein. One-dimensional peg solitaire, and duotaire. In *More games of no chance (Berkeley, CA, 2000)*, volume 42 of *Math. Sci. Res. Inst. Publ.*, pages 341–350. Cambridge Univ. Press, Cambridge, 2002.

[28] James B. Phillips and Peter J. Slater. Graph competition independence and enclaveless parameters. In *Proceedings of the Thirty-third Southeastern International Conference on Combinatorics, Graph Theory and Computing (Boca Raton, FL, 2002)*, volume 154, pages 79–100, 2002.

[29] Peter Steinbach. *Field Guide to Simple Graphs*. Design Lab, Alburquerque, NM, second edition, 1995.