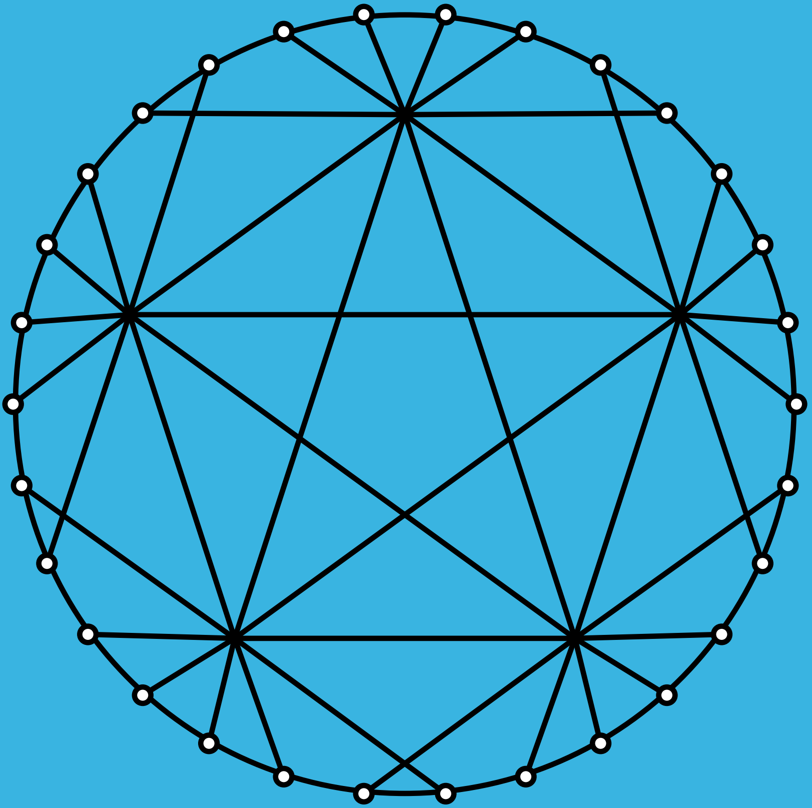
BULLETIN of The INSTITUTE of COMBINATORICS and its APPLICATIONS

Volume 90
October 2020



Boca Raton, FL, U.S.A.

ISSN: 2689-0674 (Online)
ISSN: 1183-1278 (Print)

BULLETIN OF THE ICA
Volume **90** (2020), 63–86

Peg solitaire on banana trees

JAN-HENDRIK DE WILJES*[1] AND MARTIN KREH[2]

[1]FREIE UNIVERSITÄT BERLIN, BERLIN, GERMANY
jandewiljes@mi.fu-berlin.de
[2]UNIVERSITY OF HILDESHEIM, HILDESHEIM, GERMANY
kreh@imai.uni-hildesheim.de

Abstract

In 2011, Beeler and Hoilman generalized the game of peg solitaire to arbitrary connected graphs. Since then peg solitaire has been considered on quite a few classes of graphs, but an important problem, the classification of solvable trees, remains open. In this article we consider peg solitaire on a class of trees with diameter 6, the banana trees. We will determine their solitaire number and their fool's solitaire number. Further, we introduce the graph invariant $\mathrm{ms}(G)$, which is the minimal number of edges one needs to add to G to make it solvable.

1 Introduction

In [3], Beeler and Hoilman introduced the game of peg solitaire on graphs as a generalization of the classical peg solitaire game:

Given a connected, undirected graph $G = (V, E)$, we can put pegs in the vertices of G. Given three vertices u, v, w with pegs in u and v and a hole in w (i.e., the vertex w does not contain a peg) such that $uv, vw \in E$, we can jump with the peg from u over v into w, removing the peg in v (see Figure 1). This jump will be denoted as $u \cdot \vec{v} \cdot w$.

*Corresponding author.

Key words and phrases: peg solitaire, fool's solitaire, banana tree graph
AMS (MOS) Subject Classifications: 05C57, 05C05, 91A43

Received: 10 September 2019
Accepted: 20 August 2020

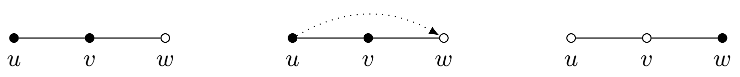


Figure 1: A jump in peg solitaire.

In general, we begin with a *starting state* $S \subset V$ of vertices that are empty (i.e., without pegs). A *terminal state* $T \subset V$ is a set of vertices that have pegs at the end of the game such that no more jumps are possible. A terminal state T is *associated* to a starting state S, if T can be obtained from S by a series of jumps. We will always assume that the starting state S consists of a single vertex.

The goal of the original game is to remove all pegs but one. This is not possible for all graphs. Therefore, we use the following notions. A graph G is called

- *solvable*, if there is some $v \in V$ such that the starting state $S = \{v\}$ has an associated terminal state consisting of a single vertex.
- *k-solvable*, if there is some $v \in V$ such that the starting state $S = \{v\}$ has an associated terminal state consisting of k vertices.
- *strictly k-solvable*, if G is k-solvable but not l-solvable for any $l < k$. In that case G has *solitaire number* $\mathrm{Ps}(G) = k$.

Peg solitaire has been considered for quite a few classes of graphs, including paths, complete graphs, stars, double stars and caterpillars (for more results and variants see [1–5, 7, 8, 10, 11]).

It remains an open problem to characterize all solvable trees. To this end, we will consider peg solitaire on a class of trees with diameter 6.

In [9], Chen, Lu and Yeh defined the following special class of trees: Given n stars $K_{1,k+1}$, we construct a graph, denoted by $B_{n,k}$ and called *banana tree*[1], by introducing a new vertex r which is connected by an edge to exactly one leaf of each star (see Figure 2 for $B_{4,7}$).

It is convenient to give the different types of vertices names. Therefore, we call any vertex of degree 1 a *leaf*, each original star centre a *plant*, the new vertex r *root* and each (original) leaf, that has been connected

[1]Note that our notation slightly differs from the one used by Chen, Lu and Yeh.

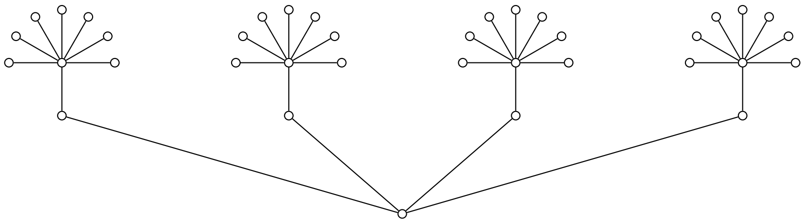


Figure 2: The banana tree $B_{4,7}$.

with r a *trunk*. We will also call a connected collection of a trunk, a plant, and k leafs *shrub* (i.e., a shrub is a set of vertices), see Figure 3. After ordering the shrubs in a certain order, denote the i-th shrub as S_i, the trunk of S_i as t_i, the plant of S_i as p_i and the leafs of S_i as $\ell_{i,j}$ for $i = 1, 2, \ldots, n, j = 1, 2, \ldots, k$.

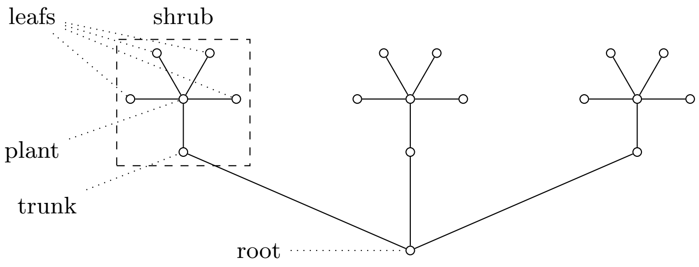


Figure 3: The banana tree $B_{3,4}$ with root, trunks, plants, leafs, and shrubs.

In this article we investigate peg solitaire on banana trees. We begin with solvability and the peg solitaire number in Section 2. In Section 3 we introduce a new graph invariant, namely the minimal number of edges that have to be added to a graph to make it solvable. We consider fool's solitaire on banana trees in Section 4 and give some related open questions in Section 5.

2 Solvability and the solitaire number of banana trees

For most parameter combinations banana trees are not solvable. Therefore, we will determine $\mathrm{Ps}(B_{n,k})$, starting with some special (partially known) cases before investigating the general case ($n \geq 3$ and $k \geq 2$).

Proposition 2.1. *We have*

(i) $\mathrm{Ps}(B_{1,0}) = \mathrm{Ps}(B_{1,1}) = 1$.

(ii) $\mathrm{Ps}(B_{1,k}) = k - 1$ *for* $k \geq 2$.

(iii) $\mathrm{Ps}(B_{2,0}) = \mathrm{Ps}(B_{2,1}) = 2$.

(iv) $\mathrm{Ps}(B_{n,0}) = 1$ *for* $n > 2$.

Proof. (i) Let P_i denote the path graph on i vertices. Then we have $B_{1,0} = P_3, B_{1,1} = P_4$ and the statement follows from [3, Theorem 2.3].

(ii) Let $DS(L, R)$ denote the double star with L resp. R pendant vertices. Then we have $B_{1,k} = DS(1, k)$ and the result follows from [4, Theorem 3.1].

(iii) This follows from $B_{2,0} = P_5, B_{2,1} = P_7$ and [3, Theorem 2.3].

(iv) Let $K_{1,n}(c; a_1, \ldots, a_n)$ denote the star graph $K_{1,n}$ with c pendant vertices adjacent to the star centre and a_i pendant vertices adjacent to the arms. Then $B_{n,0} = K_{1,n}(0; 1, \ldots, 1)$ and [7, Theorem 3.2] yield $\mathrm{Ps}(B_{n,0}) = 1$.

□

For most of the results in this and the next section we will apply the concept of packages and purges, first applied to the generalisation of peg solitaire to graphs in [6]. These objects are useful for inductive arguments and to reduce notation in proofs. A *package* P is a set of vertices together with a particular configuration of pegs and holes and a subset $C(P)$ of P, called the *catalyst*, such that a sequence of jumps exists that removes all pegs outside of $C(P)$ and returns $C(P)$ to its original configuration (if it was changed at all by the jump process). This elimination process is called a *P-purge*. Displaying a package P is usually done by giving the subgraph induced by P, while the catalyst is highlighted using a box (see for example Figure 4). Below each such drawing, we specify the necessary jumps.

Proposition 2.2. *We have*

(i) $\mathrm{Ps}(B_{2,2}) = 3$.

(ii) $\mathrm{Ps}(B_{2,k}) = 2k-2$ *for* $k \geq 3$.

(iii) $\mathrm{Ps}(B_{n,1}) = \lceil \frac{n}{2} \rceil$ *for* $n \geq 3$.

Proof. For proving (i) and (ii), we distinguish four starting states and give the jump sequences (which are mostly unique up to symmetry).

- Hole in r. After $p_1 \cdot \vec{t_1} \cdot r, t_2 \cdot \vec{r} \cdot t_1, \ell_{2,1} \cdot \vec{p_2} \cdot t_2$ no more jumps are possible.
- Hole in p_1. The sequence $r \cdot \vec{t_1} \cdot p_1, \ell_{1,1} \cdot \vec{p_1} \cdot t_1, p_2 \cdot \vec{t_2} \cdot r, r \cdot \vec{t_1} \cdot p_1, \ell_{1,2} \cdot \vec{p_1} \cdot t_1$ shows the 3-solvability for $k = 2$, since no more jumps are possible.
- Hole in $\ell_{1,1}$. There exist two possibilities, $\ell_{1,2} \cdot \vec{p_1} \cdot \ell_{1,1}$, after which we basically have the same situation as if we started with a hole in p_1 (with the last jump possible if and only if $k \geq 3$, which will be $\ell_{1,3} \cdot \vec{p_1} \cdot t_1$), and $t_1 \cdot \vec{p_1} \cdot \ell_{1,1}$. The second option will not yield a solution since after $t_2 \cdot \vec{r} \cdot t_1, \ell_{2,1} \cdot \vec{p_2} \cdot t_2$ we cannot jump any more.
- Hole in t_1. We start with $\ell_{1,1} \cdot \vec{p_1} \cdot t_1, r \cdot \vec{t_1} \cdot p_1, \ell_{1,2} \cdot \vec{p_1} \cdot t_1, p_2 \cdot \vec{t_2} \cdot r, r \cdot \vec{t_1} \cdot p_1$. If $k = 2$, no more jumps are possible. For $k \geq 3$ we have one more jump, namely $\ell_{1,3} \cdot \vec{p_1} \cdot t_1$, showing $(2k-2)$-solvability for $k \geq 3$.

  The alternative first jump $t_2 \cdot \vec{r} \cdot t_1$ is not better since after $p_1 \cdot \vec{t_1} \cdot r, \ell_{2,1} \cdot \vec{p_2} \cdot t_2$, $r \cdot \vec{t_2} \cdot p_2, \ell_{2,2} \cdot \vec{p_2} \cdot t_2$ no more jumps are possible.

Since all cases have been considered, the statements follow.

Case (iii) considers generalised stars (diameter 6), hence Proposition 2.1 (iv) immediately gives us $\mathrm{Ps}(B_{n,k}) \leq n+1$. To obtain a better bound, we use the crutch purge (see Figures 4 and 5, where the crutch package is highlighted and only pegs in the crutch package are shown).

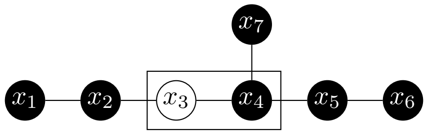


Figure 4: Crutch package. The corresponding purge is given by $x_1 \cdot \vec{x}_2 \cdot x_3, x_4 \cdot \vec{x}_3 \cdot x_2, x_6 \cdot \vec{x}_5 \cdot x_4, x_7 \cdot \vec{x}_4 \cdot x_3, x_2 \cdot \vec{x}_3 \cdot x_4$.

We start with a hole in t_1. For $i = 1, 2, \ldots, \lfloor \frac{n-1}{2} \rfloor$, we carry out the crutch purge to eliminate S_{2i-1} completely and S_{2i} as well as S_{2i+1} partially,

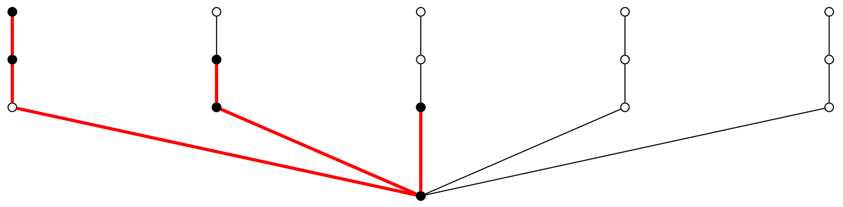


Figure 5: The banana tree $B_{5,1}$ with the crutch package highlighted.

leaving pegs in $\ell_{2i,1}, p_{2i+1}$ and $\ell_{2i+1,1}$ (and certainly in r). We will arrive at one of the following two situations:

- If n is even, we have pegs in $\ell_{i,1}$ for $i \in \{2, 4, \ldots, n-2\}$, in r, p_{n-1}, $\ell_{n-1,1}, t_n, p_n$, and in $\ell_{n,1}$, and holes elsewhere. Now the jump sequence $\ell_{n-1,1} \cdot \vec{p}_{n-1} \cdot t_{n-1}, t_{n-1} \cdot \vec{r} \cdot t_{n-2}, p_n \cdot \vec{t}_n \cdot r, r \cdot \vec{t}_{n-2} \cdot p_{n-2}, \ell_{n-2,1} \cdot \vec{p}_{n-2} \cdot t_{n-2}$ eliminates 5 pegs resulting in a terminal state containing $\frac{n}{2}$ pegs.
- If n is odd, we have pegs in $\ell_{i,1}$ for $i \in \{2, 4, \ldots, n-1\}$, in r, p_n, and in $\ell_{n,1}$, and holes elsewhere. Now the jump sequence $\ell_{n,1} \cdot \vec{p}_n \cdot t_n, r \cdot \vec{t}_n \cdot p_n$ eliminates 2 pegs resulting in a terminal state containing $\frac{n+1}{2}$ pegs.

To show that equality holds, we take a closer look at the jumps necessary to remove a peg from a leaf. The only possible jump is $\ell_{i,1} \cdot \vec{p}_i \cdot t_i$. This is only possible if t_i is empty and p_i contains a peg. For the first shrub this may hold from the beginning, for any other shrub, to achieve this, we need to empty the trunk. Since in the i-th shrub there is a peg in the leaf, the only possibility to do this is via $t_i \cdot \vec{r} \cdot t_j$. So if we do these two jumps, the i-th leaf and the root are emptied. To empty the next leaf, we have to do the analogous jumps in another shrub, which requires us to jump with a peg into the root first. This is only possible with the jump $p_m \cdot \vec{t}_m \cdot r$ for some m. Hence, from now on, to empty a leaf, we have to empty two trunks first. Inductively we see, that we can remove at most $1 + \frac{n-1}{2}$ pegs from leafs if n is odd and $1 + \frac{n}{2}$ pegs from leafs if n is even. Moreover, the last jump will not end in a leaf, hence there is at least one peg left in a vertex that is not a leaf. Thus the process of eliminating pegs with the crutch purge is best possible and indeed we have $\operatorname{Ps}(B_{n,1}) = \lceil \frac{n}{2} \rceil$. □

For $k \geq 2$ we can use the idea of Proposition 2.2 (iii) to obtain the following upper bound on $\operatorname{Ps}(B_{n,k})$ (observe that we can eliminate one more peg from a leaf because we have a peg in p_n in the terminal state).

Corollary 2.3. *For positive integers n, k with $k \geq 2$ we have*

$$\mathrm{Ps}(B_{n,k}) \leq n(k-1) + \left\lfloor \frac{n}{2} \right\rfloor.$$

For $n \geq 2$ and $(n, k) \notin \{(2,2), (3,2), (5,2), (3,3)\}$, we can do better (which can be shown with the following two results).

Proposition 2.4. *For integers n, k with $k > n \geq 3$ we have* $\mathrm{Ps}(B_{n,k}) \leq n(k-1)$.

Proof. Starting with a hole in $\ell_{1,1}$, the jump sequence $\ell_{1,2} \cdot \vec{p}_1 \cdot \ell_{1,1}, r \cdot \vec{t}_1 \cdot p_1, \ell_{1,1} \cdot \vec{p}_1 \cdot t_1$ eliminates 3 pegs from $S_1 \cup \{r\}$.

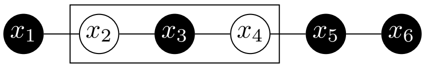


Figure 6: Wand package. The corresponding purge is $x_6 \cdot \vec{x}_5 \cdot x_4, x_4 \cdot \vec{x}_3 \cdot x_2, x_1 \cdot \vec{x}_2 \cdot x_3$.

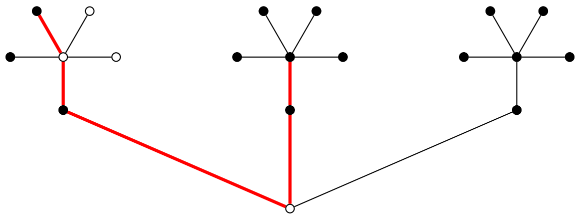


Figure 7: The banana tree $B_{3,4}$ with the wand package highlighted.

Repeatedly using the wand purge, where the two holes are in p_1 and r and the two adjacent vertices with pegs correspond to t_i and p_i for some $i \geq 2$, we can eliminate a peg from a leaf of S_1 per shrub S_i for $i = 2, 3, \ldots, n$. After that no more jumps are possible and we are left with pegs in $\ell_{1,j}$ for $j = n+2, n+3, \ldots, k$, in t_1, and in $\ell_{i,j}$ for $i = 2, 3, \ldots, n$ and $j = 1, 2, \ldots, k$. □

Using the idea from Proposition 2.4 repeatedly, we obtain the following upper bound for the general case.

Theorem 2.5. *For integers n, k with $n \geq 3$ and $k \geq 2$ we have*

$$\mathrm{Ps}(B_{n,k}) \leq n(k-1) + \left\lceil \frac{n+2}{k+1} \right\rceil - 1.$$

Proof. The case $k > n$ is dealt with in Proposition 2.4, hence we assume $k \leq n$ from now on. Define $q = \left\lfloor \frac{n+2}{k+1} \right\rfloor$, $\mathcal{S}_0 = \{S_j : 1 \leq j \leq k-1\}$ and $\mathcal{S}_i := \{S_j : i(k+1) - 1 \leq j \leq (i+1)(k+1) - 2)\}$ for $i = 1, 2, \ldots, q-1$ (see Figure 8).

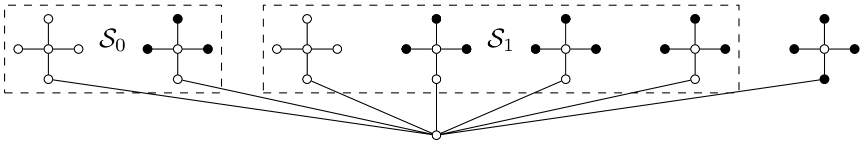


Figure 8: The banana tree $B_{7,3}$ with the sets $\mathcal{S}_0$ and $\mathcal{S}_1$ and the terminal state.

We use Proposition 2.4 on $\mathcal{S}_0 \cup \{r\}$ with a starting hole in $\ell_{1,1}$ which yields, after the first jumps $\ell_{1,2} \cdot \vec{p}_1 \cdot \ell_{1,1}, r \cdot \vec{t}_1 \cdot p_1, \ell_{1,1} \cdot \vec{p}_1 \cdot t_1$ and repeated wand purges, the intermediate state having pegs in all S_j for $j \geq k$ and in t_1 as well as in $\ell_{i,j}$ for $i = 2, 3, \ldots, k-1$ and $j = 1, 2, \ldots, k$.

Executing the jumps $p_k \cdot \vec{t}_k \cdot r$ and $t_1 \cdot \vec{r} \cdot t_k$ we have a similar peg position as before, but this time in $\mathcal{S}_1 \cup \{r\}$, where holes are exactly in r and p_k (but not in any leafs as before). We can now use k wand purges to eliminate all pegs in leafs from S_k.

Iterating this process for $i = 2, 3, \ldots, q-1$ results in a state with pegs in all leafs of S_j for $j \leq q(k+1) - 2$ with $j \notin \{1, k, 2k+1, 3k+2, \ldots, (q-1)(k+1) - 1\}$. Further, there are pegs in $t_{(q-1)(k+1)-1}$ and in all vertices of S_j for $j \geq q(k+1) - 1$.

If $q(k+1) - 2 = n$ (this being the case if and only if $n+2$ is a multiple of $k+1$), no more jumps are possible.

Otherwise we execute the jumps $p_{q(k+1)-1} \cdot \vec{t}_{q(k+1)-1} \cdot r$ and $t_{(q-1)(k+1)-1} \cdot \vec{r} \cdot t_{q(k+1)-1}$. From $S_{q(k+1)-1}$ we can now remove $n - q(k+1) + 1$ pegs from leafs using the wand purge together with the remaining full shrubs. This yields a total number of

$$\underbrace{k(k-2)}_{\text{leafs in } \mathcal{S}_0} + \underbrace{(q-1)k^2}_{\substack{\text{leafs in} \\ \mathcal{S}_1, \ldots, \mathcal{S}_{q-1}}} + \underbrace{1}_{t_{q(k+1)-1}} + \underbrace{k - (n - q(k+1) + 1)}_{\text{leafs in } S_{q(k+1)-1}} + \underbrace{k(n - q(k+1) + 1)}_{\text{leafs in } S_{q(k+1)}, \ldots, S_n}$$

pegs in our terminal state, which simplifies to the quantity in the statement. □

We will show equality shortly, but first we give a lower bound which already demonstrates one of the major ingredients of the proof and, further, settles the case $n < k$.

Proposition 2.6. *For $k \geq 2$ we have $\mathrm{Ps}(B_{n,k}) \geq n(k-1)$.*

Proof. We show that we can remove at most $n+1$ pegs from leafs. To remove a peg from a leaf, we need to perform the jump $\ell_{i,j} \cdot \vec{p_i} \cdot t_i$. If this has been done once for the i-th shrub (if it is possible, i.e., if t_i is emptied first), the plant p_i is empty. To fill p_i we have to jump $r \cdot \vec{t_i} \cdot p_i$. After this has been done once, we need to jump with a peg in the root first before we can jump again in a plant. This is only possible with the jump $p_j \cdot \vec{t_j} \cdot r$. But this would remove a peg from the plant of another shrub. Hence each time after the first time that we try to jump with a peg in r, one more plant is emptied and cannot be used to empty an adjacent leaf. Hence at most $n+1$ leafs can be emptied. Thus the number of remaining pegs in leafs is at least $nk - (n+1) = n(k-1) - 1$. Since there also has to be a peg (the one from the leaf last emptied) in a non leaf, the statement follows. □

This means, that the bound in Proposition 2.4 is in fact an equality, i.e., we have the following corollary.

Corollary 2.7. *For integers n, k with $k > n \geq 3$ we have $\mathrm{Ps}(B_{n,k}) = n(k-1)$.*

To settle the case $n \geq k$, we need additional ideas and auxiliary results, starting with the following lemma.

Lemma 2.8. *For positive integers n, k with $n \geq 3$ a terminal state with $\mathrm{Ps}(B_{n,k})$ pegs and empty root exists.*

Proof. We choose an optimal solution process and, if necessary, modify it to get the desired terminal state. Suppose $p_i \cdot \vec{t_i} \cdot r$ was the last jump (for some $i \in [n]$, where $[n] = \{1, 2, \ldots, n\}$). Then neither of the trunks contains a peg in the final case. We distinguish four cases.

- If the second to last jump was $r \cdot \vec{t_j} \cdot p_j$ (for some $j \in [n]$), we could instead jump $t_j \cdot \vec{r} \cdot t_h, p_i \cdot \vec{t_i} \cdot r, t_h \cdot \vec{r} \cdot t_i$ for some $i \neq h \neq j$. This yields a state with less than $\mathrm{Ps}(B_{n,k})$ pegs, contradicting the definition of the peg solitaire number.

- The second to last jump was of the form $\ell_{j,h} \cdot \vec{p_j} \cdot \ell_{j,h'}$ or $\ell_{j,h} \cdot \vec{p_j} \cdot t_j$ for some $j \in [n]$ and $h, h' \in [k]$. These situations cannot occur, since instead the jumps $p_i \cdot \vec{t_i} \cdot r, \ell_{j,h} \cdot \vec{p_j} \cdot t_j, r \cdot \vec{t_j} \cdot p_j$ would be possible, yielding a contradiction.

- The second to last jump was of the form $t_j \cdot \vec{r} \cdot t_{j'}$ for some $j, j' \in [n]$. Then $j' \neq i$ is impossible since $t_{j'} \cdot \vec{r} \cdot t_j$ would be possible in the terminal state. Therefore, we have $j' = i$, implying that t_i is empty before the second to last jump. At most one leaf in the shrub S_i contains a peg in the terminal state. Otherwise, we would have pegs in $\ell_{i,h}$ and $\ell_{i,h'}$ for some $h, h' \in [k]$. Instead of the last two jumps we could perform $\ell_{i,h} \cdot \vec{p_i} \cdot t_i, r \cdot \vec{t_i} \cdot p_i, \ell_{i,h'} \cdot \vec{p_i} \cdot t_i$, yielding a contradiction. So let $\ell_{i,h}$ be empty in the terminal state. Instead of $p_i \cdot \vec{t_i} \cdot r$ we perform the last jump $t_i \cdot \vec{p_i} \cdot \ell_{i,h}$, which yields the desired terminal state.

- The second to last jump was $t_j \cdot \vec{p_j} \cdot \ell_{j,h}$ for some $j \in [n]$ and $h \in [k]$, which implies that all leafs in S_j are empty. Otherwise, exchanging the last two jumps by $p_i \cdot \vec{t_i} \cdot r, t_j \cdot \vec{r} \cdot t_i, \ell_{j,h'} \cdot \vec{p_j} \cdot t_j$ for some $h' \in [k]$ yields a contradiction. For the same reason, none of the leafs in S_i contain a peg. Note that it is impossible to have only pegs in all plants and trunks since every non-starting configuration contains at least two adjacent vertices with holes. Hence, moving backwards through the previous jumps, one of them has to be of the form as considered in the first three cases, yielding the desired solution process (probably after first jumping $t_j \cdot \vec{p_j} \cdot \ell_{j,h}$ to obtain an empty trunk).

If, instead, the root is filled by $p_i \cdot \vec{t_i} \cdot r$ followed by the final jumps which only use vertices from shrubs, we can proceed as before. Therefore, the root is empty in the modified terminal state. □

We are ready to prove the previously mentioned lower bound.

Theorem 2.9. *For integers n, k with $n \geq k \geq 3$ we have*

$$\mathrm{Ps}(B_{n,k}) \geq n(k-1) + \left\lceil \frac{n+2}{k+1} \right\rceil - 1.$$

Proof. Although we have (up to automorphism) four starting holes, after one or two jumps (again up to automorphism) only one of the following three configurations can occur (cf. Figure 9).

(C1) Holes in exactly p_1, r and t_2. This is reached after the jumps $p_1 \cdot \vec{t_1} \cdot r, t_2 \cdot \vec{r} \cdot t_1$ (starting hole r) or after the jumps $t_1 \cdot \vec{p_1} \cdot \ell_{1,1}, t_2 \cdot \vec{r} \cdot t_1$ (starting hole $\ell_{1,1}$).

(C2) Holes in exactly $\ell_{1,1}, t_1$ and r. This is reached after the jumps $\ell_{1,1} \cdot \vec{p_1} \cdot t_1, r \cdot \vec{t_1} \cdot p_1$ (starting hole t_1) or after the jumps $\ell_{1,2} \cdot \vec{p_1} \cdot \ell_{1,1}, r \cdot \vec{t_1} \cdot p_1$ (starting hole $\ell_{1,1}$).

(C3) Holes in exactly t_1 and r. This is reached after the jump $t_1 \cdot \vec{r} \cdot t_2$ (starting hole t_2) or after the jump $r \cdot \vec{t_1} \cdot p_1$ (starting hole p_1).

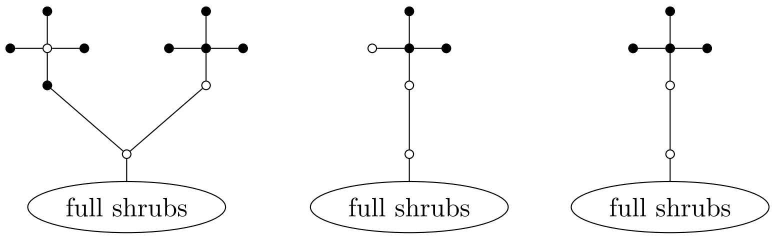


Figure 9: The configurations C1, C2, C3 (in that order).

Consider a sequence of jumps which leads to an optimal solution, i.e. its terminal state contains $\mathrm{Ps}(B_{n,k})$ vertices, none of which lies in the root, which is possible due to Lemma 2.8. If we denote by $\mathrm{IJ}_i^{(j)}$ a particular sequence of (so called inner) jumps, taking place only inside shrubs without using the root, by FR_i a jump filling the root, and by ER_i a jump emptying the root, this (optimal solution) sequence can (for some positive integer m) be written as

$$\mathrm{IJ}_1^{(1)}, \mathrm{FR}_1, \mathrm{IJ}_1^{(2)}, \mathrm{ER}_1, \mathrm{IJ}_2^{(1)}, \mathrm{FR}_2, \mathrm{IJ}_2^{(2)}, \mathrm{ER}_2, \ldots, \mathrm{IJ}_m^{(1)}, \mathrm{FR}_m, \mathrm{IJ}_m^{(2)}, \mathrm{ER}_m, \mathrm{IJ}_{m+1}^{(1)}.$$

Since every jump in $\mathrm{IJ}_i^{(2)}$ empties a plant, none of them can happen inside the shrub which was used to fill the root in FR_i. Therefore, we can exchange these jumps to obtain a (reordered) optimal solution process

$$\mathrm{IJ}_1, \mathrm{FR}_1, \mathrm{ER}_1, \mathrm{IJ}_2, \mathrm{FR}_2, \mathrm{ER}_2, \ldots, \mathrm{IJ}_m, \mathrm{FR}_m, \mathrm{ER}_m, \mathrm{IJ}_{m+1},$$

where IJ_i is the sequence of jumps in $\mathrm{IJ}_i^{(1)}$ followed by those in $\mathrm{IJ}_i^{(2)}$ (note that IJ_i might be empty). We summarize some basic facts:

- The jumps in IJ_i only remove pegs from plants (and transfer pegs between leafs or trunks inside shrubs). Further, only these jumps can empty leafs. Emptying a leaf decreases the number of pegs in plants by 1.

- The jump FR_i transfers a peg from a plant to the root and removes a peg from a trunk.
- The jump ER_i either fills a plant and removes a peg from a trunk or removes a peg from the root and transfers a peg from one trunk to another trunk.

To empty a leaf, we need to perform the jump $\ell_{i,j} \cdot \vec{p}_i \cdot t_i$, hence requiring a peg in a plant. The sequence of jumps FR_i and ER_i either removes a peg from a plant or transfers a peg from one plant to another (we call the second of these combinations of fill-root-jump and empty-root-jump a plant-transfer-jump). Hence, the number of pegs in plants counted after ER_i decreases (with growing i) monotonically. Therefore, the number of times we can empty a leaf is bounded by n, the number of pegs in plants.

For the jump $\ell_{i,j} \cdot \vec{p}_i \cdot t_i$ to be possible, we need a peg in p_i and a hole in t_i. Call such a configuration a p-configuration and the corresponding shrub a p-shrub (if the hole and the peg are reversed, we use the terms t-configuration and t-shrub). Each of the three above mentioned configurations C1–C3 contains exactly one p-shrub and at most one t-shrub. A shrub is called *accessible* if it is a p-shrub or a t-shrub.

Only in current p-shrubs leafs can be emptied immediately (meaning using inner jumps). This can happen at most k times for each shrub.[2] Hence, at some point, we need to use a jump having the root as its middle vertex, i.e. a jump $\mathrm{ER}_q = t_j \cdot \vec{r} \cdot t_i$ for some i, j, q, if we want to obtain a new accessible shrub. It is important to note that such an empty-root-jump together with the previous fill-root-jump removes a peg from a plant without removing a peg from a leaf. Therefore, for every one of these jumps, one less leaf can be emptied. We distinguish four (distinct) cases for the jump ER_q.

($\mathrm{ER}_q^{(1)}$) p_i and p_j are empty. Then the number of p-shrubs and the number of t-shrubs does not change and we obtain exactly one t-shrub which was not accessible before ER_q.

($\mathrm{ER}_q^{(2)}$) p_i contains a peg and p_j is empty. In this case, we remove a p-shrub and a t-shrub but do not get a new accessible shrub.

[2]Note that, certainly, a peg may be transferred to a leaf of this shrub if all leafs are empty. Since this is only possible via two fill-root-jumps and two empty-root-jumps and the removal of a peg in a plant (to fill the leaf), this will not happen in an optimal solution.

($\mathrm{ER}_q^{(3)}$) p_i and p_j contain a peg. Then the number of p-shrubs and the number of t-shrubs does not change and we obtain exactly one p-shrub which was not accessible before ER_q.

($\mathrm{ER}_q^{(4)}$) p_i is empty and p_j contains a peg. In this case, we obtain a new p- and a new t-shrub without removing shrubs of these types.

Note that at most two new accessible shrubs are generated by ER_q, which is the case exactly for $\mathrm{ER}_q^{(4)}$. Hence, the above mentioned fact, that only k pegs can be removed from leafs in a certain shrub, implies that (on average) we need at least one jump of type $\mathrm{ER}_q^{(1)}, \mathrm{ER}_q^{(2)}$ or $\mathrm{ER}_q^{(3)}$ for every k plant-transfer-jumps or one jump of type $\mathrm{ER}_q^{(4)}$ for every $2k$ plant-transfer-jumps.

First, we demonstrate the idea in the case that $N_4 = 0$, where N_4 is the number of jumps of type $\mathrm{ER}_q^{(4)}$. Consider configuration C2. This contains only one accessible shrub, which has $k-1$ pegs in leafs. During the solving process, at most $k-1$ of these leafs can be emptied. For all other shrubs, it is possible to empty k leafs, but first they need to be made accessible, both of which pegs in plants are needed for. Hence, in the best case, meaning emptying as many leafs as possible, the number of plants can be written as

$$n = \underbrace{k-1}_{\text{empty pegs in accessible shrub of SP2}} + (\underbrace{1}_{\text{make accessible}} + \underbrace{k}_{\text{empty leafs}}) \cdot N' + \underbrace{x}_{\text{some more jumps}}, \tag{1}$$

where $N' = \frac{n-k+1-x}{k+1}$, for some $0 \le x \le k$, and, if $x > 0$, "some more jumps" means we use at best one peg from a plant to make a shrub accessible and $x-1$ pegs from plants to empty leafs. We cannot use $N' + \left\lceil \frac{x}{k} \right\rceil$ plants to empty leafs (because they are used to make shrubs accessible), hence Equation (1) implies that at most

$$k - 1 + N'k + x - \left\lceil \frac{x}{k} \right\rceil = k - 1 + \frac{n-k+1-x}{k+1}k + x - \left\lceil \frac{x}{k} \right\rceil \tag{2}$$

leafs can be emptied beginning with configuration C2. Note that one leaf was already empty at the start. Further, the peg which was in the leaf last emptied cannot be eliminated completely in the sense that it might be removed but then the peg used to remove it is still there in the end (or the one used to remove this and so on). Combining Equations (1) and (2) and these two facts, we get that the terminal state contains at least

$$\begin{aligned} nk - &\left(k-1+\frac{n-k+1-x}{k+1}k+x-\left\lceil\frac{x}{k}\right\rceil\right) \underbrace{-1+1}_{\text{two facts}} \\ &= \frac{nk^2+1-k-x}{k+1}+\left\lceil\frac{x}{k}\right\rceil \\ &= n(k-1)+\frac{n+2}{k+1}-1+\left\lceil\frac{x}{k}\right\rceil-\frac{x}{k+1} \\ &\geq n(k-1)+\frac{n+2}{k+1}-1 \end{aligned}$$

pegs, thus settling this case. The other two starting configurations (C1 and C3) can be dealt with in the same way.

The case $N_4 > 0$ is slightly more tricky but can be related to the previous one. We, additionally, consider the number of accessible shrubs.[3] For every such shrub in the terminal state, we have a non removable peg. Note that, in the solution process, this number only increases if $ER_q^{(4)}$ is performed and decreases[4] for $ER_q^{(2)}$, both times by 2. Therefore, for every jump $ER_q^{(4)}$, we either have 2 pegs in accessible shrubs in the terminal state or some $ER_q^{(2)}$ had been performed. In the latter case (which is better than the first), one peg from a plant has to be used for $ER_q^{(2)}$, which cannot empty a leaf, and, further, this jump does not produce new accessible shrubs. This means that the ratio of pegs in plants (in the sense one jump of type $ER_q^{(i)}$ for every k plant-transfer-jumps) used for removing pegs is not better (and in general worse[5]) than in the case $N_4 = 0$. This concludes the proof. □

Combining Theorems 2.5 and 2.9, we obtain the following result.

Corollary 2.10. *For integers n, k with $n \geq k \geq 3$ we have*

$$\mathrm{Ps}(B_{n,k}) = n(k-1)+\left\lceil\frac{n+2}{k+1}\right\rceil-1.$$

Concerning the solvability of banana trees, we (by combing most of the previous results) conclude

[3]We could be more precise and consider it as a function of time, but it does not seem necessary to overcomplicate the notation in order to present the idea.

[4]It should be mentioned that also an inner jump $\ell_{i,j_1} \cdot \vec{p}_i \cdot \ell_{i,j_2}$ can decrease the number of accessible shrubs (only!) by 1, but since this jump removes a peg from a plant without emptying a leaf, it cannot be a jump in an optimal solution.

[5]In particular if there are more jumps $ER_q^{(4)}$ than $ER_q^{(2)}$.

Corollary 2.11. *$B_{n,k}$ is solvable if and only if $k = 0$ and $n \neq 2$ or $k = 1$ and $n = 1$ or $k = 2$ and $n = 1$.*

3 Making banana trees solvable via appending edges

Among other things, the solitaire number is influenced by the number of edges in a graph G in the sense that $\mathrm{Ps}(H) \leq \mathrm{Ps}(G)$ for every supergraph H of G on the same vertex set. Therefore it appears to be natural to ask for the smallest number of edges, denoted by $\mathrm{ms}(G)$, we have to add to a graph G to make it solvable.[6] Note that $\mathrm{ms}(G) = 0$ holds if and only if G is solvable. Every complete graph is solvable, therefore $\mathrm{ms}(G)$ exists for every graph G. Further, for $|V(G)| \geq 3$, we have $\mathrm{ms}(G) \leq \mathrm{Ps}(G) - 1$, since by successively connecting two vertices in a terminal state by an edge one can reduce the number of pegs by (at least) 1 in each step.

We will determine upper bounds for $\mathrm{ms}(B_{n,k})$, starting (again) with some small cases.

Proposition 3.1. *We have*

(i) $\mathrm{ms}(B_{1,0}) = \mathrm{ms}(B_{1,1}) = 0$.

(ii) $\mathrm{ms}(B_{2,0}) = \mathrm{ms}(B_{2,1}) = 1$.

(iii) $\mathrm{ms}(B_{n,0}) = 0$ *for* $n > 2$.

Proof. The statements immediately follow from Proposition 2.1. □

Proposition 3.2. *We have* $\mathrm{ms}(B_{n,1}) \leq \lceil \frac{n-1}{4} \rceil$ *for* $n \geq 3$.

Proof. We use the idea from the proof of Proposition 2.2 (iii). Recall that the terminal state contains $\ell_{i,1}$ for $i \in \{2, 4, \ldots, 2\lceil \frac{n-4}{2} \rceil\}$ and (depending on the parity of n) $\ell_{n,1}, t_{n-2}$ if n is even and $\ell_{n-1,1}, p_n$ if n is odd. Since we are not able to remove the pegs in this configuration without using too many extra edges, we will remove them in pairs as described below:

[6] The idea of adding edges to reduce the solitaire number is not new. In [1] Beeler and Gray investigated related extremal questions. They constructed graphs with the property that adding any (non existing) edge will decrease the solitaire number, calling these graphs edge-critical.

For $i = 1, 2, \ldots, \lfloor \frac{n-1}{4} \rfloor$ we add edges $p_{4i-2}p_{4i}$, start with a hole in t_1 and do $2i$ crutch purges (using the shrubs S_1, S_2, S_3, then S_3, S_4, S_5, and so on). After each odd (meaning the 1st, 3rd,...) crutch purge we have a hole in p_{4i-2} and can jump $\ell_{4i,1} \cdot \vec{p}_{4i} \cdot p_{4i-2}, \ell_{4i-2,1} \cdot \vec{p}_{4i-2} \cdot p_{4i}$ to eliminate exactly the pegs in $\ell_{4i-2,1}$ and $\ell_{4i,1}$. After the last crutch purge the configuration depends on the remainder of n modulo 4:

- If $n \equiv 0 \bmod 4$, we are left with pegs in r, t_{n-2}, t_{n-1}, t_n and $p_j, \ell_{j,1}$ for $j \in \{n-3, n-2, n-1, n\}$. This configuration can be solved after adding the edge $p_{n-2}p_n$ (using a crutch purge, involving S_{n-3}, S_{n-2} and S_{n-1}, and some other jumps).
- If $n \equiv 1 \bmod 4$, we are left with pegs in $r, p_n, \ell_{n,1}$. This configuration is solvable.
- If $n \equiv 2 \bmod 4$, we are left with pegs in r, t_n and $p_j, \ell_{j,1}$ for $j \in \{n-1, n\}$. This configuration (not being solvable since it corresponds to P_7) can be solved after adding the edge $t_{n-1}\ell_{n-1,1}$.
- If $n \equiv 3 \bmod 4$, we are left with pegs in r, t_{n-1}, t_n and $p_j, \ell_{j,1}$ for $j \in \{n-2, n-1, n\}$. This configuration can be solved after adding the edge $t_{n-1}\ell_{n-1,1}$ (using a crutch purge and some other jumps).

In the last three cases we added one edge and none in the first, resulting in a total of $\lceil \frac{n-1}{4} \rceil$ edges. ☐

In the following we will use more drawings of packages to help understanding arguments in the proofs. We use dotted lines for added edges to distinguish them from existing ones, which are displayed as solid lines.

To prove the upcoming proposition, we use the following result from [1].

Lemma 3.3 ([1, Corollary 2.2]). *A graph G is not solvable if it contains a vertex which is adjacent to at least $\frac{1}{2}|V(G)|$ leafs.*

Proposition 3.4. *Let $k \geq 4$. We have* $\mathrm{ms}(B_{1,k}) = \lceil \frac{k}{4} \rceil$ *if k or $k+1$ is a multiple of* 4 *and* $\mathrm{ms}(B_{1,k}) = \lceil \frac{k}{4} \rceil - 1$ *otherwise.*

Proof. First we show that we need to add at least $\lceil \frac{k}{4} \rceil$ resp. $\lceil \frac{k}{4} \rceil - 1$ edges. When adding an edge, at most two of the k leafs which are adjacent to p_1

become non leafs. Lemma 3.3 implies that it is necessary to add at least x edges such that

$$k - 2x < \frac{1}{2}(k+3)$$

holds to obtain a solvable graph. Distinguishing the four different residues of k modulo 4 yields the desired lower bound.

Now we verify the upper bound $\lceil \frac{k}{4} \rceil$ if k or $k+1$ is a multiple of 4. Starting with a hole in t_1 and jumping $\ell_{1,1} \cdot \vec{p}_1 \cdot t_1, r \cdot \vec{t}_1 \cdot p_1, \ell_{1,2} \cdot \vec{p}_1 \cdot t_1$, we arrive at a configuration where we can carry out the jumps $t_1 \cdot \vec{\ell}_{1,3} \cdot p_1$ and $\ell_{1,4} \cdot \vec{p}_1 \cdot t_1$ (if $k \geq 4$) after adding the edge $\ell_{1,3}t_1$. This proves the statement for $k \leq 4$.

For $k \geq 5$ we perform the timbrel purge exactly $\left(\lfloor \frac{k}{4} \rfloor - 1\right)$-times from the current state after adding the edges $\ell_{1,4i-3}\ell_{1,4i-2}$ for $i = 2, 3, \ldots, \lfloor \frac{k}{4} \rfloor$.

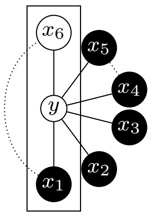


Figure 10: Timbrel package (idea from [4] for the windmill variant). The corresponding purge is given by $x_5 \cdot \vec{x}_4 \cdot y, x_2 \cdot \vec{y} \cdot x_6, x_1 \cdot \vec{x}_6 \cdot y, x_3 \cdot \vec{y} \cdot x_1$.

After the last timbrel purge we are done if k is a multiple of 4. Otherwise we need to add the edge $\ell_{1,k-1}\ell_{1,k}$ to obtain a solvable graph (and state).

Finally, let neither k nor $k+1$ be a multiple of 4 and let $d \in \{1, 2\}$ be the remainder of k modulo 4. Again we start with a hole in t_1 and we add the edges $\ell_{1,4i-3}\ell_{1,4i-2}$ for $i = 1, 2, \ldots, \lceil \frac{k}{4} \rceil - 1$. First jump $\ell_{1,k} \cdot \vec{p}_1 \cdot t_1$. The vertices p_1 and $\ell_{1,i}, i = 1, \ldots, k-d$ induce a subgraph isomorphic to a windmill variant. Peg solitaire on these graphs has been studied in [4]. Using the elimination process given there, we can reduce this subgraph such that we have pegs in $\ell_{1,1}, \ell_{1,2}, \ell_{1,3}$ and $\ell_{1,4}$ (and $\ell_{1,k-1}$ if $d = 2$) and holes elsewhere (note that there are still pegs in t_1 and r). The subgraph induced by r, t_1, p_1 and $\ell_{1,i}$ for $i \in [1, 4]$ and additionally $i = k-1$ if $d = 2$ (this subgraph contains the last remaining pegs) is solvable. □

Proposition 3.5. *For every positive integer n we have*

$$\text{ms}(B_{n,2}) \leq \left\lfloor \frac{n}{2} \right\rfloor.$$

Proof. Start with a hole in t_1. For odd n add edges $\ell_{2i,1}\ell_{2i+1,1}$ for $i = 1, 2, \ldots, \frac{n-1}{2}$ and jump $\ell_{1,1} \cdot \vec{p}_1 \cdot t_1, r \cdot \vec{t}_1 \cdot p_1, \ell_{1,2} \cdot \vec{p}_1 \cdot t_1$. Next iterate the following process:

- Execute jumps $p_{2i} \cdot \vec{t}_{2i} \cdot r, t_{2i-1} \cdot \vec{r} \cdot t_{2i}$.
- Do a bull purge on $S_{2i} \cup S_{2i+1} \cup \{r\}$ (including the new edge).

For even n add edges $\ell_{2i-1,1}\ell_{2i,1}$ for $i = 1, 2, \ldots, \frac{n}{2}$ and jump $t_2 \cdot \vec{r} \cdot t_1, p_1 \cdot \vec{t}_1 \cdot r, \ell_{2,1} \cdot \vec{p}_2 \cdot t_2, r \cdot \vec{t}_2 \cdot p_2, \ell_{2,2} \cdot \vec{p}_2 \cdot \ell_{2,1}, \ell_{2,1} \cdot \vec{\ell}_{1,1} \cdot p_1, \ell_{1,2} \cdot \vec{p}_1 \cdot t_1$. The iteration process is similar, just use shrubs S_{2i-1} and S_{2i} instead of S_{2i} and S_{2i+1}. □

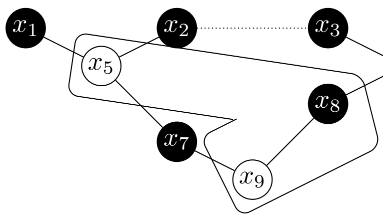


Figure 11: Bull package. The corresponding purge is given by $x_3 \cdot \vec{x}_2 \cdot x_5, x_1 \cdot \vec{x}_5 \cdot x_2, x_4 \cdot \vec{x}_6 \cdot x_3, x_2 \cdot \vec{x}_3 \cdot x_6, x_6 \cdot \vec{x}_8 \cdot x_9, x_7 \cdot \vec{x}_9 \cdot x_8$.

Theorem 3.6. *For positive integers n, k with $n \geq 2$ and $k \geq 3$ we have*

$$\mathrm{ms}(B_{n,k}) \leq \left\lceil \frac{n}{2} \right\rceil + 1.$$

Proof. For even n we add the edges $p_{2i-1}p_{2i}$ for $i = 1, 2, \ldots, \frac{n}{2}$, start with a hole in r, and use the reindeer purge iteratively exactly $\frac{n-2}{2}$ times. We add another edge, namely $t_{n-1}p_n$, and remove all but one peg from S_{n-1} and S_n.

For odd n we add edges $p_{2i-1}p_{2i}$ for $i = 1, 2, \ldots, \frac{n-1}{2}$ and use the reindeer purge exactly $\frac{n-3}{2}$ times. Then we add another two edges, namely $p_{n-1}p_n, p_np_{n-2}$, and use the triblade purge. □

Remark. It seems reasonable to assume that every shrub needs to contain at least one vertex of a "new" edge to make the graph solvable. Hence, we believe that our results are best possible.

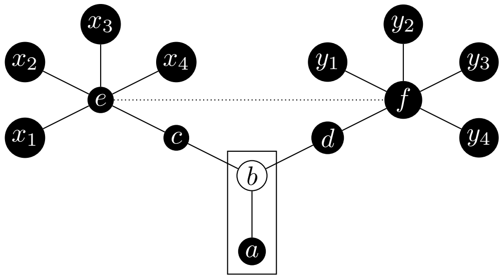


Figure 12: Reindeer package. The corresponding purge is given by $e \cdot \vec{c} \cdot b, y_1 \cdot \vec{f} \cdot e, x_1 \cdot \vec{e} \cdot f, y_2 \cdot \vec{f} \cdot e, x_2 \cdot \vec{e} \cdot f, y_3 \cdot \vec{f} \cdot e, x_3 \cdot \vec{e} \cdot f, y_4 \cdot \vec{f} \cdot e, x_4 \cdot \vec{e} \cdot f, a \cdot \vec{b} \cdot c, f \cdot \vec{d} \cdot b, c \cdot \vec{b} \cdot a$. Jumps 2-8 form a double star purge and can be executed for any number of x_i and y_i (as long as there are equally many).

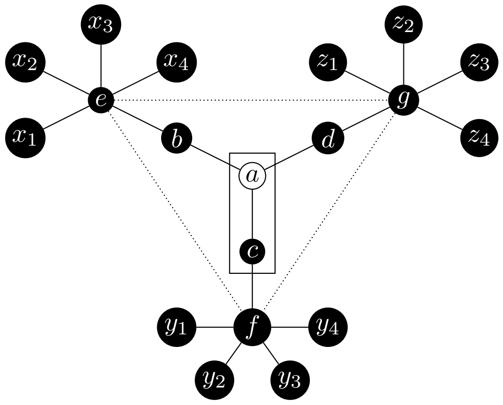


Figure 13: Triblade package. The corresponding purge is given by $e \cdot \vec{b} \cdot a, z_1 \cdot \vec{g} \cdot e, y_1 \cdot \vec{f} \cdot g, x_1 \cdot \vec{e} \cdot f, z_2 \cdot \vec{g} \cdot e, y_2 \cdot \vec{f} \cdot g, x_2 \cdot \vec{e} \cdot f, z_3 \cdot \vec{g} \cdot e, y_3 \cdot \vec{f} \cdot g, x_3 \cdot \vec{e} \cdot f, z_4 \cdot \vec{g} \cdot e, y_4 \cdot \vec{f} \cdot g, x_4 \cdot \vec{e} \cdot f, c \cdot \vec{a} \cdot b, f \cdot \vec{g} \cdot e, e \cdot \vec{b} \cdot a, d \cdot \vec{a} \cdot c$. Jumps 2-12 form in a sense a triple star purge and can be executed for any number of x_i, y_i and z_i (as long as there are equally many).

Beeler and Gray [1] asked how much edge addition can improve the solvability of a graph. They stated this problem in an extremal setting where the insertion of an arbitrary edge was considered. We are able to provide an answer to the related question "Can the difference of the two solvability numbers when adding an (arbitrary) edge be bounded by a constant (for every graph)?": From Proposition 2.2 and Theorem 3.6 we immediately get that adding at most 2 edges to $B_{2,k}$ reduces the solvability number by $2k-1$. Therefore, for arbitrary k we find a graph on $2k+5$ vertices where edge addition reduces the solvability number by at least k. Hence, $\mathrm{Ps}(G \cup e) - \mathrm{Ps}(G)$ can be arbitrarily large.

4 Fool's solitaire number of banana trees

The fool's solitaire number $\mathrm{Fs}(G)$ of a graph G is defined as the maximal cardinality of an associated terminal state when starting with a starting state that consists of exactly one hole. This means, the fool's solitaire number is the number of pegs left when playing as bad as possible.

For some graphs there are results on fool's solitaire, cf. [2, 4, 5, 7]. In particular, Beeler and Rodriguez [5] showed that $\mathrm{Fs}(G) \leq \alpha(G)$ for any graph G, where $\alpha(G)$ denotes the independence number of G. If further for any maximal independent set A the set $V \backslash A$ is an independent set with at least two vertices, we have in fact that $\mathrm{Fs}(G) \leq \alpha(G) - 1$.

For some time it was believed that $\mathrm{Fs}(G)$ always ranges between $\alpha(G) - 1$ and $\alpha(G)$ but nowadays many counterexamples are known. For example, Beeler and Walvoort [7] showed that for certain trees of diameter four, the ratio $\frac{\mathrm{Fs}(G)}{\alpha(G)}$ is asymptotically $\frac{5}{6}$ and that $\frac{5}{6}$ is a lower bound for the ratio $\frac{\mathrm{Fs}(G)}{\alpha(G)}$ for all trees of diameter four.

We will show that banana trees yield another class of graphs where $\mathrm{Fs}(G) < \alpha(G) - 1$ and give a subclass of banana trees where the ratio $\frac{\mathrm{Fs}(G)}{\alpha(G)}$ is asymptotically $\frac{3}{4}$.

We will start with some small cases.

Proposition 4.1. *We have*

(i) $\mathrm{Fs}(B_{1,0}) = \mathrm{Fs}(B_{1,1}) = \mathrm{Fs}(B_{2,0}) = 2$ *and* $\mathrm{Fs}(B_{2,1}) = 3$.

(ii) $\mathrm{Fs}(B_{1,k}) = k + 1$ *for* $k \geq 2$.

(iii) $\mathrm{Fs}(B_{n,0}) = n$ *for* $n \geq 3$.

Proof. (i) This follows from $B_{1,0} = P_3, B_{1,1} = P_4, B_{2,0} = P_5, B_{2,1} = P_7$, and [5, Proposition 3.2] resp. [5, Theorem 3.4].

(ii) We have $B_{1,k} = DS(1,k)$ and the result follows from [4, Theorem 4.2].

(iii) We have $B_{n,0} = K_{1,n}(0;1,\ldots,1)$, hence [7, Theorem 4.1] gives us $\mathrm{Fs}(B_{n,0}) = n - \left\lfloor \frac{n-2\lfloor \frac{n}{2}\rfloor+1}{3} \right\rfloor = n$.

□

Now consider banana trees with $n,k \geq 2$. In the following results we will use the independence number of $B_{n,k}$, so we will determine this first.

Proposition 4.2. *We have* $\alpha(B_{n,k}) = \begin{cases} n(k+1), & k \geq 1, \\ n+1, & k = 0. \end{cases}$

Proof. For $k \geq 1$ a maximal independent set of $B_{n,k}$ is given by the set of leafs together with the set of trunks, which has cardinality $nk+n = n(k+1)$. If $k = 0$, a maximal independent set is given by the set of plants together with the root, which has cardinality $n+1$. □

Proposition 4.3. *For banana trees $B_{2,k}$ with $k \geq 1$ we have*

$$\mathrm{Fs}(B_{2,k}) = \alpha(B_{2,k}) - 1.$$

Proof. The banana tree $B_{2,k}$ is the caterpillar $P_5(k,0,0,0,k)$ (i.e., a path with 5 vertices, where at both end vertices k vertices have been appended). Now [2, Theorem 8] yields $\mathrm{Fs}(B_{2,k}) = 2k + \lfloor \frac{5}{2} \rfloor - 1 = 2k+1 = \alpha(B_{2,k}) - 1$. □

For $n \geq 3$ we will use the duality principle developed in [3] in the form given in [5, Corollary 1.2]. For this, we define the dual of a configuration T to be the configuration T' that has pegs in the vertices where T has holes and vice versa.

Proposition 4.4 (Duality Principle [5, Corollary 1.2])**.** *On a graph G, there exists some vertex $s \in V(G)$ such that, when $S = \{s\}$, there exists some series of jumps that will yield T as a terminal state if and only if the dual T' of T is solvable.*

Theorem 4.5. *For banana trees $B_{n,k}$ with $n \geq 3$ and $k \geq 1$ we have*

$$\mathrm{Fs}(B_{n,k}) = \alpha(B_{n,k}) - \left\lceil \frac{n}{2} \right\rceil.$$

Proof. We use the duality principle. If T is a terminal state with maximal cardinality, then its dual T' is a starting state such that T' contains the complement of a maximal independent set of $B_{n,k}$, has minimal cardinality and is solvable. Hence we are looking for a set T' with these properties. We show that T' can be the set consisting of all plants and the root, together with $\lceil \frac{n}{2} \rceil$ leafs (from different shrubs).

The maximal independent set of $B_{n,k}$ is the set of leafs and trunks with cardinality $n+nk$. Its dual is the set of plants together with the root. This is not solvable, hence we have to add pegs in some leafs or trunks. We want to show that we can solve the graph by adding exactly one peg in the leafs of $\lceil \frac{n}{2} \rceil$ shrubs and that any smaller number of pegs added will not suffice.

First, add $\lceil \frac{n}{2} \rceil$ pegs as described above in $\ell_{i,1}$ for odd i. Then the last shrub will have a peg added if n is odd and will have no peg added if n is even. We begin with the following sequence of jumps:

$$\ell_{1,1} \cdot \vec{p}_1 \cdot t_1, \quad t_1 \cdot \vec{r} \cdot t_2, \quad p_2 \cdot \vec{t}_2 \cdot r.$$

This empties the first two shrubs and leaves the remaining shrubs and the root unchanged. Hence we can inductively empty every forthcoming pair of shrubs. If n is even, only one peg in the root remains and we are done. If n is odd, we conclude with the jumps

$$\ell_{n,1} \cdot \vec{p}_n \cdot t_n, \quad r \cdot \vec{t}_n \cdot p_n$$

and we are done.

It remains to show that we cannot solve the graph when adding less than $\lceil \frac{n}{2} \rceil$ pegs to the complement of the maximal independent set. To this end, add less than $\lceil \frac{n}{2} \rceil$ pegs. Note that for any shrub that has no peg added, we need to move a peg in the trunk to empty the shrub. To do this, we must jump from another trunk (with a peg in it) over the root into the trunk. Therefore, another trunk has to have a peg in it, either by adding a peg there or by adding a peg in a leaf and jumping from the leaf over a plant into the trunk. In each case, we had to add one peg to empty the shrub that has no pegs added and this leaves the root and every other shrub that has had no pegs added unchanged. Thus, for each peg added we can empty at most one shrub that has no pegs added. But if we add less than $\lceil \frac{n}{2} \rceil$ pegs, the number of pegs added is less than the number of shrubs that had no pegs added. Thus the graph remains unsolvable. Hence we need $\lceil \frac{n}{2} \rceil$ pegs implying that the fool's solitaire number is $\alpha(B_{n,k}) - \lceil \frac{n}{2} \rceil$. $\square$

In view of the preceding results we can summarize as follows.

Corollary 4.6. *We have*

$$\mathrm{Fs}(B_{n,k}) = \begin{cases} \alpha(B_{n,k}), & n = 1, \\ \alpha(B_{n,k}) - 1, & n \geq 2, k = 0, \\ \alpha(B_{n,k}) - \left\lceil \frac{n}{2} \right\rceil, & n \geq 2, k \geq 1. \end{cases}$$

This means, that for $k \geq 1, n \geq 2$, the ratio $\frac{\mathrm{Fs}(G)}{\alpha(G)}$ is asymptotically $\frac{\alpha(G) - \frac{n}{2}}{\alpha(G)} = \frac{2k+1}{2k+2}$. If $k = 2$, this gives the same asymptotic as in [7], whereas for $k = 1$ we even get the asymptotic $\frac{3}{4}$.

5 Future Work

In Section 2 we provided the solvability number of banana trees. It would be nice to also have confirmation for the corresponding conjectures in Section 3, which we believe to be true. Moreover, since the number $\mathrm{ms}(G)$ is introduced here, this number should be determined for other graph classes.

It still remains an open problem to characterise solvable trees. Here we studied special trees with diameter 6. Since there are general results about trees with diameter 4 (cf. [7]), one could try to characterise the solvable trees with diameter 6. To get closer to such a characterisation one could examine generalised banana trees: Let $B_{n,k_1,k_2,\ldots,k_n}$ denote the banana tree with n shrubs containing $k_1, k_2, \ldots, k_n$ leafs respectively. Most results about generalised banana trees should be obtainable via the same methods we presented here.

Another very interesting family of problems emerges when defining $\mathrm{Ps}(G)$, $\mathrm{ms}(G)$, $\mathrm{Fs}(G)$ depending on the starting hole in the following sense: Let $\mathrm{Ps}(G, v)$ be the smallest number of pegs in a terminal state when starting with a hole in $v \in V$. How large can $|\mathrm{Ps}(G, u) - \mathrm{Ps}(G, v)|$ become for $u, v \in V$? Similar questions can be asked for the corresponding definitions of $\mathrm{ms}(G, v)$ and $\mathrm{Fs}(G, v)$.

Acknowledgements

Some of the ideas and conjectures of this article emerged in a seminar held at the University of Hildesheim in 2019. The authors thank all participants for their input.

References

[1] R.A. Beeler and A.D. Gray, Extremal results for peg solitaire on graphs, *Bull. Inst. Combin. Appl.*, **77** (2016), 30–42.

[2] R.A. Beeler, H. Green, and R.T. Harper, Peg solitaire on caterpillars, *Integers*, **17** (2017), Art. G1.

[3] R.A. Beeler and D.P. Hoilman, Peg solitaire on graphs, *Discrete Math.*, **311(20)** (2011), 2198–2202.

[4] R.A. Beeler and D.P. Hoilman, Peg solitaire on the windmill and the double star graphs, *Australas. J. Combin.*, **52** (2012), 127–134.

[5] R.A. Beeler and T.K. Rodriguez, Fool's solitaire on graphs, *Involve*, **5(4)** (2012), 473–480.

[6] R.A. Beeler and C.A. Walvoort, Packages and purges for peg solitaire on graphs, *Congr. Numer.*, **218** (2013), 33–42.

[7] R.A. Beeler and C.A. Walvoort, Peg solitaire on trees with diameter four, *Australas. J. Combin.*, **63(3)** (2015), 321–332.

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

[9] W.C. Chen, H.I. Lu, and Y.N. Yeh, Operations of interlaced trees and graceful trees, *Southeast Asian Bull. Math.*, **21(4)** (1997), 337–348.

[10] J. Engbers and C. Stocker, Reversible peg solitaire on graphs, *Discrete Math.*, **338(11)** (2015), 2014–2019.

[11] S. Loeb and J. Wise, Fools solitaire on joins and Cartesian products of graphs, *Discrete Math.*, **338(3)** (2015), 66–71.