hexagon:2611.01349v1

THE FULL-DEGREE SPANNING TREE PROBLEM ON SQUARE GRIDS

MICHAEL PACIULLO

Abstract. We determine the maximum number of full-degree vertices in a spanning tree of the $n \times n$ square grid for every positive n, and give constructions attaining the maximum. The formula separates into three cases according to n mod 3, with the optimum one below the rounded certificate bound when $n \equiv 35 \pmod{42}$. The proof combines explicit forests, rational upper-bound certificates, and a separate exclusion of equality on the exceptional progression. The graph-theoretic result is formalized in Lean 4, with finite certificate checks evaluated natively.

1. Introduction

Let G_n be the graph on $\{0, \dots, n-1\}^2$ in which two vertices are adjacent if their coordinates differ by 1 in exactly one coordinate and agree in the other. A vertex v has *full degree* in a spanning tree T of G_n if $\deg_T(v) = \deg_{G_n}(v)$. In particular, a corner or boundary vertex can have full degree. Write $P(n)$ for the largest number of such vertices.

Bhatia, Khuller, Pless, and Sussmann studied the full-degree spanning tree problem on general graphs [1]; it remains NP-hard on bipartite planar graphs [2]. Acquaviva's thesis, advised by Deepak Bal, gives exact formulas for grids with two or three rows and proves that the optimum on an $m \times n$ grid is asymptotic to $mn/3$ as both dimensions tend to infinity [3, Chapter 3]. Acquaviva and Bal also studied random regular graphs [4].

Theorem 1. *For positive $n = 3k + s$, with $s \in \{0, 1, 2\}$, the value of $P(n)$ is given by*

s	*Range*	$P(n)$
0	$k \geq 1$	$3k^2 + 3k - 2$
1	$k \geq 5$	$3k^2 + 5k + \lfloor (3k - 13)/10 \rfloor$
2	$k \geq 6$	$3k^2 + 7k - \lfloor (k - 5)/7 \rfloor - \mathbf{1}_{n \equiv 35 \pmod{42}}$

Here $\mathbf{1}_{n \equiv 35 \pmod{42}}$ equals 1 on the indicated progression and 0 elsewhere. The remaining values are

n	1	2	4	5	7	8	10	11	13	14	17
$P(n)$	1	2	8	11	22	27	42	49	68	77	110.

2020 *Mathematics Subject Classification.* Primary 05C05; Secondary 05C10, 05C35.

Key words and phrases. grid graph, full-degree vertex, spanning tree, computer-assisted proof.

Lucena and da Cunha tested exact optimization algorithms on rectangular grids [5, Section 4.1 and Supplementary Table 5]. They certified $P(10) = 42$ and reported $149 \le P(20) \le 155$, without certifying the latter optimum. Theorem 1 gives $P(20) = 150$ and determines the exact value at every square size.

For instance, at $n = 35$ the rounded upper bound for $s = 2$ gives 440, but the optimum is 439. The exceptional correction comes from the structure required for equality in the upper bound. Sections 3–5 prove the three parts of the theorem. Appendix A locates the formal proof and the implementation of the constructions.

2. Forced forests and certificates

For $F \subseteq V(G_n)$ let

$$E(F) = \{uv \in E(G_n) : u \in F \text{ or } v \in F\}.$$

We call the vertices of F *marked* and say that F is *feasible* if $(V(G_n), E(F))$ is a forest. The following criterion is the square-grid specialization of the forest formulation of Bhatia et al. [1, Section 4.1 of the technical report], also stated by da Cunha et al. [6, Proposition 1].

Lemma 2. *A set F is feasible if and only if some spanning tree of G_n has full degree at every vertex of F. In particular $P(n) = \max\{|F| : F \text{ is feasible}\}$, and every subset of a feasible set is feasible.*

Proof. A tree having full degree at every vertex of F contains $E(F)$. Conversely, if $E(F)$ is acyclic, repeatedly join two of its components by a grid edge until there is one component. Such an edge exists at every step by connectivity of G_n; adding it creates no cycle. The resulting tree contains $E(F)$. The maximum identity follows in both directions, and deleting marks only removes forced edges. □

The last assertion allows an upper-bound proof to exclude a single cardinality: to show $P(n) \le b$, it suffices to exclude a feasible set of size $b + 1$. A lower bound needs only a feasible set of the desired size. The completion step in the proof of Lemma 2 then gives an actual spanning tree.

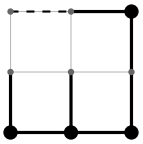


Figure 1. Four marked vertices in G_3. Thick solid edges are forced. The dashed edge joins the two components to give an optimal spanning tree.

Here are two simple restrictions used to motivate the more detailed certificates. First, a unit square must have a side with both endpoints unmarked: otherwise all four of its sides are forced and form a cycle. In particular, opposite corners cannot both be marked. This pair is an edge in the conflict graph of Lucena and da Cunha [5, Section 2]. Second, define the boundary weight $b(F) = \sum_{v \in F}(4 - \deg_{G_n}(v))$. For $n \ge 2$ this counts each marked noncorner boundary vertex once

and each marked corner twice. For a nonempty feasible set F, the induced graph $G_n[F]$ is a subgraph of the forced forest. If it has r components, then

$$|E(F)| = 4|F| - b(F) - |E(G_n[F])| = 3|F| - b(F) + r.$$

Since $|E(F)| \le n^2 - 1$ and $r \ge 1$, we get $3|F| \le n^2 - 2 + b(F)$. This bound alone does not resolve the boundary contribution with the accuracy needed in Theorem 1.

2.1. **How the finite certificates give uniform bounds.** Da Cunha et al. [6] and Lucena and da Cunha [5] use valid inequalities in exact algorithms for individual instances. Here we use counts of local configurations to obtain bounds that hold uniformly over infinite families of square sizes. For example, in each 3×3 window look at its upper-left unit square. The number of windows where that square has a given four-bit pattern equals the sum of the counts of all nine-bit window patterns with that upper-left restriction. Both counts run over the same window origins. A square near the boundary may not be the upper-left square of any window, so one cannot substitute a count over all squares. The proof keeps separate position types to account for such boundary differences.

Lemma 3 (Certificate inequality)**.** *Suppose a class of feasible sets on side n gives nonnegative count vectors x with score $c^{\mathsf{T}}x$. Let $Ax \le b(n)$ and $Hx = d(n)$ hold for all these vectors. If rational vectors y, z satisfy*

$$y \ge 0, \qquad A^{\mathsf{T}}y + H^{\mathsf{T}}z \ge c,$$

then every set in the class has score at most $y^{\mathsf{T}}b(n) + z^{\mathsf{T}}d(n)$.

Proof. Multiply the coefficient inequality by x, then use $Ax \le b(n)$ with $y \ge 0$ and use $Hx = d(n)$ with z unrestricted. □

A finite certificate can therefore be used at arbitrarily large n once two obligations are met: the types and incidences in the coefficient check cover the larger board, and the normalization counts and other right-hand sides have been evaluated at the larger size. The latter counts may be polynomial even though the optimum of the relaxation is unknown. The bounds below use fixed certificates with computed normalization counts; rounding is applied to the resulting bound on an integer score.

3. Sides divisible by three

Theorem 4. *For every integer $k \ge 1$,*

$$P(3k) = 3k^2 + 3k - 2.$$

Construction. For $k \ge 7$ the construction specifies the unmarked set U_k by placing two top cap rows, the row word W_k, and two bottom cap rows in that order. Each pattern specifies the unmarked sites of its row. An ordinary row R_a uses column $3i + 1$ in every triple $0 \le i < k$, together with $3i + 2$ for $i \le a$ and $3i$ for $i > a$. A seam row S_a is R_a with column $3a + 2$ removed. The initial word is

$$W_7 = R_4\, R_3\, R_2\, S_2\, R_1\, R_0\, S_0\, R_0\, S_1\, R_1\, R_2\, S_3\, R_3\, S_4\, R_4\, R_5\, R_6.$$

The recursion is $W_{k+1} = R_{k-2}S_{k-2}W_kR_k$, with all patterns read at the new width. The word has $3k - 4$ rows, of which $k - 2$ are seams. The cap rows, in order from

top to bottom, have no unmarked sites, all sites unmarked, columns $1, \dots, 3k-1$ unmarked, and just column 1 unmarked (in zero-based coordinates). Hence

$$|U_k| = 6k + 2k(3k-4) - (k-2) = 6k^2 - 3k + 2.$$

To certify that $F = V(G_{3k}) \setminus U_k$ is feasible, use the planar dual, with one vertex for each unit face and one for the exterior. Keep a dual edge when both endpoints of its primal edge are in U_k. These are precisely the dual edges complementary to $E(F)$. If $E(F)$ contained a cycle, the faces inside it would have no path to the exterior in this dual.

The construction proves dual connectivity by maintaining paths from the interior face rows to a distinguished face adjacent to the exterior. The path invariant uses a restricted dual graph whose edges persist under widening. It is checked at $k = 7$. In the step from k to $k+1$, the persistent paths are translated two rows down; a bridge joins the translated distinguished face to the new one. The new top and appended interior rows attach to the translated interior, and the new right slab has paths to the new distinguished face. The two outer face rows are handled by the boundary attachment lemmas. This gives an acyclic $E(F)$ with $9k^2 - |U_k|$ marks. The count is `uResidueZero_card`; the invariant and its acyclicity consequence are `persistentResidueZeroCertificate` and `forced_compl_uResidueZero_isAcyclic` in the `ResidueZero` construction files. For $1 \le k < 7$ the individual constructions are collected in `matchingDiagonalConstructions_residueZeroFinite`.

Local upper certificate. Let $n = 3k$ with $k \ge 4$, and let $E(F)$ be a forest. Orient its edges toward one root per component, choosing the upper-left corner as the root of its component. A *star* records the mark bit of its central vertex, the four neighbour bits (zero off the board), and the direction of its outgoing forced edge, or a root symbol if there is no outgoing edge. A *patch* records the nine mark bits in a 3×3 window (Figure 2). A star or patch is *physical* if it is read at a specified location from the actual marked set and, for a star, the chosen forest orientation. The certificate also checks candidate records satisfying local tests, without requiring that each candidate extend to a whole forest. A physical star cannot have marked neighbours in consecutive compass directions: their marks would force a unit cycle. Its outgoing direction, if present, must use a forced edge. Similarly, a physical patch cannot contain any of the thirteen forbidden submasks in `maskOK`: each covers all edges of a cycle inside the patch.

The certificate assigns integer charges to stars and patches, summed over the eight symmetries of the square. A star charge includes a normalization term, a root term, the objective term $-\lambda$ times its mark bit per symmetry, and signed incidence terms. A patch charge contains a normalization term and signed incidence terms. There are three types of matching data: the marks and arrow on a grid edge read from its two ends; the mark pattern at a star restricted to a patch that contains it; and the mark bits in the common strip of two adjacent patches. In each case the two readings of the same data carry opposite charges. Thus their sums vanish on the physical grid, separately in each symmetry. Root charges are nonpositive. Let $B(n)$ be the normalization budget in one view plus the fixed allowance from the chosen corner. At scale $\lambda = 4373984$, the sums S and T of star and patch charges satisfy

$$0 \le S + T \le 8B(n) - 8\lambda|F|.$$

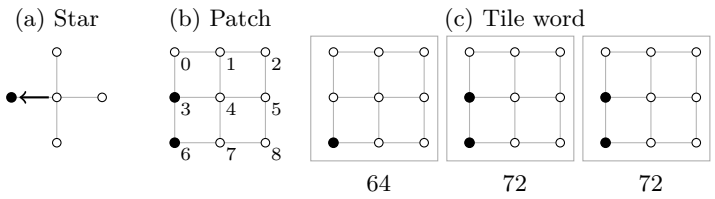


FIGURE 2. Local records. Filled circles are marked. (a) The central vertex points west along a forced edge. (b) The patch at the same location agrees with the star on their five shared marks or blanks. Its marked positions 3 and 6 give mask $2^3 + 2^6 = 72$. (c) Patch windows may overlap; the tiles here are disjoint. Read their masks from left to right to obtain the tile word $64, 72, 72$.

Here are the finite and geometric obligations behind this inequality.

(1) The eight-view charge is nonnegative at every admissible star and patch on sides $12, 15, 18, 21$. At each star site the replay ranges over the two values of the central mark bit, 16 neighbour masks and 5 direction symbols; at each patch origin it ranges over all 512 masks. The admissibility tests above determine the entries to check.
(2) On side $21 + 3j$, $j \geq 0$, star and patch locations can be represented at side 21 without changing any charge lookup or admissibility test. In each axis the representative preserves the distance to the nearer end in the six-position boundary bands and the residue modulo three in the interior. The argument applies to both the site square and the square of patch origins: it preserves the reflected labels, the nearby incidence queries and the off-board sentinel.
(3) The incidence cancellations and the corner bound hold on the original board. The normalization budget is computed there as well. Adding three positions to an axis inserts one of each interior phase in the label sum; taking the product of the two axis sums gives a quadratic in j. Its three coefficients give

$$B(21 + 3j)/\lambda = 166 + 45j + 3j^2.$$

The direct budgets at $12, 15, 18$, divided by λ, are $58, 88, 124$, respectively.

In Lean, `starChecks` and `patchChecks` perform the local replays; `star_nonneg` and `patch_nonneg` use the transport. The physical inequality uses `physical_marginals_zero` and `cancellation_of_marginals`. The budget identity is `budget_eq_score`. Dividing by 8λ gives the required upper bound throughout $k \geq 4$.

The three smaller upper bounds. The 3×3 maximum is 4 (`maxPreserved_three_three`). At side 6, tile the board by four disjoint 3×3 patches; the admissible-mask check bounds each by four marks. At side 9, a Boolean certificate excludes a set of 35 marks. The clauses count mark bits and require an edge with both endpoints unmarked on each listed cycle. The list includes rectangles and additional perimeters checked to be simple grid cycles. Every forest with 35 marks would give a satisfying assignment; the LRAT replay proves unsatisfiability. These proofs are

`ResidueZero.maxPreserved_six` and `ResidueZero.Nine.maximum_eq`. With the lower witnesses, the final equality is `ResidueZero.maxPreserved_three_mul`.

4. Sides congruent to one modulo three

Theorem 5. *For $k \geq 1$,*

$$P(3k+1) = C(k), \qquad C(k) = 3k^2 + 5k + \max\left\{0, \left\lfloor \frac{3k-13}{10} \right\rfloor\right\}.$$

Also $P(1) = 1$.

The comb construction. For $k \geq 5$, put $n = 3k+1$ and $R = n - 8$, with rows and columns numbered from zero. Mark the first and last rows in full and mark columns $0, 3, \ldots, 3k$ in rows $2, \ldots, R+1$. Rows $R+2, \ldots, R+5$ are given by a word of four-bit column masks specifying marked sites; all remaining sites are unmarked. A frontier sweep keeps the component labels of the active ports. Every new edge is inserted only between distinct components, and ports that are retired disappear from the frontier while their edges remain in the graph. The physical sweep includes the fixed comb and both boundary rows and proves that the resulting forest contains all of $E(F)$. Figure 3 shows the marks at the smallest width of the uniform construction.

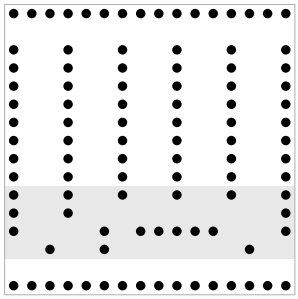


Figure 3. The 100 marked vertices in the comb on side 16.

In each mask, bits $0, 1, 2, 3$, from least to most significant, mark rows $R+2, R+3, R+4, R+5$, respectively. Start with mask 7 and repeat the following 30-column block from left to right:

$$0, 8, 3, 0, 12, 1, 0, 14, 0, 2, 8, 1, 4, 4, 5,$$
$$4, 4, 1, 8, 2, 0, 14, 0, 1, 12, 0, 3, 8, 0, 7.$$

This block returns to the same frontier state and adds 33 marks. Writing $k = 10q+r$ with $3 \leq r \leq 12$, use q blocks and append the length-$3r$ word `CombWord.ending`(r) from the table of ten endings. The ending has $3r + \lfloor (3r+7)/10 \rfloor$ marks, and the first column has three. Thus the word has length n and its mark count, together with the fixed part, is

$$2n + R(k+1) + 3k + 3 + \left\lfloor \frac{3k+7}{10} \right\rfloor = C(k).$$

The word calculation is `CombWord.exists_cap`; its physical interpretation and score are assembled in `squareMaskScore_comb`. The small lower bounds are included in `squareScore_lower_residue_one`.

Upper bound with a pitch line. A pitch row has marks at exactly the columns divisible by three; a pitch column is the transpose. Fix a pitch row with $k \geq 5$. On each side its bonus b counts marks strictly on that side, less $k+1$ per row. Cut just above the pitch row when it has a predecessor. The $k+1$ vertical edges above its marks are the *spokes*; their upper and lower endpoints are the respective caps' terminals, or teeth. Only spokes are forced across the cut: a mark between teeth in the preceding row would force a unit-square cycle with the nearest pitch-row mark.

In each cap, two terminals belong to the same class if they are joined by a forced path in that cap. If there are c_+ and c_- classes, the two join ranks are $k+1-c_+$ and $k+1-c_-$. Cut the spokes in a spanning tree extending the forced forest. Each of the $k+2$ resulting components contains a spoke endpoint, and every terminal class stays within one component. Thus $c_+ + c_- \geq k+2$, so the ranks sum to at most k. If the pitch row is the top row, the upper cap is absent and has rank and bonus zero. The lower cap has at least one terminal class, so its rank is at most k.

The cap certificate bounds the bonus b in terms of the join rank j:

$$10b - 7j \leq 3k - 13 \quad (j < k), \qquad b \leq k-1 \quad (j = k).$$

For the certificate, both caps include a copy of the pitch row. Putting that row back on the upper cap does not merge terminal classes: each forced component along the row contains one tooth and meets the upper cap only by its spoke. Nor does it change the bonus, since the added row has exactly $k+1$ marks. Padding the resulting caps preserves the terminal partitions. Orienting each forest gives an outgoing-edge record at each site. A *slit* is a vertical cut between columns; its record lists, row by row, the two marks and direction information on the horizontal edge crossing the cut. Weighted traces read successive slits while sweeping across triples of columns. The bill B of the two guarded half-traces satisfies $4b = 2(k+1)+2j-B$. The orientation retains one terminal tooth from each terminal class. Thus the number retained is $t = k+1-j$, and the required bill bounds are $5B \geq 4t+32$ for $t \geq 2$ and $B \geq 6$ for $t = 1$. These follow by excluding low-bill pairs of traces meeting at the reflected slit. The automata record the normalized budget $B_{\mathrm{prefix}} - \lfloor 4t_{\mathrm{prefix}}/5 \rfloor$ and distinguish retained counts $0, 1$ and the ten classes represented by $2, \ldots, 11$. The replay checks all affordable budget splits in the guarded-prefix extensions, from the slice immediately below the allowed budget floor through normalized budget 4. At budget 5 and above the envelope accepts all words. It checks extensions from prefix lengths $0, \ldots, 24$ and the stationary extension from the length-25 table to itself. The reflected products use left length 2 and right lengths $4, \ldots, 25$. For longer right halves the stationary table applies, and adding ten retained teeth shifts both the raw cutoff and its normalization by eight. Product closure certifies the exclusions for words of arbitrary length, so the replay places no bound on cap height.

The total mark count is $n(k+1)$ plus the two bonuses. If both ranks are less than k, the bonuses sum to at most $\lfloor (13k-26)/10 \rfloor$, which is at most $k-1+\lfloor (3k-13)/10 \rfloor$. If one rank is k, the other is zero and the two cap inequalities give the latter bound directly. This proves the upper $C(k)$. The physical argument includes the transposed cut and is assembled in `PitchCut.pitch_upper_verified`, using `CapSite.tables_verified` and `Slit.Replay.replay_verified`.

Upper bound without a pitch line. For $k \geq 19$ there are two tail charts, at $n = 58+6u$ and $n = 61+6u$, $u \geq 0$. To apply a chart to a physical forest, count stars and 3×3 patches, and record data from the 4×4 windows whose starting coordinates

are multiples of three. A *rank face* records the sixteen mark bits in one such window and the partition of its four corners into connected classes. Connectivity is taken in the graph on the whole board whose edges are forced by marks on rows or columns of index divisible by three. A connecting path may leave the window and return. The chart also counts consecutive local records along these rows and columns, with auxiliary variables for the entire lines. Each column of the certificate matrix represents a count variable, and each row gives a constraint on the counts. The resulting count vector has objective $8|F|$ in the chart's normalization.

There are two parts to the certificate. The dual replay checks the signs of the row multipliers, column domination on all non-fixed columns, and the three coefficients of the weighted right-hand side; the tail charts have zero positive residual. The physical proof checks that the count vector is nonnegative, vanishes on fixed columns, and satisfies the chart rows. It verifies the root, line-flow, line-letter and normalization rows, then proves that the remaining equality rows have zero signed sum. The latter rows are exhausted by star-only, patch-only, star–patch, patch–face and face–line overlaps. In each case the two descriptions of an overlap are drawn from the same board. The base replays identify the local coefficients, while transport of the keys and cancellation on the physical board handle all widths. In particular the normalization right-hand sides count the fine-label classes at the actual width, using their exact quadratic counts in u. No-pitch enters the line checks; the tail proof needs absence of pitch rows and columns only at indices $0, 3, \dots, 3k$.

The weighted sum gives

$$|F| \le 3k^2 + \frac{21}{4}k - \frac{79}{64}.$$

The unfloored comb expression exceeds this bound by $k/20 - 21/320 > 0$ throughout the tail, so integrality gives $|F| \le C(k)$. The dual checks are `verified58` and `verified61` in `NoPitch.Dual`. The physical row assembly uses `upper_of_face_overlaps`, `face_patch_tail` and `face_line_tail`. `maxPreserved_residue_one_tail` combines this with the pitch branch and the lower witness.

Finite widths and conclusion. The small cases through side 13 give the five values in Theorem 1. For sides $19, 22, 25, 52$, the no-pitch branch uses individual physical charts and finite dual replays, with the uniform pitch bound covering the other branch. In these finite replays the weighted bound, including the allowance for any positive column residual, is strictly below $C(k) + 1$. The remaining sides $16, 28, 31, 34, 37, 40, 43, 46, 49, 55$ use unrestricted finite stencils in `FiniteOne`. They sum maxima over local star and patch data with cancelling incidence charges; at side 43 three stencils cover the cases obtained by splitting at seven marks in each of the symmetry orbits of $(6, 9)$ and $(9, 15)$. Once the first orbit has at least eight marks, the second count is unnecessary. This exhausts the residue-one sides below 58. The complete case split is `GridGraphs.maxPreserved_residue_one`.

5. Sides congruent to two modulo three

For $n = 3k + 2$ and $k \ge 6$, put

$$T(k) = 3k^2 + 7k, \qquad U(k) = T(k) - \left\lfloor \frac{k-5}{7} \right\rfloor.$$

The residue-two formula is

$$P(3k+2) = U(k) - \mathbf{1}_{k \equiv 11 \pmod{14}}. \tag{1}$$

The six smaller sides $2, 5, 8, 11, 14, 17$ have values $2, 11, 27, 49, 77, 110$, respectively. We give the lower construction and the unrestricted upper inequality first, then explain how equality in the integer upper bound is excluded on the exceptional progression. The finite checks used below are identified by their Lean interfaces.

5.1. **Lower constructions and the period.** The correction in $U(k)$ has seven rounding phases. The construction also distinguishes the parity of k: the advancing rows stop differently on sides $6a+2$ and $6a+5$. The pair ($k \bmod 7, k \bmod 2$) repeats after 14 steps in k, or 42 in the side length. Table 1 gives a sufficient construction period.

$k \bmod 7$	k even	k odd
0	44, delayed right front	23, ordinary zipper
1	68, delayed right front	47, upper-left repair
2	50, delayed right front	71, upper-left repair
3	74, delayed right front	53, upper-left repair
4	56, alternating bands	35, rail ($U(k) - 1$)
5	38, delayed right front	59, ordinary zipper
6	62, delayed right front	41, ordinary zipper

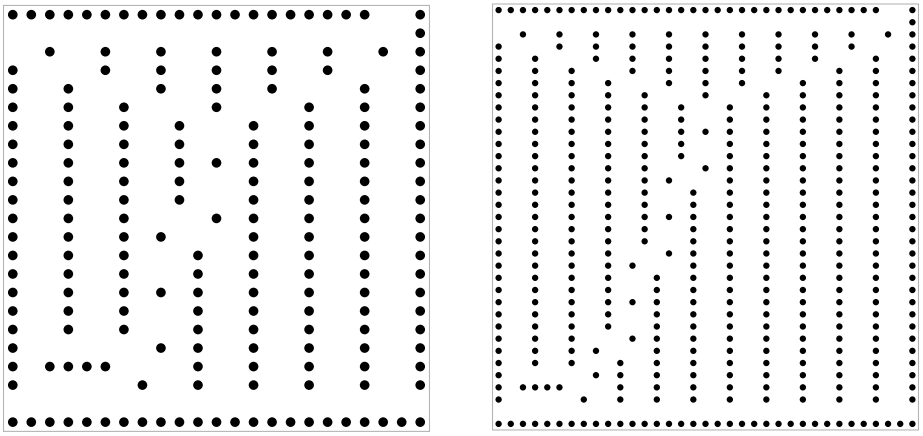
TABLE 1. Starting sides b for the constructions on $n = b + 42t$, $t \geq 0$. The marked set has size $U(k)$ except in the indicated odd cell.

The ordinary zipper uses rows of marks spaced three columns apart. The two outer arrays of vertical lines advance towards the middle; after the advancing rows, seven-row bands move the junction by one pitch of three columns at a time. The five bottom rows close the construction, with the last band possibly cut short. On adding 42 to the side, the ordinary construction has seven more advancing rows and five more seven-row bands. The upper-left repair shifts the marks along a segment of the advancing front and inserts a mark at its end. The delayed-right schedules omit the final right-front advance and use the even-width version of the repair, with an additional waiting row in the phase starting at 38. The alternating version moves the tip back and forth in successive bands and uses its own bottom closure. The exceptional rail is an ordinary layout with the ending prescribed for its phase. These are physical marking rules; their row predicates and end corrections are specified in `LowerZipper`. Figure 4 shows the ordinary zipper and the exceptional rail at their first widths.

Proposition 6 (Construction certificate)**.** *For every $k \geq 6$ there is a marked set F with $E(F)$ acyclic and*

$$|F| = U(k) - \mathbf{1}_{k \equiv 11 \pmod{14}}.$$

Proof. At $k = 6$ the separate side-20 witness has 150 marks. For $7 \leq k \leq 38$, the finite construction replay in `ResidueTwoLower` covers the stated scores, including the separate initial masks at $26, 29, 32$, before the corresponding periodic layouts. For $k \geq 39$, each side is in exactly one chart of Table 1. Direct counts of the row pieces, including the truncated band and the bottom closure, give the asserted size (`marks_card_eq_squareScore`).



Ordinary zipper, $n = 23$ Exceptional rail, $n = 35$

FIGURE 4. Optimal marked sets of sizes 196 and 439.

For acyclicity, the uniform certificate uses layers of directed forced edges, where each forced edge is admitted in both directions at the initial layer. The terminal layer is empty. At each successive layer, a straight continuation in that layer implies membership of the preceding edge in the same layer, whereas a perpendicular continuation in the previous layer implies membership of the preceding edge in the successive one. Let $r(e)$ be the number of intermediate layers containing the directed edge e. For consecutive edges e, e' on a directed traversal,

$$r(e') \le r(e) \quad \text{on a straight step,} \qquad r(e') < r(e) \quad \text{on a turn.}$$

The strict decrease follows because the perpendicular implication uses the initial layer, which contains e', and the terminal layer, which is empty. Every simple grid cycle has a turn. Going once around such a cycle would therefore give $r(e) < r(e)$.

The certificate checks are coverage of all physical forced edges by the initial layer and the straight and turn implications on all fourteen charts. They are width-parametric region checks, with the actual marking rules furnishing the physical coverage. The corresponding Lean results are `table_input_coverage` and `charts_closed`; the exported `zipper_routes` expresses the conclusion dually, as reachability of the exterior from each bounded face using edges dual to unforced grid edges. Together with the finite replay these give `squareScore_lower_residue_two`. □

5.2. **The unrestricted overlap stencil.**

Lemma 7 (Period-three upper certificate)**.** *For $n = 3k + 2 \ge 20$, every F with $E(F)$ acyclic satisfies*

$$|F| \le R(n) := \frac{n^2}{3} + \frac{20n}{21} - \frac{67}{42}, \qquad \lfloor R(n) \rfloor = U(k). \tag{2}$$

Proof. Orient each component of the forced forest towards a root, choosing the northwest corner as root of its component. Use the star and patch records defined in Section 3. In any patch read from the forest, the internal forced edges are acyclic. Adjacent stars agree on the marks at the two endpoints and on the status of their

common edge (unforced or oriented in one of the two directions). Each star and patch agree on the vertex and neighbor bits they both see. Adjacent patches agree on their six shared bits.

The rational stencil assigns signed prices to these three agreement relations, as well as normalization prices at stars and patches. Let c count the corners that are *sinks* (vertices with no outgoing edge). Each corner maps to northwest in two of the eight square isometries. Thus four times the average northwest-sink indicator is $4(2c/8) = c \geq 1$, since we rooted a component at northwest. This row has multiplier $-9/14$. We apply the eight-image average to the price of each physical occurrence, carrying its labels and arrow along with the image. Let s_v be the resulting star price (including its averaged sink charge) minus its central mark bit, and let p_A be the resulting patch price. These are the star and patch *slacks*; the certificate checks that each is nonnegative. The exact ledger is

$$R(n) - |F| = \sum_v s_v + \sum_A p_A + \frac{9}{14}(c-1). \tag{3}$$

In this identity, signed agreement terms cancel in pairs at physical incidences. The star sum includes $-9c/14$ in actual sink charges. The fixed bound $R(n)$ uses the charge $-9/14$ for one corner sink. The last term in the ledger is the difference between the fixed and actual charges. This averaging requires no symmetry of F.

Here is the scope of the finite price checks. For an axis of length L, the index i has type i if $i < d$, type $2d + 3 + i - L$ if $i \geq L - d$, and type $d + ((i - d) \bmod 3)$ otherwise. Use $L = n$ for stars and $L = n - 2$ for patch origins. The normalization types have depths 6 and 5, respectively, while agreement prices use depth 2. The finer types determine the coarse types at the required offsets and determine missing incidences at a wall. Reflection exchanges the end types and acts on bulk phases according to L mod 3. Thus the applicability and the eight-image price of a local configuration transport from a representative of its type. The Lean replay checks $s_v, p_A \geq 0$ at sides 20 and 23, transporting from 23 to $23 + 3j$. The constant part of the ledger is the sum of the normalization prices, minus $9/14$. End types occur once per axis; each bulk phase gains one occurrence for each increment of three in n. Summing the Cartesian products of those multiplicities at $n = 20 + 3j$ gives $2111/14 + (300/7)j + 3j^2 = R(n)$. The theorem `physical_rational_upper` in `StencilUpper` assembles these checks with the physical cancellation.

Since $c \geq 1$, the ledger gives the rational upper. Substituting $n = 3k + 2$ gives $R(n) - T(k) = (23 - 2k)/14$. If $k - 5 = 7a + r$ with $0 \leq r \leq 6$, this is $-a + (13 - 2r)/14$, which proves the floor identity. □

5.3. **What attaining the integer upper would force.** For the rest of the upper proof let $n = 35 + 42u$, $k = 11 + 14u$, $u \geq 0$, and suppose that $E(F)$ is acyclic with

$$|F| = U(k) = 440 + 1020u + 588u^2. \tag{4}$$

Here $U(k) = \lfloor R(n) \rfloor$ is the candidate score we will rule out. It lies $1/14$ below $R(n)$; the optimum lies one vertex below $U(k)$. Sides 35 and 56 illustrate why rounding alone cannot explain the exception: $R(35) = 440 + 1/14$ and $R(56) = 1097 + 1/14$, yet the construction attains 1097 at side 56, while 440 is impossible at side 35. We call a star or patch with positive slack a *defect*, and a perimeter edge with two unmarked endpoints a *gap*.

Lemma 8 (Exceptional equality forcing)**.** *Under* (4)*, all four corners are marked and all forced edges lie in a single tree component. Rooting that component at the northwest corner gives exactly one defect, of slack* $1/14$*. Every side is marked at distances* $3, \dots, n-4$ *from either end, and no site at distance one from the perimeter is marked. At each corner, read the two sites on the horizontal arm and then the two on the vertical arm, in order of increasing distance from the corner on each arm (Figure 5). This gives one of the words*

$$0011, \qquad 0111, \qquad 1100, \qquad 1101.$$

There are either one or two gaps. With one gap there is no inward perimeter arrow. With two gaps there is exactly one inward perimeter arrow, and its star is the unique defect.

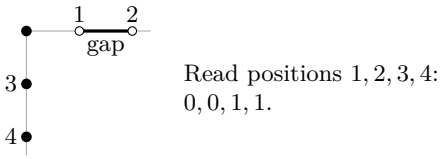


FIGURE 5. Reading a corner word, shown at the northwest corner. Filled circles are marked. The marked corner is outside the word; the thick edge between positions 1 and 2 is the gap.

Proof. A second corner sink would contribute $9/14$ to (3). If a different component contains a mark, orient it towards that mark. A marked noncorner sink costs more than $1/14$, by a star-table check, and the corner-sink penalty handles a marked corner in that component. Hence all marks and thus all forced edges are in the chosen corner component. All four corners belong to it: another corner outside it could itself be chosen as a second corner sink. If a corner is unmarked, it has a marked neighbor because it belongs to the nontrivial component. Rerooting there produces an unmarked corner sink with a marked neighbor, whose checked slack also exceeds $1/14$. These tests are the physical implications `all_active_reachable` and `corners_marked`.

Root the component at northwest again. It contains all four corners, so $c = 1$. The checks in `ExceptionalPropagation` restrict the occurrence slacks, in units of $1/14$, to $0, \frac{1}{2}, \frac{3}{4}, 1$. They use ranked messages on matching star–patch incidence labels: an occurrence outside the allowed set would force a positive message, and matching across incidences would then force a strictly descending sequence of positive ranks. A contribution of $3/4$ would leave $1/4$ of the total weight one, less than any positive contribution. Thus the only profiles are a full defect or two half-defects.

The parity check for the latter profile assigns an auxiliary signed integer weight to each occurrence. The sum on the physical board is one: the agreement contributions cancel and one constant boundary term remains. Every local occurrence of slack zero or one half is checked to have even auxiliary weight for this rooting. Thus the two-half-defect profile is impossible (`single_defect`, using `physical_sum_one`).

A second ranked incidence check, using only stars and patches of zero or full slack and the marked corners, gives the side, next-ring and corner conditions in the statement. It also checks that a perimeter star has full slack exactly when its arrow

points inward. Each listed corner word leaves every perimeter vertex marked or adjacent to a mark on the perimeter. All $4n-4$ such vertices therefore lie in the root component. If there are g gaps, $4n-4-g$ of their arrows orient forced perimeter edges. There are $4n-5$ outgoing arrows from perimeter vertices altogether: the northwest root is the only perimeter vertex without one. The remaining $g-1$ arrows point inward. Each inward arrow gives a distinct full-slack perimeter star. Since there is only one defect, there is at most one inward arrow. Hence the perimeter has one gap with no inward arrow, or two gaps with one inward arrow. This is the ledger used in `frame_of_score`. □

5.4. **Excluding the exceptional frames.**

Proposition 9 (Two-gap exclusion)**.** *For every $u \geq 0$, no forest of score* (4) *has two gaps.*

Proof. Call a corner *open* if its word is 0011 or 1100: the two zero bits on one arm give a gap. The other words, 0111 and 1101, are *closed*. The two gaps are thus at distinct corners. The physical corner exchanges normalize each corner to a canonical open or closed word while preserving the score, acyclicity, and the corners at which the gaps occur. The acyclicity check uses the two attachment vertices of each exchanged box: reflection preserves the local connection relation on these two vertices, so gluing back the unchanged exterior cannot create a cycle. Rotation of the resulting square gives a four-corner code read clockwise from northwest, using 0 for an open corner and 1 for a closed one. The code is 0011 for adjacent open corners or 0101 for opposite open corners. Each digit here describes an entire corner.

For the opposite case 0101, the certificate in `Opposite0101.Exclusion` reads a north strip and a west strip of seven sites in depth from the same board. Price and frame constraints put successive column pairs in checked suffix sets, including at the repeated middle seam. The two sets of possibilities at the junction are incompatible with agreement on the shared corner.

For the adjacent case 0011, the local coverage certificate supplies, around the forced exit column, a physical strip of nine columns whose successive triples of 3×3 tiles obey the checked band tables. Each pair of neighboring triples uses the same middle masks in the two overlapping table entries. The frontier state records the current mark row, the connectivity partition (including a root label), and whether a component contains a mark or can reach the root or a side of the strip. The physical transfer proves that the actual rows survive the update: a cycle cannot occur, and a marked component cannot disappear from the frontier with neither root nor side access. With the prescribed north input, the middle update reaches a fixed set of 13 states after one middle step. The two southern updates leave the empty set. This finite check and the physical invariant are combined in `TwoGapFrontierExclusion`; the fixed set permits any positive number of middle steps, including the count on side 35. □

Proposition 10 (One-gap exclusion)**.** *For every $u \geq 0$, no forest of score* (4) *has one gap.*

Proof. The forced perimeter is now one path. An interior component can attach to it at most once, since two attachment edges together with the interior path and the intervening perimeter subpath would form a cycle. All perimeter stars are uncharged. The seven-layer certificate in `L7Exclusion` tests the two arms of the

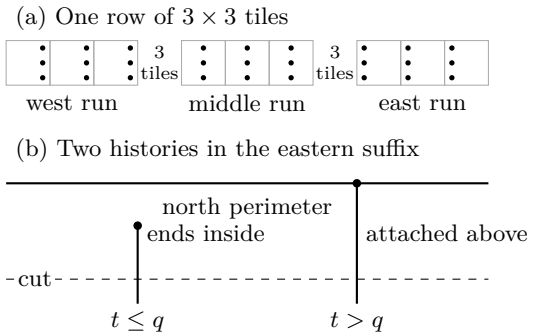


FIGURE 6. The long-fan check. (a) The three runs repeat tiles with marks in the right, middle and left columns, respectively. The two breaks are fixed three-tile transition words, drawn in compressed form. (b) Schematic forced paths in the eastern suffix at the cut, with tile column t. A path ending inside can be dropped above the cut only after checking that its upper component has no other crossing.

gap corner with their common junction; it excludes a charged patch even if the patch lies outside the tested arms. The full defect is therefore a star at an isolated, unmarked site. In the view furnished by `ShortSeed`, its zero-based coordinates are $(2, 2 + 3q)$, where q is odd and $1 \le q \le k - 2$. In these coordinates the northern tile of index j, $0 \le j < k$, has upper-left site $(1, 1 + 3j)$. Its center is marked for $j < q$; its mask is 64 for $j = q$ and 72 for $j > q$, with row-major bit indices starting at zero. The seed also retains the orientation with one owner for each forced edge. Here an owner is the endpoint whose arrow traverses that edge.

For $u = 0, 1, 2$, the physical scans assembled in `ShortEarly` exclude all seed indices. For $u \ge 3$, the reduction is from a hypothetical equality board to an index (u, q) in a finite list. The seed locates q on the north side, and the *fan* is the pattern of marks forced below it in the normalized view. The geometric exclusions determine which indices can survive; the final finite check tests whether there are any such indices.

- For a long fan ($q \ge 57$), a row of tiles has three runs separated by fixed transition words, as in Figure 6(a). Two consecutive rows of tiles determine the next row by a checked transition. As the tile-row index increases by one, the run lengths change by $(3, -6, 3)$. Iteration up to $Y = \lfloor (q - 15)/6 \rfloor$ exposes the eastern suffix of parallel forced paths. At the cut, these paths have two histories (Figure 6(b)): its columns through q have separate paths that end inside the board; its columns east of q lead to their own north attachments. The transfer to the next strip keeps track of this difference. A dangling path may be omitted above the cut only after checking it has no other crossing; a rooted path keeps its perimeter connection.
- Checked windows advance these fronts using the actual mark masks, edge owners and attachments where required. The admissible transfers are scheduled only with the appropriate margins. When a suffix reaches the checked south caps, its retained north connection is incompatible with the

forced continuations at the cap. The width and height tests, together with the alternate collision and boundary cuts, restrict any remaining long failure to a finite list of index pairs (u, q).

- For $q < 57$, short north scans, buffered rooted suffixes, and boundary and birth cuts supply the corresponding coverage. The last north-strip cuts are at $(u, q) = (3, 49), (3, 51)$. They retain the actual east and west boundary prices and read the dangling seals and the north connections at the row-nine cut in tile coordinates. At $q = 49$ the transfer starts at column 26; at $q = 51$ the plain suffix starts at 27 and a restart supplies the input for the longer transfer. Both lead to checked southern obstructions.

In `ScoredFanFailure`, the theorem `failure_from_score` supplies the seed and a surviving obstruction on the same board. The theorem `fanFailure_index` then puts its index in the list `pendingFanIndices`. The check `pendingFanIndices_empty` verifies that this finite list is empty, which rules out the seed for every $u \geq 3$, including side 161. □

Propositions 9 and 10, applied to Lemma 8, exclude the single candidate score $U(k)$ on $k \equiv 11 \pmod{14}$. At every other $k \geq 6$ Lemma 7 already gives the upper in (1); Proposition 6 supplies the matching lower in all phases, by Lemma 2. For the small sides the proofs through side 5, the separate finite checks on sides 8 and 11, and direct stencil replays at 14 and 17 give the values stated above. The residue-two assembly is the branch of `GridGraphs.Results.SquareFormula` using these bounds and the empty fan index list.

6. Boundary contribution

The bulk term $n^2/3$ is common to all three residue classes, but the linear term is not. In fact Theorem 1 gives

$$\lim_{\substack{n\to\infty \\ n\equiv s \pmod 3}} \frac{P(n) - n^2/3}{n} = \begin{cases} 1 & s = 0, \\ 11/10 & s = 1, \\ 20/21 & s = 2. \end{cases}$$

The one-vertex exception in the third class is invisible to this limit. Its proof required analyzing which physical forests could come within $1/14$ of the rational upper bound, beyond what the upper bound itself can decide.

Acknowledgments and AI disclosure

I thank Dong Kim for introducing me to the problem. Large language models substantially assisted the research, including development of the repository's code and formal proof. This manuscript was drafted with language models from the project's sources and proof notes. The verification appendix describes the checks on which the theorem relies.

This paper is licensed under the Creative Commons Attribution 4.0 International License (CC BY 4.0).

Appendix A. Verification and reproducibility

Source and toolchain. The public verification snapshot is Git commit `6fc87a38777650b7d092b2f0f2fa8a786d3c5db8`. It contains the Lean sources, the finite certificate inputs, and the standalone Python tree generator. The file `lean/lean-t`

oolchain selects Lean 4.33.1 (with its bundled Lake); lean/lake-manifest.json pins mathlib 4.33.1 at commit 0df444a360eaa60ab8c11dca51a86af692955474 and locks the other Lean dependencies. The commands below use a POSIX shell on x86-64 Linux, Git, Elan, and xz on the executable search path. The reference versions are Lake 5.0.0-src+819816b, XZ Utils 5.4.5, and Python 3.12.3 (for the generator and the independent output check).
Proof replay and axiom audit. For a fresh checkout use the commands below. The optional cache step obtains precompiled mathlib artifacts. Keep the checked-in manifest.

```
git clone https://github.com/map9959/grid-graphs.git
cd grid-graphs
git checkout --detach 6fc87a38777650b7d092b2f0f2fa8a786d3c5db8
cd lean
lean --version
lake --version
lake exe cache get
LEAN_NUM_THREADS=4 lake build \
  +GridGraphs.Results.SquareFormula:olean
lake env lean --stdin <<'LEAN'
import GridGraphs.Results.SquareFormula
#check GridGraphs.maxPreserved_square_exact
#print axioms GridGraphs.maxPreserved_square_exact
#eval GridGraphs.squareScore 161
LEAN
```

The theorem equates GridGraphs.maxPreserved at (n, n) with GridGraphs.squareScore at n, for every natural $n > 0$. Its transitive #print axioms output includes the three standard axioms propext, Classical.choice, and Quot.sound, as well as the axioms generated by native_decide for finite computations (listed with _native.native_decide in their names). It contains no sorryAx. The command above prints the complete list for the pinned snapshot; the final line evaluates the formula to 8791.
Map from the statement to the proof. In GridGraphs, numPreserved counts vertices of unchanged degree in a spanning tree of gridGraph on Fin m × Fin n; maxPreserved maximizes this count over spanning trees. maxPreserved_eq_maxForcedAcyclic identifies this maximum with the largest vertex set whose incident grid edges form a forest; SquareMaskScore expresses existence at a specified cardinality. squareScore evaluates the piecewise formula, and ExactSquareFormula asserts its equality with the maximum at every positive side.

The file GridGraphs/Results/SquareFormulaAssembly.lean combines the small cases, ResidueZero.maxPreserved_three_mul, maxPreserved_residue_one, and the residue-two bounds. The latter use squareScore_lower_residue_two and ResidueTwoUpper.Stencil.upper_of_remaining, with separate small-side bounds. In SquareFormulaFanResidual.lean, exactSquareFormula_of_pending_fans reduces the outstanding exclusions to failures at listed indices of the same physical forest. In ResidueTwoUpper/OneGap/ScoredFanFailure.lean (relative to lean/GridGraphs/), pendingFanIndices_empty checks that the list left after the geometric reductions and finite cuts is empty. exactSquareFormula applies

that fact in the final module `GridGraphs.Results.SquareFormula`, which exports `maxPreserved_square_exact`.

What the certificate replay trusts. Lean checks the graph-to-certificate reductions and the implications from successful checks. For instance, the side-eight proof checks the stored LRAT refutation using Lean's LRAT checker and its soundness theorem. For the southern-cap coverage, Lake tracks the compressed witness `SouthCapCoverageAll.xz` and loads the native checker. The file `lean/SouthCapCertificate.lean` decompresses the checked-in input, replays reverse unit propagation, and invokes `rup_sound` to justify the resulting coverage statement. This component can also be replayed on its own from `lean/`:

```
LEAN_NUM_THREADS=4 lake build +SouthCapCertificate:olean
```

The stored certificates are inputs to the replay; no external SAT or LP solver is needed. The closed propositions discharged by `native_decide` rely on the correctness of Lean's native evaluation and execution environment in addition to kernel checking of the surrounding proof. The Python generator plays no role in this proof replay.

Independently checking a generated tree. From the repository root, run the commands below. The checker reads only the emitted JSON and takes the side and target score independently: here $161 = 3 \cdot 53 + 2$, with target 8791 from the formula. For another side, change both commands and supply its target from the piecewise formula or small-side table.

```
python3 -B generate_square_tree.py 161 > tree.json
python3 -E - 161 8791 tree.json <<'PY'
import json
import sys

n, target = map(int, sys.argv[1:3])
with open(sys.argv[3], encoding="utf-8") as file:
    tree = json.load(file)
assert n > 0 and tree["side"] == n
adj = {(r, c): set() for r in range(n) for c in range(n)}

def vertex(pair):
    r, c = pair
    assert type(r) is int and type(c) is int and (r, c) in adj
    return r, c

for a, b in tree["edges"]:
    u, v = vertex(a), vertex(b)
    assert abs(u[0] - v[0]) + abs(u[1] - v[1]) == 1
    assert v not in adj[u]
    adj[u].add(v)
    adj[v].add(u)
assert len(tree["edges"]) == n*n - 1

seen, stack = {(0, 0)}, [(0, 0)]
while stack:
    for v in adj[stack.pop()] - seen:
```

```
        seen.add(v)
        stack.append(v)
assert len(seen) == n*n

full = {(r, c) for (r, c), nbrs in adj.items()
        if len(nbrs) == sum((r > 0, r+1 < n, c > 0, c+1 < n))}
reported = list(map(vertex, tree["full_degree_vertices"]))
assert full == set(reported)
assert len(reported) == len(full) == tree["score"] == target
print(f"verified side {n}, score {target}")
PY
```

Connectivity and the distinct-edge count certify that the emitted subgraph is a spanning tree. Its exact full-degree set is recomputed from adjacency, and its optimality follows by comparison with the proved upper bound.

References

[1] R. Bhatia, S. Khuller, R. Pless, and Y. J. Sussmann, *The full-degree spanning tree problem*, Networks **36** (2000), no. 4, 203–209. Open technical report.

[2] H. Broersma, O. Koppius, H. Tuinstra, A. Huck, T. Kloks, D. Kratsch, and H. Müller, *Degree-preserving trees*, Networks **35** (2000), no. 1, 26–39. Publication record.

[3] S. Acquaviva, *The Full Degree Spanning Tree Problem*, M.S. thesis, Montclair State University, 2023. `https://digitalcommons.montclair.edu/etd/1292`.

[4] S. Acquaviva and D. Bal, *Full degree spanning trees in random regular graphs*, Discrete Appl. Math. **353** (2024), 85–93. DOI.

[5] A. Lucena and A. S. da Cunha, *Stable set reformulations for the degree preserving spanning tree problem*, European J. Oper. Res. **319** (2024), no. 1, 50–61. DOI; Supplement.

[6] A. S. da Cunha, L. Simonetti, A. Lucena, and B. Gendron, *Formulations and exact solution approaches for the degree preserving spanning tree problem*, Networks **65** (2015), no. 4, 329–343. DOI.