A finite certificate for graceful labelings of subcubic lobsters

AI-generated research manuscript
Project contact: Akram Younso | AkramYounso@protonmail.com

Abstract

A lobster is a tree with a path at distance at most two from every vertex. We give a computer-assisted proof that every lobster of maximum degree at most three admits a graceful labeling. We encode the branches along a path by a word on four letters. Short labeled blocks can be joined whenever the label at their common boundary satisfies a rank condition. An explicit table of 6,104 blocks meets these local conditions and parses every nonempty word. The latter assertion is a finite computation on a directed graph of 278,215 reachable states. The supplement contains the table, a labeling algorithm, and an independent verification by finite automata.

hexagon:2611.01156v1

1 From a tree to a word

A labeling of an n-vertex tree is *graceful* if its vertex labels are a permutation of $0, \dots, n-1$ and the absolute differences on the edges are a permutation of $1, \dots, n-1$. Bermond conjectured that every lobster is graceful. Our result concerns the following unrestricted choice of branches under a degree bound.

Theorem 1. *Every finite lobster of maximum degree at most three is graceful.*

The trees of orders one and two have the required labelings. For a larger tree, take a witnessing path and extend it at its ends until both endpoints are leaves. If it starts as a single vertex, first extend it by one edge. Each extension preserves the distance bound. Call the two leaves the *caps*. Each internal path vertex has room for at most one neighbor off the path. Such a neighbor can have zero, one or two children, all leaves. Thus the internal vertices, in path order, spell a nonempty word over $\mathcal{A} = \{\mathtt{0}, \mathtt{L}, \mathtt{P}, \mathtt{C}\}$:

letter	`0`	`L`	`P`	`C`
off-path vertices	none	a leaf	a two-edge arm	a head with two leaves

Conversely, joining the indicated vertices in a path and adding the two caps realizes every nonempty word. Several words can represent the same unrooted tree. Only existence of a representation is needed.

The graph-input algorithm chooses a diameter. To see why this works, suppose a component off a diameter reaches depth $h \geq 3$ from its attachment v. Both segments of the diameter from v have length at least h, since replacing a shorter segment would lengthen the diameter. Three components of $T - v$ now reach distance three from v. A path can visit at most two of them. In the missed component a vertex at depth three has distance at least three from the path, contradicting the lobster hypothesis.

2 Joining blocks across a single edge

For a nonempty substring w, form a *block* from its internal spine vertices and their branches. Include a cap only when specified. Write e and t for the first and last spine vertices, which coincide when $|w| = 1$. An unfinished labeled prefix of order I offers port $s \in Q = \{0, 1, 2, 3\}$ if its exit label is s or $I-1-s$. Complementing all labels about $I-1$ preserves every difference, so either orientation of such a port is available.

A row (a, w, b, f) consumes w. Its source a belongs to $Q \cup \{\mathsf{S}\}$ and its destination b to $Q \cup \{\mathsf{F}\}$. Include the left cap precisely for $a = \mathsf{S}$ and the right cap precisely for $b = \mathsf{F}$. If $a = \mathsf{S}$, the vector f gives an ordinary graceful labeling of the block, offering port b at t whenever $b \in Q$.

Otherwise color the tree in two classes H and D with $e \in H$. Write B for its order. Within each class X the values of f must be a permutation of $0, \ldots, |X| - 1$. Require

$$\{f(u) + f(v) : uv \in E\} = \{0, \ldots, B-2\}, \qquad f(e) = a.$$

For $b \in Q$, require also $f(t) = |X_t| - 1 - b$, where X_t contains t. The sum condition is empty for a single vertex. Since the block is a tree, equality of the two sets implies each required sum occurs exactly once.

Lemma 2. *A row from $a \in Q$ extends every prefix offering a to a graceful labeling of the joined tree. If its destination belongs to Q, the new exit offers that port.*

Proof. Orient the old labeling g so that its exit has label a. Put $k = |D|$. On the joined tree use

vertices	new label	interval occupied
$v \in D$	$k - 1 - f(v)$	$[0, k-1]$
v in old tree	$k + g(v)$	$[k, k+I-1]$
$v \in H$	$k + I + f(v)$	$[k+I, k+I+\lvert H\rvert - 1]$.

These intervals partition the label range. The bridge joins $k + a$ to $k + I + a$, supplying difference I. The old differences are $1, \ldots, I-1$. Each new internal edge uv has difference $I + 1 + f(u) + f(v)$, giving $I+1, \ldots, I+B-1$. Substitution at t gives label b in D or $I + B - 1 - b$ in H. □

A parse may end at F, with the right cap already in place, or at port zero. In the latter case orient the exit to zero and attach a right cap labeled by the current order. Its edge supplies the new maximum difference.

Example. The rows $(\mathsf{S}, \mathtt{0}, 0, [0, 1])$ and $(0, \mathsf{C}, \mathsf{F}, [0, 1, 1, 2, 0])$ parse $\mathtt{0C}$. In the second block the low class consists of the head and the right cap, so $k = 2$. The old spine vertex and its cap receive $2, 3$. The new spine vertex, head, two children and right cap receive $4, 0, 5, 6, 1$. The bridge has difference two and the four new internal differences are $4, 5, 6, 3$.

3 Why the table covers words of every length

For each block word w and source set M, put

$$R_w(M) = \{b : (a, w, b, f) \text{ is a row for some } a \in M \text{ and some } f\}.$$

For a prefix v, let $P(v)$ be its attainable unfinished ports and initialize $P(\epsilon) = \{\mathsf{S}\}$. The maximum row length is six. For nonempty v compute

$$U(v) = \bigcup_{v=uw,\ 1\le|w|\le6} R_w(P(u)), \qquad P(v) = U(v) \cap Q.$$

Every parse has a last row. Removing it leaves a parse counted by $P(u)$, or the empty start. Conversely each term of the union appends a legal last row. Thus v admits a complete labeling by our rules whenever $U(v)$ contains F or zero. A finished parse can never supply a source for another row.

For a word of length j, store its last $\min(5, j)$ letters and P for its last $\min(6, j+1)$ prefixes, newest first, including the word itself. After appending a letter, these records supply every suffix w and every corresponding $P(u)$ needed above. Test acceptance, push the new P, and truncate the records. Equal records therefore give identical results on each next letter. Starting at the empty record with history $(\{\mathsf{S}\})$, explore all four successors of every newly reached record.

Certificate rows	Unfinished	Finished
Start with a left cap	486	6
Extend across a bridge	5,574	38

Both implementations of this recurrence exhausted their queues after visiting 278,215 records and testing $4 \cdot 278{,}215 = 1{,}112{,}860$ transitions. Every transition passed. The largest shortest distance from the initial record was 46. By induction on length, the record of every word is reachable, and the transition that reads its final letter tests its acceptance. A word of arbitrary length may revisit records many times.

Proof of Theorem 1. Encode the tree as in Section 1. The calculation above provides a parse. Its first row labels the initial block. Apply Lemma 2 successively to the remaining rows, then use the terminal convention of Section 2. The construction adds exactly the edges of the encoded tree and supplies every required difference. □

Recovering a labeling. For a specified word one can store the preceding position and row the first time each port or finished marker is reached at each position. Backtracking from an accepting destination recovers the rows in reverse order. One witness per destination suffices because Lemma 2 uses only the port rank. Parsing takes linear time in the word length for the fixed table. The supplied implementation may relabel the entire current prefix at each join.

4 How long must a block be?

Shortening the table tests which conclusions depend on the choice of rows. For $1 \le k \le 6$, keep only the supplied rows with $|w| \le k$. Breadth-first search on the truncated recurrence gives the following results. A predecessor pointer at each first visit recovers a shortest rejected word when the acceptance test fails.

Maximum length k	Rows retained	A shortest rejected word	Length
1	27	`0C0`	3
2	84	`0CL`	3
3	284	`LLLP0`	5
4	967	`00C0LC0`	7
5	2,484	`LLCL0CL`	7
6	6,104	every word accepted	

There is a stronger obstruction for $k = 2$. Enumerate *every* locally valid transition of length at most two with these four ports. For start rows enumerate all vertex permutations up to complementation. For extension rows independently permute the two index classes. Test each of the four cap choices and retain each attainable source, word and destination triple once. This yields 170 distinct transitions, compared with 84 rows in the truncated certificate. Even the complete short table rejects `0CL`. After `0` the available ports are $\{0, 1\}$. After `0C` they are $\{1, 2, 3\}$. A final singleton `L` needs source zero to accept, and a final `CL` needs source two. Neither cut works. Thus any universal certificate with these semantics needs a block of length at least three. Whether lengths three, four or five suffice with different rows is left open. Failure of a restricted parse says nothing against gracefulness of its tree.

5 Constructing labels in linear time

The parse also gives a linear-time labeling algorithm in the unit-cost model on $O(\log n)$-bit integers. At a join, all old labels receive the same affine map

$$A(x) = x + k \quad \text{or} \quad A(x) = I - 1 - x + k.$$

Make a forward pass recording the labels of the newly inserted vertices and the map applied to previous vertices. To choose the map one needs only the old order and the exit label, already known from the preceding block. Treat a final zero-port cap as one more insertion, with either the identity or complementation on the preceding tree.

In reverse order maintain the composite F of all later maps, initially the identity. Output $F(x)$ for each label x inserted at the current step, then replace F by $F \circ A$. Induction gives precisely the labels of the forward construction. Each map has slope ± 1 and its offset is a sum of at most $O(n)$ integers of magnitude $O(n)$, hence needs $O(\log n)$ bits. Each vertex is written once. With backpointers for parsing, total time and storage are $O(n)$ for the fixed certificate. The supplement compares the output to the original constructor and checks all edge differences of a 36,002-vertex example.

6 Independent checks and scope

The supplement fixes the JSON table byte for byte. Each row uses -1 for S and -2 for F. In its label vector the spine vertices come first, then the branches in spine order, each head followed by its children, then the included caps in left-to-right order.

The bundled Python verifier reconstructs the edges and checks the row equations. It also constructs labelings for all 5,460 words of lengths one through six, five longer examples and an adjacency-list input. A separate graph-input audit enumerated all 13,188 unlabelled trees through order 15 and classified the 2,233 subcubic trees by testing paths for distance-two coverage. For all 1,758 lobsters, every diameter witnessed coverage and the returned labels gave the required differences on the original edges. The constructor rejected all 475 subcubic nonlobsters. The universal step is the closure calculation. The bundled JavaScript verifier checks coverage alone. It encodes letters as base-five digits 1 through 4 and histories as base-32 digits, with separate bits for start and finish. Its packed states and intermediate arithmetic stay below 2^{53} for this table (the history shift is below 2^{35} and the stored state below $2^{30}5^5$).

A separate Python audit reads every row without filtering, computes vertex classes from parent parity and checks the local permutations and edge equations. It uses a different language calculation. Build a prefix tree of rule words for each source. An active node denotes a partially read rule from that source. Reading a letter advances each active node by one child where possible. At a completed rule, also activate the destination root for the next letter if the destination is a port. Retain the child itself whenever a longer rule can continue there. Test acceptance using the destinations of rules completed on this letter. Starting with only the start root active, induction shows that these subsets represent exactly the possible partial parses. All four transitions passed for each of the 175,472 reachable subsets. This state count differs because this calculation stores active rule prefixes directly.

Where the degree bound enters. With maximum degree four, an internal path vertex can have up to two off-path heads. A head can have $0, 1, 2$ or 3 children. The heads form an unordered multiset, since interchanging them gives the same rooted decoration. There is one empty multiset, four singletons and $\binom{5}{2} = 10$ pairs with repetition, hence 15 letters. Even a one-letter word bearing a new decoration has no row in the present table. Extending the proof would require sound rows for the larger alphabet and coverage of all mixtures of letters at compatible ports. An alphabet count supplies neither assertion.

Prior results. Morgan's theorem requires a perfect matching, and Krop's requires a matching missing one vertex [1, 2]. With three disjoint C branches, each head can match at most one of its two children, so at least three leaves remain unmatched. Luiz, Campos and Richter give an α-labeling under restrictions that exclude Y-legs and allow at most one forbidden ending [3]. A long spine can carry a C branch well inside the path, producing just such a Y-leg. An α-labeling additionally requires a single threshold crossed by every edge. The interval construction imposes no such global threshold. Ghosh's joining theorems use hypotheses on neighboring lobes, including pairwise similarity, linkage or balance [4]. Our rules permit independent branch choices at successive positions.

Bridges at marked vertices. The proof of Lemma 2 uses no path condition on the attachment vertices. Let v be any vertex of a gracefully labeled tree of order I, and e any vertex of a disjoint tree carrying the two class permutations and edge sums of Section 2. Choose the class containing e as H. The same three intervals give a graceful labeling after adding ve whenever $g(v) = f(e)$ after an optional global complement of g. Indeed the signed bridge difference is $I + f(e) - g(v)$, and the remaining differences already exhaust every length except I. This statement permits an interior branching vertex on either side. Its matching condition applies to the prescribed labelings.

There is a quantitative cost to preserving a second old attachment site. For an old vertex of oriented rank r, let $d = \min(r, I - 1 - r)$ be its distance from the nearer end of the label interval. After a join with $k = |D|$ and $h = |H|$, its new distance is

$$d' = \min(k + r,\ h + I - 1 - r) \ \geq\ d + \min(k, h).$$

Complementation preserves d. Thus each attachment of a block with at least two vertices moves *every* old vertex at least one rank farther from the nearer extreme. After four such joins a vertex present throughout cannot offer any of the four certified ports. For example, label a claw by zero at the center and $1, 2, 3$ at its leaves. Join the center to one end of a disjoint edge whose two class indices are both zero. The center now has label 1 on a scale from 0 to 5. Neither orientation allows a second identical edge block to be joined there by this rule. In contrast, adjoining a singleton at a zero-labeled hub preserves its zero and can be repeated there indefinitely. A branching recurrence must explain which other marked vertices will still be usable, or provide a relabeling that recovers them.

Priority and review. A search of the primary literature on graceful subcubic lobsters, branch restrictions, matchings and claimed general solutions found no established theorem covering this entire class. Gnang's general preprint [5], if correct, would imply Theorem 1 immediately. A related preprint was withdrawn in September 2026 for an error in its composition lemma, whose argument used version two of Gnang's paper [6]. That withdrawal does not decide the status of Gnang's revised argument. The present proof uses neither claim. The search does not establish priority. This manuscript and its certificate have been checked computationally as described here and have not undergone independent peer review.

Reproduction. From the supplement directory, use Python 3.10 or later with assertions enabled for the bundled verifier, and Node.js for the second command. The proof checks below require no third-party packages. The optional graph-input audit uses NetworkX (see the supplement README).

```
python verify_subcubic_lobster.py
node verify_lobster_language.js subcubic_lobster_rules.json
python audit.py
python truncation.py
python all_short.py
python linear.py
python marked_join.py
```

The audit also checks the SHA-256 digest of the 250,177-byte JSON table (concatenate the two lines):

```
f1e26f8a0944ee8de9fd40fdce9310c2
360d3ff7d31236737d1a10dc4403e1d7
```

Research provenance and use of AI. Akram Younso initiated an AI investigation of the Graceful Tree Conjecture. During that investigation, the agent identified the subcubic lobster theorem and produced the proof and its finite certificate. Younso describes his role in the discovery as prompting the agent to continue. He supplied no mathematical ideas or proof steps. At his request, OpenAI Codex prepared this manuscript, examined the argument and the literature, and ran the computational checks reported here.

References

[1] D. Morgan, All lobsters with perfect matchings are graceful, *Electronic Notes in Discrete Mathematics* (2002). Publisher record.

[2] E. Krop, Lobsters with an almost perfect matching are graceful (2014). `https://arxiv.org/abs/1402.3994`

[3] A. G. Luiz, C. N. Campos and R. B. Richter, On α-labellings of lobsters and trees with a perfect matching, *Discrete Applied Mathematics*. `https://doi.org/10.1016/j.dam.2019.05.004`

[4] S. Ghosh, On certain classes of graceful lobsters (2013), Section 4. `https://arxiv.org/abs/1306.2932`

[5] E. K. Gnang, A proof of the Kotzig–Ringel–Rosa Conjecture, version 3 (2025). `https://arxiv.org/abs/2202.03178v3`

[6] P. Chalise, A. Clark and E. K. Gnang, Every tree on n edges decomposes $K_{nx,nx}$ and K_{2nx+1}, withdrawal notice, version 3 (2026). `https://arxiv.org/abs/2409.01981v3`