NEW SNAKE-IN-THE-BOX RECORDS VIA SNAKEPIT SURGERY AND LEARNED CONSTRUCTION

Anonymous authors
Paper under double-blind review

ABSTRACT

The snake-in-the-box problem asks for a longest induced path in the hypercube graph Q_n. We find a length-191 snake in dimension $n = 9$—the lowest dimension where the maximum is unknown—improving the previous record of 190 that had stood for 14 years. We also establish new lower bounds in dimensions 10–13. To find these records, we introduce snakepits, collections of disjoint snakes, to expand the search space and open new routes between snakes. This motivates our new Snakepit-in-the-Box benchmark, which seeks maximal edge counts when allowing multiple components. Finally, we introduce Beam Anchor, a search-supervised learned constructor algorithm that finds 100 inequivalent length-190 snakes in dimension 9.

1 INTRODUCTION

The hypercube graph Q_n has binary strings of length n as vertices and edges between strings that differ in exactly one coordinate. A *snake* is an induced (chordless) path: consecutive vertices are adjacent and distinct nonconsecutive vertices are not. The snake-in-the-box problem, introduced by Kautz (1958), asks for the maximum snake length, $a(n)$. The problem came from coding theory as error-detecting Gray codes, and it is an ideal test of combinatorial search: snakes are easy to construct and validate, but long snakes are exceedingly rare and the optimal length is known only through dimension 8 (Östergård & Pettersson, 2015).

There are two complementary ways to search for long snakes.

Construction grows a snake one vertex at a time; each choice restricts later continuations from visiting adjacent vertices. This is a long-horizon decision problem in which the cost of poor decisions may not be realized until hundreds of moves later, so immediate reward is uninformative and credit assignment is difficult.

Modification, on the other hand, makes local changes to an existing snake. However, interior routes are highly constrained for long snakes, making them difficult to change, since most vertices are invalidated by being adjacent to a different part of the snake. We propose best-in-class techniques for both approaches.

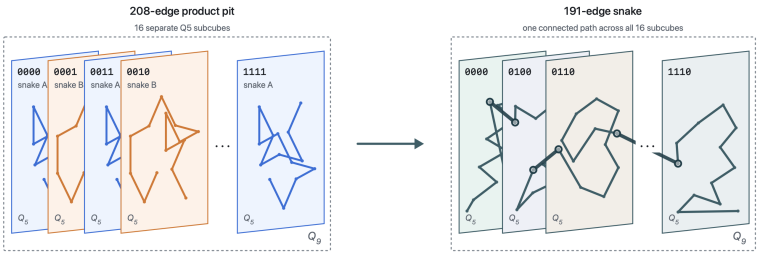


Figure 1: From a snakepit to a snake. Two vertex-disjoint length-13 snakes in Q_5 can be combined to form a 16-snake snakepit with 208 edges in Q_9. Cube-surgery annealing can rearrange such a starting state into a length-191 snake.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Table 1: Lower bounds established by this work relative to the records preceding our announcement. The final column counts inequivalent snakes found at the record length, up to the symmetries of Q_n and reversal.

dimension n	our bound $a(n) \geq$	previous record	Δ	# classes at record
9	$\mathbf{191}$	190 (Wynn, 2012)	$+\mathbf{1}$	131^*
10	$\mathbf{379}$	376 (Ace, 2025)	$+\mathbf{3}$	≥ 12
11	$\mathbf{746}^\dagger$	737 (Ace, 2026a)	$+\mathbf{9}$	≥ 72
12	$\mathbf{1476}^\dagger$	1465 (Ace, 2026a)	$+\mathbf{11}$	$\geq 34{,}688$
13	$\mathbf{2924}^\dagger$	2900 (Echols, 2026b)	$+\mathbf{24}$	≥ 30

*Discovered bank, not a census; see Section 6.1.
†Subsequent reported bounds: 750 (Echols, 2026a), 1480 and 2934 (Itty, 2026).

For modification, we gain flexibility by expanding the state space to *snakepits*, collections of disjoint snakes, which allows for easier routes between snakes and therefore makes rewards less sparse. Our algorithm uses a basic move called *cube surgery*: clear a subcube and rebuild its contents subject to the surrounding snakes. Allowing snakepits makes cube surgery a more flexible operation: a refill can change the overall connectivity structure of the snakepit. We discovered the new record snakes in dimension 9 using simulated annealing with cube surgery with a penalty for larger number of components.

For construction, we introduce Beam Anchor, which combines beam search with a learned policy to construct a single path by comparing continuations from a common prefix. From a partial snake, it gives each possible next move its own share of search and trains the policy to favor moves whose continuations reach longer snakes.

In sum, our main contributions are:

- **New records, led by** 191 **in** Q_9**.** We find a length-191 snake in dimension 9, breaking the record of 190 that had stood for 14 years, and improve snake bounds in dimensions 10–13 alongside new coil bounds. Our bank of 131 inequivalent length-191 snakes supports an empirical study of family structure and search coverage.
- **Snakepit search with cube surgery.** We relax the search state from a single snake to a snakepit and rebuild small subcubes in one annealing proposal. At equal time this beats add/remove annealing by about 40 edges in Q_9 and rediscovers the record from an empty cube.
- **Learned construction with Beam Anchor.** We apply reinforcement learning to snake construction and develop Beam Anchor, which finds 100 inequivalent length-190 snakes in Q_9, greatly expanding the previous known snake examples at the longstanding record length.
- **Snakepit-in-the-Box.** We introduce a new benchmark that naturally follows from our use of snakepits as intermediate search states: maximizing the edge count of a snakepit in a given dimension. We provide lower bounds, exact values through Q_7, and a density limit.

Sections 4 and 5 develop modification and construction; we then discuss the records, landscape and coils, and close with the snakepit benchmark.

2 Related Work

Exhaustive search established the snake optima through Q_8 (Kochut, 1996; Östergård & Pettersson, 2015). Higher-dimensional records have come from population-based search (Potter et al., 1994; Casella & Potter, 2005), permutation-based constructions (Wynn, 2012), Monte-Carlo search (Kinny, 2012; Meyerson et al., 2015), nested rollout policy adaptation (Dang et al., 2023), and beam search and local rerouting (Ace, 2025; 2026a; Echols, 2026d;b). Our higher-dimensional bounds are historical improvements; Table 1 marks subsequent advances.

Cube surgery connects penalty-based annealing (Kirkpatrick et al., 1983) with ruin-and-recreate and large-neighborhood search (Schrimpf et al., 2000; Shaw, 1998). Collections of separated snakes also occur in coil enumeration (Östergård & Pettersson, 2014); here they are intermediate states that permit repeated interior rearrangement. Beam Anchor is policy-only expert iteration: beam search supplies the teacher and policy distillation guides later search (Anthony et al., 2017). It shares the search–learning loop of AlphaZero (Silver et al., 2018), but uses no learned value head and supervises only the anchor where alternatives were compared.

3 Snakes and Snakepits

Vertices of Q_n are binary strings, with coordinates indexed from 0 to $n-1$. An edge flips one coordinate. A d-dimensional subcube leaves d coordinates free and fixes the rest.

Definition 1 (Snakes and coils)**.** A snake in Q_n is an induced path $(v_0, \ldots, v_L)$ of distinct vertices. Consecutive vertices are adjacent and nonconsecutive vertices are not. Its length is L, and it has $L+1$ vertices. We write $a(n)$ for the maximum snake length. A *coil* is an induced cycle; its length equals both its edge count and its number of distinct vertices.

The *transition sequence* records the coordinate flipped at each step. We consider snakes up to translations, coordinate permutations and reversal: to canonicalize a transition sequence, relabel coordinates by order of first appearance, do the same to the reversed sequence (and cyclic permutations for coils), and take the lexicographically smallest result. A class count always refers to this equivalence unless explicitly stated otherwise.

Definition 2 (Snakepit)**.** A snakepit F is an induced linear forest in Q_n. Equivalently, it is a collection of (possibly length 0) snakes with no cube edge between vertices in distinct components. Write $v(F)$, $c(F)$ and $e(F)$ for its numbers of vertices, components and edges, respectively. Since each component is a path, we have that $e(F) = v(F) - c(F)$.

A snake, then, is a snakepit with one component. Importantly, the intersection of a snake(pit) with a subcube is also a snakepit: deleting vertices can split paths but cannot make nonconsecutive vertices adjacent. A sample snake and snakepit in Q_4 are given in Figure 2.

The *Snakepit-in-the-Box* problem asks for the maximum edge count of a snakepit in Q_n. Its finer breakdown by component count is

$$L(n,k) = \max\{e(F) : F \text{ is a snakepit in } Q_n,\ c(F) = k\},$$

for feasible $k \geq 1$. In particular, $L(n,1) = a(n)$. Section 7 develops the table of values and bounds.

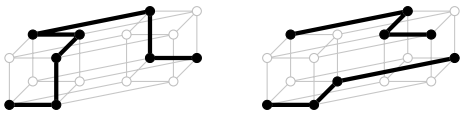


Figure 2: A snake with 7 edges (left) and a snakepit with 6 edges in two components (right), both in Q_4.

Definition 3 (Product snakepit)**.** Let A and B be snakepits in Q_m with no common vertices and let $n > m$. In $Q_n = Q_{n-m} \times Q_m$, place a copy of A over each even-parity vertex of Q_{n-m} and a copy of B over each odd-parity vertex.

Clearly the result is a snakepit satisfying

$$e(F) = 2^{n-m-1}\big(e(A) + e(B)\big), \qquad c(F) = 2^{n-m-1}\big(c(A) + c(B)\big).$$

Example (Product snakepits in Q_9)**.** Two disjoint length-13 snakes in Q_5 give a pit with 16 components, 208 edges and 224 vertices in Q_9, represented schematically on the left of Figure 1. A disjoint length-26 and length-24 pair in Q_6 gives 8 components, 200 edges and 208 vertices in Q_9. These pits contain more edges than a length-191 snake, but spread them across multiple components.

4 FINDING RECORD SNAKES: MODIFICATION

Cube-surgery annealing found our length-191 snake in Q_9. Local rerouting has contributed to previous records (Ace, 2025; Echols, 2026d), but long snakes leave little room for interior edits: a replacement must avoid vertices adjacent to the rest of the snake while reconnecting the pieces it separates. We relax the connectivity requirement and search over snakepits, allowing an edit to leave several components that can later be joined into a longer snake.

Cube surgery. Cube surgery rebuilds a small subcube while holding every exterior vertex fixed. Each proposal has four steps:

1. **Clear a region.** Save the current state, draw d uniformly from $\{2, 3, 4\}$, choose a uniform d-subcube U, and remove every occupied vertex inside it. These 4–16-vertex regions permit coordinated changes at low refill cost.
2. **Prefer connections.** Visit the unoccupied vertices of U in random order. Immediately add a vertex if it has at least one occupied neighbor and the addition is legal. This pass extends or joins existing fragments without starting a new component.
3. **Allow new fragments.** Make a fresh random pass, now allowing any legal addition, including a vertex with no occupied neighbors. Such a vertex starts a new component that later additions can grow. Both passes test against the current state, including vertices just added. Repeat the two passes until a full round adds nothing.
4. **Accept or restore.** Compare the energy of the rebuilt state with the saved original and apply the annealing acceptance rule below once to the whole edit. On rejection, restore the original state. Individual deletions and insertions receive no separate acceptance decisions.

A legal addition starts a component, extends a path at an endpoint, or joins two paths at their endpoints, so every refill is again a snakepit. Appendix B.1 gives pseudocode.

Annealing and consolidation. To balance occupied vertices against fragmentation, the annealer uses a component penalty. For a snakepit F and penalty $\beta > 0$, it minimizes

$$E(F) = \beta|c(F) - 1| - v(F). \tag{1}$$

For a single snake of length L, this is $-(L + 1)$; the empty state has energy β. A proposed change ΔE is accepted with probability $\min\{1, \exp(-\Delta E/T)\}$. The temperature repeatedly decreases from T_0 toward $T_{\min}$, then reheats. Hot phases permit fragmentation and temporary loss of vertices; cooling favors consolidation. Because the proposal is asymmetric, this is an optimization heuristic, not a claim of Boltzmann sampling.

Other search moves. Alongside surgery, the annealer proposes single-vertex additions (with the same legality rule) and removals, which can shorten or split a path.

Endpoint moves replace a longer segment in one proposal: a *wiggle* retracts $j \in \{1, \ldots, 6\}$ vertices from a path endpoint and regrows up to $j + 2$ steps through legal neighbors in random order, and deep regrowth does the same with 8–48 vertices retracted and up to 12 extra steps. Acceptance is evaluated only after regrowth.

From product snakepit to record snake. The first successful chains started from the product snakepits of Section 3. Although these pits contain more edges than a 191-snake, their fragmentation is penalized by Eq. 1: surgery rearranges the fragments while annealing favors their consolidation. The product start is a convenience rather than a requirement, since chains started from the empty snakepit also reach 191, as shown next.

The value of surgery. We compared two fixed recipes in Q_9, each started from the empty snakepit and run for one hour on one CPU core, over ten paired seeds: single-vertex addition and removal alone, and the same moves with surgery proposed 35% of the time. Both recipes allow fragmented states, so the comparison isolates the move rather than the relaxation. After one hour, the add/remove recipe had reached a median best length of 149 (range 147–155) and the surgery recipe 188 (range 186–191); every paired difference favored surgery, by 35 to 41 edges, even though the cheaper

recipe made about $2.4\times$ more proposals in the same time. One surgery run reached a validated 191 after 158 seconds. Appendix B.2 gives the protocol, all twenty runs and the progress at earlier checkpoints. The same annealer also refines the lifted snakes described next.

5 Finding Record Snakes: Construction

Our successful ordinary beam-search pipeline uses a long lower-dimensional snake as a prefix to build long snakes. Reusing such paths is an established way to make construction tractable. Beam Anchor addresses the harder task of constructing a competitive snake from the origin, without a supplied primer. We first explain the primer strategy, then how Beam Anchor discovers long snakes from scratch.

5.1 Beam search with a primer

Beam search grows valid partial snakes one edge at a time. At each depth, it generates legal extensions, ranks the candidates, and retains a bounded number for the next step. Our lifter ranks candidates by length plus the number of reachable available vertices and Gumbel noise; this favors continuations with room to grow while diversifying the search.

Starting from a strong primer reduces the part of the path that search must discover. We embed a long snake from Q_{n-1} in one half of Q_n as a fixed prefix, then grow a continuation into the other half. This generally produces a snake that crosses the new coordinate exactly once, although unrestricted continuation may cross back. Reusing lower-dimensional snakes is a common way to make record searches tractable (Kinny, 2012; Wynn, 2012), but it can miss optima: the known length-98 snake in Q_8 uses no coordinate exactly once, and extending an optimal length-50 Q_7 primer cannot reach 98 (Kinny, 2012).

We use ordinary beam search for lifting because it is substantially faster than Beam Anchor, making it practical to extend many record primers. We try both primer orientations and selected tail truncations. Each lift remains a single path, and its output seeds cube-surgery annealing (Section 4). Thus the pipeline alternates beam-search construction in a larger cube with modification of the resulting snake. The observed raw lifts $379 \to 743$, $746 \to 1470$, and $1476 \to 2916$ show how improvements propagate between dimensions. The full scoring rule, widths, and deduplication details appear in Appendix B.

5.2 Beam Anchor

Beam Anchor is a learned constructor: it builds a snake from the origin one edge at a time, with no primer, guided by a policy network. It follows the expert iteration scheme (Anthony et al., 2017): a search procedure acts as an expert that improves on the current policy, the policy is trained to imitate the expert, and the improved policy in turn guides the next round of search. In Beam Anchor the expert is a set of policy-guided beam searches, one for each legal move at the current partial snake, and the policy learns which move led to the longest snake. There is no value network.

In Q_9, Beam Anchor finds 100 inequivalent snakes of length 190. Before this project, Wynn's construction was the only published length-190 class known to us (Wynn, 2012), so these snakes substantially enlarge the published landscape at the previous record length; all 100 also occur in the bank of 1,042 length-190 classes produced by our annealer (Section 6.1). To our knowledge, it is also the first learned constructor to reach length 190 in Q_9, and it does so with no prior knowledge of the problem: it starts from the origin with a randomly initialized policy, and neither the search nor the training data draw on primers, lower-dimensional snakes or previously known snakes. Everything the policy knows it has learned from its own searches.

Why compare openings. Immediate reward is uninformative for snake construction: every legal move adds exactly one edge, but the result of a move and the vertices it excludes only becomes consequential many steps later. The key question at a partial snake is therefore global: which of the legal next moves admits the longest completion? Beam Anchor answers this by searching at every step of a construction and distilling the answers into a policy that guides later searches.

Anchor search. Let p be a partial snake, the *anchor*, and let $A(p)$ be its legal next moves, $m = |A(p)|$. A fixed budget of B beams is divided evenly among the legal moves, and we run one beam search per move $a \in A(p)$, starting from the child $p \parallel a$, the partial snake p extended by the move a, with width $\lfloor B/m \rfloor$ or $\lceil B/m \rceil$. The searches never compete for beams, so every move is explored with the same effort regardless of how the current policy ranks it. Within a search, candidates are ranked by cumulative policy log-probability perturbed by Gumbel noise, and the search runs until every beam has completed, that is, reached a state with no legal move. Its outcome is the longest snake it completed,

$$\widehat{L}(p, a) = \max\{\ell(s) : s \text{ a completed snake found by the search through } a\},$$

where ℓ is edge length. This provides a lower bound on the best completion through a.

Walks. A *walk* builds a snake from the origin one move at a time, running an anchor search at each step and sampling the next move from its outcomes. At the current anchor p it (i) runs the anchor search, (ii) records the outcomes in a persistent table,

$$\mathcal{M}[p, a] \leftarrow \max\{\mathcal{M}[p, a], \widehat{L}(p, a)\} \qquad \text{for all } a \in A(p), \tag{2}$$

and (iii) samples its next move from $\mu(a \mid p) \propto \exp\big(\widehat{L}(p, a)/\tau_{\text{walk}}\big)$ and moves the anchor to $p \parallel a$. The walk ends when the anchor has no legal move.

A walk may complete a longer snake than the anchor search found at some prefix along its path, and one could credit that length to the move the walk took there. We do not: that move is followed by further anchor searches while the other moves at the prefix are not, so it has effectively received more search budget, and crediting it would bias the targets toward the moves the walk happened to choose rather than the moves that are better.

Distillation. A cycle runs a batch of walks in parallel and then updates the policy on rows sampled from $\mathcal{M}$. For a row p, the target distribution and loss are

$$q_\tau(a \mid p) = \frac{\exp(\mathcal{M}[p, a]/\tau)}{\sum_{u \in A(p)} \exp(\mathcal{M}[p, u]/\tau)}, \qquad \mathcal{L}(\theta) = -\mathbb{E}_p \sum_{a \in A(p)} q_\tau(a \mid p) \log \pi_\theta(a \mid p), \tag{3}$$

where $\pi_\theta(\cdot \mid p)$ is the policy's distribution over the legal moves $A(p)$. Rows are drawn by sampling a depth uniformly and then a prefix at that depth; otherwise the numerous deep prefixes would crowd out the few early ones, where mistakes are most costly. The temperature τ is higher at shallow depths, where a bounded search separates openings least reliably. The updated policy guides the anchor searches of the next cycle (Algorithm 1, Appendix A).

The policy is the only learned component. A value network would have to predict, from a prefix, the length of its best completion, which may lie more than a hundred moves ahead; and to be useful at an anchor it would have to do so to within a few edges, since that is all that separates the best completions of rival moves. The policy target is easier: it only asks how the moves at one prefix rank against each other.

Policy network. The policy is a graph neural network on Q_n itself, with one node per vertex and messages along cube edges. Each vertex v starts from a vector $x_v \in \mathbb{R}^8$ of features of the partial snake p: whether v is on the snake, still available, the start, the head, or a legal next vertex; its normalized position along the snake; and the fractions of the vertices at Hamming distance one, and at distance two, from v that lie on the snake. Six residual GIN layers of width 64 turn these into embeddings $h_v \in \mathbb{R}^{64}$. Let v be the head of p, and let $v \oplus e_a$ be the vertex reached by the move a, where e_a is the unit vector of coordinate a. The logit of a is

$$g_\theta(a \mid p) = \mathrm{MLP}\big([h_v,\, h_{v \oplus e_a}]\big), \qquad \pi_\theta(a \mid p) = \frac{\exp g_\theta(a \mid p)}{\sum_{u \in A(p)} \exp g_\theta(u \mid p)},$$

with one MLP shared by all moves (Appendix A). No feature or weight refers to a coordinate label, so the network is equivariant under every automorphism of Q_n by construction.

Experiments. *Training.* We trained two different policies from two random seeds for 12 hours on an eight-device MI250 node. Beam Anchor first reached length 190 in Q_9 after about 8 and 8.5 hours, respectively, and continued finding new 190-snakes as training continued. Without learning, Beam Anchor with a uniform policy plateaus at 178 (Appendix A.5).

Search with the trained policy. The trained policies remain useful without further training. With their parameters frozen, we ran Beam Anchor walks from the origin for 12 hours on one eight-device node, with four random seeds (two for each policy). In total, across training and these frozen-policy runs, Beam Anchor found 100 inequivalent length-190 snakes.

Comparison with PPO. To assess the importance of the search, we compare Beam Anchor to a PPO (Schulman et al., 2017) baseline with the same graph neural network (Appendix A.5). Across all PPO experiments, the best snake found had length 184.

6 Records and Search Landscape

Record provenance. Table 1 reports the improvements, led by 191 in Q_9, which came from cube-surgery annealing alone. Lifting and annealing independently reached 379 in Q_{10}. In Q_{11}–Q_{13}, lifts reached 743, 1470 and 2916; annealing took them to 746 and 1476, and in Q_{13} a genetic search with mixed-integer linear programming (MILP) window refill, which fixes everything outside a window of the path and solves exactly for the longest induced path through it, supplied the 2921 seed for the final annealing step to 2924. In Appendix E we give a Q_9 and Q_{10} snake at record length; all other data and class banks will be released upon acceptance.

6.1 What the Q_9 Bank Reveals About Search

To describe the 131 inequivalent 191-snakes, the *dimension-crossing vector* (DCV) counts uses of each coordinate, sorted in decreasing order. Its entries sum to the snake's length; its largest entry, $d_{\max}$, measures how many times the most-used dimension appears.

The 191-snakes separate into a *continent* of 127 classes with $d_{\max} \in \{39, \ldots, 44\}$ and an *island* of four with $d_{\max} = 52$; no class has $d_{\max}$ between 45 and 51 (Figure 3). Both families are basins. From a neutral start, such as a product or the empty snakepit, annealing lands in either family depending on the seed; once there, it readily finds the rest of that family but never reaches the other (Appendix C.4). This separates two evaluation questions: how thoroughly a method explores a known family, and whether it reaches a different one. Beam Anchor illustrates the distinction: all 100 of its length-190 classes have $d_{\max} \leq 46$, in the continent's mode, and none lies in the island's (Figure 4, Appendix A). None of the 131 known 191s admits a one-edge extension, so a 192-snake requires a class outside this bank.

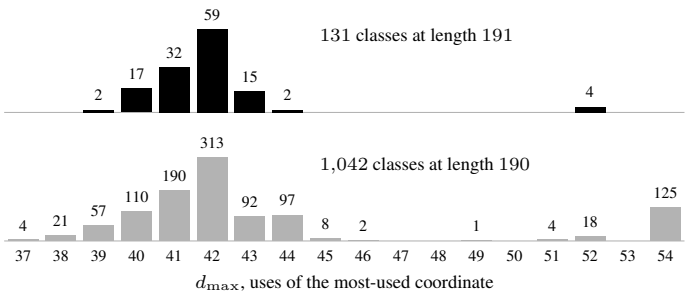


Figure 3: Distribution of $d_{\max}$ over the known Q_9 classes at length 191 (top) and 190 (bottom). At length 191 the continent ($d_{\max} \leq 44$) and the island ($d_{\max} = 52$) are separated by an empty gap; the length-190 stratum shows the same two modes.

Table 2: Coil and symmetric-coil bounds established by this work. Class counts are up to the symmetries of Q_n, reversal and rotation. The separate symmetric-coil rows refer to translation symmetry.

n	our bound	previous	Δ	# classes at record
Coils				
9	**192**	188 (Wynn, 2012)	+4	1
10	**374**	370 (Echols, 2026d)	+4	12
11	**734**[†]	728 (Ace, 2026a)	+6	9
12	**1466**[†]	1442 (Ace, 2026a)	+24	5
13	**2916**[†‡]	2840 (Echols, 2026b)	+76	13
Symmetric coils				
10	**370**	362 (Meyerson et al., 2015)	+8	2
11	**734**	718 (Echols, 2026c)	+16	1

[†]Subsequent reported bounds: coils 740, 1468, 2930 in dimensions 11–13 (Echols, 2026a; Ace, 2026b).
[‡]Found by MILP window refill from a length-2932 snake of Echols, separately from the harvest-and-close campaign.

6.2 Coils

The same searches, adapted, set new coil records (Table 2). A coil is an induced cycle (Section 3); a *symmetric* coil $(v_0, \ldots, v_{2h-1})$ also satisfies $v_{i+h} = v_i \oplus t$ for a fixed t. The records came from three routes.

Harvest and close ($n = 9$–12). A snake whose endpoints are at distance two closes into a coil two longer if a common neighbor of the endpoints is unused and adjacent to no other path vertex. In Q_9, 96 of the 1,042 length-190 classes close, all to the same 192-coil. In Q_{10}–Q_{12} no record-length snake closes, so we ran the annealer warmer with a lowered save threshold and tested the 14–24 million shorter snakes it harvested per dimension, giving 374, 734 and 1466.

MILP window refill ($n = 13$). Starting from a length-2932 snake of Echols, we refilled a window with a mixed-integer program constrained to close the path into a cycle, giving a 2916-coil (Appendix C.2).

Symmetry-restricted search (symmetric coils). Every proven coil optimum in dimensions 2–8 has a representative whose second half is the image of the first under an involutive cube automorphism. Fixing such a map halves the vertices to be chosen, so we adapted surgery (acting on vertex pairs) and Beam Anchor (building one half-path) to search within a fixed symmetry. These searches set the symmetric-coil records of 370 in Q_{10} and 734 in Q_{11}; Appendix C.3 gives both adaptations and their sweep across symmetry classes.

7 Snakepit-in-the-Box

Beyond their role in search, snakepits pose an extremal problem of their own. Let

$$\Lambda(n) = \max_{k \geq 1} L(n, k)$$

be the maximum edge count over all snakepits in Q_n. In dimensions 1–6, snakepits cannot contain more edges than an optimal snake. Dimension 7 is the first where they do: the maximum is 52, versus 50 for snakes, attained by four length-13 paths from the product construction in Section 3. Appendix D gives the computer-assisted proof.

The same construction supplies a general lower bound: the product of two vertex-disjoint length-13 snakes in Q_5 has 2^{n-5} components and $13 \cdot 2^{n-5}$ edges in Q_n, so

$$\Lambda(n) \geq 13 \cdot 2^{n-5} \qquad (n \geq 5).$$

In particular, this construction gives 104 edges in Q_8 and 208 in Q_9. A pit-mode variant of our annealer never exceeded these across 24 two-hour runs in Q_9 (Appendix D.4); their optimality is open.

Table 3: The head of the Snakepit-in-the-Box table $L(n, k)$. Dashes are infeasible; $\Lambda(n)$ maximizes over all k. Unqualified entries are exact (Appendix D); $\geq$ entries are achieved lower bounds.

n	$k=1$	2	3	4	5	6	7	8	$\Lambda(n)$
3	4	2	0	0	—	—	—	—	4
4	7	6	5	4	2	0	0	0	7
5	13	12	12	12	10	10	8	8	13
6	26	26	25	24	24	24	24	24	26
7	50	50	50	52	51	50	49	48	52
8	98	≥ 98	≥ 100	≥ 100	≥ 100	≥ 101	≥ 102	≥ 104	≥ 104
9	≥ 191	≥ 190	≥ 191	≥ 196	≥ 197	≥ 200	≥ 200	≥ 200	≥ 208

Let $D(a)$ be the maximum of $e(A) + e(B)$ over vertex-disjoint snakepits A, B in Q_a.

Proposition 1. *The edge density $\Lambda(n)/2^n$ converges, and its limit ρ satisfies*

$$\rho = \sup_a \frac{D(a)}{2^{a+1}}, \qquad \frac{13}{32} \leq \rho \leq \frac{1}{2}.$$

Proof sketch. Products of disjoint pairs make $D(a)/2^{a+1}$ nondecreasing in a, and Huang's sensitivity theorem (Huang, 2019) bounds every snakepit in Q_n, $n \geq 5$, by 2^{n-1} vertices, so the ratio is at most $1/2$. Conversely, an optimal pit and its translate along a least-used coordinate can be made disjoint while losing at most $3\Lambda(n)/n$ edges, which squeezes $\Lambda(n)/2^n$ between $D(n-1)/2^n$ and $D(n)/2^{n+1}$ up to a factor $1 - 3/(2n)$. The two disjoint length-13 snakes in Q_5 give $13/32$; Appendix D gives the full proof. □

The deficit of a pair from density $1/2$ equals its unused vertices plus its components, divided by 2^{a+1}. Improving on the Q_5 pair, or proving that no pair in any dimension does, is the benchmark's central asymptotic question. Table 3 lists the current values of $L(n, k)$.

Entries should come with an explicit snakepit and the search budget; the full table, our examples and baseline code will be released upon acceptance.

8 Conclusion

There are two key lessons from this work to highlight. First, the choice of state space and move is critical: cube surgery on snakepits reached 191 from an empty cube in minutes on one CPU core, beating both single-vertex annealing on the same states and our best learned constructor with far more compute. Second, within construction, learning does help, and the form of supervision matters: a policy trained only on its own anchored comparisons reaches 190 from the origin, where the same network trained by PPO stops at 184 and the same search with a uniform policy stops at 178.

Two concrete problems remain open: whether a length-192 snake exists in Q_9 (or the weaker question of whether a length-191 snake exists outside the known bank), and whether the product snakepits are optimal, that is, whether $\Lambda(8) = 104$ and $\Lambda(9) = 208$.

Ethics Statement

This work studies an extremal combinatorial problem and involves no living subjects or personal data. In particular, no snakes were harmed, placed in physical boxes, or operated on surgically. We do not anticipate specific societal harms arising directly from the methods or results presented here.

Reproducibility Statement

All data and code supporting the reported results will be publicly released upon acceptance. Appendix A gives the Beam Anchor algorithm, its production settings and the settings of the reported runs; Appendix B gives production surgery, annealing and lifting. Appendix C gives the record provenance, the MILP refill and the coil methods; Appendix D gives the snakepit proofs and benchmark calibration; and Appendix E prints a record snake in Q_9 and in Q_{10}. All record snakes, coils, snakepit examples and class banks will be released with verifiers.

The Use of Large Language Models

We used large language models to help structure, draft, and revise the manuscript; identify and discuss related literature; and discuss methodological choices and the interpretation of results. We also used LLMs to help write and optimize code implementing the search methods and running experiments, with the aim of reducing experiment runtime and accelerating iteration. Computational witnesses are checked by explicit validity tests, as described in the paper. The authors double-checked LLM outputs by hand and take responsibility for the final manuscript, mathematical claims, code, experimental results, and other artifacts produced with LLM assistance.

References

Tom Ace. New lower bounds for snake-in-the-box in 10-, 11-, 12-, and 13-dimensional hypercubes. Zenodo, 2025.

Tom Ace. Snake-in-the-box records (with thanks to W. Echols). https://www.minortriad.com/snake/, 2026a. First published between 26 June and 1 July 2026.

Tom Ace. Snake-in-the-box records. https://www.minortriad.com/snake/, 2026b. Leaderboard; accessed 9 September 2026.

Thomas Anthony, Zheng Tian, and David Barber. Thinking fast and slow with deep learning and tree search. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, 2017.

D. A. Casella and Walter D. Potter. New lower bounds for the snake-in-the-box problem: Using evolutionary techniques to hunt for snakes. In *Proceedings of the 18th International Florida Artificial Intelligence Research Society Conference (FLAIRS)*, pp. 264–269, 2005.

Chen Dang, Cristina Bazgan, Tristan Cazenave, Morgan Chopin, and Pierre-Henri Wuillemin. Warm-starting nested rollout policy adaptation with optimal stopping. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 12381–12389, 2023. Issue 10.

William Echols. Snake-in-the-box updates. https://williamechols.com/posts/sitb_updates.html, 2026a. Dated index of results; accessed 4 September 2026.

William Echols. Improving snakes and coils with further windowed rerouting. https://williamechols.com/posts/improving_snakes_and_coils_with_further_windowed_rerouting.html, July 2026b.

William Echols. Building symmetric coils with implicit quotient beam search. https://williamechols.com/posts/building_symmetric_coils_with_implicit_quotient_beam_search.html, July 2026c.

William Echols. Lengthening hypercube snakes with windowed rerouting. https://williamechols.com/posts/lengthening_hypercube_snakes_with_windowed_rerouting.html, June 2026d.

Hao Huang. Induced subgraphs of hypercubes and a proof of the sensitivity conjecture. *Annals of Mathematics*, 190(3):949–955, 2019. doi: 10.4007/annals.2019.190.3.6.

Nathaniel Itty. Snakes of length 1480 in Q_{12} and 2934 in Q_{13}. Listed on T. Ace's leaderboard, https://www.minortriad.com/snake/, as personal correspondence dated 31 August 2026, 2026.

William H. Kautz. Unit-distance error-checking codes. *IRE Transactions on Electronic Computers*, 7:179–180, 1958.

David Kinny. A new approach to the snake-in-the-box problem. In *Proceedings of the 20th European Conference on Artificial Intelligence (ECAI 2012)*, pp. 462–467, 2012.

Scott Kirkpatrick, C. Daniel Gelatt, and Mario P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

Krzysztof J. Kochut. Snake-in-the-box codes for dimension 7. *Journal of Combinatorial Mathematics and Combinatorial Computing*, 20:175–185, 1996.

Seth J. Meyerson, Thomas E. Drapela, William E. Whiteside, and Walter D. Potter. Finding longest paths in hypercubes: 11 new lower bounds for snakes, coils, and symmetric coils. In *Current Approaches in Applied Artificial Intelligence: Proceedings of the 28th International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems (IEA/AIE 2015)*, volume 9101 of *Lecture Notes in Artificial Intelligence*, pp. 23–32. Springer, 2015.

Patric R. J. Östergård and Ville H. Pettersson. On the maximum length of coil-in-the-box codes in dimension 8. *Discrete Applied Mathematics*, 179:193–200, 2014.

Patric R. J. Östergård and Ville H. Pettersson. Exhaustive search for snake-in-the-box codes. *Graphs and Combinatorics*, 31:1019–1028, 2015.

Laurent Perron and Frédéric Didier. CP-SAT, google OR-Tools. https://developers.google.com/optimization/cp/cp_solver, 2024.

Walter D. Potter, Robert W. Robinson, John A. Miller, Krzysztof J. Kochut, and D. Z. Redys. Using the genetic algorithm to find snake-in-the-box codes. In *Proceedings of the 7th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems (IEA/AIE-94)*, pp. 421–426, 1994.

Gerhard Schrimpf, Johannes Schneider, Hermann Stamm-Wilbrandt, and Gunter Dueck. Record breaking optimization results using the ruin and recreate principle. *Journal of Computational Physics*, 159(2):139–171, 2000.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Paul Shaw. Using constraint programming and local search methods to solve vehicle routing problems. In *Principles and Practice of Constraint Programming (CP 1998)*, volume 1520 of *Lecture Notes in Computer Science*, pp. 417–431. Springer, 1998.

David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, and Demis Hassabis. A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science*, 362(6419):1140–1144, 2018.

Ed Wynn. Constructing circuit codes by permuting initial sequences. *arXiv preprint arXiv:1201.1647*, 2012.

A BEAM ANCHOR: ALGORITHM DETAILS

This appendix gives the implementation details omitted from Section 5.2, the training loop (Algorithm 1), the production settings (Table 4) and the settings of the reported runs.

A.1 ANCHOR SEARCH

Canonical prefixes. Searches and walks use the first-use coordinate ordering of Section 3, so each prefix represents its class under coordinate permutations, moves that would only relabel are excluded from $A(p)$, and equivalent prefixes share one row of $\mathcal{M}$.

Quotas and scoring. An anchor receives one pool of $B = nb$ beams, b being the nominal width per move; slot i of the pool belongs to the $\lfloor im/B \rfloor$-th of the m legal moves, so quotas differ by at most one. Each group starts from one live copy of its child with score zero. At every depth, a live state s_i with score ϕ_i and legal action u yields the candidate

$$\phi'_{i,u} = \phi_i + \log \pi_\theta(u \mid s_i), \qquad r_{i,u} = \phi'_{i,u} + T_{\mathrm{BA}}\, g_{i,u}, \quad g_{i,u} \sim \mathrm{Gumbel}(0,1),$$

with fresh noise at each depth. Each group keeps its own top candidates by r, up to its quota; survivors carry the clean score ϕ'. States reached twice, identified by a hash of the visited set and head, keep the larger score. Searches run until every beam has completed; the capacity H exceeds any snake in Q_n, so no search is truncated. If a group completes nothing beyond its child, the walk scores that move by $\ell(p) + 1$ and writes no label. Because $\mathcal{M}$ stores maxima over searches made under successive policies, its entries are optimistic attained lengths, not unbiased values.

A.2 WALKS AND TEMPERATURES

A cycle runs P walks in lockstep, one batched search per step sharded across devices, until every walk has no legal move. During training, each walk starts from a prefix drawn uniformly from all canonical snakes of length ℓ_0 (set to 10 in our experiments) to favor opening diversity; at search time with a frozen policy, walks start at the origin.

Three temperatures appear. Call an anchor *shallow* if its length is at most ℓ_{a} and *deep* if it is at least ℓ_{b}, and let $f \in [0,1]$ be the fraction of training completed.

- The *search noise* T_{BA} above is constant.
- The *walk temperature* τ_{walk} in $\mu(a \mid p) \propto \exp(\widehat{L}(p,a)/\tau_{\mathrm{walk}})$ is $\tau_{\mathrm{s}}(f)$ at shallow anchors, $\tau_{\mathrm{d}}(f)$ at deep ones, and linear in the anchor length between ℓ_{a} and ℓ_{b}, with

  $$\tau_{\mathrm{s}}(f) = 3.5 - 1.5f, \qquad \tau_{\mathrm{d}}(f) = 1.3 - 0.3f. \tag{4}$$

  Shallow anchors run hot because the opening decides which region a walk explores and the outcomes barely separate openings there; the decrease in f anneals exploration as the policy improves.
- The *teacher temperature* τ of Eq. 3 is 2 at shallow anchors, 1 at deep ones, linear between, and constant over training; it is independent of the walk temperature.

A.3 DISTILLATION

Let $\mathcal{S}(p) \subseteq A(p)$ be the moves at p with a recorded outcome. Training rows are prefixes with $|\mathcal{S}(p)| \geq 2$; the target and the policy in Eq. 3 are both restricted to $\mathcal{S}(p)$, so an unmeasured legal move receives neither target mass nor gradient. The table holds at most C rows under least-recently-used eviction. After each cycle's walks, the policy takes U Adam steps on minibatches of R rows with global gradient clipping; the first W cycles collect only.

A.4 NETWORK DETAILS

Of the eight node features in Section 5.2, the five indicator features are binary; the position along the snake is divided by the capacity H, and the counts of snake vertices at Hamming distance one and

Algorithm 1 Beam Anchor training (one cycle)

Require: policy π_θ, table $\mathcal{M}$, walks P, width per move b, noise T_{BA}, training fraction f, updates U, minibatch R

draw P start prefixes (uniform over canonical length-ℓ_0 snakes) and replay them into states
while some walk is live **do**
  for all live anchors p_j in parallel **do**
    $\widehat{\boldsymbol{L}}_j \leftarrow$ anchor search from p_j: one group per $a \in A(p_j)$, quotas summing to nb, run to completion
    $\mathcal{M}[p_j] \leftarrow \max(\mathcal{M}[p_j], \widehat{\boldsymbol{L}}_j)$ elementwise ▷ only the anchor row
    sample $a_j \sim \mu(\cdot \mid p_j)$ at temperature $\tau_{\mathrm{walk}}(\ell(p_j), f)$; $p_j \leftarrow p_j \parallel a_j$
  end for
  freeze walks with no legal move
end while
for U updates **do**
  sample R rows: a populated depth uniformly, then a prefix p with $|\mathcal{S}(p)| \geq 2$ uniformly
  update θ by Eq. 3 with $\tau = \tau(\ell(p))$, restricted to $\mathcal{S}(p)$
end for

two are divided by n and $\binom{n}{2}$. A linear layer with ReLU embeds the features into $\mathbb{R}^{64}$, and $K = 6$ residual GIN layers with unshared weights and no normalization apply

$$h_v \leftarrow h_v + \mathrm{MLP}_k\Big((1+\epsilon_k)\, h_v + \sum_{u \sim v} h_u\Big), \qquad k = 1, \ldots, K,$$

with $u \sim v$ over the n cube neighbors, ϵ_k a learned scalar initialized at zero, and MLP_k two ReLU layers of width 64. The action head is a two-layer MLP of width 64; illegal moves are masked to -10^9 before the softmax.

Shrinking-ball inference. The action head reads the final embeddings of the head h and its n neighbors, and each layer propagates one hop, so with $B_r(h)$ the Hamming ball of radius r around h it suffices to compute $h_v^{(0)}$ on $B_{K+1}(h)$ and the output of layer j on $B_{K+1-j}(h)$. The cube is vertex-transitive, so the index tables are static. For the six-layer Q_9 policy the balls have $502, 466, 382, 256, 130, 46, 10$ vertices: $1{,}792$ message-passing rows per state instead of $7 \times 512 = 3{,}584$, with identical logits up to summation order.

Critic head. The PPO baseline adds a value head to the same trunk, unused by Beam Anchor. With $\bar{h}_{\mathcal{X}}$ the mean of the final embeddings over a vertex set $\mathcal{X}$, and $\mathcal{V}$, $\mathcal{V}_{\mathrm{snake}}$, $\mathcal{V}_{\mathrm{avail}}$, $\mathcal{V}_{\mathrm{legal}}$ the sets of all, snake, available and legal next vertices,

$$V_\theta(p) = \mathrm{MLP}\big([h_v,\, \bar{h}_{\mathcal{V}},\, \bar{h}_{\mathcal{V}_{\mathrm{snake}}},\, \bar{h}_{\mathcal{V}_{\mathrm{avail}}},\, \bar{h}_{\mathcal{V}_{\mathrm{legal}}},\, (\ell(p)+1)/H]\big),$$

with v the head, $\ell(p)$ the length of p, H the capacity and a two-layer MLP of width 64. The means are invariant under vertex relabelings, so V_θ is invariant under the automorphisms of Q_n. Since they need final embeddings of every vertex, the PPO baseline runs the full forward pass rather than the shrinking-ball one.

A.5 Settings of the reported runs

All Beam Anchor runs use the settings of Table 4.

- *Beam Anchor, trained policies.* Two policies were trained from random initializations with different seeds (Section 5.2). Each was then frozen and run twice more, with different seeds, for 12 hours on one eight-device MI250 node, with the walk temperature held at its end-of-training value ($f = 1$ in Eq. 4) throughout the run.
- *Beam Anchor, uniform policy.* The training configuration of Table 4 with π_θ replaced by the uniform distribution over legal moves in every search and walk, run for 12 hours on one MI210 device with a single seed. Best length plateaued at 178.

Table 4: Beam Anchor production settings in Q_9. Length-dependent temperatures interpolate linearly between the shallow and deep thresholds; the walk temperature also decreases linearly over training (Eq. 4).

	Parameter	Setting
Search	width per move b; beams per anchor $B = nb$	192; 1,728
	search noise T_{BA}	0.25, constant
	action-sequence capacity H	192 (never binding)
Walks	walks per cycle P	16
	training start prefix length ℓ_0	10
	shallow / deep thresholds ℓ_{a}, ℓ_{b}	10, 20 moves
	walk temperature, shallow / deep anchors	$3.5 \to 2.0$ / $1.3 \to 1.0$ over training
Distillation	teacher temperature, shallow / deep anchors	2 / 1, constant
	table capacity C; eviction	10^5 rows; least recently used
	updates per cycle U; minibatch R	12; 512
	warm-up cycles W; cycles	5; 300
	optimizer; learning rate; clipping	Adam ($\epsilon = 10^{-5}$); 3×10^{-4}; 0.5
Network	GIN layers; width	6; 64

- *PPO, same network.* The same trunk with its critic head, trained by proximal policy optimization (Schulman et al., 2017) on episodes that build one snake from the origin under the same legality mask and ordering, with reward one per move and no curriculum or imitation. Rollouts of 256 environments for 128 steps, 32 minibatches, 4 epochs per update, discount 0.999, GAE $\lambda = 0.95$, learning rate 6×10^{-4} and entropy coefficient 0.1, both annealed towards 4×10^{-4} and 0.06 over 10^8 global steps. Four seeds, each run for 12 hours on one MI210 device; the best snake is the longest rollout during training and plateaued at 184 well before the end of training.

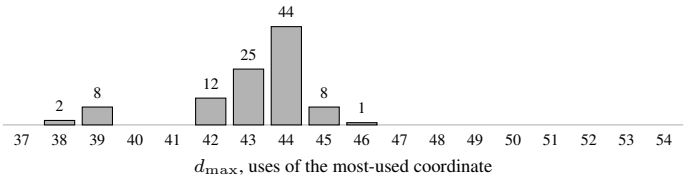


Figure 4: Distribution of $d_{\max}$ over the 100 length-190 classes found by Beam Anchor in Q_9. All lie in the continent's mode of the length-190 bank (Figure 3); none reaches the island's modes at $d_{\max} = 52$ and 54.

B Surgery, Annealing, and Lifting

B.1 Production algorithms and settings

The annealer's state is a snakepit, its energy is Eq. 1, and its temperature follows a sawtooth with period M:

$$T(t) = \max\left\{T_{\min}, T_0\left(1 - \frac{t \bmod M}{M}\right)\right\}. \tag{5}$$

The minimal move mixture (Algorithm 3) attempts surgery with probability p_s and otherwise splits evenly between vertex addition and removal; production runs also use the endpoint moves of Section 4.

Algorithm 2 Cube surgery as one annealing proposal

Require: snakepit F in Q_n, $n \geq 4$, temperature $T > 0$, penalty $\beta > 0$

save $F_0 \leftarrow F$, including all metadata
draw $d \sim \mathrm{Unif}\{2, 3, 4\}$ and a random d-subcube U
delete every occupied vertex in U
repeat
    $r \leftarrow 0$
    for all unoccupied vertices x of U, in a fresh random order **do**
        if x has an occupied neighbor and adding x preserves a snakepit **then**
            add x to F immediately; update metadata; $r \leftarrow r + 1$
        end if
    end for
    for all unoccupied vertices x of U, in a fresh random order **do**
        if adding x preserves a snakepit, including when x has no occupied neighbors **then**
            add x to F immediately; update metadata; $r \leftarrow r + 1$
        end if
    end for
until $r = 0$
$\Delta E \leftarrow E(F) - E(F_0)$ using Eq. 1
if $\Delta E > 0$ and $\mathrm{Uniform}(0, 1) \geq \exp(-\Delta E/T)$ **then**
    restore $F \leftarrow F_0$ exactly
end if
return F

Algorithm 3 Minimal snakepit annealing chain

Require: dimension $n \geq 4$, initial snakepit F, proposal budget N, $T_0, T_{\min}, M, \beta, p_s$

$S_{\mathrm{best}} \leftarrow F$ if $c(F) = 1$, otherwise no snake yet
for $t = 0, \ldots, N - 1$ **do**
    set T by Eq. 5
    draw $u \sim \mathrm{Uniform}(0, 1)$
    if $u < p_s$ **then**
        $F \leftarrow$ Algorithm 2(F, T, β)
    else if $u < p_s + (1 - p_s)/2$ **then**
        sample a cube vertex; if its addition is legal, test it using Eq. 1
    else
        attempt to sample an occupied vertex; if found, test removal using Eq. 1
    end if
    if $c(F) = 1$ and (no snake saved yet or $e(F) > e(S_{\mathrm{best}})$) **then**
        save a copy as S_{best}
    end if
end for
return the saved snake, if any, after independent validation

Table 5: Nominal forest-annealer settings; run-specific overrides are recorded in the release manifests. The minimal mixture splits nonsurgery proposals equally between addition and removal; production endpoint-move probabilities depend on the recipe.

Parameter	Nominal production setting
Peak / minimum temperature	2.0 / 10^{-4}
Fragmentation penalty	$\beta = 3.2$, per-chain jitter ± 0.30
Surgery probability / dimension	0.35 / uniform $d \in \{2, 3, 4\}$
Reheating periods	$M \in \{64, 128, 256\} \times 10^6$ moves
Chains / device; moves / launch	131,072; 2,000,000
Stall threshold; bank reseeding	30 launches; probability 0.5
Wiggle / deep regrowth	retract 1–6 / 8–48; grow up to $j + 2$ / $K_{\mathrm{removed}} + 12$

Compound proposals (surgery, wiggle, deep regrowth) make one acceptance decision and restore the saved state exactly on rejection. Chains run in parallel on GPUs; a quarter of them run cool, and reseeding draws a random temperature phase. Chains save every single snake above a floor ($20, 40, 80, 170, 340, 700, 1400, 2800$ in dimensions 6–13), with deduplication. The first successful Q_9 chains started from the 200-edge Q_6 product; the 208-edge Q_5 product and banked snakes also led to 191s.

Implementation. Occupancy is a bitset (512 bits in Q_9) with vertex, component and coordinate-use counts. Each endpoint stores its opposite endpoint, so a join that would close a cycle is detected in constant time. The same C core compiles for CPU and GPU and was fuzz-tested against brute force in dimensions 6–13.

B.2 Surgery comparison and annealing trace

We compared two fixed annealing recipes in Q_9, each initialized from an empty snakepit and run for 3,600 seconds on one dedicated CPU core for each of ten paired seed identifiers (0–9). Recipe A used single-vertex addition/removal with probabilities $0.5/0.5$; recipe B used surgery/addition/removal with probabilities $0.35/0.325/0.325$, sampling surgery dimension uniformly from $\{2, 3, 4\}$. Both used $\beta = 3.2$ and the sawtooth temperature schedule with $T_0 = 2$, $T_{\min} = 10^{-4}$ and period 64 million attempted proposals. Both allowed fragmented states. Endpoint moves, auxiliary objectives, restarts, reseeding and random temperature phases were disabled; no witness bank was supplied.

Both recipes used the same binary, and the ten pairs ran concurrently on distinct cores of one CPU socket; all 20 runs completed, with no reruns or tuning after the fact. Only a single snake counts toward the best length, and every reported snake was checked independently.

Results. The median final best snake length was 149 edges (range 147–155) for A and 188 (range 186–191) for B. Every paired difference favored B, with median 39 edges and range 35–41 (Table 6). Recipe B with seed 4 found a validated 191-edge snake after 158.484 seconds and 302,413,283 proposals.

Table 6: All ten paired outcomes after 3,600 seconds per recipe and seed in Q_9. A: add/remove; B: surgery/add/remove. Proposals count every attempt, including illegal and no-op proposals; full cycles count completed 64-million-proposal periods.

	Best snake edges			Proposals (10^9)		Full cycles	
Seed	A	B	B−A	A	B	A	B
0	152	188	36	16.015	6.686	250	104
1	147	186	39	15.300	6.384	239	99
2	149	187	38	16.151	6.735	252	105
3	148	188	40	16.194	6.753	253	105
4	155	191	36	16.145	6.733	252	105
5	152	187	35	16.142	6.715	252	104
6	148	187	39	16.148	6.739	252	105
7	149	189	40	16.131	6.739	252	105
8	147	188	41	16.202	6.720	253	105
9	149	188	39	16.161	6.727	252	105

Table 7: Best-so-far validated single-snake edge lengths at the four prespecified checkpoints, across ten runs per recipe. Each checkpoint is recorded at the first completed proposal after its deadline.

Elapsed checkpoint (s)	A median [range]	B median [range]
90	143 [143, 147]	181 [175, 184]
240	146 [143, 147]	182.5 [181, 191]
900	147 [144, 155]	185.5 [184, 191]
3,600	149 [147, 155]	188 [186, 191]

Table 7 shows progress at the prespecified checkpoints. The equal-time comparison measures the performance of these two recipes: A completed 239–253 temperature cycles, versus 99–105 for B. Their proposal counts and temperature trajectories therefore differ; the comparison does not isolate surgery under a matched annealing trajectory or establish that surgery is necessary.

Annealing trace. Traced chains started from the 256 isolated even-parity vertices show the intended cycle: hot phases carry more components and more edge changes, and cooling consolidates the pit. Few or no vertices stay occupied throughout a trace, and 120 CPU-hours of such chains recovered 86 known 190 classes and 27 known 191 classes, from both families.

B.3 Beam lifting

For a parent state s of edge length L, let $R(s)$ contain the unvisited vertices whose visited neighbors, if any, consist only of the current head. These vertices have not yet been excluded by the path, although most are not legal next steps. Let $n_{\mathrm{legal}}(s)$ count the legal one-edge extensions. A child s' receives score

$$\mathrm{score}(s') = L + 1 + a_{\mathrm{reach}}\big(|R(s)| - n_{\mathrm{legal}}(s)\big) + T_{\mathrm{lift}}\, g(s'), \tag{6}$$

where g is a standard Gumbel draw and $T_{\mathrm{lift}} \geq 0$ controls the selection noise. Equivalently, the deterministic part is the child's edge length plus $a_{\mathrm{reach}}|R(s')|$.

The count update is especially simple in a hypercube:

$$|R(s')| = |R(s)| - n_{\mathrm{legal}}(s).$$

Once a new head is chosen, all previously legal neighbors of the old head leave R: the chosen vertex becomes occupied, while the others become forbidden by the old head. No two such neighbors are adjacent, so none can remain available through adjacency to the new head. Other vertices in R remain available. The deterministic score is therefore shared by siblings and can be computed before materializing their states; noise distinguishes them.

Algorithm 4 Lifting a primer from Q_{n-1} into Q_n

Require: dimension n, primer snake, beam width B, noise scale T_{lift}, a_{reach}, seed
1: $\mathcal{S} \leftarrow \{$the primer, placed in $Q_n\}$; $\quad \mathrm{best} \leftarrow$ primer length
2: **while** $\mathcal{S} \neq \emptyset$ **do**
3: $\quad \mathcal{C} \leftarrow$ all one-edge extensions of states in $\mathcal{S}$ that remain induced paths
4: $\quad$ score children by Eq. 6; remove duplicate states; $\mathcal{S} \leftarrow$ the top B children
5: $\quad$ update best from generated children, if any
6: **end while**
7: **return** the longest snake found

The record recipe uses widths $B = 2^{16}$ over four seeds and $B = 2^{18}$ over two, with $T_{\mathrm{lift}} = 0.5$ and $a_{\mathrm{reach}} = 1$. States are deduplicated by a 64-bit hash of the visited set. Lifting all 131 known 191s recovered the same 12 length-379 classes in Q_{10}; the tested Q_8 primers did not yield 190 in Q_9.

C Records and Coils

C.1 Snake records

The Q_{12} route refined a 1470 lift to 1474, then re-lifted the improved Q_{11} bank and annealed to 1476. The Q_{13} route refined 2916 lifts to 2918 and 2919, then annealed a 2921 seed from the genetic search below to 2924. Class counts above Q_9 are harvest counts, not completeness results.

MILP window refill. Our genetic search refills a window of consecutive path vertices exactly. With the path outside the window fixed, let W consist of the two cut endpoints r and s and the unused vertices adjacent to no fixed path vertex other than r or s. With binary x_v (vertex $v \in W$

chosen) and y_{uv} (cube edge uv inside W used), and flows f on directed edges, the refill solves

$$\max \sum_{v \in W} x_v \quad \text{s.t.} \quad x_r = x_s = 1, \quad y_{uv} = x_u x_v \text{ (linearized)},$$
$$\sum_u y_{uv} \le 2x_v \ (\le 1 \text{ at } r, s), \quad \sum y = \sum x - 1,$$

with a single-commodity flow from r that must reach every chosen vertex. Setting $y_{uv} = x_u x_v$ forbids chords; the degree, edge-count and connectivity constraints make the choice a single path from r to s. Unlike the greedy refill of cube surgery, this solves the window exactly at the cost of one solver call. Scans sweep windows covering 1%, 5%, 10% and 20% of the path, ten per size, plus windows containing the endpoints. This produced the 2921 seed in Q_{13} and further record-length classes in Q_{11}.

C.2 Coils

Closure. A length-L snake whose endpoints are at distance two closes into a length-$(L+2)$ coil when a common neighbor of the endpoints is unused and adjacent to no other path vertex. Of the $1{,}042$ length-190 classes in Q_9, 229 have distance-two endpoints and 96 close, all to the same 192-coil.

Harvest and close. No record-length snake in Q_{10}–Q_{13} closes. We therefore ran the annealer warmer and with a lowered save threshold, and tested the harvested shorter snakes: 24 million in Q_{10}, 19 million in Q_{11} and 14 million in Q_{12}, giving the coil records 374, 734 and 1466.

MILP closure. The refill above adapts to coils by requiring the refilled window to join the two ends of the path into one cycle. Starting from a length-2932 snake of Echols whose endpoints were at distance 8, it produced the 2916-coil in Q_{13}.

C.3 Symmetric coils

Definition 4 (Half-turn symmetry)**.** A coil $C = (v_0, \ldots, v_{2h-1})$ in Q_n is *σ-symmetric* for an involutive cube automorphism $\sigma(x) = P(x) \oplus t$, where P permutes coordinates, $P^2 = I$ and $P(t) = t$, if $v_{i+h} = \sigma(v_i)$ for all i (indices mod $2h$).

Equivalently, the second half of the transition sequence is the first half with coordinates permuted by P; translation symmetry is the case $P = I$. For fixed σ, the half $v_0, \ldots, v_{h-1}$ determines the coil: it must be an induced path disjoint from its image, joined to it by exactly the edges $v_{h-1}\sigma(v_0)$ and $\sigma(v_{h-1})v_0$. Every proven coil optimum in dimensions 2–8 has a σ-symmetric representative.

Symmetric surgery. The annealer keeps its occupied set invariant under σ: every addition or removal acts on the pair $\{v, \sigma(v)\}$, surgery clears $U \cup \sigma(U)$ and refills in pairs, and each paired edit gets one acceptance decision. Cycle components are allowed so that fragments can close up, and the energy is $\beta|c-1| - |V|$. Maps with $\min_x d(x, \sigma(x)) < 2$ are skipped. A state is saved only if it is a single induced cycle satisfying Definition 4.

Half-path Beam Anchor. Beam Anchor builds only the half-path: adding u also occupies $\sigma(u)$, and a move to u is legal only if u and $\sigma(u)$ are unused and nonadjacent and the head is u's only occupied neighbor. A move closes the coil when its only occupied neighbors are the head and $\sigma(v_0)$; the symmetric edge then closes the other side, giving a coil of length $2h$. A closure scores $10^5 + 2h$ and an open dead end scores 1, so the anchor comparison favors moves that lead to long closures.

Sweep. Symmetric surgery, run on every fixed-point-free class of σ in Q_8–Q_{11} with one worker per class, reached 96, 192, 372 and 734, matching the optimum or best known coil length in every dimension but Q_{10}. Half-path Beam Anchor reached 48, 92, 176, 370 and 730 in Q_7–Q_{11}.

C.4 CONTINENT AND ISLAND

From a continent seed, add/remove, wiggle, surgery and deep regrowth recovered 96, 125, 127 and 127 of the 127 continent classes, and never an island class. Only regrowth steered toward the island's coordinate profile crossed, after truncating a continent snake to length 130. The same split appears at length 190: 27% of continent-type and 5% of island-type 190s extend to 191, and no simple statistic we measured separates extendable 190s from dead ends. The 190 bank looked saturated at $1{,}041$ classes until a new scaffold-based sampler found a $1{,}042$nd, a dead end with $d_{\max} = 49$. Generators trained on the bank (transformer, PatternBoost-style and GNN policies) likewise produced no 191 outside it, and produced 191s at all only when trained on extendable 190s.

D Snakepit Benchmark and Mathematical Support

D.1 Bounds and small values

Huang's sensitivity theorem shows that for $n \geq 5$ any set of more than 2^{n-1} vertices induces a vertex of degree at least $\sqrt{n} > 2$ (Huang, 2019). Hence every snakepit has at most half the cube's vertices:

$$L(n,k) + k \leq 2^{n-1}, \tag{7}$$

which also holds at $n = 4$ by enumeration. With $k_{\min}(n) = \min\{k : L(n,k) = \Lambda(n)\}$, the exact initial terms are

$$\big(\Lambda(n)\big)_{n=1}^{7} = (1,2,4,7,13,26,52), \qquad \big(k_{\min}(n)\big)_{n=1}^{7} = (1,1,1,1,1,1,4).$$

In Q_7 every optimum has exactly four components: $L(7,k)$ for $k = 1,\ldots,8$ is $50, 50, 50, 52, 51, 50, 49, 48$, and 48 persists through $k = 16$.

D.2 Exactness in dimension 7

The exact values for $n \leq 6$ come from exhaustive enumeration. For Q_7 the proof is computer-assisted. For a split along coordinate i, write $L = L_0 + L_1 + c_i$, where each layer is a snakepit in Q_6 and c_i counts cross edges. Suppose $L \geq 53$ and choose i least used, so $c_i \leq \lfloor L/7 \rfloor$; the heavier layer then has $23 \leq L_0 \leq 26$ edges. Each cross edge needs an endpoint or isolated vertex in that layer, so $k_0 \geq \lceil c_i/2 \rceil$ and $|S_0| = L_0 + k_0 \geq 27$ vertices. A depth-first enumeration of all such layers containing a fixed degree-two pattern at the origin gives $4{,}576{,}964$ labeled sets and $5{,}041$ representatives under the automorphisms of Q_6. For each representative, a CP-SAT model (Perron & Didier, 2024) enforcing induced degree at most two, acyclicity, the minimal split direction and the heavier-layer convention decides whether some second layer completes it to 53 edges; all $5{,}041$ instances are infeasible, with no time-outs. Controls recover $\Lambda(6) = 26$ through Q_5 layers and find feasible Q_7 completions at 52.

At target 52, the heavy layer has at least 26 vertices and there are $12{,}199$ representatives. Completion with fixed component count excludes $k = 2, 3$ and 5–12; the known snake optimum excludes $k = 1$, and Eq. 7 excludes $k \geq 13$. The 13 feasible representatives at $k = 4$ produce 169 projected completions, reducing to 63 classes under cube automorphisms. Every class consists of four length-13 paths with an edge-free coordinate; 62 classes have two such coordinates and one has exactly one. The same procedure at targets 51, 50 and 49, with $60{,}708$, $117{,}445$ and $496{,}524$ heavy-layer representatives, closes the remaining cells of the Q_7 row without time-outs.

D.3 Proof of Proposition 1

Let $d_n = D(n)/2^{n+1}$ and $u_n = \Lambda(n)/2^n$.

Products. For vertex-disjoint A, B in Q_a, the product of Definition 3 in Q_{a+b} and the product with the roles of A and B swapped are disjoint, so $D(a+b) \geq 2^b D(a)$ and d_n is nondecreasing. A single product with $b = 1$ gives $\Lambda(n) \geq D(n-1)$, that is, $u_n \geq d_{n-1}$. By Eq. 7, $D(n) \leq 2\Lambda(n) \leq 2^n$, so $d_n \leq \frac{1}{2}$.

Splitting. Let F be an optimal pit in Q_n and choose a coordinate i used by at most $\Lambda(n)/n$ of its edges. The translate $F \oplus e_i$ meets F exactly in the endpoints of these i-edges. Deleting both endpoints of each one from the translate removes at most three edges per pair, since both have degree at most two and share the i-edge. The result B is disjoint from F, so $D(n) \geq e(F) + e(B) \geq (2 - 3/n)\Lambda(n)$, that is, $u_n \leq d_n/(1 - 3/(2n))$.

Together, $d_{n-1} \leq u_n \leq d_n/(1 - 3/(2n))$ with d_n nondecreasing and bounded, so u_n and d_n share the limit $\rho = \sup_a d_a \leq \frac{1}{2}$. The disjoint length-13 snakes in Q_5 give $D(5) = 26$ and $\rho \geq 13/32$. □

For a disjoint pair with unused vertex set R, the deficit from $\frac{1}{2}$ is exactly $(|R| + c(A) + c(B))/2^{a+1}$, since each component has one more vertex than edges. The Q_5 pair has deficit $6/64$. Its products attain $D(6) = 52$ and $D(7) = 104$, which are exact by $D(n) \leq 2\Lambda(n)$; the first open base that could improve $13/32$ is Q_8, where a pair needs at least 209 edges.

D.4 BENCHMARK CALIBRATION

The pit-mode annealer records the best $e(F)$ for each component count it visits, with $\beta = 1$ so that the energy is $-e(F) - 1$. On Q_8, a penalty ramp from 1.2 to 4.0, surgery probability 0.38 and wiggle probability 0.28 reached 98 in each of four 40-minute runs from the empty pit, against 95–96 with fixed $\beta = 3.2$. Neither 32 runs in Q_8 nor 24 two-hour runs in Q_9 exceeded the product values 104 and 208.

E RECORD SNAKES

Each snake is written as its transition sequence: starting from the origin, the jth entry is the coordinate flipped at step j, with coordinates numbered from 0. A length-191 snake in Q_9 from the continent, with DCV $(39, 38, 35, 29, 26, 9, 9, 4, 2)$:

```
0 1 2 0 3 1 4 5 0 6 2 1 0 6 5 0 2 6 3 1 4 3 2 1 0 3 5 0 2 3
6 2 1 0 6 2 3 6 0 7 4 0 6 3 2 6 0 1 2 6 3 2 0 6 4 8 3 0 2 6
3 1 5 3 2 6 3 1 2 0 3 6 2 3 4 0 2 3 6 2 0 1 6 2 3 6 0 7 5 0
6 3 2 6 0 1 2 6 3 2 0 6 5 2 3 6 2 0 1 3 4 1 2 0 1 3 2 6 1 0
2 1 5 6 2 1 0 2 6 3 0 2 1 0 6 7 4 6 0 1 2 0 6 3 2 0 1 2 6 0
4 8 1 6 2 0 1 3 5 1 2 0 1 3 2 6 1 0 2 1 4 6 2 1 0 2 6 3 0 2
1 0 6 7 5 6 0 1 2 0 6
```

A length-379 snake in Q_{10}, with DCV $(81, 63, 62, 61, 60, 19, 16, 12, 4, 1)$:

```
0 1 2 3 0 2 4 1 0 2 3 0 1 5 3 0 2 3 6 4 2 3 0 2 1 6 0 2 3 0
7 5 1 2 4 5 3 4 2 0 3 2 1 4 3 2 0 5 8 7 1 0 2 4 1 2 3 0 1 2
4 1 7 5 4 1 2 4 3 0 2 3 5 4 2 3 0 2 1 8 0 1 4 2 3 6 4 2 1 4
7 0 1 4 2 1 0 3 2 1 4 2 0 1 5 0 3 7 4 3 2 1 4 2 0 3 4 2 1 4
3 5 1 4 2 1 8 0 2 1 4 2 0 3 4 2 1 7 0 1 2 3 0 2 4 1 0 2 3 7
6 5 4 3 0 7 1 2 4 1 0 3 4 1 2 4 6 3 2 4 1 2 3 0 7 4 3 0 2 3
4 1 2 3 0 2 4 3 7 1 9 7 4 1 2 4 3 0 2 4 1 2 3 4 7 3 0 5 1 0
2 4 1 2 3 0 1 2 4 7 6 4 1 7 0 1 2 3 0 2 4 1 0 2 3 0 1 5 3 0
2 3 6 4 2 3 0 2 1 6 0 2 3 0 7 4 3 2 1 4 2 0 3 2 1 4 5 6 3 0
4 1 2 0 3 4 6 3 2 4 1 2 0 8 1 2 4 1 0 2 3 5 4 3 2 1 4 2 0 3
4 2 1 7 0 4 5 0 1 4 2 1 0 3 2 1 4 2 0 1 5 2 1 4 2 0 3 4 6 3
2 4 1 2 0 3 1 2 4 7 3 4 2 0 3 2 1 4 3 2 0 5 1 3 7 1 0 3 2 0
1 4 2 0 3 2 1 0 7 2 0 3 2 1 4 3 6 4 2
```