Exact Escape Times of Langton's Ant in Four Rows

A finite persistent-response classification and an all-width certificate

Hao Ke

Abstract

How long can Langton's ant remain inside a rectangle with four rows? We maximize the time of its first exit over every initial colouring and every starting position and heading. For a rectangle of width n, the exact maximum is $S_{4,n} = 34n - 150$ for all $n \geq 13$; we also determine every smaller case. This proves Etse's conjectured upper bound $S_{4,n} \leq 34n$ and matches his lower-bound family from width fourteen onward. The main difficulty is that a returning ant encounters the colours changed on its previous visits, so a block cannot be replaced by a device that resets after each crossing. We classify blocks by their responses to repeated entries, retaining the changed response after each return. Exactly 269 response types occur for each choice of entry side and parity, and this collection is closed under adjoining a column. A weighted version gives an exact recurrence for the maximum escape time. Finite integer inequalities then prove its affine behaviour for every larger width. The proof combines an induction covering arbitrarily wide rectangles with exact finite certificates, and reconstructs colourings attaining each maximum.

1 The exact question and answer

A finite rectangle contains only finitely many colourings, but enlarging its width creates exponentially many initial states. We seek the longest possible residence time for every width at once. Four rows provide a case where the answer is exactly linear and the effect of every return to a previously visited block can still be classified by a finite interface. The ant moves on $\mathbb{Z}^2$. At a white cell it turns right, changes the cell to black and moves forward by one unit; at a black cell it turns left, changes the cell to white and moves. Headings N,E,S,W are $0, 1, 2, 3$, with step vectors $(0, 1), (1, 0), (0, -1), (-1, 0)$, respectively. A state at time t is observed before update t. A pose consists of a position and a heading. For

$$R_n = \{0, \ldots, n-1\} \times \{0, 1, 2, 3\},$$

the first-exit time counts the update that moves the head out of R_n. The quantity $S_{4,n}$ is maximized over all 2^{4n} colourings, all $4n$ starting cells and all four headings. Colors outside R_n are arbitrary and remain unread and unchanged before this first exit. No boundary reflection or rule change is imposed.

Theorem 1. *The small values are*

n	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*
$S_{4,n}$	*2*	*18*	*28*	*46*	*64*	*86*	*104*	*146*	*156*	*206*	*224*	*266*

For every $n \geq 13$,

$$S_{4,n} = 34n - 150. \tag{1}$$

Every maximum is attained by a complete finite colouring and an effectively reconstructible internal pose.

Etse [1] asks whether $S_{4,n} \le 34n$ in Conjecture 11 of the published MFCS version, Section 4.3.3, p. 49:14. The adjoining text and Figure 13 give lower constructions of size $34n - 150$ for $n \ge 14$ and finite experimental agreement. The full arXiv version numbers this as Conjecture 13. Theorem 1 proves the conjectured upper bound and establishes the exact maximum for every width. Etse already supplied the lower-bound family and finite-width computations; the new step is a complete persistent-interface classification and an all-width upper certificate. Our attainment construction and small-width checks make the resulting exact classification reproducible. The coefficient 34 is consequently optimal. The first-exit convention agrees with [1]; our reduction includes all internal poses and does not rely on a boundary-start lemma.

The proof has three stages. Section 2 replaces each coloured block by the responses that a caller can observe after repeated entries, with all colour changes retained. Section 3 uses the finite response catalogue to bound local visits. Sections 4–5 attach durations and prove an exact recurrence by integer induction. Closure under adjoining a column gives coverage at every width; the weighted induction gives the exact escape time.

2 Persistent one-sided responses

A block is a contiguous sub-rectangle of four rows and any finite number of columns. Fix its near side, $s = 0$ for left or $s = 1$ for right. The other vertical boundary is its far side. A caller enters horizontally through the near boundary and observes the side and row at which the ant next leaves.

Put $\epsilon = (h - x - y) \bmod 2$. Each update changes both heading parity and cell parity, so ϵ is invariant. At column x write $p = (x + \epsilon) \bmod 2$; the parity of a block means that of its near column. On horizontal entry h is odd, so the entry row satisfies $y \equiv 1 - p \pmod 2$. There are exactly two such rows,

$$I_p = (1 - p, 3 - p).$$

A horizontal exit enters the neighbouring column of parity $1-p$, so its row lies in $U_p = I_{1-p} = (p, p+2)$. Each list has channels $0, 1$ in the order displayed. A four-cell column has mask $b = \sum_{y=0}^{3} \mathbf{1}_{\text{black}}(y) 2^y$.

A caller may repeatedly choose one of the two incoming channels at this side. Each request initiates a literal excursion until the head exits the block. A near-side exit returns its channel and the behaviour of the entire resulting colouring. A top, bottom or far-side exit is terminal. The colouring is never reset between requests.

Definition 2. A finite response type is an inductively generated finite tree represented by a pair $t = (\rho_0, \rho_1)$, where $\rho_i = \perp$ or $\rho_i = (j, t')$, with $j \in \{0, 1\}$ and t' a finite response type. Its rank is

$$r(t) = \max(\{0\} \cup \{1 + r(t') : \rho_i = (j, t')\}).$$

The empty block has type $t_0 = (\perp, \perp)$.

We do not assume in advance that every physical block has a finite response type. This will follow from a checked closed algebra. A type forgets the duration and the distinction among terminal sides, since there can be no subsequent request after a terminal. It retains every channel and residual response after a successful return. It is observational equivalence for this interface: different physical colourings have the same type precisely when their future interface responses agree. Durations are reintroduced in Section 4.

2.1 Adjoining a column

Suppose a near column has mask b, parity p, and a child block has near side s, parity $1-p$, and type r. Denote the combined type by $\Phi_{s,p}(b,r)$. A literal one-column excursion takes at most two updates: a horizontal incoming heading makes a vertical move and then a horizontal move, unless a vertical boundary has already been crossed; a vertical incoming heading leaves horizontally in one update. Thus its transition table is an exact finite table.

For a parent request execute this one-column table. A vertical exit terminates. A near exit returns its channel, with residual product $\Phi_{s,p}(b',r')$. A far exit enters the child at its corresponding channel. A terminal child answer terminates the parent. Otherwise retain both the changed column mask and the child's returned residual type, reenter the column from its far side and continue. Each successful child call lowers its rank. A proposed finite product table is checked against these equations; on a parent return, the resulting type must be exactly the declared residual product and have smaller rank than the parent type.

Proposition 3 (Finite closure certificate)**.** *For each $(s,p)\in\{0,1\}^2$ there are 269 distinct canonical response types. They include t_0, have ranks at most 13, and are closed under all 16 column masks and all 269 child types. All $4\cdot 16\cdot 269 = 17,216$ product cases are verified for both input channels. The realized type sets by width have sizes*

width	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*
size	*1*	*12*	*36*	*83*	*158*	*240*	*264*	*269*	*269*

The sets at widths seven and eight are equal, not merely equinumerous. Every catalogue type has a physical representative of width at most seven.

Proof and infinite-width interpretation. The finite facts are checked by the supplied reconstruction and independent verifier. Nodes are canonical pairs of terminal or channel/child responses; acyclicity and ranks are checked. Product equations are verified by literal one-column updates, preserving the child's residual state. For coverage, induct on width. The empty block has t_0. Every wider colouring is exactly a near column and a smaller coloured block. By the induction hypothesis the latter has its exact response type. The checked equations therefore simulate the real parent and establish its exact residual type. Ranks make every request and every permitted series of returns finite. The closed table ensures the result stays in the catalogue for every width. The displayed representatives, independently replayed as complete response trees, prove realizability. Distinct canonical trees have some finite request sequence distinguishing them. Thus the catalogue is exact for the stated interface. □

Writing $\mathcal{T}_w^{s,p}$ for the realized types at width w, composition gives

$$\mathcal{T}_{w+1}^{s,p} = \{\Phi_{s,p}(b,r) : 0\le b<16,\ r\in\mathcal{T}_w^{s,1-p}\}.$$

Equality of the width-seven and width-eight sets for both parities therefore implies stabilization at every larger width. Each block permits at most 13 successful returns, and this rank is attained by a compatible request sequence.

3 A local bound independent of width

Consider a column at its first visit during a rectangle-confined run. Its left and right parts have arbitrary complete colourings, possibly modified by earlier motion, and hence have exact catalogue

types. Replace an excursion into either side by its persistent response. A return lowers the sum of the two ranks. Between responses there are at most two central-column updates. This product terminates without any assumption on the widths of the side blocks.

Proposition 4 (Sharp column and cut bounds)**.** *Before first escape from any four-row rectangle, a fixed column is updated at most 34 times. A vertical cut is crossed at most 21 times. Both bounds are attained.*

Proof. The central-column check enumerates the two parities, all pairs of 269 side types, 16 masks, and eight compatible internal row/heading poses. For each of the four rows, $h - y \equiv p \pmod 2$ allows two headings, giving exactly $2 \cdot 269^2 \cdot 16 \cdot 8 = 18,524,416$ cases. The largest local count is 34. The cut handshake chooses two parities, two entry sides, two entry channels and a pair of block types: $2 \cdot 2 \cdot 2 \cdot 269^2 = 578,888$ cases, counting the first entry crossing; its largest count is 21. Each internal return lowers a checked rank, so these are exhaustive finite products, not truncated simulations. For sharpness of the column bound, the masks

$$(15, 9, 9, 9, 7, 1, 11, 6, 6, 6, 0)$$

with head $(5, 1)$ facing west give 159 total updates and exactly 34 in column 5. For the cut bound, masks $(11, 6, 6, 2, 13, 12, 4, 12, 6, 13)$ with head $(4, 2)$ facing south give 164 updates and 21 crossings between columns 4 and 5. Both witnesses are autonomous runs with their complete initial colourings. □

Summing column counts proves $S_{4,n} \leq 34n$ immediately. This already resolves the published conjecture. It does not yet give the subtractive constant 150, because maximal column behaviours need not be simultaneously attainable. The next reduction addresses precisely that compatibility issue.

4 An exact weighted recurrence

The duration of a block cannot be maximized independently on each call: the same colouring must persist across all of them. We retain the whole finite request script in a weighted query. We construct rectangles by adjoining columns on the left, so all weighted queries use near side $s = 0$. A script records the incoming channels; the response type determines all returned channels and residual types along that script.

An *ordinary query* $F(p, t, q)$ specifies parity p, response type t and a finite channel word $q \in \{0, 1\}^*$. All requests except possibly its last must return; the last may return or terminate. Write $V_n(F(p, t, q))$ for the maximum total updates of this script over all width-n colourings of type t. If no such colouring exists, the value is $-\infty$. The empty script has zero cost only when its type is realizable at that width. Rank at most 13 permits at most thirteen successful returns, optionally followed by one terminal request. Relevant scripts therefore have at most fourteen requests; the same bound applies to the child scripts generated by the decomposition below.

A *rooted descriptor* o is terminal or a pair (j, t): the first run starts somewhere inside the block and either terminates or returns at left-side channel j, leaving ordinary type t. There are $1 + 2 \cdot 269 = 539$ descriptors per parity. A rooted query $R(p, o, q)$ performs this initial run followed by q. If o is terminal, q is empty. Its value maximizes over all complete colourings and all internal starting poses realizing that descriptor.

4.1 Complete decomposition rules

For an ordinary query enumerate every (b, r) with $\Phi_{0,p}(b, r) = t$. Run the parent script symbolically through that column and its child's exact response tree. The result is a local count c and a child script q'. This is an edge to $F(1 - p, r, q')$ of weight c.

For a rooted query there are two, and only two, possible locations for the root. If it is in the new column, enumerate all eight compatible poses, the mask and the child's ordinary type, run the initial excursion, then the parent script. The successor query is ordinary F. If the root is in the child, enumerate its rooted descriptor. A terminal child first run gives a terminal parent without touching the new column. Otherwise its output channel starts the new-column motion. The later calls form the continuation script of a rooted child query R. In both cases count only updates in the new column in c; the child's query counts its entire history exactly once.

The starting goals are all rooted descriptors with empty continuation. Closing these rules backwards yields 21,248 states and 527,764 edges, with 1,078 goals. The graph is finite because the catalogue and bounded scripts are finite. The verifier checks every required successor and edge, both root placements, and reachability of every stored state from the goal set.

Use $V_n(v)$ for the value of either kind of query. The recurrence is

$$V_0(v) = \begin{cases} 0, & v = F(p, t_0, q),\ |q| \le 1, \\ -\infty, & \text{otherwise}, \end{cases}$$
$$V_n(v) = \max_{v \xrightarrow{c} u} (c + V_{n-1}(u)). \tag{2}$$

An empty block terminates its first ordinary request without an update and has no internal starting root. These are exactly the stated base cases.

Lemma 5 (Physical exactness)**.** *The values in (2) equal the physical maxima of their queries. Every finite value is attained. Consequently*

$$S_{4,n} = \max_{v \in \mathcal{G}} V_n(v), \tag{3}$$

where $\mathcal{G}$ is the set of all 1,078 goals.

Proof. Induct on width. A real colouring splits into its near-column mask and its actual child. For rooted queries the initial head is either in that column or in the child. These exhaustive alternatives select one enumerated edge. Persistent response equivalence forces the recorded child script, so the time is local weight plus one child query value. This proves the upper inequality. Conversely, take an attaining child configuration for a finite successor value and prepend the edge's recorded column. If appropriate, use its recorded initial pose; otherwise retain the child's root. The exact response equations ensure that all successive calls are made to this same child with its evolving colours. They give the parent duration and descriptor claimed by the edge. This proves attainment and the lower inequality. Maximizing over all rooted outcomes gives (3). No boundary-entry assumption is used. □

Rooted queries supply the internal starting poses, while whole-script queries preserve compatibility across repeated entries into the same child.

5 All-width affine stabilization by local integer inequalities

The recurrence through width 23 is computed exactly. Define integer vectors on all 21,248 states by

$$A = V_{20}, \quad B = V_{21}, \quad d = V_{22} - V_{20}, \quad e = V_{23} - V_{21}.$$

The certificate checks that A and B are finite everywhere. On every edge $v \xrightarrow{c} u$ it verifies

$$B(v) \geq c + A(u), \qquad e(v) \geq d(u), \tag{4}$$

$$A(v) + d(v) \geq c + B(u), \qquad d(v) \geq e(u). \tag{5}$$

For each vertex there is an edge attaining both inequalities in (4), and an edge, possibly different, attaining both in (5). There are 2,111,056 integer inequalities and 42,496 tight-edge existence checks.

Lemma 6 (Explicit affine certificate)**.** *These checked inequalities and tight edges imply, for every* $k \geq 0$*,*

$$V_{20+2k} = A + kd, \qquad V_{21+2k} = B + ke.$$

Proof. Substitution into (2) gives $B + ke$ as an upper bound by (4); a simultaneously tight edge attains it. Substituting once more, (5) gives and attains $A + (k+1)d$. The base values are the exact computed V_{20}, V_{21}. This proves the assertion by induction, not by an empirical period-detection rule. □

For every goal the checked envelopes are

$$A \leq 530, \quad B \leq 564, \quad d \leq 68, \quad e \leq 68.$$

There is a goal with $A = 530, d = 68$ and a goal with $B = 564, e = 68$. Lemma 6 and (3) give $34n - 150$ for all $n \geq 20$. The goal maxima computed directly at widths 13 through 19 already have this affine value, before the full state-vector induction starts at width 20. These values and the twelve smaller widths prove Theorem 1.

This is also a constructive lower-bound proof. At each width choose an attaining edge, output its near-column mask and recurse on the child. Exactly one rooted to ordinary transition supplies the initial head. The path ends at a valid empty block base case. Thus every n yields a complete maximizing seed, with n explicit masks. Its reconstruction is an output-sensitive finite algorithm, not a demand to simulate an unbounded run. In particular the asymptotic lower coefficient 34 is established independently of the earlier lower construction.

6 Computational proof obligations

The finite certificate has four roles: it verifies the response products and ranks of Proposition 3, the local products of Proposition 4, the complete recurrence graph of Lemma 5, and the integer inequalities and tight edges of Lemma 6. Each finite check is connected above to an induction covering all widths. Appendix A describes the verifier, reconstruction and literal trajectory controls.

7 Consequences

The response catalogue resolves both the linear upper bound and the exact maximum for four rows. It also has an infinite-strip consequence. If an orbit stays forever in a four-row band, each fixed column is updated at most 34 times: every finite prefix fits inside a finite rectangle, to which Proposition 4 applies. Such a confined orbit is therefore transient: every fixed cell is visited only finitely often. This conclusion also holds on an infinite initial colouring.

8 Limitations

The infinite-strip consequence assumes that the orbit stays in the band. A general planar orbit can leave a band and return after modifying its exterior. Extending the method requires interfaces that account for those excursions. For larger fixed heights, a first question is whether repeated entries still admit a finite response catalogue. The four-row proof isolates the closure property and the weighted induction that such a catalogue would need to support. The finite tables and their coverage arguments are verified by exact programs and written inductions. Their translation into a formal proof assistant remains a separate task; the computational trust base is described in Appendix A.

A Computational verification and reproduction

A.1 Certificate architecture and controls

The supplied code has a Python constructor and a separately written C++17 verifier. The latter reconstructs all column products from its own literal turn/flip rule, checks the response ranks, independently builds the expected edge list at every weighted state, checks completeness of the goal set, and recomputes all dynamic values. It then verifies every inequality and tight-edge obligation. A second Python integer DP repeats the affine-vector calculation and the induction check.

The canonical response/graph serialization has SHA256

```
3e01b0422cf2d921fb0ed4aaa690d8f89ffacfa92973a25c3aea590ac8d1110b
```

and the checked A, B, d, e row serialization has SHA256

```
4fb5165094326189694eb227157b578497e089f19eff2abed050b14286c27bca
```

Readable deterministic reconstruction avoids a large opaque table dump. Hashes pin the exact finite claims; the mathematical verification is the local composition, coverage and integer conditions, not the hash itself.

The scoped verification includes the following distinct controls.

- All 1,076 side/parity-specific representatives have their complete future same-side reply trees compared to direct full-colour simulation. Every colouring through width three is checked as well: 17,472 side/parity behaviours.
- Native literal simulation exhausts 4,399,360 colouring/pose inputs through width four, making 20,978,524 updates. It matches the exact maxima but is not the proof for larger widths.

- Twenty-nine reconstructed maximizing seeds, through width 1001, are replayed independently in Python, totaling 89,950 updates. Additional random wide cases and full-plane exterior-marker controls preserve complete fields.
- Altered response data, missing coverage counts, an incorrect recurrence weight and a damaged affine vector are rejected. Resource exhaustion never means a negative answer or an accepted certificate.

The default check runs in a disposable directory and compares a fresh report to a frozen report without overwriting it. Generation is a separate explicit mode. The finite tables and their coverage arguments have not been translated into a formal proof assistant. The trusted base includes the written reductions, the exact verifier source, language implementations and integer execution.

From the repository root run

```
python3 four_row_escape/reproduce.py
```

The only runtime requirements are Python 3.10 or later and a C++17 compiler. The implementation is in `four_row_escape/verification/four_row_boundary/`. The source archive contains the full verifier, the reconstruction code, and the expected report used to check reproduction. These computational sources are also fixed at repository commit `e932522`, four-row project; `sources.json` lists their individual hashes.

An optional witness export is produced by

```
python3 four_row_escape/verification/four_row_boundary/check.py \
  --witnesses /tmp/four-row-maximizers.txt
```

Each row is `n x y h time mask0 ... mask(n-1)`. A mask's bit y is the initial black colour at that row; all other cells are white unless expressly added as exterior controls. The first-exit calculation does not read outside the rectangle. The source also records an attaining 34-update column and an autonomous 21-crossing cut witness.

Notes on sources

For reference [1]: Published conjecture number 11; preprint conjecture number 13.

References

[1] Kossi Roland Etse. How Long Can The Escaping Ant Be Confined? In *51st International Symposium on Mathematical Foundations of Computer Science (MFCS 2026)*, LIPIcs 386, 49:1–49:17, 2026. doi:10.4230/LIPIcs.MFCS.2026.49. Full version: arXiv:2606.26677v1.