

Optimal Local Decompression on Graphs of Odd Maximum Degree

Lennart Rudolph 

October 13, 2026

Abstract

An arbitrary subset of the edges of a communication graph can be stored as short vertex labels, provided a distributed decoder can recover each incident edge’s membership from a bounded neighborhood. Balliu, Brandt, Kuhn, Nowicki, Olivetti, Rotenberg and Suomela asked whether two bits per vertex suffice on cubic graphs. We answer this question affirmatively. More generally, for every fixed odd maximum-degree bound $\Delta \geq 3$, every edge subset of every graph of maximum degree at most Δ has a locally decodable encoding with exactly $(\Delta + 1)/2$ bits per vertex. This uniform integer budget is optimal by counting, and decoding takes $\Delta^{O(\Delta)}$ rounds in the LOCAL model. The construction uses the authors’ one-bit advice scheme for almost-balanced orientations. Separated marker vertices define bounded cells; nonmarker labels encode both a quotient-orientation advice bit and the membership bits of the edges owned by the cell. Reserving one of four labels as a marker leaves three data symbols, whose rate exceeds the cubic edge density. The orientation advice is decoded before ownership, avoiding a circular encoding. The encoder is unrestricted and the result is a locality theorem, with no claim of practical constants.

1 The question and the result

We consider finite simple graphs. The graph is the communication network: vertices learn remote topology through communication, rather than receiving a global description of it. The unknown object is an arbitrary subset $X \subseteq E(G)$. An encoder assigns one fixed-length bit string to each vertex. A deterministic distributed algorithm must then recover, at both endpoints of every edge, whether that edge belongs to X . Its communication radius may depend on the maximum-degree bound, but not on the number of vertices. The encoder may inspect the entire graph, the identifiers and X .

Balliu et al. [1] develop local computation with advice and obtain an encoding using $\lceil d(v)/2 \rceil + 1$ bits at a vertex of degree $d(v)$. Their decoder first reconstructs an almost-balanced orientation and then reads membership bits stored by the tails of the oriented edges. In Section 1.8, question 4, they ask explicitly whether an arbitrary edge subset of a 3-regular graph can instead be encoded using two bits per vertex and decompressed locally. One bit is impossible by counting; three bits are immediate. We prove the optimal two-bit upper bound, together with an odd-degree extension.

Theorem 1. *In the model of Section 2, fix an odd integer $\Delta = 2k - 1 \geq 3$. There exist an encoder and a deterministic LOCAL decoder such that, for every finite simple graph G of maximum degree at most Δ , every admissible identifier assignment, every fixed local port numbering, and every $X \subseteq E(G)$:*

- (i) *the encoder stores exactly k bits at every vertex;*
- (ii) *the decoder recovers the membership of every incident edge at both endpoints;*
- (iii) *the decoder terminates in $\Delta^{O(\Delta)}$ rounds, independently of $|V(G)|$.*

No smaller uniform integer number of bits per vertex works for this graph class, even if the decoder is allowed to inspect the entire graph.

Corollary 2. *Every edge subset of every graph of maximum degree at most three has a two-bit-per-vertex encoding with a constant-round LOCAL decoder. Two is the optimal uniform integer budget for cubic graphs.*

The upper bound uses no regularity assumption. Optimality is a worst-case statement over the class with the specified maximum-degree bound; it does not say that every individual sparse graph needs k bits. The bound is uniform per vertex, whereas the earlier general upper bound is sensitive to each vertex’s degree.

The contribution is a fixed-alphabet packing reduction that attains this exact integer budget. Recognizable markers, clustering and identifier-ordered storage already occur in the advice constructions of Balliu et al. [2, Appendices C.1 and E]. Here the strict rate surplus of the $2^k - 1$ nonmarker symbols pays for a root and one orientation-advice bit per cell. Decoding that bit before interpreting the payload permits almost-balanced ownership on a quotient graph. The orientation theorem is an essential external input; the treatment of virtual graph parameters is part of the reduction. We do not resolve the corresponding exact budget at even maximum degree.

Context and dependencies. The identifier-based LOCAL setting follows the classical study of locality by Linial [3]. Distributed advice has a longer history: Fraigniaud, Gavoille, Ilcinkas and Pelc [4] study advice for fast coloring under a total-advice objective, distinct from the fixed per-vertex budget here. Our separated roots form an elementary ruling set in the standard terminology [5, Definition 2.3]; their selection is centralized encoder work, so no distributed ruling-set algorithm is invoked. The direct predecessor is [1, 2]: it supplies the problem, the orientation method, the preceding upper bound and the counting lower-bound argument. The new step is the packing and ownership reduction above, not a new general method of graph clustering or advice computation.

2 Model and the imported orientation theorem

In the LOCAL model, messages and local computations have no size bound. Vertices communicate synchronously along edges, and round complexity measures communication alone. Initially a vertex knows its identifier, local communication ports and degree, together with the graph order n and exact maximum degree, as in [2, Section 3.1]. Vertices have pairwise distinct identifiers $\text{id}(v) \in [n^c]$, for a known constant integer $c \geq 1$. For our graph class the decoder also uses the fixed bound Δ . Identifiers and physical ports are fixed input data, not storage chosen to encode X . The advice may depend on the identifiers. Correctness is required for the encoder’s labels; on

arbitrary invalid labels the decoder must terminate but may return arbitrary bits. The empty graph is vacuous.

Here $[m] = \{1, \dots, m\}$, and local ports at v are numbered $1, \dots, d(v)$. All graphs are undirected, and $B_R(v)$ denotes the vertices at graph distance at most R from v in its connected component. The decoder is one algorithm for the whole graph class, with the stated public parameters; neither the global topology nor a partition is part of a vertex's initial input.

We use the following published theorem, without strengthening its advice budget or changing its model.

Theorem 3 (Balliu et al., full-version Corollary 15). *For each $D \geq 1$, an almost-balanced orientation of a graph with maximum degree D , supplied its exact order and maximum degree, can be recovered by a deterministic LOCAL algorithm from one advice bit per vertex in $T(D) = D^{O(1)}$ rounds. Almost-balanced means*

$$|d^+(v) - d^-(v)| \leq 1 \quad \text{for every vertex } v.$$

Here we fix any permitted constant sparsity parameter in the uniform fixed-length sparse schema of [2, Appendix B, Corollary 15]. We use its existence and one-bit budget, not its sparsity. Once Δ and hence D are fixed, we fix this decoder and a time bound $T(D)$ before choosing the physical graph, its identifiers or X . The proof in that source is an essential mathematical dependency of the present theorem.

Total behavior on arbitrary advice. We may take the imported decoder's per-round functions to be total, without requiring a correctness guarantee for invalid advice. To see this even for an implementation that may fail to finish a local computation on bad advice, fix its public parameters $N, D, c, T(D)$. There are finitely many admissible port-numbered graphs with N vertices, maximum degree exactly D and identifiers in $[N^c]$. Fairly interleave simulations of every such graph with every one-bit advice assignment until each graph has a completed transcript of at most $T(D)$ rounds giving an almost-balanced orientation. This search terminates by Theorem 3.

Record initialization, message emission, state updates and outputs from the selected transcripts, keyed by their complete local configurations, including round and public inputs. Determinism makes repeated keys consistent. On any unrecorded configuration, use a fixed finite default. Replay preserves each selected transcript by induction, so every graph still has good one-bit advice, with the same round bound. The finite table is computed uniformly from the public parameters and the fixed algorithm, independently of the actual input graph. Its potentially enormous local computation costs no communication rounds. We use this totalized decoder below, padding halted states by idle states. This normalization is only an existence argument; it gives no efficient implementation or sparse-advice claim beyond what is used here.

We will apply Theorem 3 to a virtual graph whose vertices represent cells. The virtual graph can have a different order and maximum degree from the physical graph. Section 4.1 specifies a padding construction so that the decoder can supply the exact parameters required by the imported algorithm. No assumption that this algorithm ignores the graph order is needed.

3 Marker cells and their storage capacity

Fix $k \geq 2$, put $\Delta = 2k - 1$, and use the constants

$$q = 2^k - 1, \quad r = 16(k + 1), \quad M = (\Delta + 1)^{2r}, \quad D = \Delta M. \quad (1)$$

All are independent of the graph order. A vertex label is one of the 2^k symbols $0, 1, \dots, q$. Symbol 0 is reserved for a cell root in every component with more than r vertices. The remaining symbols represent base- q digits by subtracting one.

Lemma 4 (Capacity). *For every integer $s \geq r + 1$,*

$$q^{s-1} \geq 2^{1+L_s}, \quad L_s = \left\lceil \frac{\Delta s}{2} \right\rceil. \quad (2)$$

Thus the nonroot vertices of a cell of size s can hold one advice bit and L_s payload bits.

Proof. Since $k \geq 2$,

$$\frac{q^2}{2^\Delta} = 2(1 - 2^{-k})^2 \geq \frac{9}{8}, \quad \left(\frac{9}{8}\right)^8 > 2.$$

Write $s = 2t$ or $s = 2t + 1$. The lower bound on s implies $t \geq 8(k + 1)$, so

$$q^{2t} \geq 2^{\Delta t} \left(\frac{9}{8}\right)^t > 2^{\Delta t + k + 1}.$$

For odd s , this is precisely the required bound, because $L_s = \Delta t + k$. For even s , divide by $q < 2^k$ to obtain $q^{s-1} > 2^{\Delta t + 1} = 2^{L_s + 1}$. $\square$

For cubic graphs, each vertex has four labels: one marks a root and the other three store ternary digits. There are approximately 1.585 data bits per nonroot vertex, exceeding the 1.5 edges per vertex of a cubic graph. A sufficiently large cell absorbs the cost of its root and its one orientation-advice bit. This strict rate surplus is why reserving a marker does not lose the optimal integer budget.

Lemma 5 (Locally recognizable cells). *In each connected component C with $|C| > r$, one can choose roots and a partition into cells C_ρ , indexed by the roots, with the following properties:*

- (i) *distinct roots have distance greater than $2r$;*
- (ii) *each vertex belongs to its nearest root, with ties broken by root identifier;*
- (iii) *each cell is connected, has radius at most $2r$ around its root, and has size between $r + 1$ and M ;*
- (iv) *the root markers and a bounded neighborhood determine the cells, their internal edges, and their boundary edges with the root of the cell at the other endpoint.*

Proof. Choose a maximal set of vertices at pairwise distance greater than $2r$, for example greedily in increasing identifier order. This is an encoder operation; it is not computed by the decoder without advice. Maximality makes every vertex lie within distance $2r$ of a root. Assign it to the root minimizing the pair consisting of distance and identifier.

If v is assigned to ρ , every predecessor u on a shortest ρ - v path is also assigned to ρ . Indeed, a root that beat ρ at u would, by the triangle inequality, beat it at v , either by distance or by the same identifier tie-break. The cells are therefore connected and have the stated radius.

Every vertex at distance at most r from ρ is strictly closer to ρ than to any other root: separation gives distance at least $2r + 1 - t > t$ to the latter when its distance to ρ is $t \leq r$. Hence

$$B_r(\rho) \subseteq C_\rho \subseteq B_{2r}(\rho).$$

In a component with more than r vertices, every radius- r ball contains at least $r + 1$ vertices. Either a shortest path of length r exists from its center, or the ball contains the entire component. Finally,

$$|B_{2r}(\rho)| \leq \sum_{j=0}^{2r} \Delta^j \leq (\Delta + 1)^{2r} = M.$$

A vertex needs only the markers within radius $2r$ to find its root. The root ρ can determine the cell assignment of every vertex in $B_{2r}(\rho)$ and of each of its neighbors by examining radius $4r + 2$. Indeed, a path of length at most $2r$ starting within distance $2r + 1$ of ρ lies entirely in $B_{4r+1}(\rho)$. The extra round supplies incident-edge and port information. Thus this view determines the same relevant distances as the whole graph, the root's entire cell, and all its boundary edges with the root identifier at the other endpoint. No unavailable remote topology is used. $\square$

Small components are recognized before interpreting any label as a marker. For any vertex v ,

$$|\text{component}(v)| \leq r \iff |B_r(v)| \leq r. \quad (3)$$

The reverse implication follows from the same ball-size observation. In this case the whole component is visible within radius r . Consequently arbitrary zero labels used in small components cannot be confused with large-component root markers.

4 Balanced ownership on the quotient

Consider the cells of all large components. Internal edges have both endpoints in one cell. Boundary edges join two different cells. Contracting the cells and discarding internal loops gives a multigraph; the number of parallel edges between two roots is visible to both roots.

Order original edges lexicographically by the pair consisting of their smaller and larger endpoint identifiers. Both roots see the same complete list $e_0, \dots, e_{m-1}$ of edges between their cells. Put $p = \lfloor m/2 \rfloor$ and pair e_j with e_{j+p} for $0 \leq j < p$, leaving e_{2p} unpaired when m is odd. For each pair, orient its first member from the smaller root identifier to the larger, and its second member in reverse. The unpaired edges form a simple graph H_0 on the roots. Its maximum degree is at most $D = \Delta M$: a cell has at most ΔM boundary edges, and hence at most that many residual neighbors. Apply Theorem 3 with the parameter padding below to orient the edges of H_0 almost-balanced. This orients all boundary edges without requiring an orientation theorem for multigraphs.

Lemma 6 (Ownership bound). *Assign each internal edge to its cell and each boundary edge to its tail cell. Every edge has exactly one owner. A cell of size s owns at most $L_s = \lceil \Delta s/2 \rceil$ edges.*

Proof. Let e and b be the numbers of internal and boundary edges of a cell. If it has p paired boundary-edge pairs and d unpaired edges, then $b = 2p + d$. The pairs contribute exactly p outgoing edges. Almost-balance of the residual simple quotient bounds its outgoing unpaired edges by $\lceil d/2 \rceil$. The total number owned is therefore at most

$$e + p + \lceil d/2 \rceil = e + \lceil b/2 \rceil = \left\lceil \frac{2e + b}{2} \right\rceil \leq \left\lceil \frac{\Delta s}{2} \right\rceil.$$

The last inequality is the degree-sum identity within the cell, counting each internal edge twice and each boundary edge once. The ownership rule makes the assignment unique. $\square$

4.1 Supplying the virtual algorithm's parameters

The vertices of H_0 need not know its total order or its exact maximum degree. We avoid assuming these quantities are locally computable.

Form a virtual simple graph H on all n original vertex identifiers, putting precisely the edges of H_0 between root vertices and leaving every other original vertex isolated. Add a disjoint clique on $D + 1$ fresh virtual vertices. The resulting graph has known order

$$N = n + D + 1$$

and exact maximum degree D , even if H_0 has no edges. Assign the dummy clique identifiers $n^c + 1, \dots, n^c + D + 1$. They are distinct from all original identifiers and at most N^c , since $c \geq 1$ is an integer. At each virtual vertex, number ports by increasing neighbor identifier. Use this same convention in the encoder and in the simulation. The resulting disconnected graph is a valid input to the orientation advice encoder.

Only its advice bits at roots are needed. The other original vertices are isolated in H , and the dummy clique is in a disjoint component, so neither their messages nor their advice can affect a root. A physical root simulates its vertex of H with its own identifier, its residual-quotient degree, the known values N, D , and its decoded advice bit. This gives exactly the same output as the imported algorithm on H . The padding is conceptual: no dummy labels are stored in the physical graph.

5 Encoding and decoding

5.1 The encoder

In a component of size $s \leq r$, sort vertices and edges by identifiers. Its m edge-membership bits fit in ks bits because

$$m \leq \lfloor \Delta s / 2 \rfloor \leq ks.$$

Pad with zeros and distribute the resulting string as k bits at each vertex in vertex order. No marker restriction applies in these components.

Choose the roots and cells of Lemma 5 in each larger component. Form the single padded quotient H from the cells of all these components, as in Section 4.1, and choose one good advice assignment on H . Specifically, take the first assignment in identifier-ordered lexicographic order whose simulation by the fixed, totalized $T(D)$ -round decoder produces an almost-balanced orientation. This finite search succeeds by Theorem 3 and its normalization; efficiency is not required. The roots, advice and decoded orientation are therefore fixed by the graph and identifiers before X is encoded.

Denote the bit at root ρ by $a_\rho \in \{0, 1\}$. The resulting orientation determines each cell's owned edges as in Lemma 6. Sort those edges by their endpoint identifiers, record their membership in X , and append zeros to obtain a payload of exactly L_s bits for a cell of size s .

Use little-endian positional order: the first payload bit has coefficient one, and subsequent bits have coefficients $2, 4, \dots$. Interpret the payload as an integer $z_\rho \in [0, 2^{L_s})$ and form

$$w_\rho = 2z_\rho + a_\rho. \tag{4}$$

By Lemma 4, this integer has a base- q representation with exactly $s - 1$ digits after padding the high positions with zeros. Order the nonroot vertices by identifier, store the digits from least to most significant using labels $1, \dots, q$, and give the root label 0. Every physical vertex receives one of 2^k symbols, hence exactly k bits.

5.2 The decoder

The decoder performs the following operations, with all orders fixed by identifiers.

1. *Recognize small components.* Use (3). A vertex in a small component gathers its labels and topology, concatenates the k -bit labels in vertex order, and reads the component's edge bits in edge order.
2. *Recover the cells.* In every other component, use precisely the zero labels as roots. Assign a vertex to its nearest zero within distance $2r$, with identifier ties; leave it unassigned if none is visible. On encoded inputs this recovers exactly the cells of Lemma 5. Read their internal and boundary edge lists from the stated local views.
3. *Read the packet.* Each root lists its nonroot vertices as $v_0, \dots, v_{m-1}$ in increasing identifier order. Writing $\ell(v)$ for the received label, set

$$w_\rho = \sum_{i=0}^{m-1} (\ell(v_i) - 1)q^i, \quad a_\rho = w_\rho \bmod 2, \quad z_\rho = \lfloor w_\rho/2 \rfloor.$$

The empty sum is 0. Selected nonroot labels are nonzero by Lemma 8, so each $\ell(v_i) - 1$ is a valid base- q digit. The bit a_ρ is the orientation advice; no edge orientation or payload ordering is needed to obtain it.

4. *Orient the quotient.* Pair parallel boundary edges canonically and simulate the imported orientation algorithm on the residual simple quotient, with the virtual parameters from Section 4.1.
5. *Deliver the data.* Each root now knows its owned edge list. Interpret z_ρ as the padded L_s -bit payload and assign its first bits to that list. Deliver each membership bit to both endpoints of its edge.

Lemma 7 (Correctness). *For every $X \subseteq E(G)$, the decoder applied to its encoded labels returns its membership bits correctly at both endpoints of every edge.*

Proof. The topology determines the correct small-component branch, where raw bit packing is injective. In a large component, nonroot labels are nonzero, so marker recognition recovers exactly the encoder's roots. The distance and identifier rule therefore recovers the same cells and digit positions.

Uniqueness of base- q expansion recovers w_ρ . Euclidean division by two in (4) recovers exactly a_ρ and z_ρ . Hence the virtual decoder receives the same advice, identifiers, topology and global parameters as the encoder's simulation, and produces the same orientation. Its canonical owned edge lists are therefore the encoder's lists. Reading the recovered payload supplies exactly the stored bits. Every edge has a unique owner, so both endpoints receive its correct membership bit. $\square$

The order of these operations is essential: cells determine packet positions; packets determine advice; advice determines ownership; ownership determines the meaning of the payload. The payload is never needed to decode its own ownership rule.

Lemma 8 (Bounded cells on arbitrary labels). *In a large component, apply the decoder's radius- $2r$ root rule to any labeling. Every zero marker owns itself. Every nonempty cell is connected, has radius at most $2r$ and size at most M . These conclusions require neither marker separation nor coverage.*

Proof. For a zero marker ρ , its cell is precisely its identifier-tiebroken Voronoi cell among all zero markers, intersected with $B_{2r}(\rho)$. A selected root remains selected along every shortest path from that root to an assigned vertex: a competing zero that beat it at a predecessor would also be within $2r$ of the original vertex and beat it there. Thus the whole shortest path remains in the bounded cell. A marker owns itself since it is its unique distance-zero marker. The size bound follows from containment in $B_{2r}(\rho)$ and the ball bound in Lemma 5. $\square$

The assigned or unassigned status of a cell's vertices and their neighbors is determined by radius- $2r$ views contained in the root's radius- $4r + 2$ view. On arbitrary labels, form the residual quotient only from edges joining distinct assigned cells, using the same parallel-edge pairing. Each cell has at most M vertices, so the residual degree is at most $\Delta M = D$. The padding in Section 4.1 still gives order $N = n + D + 1$ and exact maximum degree D . Edges touching unassigned vertices receive the default bit 0. Small-component zeros remain ordinary data. Cells smaller than $r + 1$ may fail the capacity inequality, but their packet reader remains defined.

Lemma 9 (Locality). *The decoder can be implemented in at most*

$$6r + 3 + (4r + 1)T(D)$$

LOCAL rounds, where r, D are the fixed constants in (1) and T is the round bound in Theorem 3.

Proof. First flood identifiers and labels for $4r + 2$ rounds. A first-round exchange of identifiers and port numbers also lets every vertex learn its incident edges; flooding these records supplies complete incident-edge and port information for vertices within distance $4r + 1$. This is the view needed in Lemma 5: each root knows its cell, packet, boundary-edge list and residual neighbors. Each assigned nonroot can choose a parent on a shortest path to its root, breaking ties by identifier. The shortest-path property keeps these parent paths inside the cell. A root also knows source routes through its own cell, so no further tree-setup exchange is required.

A residual edge has a canonical representative boundary edge. Route a virtual message from its sender root down a source route to that boundary edge, across it, and then up the receiver's parent path. The route has length at most $2r + 1 + 2r = 4r + 1$. Tag messages by their virtual round, sender and receiver root identifiers, and representative edge. At the destination, the sender identifier determines the receiving virtual port from the destination's sorted residual-neighbor list. The sender need not know the other cell's topology or port numbering. All routes operate simultaneously: unbounded LOCAL messages carry all tags and payloads, so there is no congestion factor.

Initialize each root with its decoded advice and the exact virtual initial data specified in Section 4.1. We claim that its state after t blocks of $4r + 1$ physical rounds equals its state after t rounds of the fixed imported algorithm on H . The case $t = 0$ is initialization. For the induction step, equal states produce equal outgoing messages; routing delivers each message to the correct virtual port before the block ends; the same transition rule then gives equal next states. Early messages wait until the block boundary, and halted states are extended by idle states. This proves the claim through $T(D)$ blocks. All instructions are fixed independently of G and X ; only the stated public inputs and gathered local views vary.

Finally, a root broadcasts its owned-edge bits through its cell in $2r$ rounds. One additional exchange along every edge of a large component reaches the other endpoints and reconciles their candidate bits as specified below. Small components finish within the initialization phase. The total is

$$(4r + 2) + (4r + 1)T(D) + (2r + 1) = 6r + 3 + (4r + 1)T(D).$$

This is a fixed schedule on every label assignment. Use the totalized decoder from Section 2 and stop after exactly $T(D)$ virtual rounds. Lemma 8 keeps all routes inside cells of radius at most $2r$; discovery never exceeds the stated radii. Packet values, indices and message tags are range checked. Invalid payloads use fixed defaults. Endpoints exchange their candidate bits in the final round. They output the common value if the combined candidate set is nonempty and all candidates agree, and 0 if that set is empty or conflicting. Thus the single candidate supplied by a valid edge’s owner is accepted. No test of the actual graph’s global marker separation, coverage or advice validity is performed. Termination follows from the fixed schedule; correctness is required only on the encoder’s labels. $\square$

Proof of Theorem 1. The construction uses exactly k bits per vertex. Correctness and locality follow from Lemmas 7 and 9. Since $r = O(k) = O(\Delta)$, $D = \Delta(\Delta + 1)^{2r}$ and $T(D) = D^{O(1)}$, the round bound is $\Delta^{O(\Delta)}$.

For optimality, we repeat the information-counting argument underlying [1, Section 1.4.1], specialized to the uniform integer budget. Fix $G = K_{\Delta+1}$, its identifiers and its ports. Writing $n = \Delta + 1$, the graph has $\Delta n/2$ edges and therefore $2^{\Delta n/2}$ possible subsets. A uniform fixed budget of $k-1$ bits at every vertex gives at most $2^{(k-1)n}$ distinct inputs to any fixed decoder. Since $k-1 = (\Delta-1)/2 < \Delta/2$, the number of label assignments is too small. Two different subsets would receive the same labels on the same fixed graph, making correct decoding of both impossible, even without any locality restriction. $\square$

6 Discussion

The result closes the fixed integer label-budget gap in the published cubic question and, more generally, at every fixed odd maximum-degree bound. It does not optimize the decoding radius or produce a practical compressed graph representation: it encodes edge membership on an existing communication network, the encoder is unrestricted, and the constants inherited from cell formation and the orientation theorem are large. An efficient centralized encoder or a useful small-radius implementation is a separate question.

At even maximum degree $\Delta = 2k$, reserving one of 2^k symbols leaves rate $\log_2(2^k - 1) < \Delta/2$. Thus this construction does not reach the counting bound of k bits. Applying the next odd-degree bound gives $k+1$ bits, matching the earlier uniform upper bound. Whether the exact even-degree budget can be attained remains outside this paper.

The underlying packing principle is more general: a marker alphabet with B symbols has $B-1$ data symbols, so sufficiently large bounded cells absorb constant metadata whenever $B-1 > 2^{\Delta/2}$. Almost-balanced ownership converts the edge density into the payload bound. Here the one-bit quotient-orientation schema makes the metadata constant per cell.

7 Formalization and reproducibility

The companion formalizes the finite reduction used in the proof, with Lean 4.35.0-rc2 and pinned Mathlib dependencies. Its principal graph-native result is `graph_marker_reconstruction` in `LocalDecompression.Reduction`. It constructs vertex labels whose zero symbols recover the roots and whose payload recovers every edge-bit assignment. It assumes a connected large component, separated roots, the degree bound and good advice for a supplied residual-orientation decoder. The orientation family and advice are fixed before the encoder, which works for every

edge-bit assignment. The same simple graph supplies the metric, edges and degrees; no premise assumes recovery of the unknown bits.

Table 1 identifies the main correspondence. Declaration names are relative to the namespace `LocalDecompression`. Formal sorting uses supplied linear orders, independent of the numerical identifiers used to break Voronoi ties. The written algorithm uses identifier-induced orders throughout.

Paper claim	Lean result and scope
Cell capacity	<code>odd_degree_capacity</code> : the odd-degree alphabet inequality.
Partition geometry	<code>exists_large_component_partition</code> , <code>component_has_large_partition</code> and <code>small_component_recognition</code> : separated cells and the small-component test.
Quotient ownership	<code>QuotientOrientation.lifted_balanced_of_simpleGraph</code> and <code>Reduction.ownedEdges_card_bound</code> : actual parallel-edge pairing, residual quotient and payload bound.
Marked reconstruction	<code>Reduction.graph_marker_reconstruction</code> : vertex-label roundtrip and exact recovery of the marker set.
Payload locality	<code>Reduction.graph_marker_decoder_local</code> and its <code>_right</code> counterpart: edge-bit lookup at both endpoints on a fixed graph and covered marker set, assuming residual-oracle locality.
Changing markers	<code>MarkerLocality.cell_data_four_r</code> : cell-site and incident-edge/endpoint-owner records under coverage; <code>MarkerLocality.marker_packet_reader_eq</code> : equality of the actual ordered packet reader across different marker sets.
Arbitrary labels	<code>BoundedCells.boundedCell_induced_path</code> and <code>BoundedPacket.boundedAdvice_local</code> : bounded-cell geometry and ordered packet/advice locality without coverage.
Virtual parameters	<code>Padding.padded_input_parameters</code> : the root embedding and disjoint clique have the exact order, degree and admissible identifiers.
Counting optimum	<code>completeGraph_lower_bound</code> : the uniform integer budget cannot be reduced.

Table 1: Correspondence between the written proof and its Lean companion. Each declaration retains its explicit hypotheses; the table does not assert an unconditional formal LOCAL theorem.

The orientation input requires only the sufficient outdegree bound $2d^+ \leq d + 1$, which follows from almost balance but is not equivalent to it. The quotient lift proves the boundary bound from that input. The published orientation theorem is not introduced as a Lean axiom. Across-marker comparisons fix the physical graph; the existing global packet reader is compared under marker coverage for both labelings.

The modules `BoundedCells` and `BoundedPacket` treat arbitrary labels on a connected physical component. They prove Lemma 8, the degree-only cell-size bound and stability of complete bounded cells and storage sites. The bounded reader uses the existing codec’s increasing site enumeration and digit offset. With selection radius R , full-label agreement within radius $2R$ preserves its ordered word, packed value and extracted advice, without coverage or separation. Selected nonroot labels are nonzero; empty packets, absent markers and singleton cells give value and advice zero. Under marker coverage the reader agrees with the existing `wordsOfLabels`; on the encoder’s labels it recovers the chosen advice bit. These bridges check the ordering and advice-first dataflow, but do not extend the assembled `markerDecode` locality theorem to arbitrary labels.

The predicate `LocalAt` expresses label dependence on a fixed topology, not a uniform operational LOCAL algorithm. The invocation of the published orientation algorithm across varying quotient graphs, its message-level simulation and totalization, and the assembled execution on mixed connected components remain written arguments. Thus the companion is a partial mechanization of Theorem 1. Its extensional locality estimates and the execution schedule in Lemma 9 are separate bounds. Independent human review is not claimed.

The archive contains production Lean sources, pinned build/submission metadata, licences and one Python script. Its seven-goal Palomar-style interface includes the finite theorem `PalomarDecompression.adviceFirstGraphReconstruction`; the graph-native results in Table 1 are also included in the checked project. The standalone `Challenge` module has deliberate statement placeholders and is excluded from the completed proof closure. Local comparisons and metadata validation do not establish Palomar service acceptance.

Production sources were rebuilt in a fresh project using the pinned dependency artifacts. The audit includes instances and generated logical declarations, checks their full types and transitive axioms, and binds the delivered source bytes to fresh kernel replay and exported-proof checks with Lean, NanoDa and Con-Ron. These checks support the stated formal propositions, separately from statement fidelity. Only `propext`, `Quot.sound` and `Classical.choice` occur as axioms in the completed proof closure.

Install the toolchain named in `lean-toolchain`; a clean installation needs network access to fetch the pinned dependencies and cache. From the archive's `palomar/` directory, run:

```
lake exe cache get
lake build LocalDecompression Solution
lake env leanchecker --fresh Solution
```

The optional command `lake build Challenge` builds the submission goals and reports their seven intentional placeholders. The pinned manifest should be retained. From the archive root, run:

```
python3 scripts/check_invariants.py
```

This script uses only the Python 3.9 standard library or later. It checks 160 packets and 16 graph and identifier fixtures, including disconnected and small components. These are finite tests; the script does not implement the imported orientation algorithm.

Acknowledgments and responsibility

This result relies on the advice framework and almost-balanced orientation theorem of Balliu, Brandt, Kuhn, Nowicki, Olivetti, Rotenberg and Suomela, as well as the earlier distributed-computing work cited above. The formal development relies on Lean and the Mathlib contributors. OpenAI ChatGPT/Codex assisted extensively with research, proof development, formalization and writing. The author retains responsibility for the claims and attribution. Independent human peer review and endorsement by the authors of the cited work are not claimed.

References

- [1] A. Balliu, S. Brandt, F. Kuhn, K. Nowicki, D. Olivetti, E. Rotenberg and J. Suomela. Distributed computation with local advice. In *39th International Symposium on Distributed Computing (DISC 2025)*, LIPIcs 356, 12:1–12:19, 2025. [doi:10.4230/LIPIcs.DISC.2025.12](https://doi.org/10.4230/LIPIcs.DISC.2025.12). The target question is Section 1.8, item 4.
- [2] A. Balliu, S. Brandt, F. Kuhn, K. Nowicki, D. Olivetti, E. Rotenberg and J. Suomela. Distributed computation with local advice, full version. [arXiv:2405.04519v2](https://arxiv.org/abs/2405.04519v2), August 22, 2025. Appendix B, Corollary 15 supplies the one-bit almost-balanced orientation schema.
- [3] N. Linial. Locality in distributed graph algorithms. *SIAM Journal on Computing* 21(1):193–201, 1992. [doi:10.1137/0221015](https://doi.org/10.1137/0221015).
- [4] P. Fraigniaud, C. Gavoille, D. Ilcinkas and A. Pelc. Distributed computing with advice: information sensitivity of graph coloring. *Distributed Computing* 21(6):395–403, 2009. [doi:10.1007/s00446-008-0076-y](https://doi.org/10.1007/s00446-008-0076-y).
- [5] L. Barenboim, M. Elkin, S. Pettie and J. Schneider. The locality of distributed symmetry breaking. *Journal of the ACM* 63(3), Article 20, 20:1–20:45, 2016. [doi:10.1145/2903137](https://doi.org/10.1145/2903137).