Noname manuscript No.
(will be inserted by the editor)

Bounding the Locality of Distributed Routing Algorithms

Prosenjit Bose · Paz Carmi · Stephane Durocher

Received: date / Accepted: date

Abstract We examine bounds on the locality of routing. A local routing algorithm makes a sequence of distributed forwarding decisions, each of which is made using only local information. Specifically, in addition to knowing the node for which a message is destined, an intermediate node might also know 1) its local neighbourhood (the subgraph corresponding to all network nodes within k hops of itself, for some fixed k), 2) the node from which the message originated, and 3) the incoming port (which of its neighbours last forwarded the message). Our objective is to determine, as k varies, which of these parameters are necessary and/or sufficient to permit local routing on a network modelled by a connected undirected graph. In particular, we establish tight bounds on k for the feasibility of deterministic k-local routing for various combinations of these parameters, as well as corresponding bounds on dilation (the worst-case ratio of actual route length to shortest path length).

Keywords distributed algorithms · local routing · dilation

1 Introduction

1.1 Local Routing

Unicast communication in a network is achieved by a routing algorithm that computes a sequence of forwarding decisions that determine the route followed by a message (e.g., a packet) as it travels to its destination. Traditionally, routing tables are constructed as a function of the network topology to provide efficient routing, where dilation decreases as memory and table sizes increase (e.g., see [14–17, 24, 25]). In many networks, implementing centralized routing algorithms or, more generally, routing algorithms whose initialization requires knowledge of the entire network topology is impractical; reasons include that the network is too large, that the topology of the entire graph is unknown, or that the network changes dynamically [26], such as in an ad hoc wireless network, where each node can periodically acquire and update information about its neighbourhood, but not necessarily about distant nodes in the network. Alternatively, a *local* routing algorithm makes a series of distributed forwarding decisions, computed at each of the intermediate nodes along the route; when a node receives a message, it selects a port (i.e., one of its neighbours) to which to forward the message using only local information. Specifically, each node is only aware of the subset of the network consisting of nodes within k hops from itself, for some k. Consequently, the route cannot be precomputed entirely in general.

Some of these results appeared in preliminary form at the 28th ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing (PODC 2009) [2].

This research was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC).

P. Bose
School of Computer Science, Carleton University
Ottawa, Canada
E-mail: jit@scs.carleton.ca

P. Carmi
Department of Computer Science
Ben-Gurion University of the Negev
Beer-Sheva, Israel
E-mail: carmip@cs.bgu.ac.il

S. Durocher
Department of Computer Science, University of Manitoba
Winnipeg, Canada
E-mail: durocher@cs.umanitoba.ca

Furthermore, message overhead and local memory are often limited [21]. In particular, a network node cannot be expected to maintain a history of messages that have passed through it (i.e., the network is *memoryless*). Similarly, the message overhead cannot store the set of nodes visited by the message (i.e., the routing algorithm is *stateless*).

Although a straightforward flooding algorithm is possible, such a strategy has obvious drawbacks, including high traffic loads [26], cyclic behaviour (if the network is memoryless), and requiring knowledge of an upper bound on the diameter of the network to ensure both termination and successful delivery. In this paper we consider single-path deterministic routing algorithms.

We represent a network by a connected, unweighted, undirected, simple graph G with unique vertex labels. A network node (graph vertex) is identified by its label. In some networks, a node's label may provide information about its neighbourhood in the network (e.g., a grid graph node can be labelled by its grid coordinates). In general, we suppose that the vertex labelling is independent of the graph; that is, a node's label does not encode additional information about the topology of the graph or the node's neighbourhood. Equivalently, we consider routing algorithms that succeed on any (possibly adversarial) permutation of the vertex labels of G. We assume that every node knows its own label as well as the labels of its neighbours. A message also requires a destination node, identified by that node's label. Some or all of the following additional information may be available to an intermediate node u to compute the next node to which a message should be forwarded:

1. **origin-awareness**: knowledge of the node from which the message originated,
2. **predecessor-awareness**: knowledge of the incoming edge (port) along which the message was forwarded to u (equivalently, the neighbour of u that last forwarded the message), and
3. ***k*-locality**: knowledge of the k-neighbourhood of u (i.e., the subgraph of G consisting of all paths rooted at u with length at most k).

Our objective is to determine which of these parameters are necessary and/or sufficient to permit local routing as k varies.

1.2 Overview of Results

We identify tight bounds on the value of the locality parameter k for the feasibility of k-local routing in each of the four combinations of constraints: predecessor-aware or predecessor-oblivious, and origin-aware or origin-oblivious. In each case, let $T(n)$ denote the corresponding threshold. That is, for every $k < T(n)$, every k-local routing algorithm is defeated by some connected graph on n vertices. Similarly, for every $k \geq T(n)$, there exists a k-local routing algorithm that succeeds on all connected graphs on n vertices. Our main result is the identification of the values of $T(n)$; see Table 1. In addition, we establish a lower bound of $S(k) = 2n/k - 3$ on the worst-case dilation of any k-local routing algorithm and show this bound is tight for three of the four combinations of constraints; see Table 2.

$T(n)$	origin-aware	origin-oblivious
predecessor-aware	$n/4$	$n/3$
predecessor-oblivious	$n/2$	$n/2$

Table 1 **Main result:** there exists a k-local routing algorithm when $k \geq T(n)$, but no k-local routing algorithm exists when $k < T(n)$, where n denotes the number of network nodes. Rounding operators are omitted; see Theorems 1 through 3, Corollaries 2 and 5, and Theorems 5 through 8 for exact values of $T(n)$.

k	$n/4$	$n/3$	$n/2$
lower bound	5	3	1
upper bound	6	3	1

Table 2 Bounds on the dilation attainable by a k-local routing algorithm. $S(k) \geq 2n/k - 3$ is a lower bound on the worst-case dilation of any k-local routing algorithm. The bound on $S(k)$ is tight when $k \in \{n/3, n/2\}$ and is bounded from above by 6 when $k = n/4$. See Theorems 4, 7, 8, and 6.

Thus, our objective is to identify bounds on the feasibility of guaranteed delivery for memoryless, stateless, deterministic local routing. Specifically, we answer the question of exactly how much of the network each node must be aware for local routing to succeed. Local routing succeeds in particular settings, e.g., on some classes of geometric graphs, such as unit disc graphs, planar graphs, and triangulations (see Section 3). In this paper, we seek to determine whether similar local routing algorithms are possible beyond these restricted classes of graphs. As we show, guaranteeing delivery using local routing in arbitrary connected graphs requires nodes to have knowledge of a neighbourhood of size $\Omega(n)$ nodes in some cases, suggesting that local information (small k) is insufficient and that the routing algorithm must be modified or extended using a different paradigm, such as centralized routing (routing tables), randomization, increased memory (passed with the message or stored at each node), or leveraging specific additional graph properties (e.g., a geometric embedding).

2 Modelling Local Routing

In this section we formalize our model for local routing.

2.1 k-Local Routing Functions

Given a (simple undirected) graph G, we employ standard graph-theoretic notation, where $V(G)$ denotes the vertex set of G; $E(G)$ denotes the edge set of G; for each vertex $v \in V(G)$, $\operatorname{Adj}(v) = \{u \mid \{u, v\} \in E(G)\}$ denotes the set of vertices adjacent to v; and $\deg(v) = |\operatorname{Adj}(v)|$ denotes its degree. Let $\operatorname{dist}(u, v)$ denote the (unweighted) graph distance between vertices u and v, i.e., the number of edges in a shortest path from u to v. The *girth* of G is the length of its shortest cycle or ∞ if G is acyclic.

The *k-neighbourhood* of a vertex $u \in V(G)$, denoted $G_k(u)$, is the subgraph of G that contains all paths rooted at u with length at most k. A routing algorithm is *origin-aware*, *predecessor-aware*, and *k-local* if it can be defined as a function $f(s, t, u, v, G_k(u))$, where

– $s \in V(G)$ is the origin node,
– $t \in V(G)$ is the destination node,
– the message is currently at node $u \in V(G)$ (we say u is the *current* node),
– node u received the message from its neighbour $v \in \operatorname{Adj}(u)$ (let $v = \perp$ before s forwards the message for the first time),
– $G_k(u)$ is the k-neighbourhood of u, and
– $f(s, t, u, v, G_k(u))$ returns the neighbour of u to which the message should be forwarded (i.e., the port to which node u must forward the packet).

Every k-local routing algorithm $\mathcal{A}$ has a corresponding routing function f. A sequence of calls to function f returns a sequence of forwarding decisions that corresponds to a walk, i.e., the *route*, through G originating at s.

We consider two constraints on k-local routing algorithms: an *origin-oblivious* k-local routing algorithm is not provided the parameter s, and a *predecessor-oblivious* k-local routing algorithm is not provided the parameter v.

To simplify notation for predecessor-aware algorithms, let $f_u(v)$ denote the *local routing function* at node u for a given s, t, u, and $G_k(u)$, where $f_u(v) = f(s, t, u, v, G_k(u))$. That is, $f_u(v)$ returns the neighbour of u to which the message is forwarded as a function of the neighbour v from which it is received.

Observation 1 *Given any $k \geq 1$, any predecessor-aware k-local routing algorithm $\mathcal{A}$, any connected graph G, and any $\{s, t\} \subseteq V(G)$, the direction in which a message traverses a given edge uniquely determines the next forwarding decision by $\mathcal{A}$. In particular, if $\mathcal{A}$ successfully delivers a message to t, then the message has traversed each edge in $E(G)$ at most once in each direction.*

Naturally, not all routing functions can be implemented efficiently as local routing algorithms. The routing function model allows stronger negative results to be established for a more general class of routing algorithms, regardless of implementation concerns. With respect to positive results, the routing algorithms we present can be implemented efficiently locally; implementation details are not the focus of this paper.

Let C denote a connected component of $G_k(u) \setminus \{u\}$. We refer to C as a *local component* of u. If $v \in V(C) \cap \operatorname{Adj}(u)$, then we say C is rooted at v (C can have multiple roots). If C contains a vertex z such that $\operatorname{dist}(u, z) = k$, then (relative to u) C is an *active component*, edge $\{u, v\}$ is an *active edge* (where $v \in V(C)$), v is an *active neighbour* of u, and a shortest path from u to z is an *active path*. In other words, C extends to the limit of u's knowledge: node z may have neighbours outside C, but this information is not known locally at u. If C is an active component of u and every active path in C passes through some vertex $w \neq u$, then C is a *constrained active component* and w is a *constraint vertex*. If a local component C is not an active component, then we say C is a *passive component*. If u is connected to its local component C by a single edge (i.e., C has a unique root), then we say C is an *independent component*. Every independent active component is a constrained active component. See Figure 1.

A cycle C is a *local cycle* at node u if u lies on C and C has length at most $2k$. That is, $C \subseteq G_k(u)$, $|V(C)| \leq 2k$, and $u \in V(C)$.

2.2 Evaluating Routing Algorithms

A routing algorithm $\mathcal{A}$ defined by a routing function f *succeeds* (synonymously, *guarantees delivery*) if for all graphs G and all origin-destination pairs (s, t) in G, the sequence of values returned by f corresponds to a walk from s to t in G. Otherwise, $\mathcal{A}$ is *defeated* by (or *fails* on) some graph G and some pair (s, t) in G. A routing algorithm $\mathcal{A}$ has *dilation* bounded by δ if for all graphs G and all origin-destination pairs (s, t) in $V(G)$ such that $s \neq t$, $r_{\mathcal{A}}(s, t) / \operatorname{dist}(s, t) \leq \delta$, where $r_{\mathcal{A}}(s, t)$ denotes the length of the route from s to t returned by $\mathcal{A}$. A routing algorithm that guarantees dilation δ is sometimes said to have a *stretch factor* [11] bounded by δ or to be *δ-competitive* [3].

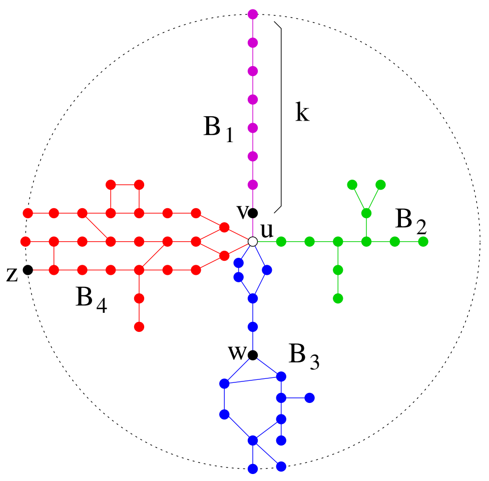

Fig. 1 In this example, $G_8(u)$ consists of four local components, corresponding to the four connected components of $G_8(u) \setminus \{u\}$. B_1, B_3, and B_4 are active components. B_2 is a passive component. B_1 and B_3 are constrained active components, but B_2 and B_4 are not. B_1 and B_2 are independent components, but B_3 and B_4 are not. Node v is an active neighbour of u and edge $\{u, v\}$ is an active edge. Node w is a constraint vertex in the constrained active component B_3. All three paths from u to z of length eight are active paths.

Other measures of quality in routing include congestion (traffic load), running time, and scalability. This paper considers questions of existence of successful routing algorithms. As such, the measures of interest are guaranteed delivery and bounds on dilation.

3 Related Work

In *position-based* routing, network nodes are embedded in some space (typically $\mathbb{R}^2$ or $\mathbb{R}^3$) and each node knows its spatial coordinates (i.e., nodes are *location-aware*). Position-based routing is also known as geo-routing, geographic routing, or geometric routing. Many recent results related to local routing are position based (e.g., [1, 4, 5, 10, 12, 13, 18, 20–22, 26]); we briefly describe some of these and discuss the interdependence between position-based and *position-oblivious* routing.

Greedy routing [12] (forward the message to the neighbour closest to the destination), compass routing [21] (forward the message along the edge that forms the smallest angle with the line segment to the destination), and greedy-compass routing [1] (apply greedy routing to the two edges adjacent to the line segment to the destination) are three well-known position-based routing algorithms, each of which succeeds on specific classes of graphs but is defeated by some planar graph [4]. All three algorithms are predecessor-oblivious, origin-oblivious, and 1-local.

To show that a routing algorithm fails on some class of graphs $\mathcal{G}$, it suffices to identify a graph in $\mathcal{G}$ on which the algorithm cycles infinitely without reaching the destination. Stronger negative results are those that apply to all routing algorithms, showing that no routing algorithm succeeds on a given class of graphs. Bose et al. [1] show that every position-based, predecessor-oblivious, origin-oblivious, 1-local routing algorithm is defeated by some convex subdivision.

Face routing [21] was one of the first position-based 1-local routing algorithms to succeed on more general classes of graphs embedded in the plane. In brief, face routing forwards the message in a clockwise direction along the edges of a face, and along the sequence of faces that intersect the line segment between the origin and destination nodes. Forward progress is guaranteed by storing a parameter such as the furthest intersection of the line segment with a visited face. As such, face routing is not stateless since it requires $\Theta(\log n)$ bits to be stored with the message. Face routing succeeds on planar graphs [21], on unit disc graphs [5], and on d-quasi unit disc graphs for any $d \in [1/\sqrt{2}, 1]$ [22]. See Bose et al. [5] and Kuhn et al. [22] for definitions of unit disc graphs and quasi unit disc graphs, respectively. Fraser considers a generalization of face routing to graphs embedded on tori [18]. See Guan [20] and Stojmenović [26] for discussions of face routing and its variants.

Although our discussion focuses on deterministic routing algorithms, we briefly note that randomized solutions permit k-local routing on more general classes of graphs. Chen et al. [9] show that while randomization can provide an (expected) guarantee of delivery, the expected dilation remains high. Specifically, they show that for every randomized position-based, predecessor-oblivious, origin-oblivious, 1-local routing algorithm, there exists a convex subdivision in the plane on which the expected route length is $\Omega(n^2)$, matching the expected length of a random walk from s to t. Flury and Wattenhofer consider the problem of randomized local routing on unit ball graphs [13] and show that any randomized position-based local routing algorithm has expected route length $\Omega(l^3)$, where l denotes the length of the shortest path.

Durocher et al. [10] show that for every fixed k, every origin-aware, predecessor-aware, k-local routing algorithm fails on some unit ball graph. The proof has two parts. First, the corresponding position-oblivious result is proven: for every fixed k, every origin-aware, predecessor-aware, k-local routing algorithm fails on some graph. Next, a k-local reduction from (unembedded) graphs to unit ball graphs is used to show that if some (possibly position-based) k-local routing algorithm succeeds on unit ball graphs, then some position-oblivious k-local routing algorithm succeeds on all graphs. This interdependence between position-based and position-oblivious routing algorithms motivates the question of exploring the boundary between feasibility and impossibility of local routing algorithms as a function of the local information available. In this paper we consider the position-oblivious case.

4 When Local Routing is Impossible: Negative Results

In this section we present negative results: every k-local routing algorithm fails on some graph when the degree of locality k is less than the given bound. For each combination of origin-awareness/obliviousness and predecessor-awareness/obliviousness, we demonstrate a counter-example consisting of a set of graphs such that any k-local routing algorithm fails on at least one of the graphs in the set.

4.1 Properties of Local Routing Functions

The proofs of Theorems 1 through 3 refer to Lemma 1 and Corollary 1, which generalize an observation of Durocher et al. [10] showing that if a k-local routing algorithm guarantees delivery, then each local routing function corresponds to a circular permutation (under certain conditions). Recall that a circular permutation of n distinct elements is an ordering of these elements in a cycle.

Lemma 1 *Given an arbitrary graph G and any node $u \in V(G)$ such that*

1. $\deg(u) \geq 2$,
2. *every local component of u is an independent active component, and*
3. *neither the origin node s nor the destination node t is in $G_k(u)$,*

if $\mathcal{A}$ is an origin-aware, predecessor-aware, k-local routing algorithm that guarantees delivery, then the local routing function of $\mathcal{A}$ at u is a circular permutation of $\mathrm{Adj}(u)$.

Proof Choose any $k \geq 1$, any node u, and any k-neighbourhood $G_k(u)$ such that Properties 1 through 3 hold. Suppose $\mathcal{A}$ is any k-local routing algorithm that guarantees delivery for which the local routing function f_u is not a circular permutation.

Case 1. Suppose f_u is not a permutation. That is, f_u is not surjective. Therefore, there exists some $v \in \mathrm{Adj}(u)$ such that for all $w \in \mathrm{Adj}(u)$, $f_u(w) \neq v$. Let B_1 denote

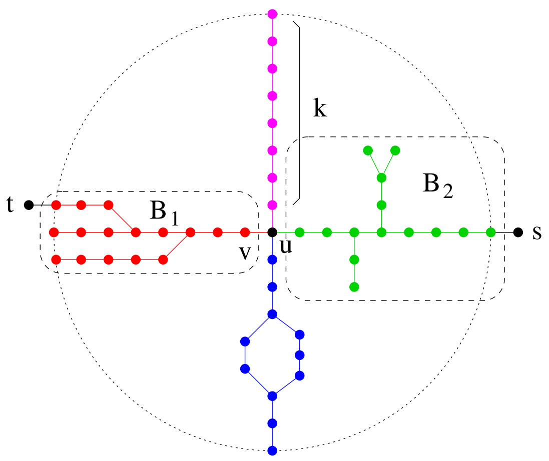

Fig. 2 This example illustrates the graph constructed in Case 1 of Lemma 1 for a given $G_k(u)$ when $k = 8$. Note that $G_k(u)$ consists of independent active components.

the local component of u that contains v and let B_2 denote any other local component of u. By Property 2, each local component of u is an active component. Let G denote a graph that contains $G_k(u)$ such that node t has degree one and is the only node adjacent to B_1 outside $G_k(u)$. Similarly, let node s have degree one such that it is the only node adjacent to B_2 outside $G_k(u)$. See Figure 2. Since for all $w \in \mathrm{Adj}(u)$, $f_u(w) \neq v$, the message will never enter B_1 and, consequently, will never reach t. Therefore, Algorithm $\mathcal{A}$ fails on graph G, deriving a contradiction.

Case 2. Suppose f_u is a permutation but not a derangement (a derangement is a complete permutation). Therefore, $f_u(v) = v$ for some $v \in \mathrm{Adj}(u)$. Let G be a graph as defined in Case 1, with the exception that nodes s and t are interchanged. It follows that the message will never enter any local component other than B_1 and, consequently, will never reach t. Therefore, Algorithm $\mathcal{A}$ fails on graph G, deriving a contradiction.

Case 3. Suppose f_u is a derangement but not a circular permutation. Therefore, f_u cannot be expressed as a single permutation cycle. Let $(a_1 \ldots a_k)$ and $(b_1 \ldots b_j)$ denote any two distinct permutation cycles of f_u. Observe that $\{a_1, \ldots a_k\}$ and $\{b_1, \ldots, b_j\}$ are disjoint subsets of $\mathrm{Adj}(u)$. Let G be a graph as defined in Case 1, with the exception that node s is adjacent to a local component B_1 rooted at a node in $\{a_1, \ldots a_k\}$ and t is adjacent to a local component B_2 rooted at a node in $\{b_1, \ldots, b_j\}$. It follows that the message will never enter B_2 and, consequently, will never reach t. Therefore, Algorithm $\mathcal{A}$ fails on graph G, deriving a contradiction.

All three cases derive a contradiction and our assumption must be false. Therefore, the local routing function f_u must be a circular permutation. □

In other words, without additional information on which to base a local routing decision, an intermediate node u must try all possibilities and sequentially forward the message to each of its neighbours. When node u has degree two, a unique circular permutation is possible: a message received from one neighbour of u must be forwarded to the opposite neighbour. If node u has degree j, then $(j-1)!$ circular permutations are possible. If routing Algorithm $\mathcal{A}$ is origin oblivious, then Lemma 1 gives:

Corollary 1 *Given an arbitrary graph G and any node $u \in V(G)$ such that*

1. $\deg(u) \geq 2$,
2. *every local component of u is an independent active component, and*
3. *the destination node t is not in $G_k(u)$,*

if $\mathcal{A}$ is an origin-oblivious, predecessor-aware, k-local routing algorithm that guarantees delivery, then the local routing function of $\mathcal{A}$ at u is a circular permutation on $\mathrm{Adj}(u)$.

Proof Given any node u, any $k \geq 1$, and any k-neighbourhood $G_k(u)$, if Properties 1 through 3 hold (as defined in Lemma 1), then the local routing function f_u is a circular permutation by Lemma 1. Since $\mathcal{A}$ is origin oblivious, function f_u remains constant for any given $G_k(u)$ and t, regardless of s. In particular, f_u is a circular permutation regardless of whether or not s is contained in $G_k(u)$. The result follows. □

Theorems 1 through 3 and Corollary 2 establish lower bounds corresponding to each of the four combinations of k-local routing algorithms: origin-aware/oblivious and predecessor-aware/oblivious.

4.2 Predecessor Aware and Origin Aware

Theorem 1 *For every $k < \lfloor (n+1)/4 \rfloor$, every origin-aware, predecessor-aware, k-local routing algorithm fails on some connected graph.*

Proof Choose any $k < \lfloor (n+1)/4 \rfloor$, $k \in \mathbb{Z}^+$. Therefore, $k \in \{1, \ldots, r\}$, where $r = \lfloor (n-3)/4 \rfloor$. Let G_1, G_2, and G_3 denote the graphs illustrated in Figure 3, such that each path P_1 through P_4 consists of r vertices that are labelled consistently relative to node u in all three graphs. In each graph, $G_k(u)$ is a tree consisting of four paths of length k rooted at u, none of which contains s nor t. In addition to the $4r$ nodes in paths P_1 through P_4, each graph includes nodes u, s, and t. Depending on the value of $n \bmod 4$, between zero and three extra nodes remain; these are added between s and P_1 to bring the total number of nodes to n. Any successful routing algorithm must pass the message across P_1 to node u. Since u has degree four, its local routing function is one of six possible circular permutations by Lemma 1. The remaining nodes have degree at most two. Therefore, when the message is passed to a node on a path that does not contain s or t, by Lemma 1, the message must continue forward until it returns again to u. As shown in Table 3, for each of the six possible routing strategies, the message never enters the path containing t in at least one of the graphs G_1, G_2, or G_3. That is, for every routing strategy $\mathcal{A}$, there exists a graph on which $\mathcal{A}$ fails. □

routing strategy	circular permutation	succeeds	fails
1	$(P_1P_2P_3P_4)$	G_1, G_3	G_2
2	$(P_1P_2P_4P_3)$	G_1, G_2	G_3
3	$(P_1P_3P_2P_4)$	G_2, G_3	G_1
4	$(P_1P_3P_4P_2)$	G_1, G_2	G_3
5	$(P_1P_4P_2P_3)$	G_2, G_3	G_1
6	$(P_1P_4P_3P_2)$	G_1, G_3	G_2

Table 3 Each routing strategy corresponds to a circular permutation of the neighbours of s.

4.3 Predecessor Aware and Origin Oblivious

Using an argument similar to the proof of Theorem 1, we now show that the lower bound on the locality parameter k increases to $\lfloor (n+1)/3 \rfloor$ for origin-oblivious k-local routing algorithms:

Theorem 2 *For every $k < \lfloor (n+1)/3 \rfloor$, every origin-oblivious, predecessor-aware, k-local routing algorithm fails on some connected graph.*

Proof Choose any $k < \lfloor (n+1)/3 \rfloor$, $k \in \mathbb{Z}^+$. Therefore, $k \in \{1, \ldots, r\}$, where $r = \lfloor (n-2)/3 \rfloor\}$. Let G_1, G_2, and G_3 denote the graphs illustrated in Figure 4, such that each path P_1 through P_3 consists of r vertices that are labelled consistently relative to node s in all three graphs. In each graph, $G_k(s)$ is a tree consisting of three paths of length k rooted at s, none of which contains t. In addition to the $3r$ nodes in paths P_1 through P_3, each graph includes nodes s and t. Depending on the value of $n \bmod 3$, between zero and two extra nodes remain; these are added between t and the corresponding path P_i nearest to t to bring the total number of nodes to n. Since node s has degree three, its local routing function is one of two possible circular permutations by Corollary 1. A routing strategy must specify the direction in which a message initially leaves node s (three directions

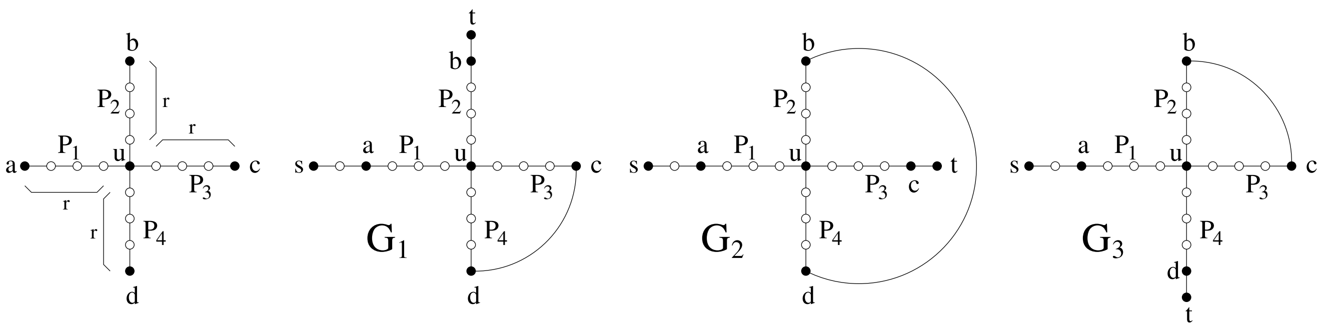

Fig. 3 The k-neighbourhood $G_k(u)$ is identical in graphs G_1, G_2, and G_3. In this example, suppose n mod $4 = 0$. Consequently, one extra node is added between s and P_1 such that the total number of nodes is n.

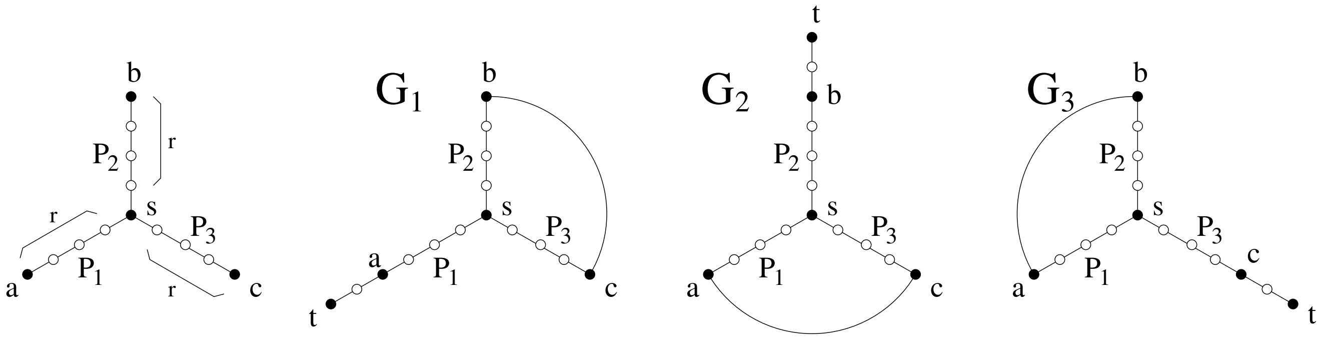

Fig. 4 The k-neighbourhood $G_k(s)$ is identical in graphs G_1, G_2, and G_3. In this example, suppose n mod $3 = 0$. Consequently, one extra node is added next to t such that the total number of nodes is n.

are possible). The remaining nodes have degree at most two. Therefore, when the message is passed to a node on a path that does not contain t, by Corollary 1, the message must continue forward until it returns again to node s. As shown in Table 4, for each of the six possible routing strategies, the message never enters the path containing t in at least one of the graphs G_1, G_2, or G_3. That is, for every routing strategy $\mathcal{A}$, there exists a graph on which $\mathcal{A}$ fails. □

4.4 Predecessor Oblivious and Origin Aware

When knowledge of the predecessor node is withheld, the lower bound on the locality parameter k increases to $\lfloor n/2 \rfloor$ for k-local routing algorithms:

Theorem 3 *For every $k < \lfloor n/2 \rfloor$, every origin-aware, predecessor-oblivious, k-local routing algorithm fails on some connected graph.*

Proof Choose any $k < \lfloor n/2 \rfloor$, $k \in \mathbb{Z}^+$. Therefore, $k \in \{1, \ldots, r\}$, where $r = \lfloor n/2 \rfloor - 1$. Let G_1 denote a path of n vertices with the origin node s located at the $(r+1)$st vertex and the destination node t located at the far end. Let G_2 denote the analogous graph upon moving node t to the opposite end of the path. Let the remaining nodes be labelled consistently relative to node s in both graphs. See Figure 5. The k-neighbourhood $G_k(s)$ is identical in G_1 and G_2. If Algorithm $\mathcal{A}$ sends the message right at s, then $\mathcal{A}$ fails on graph G_2 since it must eventually send the message left, at which point its behaviour becomes cyclic. Similarly, if Algorithm $\mathcal{A}$ sends the message left at s, then it fails on graph G_1. □

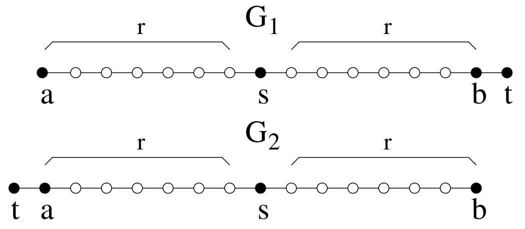

Fig. 5 For any $k < \lfloor n/2 \rfloor$, the k-neighbourhood of s does not contain t.

4.5 Predecessor Oblivious and Origin Oblivious

Finally, if we further constrain the knowledge available to intermediate nodes (e.g, remove knowledge of the origin), then the lower bound on the locality parameter k given in Theorem 3 applies:

Corollary 2 *For every $k < \lfloor n/2 \rfloor$, every origin-oblivious, predecessor-oblivious, k-local routing algorithm fails on some connected graph.*

routing strategy	circular permutation	initial direction	succeeds	fails
1	$(P_1P_2P_3)$	toward a	G_1, G_3	G_2
2	$(P_1P_2P_3)$	toward b	G_1, G_2	G_3
3	$(P_1P_2P_3)$	toward c	G_2, G_3	G_1
4	$(P_1P_3P_2)$	toward a	G_1, G_2	G_3
5	$(P_1P_3P_2)$	toward b	G_2, G_3	G_1
6	$(P_1P_3P_2)$	toward c	G_1, G_3	G_2

Table 4 Each routing strategy corresponds to a circular permutation of the neighbours of u paired with an initial direction.

Proof The result follows by Theorem 3. □

4.6 Dilation

We now consider lower bounds on dilation for k-local routing algorithms.

Theorem 4 *For any $k < n/2$, no k-local routing algorithm can guarantee dilation less than*

$$\frac{2n-3k-1}{k+1}, \tag{1}$$

regardless of whether the algorithm is predecessor-aware/oblivious or origin-aware/oblivious.

Proof Choose any n, any $k \in [1, n/2)$, and any k-local routing algorithm $\mathcal{A}$. If $\mathcal{A}$ fails on some graph, then $\mathcal{A}$ has unbounded dilation. In particular, the dilation exceeds (1). Therefore, suppose that $\mathcal{A}$ succeeds on all graphs. Given a set of n distinct vertex labels, let $\mathcal{P}$ denote the corresponding set of all $n!/2$ distinct paths of length n. Suppose the origin and destination nodes are labelled s and t, respectively. For every path $P \in \mathcal{P}$, the local neighbourhood of every internal node on P has two independent components; since $n \geq 2k+1$, at most one of these components is passive. By Observation 1, if the message changes direction at a node that has two active components, then nodes on the path beyond the corresponding local component (where t could be located) will never be visited. Consequently, for any node u, if $G_k(u)$ has two active components, a message received from u's left neighbour must be forwarded to its right neighbour, and vice-versa (i.e., the local routing function at u is a circular permutation).

Consider any local neighbourhood $G_k(u)$ that is a path of length $2k$. Thus, $G_k(u)$ has two active components. There exist paths P and P' in $\mathcal{P}$ that contain $G_k(u)$ such that s lies to the left of u in both P and P', and t lies to the left of $G_k(u)$ in P but to the right of $G_k(u)$ in P'. Consider the first time node u receives the message. Algorithm $\mathcal{A}$ sends the message toward the destination t in path P or path P', and away from t in the other. Consequently, there exists a path P'' in $\mathcal{P}$ such that $\operatorname{dist}(s,t) = k+1$, $\mathcal{A}$ initially forwards the message away from s, and $\mathcal{A}$ continues forwarding the message away from t while both local components of the current node are active. In particular, Algorithm $\mathcal{A}$ can send the message away from s to $n-2k-1$ nodes before the message reaches a node that has a passive component such that for each node u visited, $\operatorname{dist}(u,t) \geq k+1$. See Figure 6. The message must return to s before proceeding in the opposite direction back to t. The corresponding route has length at least $2(n-2k-1)+\operatorname{dist}(s,t) = 2n-3k-1$, while the shortest path has length $\operatorname{dist}(s,t) = k+1$. □

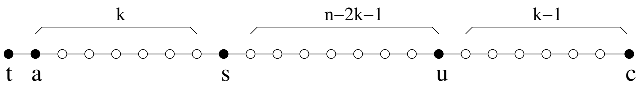

Fig. 6 Node u is the leftmost node right of s that can confirm that t does not lie to the right of c.

The bound on dilation (1) is perhaps more clearly expressed in the limit as the number of nodes approaches infinity and $k = c \cdot n$ for some constant $c \in (0,1)$. We denote this limit by $S(k)$:

$$S(k) = \lim_{n\to\infty} \frac{2n-3k-1}{k+1} = \frac{2n}{k} - 3. \tag{2}$$

Of particular interest are the values of $k \in \{n/4, n/3, n/2\}$, for which the corresponding bounds on dilation are 5 (when $k = n/4$), 3 (when $k = n/3$), and 1 (when $k \to n/2$). As shown in Theorems 7 and 8 and Corollary 5, these bounds are tight for $k = n/3$ and $k = n/2$.

5 When Local Routing is Possible: Routing Strategies

In this section we present positive results: there exists a successful k-local routing algorithm when the degree of locality k exceeds the given bound. We describe a k-local routing algorithm for each combination of origin-awareness/obliviousness and predecessor-awareness/obliviousness.

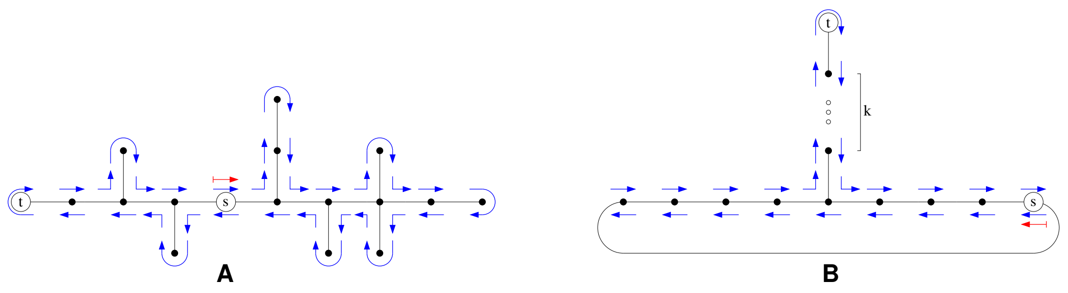

Fig. 7 The red arrow denotes the neighbour to which the message is initially forwarded from the origin s. Blue arrows denote subsequent forwarding decisions as a function of the neighbour from which the message was received. (**A**) The right-hand rule guarantees delivery on any tree. (**B**) The right-hand rule can fail if some cycle has length greater than $2k$. Furthermore, it can occur that the message is never forwarded to a node whose k-neighbourhood contains the destination t. In this example, if $k \leq 4$ then the right-hand rule forwards the message from s counter-clockwise around the cycle with t being excluded from every visited node's k-neighbourhood.

5.1 Predecessor Aware and Origin Aware

Given any $k \geq n/4$, we describe a predecessor-aware, origin-aware, k-local routing algorithm that succeeds on all connected graphs on n vertices.

Motivation: Generalizing the Right-Hand Rule

Routing on a tree is easily accomplished using a right-hand rule. That is, the message is passed along the sequence of edges on the face determined by any non-crossing embedding of the tree in the plane. Specifically, if every local routing function is a circular permutation, then every message is guaranteed to reach its destination (i.e., knowledge of the embedding is not required). A right-hand rule can be implemented on any graph G by selecting a subset of its edges that forms a spanning tree of G.

If G contains only local cycles, then any cycle on which a node u lies is entirely visible in $G_k(u)$. Node u can label one edge on every local cycle as *dormant*. If this labelling rule were applied consistently at all nodes and the message were forwarded only across *routing* (non-dormant) edges, then it would suffice to define each local routing function to be a cyclic permutation of its routing edges.

If G may contain cycles of arbitrary length, then the right-hand rule cannot be applied directly since a node u has no knowledge of cycles not entirely contained in $G_k(u)$. In particular, cyclic behaviour can occur such that the message never comes within distance k of the destination. See Figure 7. Since $k \geq n/4$, however, the number of cycles of length $2k+1$ or greater is limited. Using a simple set of k-local rules which we now define, we show how to guarantee delivery in any graph G.

Preprocessing: Identifying Routing Edges in $\boldsymbol{G_k(u)}$

When a message arrives at a node u, the algorithm begins with a k-local preprocessing step to identify the edges of $G_k(u)$ on which routing takes place. We call these edges *routing edges* and denote the corresponding edge-induced subgraph of $G_k(u)$ by $G'_k(u)$. Specifically, some edges of $G_k(u)$ may be identified as *dormant edges* locally at u. Once dormant edges are removed from $G_k(u)$, the remaining edges that lie on paths rooted at u with length at most k are identified as routing edges. Graph $G'_k(u)$ is not always a spanning subgraph of $G_k(u)$; any remaining edges of $G_k(u)$ (those that are neither routing nor dormant edges) are edges whose distance from u along routing edges exceeds k and, consequently, are not included in $G'_k(u)$. Every edge adjacent to u, however, is identified either as a routing or dormant edge; this classification forms the basis of our routing algorithm.

The unique labelling of nodes determines a strict total order on the edges of G (e.g., label each edge by concatenating the labels of its endpoints and order edge labels lexicographically). We refer to the label of an edge e as its *rank*, denoted rank(e), where edge a precedes edge b in the order if and only if rank$(a) <$ rank(b). In particular, any set of edges has an edge of minimum rank. Using a technique similar to those applied by Li et al. [23] and Chávez et al. [7] (used to break cycles in $G_k(u)$ to construct a k-local minimum spanning tree on a unit disc graph) graph $G'_k(u)$ is constructed locally at node u by classifying the edge of minimum rank on every local cycle of u as a dormant edge. See Figure 8. A local cycle may contain multiple dormant edges if two or more local cycles share a common edge. See Figure 9.

If edge e is not classified as a dormant edge in any local neighbourhood, then we say e is *consistent*. Otherwise, we say e is *inconsistent*. A consistent path or cycle is one whose edges are all consistent. Similarly,

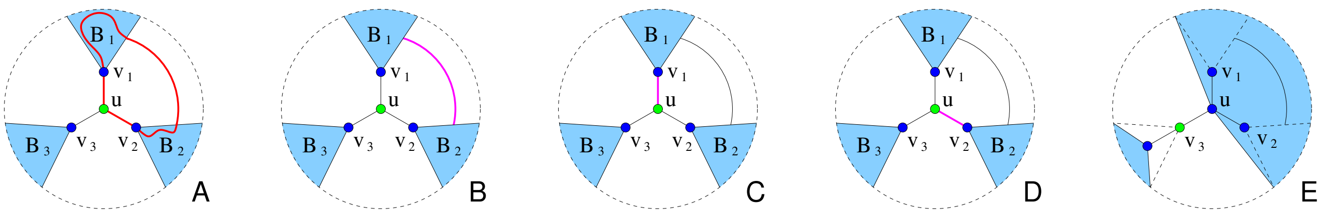

Fig. 8 ***k*-local preprocessing.** Suppose node u has three active neighbours, v_1 through v_3, and $G_k(u)$ contains a local cycle that includes vertices v_1, u, and v_2 (**A**). The preprocessing step classifies one of the edges on the cycle as a dormant edge (magenta). The selected edge may be distant from u (**B**) or adjacent to u (**C** and **D**). The choice of dormant edge does not affect nodes not on the cycle (e.g., v_3) since all edges of the local cycle lie in the same local component of the corresponding vertex (**E**); in particular, none of the cycle's edges are adjacent to v_3.

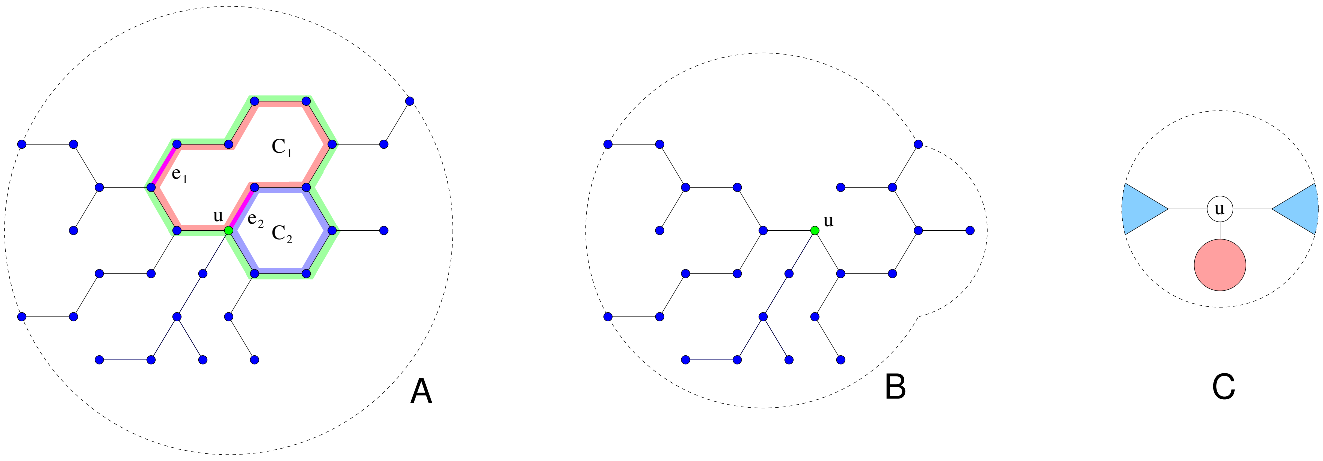

Fig. 9 (**A**) Suppose $k = 5$ and edges e_1 and e_2 have the lowest and second-lowest ranks, respectively, among all edges in $G_k(u)$. $G_k(u)$ contains two local cycles: C_1 (light red) and C_2 (light blue). $G_k(u)$ also contains a third cycle (light green), but its length exceeds $2k$. Edges e_1 and e_2 are classified as dormant for cycles C_1 and C_2, respectively. (**B**) The resulting subgraph $G'_k(u)$ is illustrated. Edges and vertices whose distance from u along routing edges is greater than k are not included in $G'_k(u)$, even if these are routing edges that appear in $G_k(u)$. (**C**) This simplification of $G'_k(u)$ illustrates that u has two independent active components (light blue) and one independent passive component (light red).

some edge in an inconsistent path or cycle is inconsistent.

We establish some properties of the set of consistent edges in Lemmas 2 and 3, and Proposition 1.

Lemma 2 *Every edge adjacent to u in $G'_k(u)$ is consistent.*

Proof Suppose there exists an inconsistent edge $e = \{u, v\}$ such that e is a routing edge in $G'_k(u)$. Therefore, edge e is dormant in $G_k(w)$ for some node w. Furthermore, there exists a local cycle C in $G_k(w)$ that contains nodes u, v, and w on which edge e has minimum rank. Since the cardinality of C is at most $2k$, cycle C must be contained in $G_k(u)$ and, consequently, edge e is classified as dormant in $G_k(u)$, deriving a contradiction. □

Lemma 3 is similar to that of Li et al. [23, page 5, Lemma 2]. Lemma 3 is included here for completeness since the definition of edge consistency used by Li et al. differs slightly from the one used in this paper.

Lemma 3 *Given any two nodes u and v in G, there exists a consistent path from u to v.*

Proof Let D denote the set of inconsistent edges that lie on paths from u to v in G and let $e = \{a, b\}$ denote the edge of maximum rank in D. Since edge e is classified as dormant in $G_k(w)$ for some node w, it must lie on a local cycle C consisting of edges whose ranks are greater than rank(e). Since e has maximum rank among all edges in D, therefore, the path $C \setminus \{e\}$ joining a and b is consistent. Consider the set $D' = D \setminus \{e\}$ and let $e' = \{a', b'\}$ denote the edge of maximum rank in D'. Using an analogous argument it follows that there exists a consistent path from a' to b'. In particular, if some path from a' to b' includes edge e, there exists a corresponding consistent path from a' to b' that avoids e by following the path $C \setminus \{e\}$. This argument can be repeated recursively until $D' = \varnothing$. □

Observe that all components in $G'_k(u)$ are independent components. The number of active neighbours of u

in $G'_k(u)$ is its *active degree*. Since an active edge joins u to a component containing at least $k \geq n/4$ nodes and $4k+1 > n$, we get the following proposition:

Proposition 1 *Every node has active degree at most 3.*

Routing Algorithm

We now describe the local routing algorithm applied at each node u upon receiving a message. If the destination t is in $G_k(u)$, then the message is forwarded along a shortest path to t (see Algorithm 1, Case 1). If t is not in $G_k(u)$ (and therefore not in $G'_k(u)$), then the message is forwarded along a routing edge into an active component of u (since passive components are dead ends with respect to consistent edges). The algorithm must ensure that previous forwarding decisions are not repeated (without explicitly recording these). In brief, the algorithm applies a strategy inspired by the right-hand rule to sequentially explore all active components, except when doing so could lead to a dead end or to cyclic behaviour. Potential future cycling is identified and averted by using the origin node s as a point of reference; if the message is sent toward s, its direction is eventually altered to avoid repeating previous routing decisions (see Algorithm 1, Cases 2 and 4). The preprocessing step need not be repeated unless the network topology changes. Once a node has identified its routing edges, a simple set of rules determine forwarding decisions, defined as a function of five local navigational cues at u:

1. whether the destination node t lies in $G_k(u)$,
2. whether node u is the origin node s (i.e., $u = s$),
3. whether the origin node s lies in a passive component of $G'_k(u)$,
4. the number of active components in $G'_k(u)$, and
5. the neighbour of u from which the message was received.

The routing algorithm consists of four cases outlined below, each of which applies simple deterministic rules to make a forwarding decision. Let u denote the current node.

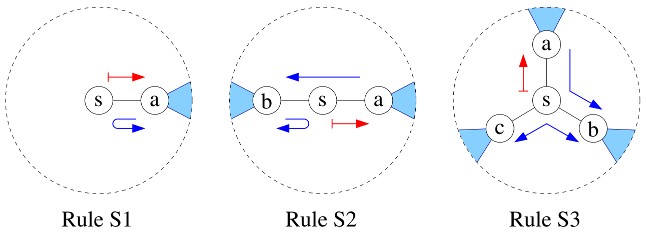

Fig. 10 Algorithm 1, Case 2: The message is at the origin node s. Suppose nodes are labelled such that rank$(a) <$ rank$(b) <$ rank(c). Light blue regions denote active components of s. If s has i active components, then rule Si is applied. Passive components are not illustrated. The red arrow denotes the neighbour to which the message is initially forwarded from s. Blue arrows denote subsequent forwarding decisions as a function of the neighbour from which the message was received.

.

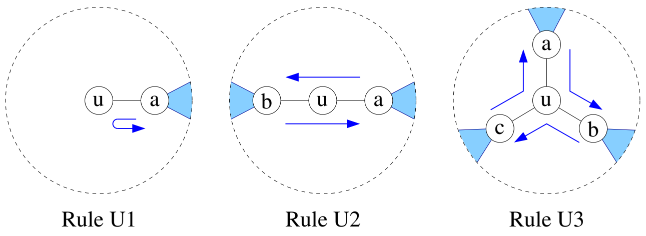

Fig. 11 Algorithm 1, Case 3: The message is at a node $u \neq s$ such that either s is in an active component of $G'_k(u)$ or $s \notin V(G'_k(u))$. Light blue regions denote active components of u. If u has i active components, then rule Ui is applied. Passive components are not illustrated. Blue arrows denote forwarding decisions as a function of the neighbour from which the message was received.

.

Algorithm 1: $(n/4)$-local, origin-aware, predecessor-aware routing algorithm

CASE 1. Suppose dist$(u,t) \leq k$. That is, $t \in V(G_k(u))$. The algorithm forwards the message to any neighbour of u on a shortest path from u to t until the message arrives at t.

CASE 2. Suppose dist$(u,t) > k$ and $u = s$. Forwarding decisions are illustrated in Figure 10. If $v = \perp$ (i.e., the message is being sent from the origin s for the first time) then s forwards the message to its active neighbour of lowest rank (node a).

CASE 3. Suppose dist$(u,t) > k$, $u \neq s$, and either $s \notin V(G'_k(u))$ or s is in an active component of u. Forwarding decisions are illustrated in Figure 11.

CASE 4. Suppose dist$(u,t) > k$, $u \neq s$, and s is in a passive component of $G'_k(u)$. Forwarding decisions are illustrated in Figure 12. If the message is received from the passive component containing the origin node s, then u forwards the message to its active neighbour of lowest rank (node a).

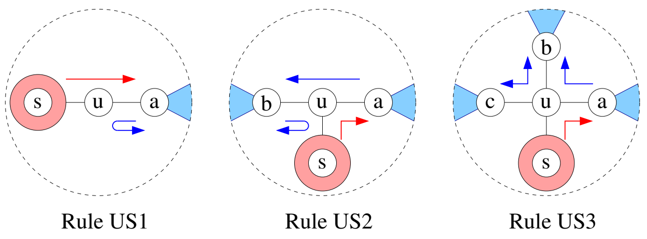

Fig. 12 Algorithm 1, Case 4: The message is at a node $u \neq s$ such that s is in a passive component of $G'_k(u)$. Suppose nodes are labelled such that $\mathrm{rank}(a) < \mathrm{rank}(b) < \mathrm{rank}(c)$. Light blue regions denote active components of u. If u has i active components, then rule USi is applied. The light red region denotes the passive component of u that contains s. Other passive components are not illustrated. The red arrow denotes the neighbour to which the message is initially forwarded from the passive component containing s. Blue arrows denote subsequent forwarding decisions as a function of the neighbour from which the message was received.

The success of Algorithm 1 relies on the property that each node has active degree at most 3. Notice that the lower bound argument of the proof of Theorem 1 consists of graphs that have one node with active degree 4; by Proposition 1, this cannot occur when $k \geq n/4$.

Properties of Algorithm 1

We begin by establishing the following properties which are used in Lemma 7 to show the correctness of Algorithm 1. Corollaries 3 and 4 follow from the definition of Algorithm 1 and by Lemma 2.

Corollary 3 *Algorithm 1 forwards a message only along consistent edges.*

Corollary 4 *Any message that enters a passive component must pass through its root. Furthermore, Algorithm 1 forwards a message into a passive component if and only if that component contains the destination node t.*

Lemma 4 *If Rules S1, U1, or US1 are applied at a node u to reverse the direction of a message, then $G'_k(u)$ has a passive component containing at least $k-1$ nodes.*

Proof Suppose Rule U1 is applied at a node u such that u receives the message from its neighbour a and returns the message immediately back to a. By the definition of Rule U1, node a is the unique active neighbour of u. Furthermore, by the definition of Algorithm 1, Rule U1 is applied only if $\mathrm{dist}(u,t) > k$. It follows that $\mathrm{dist}(a,t) > k$, otherwise, (by Algorithm 1) the message would not have been forwarded to u. Consequently, node a forwarded the message to an independent active component C of $G'_k(a)$. Every active component contains at least k nodes. Since u is a constraint vertex in C, it follows that u has a passive component containing at least $k-1$ nodes. The result follows by applying an analogous argument to Rules S1 and US1. □

Lemma 5 *Every consistent cycle in G has length at least $2k+1$.*

Proof Some edge on every cycle C of length at most $2k$ is classified as dormant in $G_k(u)$ for every node u in C. The result follows. □

Expressed in graph-theoretic terms, the graph induced by the consistent edges of G has girth at least $2k+1$.

Lemma 6 *Any graph of girth at least g that contains two or more cycles has at least $3g/2-1$ vertices.*

Proof Choose any g and any graph G with girth at least G that contains at least two cycles.

Case 1. Suppose G contains two vertex-disjoint cycles. Each cycle has at least g vertices. Therefore, $|V(G)| \geq 2g$.

Case 2. Suppose G contains two cycles that have a single vertex in common. Similarly, $|V(G)| \geq 2g-1$.

Case 3. Suppose all pairs of cycles in G have at least two vertices in common. Any two intersecting cycles define at least one additional cycle. It suffices to show that a graph of girth g with exactly three cycles has at least $3g/2-1$ vertices. Such a graph consists of three paths joined at two vertices of degree three. Let a, b, and c denote the number of vertices on each path, respectively, not including the two vertices of degree three. Therefore,

$$\min\{a+b, a+c, b+c\} + 2 \geq g$$
$$\Rightarrow a+b+c+2 \geq \frac{3g}{2} - 1.$$

The result follows since $|V(G)| \geq a+b+c+2$. □

Correctness of Algorithm 1

We now prove the correctness of Algorithm 1 and derive a tight bound on the corresponding dilation.

Lemma 7 *Given any connected graph G on n nodes, any $k \geq n/4$, and any $\{s,t\} \subseteq V(G)$, Algorithm 1 successfully delivers a message from node s to node t.*

Proof By Proposition 1, every node u has at most three active components. Consequently, one of Cases 1 through 4 of Algorithm 1 is applicable at each step.

Suppose Algorithm 1 is defeated by some graph G, for some $k \geq n/4$ and some origin-destination pair $(s,t) \in V(G) \times V(G)$. Since n is finite and Algorithm 1 is deterministic, the message must visit a repeating sequence of vertices and edges in G. Let R denote the corresponding repeating subgraph of G. Let $S \subseteq V(G)$ denote the set of vertices of G not in R visited prior to entering the repeating sequence R. Finally, let $T = V(G) \setminus (V(R) \cup S)$ denote the set of vertices of G that are never visited. The set $\{V(R), S, T\}$ partitions $V(G)$.

Since the message does not reach its destination t, Case 1 of Algorithm 1 never occurs. Consequently,

$$\begin{aligned} \forall v \in V(R) \cup S, \qquad & \operatorname{dist}(v,t) \geq k+1 \\ \Rightarrow \qquad & |T| \geq k+1. \end{aligned} \tag{3}$$

Furthermore,

$$\begin{aligned} |V(R)| &= n - |T| - |S|, && \text{by definition of } R,\ S, \text{ and } T, \\ &\leq n - |T| \\ &\leq 4k - |T|, && \text{since } k \geq n/4, \\ &\leq 3k - 1, && \text{by (3).} \end{aligned} \tag{4}$$

By Lemmas 5 and 6, and Inequality (4), R contains at most one consistent cycle. By Lemma 3, there exists a consistent path in G from t to some vertex in $V(R)$. Let $\{q_R, q_T\}$ denote the last edge on such a path such that $q_R \in V(R)$ and $q_T \in T$. It follows that q_T is an active neighbour of q_R that is never visited. Consequently, neither Rule S1, U1, nor US1 is applied at q_R. Similarly, since q_R is visited multiple times (i.e., $q_R \in V(R)$), neither Rule S2, U2, nor US2 is applied at q_R, since each of these rules implies q_T having received the message. Thus, only Rules S3, U3, or US3 may be applied at q_R, implying the following observation:

Observation 2 *Node q_R has active degree three. Furthermore, two of its active neighbours are in $V(R)$ while the third, q_T, is in T.*

Case 1. Suppose graph R contains a consistent cycle. By Lemma 5, this cycle must contain at least $2k+1$ nodes. Consequently,

$$|V(R)| \geq 2k+1. \tag{5}$$

Since $k \geq n/4$, by (3) and (5), we have,

$$|S| \leq k-2. \tag{6}$$

The following observation follows from Corollary 4, Lemma 4, and (6):

Observation 3 *Neither Rule S1, U1, nor US1 can be applied in the repeating sequence.*

Therefore, the message can reverse its direction at a node in R if and only if Rule S2 or US2 is applied. Consequently, every vertex in R has active degree two or greater in G. Rule S2 applies if and only if s has two active components. Conversely, Rule US2 can be applied only if s has only one active component since s is in a passive component of u. Therefore at most one of Rule S2 or US2 is applicable, implying there is at most one node in R at which the message can reverse its direction. Both Rules S2 and US2 initially forward the message in the opposite direction from that in which the reversal occurs. Consequently, the active component in which the message is originally forwarded cannot be in R if the message reverses its direction by Rule S2 or US2 in R. Therefore, either the initial active component is in S, contradicting (6), or the message reverses its direction in that component, requiring that one of Rule S1, U1, or US1 be applied, contradicting Observation 3. Thus, none of Rules S1, U1, US1, U2, or US2 can be applied to reverse the message direction in R. We conclude that R consists of a single connected cycle (without any dangling branches).

Case 1a. Suppose the origin node s is in a passive component of some node $p \in V(R)$ that has active degree two. Consider the first time node p receives the message. Rule US2 is applied a first time, forwarding the message in one direction around cycle R. Since $s \notin V(R)$ it follows that Rule S2 is not applied and the message continues around cycle R until it returns to node p. Rule US2 is applied once again, forwarding the message in the opposite direction around cycle R. Upon returning to node p, the message is again forwarded in the same direction, and continues cycling in this direction infinitely. The message has visited every node in R at least once from each direction. By Observation 2, Rule U3 is applied at q_R. In particular, Rule U3 is applied from two directions. In one of these directions the message will be forwarded along the edge $\{q_R, q_T\}$. Since $q_T \notin V(R)$, we derive a contradiction.

Case 1b. Suppose the origin node s is in a passive component of some node $p \in V(R)$ that has active degree three. We derive a contraction by an argument analogous to Case 1a, by applying Rule US3 instead of Rule US2 at node p.

Case 1c. Suppose the origin node s is in $V(R)$ and s has active degree two. We derive a contraction by an argument analogous to Case 1a, by applying Rule S2 instead of Rule US2 at node $p = s$.

Case 1d. Suppose the origin node s is in $V(R)$ and s has active degree three. We derive a contraction by an argument analogous to Case 1b, by applying Rule S3 instead of Rule US3 at node $p = s$.

Case 1e. Suppose the origin node s is in an active component of some node $p \in V(R)$ and $s \in S$. Therefore, p has active degree three. Since the message cannot reverse its direction in R and R is a cycle, the entire active component containing s must be in S, implying $|S| \geq k$. We derive a contradiction by (6).

Case 2. Suppose graph R is acyclic. Since R must be connected, R is a tree. Consequently, the message crosses every edge in $E(R)$ from both directions. By Observation 2, Rule U3 is applied at q_R. Again, Rule U3 is applied from two directions. In one of these directions the message will be forwarded along the edge $\{q_R, q_T\}$. Since $q_T \notin V(R)$, we derive a contradiction.

Each case derives a contradiction, implying our assumption must be false. Therefore, the message does not visit any repeating sequence of vertices. Since the graph is finite, the message must eventually reach node t. □

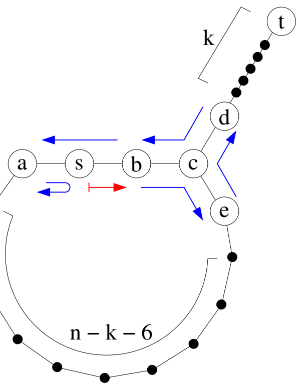

Fig. 13 The red arrow denotes the neighbour to which the message is initially forwarded from s. Blue arrows denote subsequent forwarding decisions as a function of the neighbour from which the message was received. Algorithm 1 forwards the message from the origin node s clockwise around the cycle back to node s. The message is then forwarded counter-clockwise around the cycle back to node c before being forwarded along the path from c to the destination node t. This route has length $2n - k - 3$, whereas the shortest path has length $k + 3$.

Lemma 8 *Algorithm 1 has dilation at most 7. Furthermore, for any $\epsilon > 0$, there exists a network on which Algorithm 1 has dilation at least $7 - \epsilon$.*

Proof Lemma 7 shows that every message eventually reaches its destination. By Observation 1, a message can traverse each edge at most twice, once in each direction. Nodes at which Case 1 of Algorithm 1 applies are visited at most once. By Corollary 3, the message travels only along consistent edges of G. We partition the consistent edges into those that forward the message exactly once and those that forward the message at most twice, and bound the cardinalities of these two sets.

Case 1. Suppose neither Case 2, 3, nor 4 of Algorithm 1 is applied. Therefore, only Case 1 is applied, the message follows a shortest path to its destination, and the dilation is 1.

Case 2. Suppose Cases 2, 3, or 4 of Algorithm 1 are applied. Therefore,

$$\operatorname{dist}(s,t) \geq k + 1. \tag{7}$$

In particular, Case 1 of Algorithm 1 is applied at the last $k + 1$ nodes, each of which is visited exactly once. These nodes form a path T of length k. That is,

$$|V(T)| = k + 1 \text{ and } |E(T)| = k. \tag{8}$$

Let Q denote the consistent subgraph of G induced by the remaining nodes at which Cases 2, 3, or 4 may be applied. Since $k \geq n/4$ and by (8),

$$|V(Q)| = n - |V(T)| \leq 4k - (k + 1) = 3k - 1. \tag{9}$$

By Lemmas 5 and 6, and by (9), it follows that Q contains at most one cycle. Therefore,

$$|E(Q)| \leq |V(Q)| \leq 3k - 1, \tag{10}$$

where each edge in $E(Q)$ is visited at most twice. Each edge in $E(T)$ is visited exactly once. One additional edge of G is followed once from Q to T. Therefore, the dilation, $S(k)$, is bounded by

$$S(k) \leq \frac{|E(T)| + 2|E(Q)| + 1}{\operatorname{dist}(s,t)} \leq \frac{k + 2(3k-1) + 1}{k+1} < 7, \tag{11}$$

by (7), (8), and (10).

We now show that (11) is tight. Choose any $\epsilon > 0$ and let $n \geq 96/\epsilon$. Let G denote the graph on n vertices illustrated in Figure 13 along with the corresponding local routing functions for nodes s and c (Rules S2 and U3, respectively). Observe that Rule U2 applies at all remaining nodes on the cycle and Case 1 of Algorithm 1 applies at all nodes on the path from node d to the destination node t. The route followed by Algorithm 1 has length $2n - k - 3$, whereas the shortest path has length $k+3$. Therefore, when $k = n/4$, Algorithm 1 has dilation

$$\frac{2n - n/4 - 3}{n/4 + 3} = 7 - \frac{96}{n + 12} > 7 - \epsilon,$$

showing that (11) is tight. □

Theorem 5 follows from Lemmas 7 and 8:

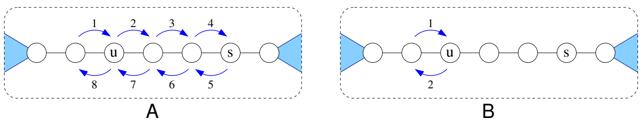

Fig. 14 (**A**) In the original definition of Algorithm 1, the repeated application of Rule U2 forwards the message right from u until it reaches s. Here, Rule S2 is applied, sending the message back to u. (**B**) Algorithm 1B reverses the direction of the message pre-emptively at node u (Rule U2c).

Theorem 5 *For every $k \geq n/4$, there exists an origin-aware, predecessor-aware, k-local routing algorithm that succeeds on all connected graphs while guaranteeing dilation at most 7.*

Reducing Dilation

Lemma 8 shows that Algorithm 1 cannot guarantee dilation less than $7 - \epsilon$ for any $\epsilon > 0$. As shown in Theorem 4, no $(n/4)$-local routing algorithm can guarantee dilation less than 5. In Appendix A we describe a refinement of Rule U2 that reduces the dilation of Algorithm 1 to at most 6. We refer to this new routing strategy as Algorithm 1B.

Informally, the modification to Rule U2 applies Rules S2 or US2 pre-emptively if the current node has sufficient information to determine that one of these two rules is about to be applied at a node in its local neighbourhood. That is, instead of forwarding the message in a given direction only to have the message reverse its direction a few hops away, the direction of the message is reversed immediately. See Figure 14.

We prove the following theorem in Appendix A:

Theorem 6 *For every $k \geq n/4$, there exists an origin-aware, predecessor-aware, k-local routing algorithm that succeeds on all connected graphs while guaranteeing dilation at most 6.*

5.2 Predecessor Aware and Origin Oblivious

Given any $k \geq n/3$, we describe a predecessor-aware, origin-oblivious, k-local routing algorithm that succeeds on all connected graphs on n vertices.

Routing Algorithm

The k-local preprocessing step of Algorithm 1 is applied to identify a set of routing edges in $G_k(u)$ which we denote by $G'_k(u)$. Since $3k + 1 > n$ and an active component contains at least k nodes, we get the following proposition, analogous to Proposition 1:

Proposition 2 *Every node has active degree at most 2.*

Forwarding decisions are determined by the following set of rules, similar to those defined in Algorithm 1. Let u denote the current node.

Algorithm 2: $(n/3)$-local, origin-oblivious, predecessor-aware routing algorithm

Case 1. Suppose $\text{dist}(u, t) \leq k$. That is, $t \in V(G_k(u))$. The algorithm forwards the message to any neighbour of u on a shortest path from u to t until the message arrives at t.

Case 2. Suppose $\text{dist}(u, t) > k$, $u = s$, and $v = \perp$. That is, the message is being sent from the origin for the first time. The algorithm forwards the message along any active edge of u.

Case 3. Suppose $\text{dist}(u, t) > k$ and $v \neq \perp$. Forwarding decisions are illustrated by Rules U1 and U2 in Figure 11. If the message arrived at u via a passive component of u, then the algorithm forwards the message along any active edge of u.

Correctness of Algorithm 2

We now prove the correctness of Algorithm 2 and derive a tight bound on the corresponding dilation. Observe that Lemmas 2, 3, 5, and 6, and Corollaries 3 and 4 apply directly to Algorithm 2. Furthermore, the following lemma follows by an argument analogous to the proof of Lemma 4:

Lemma 9 *If Rule U1 is applied at a node u to reverse the direction of a message, then $G'_k(u)$ has a passive component containing at least $k - 1$ nodes.*

Using arguments similar to the proofs of Lemmas 7 and 8, we now prove the correctness of Algorithm 2 and derive a tight bound on the corresponding dilation.

Lemma 10 *Given any connected graph G on n nodes, any $k \geq n/3$, and any $\{s, t\} \subseteq V(G)$, Algorithm 2 successfully delivers a message from node s to node t.*

Proof By Proposition 2, u has at most two active components. Consequently, one of Cases 1 through 3 of Algorithm 2 must apply at every node u.

Suppose Algorithm 2 is defeated by some graph G, for some $k \geq n/4$ and some origin-destination pair $(s, t) \in V(G) \times V(G)$. Since n is finite and Algorithm 2 is deterministic, the message must visit a repeating sequence of vertices and edges in G. Let R denote the corresponding repeating subgraph of G. Let $T = V(G) \setminus V(R)$ denote the set of remaining vertices of G, including all those that are never visited. By Lemma 3, there exists a consistent path in G from t to some vertex in $V(R)$. Let $\{q_R, q_T\}$ denote the last edge on such a path such that $q_R \in V(R)$ and $q_T \in T$. Since the message does not reach its destination t, Case 1 of Algorithm 2 never occurs. Consequently,

$$\forall v \in V(R),\ \operatorname{dist}(v,t) \geq k+1,$$

implying that q_T is an active neighbour of q_R. By Proposition 2, node q_R has at most two active neighbours, one of which is node q_T. Since the message is in a repeating sequence, Case 3 of Algorithm 2 applies. In particular, Rule U2 applies, and the message is forwarded to node q_T. Since $q_T \notin V(R)$, we derive a contradiction. Therefore, the message does not visit any repeating sequence of vertices. Since the graph is finite, the message must eventually reach node t. □

Lemma 11 *Algorithm 2 has dilation at most 3.*

Proof Lemma 10 shows that every message eventually reaches its destination. By Observation 1, a message can traverse each edge at most twice, once in each direction. Nodes at which Case 1 of Algorithm 2 applies are visited at most once. By Corollary 3, the message travels only along consistent edges of G. We partition the consistent edges into those that forward the message exactly once and those that forward the message at most twice, and bound the cardinalities of these two sets.

Case 1. Suppose neither Case 2 nor 3 of Algorithm 2 is applied. Therefore, only Case 1 is applied, the message follows a shortest path to its destination, and the dilation is 1.

Case 2. Suppose Cases 2 or 3 of Algorithm 2 are applied. Therefore,

$$\operatorname{dist}(s,t) \geq k+1. \tag{12}$$

In particular, Case 1 of Algorithm 2 is applied at the last $k+1$ nodes, each of which is visited exactly once. These nodes form a path T of length k. That is,

$$|V(T)| = k+1 \text{ and } |E(T)| = k. \tag{13}$$

Let Q denote the consistent subgraph of G induced by the remaining nodes at which Cases 2 or 3 may be applied. Since $k \geq n/3$ and by (13),

$$|V(Q)| = n - |V(T)| \leq 3k - (k+1) = 2k-1. \tag{14}$$

By Lemmas 5 and 6, and by (14), it follows that Q is acyclic. Therefore,

$$|E(Q)| < |V(Q)| \leq 2k-1. \tag{15}$$

Case 2a. Suppose Rule U1 is never applied. Thus, the message never changes direction. Since Q is acyclic, the message never visits any edge in $E(Q)$ more than once. One additional edge of G is followed once from Q to T. Each edge in $E(T)$ is visited exactly once. Therefore, the dilation, $S(k)$, is bounded by

$$S(k) \leq \frac{|E(T)| + |E(Q)| + 1}{\operatorname{dist}(s,t)} \leq \frac{3k}{k+1} < 3,$$

by (12), (13), and (15).

Case 2b. Suppose Rule U1 is applied at some node u. By Lemma 9 and (15), at least $k-1$ edges in $E(Q)$ are not visited. Consequently, at most k edges of $E(Q)$ are visited, each of which is visited at most twice. One additional edge of G is followed once from Q to T. Each edge in $E(T)$ is visited exactly once. Therefore, the dilation, $S(k)$, is bounded by

$$S(k) \leq \frac{3k+1}{k+1} < 3.$$

In all cases, we get that $S(k) < 3$. □

Theorem 7 follows from Lemmas 10 and 11:

Theorem 7 *For every $k \geq n/3$, there exists an origin-oblivious, predecessor-aware, k-local routing algorithm that succeeds on all connected graphs while guaranteeing dilation at most 3.*

As shown in Theorem 4, no $(n/3)$-local routing algorithm can guarantee dilation less than 3. Therefore, Algorithm 2 is optimal with respect to worst-case route length.

5.3 Predecessor Oblivious and Origin Oblivious

Given any $k \geq \lfloor n/2 \rfloor$, we describe a predecessor-oblivious origin-oblivious k-local routing algorithm that succeeds on all connected graphs on n vertices.

Since $2k+1 \geq n$ and an active component contains at least k nodes, we get the following proposition, analogous to Propositions 1 and 2:

Proposition 3 *Every node has active degree at most 2.*

We begin by establishing the following property which is used to show the correctness of Algorithm 3 in Lemma 13:

Lemma 12 *Given any connected graph G on n nodes, any $k \geq \lfloor n/2 \rfloor$, and any $\{u,t\} \subseteq V(G)$, either* $\operatorname{dist}(u,t) \leq k$ *or $G_k(u)$ has one constrained active component.*

Proof By Proposition 3, u has at most two active components.

Case 1. Suppose u has no active components. The entire network is contained in $G_k(u)$ and, therefore, $\text{dist}(u,t) \leq k$.

Case 2. Suppose u has two active components. Since $2k+1 \geq n$, the entire network is contained in $G_k(u)$ and, therefore, $\text{dist}(u,t) \leq k$.

Case 3. Suppose u has one unconstrained active component. Since every unconstrained active component contains at least $2k$ vertices and $2k+1 \geq n$, the entire network is contained in $G_k(u)$ and, therefore, $\text{dist}(u,t) \leq k$.

Consequently, if $\text{dist}(u,t) > k$, node u must have one constrained active component. □

Forwarding decisions are determined by the following set of rules. Let u denote the current node.

Algorithm 3: $(n/2)$-local, origin-oblivious, predecessor-oblivious routing algorithm

Case 1. Suppose $\text{dist}(u,t) \leq k$. That is, $t \in V(G_k(u))$. The algorithm forwards the message to any neighbour of u on a shortest path from u to t until the message arrives at t.

Case 2. Suppose $\text{dist}(u,t) > k$ and u has one constrained active component. Let v denote the constraint vertex in $G_k(u)$ that is furthest from u. The algorithm forwards the message to any neighbour of u that reduces the distance to v. This procedure continues until the algorithm enters Case 1.

Lemma 13 *Given any connected graph G on n nodes, any $k \geq \lfloor n/2 \rfloor$, and any $\{s,t\} \subseteq V(G)$, Algorithm 3 successfully delivers a message along a shortest path from s to t.*

Proof By Lemma 12, either Case 1 or Case 2 of Algorithm 3 must apply at every node u. In Case 1, the distance from the current node u to the destination node t, $\text{dist}(u,t)$, decreases by one unit each time the message is forwarded. In Case 2, observe that $\text{dist}(u,t) = \text{dist}(u,v) + \text{dist}(v,t)$. Therefore a decrease in $\text{dist}(u,v)$ implies an equal decrease in $\text{dist}(u,t)$. It follows that the route from s to t has length $\text{dist}(s,t)$. Therefore, Algorithm 3 finds a shortest path from s to t. □

Theorem 8 follows from Lemma 13:

Theorem 8 *For every $k \geq n/2$, there exists an origin-oblivious, predecessor-oblivious, k-local routing algorithm that succeeds on all connected graphs and finds a shortest path from the origin to the destination.*

An alternative solution to Algorithm 3 can be defined analogously to Algorithms 1 and 2. That is, when at a node u, the same preprocessing step could be applied to identify routing edges. Upon doing so, a node has active degree at most 3 in Algorithm 1 ($k \geq n/4$), active degree at most 2 in Algorithm 2 ($k \geq n/3$), and active degree at most 1 in Algorithm 3 ($k \geq n/2$). Ignoring predecessor- and origin-awareness/obliviousness, this trend suggests that the locality parameter k is inversely proportional to the number of possible forwarding decisions that a k-local routing algorithm must consider at each node.

5.4 Predecessor Oblivious and Origin Aware

An origin-aware, k-local routing algorithm does not require a greater locality parameter k than does an origin-oblivious k-local routing algorithm to guarantee delivery. In other words, providing knowledge of the origin cannot hinder an origin-oblivious routing algorithm. This gives the following corollary:

Corollary 5 *For every $k \geq n/2$, there exists an origin-aware, predecessor-oblivious, k-local routing algorithm that succeeds on all connected graphs and finds a shortest path from the origin to the destination.*

6 Discussion and Directions for Future Research

6.1 Optimal Dilation

Algorithms 1 is a predecessor-aware, origin-aware, $(n/4)$-local routing algorithm. Lemma 8 shows a tight bound of 7 on the dilation of Algorithm 1. In Appendix A we describe how to modify Algorithm 1 to reduce its dilation: Lemma 16 shows a tight bound of 6 on the dilation of the modified Algorithm 1. Theorem 4 implies a lower bound of 5 on the worst-case dilation of any $(n/4)$-local routing algorithm. Determining the exact bound on the dilation possible for any $(n/4)$-local routing algorithm within the interval $[5,6]$ remains open. One possible strategy for further reducing the dilation of Algorithm 1 from 6 to 5 is to modify the preprocessing step such that dormant edges on local cycles in $G_k(s)$ are selected to maximize the active degree of nodes in $G_k(s)$.

6.2 Directed Graphs

The results of this paper concern k-local routing on undirected graphs. Of course, the analogous questions can be posed in the setting of directed graphs, many of which remain open. Preliminary investigations of local routing on directed graphs have been made by Chávez et al. [8] who describe 1-local routing algorithms for Eulerian graphs and outerplanar graphs and Fraser et al. [19] who show that every 1-local routing algorithm requires $\Omega(n)$ bits of memory on directed graphs (i.e., no stateless 1-local routing algorithm exists).

6.3 Using Additional Memory

Relaxing constraints and allocating additional memory the message overhead to store state information allows more general solutions to the local routing problem. Braverman [6] shows that there exists a position-oblivious 1-local routing algorithm using $\Theta(\log n)$ state bits that succeeds on all graphs. An interesting open question is to determine whether there is a corresponding lower bound. Alternatively, does there exist a position-oblivious, origin-aware, predecessor-aware, k-local routing algorithm with $o(\log n)$ state bits for $k \in O(1)$? In general, can we identify tight bounds on memory requirements for deterministic k-local routing under various models?

Acknowledgements

The authors wish to thank Therese Biedl who observed that our origin-oblivious and predecessor-oblivious $(n/2)$-local routing algorithm (Algorithm 3) identifies a shortest path. Also, the authors thank the anonymous reviewers for their helpful suggestions.

References

1. P. Bose, A. Brodnik, S. Carlsson, E. D. Demaine, R. Fleischer, A. López-Ortiz, P. Morin, and I. Munro. Online routing in convex subdivisions. *International Journal of Computational Geometry and Applications*, 12(4):283–295, 2002.
2. P. Bose, P. Carmi, and S. Durocher. Bounding the locality of distributed routing algorithms. In *Proceedings of the ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing (PODC)*, volume 28, pages 250–259. ACM, 2009.
3. P. Bose and P. Morin. Competitive online routing in geometric graphs. *Theoretical Computer Science*, 324:273–288, 2004.
4. P. Bose and P. Morin. Online routing in triangulations. *SIAM Journal on Computing*, 33(4):937–951, 2004.
5. P. Bose, P. Morin, I. Stojmenović, and J. Urrutia. Routing with guaranteed delivery in ad hoc wireless networks. *Wireless Networks*, 7(6):609–616, 2001.
6. M. Braverman. On ad hoc routing with guaranteed delivery. In *Proceedings of the ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing (PODC)*, volume 27, page 418. ACM, 2008.
7. E. Chávez, S. Dobrev, E. Kranakis, J. Opatrny, L. Stacho, and J. Urrutia. Local construction of planar spanners with irregular transmission ranges. In *Proceedings of the Latin American Symposium on Theoretical Informatics (LATIN)*, volume 3887 of *Lecture Notes in Computer Science*, pages 286–297. Springer, 2006.
8. E. Chávez, S. Dobrev, E. Kranakis, J. Opatrny, L. Stacho, and J. Urrutia. Route discovery with constant memory in oriented planar geometric networks. *Networks*, 48(1):7–15, 2006.
9. D. Chen, L. Devroye, V. Dujmović, and P. Morin. Memoryless routing in convex subdivisions: Random walks are optimal. In *Proceedings of the European Workshop on Computational Geometry (EuroCG)*, pages 109–112, 2010.
10. S. Durocher, D. Kirkpatrick, and L. Narayanan. On routing with guaranteed delivery in three-dimensional ad hoc wireless networks. *Wireless Networks*, 16:227–235, 2010.
11. S. Durocher, E. Kranakis, D. Krizanc, and L. Narayanan. Balancing traffic load using one-turn rectilinear routing. *Journal of Interconnection Networks*, 10(1–2):93–120, 2009.
12. G. G. Finn. Routing and addressing problems in large metropolitan-scale internetworks. Technical Report ISI/RR-87-180, Information Sciences Institute, 1987.
13. R. Flury and R. Wattenhofer. Randomized 3D geographic routing. In *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*, pages 834–842. IEEE, 2008.
14. P. Fraigniaud and C. Gavoille. Local memory requirement of universal routing schemes. In *Proceedings of the ACM SIGACT-SIGOPS Symposium on Parallel Algorithms and Architecture (SPAA)*, pages 183–188. ACM, 1996.
15. P. Fraigniaud and C. Gavoille. Universal routing schemes. *Distributed Computing*, 10:65–78, 1997.
16. P. Fraigniaud and C. Gavoille. Interval routing schemes. *Algorithmica*, 21(2):155–182, 1998.
17. P. Fraigniaud and C. Gavoille. A theoretical model for routing complexity. In *Proceedings of the International Colloquium on Structural Information and Communication Complexity (SIROCCO)*, pages 98–113. Carleton Scientific, 1998.
18. M. Fraser. Local routing on tori. *Adhoc and Sensor Wireless Networks*, 6:179–196, 2008.
19. M. Fraser, E. Kranakis, and J. Urrutia. Memory requirements for local geometric routing and traversal in digraphs. In *Proceedings of the Canadian Conference on Computational Geometry (CCCG)*, volume 20, 2008.
20. X. Guan. *Face Routing in Wireless Ad-Hoc Networks.* PhD thesis, University of Toronto, 2009.
21. E. Kranakis, H. Singh, and J. Urrutia. Compass routing on geometric networks. In *Proceedings of the Canadian Conference on Computational Geometry (CCCG)*, volume 11, pages 51–54, 1999.
22. F. Kuhn, R. Wattenhofer, and A. Zollinger. Ad-hoc networks beyond unit disk graphs. In *Joint Workshop on Foundations of Mobile Computing*, pages 69–78. ACM, 2003.
23. X.-Y. Li, Y. Wang, and W.-Z. Song. Applications of k-local MST for topology control and broadcasting in wireless ad hoc networks. *IEEE Transactions on Parallel and Distributed Systems*, 15(12):1057–1069, 2004.
24. D. Peleg and E. Upfal. A trade-off between space and efficiency for routing tables. *Journal of the ACM*, 36(3):510–530, 1989.
25. N. Santoro and R. Khatib. Labelling and implicit routing in networks. *The Computer Journal*, 29(1):5–8, 1985.
26. I. Stojmenović. Position based routing in ad hoc networks. *IEEE Communications Magazine*, 40(7):128–134, 2002.

A Reducing Dilation

We describe a refinement of Rule U2 that reduces the dilation of Algorithm 1 to at most 6. We refer to this new routing strategy as Algorithm 1B.

Modified Routing Algorithm

The new rule is defined formally as follows, replacing Rule U2 of Algorithm 1. Let u denote the current node. Recall that Rule U2 applies if u has two active components, $\mathrm{dist}(u,t) > k$, $u \neq s$, and either $s \notin G'_k(u)$ or s is in an active component of u. Let $\mathrm{dist}'(\cdot,\cdot)$ denote graph distance in $G'_k(u)$ ($\mathrm{dist}(\cdot,\cdot)$ still denotes graph distance in G).

Case U2a. Suppose $s \notin V(G'_k(u))$ or $\mathrm{dist}'(u,s) = k$. Rule U2 remains unchanged; forwarding decisions are illustrated in Figure 11.

In the remaining cases, U2b–U2f, u has two active components, $\mathrm{dist}'(u,s) < k$, $u \neq s$, $\mathrm{dist}(u,t) > k$, and s is in an active component of u.

Case U2b/c. Suppose the origin node s is a constraint vertex in $G'_k(u)$. Suppose furthermore that s has neighbours c and d in $G'_k(u)$ that are also constraint vertices in $G'_k(u)$. Without loss of generality, assume $\mathrm{dist}'(u,c) = \mathrm{dist}'(u,s) + 1$ and $\mathrm{dist}'(u,d) = \mathrm{dist}'(u,s) - 1$. Note, if $\mathrm{dist}'(u,s) = 1$ then $d = u$. If $\mathrm{rank}(c) > \mathrm{rank}(d)$ (Case U2b), then forwarding decisions are illustrated in Figure 15B. Otherwise (Case U2c), forwarding decisions are illustrated in Figure 15C. In Case U2c, u can determine the imminent application of Rule S2 and applies this rule pre-emptively.

Case U2d/e. Suppose the origin node s is in a passive component of some constraint vertex e in $G'_k(u)$. Suppose furthermore that e has neighbours c and d in $G'_k(u)$ that are also constraint vertices in $G'_k(u)$. Without loss of generality, assume $\mathrm{dist}'(u,c) = \mathrm{dist}'(u,e)+1$ and $\mathrm{dist}'(u,d) = \mathrm{dist}'(u,e) - 1$. Note, if $\mathrm{dist}'(u,e) = 1$ then $d = u$. If $\mathrm{rank}(c) > \mathrm{rank}(d)$ (Case U2d), then forwarding decisions are illustrated in Figure 16D. Otherwise (Case U2e), forwarding decisions are illustrated in Figure 16E. In Case U2e, u can determine the imminent application of Rule US2 and applies this rule pre-emptively.

Case U2f. Suppose none of Cases U2b to U2e apply. Rule U2 remains unchanged; forwarding decisions are illustrated in Figure 11.

Correctness of Algorithm 1B

We now prove the correctness of Algorithm 1B and derive a tight bound on the corresponding dilation.

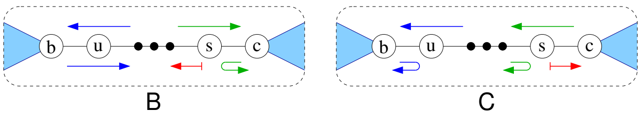

Fig. 15 Algorithm 1B, Cases U2b/c: The message is at node u. Light blue regions denote the portion of each active component of u that extends to the boundary of $G_k(u)$. Passive components are not illustrated. Blue arrows denote forwarding decisions at u as a function of the neighbour from which the message is received. Green and red arrows denote anticipated forwarding decisions at s (Rule S2) as a function of the neighbour from which s receives the message.

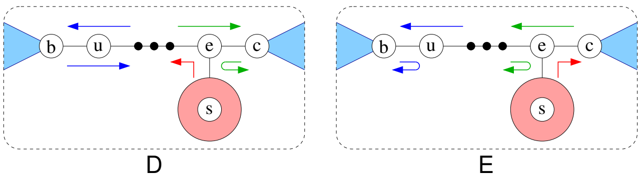

Fig. 16 Algorithm 1B, Cases U2d/e: The message is at node u. Light blue regions denote the portion of each active component of u that extends to the boundary of $G_k(u)$. The light red region denotes the passive component of e that contains s. Other passive components are not illustrated. Blue arrows denote forwarding decisions at u as a function of the neighbour from which the message is received. Green and red arrows denote anticipated forwarding decisions at e (Rule US2) as a function of the neighbour from which e receives the message.

Given any connected graph G on n nodes, any $k \geq n/4$, and any $\{s,t\} \subseteq V(G)$, let $\{\sigma_i\}$ and $\{\sigma'_i\}$ denote the sequences of edges in $E(G)$ visited by Algorithms 1 and 1B, respectively, on a route from s to t. For any $x \geq 1$, let $\{a_i\}_x$ denote the first j edges in the sequence $\{a_i\}$, where $j = \min(x, |\{a_i\}|)$.

Lemma 14 *For any $x \geq 1$, $\{\sigma'_i\}_x$ is a subsequence of $\{\sigma_i\}$.*

Proof We use induction on x. When the message is first forwarded from the origin node s, both routing algorithms apply the identical Rules S1, S2, or S3. Therefore, $\sigma_1 = \sigma'_1$ and the claim holds when $x = 1$. Choose any $y \geq 1$ and assume the claim holds for $x = y$. We show the claim holds for $x = y + 1$.

Case 1. Suppose $\{\sigma'_i\}_y = \{\sigma'_i\}_{y+1}$ (i.e., $\{\sigma'_i\}$ contains at most y elements). The claim holds for $x = y + 1$ by the inductive hypothesis.

Case 2. Suppose $\{\sigma'_i\}_y \neq \{\sigma'_i\}_{y+1}$. Let u, v, and b denote nodes in $V(G)$ such that $\sigma'_y = (v,u)$ and[1] $\sigma'_{y+1} = (u,b)$. That is, u denotes the current node, having received the message from node v, and b denotes the node to which Algorithm 1B forwards the message after u. By the inductive hypothesis, $(v,u) \in \{\sigma_i\}$. That is, Algorithm 1 forwards the message from v to u. We show

[1] Although graph edges are undirected, ordered pairs are used here to denote the direction in which a message is forwarded across an edge: $(\mathit{sender}, \mathit{receiver})$.

that Algorithm 1 eventually forwards the message from u to b.

Consider the forwarding rule applied by each algorithm at node u.

CASE 2A. Suppose both algorithms make identical forwarding decisions, sending the message from u to b. Thus, (u, b) must follow (v, u) in both sequences and the claim holds for $x = y + 1$.

The algorithms differ only in Rules U2c and U2e. Furthermore, the forwarding decisions made by each algorithm differ only if the message was received from node b (i.e., $v = b$), where b denotes the active neighbour of u opposite s (see Figures 15C and 16E).

CASE 2B. Suppose Algorithm 1B applies Rule U2c and $v = b$. Let a denote the active neighbour of s opposite b. Let node c denote the active neighbour of s opposite a. See Figure 15C. Algorithm 1 applies Rule U2 and forwards the message from u to a whereas Algorithm 1B reverses the direction of the message, sending it from u back to b. Since c is a constraint vertex in $G'_k(u)$, no node on the path P from b to c can have an active edge outside P. In particular, Algorithm 1 can only forward the message toward s or reverse its direction back toward u. If Algorithm 1 forwards the message all the way to s, then Rules S1 or S2 apply. Otherwise, Algorithm 1 applies Rule U1 before the message reaches s. In either case, Algorithm 1 sends the message back to u and then to b.

Observe that no rule other than S1, S2, U1, or U2 can be applied by Algorithm 1 when forwarding the message along path P. In particular, because the message started at node s, $\text{dist}(p, t) > k$ for all nodes $p \in V(P)$; otherwise, Case 1 of Algorithms 1 and 1B would have applied initially at node s, and Rule U2c could not have been applied by Algorithm 1B at node u.

Consequently, (u, b) must appear after (v, u) in sequence $\{\sigma_i\}$. Therefore, by the inductive hypothesis, the claim holds for $x = y + 1$.

CASE 2C. Suppose Algorithm 1B applies Rule U2e and $v = b$. The claim holds for $x = y + 1$ by an argument analogous to Case 2b. □

Lemma 15 *Given any connected graph G on n nodes, any $k \geq n/4$, and any $\{s, t\} \subseteq V(G)$, Algorithm 1B successfully delivers a message from node s to node t.*

Proof Algorithms 1 and 1B each terminate only once the message reaches t. Therefore, it suffices to show that $\{\sigma'_i\}$ is finite. By Lemma 7, Algorithm 1 guarantees delivery. In particular, $\{\sigma_i\}$ is finite. By Lemma 14, therefore $\{\sigma'_i\}$ is also finite. □

Lemma 16 *Algorithm 1B has dilation at most 6. Furthermore, for any $\epsilon > 0$, there exists a network on which Algorithm 1B has dilation at least $6 - \epsilon$.*

Proof Lemma 15 shows that every message eventually reaches its destination. Observe that Lemmas 4–6 and Corollary 3 apply directly to Algorithm 1B. By Corollary 3, the message travels only along consistent edges of G.

CASE 1. Suppose neither Case 2, 3, nor 4 of Algorithm 1B is applied. Therefore, only Case 1 is applied, the message follows a shortest path to its destination, and the dilation is 1.

CASE 2. Suppose Cases 2, 3, or 4 of Algorithm 1B are applied. Therefore,

$$\text{dist}(s, t) \geq k + 1. \tag{16}$$

In particular, Case 1 of Algorithm 1B is applied at the last $k + 1$ nodes, each of which is visited exactly once. These nodes form a path T of length k. That is,

$$|V(T)| = k + 1 \text{ and } |E(T)| = k. \tag{17}$$

Let Q denote the consistent subgraph of G induced by the remaining nodes at which Cases 2, 3, or 4 may be applied. By Observation 1, a message can traverse each edge in Q at most twice, once in each direction. Since $k \geq n/4$ and by (17),

$$|V(Q)| = n - |V(T)| \leq 4k - (k + 1) = 3k - 1. \tag{18}$$

By Lemmas 5 and 6, and by (18), it follows that Q contains at most one cycle. Therefore,

$$|E(Q)| \leq |V(Q)| \leq 3k - 1. \tag{19}$$

CASE 2A. Suppose Q is a tree and the message does not reverse direction. Therefore, each vertex in $V(G)$ is visited at most once. Since $k \geq n/4$ and by (16), the dilation, $S(k)$, is bounded by

$$S(k) \leq \frac{|V(G) - 1|}{\text{dist}(s, t)} \leq \frac{n - 1}{k + 1} \leq \frac{n - 1}{n/4 + 1} < 4.$$

CASE 2B. Suppose Q is a tree and the message reverses direction at least once. If a reversal results from Rules S2, US2, U2c, or U2e, then edge (u, b) will have been traversed twice in the same direction (see Figures 10, 12 15, and 16); by Observation 1, the message will never reach t, contradicting Lemma 15. Therefore, a reversal must result from Rules S1, U1, or US1. By Lemma 4, at least $k - 1$ edges in $E(Q)$ are not visited. Let B denote this set of edges. Thus,

$$|B| \geq k - 1. \tag{20}$$

The remaining edges in $E(Q) \setminus B$ are visited at most twice. One additional edge of G is followed once from Q to T. Therefore, the dilation, $S(k)$, is bounded by

$$S(k) \leq \frac{|E(T)| + 2(|E(Q)| - |B|) + 1}{\text{dist}(s,t)} \leq \frac{5k-3}{k+1} < 5,$$

by (16), (17), (19), and (20).

Case 2c. Suppose Q contains a cycle C and the message never reverses direction. By Lemma 5,

$$|E(C)| \geq 2k + 1. \tag{21}$$

Let $B = E(Q) \setminus E(C)$. By (19) and (21),

$$|B| \leq k - 2. \tag{22}$$

Consequently, every edge in $E(C)$ is visited at most once and every edge in B is visited at most twice. One additional edge of G is followed once from Q to T. Therefore, the dilation, $S(k)$, is bounded by

$$\begin{aligned} S(k) &\leq \frac{|E(T)| + 2|B| + |E(C)| + 1}{\text{dist}(s,t)} \\ &= \frac{|E(T)| + |B| + |E(Q)| + 1}{\text{dist}(s,t)} \\ &\leq \frac{5k-2}{k+1} \\ &< 5, \end{aligned}$$

by (16), (17), (19), and (22).

Case 2d. Suppose Q contains a cycle C and the message reverses direction at least once. Let $B = E(Q) \setminus E(C)$. Again, (21) and (22) hold as in Case 2c. By (22) and Lemma 4, a reversal cannot result from Rules S1, U1, or US1 of Algorithm 1B. Furthermore, (22) implies that a reversal cannot result from Rules S2 or US2, since Rules U2c or U2e would have been applied earlier. Thus, the reversal must result from Rules U2c or U2e. By Observation 1, at most one reversal can occur.

Let u denote the node at which Rule U2c or U2e is applied. If Rule U2c is applied, then let nodes be labelled as in Figure 15C and let I denote the path from u to s in $G'_k(u)$. If Rule U2e is applied, then let nodes be labelled as in Figure 16E and let I denote the path from u to e in $G'_k(u)$. The message is never forwarded along any edge in $E(I)$. If $|E(I)| < k - 1$, then some passive component D adjacent to I in $G'_k(u)$ must have appeared active in $G'_k(b)$ (otherwise, Rules U2c or U2e would have applied at node b). Consequently,

$$|E(I)| + |E(D)| \geq k - 1 \tag{23}$$

Furthermore, since $D \subseteq B$ and by (22),

$$|E(D)| \leq |E(B)| \leq k - 2. \tag{24}$$

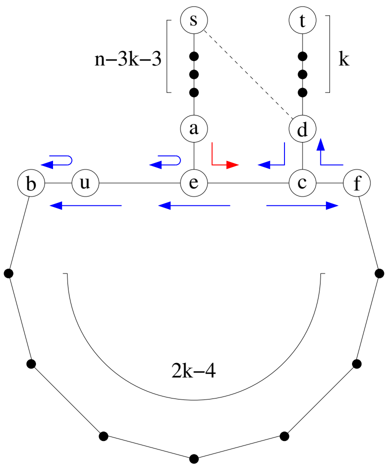

Fig. 17 Solid line segments denote routing edges and the dotted line segment denotes a dormant edge. The red arrow denotes the neighbour to which the message is initially forwarded from the passive component containing s. Blue arrows denote subsequent forwarding decisions as a function of the neighbour from which the message is received. When $k = n/4$, Algorithm 1B forwards the message from the origin node s along the path to node e, and then clockwise around the cycle to node u. Node u applies Rule U2e, sending the message counter-clockwise around the cycle back to node c before being forwarded along the path from c to the destination node t. This route has length $n + 2k - 6$, whereas the shortest path (via the dormant edge $\{s,d\}$) has length $k+1$.

Thus, each edge in Q is traversed at most twice. However, not all edges in Q are traversed twice: edges in I are never traversed and edges in D are traversed at most once. One additional edge of G is followed once from Q to T. Therefore, the dilation, $S(k)$, is bounded by

$$\begin{aligned} S(k) &\leq \frac{2(|E(Q)| - |E(I)| - |E(D)|) + |E(D)| + |E(T)| + 1}{\text{dist}(s,t)} \\ &\leq \frac{2[(3k-1) - (k-1)] + (k-2) + k + 1}{k+1} \\ &\leq \frac{6k-1}{k+1} \\ &< 6, \end{aligned} \tag{25}$$

by (16), (17), (19), (23), and (24).

In all cases, we get that $S(k) < 6$.

We now show that (25) is tight. Choose any $\epsilon > 0$ and let $n \geq 48/\epsilon$. Let G denote the graph on n vertices illustrated in Figure 17 along with the corresponding local routing functions for nodes u, e, and c (Rules U2e, US2 and U3, respectively). Let edge $\{s,d\}$ have minimum rank. Node s lies on a cycle of length $n - 3k + 1$. Since $k \geq n/4$, this cycle is local and edge $\{s,d\}$ is identified as dormant. Rule S1 applies at node s. Rule US1 applies at all nodes on the path from s to a. Case 1 of Algorithm 1B applies at all nodes on the path from node d to the destination node t. Rule U2 applies at all remaining nodes other than u, e, and c. The route followed by Algorithm 1B has length $n + 2k - 6$, whereas the shortest path has length $k + 1$. Therefore, when $k = n/4$, Algorithm 1B has dilation

$$\frac{n + 2(n/4) - 6}{n/4 + 1} = 6 - \frac{48}{n + 4} > 6 - \epsilon,$$

showing that (25) is tight. □

Theorem 6 follows from Lemmas 15 and 16:

Theorem 6 *For every $k \geq n/4$, there exists an origin-aware, predecessor-aware, k-local routing algorithm that succeeds on all connected graphs while guaranteeing dilation at most 6.*