Available online at www.sciencedirect.com
ScienceDirect
Discrete Applied Mathematics 155 (2007) 598–610
DISCRETE APPLIED MATHEMATICS
www.elsevier.com/locate/dam

Average stretch analysis of compact routing schemes

Tamar Eilam[a], Cyril Gavoille[b], David Peleg[c,1]

[a]*IBM T.J. Watson Research Center, 19 Skyline Drive, Hawthorne, NY 10532, USA*
[b]*LaBRI, Université Bordeaux I, 351, cours de la Libération, 33405 Talence Cedex, France*
[c]*Department of Computer Science and Applied Mathematics, The Weizmann Institute of Science, Rehovot 76100, Israel*

Received 3 March 2004; received in revised form 15 August 2006; accepted 12 September 2006
Available online 25 October 2006

Abstract

This paper presents some analytic results concerning the *pivot interval routing* (*PIR*) strategy of [T. Eilam, C. Gavoille, D. Peleg, Compact routing schemes with low stretch factor, J. Algorithms 46(2) (2003) 97–114, Preliminary version appeared. in: Proceedings of the 17th ACM Symposium on Principles of Distributed Computing, June 1998, pp. 11–20.] That strategy allows message routing on every weighted n-node network along paths whose *stretch* (namely, the ratio between their length and the distance between their endpoints) is at most five, and whose average stretch is at most three, with routing tables of size $O(\sqrt{n}\log^{3/2} n)$ bits per node. In addition, the route lengths are at most $2D$ ($\lceil 1.5D \rceil$ for uniform weights) where D is the weighted diameter of the network. The PIR strategy can be constructed in polynomial time and can be implemented so that the generated scheme is in the form of an *interval routing scheme* (*IRS*), using at most $O(\sqrt{n \log n})$ intervals per link. Here, it is shown that there exists an unweighted n-node graph G and an identity assignment ID for its nodes such that for every $R \in \mathrm{PIR}$ on G with a set of pivots computed by a greedy cover algorithm (respectively, a randomized algorithm), $\mathrm{AvStr}_G(R) > 3 - o(1)$ (respectively, with high probability). Also, it is shown that for almost every unweighted n-node graph G, and for every $R \in \mathrm{PIR}$ on G, $\mathrm{AvStr}_G(R) = 1.875 \pm o(1)$. A comparison between PIR and HCP_k, the hierarchical routing strategy presented in [B. Awerbuch, A. Bar-Noy, N. Linial, D. Peleg, Improved routing strategies with succinct tables, J. Algorithms 11 (1990) 307–341.] is also given.

Keywords: Compact routing schemes; Interval routing schemes; Pivot interval routing; Stretch factor; Average stretch

1. Introduction

1.1. Background

The efficiency of routing schemes is commonly studied in terms of two conflicting parameters: the amount of *memory* kept in each node for routing purposes and the *stretch factor* of the routing scheme, namely, the maximum ratio between the length of the path traversed by a message and that of the shortest path between its source and destination. A series of papers (e.g., [2–5,24]) established the existence of a tradeoff between the memory requirements of a routing scheme and its worst-case stretch factor. In [24] it was shown that any routing strategy achieving stretch factor $s \geqslant 1$ must use a total of $\Omega(n^{1+1/(2s+4)})$ bits of routing information in the network. Moreover, any routing scheme

[1] Supported in part by grants from the Israel Science Foundation and from the Israel Ministry of Science and Art.
E-mail addresses: eilamt@us.ibm.com (T. Eilam), gavoille@labri.u-bordeaux.fr (C. Gavoille), david.peleg@weizmann.ac.il (D. Peleg).

0166-218X/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.dam.2006.09.010

with stretch factor $s < 3$ must use a total of $\Omega(n^2)$ bits [11,17], and an optimal bound of $\Theta(n^2 \log n)$ bits holds for stretch $s < 1.4$ including the shortest paths case $s = 1$ (cf. [21]). Conversely, a number of routing strategies proposed in the literature achieve almost optimal efficiency–space tradeoffs. Specifically, in [24] it was shown that for every graph and every integer $k \geqslant 1$ it is possible to construct a hierarchical routing scheme with stretch factor $O(k)$ which uses a total of $O(k^3 n^{1+1/k} \log n)$ bits of memory and names each node with an $O(\log^2 n)$ bit label. The hierarchical routing scheme presented in [2] uses $O(kn^{1/k} \log n)$ bits of memory per node and guarantees a stretch factor $O(k^2 9^k)$. The scheme presented in [4] guarantees, for every $k \geqslant 1$, a stretch factor of $O(k^2)$, while using $O(kn^{1/k} \log^2 n \log D)$ memory bits per node, where D is the weighted diameter of the network.

The major disadvantage of all the proposed hierarchical routing strategies is that they are rather complex, and thus are impractical especially for high-speed networks for which the latency in each node must be very short.

Subsequently, considerable attention has been given recently to an opposing design philosophy, focusing on *simple* and *uniform* compact routing strategies. Many compact routing strategies were proposed in the last decade (see, e.g., [10,12–14,25,27]).

The most popular such scheme is the *interval routing scheme* (*IRS*). It is presented in [25,27] and implemented in the T9000 transputer router chip of INMOS. The idea of this scheme is to label nodes with unique integers from $\{1, \ldots, n\}$, and to label the outgoing arcs in every node with a set of destination labels in the form of a set of consecutive intervals of the name segment. The collection of sets that label the outgoing arcs of a node forms a partition of the name segment. When invoking the delivery protocol, a message is sent on the unique outgoing arc-labeled by a set that contains the destination label. While the preprocessing stage of such a routing scheme (which is performed once in the initialization of the network) might be complex, the delivery protocol consists of simple decision functions in every node that depends only on the destination.

One of the desirable goals in interval routing is to minimize the maximum number of intervals that label an arc. Unfortunately, while many lower bounds are known for this problem (see, e.g., [9,15,18,22,26]), few tradeoff results between efficiency and space are known for any of these strategies for general graphs. For interval routing, a universal strategy which is based on routing on a BFS-tree is presented in [25,27]. This scheme uses only one interval per edge, thus the memory in a node with degree d is $O(d \log n)$, but it implies no upper bound on the stretch factor (cf. an n-node ring). On the other extreme, it is shown in [20] that it is possible to generate for every network an IRS which uses at most $n/4 + o(n)$ intervals per edge and guarantees stretch factor $s = 1$. Lately, [22] showed that for every graph there exists an IRS under which every message traverses a path of length at most $\lceil 1.5D \rceil$, where D is the diameter of the network, and which labels every arc with at most $\sqrt{n \ln n} + O(1)$ intervals. While this result implies an upper bound on the *dilation*, i.e., the length of paths traversed by messages, it does not imply any non-trivial upper bound on the stretch factor. Moreover, the paper does not present any efficient (say, polynomial time) preprocessing algorithm for generating such an IRS for a given graph. For other results, a recent survey on IRS is presented in [16].

In an attempt to take all of the considerations discussed above into account, two polynomial time constructible routing strategies, termed *pivot interval routing* (*PIR*), were presented in [8]. The first one, PIR1, generates for every weighted graph with arbitrary link costs, a routing scheme with stretch factor $s \leqslant 5$ that uses $O(\sqrt{n} \log^{3/2} n)$ bits per node, and requires $O(\log n)$ latency (defined as the time required to extract the outgoing link on which a message is to be forwarded in a node). The average stretch factor of these schemes is $\bar{s} \leqslant 3$. Moreover, the PIR1 preprocessing algorithm actually generates for every graph an IRS which labels every arc with at most $\alpha\sqrt{n \log n}$ intervals, where $\alpha \approx 1.17$. The dilation guaranteed by the PIR1 strategy is $2D$, where D is the weighted diameter of the network. For the unweighted case, the slightly different routing strategy PIR2 (which also generates for every graph an IRS with still $\beta\sqrt{n \log n}$ intervals per arc, where $\beta \approx 2.00$) achieves the same memory requirements, stretch factor and average stretch factor as PIR1 but guarantees a better dilation bound of $\lceil 1.5D \rceil$. Thus, these routing strategies provide the first constructive method for designing for every graph an IRS with constant stretch factor and $o(n)$ intervals per arc.

As the constructed routing schemes are IRSs, they enjoy the following properties: they employ a simple decision function, depending only on the destination of the message, which is the only information coded in the header of the message. Consequently, nodes executing the delivery protocol do not have to keep information on passing messages in a local memory nor do they have to encode information by rewriting the header of the message. Also, the routing paths traversed by messages are loop free. These properties are an advantage over the hierarchical routing schemes in which headers of messages are rewritten, a message can bounce back to its originator, and the routing decision is complex and does not depend only on the destination of the message.

1.2. Our results

In this paper we supplement the construction methods of [8] with a number of analytic results concerning the quality of the schemes. We first prove that the bound of $\bar{s} \leqslant 3$ on the average stretch factor of the proposed routing strategies is tight. Nevertheless, we then show, using the family of random graphs $\mathscr{G}_{n,p}$ (with $p = \frac{1}{2}$), that for almost all graphs (specifically, all but a $1/n^3$ fraction of all the n-node graphs), the average stretch factor guaranteed by the PIR1 and PIR2 routing strategies is asymptotically equal to 1.875. Note that other efficiency–space tradeoffs have been proposed for random and Kolmogorov random graphs (see [7,19] for details). For instance, for optimal stretch $s = 1$, it is proposed in [19] a scheme with at most $n + O(\log^4 n)$ bits per node for almost all graphs (actually, an IRS with at most 2 intervals per link).

The PIR1 and PIR2 routing strategies are also compared with HCP_k, one of the hierarchical routing strategies proposed in [3]. It is shown that while the difference between the average stretch factor achieved by both strategies on every graph does not exceed $O(1)$, there are concrete examples in which our strategy is superior in terms of the average stretch factor.

The rest of the paper is organized as follows. In Section 2 we give a precise definition of routing schemes and efficiency measures, define interval routing and overview the covering techniques used for constructing the PIR schemes. In Section 3 we review the PIR routing strategies. In Section 4 we show that for every $R \in \mathrm{PIR}$ on the unweighted n-vertex complete graph K_n, $\mathrm{AvStr}_{K_n}(R) = 2 - o(1)$. We also show that there exists an unweighted n-node graph G and an identity assignment ID for its nodes such that for every $R \in \mathrm{PIR1}$ on G with a set of pivots computed by the greedy cover algorithm (respectively, the randomized algorithm), $\mathrm{AvStr}_G(R) > 3 - o(1)$ (respectively, with high probability). Finally, we show that for almost every unweighted n-node graph G, and for every $R \in \mathrm{PIR1}$ on G, $\mathrm{AvStr}_G(R) = 1.875 \pm o(1)$. A comparison between PIR and HCP_k, the hierarchical routing strategy presented in [3] is given in Section 5. In particular, it is shown that for every n-node graph, and for every two routing schemes $R_{\mathrm{PIR1}} \in \mathrm{PIR1}$ and $R_{\mathrm{CP}} \in \mathrm{CP}$ on G, $\mathrm{AvStr}_G(R_{\mathrm{PIR1}}) \leqslant \mathrm{AvStr}_G(R_{\mathrm{CP}}) + o(1)$. Also, letting T be the star $K_{1,n-1}$, and letting r be its root, we show that if $\mathrm{ID}(r)$ is not the lowest ID, then for all $R_{\mathrm{PIR1}} \in \mathrm{PIR1}$ and $R_{\mathrm{CP}} \in \mathrm{CP}$ on T with set of pivots computed by the random or the greedy cover algorithm, $\mathrm{AvStr}_T(R_{\mathrm{CP}}) > 2 - o(1)$ (with high probability in the randomized case), whereas $\mathrm{AvStr}_T(R_{\mathrm{PIR1}}) = 1$.

2. Model and definitions

2.1. Routing schemes and complexity measures

A point-to-point communication network is modeled as a symmetric, weighted, finite digraph $G = (V, E, \omega)$, $|V| = n$, where the set of nodes represent the processors of the network and every pair of two opposite arcs represents a bidirectional communication link. Every arc of the network $e \in E$ is associated with a non-negative weight $\omega(e)$ (i.e., its cost) defining a metric. We assume that for every two opposite arcs e_1 and e_2, $\omega(e_1) = \omega(e_2)$. In the special case of uniform unit weight links, we say that the graph is *unweighted*, and denote it simply by $G = (V, E)$. Graphs are connected and do not contain self-loops or multiple arcs. We assume that every node v is named with a unique identity integer $\mathrm{ID}(v)$. In what follows, we informally confuse between the node v and its name $\mathrm{ID}(v)$. Note that the identities induce a total order on the nodes, thus for every two nodes $u, v \in V$ either $u < v$ or $v < u$.

The length of a directed path in the graph is the sum of weights of its arcs. The *distance* $d_G(u, v)$ between two nodes $u, v \in V$ is the length of a shortest path connecting them. The *diameter* of the graph G is defined as $D = \max\{d_G(u, v) \mid u, v \in V\}$. For a node $v \in V$, let E_v denote its set of outgoing arcs, and denote its *degree* by $\deg(v) = |E_v|$.

A *routing scheme* R is a distributed algorithm whose role is to deliver messages between nodes of the network. The routing scheme consists of certain *distributed data structures* in the network, and a *delivery protocol*, which can be invoked in any node u with two parameters: a *routing label* $\mathscr{L}(v)$ of the destination node v, and the message's information field. The message is delivered to v via a sequence of transmissions determined uniquely by the distributed data structure. The length of the route traversed by a message from u to v in the graph G according to the routing scheme R is denoted by $\rho_R(u, v)$.

A *universal routing strategy* is a function that returns, for every n-node graph G, a routing scheme on G. It is implemented by a *preprocessing algorithm*, performed during setup time in order to construct the distributed data structures and the labels required for the routing scheme.

Let R be a routing scheme on an n-node graph G. Given a node u, the *memory requirement* of u, denoted by $\text{Memory}_G(R, u)$, is the smallest number of bits that are required in order to code R_u. Let us denote the *total memory requirements* of a routing scheme R on G by $\text{Memory}_G(R) = \sum_{u \in V} \text{Memory}_G(R, u)$. The *latency* of R in u, denoted by $\text{Latency}_G(R, u)$, is the time complexity of R_u per node in the standard $O(\log n)$-word RAM model. It corresponds to the time required to extract from R the outgoing link on which the message is forwarded in u. The maximum and average stretch factors of R are, respectively, defined as

$$\text{Stretch}_G(R) = \max_{u \neq v} \left\{ \frac{\rho_R(u, v)}{d_G(u, v)} \right\} \quad \text{and} \quad \text{AvStr}_G(R) = \frac{1}{n(n-1)} \sum_{u \neq v} \frac{\rho_R(u, v)}{d_G(u, v)}.$$

A routing scheme of stretch factor 1 is termed a *shortest path* routing scheme. The *dilation* of a routing scheme R is the maximal length of a path traversed by a message. Formally,

$$\text{Dilation}_G(R) = \max_{u \neq v} \left\{ \rho_R(u, v) \right\}.$$

2.2. Interval routing

An *IRS* R on G is a routing scheme consisting of a pair $(\mathscr{L}, \mathscr{I})$, generated in the preprocessing step, where $\mathscr{L}$ is a *node-labeling*, $\mathscr{L} : V \to \{1, \ldots, n\}$, and $\mathscr{I}$ is an *arc-labeling*, $\mathscr{I} : E \to 2^{\mathscr{L}(V)}$, that satisfy the following condition. For any node u, the collection of sets that label all the outgoing arcs of u forms a partition of the name range (possibly excluding u itself[2]). Formally, for every $u \in V$,

1. $\bigcup_{e \in E_u} \mathscr{I}(e) \cup \mathscr{L}(u) = \{1, \ldots, n\}$.
2. $\mathscr{I}(e_1) \cap \mathscr{I}(e_2) = \emptyset$ for every two distinct arcs $e_1, e_2 \in E_u$.

The delivery protocol is defined as follows. Given a destination node v, set the first header to be $h_1 = \mathscr{L}(v)$. Also, for every node u, input port q and header h, set $H_u(q, h) = h$ and $P_u(q, h) = p$ if and only if $h \in \mathscr{I}(u, v)$ and p is the output port number of the arc (u, v). Namely, the message is sent on the arc which is labeled by a set that contains the destination label. Note that we have the freedom to relabel the output port number of each arc in order to optimize the memory requirements of the scheme at u.

Given an integer n and a subset $I \subseteq \{1, \ldots, n\}$, define the *compactness of I w.r.t. n*, denoted by $c_n(I)$, as the smallest integer k such that I can be represented by the union of k intervals $[a, b]$ of consecutive integers from $\{1, \ldots, n\}$, with n and 1 being considered as consecutive (cyclically). The *compactness* of an IRS $R = (\mathscr{L}, \mathscr{I})$ on G, denoted by $\text{Comp}_G(R)$, is the maximum, over all arcs $e \in E$, of the compactness $c_n(\mathscr{I}(e))$ of the set $\mathscr{I}(e)$ labeling e. Intuitively, smaller compactness and degrees imply smaller routing tables. The interval routing strategy presented in [25], based on routing on a minimum spanning tree, has compactness 1 (for every graph) but unbounded stretch factor. Another example is the ring C_n which admits an IRS R of compactness 1 with stretch factor 1 [25]. Therefore, $\text{Memory}_{C_n}(R, u) = O(\log n)$ for every node u, as it suffices for u to store its label and the intervals assigned to the two arcs. Actually, every graph G supporting an IRS R of compactness k satisfies $\text{Memory}_G(R, u) = O(dk \log(n/k))$ where $d = \deg(u)$, using output port relabeling from $\{1, \ldots, d\}$ (cf. [16]).

2.3. Balls, neighborhoods and covers

Our routing scheme constructions are based on the notions of neighborhoods, balls and covers. For every node v we can order all the nodes of the graph w.r.t. v by increasing distance from v, breaking ties by increasing node identities. Formally, $x \prec_v y$ if and only if either $d_G(x, v) < d_G(y, v)$, or $d_G(x, v) = d_G(y, v)$ and $x < y$. The *t-ball* $\mathscr{B}_v(t)$ of v is the set of the first t nodes according to the node ordering $\prec_v$. The *r-neighborhood* of a node $v \in V$ is defined as $\Gamma(v, r) = \{u \in V \mid d_G(v, u) \leqslant r\}$. Hence, intuitively, a ball is a neighborhood defined by volume rather than by radius.

Following is a simple fact which holds for both neighborhoods and balls.

---

[2] A labeling excluding u from its arc-labels is termed *strict*. Although non-strict labeling may produce more compact schemes, in this paper we restrict our attention to strict labeling only.

Fact 2.1 (*Monotonicity*). *If $u \in \mathscr{B}_v(t)$ (respectively, $u \in \Gamma(v, r)$) then for every node x on a shortest path from v to u, $u \in \mathscr{B}_x(t)$ (respectively, $u \in \Gamma(x, r)$).*

Consider a collection $\mathscr{H}$ of subsets of size t of elements from a set $\mathscr{V}$. A set $P \subseteq \mathscr{V}$ is said to *cover* the collection $\mathscr{H}$ if for every $A \in \mathscr{H}$, $A \cap P \neq \emptyset$. We review two techniques presented in [3] for generating relatively small covers for a given collection of sets of equal size.

The first technique is by using a greedy algorithm that starts with $P = \emptyset$ and iteratively adds to the set P an element in $\mathscr{V}$ occurring in a maximum number of uncovered sets. The algorithm stops when P becomes a cover. The set P is termed a *greedy cover* for $\mathscr{H}$.

Lemma 2.2 (*Lovász [23]*). *Let P be a greedy cover for $\mathscr{H}$. Then $|P| < |\mathscr{V}|(\ln|\mathscr{H}| + 1)/t$.*

The second method is randomized, and takes each element of $\mathscr{V}$ to the set P with probability $(c \ln|\mathscr{H}|)/t$, for some constant $c > 1$.

Lemma 2.3 (*Awerbuch et al. [3]*). *Let P be the set constructed by the randomized cover algorithm under the assumptions that $|\mathscr{V}| \geqslant 2t$ and $\ln|\mathscr{H}| = o(t)$. Then with probability at least $1 - 1/|\mathscr{H}|^{c-1}$, P is a cover for $\mathscr{H}$ and $|P| \leqslant (2c|\mathscr{V}| \ln|\mathscr{H}|)/t$.*

For our needs, $|\mathscr{H}| = |\mathscr{V}| = n$. In this paper, we are interested in the r-neighborhoods of nodes only for the case $r = \lceil D/2 \rceil$, where D is the diameter of the graph, and only for unweighted graphs. In particular, we would like to use such neighborhoods in order to construct a small $\lceil D/2 \rceil$-dominating set for our graph.

For every node v_i, $1 \leqslant i \leqslant n$, let $W_i = \Gamma(v_i, \lceil D/2 \rceil)$. Note that a cover for the set family $\mathscr{W} = \{W_1, \ldots, W_n\}$ (i.e., a set $X \subseteq V$ whose intersection with each W_i is non-empty) is a $\lceil D/2 \rceil$-dominating set for the graph. Also, note that (since the weights on the graph arcs are uniform) $\mathscr{W}$ is an intersecting hypergraph, namely, the intersection $W_i \cap W_j$ is non-empty for every $i \neq j$, since otherwise $d_G(v_i, v_j) > D$. It is therefore easy to verify that $\mathscr{W}$ has a cover of cardinality $O(\sqrt{n \log n})$. Note that this cannot be deduced directly from Lemma 2.2, since $|W_i|$ is not necessary in $\Theta(\sqrt{n \log n})$. Nevertheless, we have the following.

Lemma 2.4. *For every n-node unweighted graph G there exists a $\lceil D/2 \rceil$-dominating set X of cardinality $|X| < \sqrt{n(1 + \ln n)}$. Moreover, this set can be constructed by a straightforward greedy algorithm.*

Proof. We rely on the following simple fact. Consider a step of the greedy algorithm, and let U be the collection of all the sets W_i that were not covered so far. Let $|U| = m$. Then we claim that there is a node $v \in V$ that occurs in at least $m\sqrt{\ln n/n}$ sets in U. To see this, let d_v denote the number of sets in U containing v, for every node v. Then,

$$\sum_v d_v = \sum_{W_i \in U} |W_i| > m\sqrt{n \ln n}.$$

Since there are at most $n - |X| \leqslant n$ nodes in the sets remaining in U, we get by averaging that there must be a node v for which $d_v \geqslant m\sqrt{n \ln n}/n$, as required.

We now claim that applying the greedy algorithm to $\mathscr{W}$ yields a $\lceil D/2 \rceil$-dominating set X for G of cardinality $|X| \leqslant \sqrt{n \ln n}$. To see that, denote by $f(i)$ the number of sets remaining in U after i nodes were picked into X by the greedy algorithm. We get that

$$f(i) \leqslant \begin{cases} n & i = 0, \\ f(i-1) - f(i-1)/\sqrt{n/\ln n} & \text{otherwise.} \end{cases}$$

This yields

$$f(i) \leqslant n\left(1 - \frac{1}{\sqrt{n/\ln n}}\right)^i,$$

and hence the set $\mathscr{W}$ is exhausted after no more than $\sqrt{n \ln n}$ elements have been added to X. □

3. The PIR strategy

We now briefly overview the PIR strategies of [8]. The first one, PIR1, generates a routing scheme for every weighted graph (Theorem 3.1) and the second one, PIR2, generates a routing scheme for unweighted graphs with improved dilation (Theorem 3.2). These strategies are reminiscent of (and borrow some ideas from) the *covering pivots* (*CP*) of [3]; a more thorough comparison is made in Section 5.

Theorem 3.1. *For every n-node weighted graph* $G = (V, E, \omega)$ *with weighted diameter* D *there exists an IRS* $R = (\mathscr{L}, \mathscr{I})$ *on* G *such that* $\mathrm{Memory}_G(R, u) = \mathrm{O}(\sqrt{n}\log^{3/2} n)$ *and* $\mathrm{Latency}_G(R, u) = \mathrm{O}(\log n)$, *for every* $u \in V$, $\mathrm{Stretch}_G(R) \leqslant 5$, $\mathrm{AvStr}_G(R) \leqslant 3$, $\mathrm{Dilation}_G(R) \leqslant 2D$ *and* $\mathrm{Comp}_G(R) \leqslant \alpha\sqrt{n \log n}$, *where* $\alpha \approx 1.17$. *Moreover*, R *can be constructed in polynomial time in* n.

Intuitively, the idea of the PIR1 strategy is first to find the collection of t-balls of all the nodes of the graph, then to cover this collection by a (comparatively small) set of nodes termed *pivots*, and finally to label the nodes and arcs of the graph in such a way that a message with source u and destination v will be routed as follows. If the destination v is in u's t-ball, then the message will traverse a shortest path from u to v. Otherwise, it will traverse (in the worst case) a shortest path to the pivot nearest to v, and then a shortest path from that pivot to v itself.

We present the preprocessing algorithm of the PIR1 strategy for any given ball size t and for any given cover P of the collection of t-balls of nodes in the graph. The value of t is determined later where it is also shown how to construct a cover P such that PIR1 will satisfy the properties in Theorem 3.1. We now give a formal description of the PIR1 strategy. The preprocessing algorithm consists of three parts. In the first part, some preliminary structures are constructed which are used later to define the node and arc-labeling functions $\mathscr{L}$ and $\mathscr{I}$. The delivery protocol of PIR1 is the usual delivery protocol for IRSs.

3.1. Strategy PIR1: preprocessing and delivery protocol

We consider a weighted graph $G = (V, E, \omega)$. We construct the IRS $R = (\mathscr{L}, \mathscr{I})$ as follows. The size of the balls is fixed to $t = \sqrt{n(1 + \ln n)/2}$.

The preprocessing algorithm starts with some preliminary constructions. Let P be a cover for the collection of t-balls of the nodes, $\{\mathscr{B}_v(t)\}_{v\in V}$. Let $\ell = |P|$. Assign to every node v its pivot $p(v) \in P$, where $p(v)$ is the nearest node to v in P (breaking ties by increasing node identities). For every pivot $p \in P$, let $S_p = \{v \mid p(v) = p\}$ be its set of "clients". For every pivot $p \in P$, construct a minimum BFS spanning tree T_p rooted at p and spanning the entire graph G, and let $\widehat{T}_p$ be the subtree of T_p induced by p and its set of clients S_p.

Next, the nodes are labeled as follows. Assume that $P = \{p_1, \ldots, p_\ell\}$. We start by labeling the nodes in S_{p_1}. The labeling is performed by traversing the tree $\widehat{T}_{p_1}$ (i.e., the subtree spanning S_{p_1}), and assigning the nodes of $\widehat{T}_{p_1}$ a DFS (preorder) numbering in sequential ascending order, starting from 1. In order to give an efficient implementation of the scheme with low memory and low latency, a DFS is imposed so that, at any node x of $\widehat{T}_{p_1}$, the children of x are visited in a non-decreasing order of their number of descendants in the subtree. Once all the nodes of S_{p_1} have been labeled, we continue by labeling the nodes in S_{p_2} in the same manner (traversing the tree $\widehat{T}_{p_2}$), starting from the integer $1 + |S_{p_1}|$. Then we label the nodes of $S_{p_3}, \ldots, S_{p_\ell}$ in the same way, provided the node-labeling $\mathscr{L}$.

The arcs are then labeled as follows. For every node $x \in V$, we label every arc $e \in E_x$ by a set of destinations $\mathscr{I}(e) \subseteq \{1, \ldots, n\}$ in three main steps. We start by fixing $\mathscr{I}(e) = \emptyset$ for every $e \in E_x$, and then we label the arcs E_x of every node x as follows. For every $A \subseteq V$, we define $\mathscr{L}(A) = \{\mathscr{L}(a) \mid a \in A\}$.

(1) If x is not a leaf in the tree $\widehat{T}_{p(x)}$, let $s_1, \ldots, s_j$ be its successors in $\widehat{T}_{p(x)}$ and let $\widehat{T}_{s_i}$ be the subtree of $\widehat{T}_{p(x)}$ rooted at s_i, for $1 \leqslant i \leqslant j$. Assign $\mathscr{I}(x, s_i) = \{\mathscr{L}(v) \mid v \in \widehat{T}_{s_i}\}$, for every $1 \leqslant i \leqslant j$. Define $L_1 = \bigcup_{1\leqslant i\leqslant j} \mathscr{I}(x, s_i)$.
(2) Define $L_2 = \mathscr{L}(\mathscr{B}_x(t)) \backslash L_1$. For every node $v \neq x$ such that $\mathscr{L}(v) \in L_2$, let $e \in E_x$ be an arc on a shortest path from x to v (if there is more than one such arc, choose one arbitrarily). Assign $\mathscr{I}(e) = \mathscr{I}(e) \cup \{\mathscr{L}(v)\}$.
(3) Let $L_3 = \mathscr{L}(V) \backslash (L_1 \cup L_2)$. For every $p \in P$, let $e_p \in E_x$ be the arc from x to its predecessor on the tree T_p. Assign $\mathscr{I}(e_p) = \mathscr{I}(e_p) \cup (\mathscr{L}(S_p) \cap L_3)$.

Finally, the delivery protocol operates as follows. In a node x, a message M with destination $v \neq x$ is sent on the unique arc $e \in E_x$, such that $\mathscr{L}(v) \in \mathscr{I}(e)$.

3.2. Strategy PIR2

Theorem 3.2. *For every n-node unweighted graph $G = (V, E)$ with diameter D there exists an IRS $R = (\mathscr{L}, \mathscr{I})$ on G such that* $\mathrm{Memory}_G(R, u) = O(\sqrt{n}\log^{3/2} n)$ *and* $\mathrm{Latency}_G(R, u) = O(\log n)$, for every $u \in V$, $\mathrm{Stretch}_G(R) \leqslant 5$, $\mathrm{AvStr}_G(R) \leqslant 3$, $\mathrm{Dilation}_G(R) \leqslant \lceil 1.5D \rceil$ *and* $\mathrm{Comp}_G(R) \leqslant \beta\sqrt{n\log n}$, *where* $\beta \approx 2.00$. *Moreover, R can be constructed in polynomial time in n.*

Strategy PIR2 is very similar to PIR1, except that the preprocessing algorithm constructs a pivot collection covering simultaneously the collection of t-balls and the collection of $\lceil D/2 \rceil$-neighborhoods around the nodes of G. Formally, the only necessary change is to replace the first step of the preprocessing algorithm of PIR1 by the following: let P_1 be a cover for the collection of t-balls of the nodes, $\{\mathscr{B}_v(t)\}_{v\in V}$. Let P_2 be a cover for the collection of the $\lceil D/2 \rceil$-neighborhoods of the nodes, $\{\Gamma(v, \lceil D/2 \rceil)\}_{v\in V}$. Set $P = P_1 \cup P_2$.

This change is responsible for the improvement in the dilation bound. Clearly, since P is a cover for the collection of t-balls of the nodes, the PIR2 strategy is a special case of the PIR1 strategy.

4. Analytic results for PIR

We start with some analytic results on the PIR strategy. Let PIRi denote the set of all routing schemes generated by the PIRi strategy ($i = 1, 2$) with t-balls on the class of n-node graphs for $t = \Theta(\sqrt{n\log n})$, and a cover P of size $O(\sqrt{n\log n})$. Let us set PIR = PIR1 ∪ PIR2. Since t is fixed from now on, let us denote, for every node u, $\mathscr{B}_u = \mathscr{B}_u(t)$.

The results shown hereafter hold regardless of whether the cover P is constructed using the randomized or the greedy algorithm, and some of these results hold for *any* cover P of size $O(\sqrt{n\log n})$. Moreover, most of the results apply also for the PIR2 strategy.

We first describe a property of the average stretch factor.

Proposition 4.1. *For every $R \in$ PIR on the unweighted n-vertex complete graph K_n,* $\mathrm{AvStr}_{K_n}(R) = 2 - o(1)$.

Proof. Let $R \in$ PIR be a routing scheme on K_n with pivot set P. Observe that the route from a node u to a node v would be of length 1 if $v \in \mathscr{B}_u$ and of length 2 otherwise, and that $t = |\mathscr{B}_u|$ is independent of the choice of the set of pivots. Consequently,

$$\mathrm{AvStr}_{K_n}(R) = \frac{1}{n(n-1)}\sum_{u\neq v}\rho_R(u, v) = \frac{n}{n(n-1)}(1\cdot(t-1) + 2\cdot(n-t)) = 2 - o(1)$$

for $t = \Theta(\sqrt{n\log n})$. □

The next result shows that the bound of Theorem 3.1 on the average stretch factor is tight, as an average stretch factor of 3 can be reached asymptotically with the PIR1 strategy.

Theorem 4.2. *There exists an unweighted n-node graph G and an identity assignment* ID *for its nodes such that for every $R \in$ PIR1 on G with a set of pivots computed by the greedy cover algorithm* (*respectively*, *the randomized algorithm*), $\mathrm{AvStr}_G(R) > 3 - o(1)$ (*respectively*, *with high probability*).

Proof. Given two suitable integers $m, k \geqslant 1$, let us construct an unweighted n-node graph $G = (V, E)$ defined as follows. Let $V = A \cup B$, where A is a clique of size mk, consisting of k subsets of size m, denoted by $A_1, ..., A_k$. Likewise, B is composed of $2k$ cliques of size $t - 1$, denoted by $B_1, ..., B_k$ and $B'_1, ..., B'_k$. In addition, for every $i \in \{1, \ldots, k\}$, each node $u \in B_i$ is connected to all the nodes of A_i, and B'_i is connected to B_i by a matching. (See Fig. 1.) Hence, $n = |V| = mk + 2(t-1)k$ (where m and k must be selected so as to insure that $t = \Theta(\sqrt{n\log n})$, as required from the size of the balls of PIR1 schemes).

Consider the following identity assignment ID for the nodes of G, selected in order to enforce a specific choice of the closest pivot. The identities for nodes of B are selected arbitrarily from the set $\{1, \ldots, |B|\}$, and the identities of

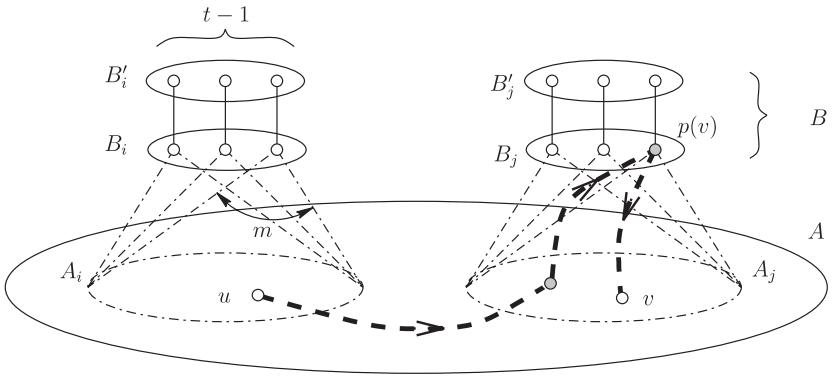

Fig. 1. A example for proving an average stretch factor of $3 - o(1)$.

the nodes of A are selected arbitrarily from the set $\{|B| + 1, \ldots, n\}$. Then the balls around the vertices of G satisfy the following claim.

Claim 4.3. *For every $u \in V$, $\deg(u) \geqslant t - 1$. Moreover, if $u \in A_i$, then $\mathscr{B}_u = \{u\} \cup B_i$, and if $u \in B_i \cup B'_i$, then $\mathscr{B}_u \subseteq B_i \cup B'_i$.*

Proof. For every $u \in A$ and $v \in B$, $\deg(u) = |A| - 1 + t$, and $\deg(v) \geqslant t - 1$. Hence, for every $u \in V$, $\mathscr{B}_u$ consists of u and the $t - 1$ lowest-ID neighbors of u. Let us consider some $i \in \{1, \ldots, k\}$. For each $u \in A_i$, the $t - 1$ lowest-ID neighbors of u consist of all the nodes of B_i, so $\mathscr{B}_u \backslash \{u\} = B_i$. For each $u \in B_i \cup B'_i$, u has exactly $t - 1$ neighbors in $B_i \cup B'_i$, because the identities of the nodes of B are lower than those of the nodes of A. Hence $\mathscr{B}_u \subseteq B_i \cup B'_i$. □

Let P be the set of pivots. By Claim 4.3, for every $i \in \{1, \ldots, k\}$, and for every $u \in A_i \cup B_i \cup B'_i$, the ball $\mathscr{B}_u$ satisfies $\mathscr{B}_u \subseteq A_i \cup B_i \cup B'_i$. Hence, no matter how P is chosen, there exists at least one pivot $p \in P$ such that $p \in A_i \cup B_i \cup B'_i$. Let $\mathscr{E}_{OK}$ denote the event that for every $i \in \{1, \ldots, k\}$ there is a pivot $p \in P$ such that $p \in B_i$. Then the probability that $\mathscr{E}_{OK}$ holds satisfies the following claim.

Claim 4.4. (1) *If P is computed by the greedy cover algorithm, then $\mathbb{P}(\mathscr{E}_{OK}) = 1$.*
(2) *If P is computed by the randomized algorithm, then $\mathbb{P}(\mathscr{E}_{OK}) > 1 - 1/\sqrt{n}$.*

Proof. (1) Let p be the first pivot selected in $A_i \cup B_i \cup B'_i$. Note that if $p \in A_i$ then p covers only the ball around p itself. Also, if $p \in B'_i$ then p covers at most $\deg(p) = |B'_i| + 1 = t$ balls. On the other hand, if $p \in B_i$ then p covers at least $|A_i| + |B_i| + 1 = m + t$. It follows that for each i, the greedy algorithm will necessarily choose a pivot in B_i.

(2) Recall that the randomized algorithm randomly picks each node of V into the cover with probability $\gamma = (c \ln n)/t$ for constant $c > 1$. For every $i \in \{1, \ldots, k\}$, let $X_i = |B_i \cap P|$ denote the random variable counting the number of pivots in B_i. For every i,

$$\mathbb{P}(X_i = 0) = (1 - \gamma)^{|B_i|} = (1 - \gamma)^{t-1},$$

so $\mathbb{P}(\exists i, X_i = 0) \leqslant k(1 - \gamma)^{t-1}$. It follows that

$$\mathbb{P}(\mathscr{E}_{OK}) = \mathbb{P}(\forall i, X_i > 0) \geqslant 1 - k(1 - \gamma)^{t-1}.$$

Note that $k = n/(m + 2(t - 1)) \leqslant \sqrt{n}/2$ (for n large enough). In order to bound the probability we use the following known fact.

Fact 4.5. *For all* $0 < \alpha \leqslant 1/2$ *and* $\beta > 0$, $(1-\alpha)^{\beta} < \mathrm{e}^{-\alpha\beta}$.

Hence,

$$\begin{aligned}\mathbb{P}(\mathscr{E}_{OK}) &\geqslant 1 - k\mathrm{e}^{-\gamma(t-1)} \geqslant 1 - \frac{\sqrt{n}}{2}\mathrm{e}^{-c(1-1/t)\ln n} \\ &> 1 - \tfrac{1}{2}n^{-(1/2-1/t)} > 1 - n^{-1/2},\end{aligned}$$

completing the proof of Claim 4.4. □

By Claim 4.4, every clique B_i contains at least one pivot. Consider a routing scheme $R \in \mathrm{PIR1}$ on G with pivot set P. We want to lower bound $\mathrm{AvStr}_G(R)$. Because $d_G(u, v) = 1$ for all $u, v \in A$, we have

$$\mathrm{AvStr}_G(R) > \frac{1}{n(n-1)} \sum_{u \neq v \in A} \rho_R(u, v).$$

For every node $u \in V$, let $p(u) \in P$ denote the closest pivot of u, according to the distance definition. For every $p \in P$, let T_p be the minimum spanning tree rooted in p and used by R. Consider some pair $u, v \in A$, $u \neq v$. Note that if $d_G(u, p(v)) = 2$ and $\{u, v\}$ is not an edge of $T_{p(v)}$ then $\rho_R(u, v) = 3$. Given u, the number of nodes $v \in A$ such that $\{u, v\}$ is an edge of $T_{p(v)}$ is $k-1$, since all the nodes of a same component A_i share a unique pivot in B_i, and hence use the same tree to route inside A_i. Moreover, $d_G(u, p(v)) = 1$ if and only if u and v are both in the same A_i. In total, the number of nodes v such that $\rho_R(u, v) \neq 3$ is at most $m + k - 1$, so

$$\sum_{v \in A \setminus \{u\}} \rho_R(u, v) \geqslant 3(|A| - 1 - (m + k - 1)) = 3mk - 3(m + k).$$

Summing over all $u \in A$, it follows that

$$\sum_{u \neq v \in A} \rho_R(u, v) \geqslant 3(mk)^2 - 3mk(m + k).$$

Letting $k = \sqrt{n}/\ln n$, we have $mk = n - 2(t-1)k = n - cn/\sqrt{\ln n}$ and $m + k < c'\sqrt{n}\ln n$, for large n and for suitable constants $c, c' > 0$. Hence,

$$\begin{aligned}\mathrm{AvStr}_G(R) &> \frac{3(n - cn/\sqrt{\ln n})^2 - 3(n - cn/\sqrt{\ln n})c'\sqrt{n}\ln n}{n(n-1)} \\ &> \frac{3n^2 - 6cn/\sqrt{\ln n} - 3c'n^{3/2}\sqrt{\ln n}}{n^2} > 3 - \frac{6c}{n\sqrt{\ln n}} - \frac{3c'\sqrt{\ln\, n}}{\sqrt{n}} \\ &> 3 - \mathrm{O}\left(\sqrt{\frac{\log n}{n}}\right),\end{aligned}$$

completing the proof of Theorem 4.2. □

Finally, we show that the average stretch factor of the PIR1 strategy is asymptotically 1.875 for almost all unweighted graphs. This is done as follows. Let $\Gamma(v) = \Gamma(v, 1)$. Call an n-vertex graph $G = (V, E)$ *typical* if it enjoys the following properties:

[TYP1] G is of diameter 2.
[TYP2] For every node $v \in V$, $n/2 - 3\sqrt{n\ln n} \leqslant |\Gamma(v)| \leqslant n/2 + 3\sqrt{n\ln n}$,
[TYP3] For every two nodes $v, w \in V$, $n/4 - 3\sqrt{n\ln n} \leqslant |\Gamma(v) \cap \Gamma(w)| \leqslant n/4 + 3\sqrt{n\ln n}$.

Recall that $\mathscr{G}_{n,p}$ denotes the class of n-node random graphs, where p represents the probability to have an edge between any two nodes. For random graphs selected from $\mathscr{G}_{n,1/2}$ we have the following. (The simple bounds in the definition of a typical graph suit our needs; see [6] for sharper statements.)

Lemma 4.6. *With probability at least* $1 - 1/n^3$, *a random unweighted graph* $G \in \mathscr{G}_{n,1/2}$ *is typical.*

Proof. Consider a random unweighted graph $G \in \mathcal{G}_{n,1/2}$. We prove only that with high probability, G satisfies the third property, [TYP3]. Fix a pair of nodes $v, w \in V$. For every other node x, let $J(x)$ denote the event that x is adjacent jointly to both v and w, namely,

$$J(x) = \text{“}\, x \in \Gamma(v) \cap \Gamma(w)\text{”}\,.$$

Note that this event occurs with probability $\frac{1}{4}$, and the events $J(x)$ and $J(x')$ are independent for every $x, x' \in V$. Let Z denote a random variable counting $|\Gamma(v) \cap \Gamma(w)|$. Then $Z = \sum_{x \neq v,w} z_x$, where z_x is the characteristic random variable of the event $J(x)$. Hence, Z is the sum of $n-2$ mutually independent Bernoulli variables, and its expected value is $\mathbb{E}(Z) = (n-2)/4$, and hence applying Chernoff's bound (cf. [1]) we get

$$\begin{aligned}\mathbb{P}\left(Z > \frac{n-2}{4} + 3\sqrt{n \ln n}\right) &< \mathbb{P}\left(Z > \left(1 + 7\sqrt{\frac{\ln n}{n}}\right)\mathbb{E}(Z)\right)\\ &< \exp\left(-\frac{49 \ln n\, \mathbb{E}(Z)}{3n}\right) < \frac{1}{n^3}\end{aligned}$$

for $n \geqslant 8$, and likewise

$$\mathbb{P}\left(Z < \frac{n-2}{4} - 3\sqrt{n \ln n}\right) < \frac{1}{n^3},$$

and the claim follows. □

Lemma 4.7. *For every unweighted n-node typical graph G, and for every* $R \in$ PIR1 *on* G, $\mathrm{AvStr}_G(R) = 1.875 \pm o(1)$.

Proof. Consider an unweighted n-node typical graph G. Fix some node v in G. Note that by property [TYP1], the nodes of $V \setminus \{v\}$ are partitioned into two sets, $A = \Gamma(v) \setminus \{v\}$ and $B = V \setminus \Gamma(v)$, where every node in B is adjacent to some node of A. Also note that by choice of the pivots, $p(v)$ is necessarily in A. Let $C = \Gamma(v) \cap \Gamma(p(v))$. By property [TYP3] we have

$$\frac{n}{4} - 3\sqrt{n \ln n} \leqslant |C| \leqslant \frac{n}{4} + 3\sqrt{n \ln n}. \tag{1}$$

By property [TYP2] we have

$$\frac{n}{2} - 3\sqrt{n \ln n} \leqslant |\Gamma(v)| \leqslant \frac{n}{2} + 3\sqrt{n \ln n}. \tag{2}$$

Let us calculate the average stretch over all the routes leading to v. For every node $x \in V$, let $s(x) = \rho_R(x, v)/d_G(x, v)$ denote the stretch of the route from x to v. By case analysis we show that $s(x)$ can assume only four possible values, namely, 1, 1.5, 2 or 3. Denote the set of vertices x with $s(x) = z$ by M_z.

Optimal stretch ($s(x) = 1$) is achieved for nodes x in the set $M_1 = \mathcal{B}_v \cap (B \cap \Gamma(p(v)))$, where in $\mathcal{B}_v$ the shortest path and the actual route are both of length 1, and in $B \cap \Gamma(p(v))$ the length of both paths is 2. By construction, $|\mathcal{B}_v| = t = \Theta(\sqrt{n \log n})$, so $|\mathcal{B}_v| \leqslant c_0\sqrt{n \ln n}$ for some constant $c_0 > 0$. By the definition of B,

$$B \cap \Gamma(p(v)) = \Gamma(p(v)) \cap (V \setminus \Gamma(v)) = \Gamma(v) \setminus \Gamma(p(v)) = \Gamma(v) \setminus C,$$

so $|B \cap \Gamma(p(v))| = |\Gamma(v)| - |C|$. By inequalities (1) and (2),

$$\frac{n}{4} - 6\sqrt{n \ln n} \leqslant |B \cap \Gamma(p(v))| \leqslant \frac{n}{4} + 6\sqrt{n \ln n}. \tag{3}$$

Hence,

$$\frac{n}{4} - 6\sqrt{n \ln n} \leqslant M_1 \leqslant \frac{n}{4} + 6\sqrt{n \ln n}. \tag{4}$$

The only nodes x for which $s(x) = 1.5$ are those nodes in $M_{1.5} = B \setminus \Gamma(p(v))$. As

$$B \setminus \Gamma(p(v)) = (V \setminus \Gamma(v)) \setminus \Gamma(p(v)),$$

we have $|B\backslash\Gamma(p(v))| = |V| - |\Gamma(v)| - |C|$, and by inequalities (1) and (2) again,

$$\frac{n}{4} - 9\sqrt{n\ln n} \leqslant |M_{1.5}| = |B\backslash\Gamma(p(v))| \leqslant \frac{n}{4} + 9\sqrt{n\ln n}. \tag{5}$$

The nodes x for which $s(x) = 2$ are those in $M_2 = (A\backslash\mathscr{B}_v) \cap \Gamma(p(v))$. As

$$|C\backslash\mathscr{B}_v| \leqslant |(\Gamma(v)\backslash\mathscr{B}_v) \cap \Gamma(p(v))| \leqslant |C|,$$

we have, by inequality (1) again,

$$\frac{n}{4} - (3 + c_0)\sqrt{n\ln n} \leqslant |M_2| = |(A\backslash\mathscr{B}_v) \cap \Gamma(p(v))| \leqslant \frac{n}{4} + 3\sqrt{n\ln n}. \tag{6}$$

Finally, the nodes x for which $s(x) = 3$ are those in $M_3 = A\backslash(\mathscr{B}_v \cup \Gamma(p(v)))$. As $A\backslash(\mathscr{B}_v \cup \Gamma(p(v))) = \Gamma(v) - C - \mathscr{B}_v$, by inequality (3),

$$\frac{n}{4} - (6 + c_0)\sqrt{n\ln n} \leqslant |M_3| = |A\backslash(\mathscr{B}_v \cup \Gamma(p(v)))| \leqslant \frac{n}{4} + 6\sqrt{n\ln n}. \tag{7}$$

By inequalities (4)–(7), it follows that the average stretch from v to some $x \in V$, $x \neq v$, is at most

$$\begin{aligned}&\frac{1}{n-1}\left(1\cdot|M_1| + 1.5\cdot|M_{1.5}| + 2\cdot|M_2| + 3\cdot|M_3|\right)\\ &\quad\leqslant \frac{1}{n-1}\left((1 + 1.5 + 2 + 3)\cdot\frac{n}{4} + \mathrm{O}\left(\sqrt{n\log n}\right)\right) \leqslant 1.875 + \mathrm{O}\left(\sqrt{\frac{\log n}{n}}\right)\end{aligned}$$

and at least $1.875 - \mathrm{O}(\sqrt{(\log n)/n})$, and hence the same bounds apply to the average over all destinations v, yielding

$$1.875 - \mathrm{O}\left(\sqrt{\frac{\log n}{n}}\right) \leqslant \mathrm{AvStr}_G(R) \leqslant 1.875 + \mathrm{O}\left(\sqrt{\frac{\log n}{n}}\right). \qquad \square$$

Lemmas 4.6 and 4.7 immediately yield the following.

Theorem 4.8. *For almost every unweighted n-node graph G, and for every $R \in$ PIR1 on G, $\mathrm{AvStr}_G(R) = 1.875 \pm \mathrm{o}(1)$.*

5. PIR vs. CP schemes

This section compares the PIR1 strategy with other general strategies developed in [3], in particular the CP strategies. The strategy is similar to PIR, and imposes similar memory requirements, but gives a better bound on the stretch factor. On the other hand, the CP schemes are not IRSs and thus do not enjoy the benefits of IRSs discussed above. In particular, the paths followed by messages in the CP schemes are not loop free. We show that the average stretch factor guaranteed by the PIR strategies is never worse than that guaranteed by the CP strategy, whereas for trees the PIR strategy is strictly better.

Let HCP_k be the routing strategy developed in [3] called *hierarchical covering pivot*.

Theorem 5.1 (*Awerbuch et al. [3]*). *For every n-node weighted graph G, and for every integer $k \geqslant 1$, HCP_k generates a routing scheme R on G such that*[3] $\mathrm{Memory}_G(R) = \mathrm{O}(kn^{1+1/k}\log^{2-1/k} n)$ *and* $\mathrm{Stretch}_G(R) \leqslant 2^k - 1$. *Moreover, R is constructible in polynomial time.*

The HCP_k strategy first selects a hierarchy of sets of pivots, $P_{k-1} \subset \cdots \subset P_1 \subset P_0 = V$, each of size about $n^{1-i/k}$. Each pivot in P_i is able to route on a shortest path to nodes in its neighborhood of size roughly $n^{(i+1)/k}$. When a node u wants to send a message to v, the message is transferred through the chain of pivots associated with u from successively higher levels, until it reaches a pivot able to route directly to v. For concreteness, let us consider $k = 2$. Then the scheme

[3] In the original paper the bound was $\mathrm{O}(kn^{1+1/k}\log^2 n)$, but it can be improved by a factor $\log^{1/k} n$ with a slightly better optimization.

generated by HCP_2 routes messages through one level of pivots, P_1. ($P_0 = V$ is not considered here.) Hence the route from u to v follows, successively, u, the pivot $p(u)$ closest to u in P_1, and then v. In comparison, the route generated by the PIR1 or PIR2 strategy would go through u, $p(v)$ and then v, allowing a loop-free routing, and bounding the compactness of its interval routing implementation.

In the analysis of the memory complexity of the HCP_k strategy it is assumed that the length of node identities is $O(\log n)$ bits (the pivots need to keep some translation tables storing the identities of some nodes). For every $k > 1$, the scheme generated by HCP_k is not loop free, and the upper bound it gives on the local memory requirement of a node is no better than $O(n \log n)$ bits.

Assuming the same total memory requirements, HCP_2 and PIR are similar strategies, although the bound on the stretch factor is only 3 for HCP_2 and 5 for PIR. We are therefore interested in the schemes of the strategy HCP_2. In [3], the HCP_2 strategy is called the CP strategy. We denote by t the common size of the neighborhoods (t-balls) used for both strategies, and choose $t = \Theta(\sqrt{n \log n})$ to the size optimizing the total memory requirements.

We first show that in terms of their average stretch factor, the schemes of PIR1 are asymptotically never worse than those of CP.

Theorem 5.2. *For every n-node graph, and for every two routing schemes* $R_{\mathrm{PIR1}} \in \mathrm{PIR1}$ *and* $R_{\mathrm{CP}} \in \mathrm{CP}$ *on* G, $\mathrm{AvStr}_G(R_{\mathrm{PIR1}}) \leqslant \mathrm{AvStr}_G(R_{\mathrm{CP}}) + o(1)$.

Proof. Consider a set of pivots P for both schemes, and any two nodes u, v at distance d. Three cases may occur:

(1) $\mathscr{B}_u \cap \mathscr{B}_v = \emptyset$;
(2) $v \in \mathscr{B}_u$ and $u \in \mathscr{B}_v$ and
(3) $v \in \mathscr{B}_u$ and $u \notin \mathscr{B}_v$, or the reverse.

For cases (1) and (2)

$$\rho_{R_{\mathrm{PIR1}}}(u, v) + \rho_{R_{\mathrm{PIR1}}}(v, u) = \rho_{R_{\mathrm{CP}}}(u, v) + \rho_{R_{\mathrm{CP}}}(v, u).$$

In case (3), $\rho_{R_{\mathrm{PIR1}}}(u, v) = \rho_{R_{\mathrm{CP}}}(u, v) = d$, while in the opposite direction $\rho_{R_{\mathrm{PIR1}}}(v, u) \leqslant 5d$ but $\rho_{R_{\mathrm{CP}}}(v, u) \leqslant 3d$. So R_{PIR1} may induce longer routing paths for case (3). However, given a node u, the number of pairs (u, v) such that $v \in \mathscr{B}_u$ and $u \notin \mathscr{B}_v$ is bounded by $|\mathscr{B}_u| = t$. Therefore, the total number of such pairs in the whole graph is at most nt, which is only a $t/(n-1)$ fraction of all the pairs. For each such pair the stretch can increase by at most 2, so $\mathrm{AvStr}_G(R_{\mathrm{PIR1}}) - \mathrm{AvStr}_G(R_{\mathrm{CP}}) \leqslant 2t/(n-1) = o(1)$, as $t = \Theta(\sqrt{n \log n})$. □

As a corollary of Theorem 5.2, we have that $\mathrm{AvStr}_{K_n}(R) \geqslant 2 - o(1)$ for $R \in \mathrm{CP}$ on K_n (Proposition 4.1), and also that there exists some worst-case graph G with specific node IDs such that $\mathrm{AvStr}_G(R) > 3 - o(1)$, for every $R \in \mathrm{CP}$ on G with pivot set chosen randomly or computed with the greedy algorithm (Theorem 4.2). Similar to Theorem 4.8, almost every graph G satisfies $\mathrm{AvStr}_G(R) \geqslant 1.875 - o(1)$ for $R \in \mathrm{CP}$ on G.

Next we illustrate the advantage of the PIR1 strategy over the CP strategy w.r.t. average stretch factor.

Theorem 5.3. *Let* T *be the star* $K_{1,n-1}$, *and let* r *be its root. If* $\mathrm{ID}(r)$ *is not the lowest* ID, *then for all* $R_{\mathrm{PIR1}} \in \mathrm{PIR1}$ *and* $R_{\mathrm{CP}} \in \mathrm{CP}$ *on* T *with set of pivots computed by the random or the greedy cover algorithm*, $\mathrm{AvStr}_T(R_{\mathrm{CP}}) > 2 - o(1)$ (*with high probability in the randomized case*), *whereas* $\mathrm{AvStr}_T(R_{\mathrm{PIR1}}) = 1$.

Proof. Clearly, $\mathrm{AvStr}_T(R) = 1$ for every $R \in \mathrm{PIR1}$, as PIR1 always routes optimally on trees. Suppose that the IDs assigned to the nodes of T are 1–n, and that $\mathrm{ID}(r) \neq 1$. Let $l = t - 2$. Note that in this case, the ball $\mathscr{B}_v$ built by the scheme around each leaf v consists of the nodes $\{1, \ldots, x\} \cup \{r, v\}$, where

$$x = \begin{cases} l, & v, r > l, \\ l+1, & v > l \text{ and } r \leqslant l \text{ or vice versa}, \\ l+2, & v, r \leqslant l, \end{cases}$$

and similarly for the ball $\mathscr{B}_r$ around the root.

Let us first consider the greedy pivot selection algorithm. Assume that in the greedy algorithm, ties (between two candidates to join the pivot set) are broken in favor of the smallest-ID candidate. Note that when selecting the first pivot,

all nodes in $\{1,\ldots,l\}\cup\{r\}$ are equal candidates, and hence the algorithm will pick 1 as the first (and only) pivot. Hence, the resulting routing scheme $R \in \mathrm{CP}$ will route messages between all but $\mathrm{O}(nt)$ pairs of nodes (specifically, between any $v, w \in V$ such that $v, w > l$ and $v, w \neq r$) in four steps rather than two. Thus, $\mathrm{AvStr}_T(R_{\mathrm{CP}}) > 2 - \mathrm{O}(t/n) = 2 - \mathrm{o}(1)$, as $t = \Theta(\sqrt{n \log n})$.

As for the randomized cover algorithm, we are guaranteed that with high probability all balls are covered by at least one pivot, but the probability that r was chosen as pivot is roughly $\ln n/\sqrt{n}$, and moreover, the probability that r was the *smallest* ID pivot chosen is even smaller. Hence, with probability at least $1 - \ln n/\sqrt{n}$, the analysis of the greedy strategy applies to the randomized strategy as well, and hence with this probability $\mathrm{AvStr}_T(R_{\mathrm{CP}}) > 2 - \mathrm{O}(\sqrt{(\log n)/n})$. □

References

[1] N. Alon, J.H. Spencer, The Probabilistic Method, Wiley, New York, 1992.
[2] B. Awerbuch, A. Bar-Noy, N. Linial, D. Peleg, Compact distributed data structures for adaptive routing, in: 21st Symposium on Theory of Computing (STOC), vol. 2, 1989, pp. 230–240.
[3] B. Awerbuch, A. Bar-Noy, N. Linial, D. Peleg, Improved routing strategies with succinct tables, J. Algorithms 11 (1990) 307–341.
[4] B. Awerbuch, D. Peleg, Sparse partitions, in: 31st Symposium on Foundations of Computer Science (FOCS), 1990, pp. 503–513.
[5] B. Awerbuch, D. Peleg, Routing with polynomial communication–space trade-off, SIAM J. Discrete Math. 5 (1992) 151–162.
[6] B. Bollobás, Random Graphs, Academic Press, New York, 1975.
[7] H. Buhrman, J.-H. Hoepman, P. Vitányi, Space-efficient routing tables for almost all networks and the incompressibility method, SIAM J. Comput. 28 (1999) 1414–1432.
[8] T. Eilam, C. Gavoille, D. Peleg, Compact routing schemes with low stretch factor, J. Algorithms 46 (2) (2003) 97–114 Preliminary version appeared in: Proceedings of the 17th ACM Symposium on Principles of Distributed Computing, June 1998, pp. 11–20.
[9] T. Eilam, S. Moran, S. Zaks, A lower bound for linear interval routing, in: Ö. Babaoğlu, K. Marzullo (Eds.), 10th Workshop on Distributed Algorithms (WDAG), Lecture Notes in Computer Science, vol. 1151, Springer, Berlin, October 1996, pp. 191–205.
[10] M. Flammini, G. Gambosi, U. Nanni, R.B. Tan, Multidimensional interval routing schemes, Theoret. Comput. Sci. 205 (1998) 115–133.
[11] P. Fraigniaud and C. Gavoille, Memory requirement for universal routing schemes, in: 14th ACM Symposium on Principles of Distributed Computing, August 1995, pp. 223–230.
[12] G.N. Frederickson, R. Janardan, Separator-based strategies for efficient message routing, in: 27th Symposium on Foundations of Computer Science (FOCS), May 1986, pp. 428–437.
[13] G.N. Frederickson, R. Janardan, Designing networks with compact routing tables, Algorithmica 3 (1988) 171–190.
[14] G. Gambosi, P. Vocca, Topological routing schemes, in: Ö. Babaoğlu, K. Marzullo (Eds.), 10th Workshop on Distributed Algorithms (WDAG), Lecture Notes in Computer Science, vol. 1151, Springer, Berlin, October 1996, pp. 206–219.
[15] C. Gavoille, On the dilation of interval routing, Comput. J. 43 (2000) 1–7.
[16] C. Gavoille, A survey on interval routing, Theoret. Comput. Sci. 245 (2000) 217–253.
[17] C. Gavoille, M. Gengler, Space-efficiency of routing schemes of stretch factor three, in: D. Krizanc, P. Widmayer (Eds.), Fourth Colloquium on Structural Information and Communication Complexity (SIROCCO), Carleton Scientific, July 1997, pp. 162–175.
[18] C. Gavoille, E. Guévremont, Worst case bounds for shortest path interval routing, J. Algorithms 27 (1998) 1–25.
[19] C. Gavoille, D. Peleg, The compactness of interval routing for almost all graphs, in: S. Kutten (Ed.), 12th Symposium on Distributed Computing (DISC), Lecture Notes in Computer Science, vol. 1499, Springer, Berlin, September 1998, pp. 161–174.
[20] C. Gavoille, D. Peleg, The compactness of interval routing, SIAM J. Discrete Math. 12 (1999) 459–473.
[21] C. Gavoille, S. Pérennès, Memory requirement for routing in distributed networks, in: 15th ACM Symposium on Principles of Distributed Computing, May 1996, pp. 125–133.
[22] R. Kráľovič, P. Ružička, D. Štefankovič, The complexity of shortest path and dilation bounded interval routing, in: C. Lengaur, M. Griebl, S. Gorlatch (Eds.), Third Euro-Par Conference, Lectures Notes in Computer Science, vol. 1300, Springer, August 1997, pp. 258–265.
[23] L. Lovász, On the ratio of optimal integral and fractional covers, Discrete Math. 13 (1975) 383–390.
[24] D. Peleg, E. Upfal, A trade-off between space and efficiency for routing tables, J. ACM 36 (1989) 510–530.
[25] N. Santoro, R. Khatib, Labelling and implicit routing in networks, Comput. J. 28 (1985) 5–8.
[26] S.S.H. Tse, F.C.M. Lau, An optimal lower bound for interval routing in general networks, in: D. Krizanc, P. Widmayer (Eds.), Fourth Colloquium on Structural Information and Communication Complexity (SIROCCO), Carleton Scientific, July 1997, pp. 112–124.
[27] J. van Leeuwen, R.B. Tan, Computer networks with compact routing tables, in: G. Rozemberg, A. Salomaa (Eds.), The Book of L, Springer, Berlin, 1986, pp. 259–273.