Laboratoire Bordelais de Recherche en Informatique
UMR 5800 - Université Bordeaux I, 351, cours de la Libération,
33405 Talence CEDEX, France

Research Report `RR-1289-03`

Lowers Bounds for Oblivious Single-Message End-to-End Communication

by Pierre Fraigniaud and Cyril Gavoille

February, 2003

Lower Bounds for Oblivious Single-Message End-to-End Communication

Pierre Fraigniaud* Cyril Gavoille†

Abstract

The end-to-end communication problem is a protocol design problem, for sending a (sequence of) packet(s) from a specified source-node s to a specified target-node t, through an unreliable asynchronous communication network G. The protocol must insure reception and termination. In this paper, we measure the complexity of the protocol in term of header size, i.e., the quantity of information that must be attached to the packets to insure their delivery. We show that headers of $\Omega(\log\log\tau)$ bits are required in every network, where τ denotes the tree-width of the network. In planar networks, $\Omega(\log\tau)$ bits are required. In particular, this latter lower bound closes the open problem by Adler and Fich in PODC '99 about the optimality of the *hop-count* protocol in square meshes.

Keywords: End-to-End, Sequence Transmission, Tree-Width.

1 Introduction

The *end-to-end communication* problem is the problem of sending a (sequence of) packet(s) from a specified source-node s to a specified target-node t, through an *unreliable* communication network G (see, e.g., [12, 27]). *Sequence transmission* [36] problem and *reliable communication* [22] problem are other names for the end-to-end communication problem (cf. the survey [23]). By unreliable network, it is generally meant that links can lose, reorder, and duplicate messages. Moreover, networks are assumed to be *asynchronous*, i.e., the time for a packet to traverse a link is finite but otherwise unbounded. In particular, a processor cannot distinguish between an inoperational link and an operational link which is just very slow. Hence, an instance of the end-to-end communication problem is described by an (unreliable and asynchronous) network, modeled by an undirected graph $G = (V, E)$, and two nodes s and t of G. Solving the problem consists to design a distributed protocol which (1) allows s to send a packet, or a sequence of packets, to t through the network G, and (2) generates a finite amount of traffic for each packet. In other words, the end-to-end protocol must satisfy the two following requirements: (1) *reception*, i.e., the target must eventually receive at least one copy of each packet sent by the source; (2) *termination*, i.e., after a finite time, no copy of the packet(s) remains in the network.

*CNRS, Laboratoire de Recherche en Informatique, Bât. 490, Univ. Paris-Sud, 91405 Orsay cedex, France. *http://www.lri.fr/~pierre*. Supported by the Action Spécifique Dynamo of CNRS.*pierre@lri.fr*

†Laboratoire Bordelais de Recherche en Informatique, Université Bordeaux I, 33405 Talence cedex, France. *http://dept-info.labri.fr/~gavoille*, *gavoille@labri.fr*

1.1 The oblivious single-message end-to-end communication problem

For the sake of simplicity, this paper will consider a static model, that is we assume that each link is either operational or not. If there exists at least one operational path from s to t in G, it is the role of the protocol to find such a non-faulty path, which is of course *a priori* unknown. Note that the case of dynamic faults, that is when links can alternate between being operational and inoperational, can be treated similarly by assuming, as in [1, 13, 26], the infinitely frequent path stability, i.e., infinitely often there is a path P from s to t such that a packet sent from s along P will arrive at t (see also [3]).

Performances of end-to-end communication protocols are commonly measured in terms of (1) the amount of communication performed over the links of the network, and (2) the amount of storage space used by intermediate nodes in the networks. *Oblivious* protocols, a.k.a. *memoryless* protocols, take their routing decision at every node x (i.e., on which link(s) x has to forward a packet p) based solely on the content of the header of the packet. In particular, a node does not store any knowledge on the traffic that previously passed through it. It can however forward several copies of the received packet, and it can modify the header of this packet. As mentioned in [1], the practical advantage of oblivious protocols is their high tolerance to processor crash. Indeed, as soon as a node recovers from a crash, the protocol is ready to proceed with no risk of corruption due to an altered writable memory (RAM). Moreover, the extremal behavior of oblivious protocols (in the sense that they consume no local memory at all) allows to concentrate the analysis of end-to-end communication protocols on the amount of information transmitted over the links of the network. More specifically, this paper focuses on minimizing the *packet-header size*, i.e., the quantity of additional information that must be attached to every packet to insure their correct delivery.

The header content has two distinct functions: (1) to control the order in which packets arrive at the destination; (2) to find a route from the source s to the destination t (reception), and to insure that residual packets are eventually removed from the network (termination). In this paper, we are interested by the routing part of the problem, that is, by the problem of finding the non faulty route from the source to the destination, while insuring termination. We will therefore concentrate our analysis on the process of sending a single message from s to t. In other words, we consider the *single message* end-to-end communication problem, as opposed to the *stream-of-messages* end-to-end communication problem, the latter problem requiring the transmission of a sequence of messages from the source to the destination [20]. Hence, let us summarize our problem.

Our problem. We are given an unreliable and asynchronous network G, and two nodes s and t of G. We consider the design of an oblivious distributed protocol which allows the transmission of a packet from s to t (if there is a fault-free path between s and t in G), and which eventually lets the network empty of packets. Such a protocol is required to use packet-headers of small size. The quality of the protocol is indeed measured by the maximum size of the headers involved during its execution.

Previous works. In the context of static link failures (i.e., every link is operational or not but its status does not change during the execution of the routing protocol), the *hop-count* protocol [30] uses headers of size $O(\log n)$ in n-node networks. It proceeds as follows. The source sends a copy of the original packet to all its neighbors, with header 1. A node receiving a packet whose header contains the hop count $i < n-1$ updates the header by replacing i by $i+1$, and forwards a copy of the packet to each of its neighbors. A node receiving a packet whose header contains the hop count $n-1$ removes the packet from the network. If s and t are connected despite the faulty links, then a path of length at most $n-1$ exists between s and t, and therefore at least one copy of the packet sent by s eventually arrives in t. Moreover, the remaining copies of the packet are removed from the network after a finite time since no packet can traverse more than n links.

In [1], Adler and Fich showed that if a network G contains H as a *minor*, i.e., if H can be obtained from G by edge contraction, node and edge deletion, then the oblivious single-message end-to-end communication problem in G requires headers of size at least as large as for H. Hence the problem is closed under taking minors. They also showed that the complete graph of n nodes requires headers of size $\Omega(\log n)$. Therefore, an n-node network G which contains the complete graph of k nodes as a minor requires headers of size $\Omega(\log k)$. Adler and Fich also gave upper bounds on the header size based on the notion of *feedback vertex sets*. A feedback vertex set is a set $S \subseteq V$ such that every cycle in the network contains at least a node in S. They showed that if there exists a feedback vertex set of size f, then there exists an oblivious protocol using headers with size $O(\log f)$. They also pointed out that any minimum feedback vertex set of an $\sqrt{n} \times \sqrt{n}$ mesh is of size $\Omega(n)$, and thus the feedback vertex set protocol does not offer significant improvement in the mesh, compared to the hop-count protocol. Since the mesh is planar, it does not contain K_5 as minor, and thus it lets a big gap between the best known upper and lower bounds for mesh networks. Adler, Fich, Goldberg, and Paterson [2] recently closed this gap for "narrow" meshes, i.e., $p \times q$ meshes with $p = O(1)$. They have shown that there exists a protocol for n-node meshes using headers of size $O(p(\log p + \log\log n))$, and that any $p \times q$ mesh requires headers of size $\Omega(\log\log n)$ for $3 \leqslant p \leqslant q$. Constant size header protocol exists if $p \leqslant 2$.

1.2 Outline of our Results

In this paper, we provide lower bounds of the header-size of oblivious end-to-end protocols as a function of the tree-width of the input graph. The notion of tree-width is a very rich concept with many algorithmic implications. Many NP-complete problems are polynomial for graphs of bounded tree-width [9, 15]. In addition, the notion of tree-with is in the kernel of the graph minor theory [18]. Recall that the tree-width can be seen as a measure of "how far" a graph is from a tree. More precisely, let $G = (V, E)$ be a graph of n nodes, a *tree-decomposition* of G is a pair $(\mathbf{S}, T)$ where $\mathbf{S}$ is a collection $\{S_i \mid i \in I\}$ of subsets of nodes and $T = (I, F)$ is a tree, such that the three following conditions are satisfied:

1. $\bigcup_{i\in I} S_i = V$;
2. For every edge $e = \{u, v\}$ of G, there exists $i \in I$ such that both u and v belong to S_i;
3. For every $u \in V$, the subgraph of T induced by the set of nodes $\{i \in I \mid u \in S_i\}$ is a tree.

For any graph G, there exists at least one tree-decomposition of G by choosing $\mathbf{S} = \{V\}$, i.e., T is reduced to a single node. The *width* of a tree-decomposition $(\mathbf{S}, T)$ is defined as $\max_{i\in I} |S_i| - 1$. The *tree-width* $\mathrm{tw}(G)$ of G is then defined as the minimum of the width of any tree-decomposition of G. A tree has tree-width 1, a cycle has tree-width 2, a square mesh has tree-width $\sqrt{n}$, and a complete graph has tree-width $n-1$. Determining the tree-width of a graph is NP-hard [10]. However, recognizing graphs of tree-width at most k is linear for $k = O(1)$, and there is an $O(\log n)$-approximation algorithm for computing the tree-width of an arbitrary graph (see [17] and also [16, 31]).

We show the following general lower bound:

Theorem 1.1 *Any protocol designed for the instance (G,s,t) of the oblivious single-message end-to-end communication problem uses headers of size at least $\Omega(\log\log\tau)$ bits where τ is the tree-width of the graph $G_{s,t}$ obtained from G, s, and t by deleting every edge e not on a simple path from s to t.*

As we will see in more details in Section 3, the bound of Theorem 1.1 derives from an optimal bound for square meshes, and from an upper bound of the "excluding grid" theorem of Robertson and Seymour [33]. This latter bound is likely far from best possible and, as mentioned in [19], Robertson, Seymour and Thomas [34] think that the upper bound might be exponentially improved so that the lower bound of Theorem 1.1 would be $\Omega(\log\tau)$.

Actually, for planar graphs, we derive a larger lower bound:

Theorem 1.2 *Any protocol designed for the instance (G,s,t) of the oblivious single-message end-to-end communication problem, G planar, uses headers of size at least $\Omega(\log\tau)$ bits where τ is the tree-width of the graph $G_{s,t}$.*

In particular Theorem 1.2 solves the conjecture mentioned in [1] stating that headers of $\Omega(\log n)$ bits are required to insure packet-delivery in the two dimensional square mesh. All these results establish (as also suggested in [1]) connections between, on one hand, the number of header bits needed to send a single message through an unreliable network G, and, on the other hand, graph-theoretic properties of G's topology.

Finally, notice that our lower bounds obviously apply to the stream-of-messages problem too.

1.3 Related Works

The case of links with fixed or bounded traversal time has been considered in [8, 39]. Probabilistic faults and delivery times have been considered in [25, 28, 29]. In the deterministic setting, i.e., the context of this paper, solutions for the stream-of-messages end-to-end problem differ according to the type of faults. Wand and Zuck [38] have shown that any protocol tolerating both packet reordering and duplication requires unbounded headers. Afek *et al.* [6] have shown that packet reordering and loss create the same effect, that is either unbounded headers or non termination (i.e., the same packet can be received an unbounded number of times). Fekete and Lynch [21] have shown that just packet loss implies that some header information must be attached to the packets for the stream of messages to be treated correctly. These three latter results hold even is G consists of a single edge $\{s,t\}$. Despite these impossibility results, efficient protocols have nevertheless been successfully designed. We refer to [11, 35, 37] if links are subject to packet duplication, reordering and loss, to [14, 28] if links are subject to packet duplication and loss (but no reordering), and to [5, 7, 27] in case of static link-failures. In [1], it is noticed that the latter protocols can be adapted to the case where links can loose packets –but otherwise transmit them in order and without duplication, by using the techniques in [4]. Although some of these protocols use very short headers, non of them is oblivious, i.e., they all require the local storage of information at intermediate nodes during the execution of the protocol. The stream-of-message protocol of Dolev and Welch [20] is oblivious (i.e., intermediate processors do not change state) and apply to the static link failure (i.e., the model of this paper). It uses headers of $O(\log p)$ bits where p is the number of distinct simple paths between s and t in G. Although p can be quite large in general (e.g., $\Theta(n!)$ in K_n), this protocol was proved optimal for many topologies, including complete graphs, meshes, and series-parallel graphs (see [1]). Fich and Jakoby [24] considered the same model for directed acyclic graphs. They proved that a single bit header suffices in DAG, which contrasts with the case of arbitrary graphs.

Several oblivious protocols have been proposed in the literature to solve the single-message end-to-end problem. We already presented the *hop-count* protocol described by Postel [30], using headers of $O(\log n)$ bits in n-node graphs. We also mentioned the protocol of Dolev and Welch [20] which applies to the single-message problem as well. Adler and Fich [1] derived lower and upper bounds for the header size in specific networks (e.g. meshes, hypercubes, butterflies, de Bruijn, etc.). Adler, Fich, Goldberg, and Paterson [2] addressed the problem in $p \times q$ meshes, $p = O(1)$. They proved the somewhat surprising result stating that headers of size $\Theta(\log\log n)$ bits are required in $3 \times n/3$ meshes (whereas headers of constant size are sufficient in $2 \times n/2$ meshes).

For a more detailed descriptions of the end-to-end protocols mentioned above, we refer to [23], and the references therein.

2 Preliminary Results

We start by proving the following lemma which is an extension to arbitrary networks of a result by Adler and Fich [1] valid for complete graphs only.

Lemma 2.1 *Let $e_i = \{x_i, y_i\}$, $i = 1, \ldots, k$, be k distinct edges of G. Let $\Pi \subseteq \Sigma_k$ be a subset of permutations of k symbols such that, for any $\pi \in \Pi$, there is a simple path P_π in G from s to t, traversing the e_i's from the x_i's to the y_i's, in the order $e_{\pi(1)}, e_{\pi(2)}, \ldots, e_{\pi(k)}$. Then any oblivious end-to-end communication protocol from s to t requires headers of at least $(\log_2 |\Pi|)/k$ bits.*

Proof. Let us given an oblivious end-to-end communication protocol A from s to t. Let h be the total number of distinct headers involved in protocol A. We use the same terminology as in [1]. The *transcript* of a simple path P_π is defined as the word w_π of length k on the alphabet $\{0, \ldots, h-1\}$ obtained by concatenating the headers of the packets transmitted through the e_i's, from the x_i's to the y_i's, when only the edges of P_π are operational. More precisely, let us consider a packet b_π going from s to t along P_π. This packet generates a word w_π whose ith letter is the value of the header h_i of b_i when traversing the edge e_i from x_i to y_i. This word is the transcript of P_π. (Obviously, the transcript may depend on the choice of the packet b_π. This packet is chosen arbitrarily.) There are at most h^k transcripts, and therefore, if $h^k < |\Pi|$, then at least two paths have the same transcript. Thus assume, for the purpose of contradiction, that $h^k < |\Pi|$, and let P_π and $P_{\pi'}$ be two paths with the same transcript $h_1 h_2 \ldots h_k$. Since $\pi \neq \pi'$, there is a pair (i,j), $i \neq j$, such that e_i is traversed before e_j in P_π, and e_i is traversed after e_j in $P_{\pi'}$.

Assume now that both paths P_π and $P_{\pi'}$ are operational. Since the protocol has no way to distinguish an operational link from a link that is just very slow, and since the protocol is oblivious, the headers of packets b_π and $b_{\pi'}$, respectively following P_π and $P_{\pi'}$, will be the same as those used to traverse these two paths when only one of them is operational. In particular, when y_i receives a packet from x_i with header h_i, it cannot know whether P_π, or $P_{\pi'}$, or both paths, are operational. Therefore, y_i forwards the packet through both paths. The same holds for y_j.

As a consequence, the following situation occurs. A packet traverses e_i with header h_i, eventually reaches x_j along P_π, traverses e_j with header h_j, eventually reaches x_i along $P_{\pi'}$, traverses e_i again, still with header h_i, and so on. This creates an infinite loop, in contradiction with the termination requirement. Therefore $h^k < |\Pi|$ cannot hold, and thus $h \geqslant |\Pi|^{1/k}$. ■

The main theorem is the following:

Theorem 2.1 *Any oblivious end-to-end communication protocol in the $p\times q$ mesh requires headers of at least $\Omega(\log\min\{p,q\})$ bits.*

Proof. Let $M_{p,q}$ be the $p\times q$ mesh. Assume, w.l.o.g., that $p\leqslant q$ (otherwise exchange the role of p and q). We see the mesh $M_{p,q}$ as with p rows and q columns. Rows are labeled from 0 to $p-1$, and columns from 0 to $q-1$. One can draw the mesh so that $(0,0)$ is the top-left corner, and $(p-1,q-1)$ is the bottom-right corner. Let s and t be the source node and the target node, respectively. The $p\times q$ mesh contains three node-disjoint $(p-2)\times\lfloor\frac{q-2}{3}\rfloor$ sub-meshes. At least one of these sub-meshes does not contain s nor t. Let M be this sub-mesh. One can construct two disjoint paths P_s and P_t in $M_{p,q}\setminus M$, respectively leading from s to the "top-left" corner of M, and from t to the "top-right" corner of M. Any oblivious end-to-end protocol must perform successfully even if all links in $M_{p,q}\setminus(M\cup P_s\cup P_t)$ are faulty. Hence one can assume, w.l.o.g., that s is node $(0,0)$ and t is node $(0,q-1)$.

Let $k=\lfloor\min\{\frac{p-1}{4},\sqrt{\frac{q}{12}}\}\rfloor$. Hence $p\geqslant 4k+1$ and $q\geqslant 12k^2\geqslant 4k^2+6k+2$. Let us consider the edges $e_i=\{x_i,y_i\}$, $i=1,\ldots,k$, where $x_i=(2k,i(4k+2)-1)$ and $y_i=(2k,i(4k+2))$. Fig. 1 illustrates the idea of the construction: 8 edges are displayed on row $2k$ (the horizontal and vertical scales are different). From the setting of k, there are exactly $2k$ rows above row $2k$ (which contains all the e_i's), and at least $2k$ rows bellow row $2k$. Also, there are exactly $4k+1$ columns separating column 0 from e_1, $4k$ columns separating two consecutive e_i's, and at least $4k$ columns separating column $q-1$ from e_k.

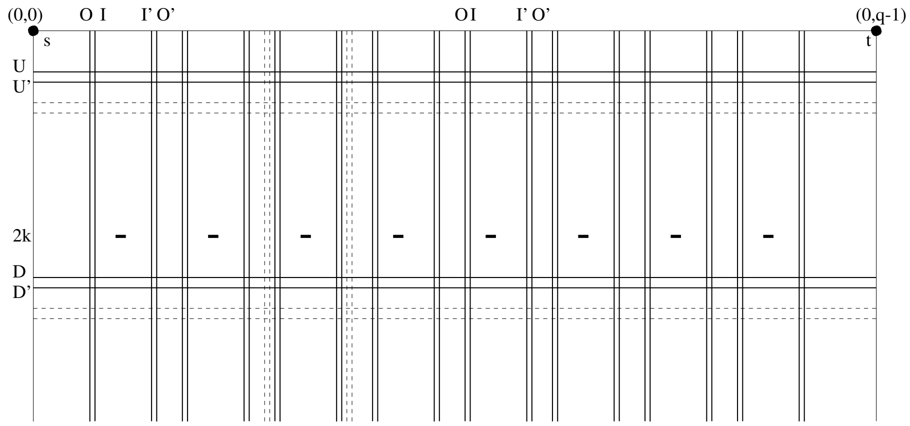

Figure 1: Associated rows and columns.

Let $\pi\in\Sigma_k$ be any permutation k symbols. Let us show that there is a simple path from s to t, passing through e_i from x_i to y_i for every i, in the order $e_{\pi(1)},e_{\pi(2)},\ldots,e_{\pi(k)}$. To each $e_{\pi(i)}$ are *associated* 4 rows and $4k$ columns. The rows associated to $e_{\pi(i)}$ are rows

$$U_i=2i-2,\ U_i'=2i-1,\ D_i=2k+2i-2,\ \text{and}\ D_i'=2k+2i-1.$$

Fig. 1 shows the 4 rows associated to some $e_{\pi(i)}$. U stands for "up", and D for "down". The dashed rows are examples of rows associated to some $e_{\pi(j)}$, $j>i$. Note that, since $p\geqslant 4k+1$, row $p-1$ is not associated to any $e_{\pi(i)}$. Let $c_i=i(4k+2)-1$ be the column coordinate of x_i, $i=1,\ldots,k$.

The columns associated to $e_{\pi(i)}$ are columns

$$I_{i,j} = c_j - 2i + 1,\ O_{i,j} = c_j - 2i,\ I'_{i,j} = c_j + 2i - 1, \text{ and } O'_{i,j} = c_j + 2i,\ j = 1, \ldots, k.$$

Fig. 1 shows the $4k$ columns associated to some $e_{\pi(i)}$. I stands for "inside", and O for "outside". The dashed columns are examples of columns associated to some $e_{\pi(j)}$, $j > i$. There is no overlapping between associated columns. Moreover, columns 0 and $q-1$ are not associated columns.

Now, for $i = 1, \ldots, k$, we define the path P_i as follows. (The construction is illustrated on Fig. 2 where the integer displayed above an edge indicates the order of the edge in the permutation π.)

P_1 starts from node $s = (0, 0)$ following the row U_1 until it reaches column $I_{1,j}$ where $j = \pi(1)$. Then P_1 follows $I_{1,j}$ downward until row $2k$. At this point P_1 follows row $2k$ rightward until it traverses edge $e_{\pi(1)}$. At node y_j, P_1 turns down to reach row $2k+1$, and follows this row leftward until column $O_{1,j}$. Then it goes upward on $O_{1,j}$ until row U'_1, and finally follows this row until node $(1, 0)$ where it ends.

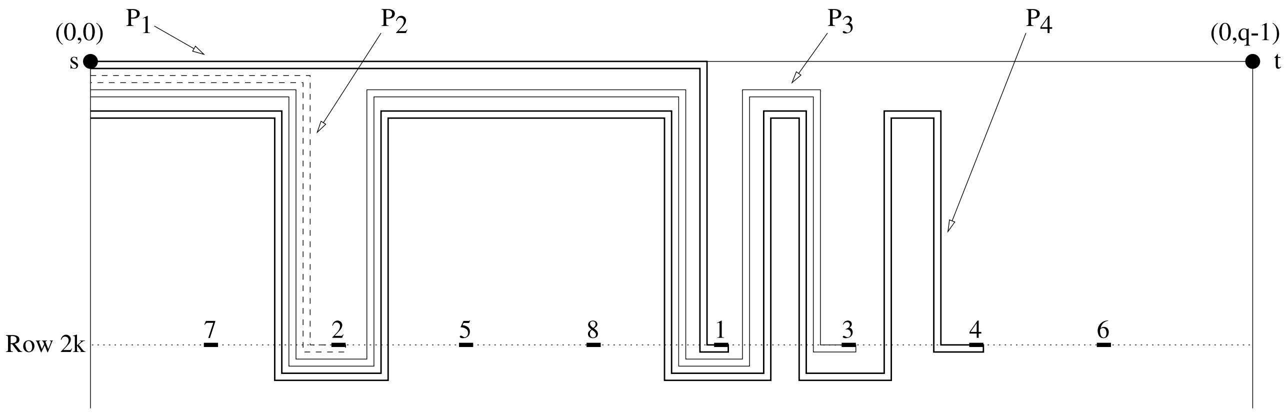

Figure 2: The path P_i's.

P_i starts from node $(2(i-1), 0)$. Let $l_1, l_2, \ldots, l_h$ be the indices $l < i$ such that $e_{\pi(l)}$ is on the left of $e_{\pi(i)}$. For instance, on Fig. 2, this sequence is 2, 1, 3, for $e_{\pi(4)}$. Up to relabeling, one can assume, w.l.o.g., that $e_{\pi(l_a)}$ is on the left of $e_{\pi(l_{a+1})}$. Let $j = \pi(i)$. P_i goes rightward from node $(2(i-1), 0)$ along row U_i until it reaches column I_{i,l_1}. Then it goes downward until it reaches row D_i, and turns right. P_i follows D_i until column I'_{i,l_1}, and then goes upward along that column, until it reaches row U_i. This "detour" around the edges $e_{\pi(l_a)}$'s is repeated for each $1 \leqslant a \leqslant h$. (See for instance the path P_4 on Fig. 2.) More precisely, let $a \in \{1, \ldots, h\}$. P_i remains on U_i until it reaches column I_{i,l_a}. Then it goes downward until it reaches row D_i, and turns right. P_i follows D_i until column I'_{i,l_a}, and then goes upward along that column, until it reaches row U_i again.

After the last detour around $e_{\pi(l_h)}$, P_i goes rightward on U_i until it reaches $I_{i,j}$. Then it goes downward on $I_{i,j}$ until it reaches row $2k$, and turn rightward along that row. P_i follows row $2k$ until it traverses $e_j = e_{\pi(i)}$. At node y_j, P_i turns down to reach row $2k+1$, and follows this row leftward until column $O_{i,j}$. Then it goes upward on $O_{i,j}$ until row U'_i. At this point, P_i starts its journey leftward on U'_i, back to column 0.

More precisely, P_i follows the same shape of path as when it went rightward from column 0 (see Fig. 2). That is, it proceeds along U'_i but during detours around edges $e_{\pi(l_a)}$'s. Let $a \in \{1, \ldots, h\}$. A detour around $e_{\pi(l_a)}$ consists of leaving U'_i to go downward along O'_{i,l_a} until row D'_i is reached.

Then P_i follows $D_i^\prime$ leftward until column O_{i,l_a}, and follows this column upward until it reaches again $U_i^\prime$. After the last detour (i.e., the detour around $e_{\pi(l_1)}$), P_i follows row $U_i^\prime$ until node $(2i-1,0)$ where it ends.

The following two properties are satisfied:

$\mathcal{P}1$. For every $1 \leqslant i \leqslant k$, P_i is a simple path.

$\mathcal{P}2$. For every $i \neq j$, P_i and P_j have no node in common. More precisely, the paths fit one into another as displayed on Fig. 3. In this figure, $a < b < i < j$. P_a goes straight above e because there is no need of a detour as P_b is not set yet. P_b traverses e. Both P_i and P_j must go around e. By construction, since $i < j$, they follow different associated rows and columns and therefore do not intersect.

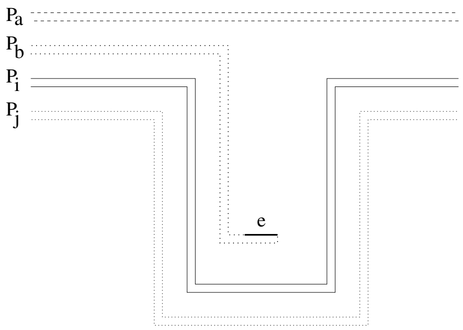

Figure 3: Two paths P_i and P_j do not intersect.

Now, let P be the following path from s to t:

$$P = P_1, f_1, P_2, f_2, \ldots, P_{k-1}, f_{k-1}, P_k, Q$$

where f_i is the edge between nodes $(2i-1,0)$ and $(2i,0)$, and Q is the simple path from node $(2k-1,0)$ to t following column 0, then row $p-1$, and finally column $q-1$. From properties $\mathcal{P}1$ and $\mathcal{P}2$, P is a simple path from s to t which traverse edges e_i's in the order $e_{\pi(1)}, e_{\pi(2)}, \ldots, e_{\pi(k)}$.

The construction applies for any permutation $\pi \in \Sigma_k$. Therefore, thanks to Lemma 2.1, any oblivious end-to-end communication protocol in the $p \times q$ mesh requires headers of at least $\Omega(\log(k!)/k) = \Omega(\log k)$ bits. Since $k = \lfloor \min\{\frac{p-1}{4}, \sqrt{\frac{q}{12}}\}\rfloor$, $\log k = \Omega(\log \min\{p,q\})$ which completes the proof. ■

3 Proofs of Theorems 1.1 and 1.2

3.1 Proof of Theorem 1.1

We use the "excluding grid" theorem of Robertson and Seymour, whose short proofs can be found in [18, 19]:

Theorem 3.1 (Robertson & Seymour [33]) *For every integer r there is an integer k such that every graph of tree-width at least k has an $r \times r$ mesh as minor.*

So, let us define $f(r)$ as the smallest integer k satisfying Theorem 3.1. The constructive proof of the excluding grid theorem, given in [19] shows that $f(r) \leqslant 2^{5r^5 \log_2 r}$. Let τ be the tree-width of $G_{s,t}$, i.e., the graph obtained from G by deleting every edge not on a simple path from s to t. Let $r = \lfloor (\frac{1}{5} \log_2 \tau)^{1/6} \rfloor$. By the excluding grid theorem, $G_{s,t}$ contains an $r \times r$ mesh as minor since $\tau \geqslant f(r)$.

Now, any oblivous end-to-end protocol on $G_{s,t}$ requires headers of size as least as large as those required for the $r \times r$ mesh (recall that the end-to-end problem is closed under minor taking, as shown by Alder and Fich [1]). By Theorem 2.1, the headers for $r \times r$ meshes are of size at least $\Omega(\log r) = \Omega(\log \log \tau)$, which completes the proof of Theorem 1.1.

Remark. Any polynomial upper bound on $f(r)$ in r would prove a lower bound of $\Omega(\log \tau)$ on the header size. So far, the best known upper bound on $f(r)$ is $f(r) \leqslant 2^{9r^5}$ (cf. [34]). It is conjectured that the correct order of magnitude for $f(r)$ is $O(r^2 \log r)$.

3.2 Proof of Theorem 1.2

For $r \geqslant 0$, and $c > 0$, two integers, the $r \times c$ *cylinder* is the graph composed of r radial lines and c circles as shown on Fig. 4.

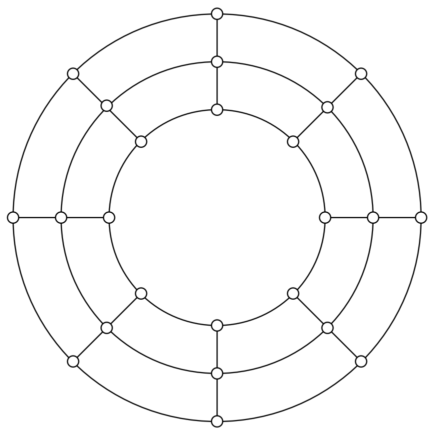

Figure 4: An 8×3 cylinder.

Theorem 3.2 (Robertson & Seymour [32], p. 62) *If G is planar and has an $r \times r$ cylinder as minor, then G has tree-width at most $\frac{3}{2}(r^2 + 2r) - 2$.*

It is clear that an $r \times r$ cylinder contains an $r \times r$ mesh as minor. Let $\tau = \mathrm{tw}(G_{s,t})$, and let $r = \lfloor \sqrt{\tau/5} \rfloor$ so that $\tau > \frac{3}{2}(r^2 + 2r) - 2$. Thus, $G_{s,t}$ contains the $r \times r$ mesh as minor. Therefore the header size for end-to-end communication in $G_{s,t}$ is at least the header size for end-to-end communication in the $r \times r$ mesh, that is at least $\Omega(\log r) = \Omega(\log \tau)$ by Theorem 2.1. This completes the proof of Theorem 1.2.

4 Conclusion

This paper provides lower bounds on the header-size for arbitrary and planar graphs. The design of matching upper bounds remains open. Note that the tree-width of a graph G can be alternatively defined as the minimum k for which G is a *partial k-tree*, i.e., a subtree of a k-tree, where k-trees are inductively defined as follows: A clique with $k+1$ vertices is a k-tree; given a k-tree T with n vertices, a k-tree with $n+1$ vertices is constructed by adding a new vertex x to T, x being adjacent to every node of a k-clique of T, and non adjacent to any of the $n-k$ other nodes of T. This definition connects the tree-width of a graph with its triangulation. Recall that a graph is triangulated if it contains no chordless cycle of length greater than three. A triangulation of a graph G is then a triangulated graph H with the same set of vertices and such that G is a subgraph of H. The tree-width of a graph G is then the maximum size (minus one) of a clique in a minimal triangulation of G, i.e., a triangulation of G with the minimum number of edges. Solving the end-to-end communication problem in k-trees may be the good way to solve the problem for arbitrary graphs, although it looks like a challenging task.

Acknowledgments. Both authors are thankful to Ioan Todinka for many helpful comments.

References

[1] M. Adler and F. Fich. The Complexity of End-to-End Communication in Memoryless Networks. In 18th ACM Symposium on Principles of Distributed Computing (PODC), pages 239-248, 1999.

[2] M. Adler, F. Fich, L. Goldberg, and M. Paterson. Tight Size Bounds for Packet Headers in Narrow Meshes. In 27th Int. Colloquium on Automata, Languages and Programming (ICALP), Springer, LNCS 1853, pages 756-767, 2000.

[3] Y. Afek, B. Awerbuch, and E. Gafni. Applying Static Network Protocols to Dynamic Networks. In 28th IEEE Annual Symposium on Foundations of Computer Science (FOCS), pages 358-370, 1987.

[4] Y. Afek, and E. Gafni. End-to-End Communication in Unreliable Networks. In 7th ACM Symposium on Principles of Distributed Computing (PODC), pages 131-148, 1988.

[5] Y. Afek, and E. Gafni. Bootstrap Network Resynchonization. In 10th ACM Symposium on Principles of Distributed Computing (PODC), pages 295-307, 1991.

[6] Y. Afek, H. Attiya, A. Fekete, M. Fischer, N. Lynch, Y. Mansour, D.-W. Wang, and L. Zuck. Reliable Communication Over Unreliable Channels. Journal of the ACM 41(6):1267-1297 (1994).

[7] Y. Afek, B. Awerbuch, E. Gafni, Y. Mansour, A. Rosén, and N. Shavit. *Slide*–The Key to Polynomial End-to-End Communication. Journal of Algorithms 22(1):158-186 (1997).

[8] A. Aho, J. Ullman, A. Wyler, and M. Yannakakis. Bounds on the Size and Transmission Rate of Communication Protocols. Computers and Math with Applications 8(3):205-214 (1982).

[9] S. Arnborg. Efficient Algorithms for Combinatorial Problems on Graphs with Bounded Decomposability–A Survey. BIT 25:2-23 (1985).

[10] S. Arnborg, D. Corneil, and A. Proskurowski. Complexity of Finding Embeddings in a k-Tree. SIAM J. Alg. Disc. Meth. 8:277-284 (1987).

[11] B. Awerbuch and S. Even. Reliable Broadcast Protocols in Unreliable Networks. Networks 16:381-396 (1986).

[12] B. Awerbuch, Y. Mansour, and N. Shavit. Polynomial End-to-End Communication. In 30th IEEE Annual Symposium on Foundations of Computer Science (FOCS), pages 358-363, 1989.

[13] B. Awerbuch, O. Goldreich, and A. Herzberg. A Quantitative Approach to Dynamic Networks. In 9th ACM Symposium on Principles of Distributed Computing (PODC), pages 189-203, 1990.

[14] K. Bartlett, R. Scantleburg, and P. Wilkinson. A Note on Reliable, Full-Duplex Transmission over Half-Duplex Links. Communication of the ACM 12:260-261 (1969).

[15] H. Bodlaender. A Tourist Guide through Treewidth. Acta Cybernetica 11:1-21 (1993).

[16] H. Bodlaender. A Linear-Time Algorithm for Finding Tree-Decompositions of Small Treewidth. In 25th ACM Symposium on Theory of Computing (STOC), pages 226-234, 1993.

[17] H. Bodlaender, J. Gilbert, H. Hafsteinsson, and T. Kloks. Approximating Treewidth, Pathwidth, Frontsize, and Shortest Elimination Tree. Journal of Algorithms 18:238-255 (1995).

[18] R. Diestel. Graph Theory (2nd edition). Springer-Verlag, New York, 2000.

[19] R. Diestel, T.R. Jensen, K.Y. Gorbunov, and C. Thomassen. Highly connected sets and the excluded grid theorem. Journal of Combin. Theory B 75:61-73 (1999).

[20] S. Dolev and J. Welch. Crash Resilient Communication in Dynamic Networks. IEEE Transactions on Computers 46(1):14-26 (1997).

[21] A. Fekete, and N. Lynch. The Need for Headers: An Impossibility Result for Communication over Unreliable Channels. In 1st Int. Conf. on Theories of Concurrency: Unification and Extension (CONCUR), Springer, LNCS 458, pages 199-215, 1990.

[22] A. Fekete, N. Lynch, Y. Mansour, and J. Spinelli. The Impossibility of Implementing Reliable Communication in the Face of Crashes. Journal of the ACM 40(3):1087-1107 (1993).

[23] F. Fich. End-to-End Communication Protocols. In 2nd Int. Conference on Principles of Distributed Systems (OPODIS), Hermes, pages 37-43, 1998.

[24] F. Fich and A. Jakoby. Short Headers Suffice for Communication in a DAG with Link Failures. In 14th International Symposium on Distributed Computing (DISC), Springer, LNCS, pages 360-373, 2000.

[25] O. Goldreich, A. Herzberg, and Y. Mansour. Source to Destination Communication in the Presence of Faults. In 8th ACM Symposium on Principles of Distributed Computing (PODC), pages 85-101, 1989.

[26] A. Herzberg. Connection-Based Communication in Dynamic Networks. In 11th ACM Symposium on Principles of Distributed Computing (PODC), pages 13-24, 1992.

[27] E. Kushilevitz, R. Ostrovsky, and A. Rosén. Log-Space Polynomial End-to-End Communication. In 28th ACM Symposium on Theory of Computing (STOC), pages 559-568, 1995.

[28] R. Ladner, A. LaMarca, and E. Tempero. Counting Protocol for Reliable End-to-End Transmission. Journal of Computer and System Sciences 56:96-111 (1998).

[29] Y. Mansour and B. Schieber. The Intractability of bounded Protocols for On-Line Sequence Transmission over Non-FIFO Channels. Journal of the ACM 39(4):783-799 (1992).

[30] J. Postel. Internet Protocol. Internet Engineering Task Force Request For Comments 791, 1981.

[31] B. Reed. Finding Approximate Separators and Computing Treewidth Quickly. In 24th ACM Symposium on Theory of Computing (STOC), pages 221-228 (1992).

[32] N. Robertson, and P.D. Seymour. Graph Minors. III. Planar Tree-Width. Journal of Combin. Theory B 36:49-64 (1984).

[33] N. Robertson, and P.D. Seymour. Graph Minors. V. Excluding a planar graph. Journal of Combin. Theory B 41:92-114 (1986).

[34] N. Robertson, P.D. Seymour, and R. Thomas. Quickly excluding a planar graph. Journal of Combin. Theory B 62:323-348 (1994).

[35] N. Stenning. A Data Transfer Protocol. Computer Networks 1(2):99-110 (1976).

[36] E. Tempero, and R. Ladner. Recoverable Sequence Transmission Protocols. Journal of the ACM 42(5):1059-1090 (1995).

[37] U. Vishkin. A Distributed Orientation Algorithm. IEEE Transaction on Information Theory 29, 624-629 (1983).

[38] D.-W. Wang, and L. Zuck. Tight Bounds for the Sequence Transmission Problem. In 8th ACM Symposium on Principles of Distributed Computing (PODC), pages 73-83, 1989.

[39] D.-W. Wang, and L. Zuck. Real-Time Sequence Transmission Problem. In 10th ACM Symposium on Principles of Distributed Computing (PODC), pages 111-124, 1991.