# Universal Routing Schemes

Pierre Fraigniaud and Cyril Gavoille *

Laboratoire de l'Informatique du Parallélisme - CNRS
École Normale Supérieure de Lyon
69364 Lyon cedex 07
France

**Abstract**

In this paper, we deal with the compact routing problem, that is implementing routing schemes that use a minimum memory size on each router. A *universal* routing scheme is a scheme that applies to all $n$-node networks. In [31], Peleg and Upfal showed that one can not implement a universal routing scheme with less than a total of $\Omega(n^{1+1/(2s+4)})$ memory bits for any given stretch factor $s \geq 1$. We improve this bound for stretch factors $s$, $1 \leq s < 2$, by proving that any near-shortest path universal routing scheme uses a total of $\Omega(n^2)$ memory bits in the worst-case. This result is obtained by counting the minimum number of routing functions necessary to route on all $n$-node networks.

Moreover, and more fundamentally, we give a tight bound of $\Theta(n \log n)$ bits for the *local* minimum memory requirement of universal routing scheme of stretch factors $s$, $1 \leq s < 2$. More precisely, for any fixed constant $\varepsilon$, $0 < \varepsilon < 1$, there exists an $n$-node network $G$ on which at least $\Omega(n^{\varepsilon})$ routers require $\Theta(n \log n)$ bits each to code any routing function on $G$ of stretch factor $< 2$. This means that, whatever you choose the routing scheme, there exists a network on which one can not compress locally the routing information better than routing tables do.

## 1 Introduction

The *general routing problem* in a network (as opposed to the *permutation routing problem* [28] or the *broadcasting problem* [13]) consists in finding a routing *protocol* or routing *function* such that, for any source-destination pair, any message from the source can be routed to the destination. $XY$-routing [4] or $e$-cube routing [5] are such protocols. The efficiency of a protocol is measured in terms of latency (related to the length of the paths) and/or throughput (related to link or node congestion). Finding shortest paths in a network is easy [22], and finding paths minimizing congestion can be done easily for a large class of networks (namely Cayley graphs) [24]. However, the routings protocols that are obtained may be of no use from a VLSI point of view. Indeed, the implementation of these protocols may require such a large amount of hardware that they can not be used in practice.

*Both authors are supported by the research programs ANM and PRS of the CNRS, and by the DRET of the DGA.

For instance, implementing $XY$ or $e$-cube routing is simple because routing is locally performed by respectively comparison and xor-ing of the local address with the destination address. However, networks do not necessarily have the simple structure of meshes or hypercubes. For instance routing in *pancake* networks requires to decompose any given permutation in a product of particular ones, and this is definitely not easy (see Problem 39 in [14]). Also, networks may even have no structure at all so that a routing algorithm can not be clearly defined on them. In these cases, *routing tables* are generally used: when a node receives a message, it consults its local table to determine the output port of the message. Such a process is fast, that is the latency of each router is small. However each router needs to store a large table of size $\Theta(n \log d)$ for a $d$-regular $n$-node network. Moreover, since routers must be replicated $n$ times in an $n$-node network, such a large local memory requirement implies a global information of $\Theta(n^2 \log d)$ bits for the whole network, just for routing.

Many authors propose methods to compress the routing tables (see for instance [16, 17, 32]). Among them, *interval routing* [3, 10, 11, 33, 32] was chosen by *Inmos* for its C104 routing chip [6]. A router finds the direction to forward a message by determining the interval (a set of consecutive addresses) that contains the destination address of the message, each interval being associated to one particular direction; eventually, more than one interval per direction can be required if the addresses using this link are not consecutive integers. Unfortunately, it was proven in [8, 11, 27] that the number of intervals per link necessary to route in any network is not bounded when using shortest path routing. Actually, it was recently proven in [19] that this number is [1]$\Theta(n)$ (it was even noticed in [26] that in fact $\Theta(n)$ routers require such an amount of intervals). Since the C104 routing chip allows a few intervals per link only, this means that there exist networks such that no shortest path routing protocol can be implemented on them using C104 chips. In this sense, this chip is not "universal", that is it can not be used for all possible networks.

To formalize this concept, let us define a *routing scheme* to be a function that returns a routing protocol $R$ (or a routing function) for any network $G$. For instance, the *shortest path interval routing scheme* is the selection, for any network $G$, of a shortest path routing function $R$ whose implementation minimizes the number of intervals per link. For any network, the obtained routing function $R$ may require a large number of intervals but it exists. The shortest path interval routing scheme is therefore said to be *universal* because it applies to all networks. On the contrary, the shortest path 1-interval routing scheme, that is the routing scheme that returns a shortest path routing function that can be implemented using a unique interval per link, is only *partial* in the sense that there exist networks on which this routing scheme is not defined [11].

The universality of a routing scheme is counterbalanced by the large memory that may require the routers to implement the routing functions of this scheme. For instance, it is proved in [19] that to be implemented on any network the shortest path interval routing scheme requires a router with $\Theta(n)$ items of size $O(\log n)$ bits. Partial routing schemes require less memory (for instance $O(\log n)$ memory bits for the shortest path 1-interval routing scheme on bounded degree networks), but they do not apply on all networks. Informally, the question about the size of a router could be: "what is the minimum size of a universal routing scheme providing routing functions satisfying some given constraint as shortest paths?". In other words, how much and in what situation is it possible to compress the routing tables? This problem is addressed in this paper.

In fact, Peleg and Upfal partially answered to this question in [31]. They showed that any shortest path universal routing scheme uses a total of $\Omega(n^{7/6})$ memory bits. Therefore, we can

[1] This number remains $\Theta(n)$ for 3-regular networks of order $O(n)$, see [20]

derive from their global lower bound that any shortest path universal routing scheme requires routers with at least $\Omega(\sqrt[6]{n})$ memory bits. In this paper, we improve the result of Peleg and Upfal by showing that any shortest path universal routing scheme uses a total of $\Omega(n^2)$ memory bits. Moreover, we show that most of the routers require $\Omega(n \log n)$ bits of local memory.

To be more precise, we do not consider only shortest path routing, but also routing for which the ratio length of the routing paths over length of shortest paths is bounded by a constant $s$ called the *stretch factor*. We show that any routing scheme of stretch factor $s < 2$ uses a total of $\Omega(n^2)$ memory bits. Hence, for small stretches, our result improves the lower bound of Peleg and Upfal that is $\Omega(n^{1+1/(2s+4)})$ bits for any stretch factor $s$ [31]. Moreover, we also prove that, for any stretch factor $s < 2$, some routers may require $\Omega(n \log n)$ bits of local memory. This shows that routing information needs (asymptotically) the same number of bits as the routing tables to be coded in some routers.

In the following section we describe the model of routing functions considered in this paper, and we introduce the definitions of *global* and *local memory requirement* for a routing function, and for a graph. Section 3 states a $\Omega(n^2)$ lower bound on the global memory requirement for universal routing schemes. This lower bound is based on the number of non isomorphic routing functions. In Section 4, we use the same technique to prove a lower bound on the global memory requirement of bounded degree graphs. Section 5 introduces a general tool to prove lower bounds on the local memory requirement, namely the generalized matrices of constraints. This tool allows us to derive a tight bound of $\Theta(n \log n)$ bits for the local memory requirement. Finally, Section 6 summarizes the state of the art of memory requirement of universal routing schemes.

Note that simpler arguments to derive the $\Omega(n^2)$ bound on the global memory requirement have been recently presented in [26]. These arguments are based on constructions taken from [12], but make use of the Kolmogorov Complexity theory rather than counting arguments.

## 2 Statement of the Problem

We consider the standard model of point-to-point communication networks described as finite connected symmetric digraphs $G = (V, E)$. In the following, we denote by $n$ the number of vertices of such a graph. The vertices represent the processors or the nodes of the network, and the edges represent bidirectional communication links between the nodes (to each edge corresponds two symmetric arcs). A vertex can directly communicate with its neighbors only, and messages between nonadjacent vertices are sent along a path connecting them in the network. We denote by $\deg(x)$ the degree of the vertex $x$ in $G$. In the following, we always refer to *finite connected symmetric digraph* by the simple term *graph*.

We use a general definition of routing function similar to the one given in [31].

**Definition 1 (Routing Function)** *Let $G = (V, E)$ be any graph. Let $\mathcal{H}$ be any finite set called* header *set. A* routing function *on $G$ is a triple $(I, H, P)$ of functions,*

- *$I$ is the* initialization *function, $I : V \times V \to \mathcal{H}$,*
- *$H$ is the* header *function, $H : V \times \mathcal{H} \to \mathcal{H}$, and*
- *$P$ is the* port *function, $P : V \times \mathcal{H} \to E \cup \{\epsilon\}$,*

*such that, for every pair* $(x, y)$ *of vertices of* $G$, $x \neq y$, *there exists a family of* $k+1$ *vertices* $\{v_i \,|\, v_i \in V, 0 \leq i \leq k\}$ *and a family of* $k+1$ *headers* $\{h_i \,|\, h_i \in \mathcal{H}, 0 \leq i \leq k\}$ *such that* $v_0 = x$, $v_k = y$, $h_0 = I(x, y)$ *and, for every* $i \in \{0, \ldots, k-1\}$, $H(v_i, h_i) = h_{i+1}$, $P(v_i, h_i) = (v_i, v_{i+1})$ *and* $P(v_k, h_k) = \epsilon$.

In other words, if $R = (I, H, P)$ is a routing function on a graph $G$, the path built by $R$ is the following. If the vertex $x$ wants to send a message $M$ to the vertex $y$ in $G$, it prepares a header $h$ for the message $M$ by computing $h = I(x, y)$, and attaches it to $M$. Then, it selects the output port $p$ by computing $p = P(x, h)$, and sends the message out by this port. Now, suppose that a message $M$ with a header $h'$ arrives at some vertex $z$. The first task of $z$ consists in computing a new header $h'' = H(z, h')$. Then, $z$ determines the output port by computing $p' = P(z, h'')$. If $p' = \epsilon$, then the routing process ends ($M$ is sent to the memory of the corresponding processor). Otherwise, $z$ forwards $M$ on the output port $p'$ after having replaced the old header $h'$ attached to $M$ by the new one $h''$.

**Notation.** The distributed representation of $R$ is the collection $\{R_x = (I_x, H_x, P_x) \,|\, x \in V\}$ such that, for every $x, y \in V$ and $h \in \mathcal{H}$, $I_x(y) = I(x, y)$, $H_x(h) = H(x, h)$ and $P_x(h) = P(x, h)$. For every $x \in V$, $R_x$ is called the *local routing function* of $R$ in $x$. The so called "stop-value" $\epsilon$ ends the protocol of the routing, but $\epsilon$ can be interpreted locally as a constant that is the label of the special link connecting the processor assigned to the router.

We consider routing functions such that every message follows a path that depends only on the source and the destination of this message. However, the result returned by a local header function $H_x$ depends on the current header attached to the message: the header of a message can be modified all along its route. Since the result returned by a local port function $P_x$ depends on the header, it allows to employ flexible *trial-and-error* routing functions, that is paths induced by the routing can contain loops. This property may allow to decrease the local memory requirement for the storage of routing information because it allows to visit many nodes to get more information attached on the header.

Our model is quite general, and capture many particular cases. For instance, our lower bounds on the memory requirement also apply to the usual routing tables that are routing functions $R = \{R_x = (Identity, Identity, P_x) \,|\, x \in V\}$ on a graph $G = (V, E)$ with $\mathcal{H} = V$, that is the headers correspond to the names of the destinations.

We recall now the definition of the *stretch factor* introduced in [31]:

**Definition 2 (Stretch Factor)** *Let* $R$ *be a routing function on a graph* $G$. *For every pair* $(x, y)$ *of vertices of* $G$, *we denote by* $d_G(x, y)$ *the distance between* $x$ *and* $y$ *in* $G$, *and we denote by* $d_R(x, y)$ *the length of the routing path built by* $R$ *between* $x$ *and* $y$. *The* stretch factor *of* $R$ *on* $G$ *is denoted by* $s(R, G)$ *and satisfies*

$$s(R, G) = \max_{x \neq y} \frac{d_R(x, y)}{d_G(x, y)}$$

The stretch factor has at least two main interests. First, it measures the efficiency of the routing, better than considering the length of the longest path only. Furthermore, the stretch factor allows to tune the tradeoff between space (needs of collecting informations) and the time (length of the paths).

The compact routing problem is related to the following definition:

**Definition 3 (Memory Requirement of a Router)** *Let $G$ be any graph and let $x$ be any vertex of $G$. For every routing function $R$ on $G$, we denote by $\mathrm{MEM}(G, R, x)$ the minimum memory requirement of $R$ in $x$, i.e., the minimum number of bits necessary to store the local function $R_x$.*

The parameter $\mathrm{MEM}(G, R, x)$ is simply the Kolmogorov complexity [25] of $R_x$, that is the size of the smallest program that writes $R_x$ on its output. Of course, it is defined for a fixed coding strategy. From the definition above, the memory requirement does not take into account the size of the header. Indeed, to be as general as possible, we assume headers of unbounded size.

Definition 3 allows us to define the local and global memory requirement as follows.

**Definition 4 (Global and Local Memory Requirement)** *Let $R$ be a routing function on a graph $G$. We define:*

- *the* global memory requirement *as* $\mathrm{MEM}_{global}(G, R) = \sum_x \mathrm{MEM}(G, R, x)$
- *the* local memory requirement *as* $\mathrm{MEM}_{local}(G, R) = \max_x \mathrm{MEM}(G, R, x)$

For every stretch factor $s$, the *global memory requirement* of $G$ is defined as $\mathrm{MEM}_{global}(G, s) = \min_R \mathrm{MEM}_{global}(G, R)$, and the *local memory requirement* of $G$ is defined as $\mathrm{MEM}_{local}(G, s) = \min_R \mathrm{MEM}_{local}(G, R)$, where the minimum is taken over all the routing functions $R$ on $G$ of stretch factor at most $s$, and such that vertex-labels are in $\{1, \ldots, n\}$, and port-labels of every vertex $x$ are in $\{1, \ldots, \deg(x)\}$.

It is interesting to consider both global and local memory requirements because it is possible that $\mathrm{MEM}_{global}(G, s) \ll n\,\mathrm{MEM}_{local}(G, s)$ for some graph $G$ of $n$ vertices and for some stretch factor $s$. In this paper, we focus our attention on the worst-cases on $\mathrm{MEM}_{global}(G, s)$ and $\mathrm{MEM}_{local}(G, s)$, for small values of $s$. For every integer $n$, we define $\mathrm{MEM}_{global}(n, s) = \max_G \mathrm{MEM}_{global}(G, s)$ and $\mathrm{MEM}_{local}(n, s) = \max_G \mathrm{MEM}_{local}(G, s)$, where the maximum is taken over all the graphs $G$ of order $n$.

For example, $\mathrm{MEM}_{local}(H_n, 1) = O(\log n)$, where $H_n$ is the hypercube of order $n$, by using $e$-cube routing. It is known that, for all acyclic graphs [32], for all outer-planar graphs [15], and for all unit circular-arc graphs [11] $G$ of order $n$ and of maximum degree $d$, $\mathrm{MEM}_{local}(G, 1) = O(d \log n)$ because the 1-interval routing scheme applies to these graphs (1 interval per arc). For every chordal graph $G$, it is shown in [29] and [31] that $\mathrm{MEM}_{global}(G, 3) = O(n \log^2 n)$. Let us consider the complete graph $K_n$ of order $n$. In general, in a vertex $x$, a local routing function $R_x$ on $K_n$ is stored with $\Theta(n \log n)$ bits for a random labeling of ports of $x$, or a port labeling chosen by an adversary. Indeed, an adversary can choose some permutation $\pi$ between the $n-1$ port-labels and the $n-1$ neighbors of $x$ in such a way that reaching any neighbor of $x$ implies to know the full description of $\pi$, i.e., $\lceil \log_2((n-1)!) \rceil = \Theta(n \log n)$ bits. However, some routing function $R'_x$ on $K_n$ can be stored in $x$ in a more compact way using $O(\log n)$ bits. For example, vertices can be labeled by integers from $\{1, \ldots, n\}$, and port by integers from $\{1, \ldots, n-1\}$ such that the ports connecting vertices $x$ and $y$ is labeled $y$ if $x > y$, and $y - 1$ if $y > x$. Note that we do not make difference between the vertex $x$ and its label. If headers are labels of destination, the routing function in $x$, $R'_x = (Identity, Identity, P'_x)$, is defined by its port function $P'_x$. It executes the

following algorithm: if the destination is $x$ then the message is arrived, otherwise the message is forwarded on the port that has the same name as the destination if $x > y$, and on the port $y - 1$ otherwise. Clearly $R' = \{R'_x = (I'_x, H'_x, P'_x) \mid x \in V\}$ defines a shortest path routing function on $K_n$, and the previous algorithm can be described with only $O(\log n)$ bits for the label $x$. Therefore, we have $\mathrm{MEM}_{local}(K_n, 1) = O(\log n)$. In general we are interesting to find the "best" routing function on a given graph, that is the best vertex and port labeling, and the best choice of routing paths that minimize coding of the routing function.

Our first main contribution is the following:

**Theorem 1** *For every $s$, $1 \leq s < 2$, $\mathrm{MEM}_{global}(n, s) = \Omega(n^2)$.*

Note that Theorem 1 directly implies that, for every $s$, $1 \leq s < 2$, $\mathrm{MEM}_{local}(n, s) = \Omega(n)$. However, we will see in Section 5 that, in the general case, the whole routing information can not be balanced between the $n$ nodes of the networks and therefore $\Omega(n)$ is not a tight lower bound for $\mathrm{MEM}_{local}$. Theorem 1 is based on counting the number of non isomorphic routing functions. This is done in the next section that also contains the proof of Theorem 1 (see Section 3.4).

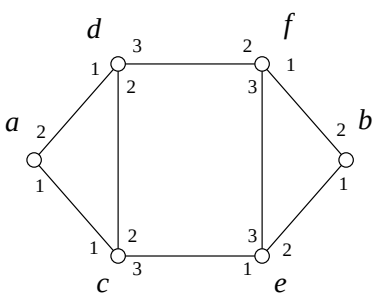
Our next contribution states that:

**Theorem 2** *For every $s$, $1 \leq s < 2$, $\mathrm{MEM}_{local}(n, s) = \Theta(n \log n)$. More precisely, for any fixed constant $\varepsilon$, $0 < \varepsilon < 1$, there exists an $n$-node network $G$ on which at least $\Omega(n^{\varepsilon})$ routers require $\Theta(n \log n)$ bits each to code any routing function on $G$ of stretch factor $< 2$.*

Theorem 2 is based on a precise study of the memory requirement of small sets of vertices of some particular types of graphs. This study is done in Section 5 that also contains the proof of Theorem 2 (see Section 5.3).

# 3 Number of Routing Functions

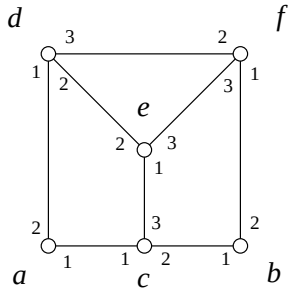
As said in [31], the same routing function can be used on several non isomorphic graphs because the neighbors of a vertex $x$ can be specified by the local port function $P_x$ rather than the one-to-one labeling of the set of arcs. Indeed, it is not space efficient for a local port function $P_x$ to return an absolute label of the output arc, but it saves space to return a local label of this port between 1 and $\deg(x)$. The example on Figure 1 shows two *isomorphic* routing functions of stretch factor 1.5 on two non isomorphic graphs, $G = (V, E)$ and $G' = (V', E')$. The two routing functions are represented by a unique routing table $T$. In this table, the output port $\epsilon$ that connects the router to the memory of its associated processor is labeled 0. In this example $\mathcal{H} = V = V' = \{a, b, c, d, e, f\}$. The entry $T_{x,y}$ of the routing table $T$ returns the local port function $P_x(y)$ (that is an integer in $\{1, \ldots, \deg(x)\}$ in this example). $T$ represents a routing function of type $R = (Identity, Identity, P)$. To simplify the drawing, figures and graphs are drawn as undirected although each edge represents symmetric arcs. For instance, the routing path built by $R$ on $G$ between vertices $c$ and $b$ is $c, d, f, b$ using successively ports 2, 3 and 1. The stretch factor of $R$ on $G$ is 1.5 because $d_R(x, y)$ is minimum for every couple of vertices $(x, y)$ of $G$ but $(c, b)$ for which $d_R(c, b) = 3$ whereas $d_G(c, b) = 2$. On $G'$, $R'$ represented by the same table is also of stretch 1.5 because $d_R(x, y)$ is minimum but for $(a, b)$, $(e, b)$, and $(c, d)$ for which $d_R(a, b) = d_R(e, b) = d_R(c, d) = 3$ whereas $d_{G'}(a, b) = d_{G'}(e, b) = d_{G'}(c, d) = 2$.




$s(R, G) = 1.5$

| $\nearrow$ | $a$ | $b$ | $c$ | $d$ | $e$ | $f$ |
|---|---|---|---|---|---|---|
| $a$ | 0 | 2 | 1 | 2 | 1 | 2 |
| $b$ | 1 | 0 | 1 | 2 | 1 | 2 |
| $c$ | 1 | 2 | 0 | 2 | 3 | 3 |
| $d$ | 1 | 3 | 2 | 0 | 2 | 3 |
| $e$ | 1 | 2 | 1 | 2 | 0 | 3 |
| $f$ | 2 | 1 | 3 | 2 | 3 | 0 |

$T$




$s(R, G') = 1.5$

Figure 1: Two nonisomorphic graphs $G$ and $G'$ supporting the same routing function $R$ (described by the table $T$), both with stretch factor 1.5.

**Remark.** Note that the number of paths induced by $R$ that are not shortest are different in $G$ and $G'$. More generally, two isomorphic routing functions on two nonisomorphic graphs have not necessarilly the same stretch factor.

Let us now define more formally what "isomorphic routing functions" means.

**Definition 5 (Isomorphic Routing Function)** *Let $R = (I, H, P)$ be any routing function on a graph $G$ with header set $\mathcal{H}$, and $R' = (I', H', P')$ be any routing function on a graph $G'$ with header set $\mathcal{H}'$. The function $R$ is* isomorphic *to $R'$ if and only if there exist three bijections $\varphi$, $\psi$ and $\gamma$ where $\varphi : V \to V'$, $\psi : E \to E'$, $\gamma : \mathcal{H} \to \mathcal{H}'$ such that for every $x, y \in V$, $x \neq y$, and for every $h \in \mathcal{H}$:*

(1) $I'(\varphi(x), \varphi(y)) = \gamma \circ I(x, y)$
(2) $H'(\varphi(x), \gamma(h)) = \gamma \circ H(x, h)$
(3) $P'(\varphi(x), \gamma(h)) = \psi \circ P(x, h)$

Informally, the vertices and the arcs of both graphs $G$ and $G'$, so as the headers of $\mathcal{H}$ and $\mathcal{H}'$, can be labeled in such a way that a same table $T$ (as the one of Figure 1) can be used for both routing functions.

Intuitively, the number of nonisomorphic graphs that support the same routing function raises with the stretch factor. More precisely, Peleg and Upfal have shown in [31] that the number of nonisomorphic routing functions of stretch factor at most $s$ is at least $2^{\Omega(n^{1+1/(2s+4)})}$. This implies $\mathrm{MEM}_{\mathit{global}}(n, s) = \Omega(n^{1+1/(2s+4)})$, for every $s$. Indeed, every routing scheme has to distinguish each of the nonisomorphic routing functions. We will show that, in fact, there are $2^{\Theta(n^2)}$ nonisomorphic routing functions of stretch factor at most 2. The formalization of counting nonisomorphic routing functions is based on the notion of routing scheme:

**Definition 6 (Routing Scheme)** *Let $n$ be any integer. Let $\mathcal{G}$ be any set of graphs of order $n$, and let $\mathcal{R}$ be any set of routing functions. A* routing scheme *is a function $RS : \mathcal{G} \to \mathcal{R}$ such that, for every graph $G \in \mathcal{G}$, $RS(G)$ is a routing function of $\mathcal{R}$ on $G$. Moreover $RS$ is said to be a* universal routing scheme *if $\mathcal{G}$ is the set of all the graphs of order $n$.*

Table-routing is a universal routing scheme since any graph support a routing function that can be coded by a table. On the other hand, as we saw in the introduction, 1-interval routing scheme is not a universal routing scheme. The definition of routing scheme allows to derive a lower bound on the number of nonisomorphic routing functions. Indeed, the minimum cardinality of the range of $RS$ over all universal routing schemes $RS$ from $\mathcal{G}$ to $\mathcal{R}$ is the number of nonisomorphic routing functions of the class $\mathcal{R}$.

We will construct a large class of graphs denoted by $\mathcal{G}_n$ such that two different graphs of $\mathcal{G}_n$ do not support two isomorphic routing functions of stretch factor $< 2$. To construct the class $\mathcal{G}_n$, composed of the so called *graphs of constraints*, we will extend the notion of *matrices of constraints* introduced in [8, 19]. Each matrix of constraints of a graph represents a set of constraints on the paths that must be followed by any routing function of stretch factor $s$.

### 3.1 Matrices of Constraints

**Definition 7 (Matrix of Constraints)** *Let $G = (V, E)$ be a graph, let $V' \subset V$ and $E' \subset E$. Let $s$ be any real $\geq 1$. A* matrix of constraints *of $E'$ on $V'$ of stretch factor $s$ in $G$ is a $|V'| \times |E'|$ boolean matrix $M = (m_{v,e})$, $v \in V', e \in E'$, such that the rows of $M$ are labeled by the vertices of $V'$, and the columns of $M$ are labeled by the arcs of $E'$. Morever $M$ must satisfy that $\forall v \in V', \forall e = (x, y) \in E'$,*

- *$m_{v,e} = 1 \Leftrightarrow$ all paths of length at most $s \cdot d_G(x, v)$ from $x$ to $v$ in $G$ use the arc $e$;*
- *$m_{v,e} = 0 \Leftrightarrow$ no path of length at most $s \cdot d_G(x, v)$ from $x$ to $v$ in $G$ uses the arc $e$.*

In general, any two subsets $V' \subset V$, and $E' \subset E$ do not define a matrix of constraints for $G$, because it can exists for example many shortest paths between $x$, the tail of $e \in E'$, and $v \in V'$ some using $e$, and other not using $e$. Given a graph $G$, and a $p \times q$ boolean matrix of constraints $M$ of $G$, every other matrix that belongs to the equivalence class of $M$, using row and column permutation as a congruence operator, is also a matrix of constraints of $G$.

**Definition 8 (Index, and Canonical Representative)** *Let $M = (m_{i,j})$ be a $p \times q$ boolean matrix, $i \in \{1, \ldots, p\}$ and $j \in \{1, \ldots, q\}$. The* index *of $M$ is the positive integer*

$$\mu(M) = \sum_{1 \leq i \leq p, 1 \leq j \leq q} m_{i,j} 2^{(p-i)+p(q-j)}$$

*For any set $S$ of boolean matrices, the* canonical representative *of $S$ is the unique boolean matrix $M_0 \in S$ such that $\mu(M_0) = \max_{M \in S} \mu(M)$.*

The index $\mu(M)$ of a $p \times q$ boolean matrix $M$ can be seen as the value of the binary expression of $M$ obtained by concatenating the $q$ words of $p$ bits formed by the columns of $M$. The canonical representative of a set $S$ is the matrix that has the largest index among the matrices in $S$.

**Notation.** For every pair of integers $(p, q)$, $\mathcal{M}_{p,q}$ denotes the set of $p \times q$ boolean matrices $M$ such that $M$ is the canonical representative of its equivalence class of matrices under row permutations, column permutations, column bit-complementation. Moreover, by definition, $\mathcal{M}_{p,q}$ does not contain

matrices whose some column contains exactly one 0-entry. For example:

$$\mathcal{M}_{4,2} = \left\{ \begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \begin{bmatrix} 1 & 1 \\ 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \right\} \tag{1}$$

**Lemma 1** *For every pair of integers* $(p, q)$,

$$\frac{(2^p - p)^q}{p!\, q!\, 2^q} \leq |\mathcal{M}_{p,q}| = 2^{\Omega(pq - p\log p - q\log q)}$$

**Proof.** There are $(2^p - p)^q$ matrices without column having a single 1-entry. Therefore, there are at least $(2^p - p)^q/(p!q!)$ different classes of matrices without single 1-entry on each column, even up to a permutation of rows and columns. Now, there are a most $2^q$ possible ways of bit-complementing the $q$ columns of a $p \times q$ boolean matrix. □

For more details about the number of classes of boolean matrices, see [23].

Let $\sim$ be the following equivalence relation of matrices: $A \sim B$ if and only if $A$ can be obtained from $B$ by row and column permutations. In the following, if $A$ and $B$ are two $p \times q$ boolean matrices, then we denote by $[AB]$ the concatenation of the matrices $A$ and $B$, that is a $p \times 2q$ boolean matrix whose $q$ first columns are the columns of $A$, and the $q$ last columns are the columns of $B$.

The following lemma will be useful for counting the number of non isomorphic routing functions.

**Lemma 2** *Let* $M, M' \in \mathcal{M}_{p,q}$. *If* $M \neq M'$ *then* $[M\overline{M}] \not\sim [M'\overline{M'}]$.

**Proof.** Let $(p, q)$ be any pair of integers, and $M, M' \in \mathcal{M}_{p,q}$ such that $M \neq M'$. By definition of the set $\mathcal{M}_{p,q}$, $M \neq M' \Rightarrow M \not\sim M'$. Let us show that

$$\max_{X \sim [M\overline{M}]} \mu(X) = \mu([MY]) \tag{2}$$

for a suitable $p \times q$ boolean matrix $Y$. Assume by contradiction that $\max_{X \sim [M\overline{M}]} \mu(X) = \mu([NZ])$ with $N \neq M$. Thus, there exists a permutation $r$ of rows of $[M\overline{M}]$, and a permutation $c$ of columns, such that $[NZ] = r(c([M\overline{M}]))$ and $\mu([NZ]) > \mu([MY])$, for any matrix $Y$ obtained from $\overline{M}$ by column permutations. In the following, we denote by $m_i$ a column of $M$, $i \in \{1, \ldots, q\}$, and by $\overline{m_i}$ the corresponding column in the matrix $\overline{M}$. Since $N \neq M$, $N$ has a column $\overline{m_j}$ up to a permutation of its rows. Moreover, the column $m_j$ can not belong to the matrix $Z$ because $M \in \mathcal{M}_{p,q}$, and the matrix obtained from $M$ by complementing the column $j$ and applying the row permutation $r$ would yield a greater index than $\mu(M)$. Thus, $N$ contains both columns $m_j$ and $\overline{m_j}$, and therefore the matrix $Z$ contains a column $m_k$ of $M$. We have $\mu(m_j) > \mu(m_k)$ (each column is considered as a $p \times 1$ matrix), otherwise the matrix obtained from $[NZ]$ by exchanging the columns $m_j$ and $m_k$ would have an index greater than $\mu([NZ])$. Similarly $\mu(\overline{m_j}) > \mu(m_k)$. Since $\mu(M)$ is maximum under row, column and column bit-complementation, it follows that $\mu(m_k) > \mu(\overline{m_k})$. Thus, $\mu(\overline{m_j}) > \mu(m_k) > \mu(\overline{m_k})$, i.e., $\mu(\overline{m_j}) > \mu(\overline{m_k})$. That is a contradiction with $\mu(m_j) > \mu(m_k)$.

We conclude that all the columns of $N$ are the columns of $M$. Now, it is easy to see that $r$ is necessarily the identity function, otherwise $\mu(M)$ would not be maximum. Thus $N = M$, and we get Equation 2. Similarly $\max_{X \sim (M'\overline{M'})} \mu(X) = \mu([M'Y'])$, for a suitable $p \times q$ boolean matrix $Y'$. Since $\mu(M) \neq \mu(M')$ implies $\mu([MY]) \neq \mu([M'Y'])$, $\max_{X \sim [M'\overline{M'}]} \mu(X) \neq \max_{X \sim [M\overline{M}]} \mu(X)$, i.e., $[M\overline{M}] \not\sim [M'\overline{M'}]$. □

### 3.2 Graphs of Constraints

We will show that there is a bijective correspondence between the set of matrices of constraints and a particular set of graphs. For any symmetric digraph $G = (V, E)$, we denote by $\{a, b\}$ the pair of arcs $\{(a, b), (b, a)\} \subset E$.

**Definition 9** *Let $M = (m_{i,j})$ be any $p \times q$ boolean matrix, $i \in \{1, \ldots, p\}$ and $j \in \{1, \ldots, q\}$. The* graph of constraints *of $M$ is a symmetric digraph $G = (V, E)$ such that:*

- $V = \{a_1, \ldots, a_q\} \cup \{b_1, \ldots, b_q\} \cup \{c_1, \ldots, c_q\} \cup \{v_1, \ldots, v_p\}$;
- $\{c_j, a_j\}$ *and* $\{c_j, b_j\} \in E$, *for every* $j \in \{1, \ldots, q\}$;
- *for every* $i \in \{1, \ldots, p\}$ *and* $j \in \{1, \ldots, q\}$, $\{a_j, v_i\} \in E$ *if and only if* $m_{i,j} = 0$;
- *for every* $i \in \{1, \ldots, p\}$ *and* $j \in \{1, \ldots, q\}$, $\{b_j, v_i\} \in E$ *if and only if* $m_{i,j} = 1$;
- *no other arc belongs to* $E$.

Equation 3 below shows three examples of graphs of constraints (see the set denoted $\mathcal{G}_{4,2}$), one for each matrix of the set $\mathcal{M}_{4,2}$ of Equation 1. One can check that these three graphs do not support isomorphic routing function. We will see that this result holds in general for matrices in $\mathcal{M}_{p,q}$. Again, the following lemma will be useful for our proof.

**Lemma 3** *Let $M$ be any $p \times q$ boolean matrix, and let $G = (V, E)$ be its graph of constraints. Then $M$ is a matrix of constraints of stretch factor $< 2$ of $G$. Moreover*

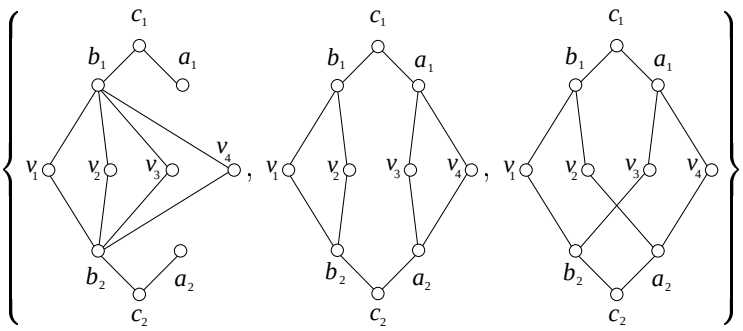
- $|V| = p + 3q$;
- *For every* $j \in \{1, \ldots, q\}$, $\deg(a_j) + \deg(b_j) = p + 2$ *and* $\deg(c_j) = 2$;
- *For every* $i \in \{1, \ldots, p\}$, $\deg(v_i) = q$.

**Proof.** Consider the graph of constraints $G$ of a $p \times q$ boolean matrix $M$. Let $A$ be the set of arcs $(c_j, b_j)$, $j \in \{1, \ldots, q\}$, and let $B = \{v_1, \ldots, v_p\}$. Clearly, by definition of $G$, $M$ is a matrix of constraints of $A$ on $B$ in $G$ of stretch factor $< 2$. Indeed, every arc $e \in A$ is a bridge or belongs to chord-less cycle of length at least 6, and there exists a unique arc $e \in A$ such that $d_G(c_j, v_i) = 2$. Thus, if a routing function $R$ on $G$ uses arc $e$ when it should not, or does not use $e$ when it should, then $d_R(c_j, v_i) \geq d_G(c_j, v_i) + 2$. Indeed, if $e$ is not a bridge, the path constructed by $R$ could go to a wrong neighbor, then go back in $c_j$, and finally go to $v_i$ by a shortest path. Therefore the stretch factor of $R$ will be at least $(2 + 2)/2 \geq 2$. □

**Remark.** These graphs have a diameter at most 6, and thus have an unbounded degree.

For any pair of integers $(p, q)$, we denote by $\mathcal{G}_{p,q}$ the set of graphs of constraints of all the matrices of $\mathcal{M}_{p,q}$. Equation 3 gives the set $\mathcal{G}_{4,2}$ where the graphs are represented undirected.

$$\mathcal{G}_{4,2} = \left\{ \right. \quad , \quad , \quad \left. \right\} \tag{3}$$




One can check that the set $\mathcal{G}_{p,q}$ contains graphs that are pairwise non isomorphic. For the purpose of our proof however, we need something stronger, that is they do not support isomorphic routing functions of stretch factor $< 2$. This is proved in the next section.

### 3.3 Cardinality of the Range of a Universal Routing Scheme

For every function $f : A \to B$, we denote by $f(A)$ the range of $f$, i.e., $f(A) = \{y \in B \mid \exists x \in A, y = f(x)\}$.

**Theorem 3** *Let $\mathcal{R}$ be the set of routing functions of stretch factor $< 2$ on the class $\mathcal{G}$ of graphs of order $n$. For any universal routing scheme $RS : \mathcal{G} \to \mathcal{R}$,*

$$|RS(\mathcal{G})| = 2^{\Theta(n^2)}$$

In a sense, we can not do "better" because it is well-known that the number of non isomorphic graphs of order $n$ is roughly $2^{\binom{n}{2}}/n! = 2^{\Theta(n^2)}$ (see [34]).

**Proof.** Let $RS$ be any universal routing scheme of $\mathcal{G}$ on $\mathcal{R}$. $\mathcal{R}$ is composed of routing functions of stretch factor $< 2$. Consider, for every integer $n \geq 20$, the subset of graphs $\mathcal{G}_n \subset \mathcal{G}$ such that $\mathcal{G}_n = \mathcal{G}_{p,q}$ where $p = \lfloor n/4 \rfloor - 3 + ((n \bmod 4) \bmod 3)$ and $q = p + 4\lfloor (n+5)/4 \rfloor - n$. Since the order of the graphs in $\mathcal{G}_{p,q}$ is $p + 3q$ (Lemma 3), then, for every $n \geq 20$, the order of the graphs in the set $\mathcal{G}_n$ is exactly $n$. Let $M, M' \in \mathcal{M}_{p,q}$, $M \neq M'$. Let $G = (V, E)$ and $G' = (V', E')$ be the two corresponding graphs of constraints in $\mathcal{G}_n$. For every routing function $R = (I, H, P)$ of header set $\mathcal{H}$ on $G$, and for every routing function $R' = (I', H', P')$ of header set $\mathcal{H}'$ on $G'$, we will show that if $s(R, G) < 2$ and $s(R', G') < 2$, then $R$ is non isomorphic to $R'$. We partition the set of vertices $V$ of $G$ in four subsets $A$, $B$, $C$, and $D$, grouping respectively the vertices $a_j$'s, $b_j$'s, $c_j$'s and $v_i$'s of $G$, $j \in \{1, \ldots, q\}$ and $i \in \{1, \ldots, p\}$ (see Definition 9). We obtain $A'$, $B'$, $C'$, and $D'$ from the vertices of $V'$ in the same way.

Assume that $R$ and $R'$ are isomorphic. Then there exist three bijections $\varphi$, $\psi$ and $\gamma$ such that for every $x, y \in V, x \neq y$, and for every $h \in \mathcal{H}$, Equations (1), (2) and (3) of Definition 5 are satisfied. Let $x \in V$ and $x' = \varphi(x) \in V'$. Let us show that $x$ and $x'$ must have the same degree: assume without loss of generality that $\deg(x) > \deg(x')$. From Equation (3) of Definition 5, this implies that an arc $(x, y)$ of $E$ is not used by $R$. Now, graphs of constraints are without loops and

multi-arcs, therefore the path from $x$ to $y$ in $G$ induced by $R$ is of length at least 2: a contradiction with the fact that $s(R,G) < 2$. Thus if $R$ and $R'$ are isomorphic, then $x$ and $\varphi(x)$ have the same degree, for every $x \in V$.

From Lemma 3, and by definition of the parameters $p$ and $q$ as functions of $n$, since $\lfloor n/4 \rfloor - 3 + ((n \bmod 4) \bmod 3) \geq 2$ and $4\lfloor (n+5)/4 \rfloor - n \geq 2$, for any $n \geq 20$, we get that $\deg(c) < \deg(t) < \deg(v)$, for any vertices $c \in C \cup C'$, $v \in D \cup D'$, and $t \in A \cup B \cup A' \cup B'$. Therefore $\varphi(C) = C'$, $\varphi(D) = D'$, and $\varphi(A \cup B) = A' \cup B'$. Let $Z = \{(c,t) \in E \mid c \in C \text{ and } t \in A \cup B\}$ be the set of arcs of $E$ from vertices of $C$ to vertices of $A \cup B$, and let $Z' = \{(c,t) \in E' \mid c \in C' \text{ and } t \in A' \cup B'\}$. Since $\varphi(C) = C'$, we get from Equation (3) of Definition 5 that $\psi(Z) = Z'$. Indeed, for every $c \in C$ and for every $h \in \mathcal{H}$, $P(c,h) \in Z$ and $P'(\varphi(c), \gamma(h)) \in Z'$. Let $N = [M\overline{M}]$ be the $p \times 2q$ boolean matrix composed of the matrix $M$ and its bit complement matrix $\overline{M}$. Let $N' = [M'\overline{M'}]$ be the $p \times 2q$ boolean matrix obtained similarly from $M'$. The matrix $N$ (respectively $N'$) is a matrix of constraints of $Z$ on $D$ in $G$ (respectively a matrix of constraints of $Z'$ on $D'$ in $G'$). Let $v \in D, e = (c,t) \in Z$ and $h_{c \to v} = I(c,v)$. The header $h_{c \to v}$ is the header generated by the routing function $R$ to send a message from $c$ to $v$ in $G$. $\gamma(h_{c \to v}) = \gamma \circ I(c,v) = I'(\varphi(c), \varphi(v))$ from Equation (1) of Definition 5. Thus $\gamma(h_{c \to v}) = h'_{\varphi(c) \to \varphi(v)}$ is the header generated by the routing function $R'$ to send a message from $\varphi(c)$ to $\varphi(v)$ in $G'$. Now, from the definition of $N = (n_{v,e})$, for every $v \in D$, for every $e = (c,t) \in Z$, we have

$$n_{v,e} = 1 \Leftrightarrow P(c, h_{c \to v}) = e.$$

This implies that

$$n_{v,e} = 1 \Leftrightarrow P'(\varphi(c), h'_{\varphi(c) \to \varphi(v)}) = \psi(e).$$

Thus

$$n_{v,e} = 1 \Leftrightarrow n'_{\varphi(v), \psi(e)} = 1.$$

Hence $N$ and $N'$ are equivalent under row and column permutation (the ones generated by $\varphi$ and $\psi$ on $D$ and $Z$ respectively). On the other hand, we know that $M \neq M'$ (they are two different canonical representatives of two different equivalence classes of $\mathcal{M}_{p,q}$). Therefore, applying Lemma 2, $[M\overline{M}] \not\sim [M'\overline{M'}]$, where $\sim$ is the equivalence relation "row and column permutation". This is a contradiction with $N \sim N'$. Hence $R$ and $R'$ can not be isomorphic.

Therefore $|\mathcal{M}_{p,q}| = |RS(\mathcal{G}_n)|$. On the other hand, $|RS(\mathcal{G}_n)| \leq |RS(\mathcal{G})| = 2^{O(n^2)}$. Since, from Lemma 1, $|\mathcal{M}_{p,q}| = 2^{\Omega(pq - p\log p - q\log q)}$, choosing $p = \Theta(n)$ and $q = \Theta(n)$ implies $|RS(\mathcal{G})| = 2^{\Theta(n^2)}$. □

### 3.4 Distributed *versus* Centralized Routing

**Proof of Theorem 1.**
Let $n$ be any integer. Let $s$ be any real, $1 \leq s < 2$, and let $RS^*$ be a "best" universal routing scheme on routing functions of stretch at most $s$, i.e., a routing scheme such that for every graph $G$ of order $n$, $\mathrm{MEM}_{global}(G,s) = \mathrm{MEM}_{global}(G, RS^*(G))$. Applying Theorem 3, we get $|RS^*(\mathcal{G})| = 2^{\Theta(n^2)}$ where $\mathcal{G}$ is the set of all the graphs of order $n$. Therefore, for every $n$, there exists a graph $G$ of order $n$ such that $\mathrm{MEM}_{global}(G,s) = \Omega(n^2)$ because $RS^*(G)$ is the routing function that requires the minimum memory for the graph $G$. □

Note that, in this paper, we are interested in the problem of *distributed* routing. From a centralized point of view, Theorem 3 give a tight bound on the worst-case memory requirement for universal *centralized* routing scheme.

**Corollary 1** *The maximum, taken on all graphs $G$ of order $n$, of the minimum number of bits necessary to store any routing function $R$ on $G$ of stretch factor $s < 2$ is $\Theta(n^2)$ bits.*

**Proof.** From Theorem 3, $\Omega(n^2)$ bits are necessary. On the other hand, for any graph $G = (V, E)$ of $n$ vertices, we can construct $R = (I, H, P)$ as follows: let $M$ be the adjacency matrix of $G$. The initialization function is $I = \{I_x = Identity \mid x \in V\}$, the header function is $H = \{H_x = Identity \mid x \in V\}$, and the header set $\mathcal{H} = V$. For any pair of vertices $(x, y)$ of $G$, the port function $P$ returns the arc $e = (x, z) \in E$ such that $e$ belongs to a shortest path from $x$ to $y$. The function $P$ is just an algorithm for finding a shortest path in $G$ that uses the matrix $M$ [7, 9]. Moreover $R$ is of stretch factor 1. Clearly, the adjacency matrix and the program for computing shortest paths can be coded with $O(n^2)$ bits. □

Unfortunately, Theorem 3 does not indicate how to evenly distribute the $\Theta(n^2)$ bits of centralized information among the vertices. We will see later that an even distribution of $O(n)$ bits on each router can not be achieved in general: there is a cost to pay in order to distribute the routing function. However, before studying the local memory requirement, we will first refine Theorem 3 in the case of bounded degree graphs.

# 4 Number of Routing Functions on Bounded Degree Graphs

In this section, we will consider the class of graphs of maximum degree $d$. We will compute, like in Section 3.3, the minimum number of distinct outputs of any routing scheme from the set of bounded degree graphs to the class of shortest path routing functions.

## 4.1 Cardinality of the Range of Bounded Degree Routing Schemes

**Theorem 4** *Let $\mathcal{G}$ be the class of graphs of order $n$ and of maximum degree at most 3. Let $\mathcal{R}$ be a class of shortest path routing functions. For any routing scheme $RS : \mathcal{G} \to \mathcal{R}$,*

$$|RS(\mathcal{G})| = 2^{\Omega(n)}$$

The proof is based on the construction of a class of graphs of constraints of degree at most $d$. We call *$d$-graphs of constraints* this class of graphs. We will show later that any two graphs belonging to this class do not support isomorphic shortest path routing functions.

**Lemma 4** *For every pair of integers $(p, q)$, for every $p \times q$ boolean matrix $M$, and for every $d \geq 3$, there exists a graph $G = (V, E)$ such that:*

(1) $|V| \leq 24pq + 10p - 5q$;
(2) $[M\overline{M}]$ *is a matrix of constraints of $G$;*

(3) $V = L \cup H \cup W$, *with* $|L| = p$ *and* $|H| = q$, *and satisfying that, for any* $(u, v, w) \in L \times H \times W$, $\deg(u) = 1$, $\deg(v) = 2$, *and* $3 \leq \deg(w) \leq d$.

Note that (3) implies that $G$ has a maximum degree $d$.

**Proof.** Let $(p, q)$ be any pair of integers, let $d$ be any integer $\geq 3$, and let $M$ be any $p \times q$ boolean matrix. Let us describe the construction of a graph $G = (V, E)$ that satisfies properties (1), (2), and (3) of Lemma 4. This graph $G$ is composed of three levels of vertices, respectively called "High", "Medium", and "Low". We denote by $H = \{c_1, \ldots, c_q\}$ the set of vertices of the High level, by $L = \{v_1, \ldots, v_p\}$ the set of vertices of the Low level, and by $W$ the set of vertices of the Medium level. $G$ can also be seen as composed of $p$ graphs denoted $G_{v_i}$, $i \in \{1, \ldots, p\}$, and of $q$ graphs denoted $G_{c_j}$, $j \in \{1, \ldots, q\}$. $G$ has the general form shown on the left hand side of Figure 2.

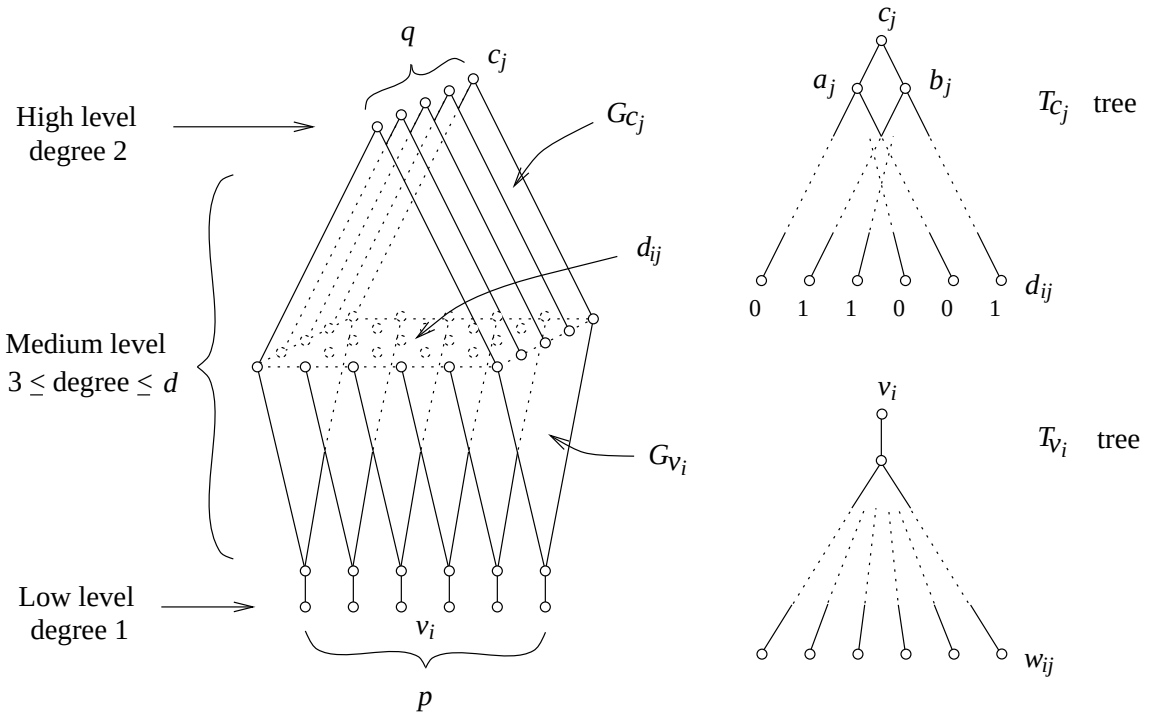



Figure 2: A $d$-graph of constraints.

The structure of each $G_{c_j}$ is as follows. Let $c_j$ be any vertex of the High level $H$, $j \in \{1, \ldots, q\}$. This vertex is the root of a (non necessarily complete) $(d-1)$-ary tree of $p$ leaves, denoted by $T_{c_j}$. The $p$ leaves of $T_{c_j}$ are vertices of $W$. They are denoted by $d_{i,j}$, $i \in \{1, \ldots, p\}$. Vertex $c_j$ is of degree 2. The two children of $c_j$ are vertices of $W$. They are denoted by $a_j$ and $b_j$. These two vertices are the roots of subtrees of $T_{c_j}$, denoted by $T_{a_j}$ and $T_{b_j}$, such that, for every $i \in \{1, \ldots, p\}$, $d_{i,j}$ is a leaf of $T_{a_j}$ if and only if $m_{i,j} = 0$, and $d_{i,j}$ is a leaf of $T_{b_j}$ if and only if $m_{i,j} = 1$ (see the right hand side of Figure 2). We also force all the leaves of $T_{c_j}$ to be at the same distance $\lceil \log_{d-1} p \rceil$ from the root $c_j$ (therefore each internal node may have only one child). The number of vertices of the tree $T_{c_j}$ is $|T_{c_j}| = |T_{a_j}| + |T_{b_j}| + 1$. Let us evaluate this value. Let $p_0$ be the number of 0-entries in the column $j$ of $M$. If $p_0 = 0$ then $|T_{a_j}| = 1$, i.e., $\deg(a_j) = 1$. Otherwise, $|T_{a_j}| \leq 2p_0 + \lceil \log_{d-1} p \rceil - \lceil \log_{d-1} p_0 \rceil$. Hence $|T_{a_j}| \leq 2p$, because $T_{a_j}$ has $p_0$ leaves, $1 \leq p_0 \leq p$,

and a depth of at least $\lceil \log_{d-1} p_0 \rceil$. Similarly $|T_{b_j}| \leq 2p$. Therefore $|T_{c_j}| \leq 4p+1$. Note that some vertices of $T_{c_j}$ can be of degree 1 or 2. To satisfy property (3), we need every vertex of $W$ to be of degree at least 3. Therefore, we transform the tree $T_{c_j}$ in a graph, denoted $G_{c_j}$, such that (a) every vertex $x$ of $T_{c_j}$, but the root $c_j$ and its leaves, has a degree $3 \leq \deg(x) \leq d$, and (b) $c_j$ is at the same distance from all its leaves in $G_{c_j}$. Let $x$ be a vertex of $T_{c_j}$, $x \in W$, $\deg(x) < 3$. Figure 3 shows how to replace some edges of $T_{c_j}$ in order to satisfy properties (a) and (b). The number of vertices of $G_{c_j}$ is $|G_{c_j}| \leq |T_{c_j}| + 5((|T_{c_j}| - 1) - (p+2)) + 4 \leq 19p - 5$, because at most $|T_{c_j}| - p - 3$ edges have to be replaced by adding at most 5 vertices (we do not need to replace the edges of the first and the last level, that is $p+2$), and, in the worst-case, $a_j$ or $b_j$ is of degree 1 (that adds 4 vertices).

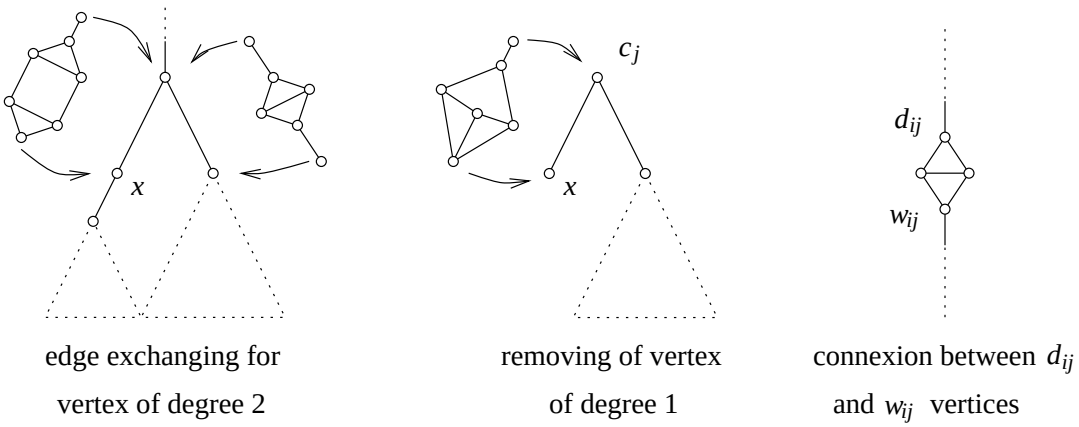



Figure 3: Edge exchanging.

The structure of $G_{v_i}$ is as follows. Let $v_i$ be a vertex of the Low level $L$, $i \in \{1, \ldots, p\}$. The vertex $v_i$ is of degree 1, and it is the root of a $(d-1)$-ary tree, denoted by $T_{v_i}$, with $q$ leaves in $W$ denoted by $w_{i,j}$, $j \in \{1, \ldots, q\}$ (see the right hand side of Figure 2). All leaves of $T_{v_i}$ are forced to be at the same distance $\lceil \log_{d-1} q \rceil$ from the root $v_i$. The number of vertices of the tree $T_{v_i}$ is $|T_{v_i}| \leq 2q + \lceil \log_{d-1} q \rceil$. Similarly, we transform each tree $T_{v_i}$, that may contain vertices of degree 2, in a graph $G_{v_i}$ such that every vertex $x$ of $G_{v_i}$, but the root $v_i$ and the leaves, has a degree $3 \leq \deg(x) \leq d$. Moreover we force the root $v_j$ to be at the same distance from the leaves in $G_{v_i}$. To do that, we use the previous transformations described in Figure 3. The number of vertices of $G_{v_i}$ is $|G_{v_i}| \leq |T_{v_i}| + 5(\lceil \log_{d-1} q \rceil - 1) \leq 2q + 6\lceil \log_{d-1} q \rceil - 5 \leq 3q + 10$ because, at each level of $T_{v_i}$, at most one vertex is of degree 2, and $\lceil \log_{d-1} q \rceil \leq (q+15)/6$, for every $q \geq 1$ and for every $d \geq 3$.

Let us now show how are connected the vertices $d_{i,j}$ with the vertices $w_{i,j}$. For every $i \in \{1, \ldots, p\}$ and for every $j \in \{1, \ldots, q\}$, the graph $G_{c_j}$ is connected to the graph $G_{v_i}$ in order to form the graph $G$. This is done by adding two vertices between vertices $d_{i,j}$ and $w_{i,j}$ as it is shown in Figure 3 (that adds 2 vertices by $d_{i,j}$-$w_{i,j}$ connection). Therefore, every vertex, but the one of $H$ and $L$, are of degree at most $d$ and at least 3. Since the vertices of $L$ have degree 1, and since the vertices of $H$ have degree 2, property (3) holds. Clearly, the order of $G$ is $|V| \leq q(19p-5) + p(3q+10) + 2pq = 24pq + 10p - 5q$, which proves property (1).

Finally, for every $j \in \{1, \ldots, q\}$, the arc $(c_j, a_j)$ induces a constraint on all the $p$ vertices $v_i$ of $L$, $i \in \{1, \ldots, p\}$. This constraint corresponds exactly to the $j$-th column of $M$. Indeed, one can check

that the unique shortest path from $a_j$ to $v_i$ has to traverse $d_{i,j}$. Similarly, the arc $(c_j, b_j)$ induces a constraint on all the $p$ vertices $v_i$ of $L$. This constraint corresponds to the bit-complementation of the $j$-th column of $M$. Therefore we have proved property (2). This implies that such a graph $G$ is a $d$-graph of constraints. □

### 4.2 Proof of Theorem 4.

Let $d \geq 3$, and let $n$ be a sufficiently large integer. Let $RS$ be any routing scheme of $\mathcal{G}$ on the set $\mathcal{R}$ of shortest path routing, where $\mathcal{G}$ is the class of graphs of order $n$ and maximum degree $d$. Consider the subset $\mathcal{G}_{n,d} \subset \mathcal{G}$ of the $d$-graphs of constraints of order $n$. Let $p = q = \lfloor \sqrt{n}/5 \rfloor$. From Lemma 4, we can construct, from any matrix of $\mathcal{M}_{p,q}$, a graph $G'$ with degree $\leq d$ and of order $n' \leq 24pq + 10p - 5q \leq (5p)^2 \leq n$. We can transform $G'$ in a graph $G$ of order $n$ by adding a path $A$ of $n - n'$ vertices to the vertex $v_1$ of the construction of Lemma 4. The former vertex $v_1$ in $G'$ is replaced in $G$ by the root of this path. The graph $G$ satisfies conditions (1) and (2) of Lemma 4, but not property (3) because the degree of the new vertices on the path $A$ of length $n - n' - 1$ is 2.

Let $M, M' \in \mathcal{M}_{p,q}$, $M \neq M'$. Let $G = (V, E)$ and $G' = (V', E')$ be the two corresponding $d$-graphs of constraints of $\mathcal{G}_{n,d}$. For every shortest path routing function $R = (I, H, P)$ on $G$ of header set $\mathcal{H}$, and for every shortest path routing function $R' = (I', H', P')$ on $G'$ of header set $\mathcal{H}'$, we will show that $R$ is non isomorphic to $R'$. Referring to the notations of the proof of Lemma 4, we decompose $V = L \cup H \cup W \cup A$ and $V' = L' \cup H' \cup W' \cup A'$, where $A$ and $A'$ are the sets of vertices of the added path to obtained $d$-graphs of constraints of order $n$ exactly. Moreover, we set $Z = \{(c_j, a_j), (c_j, b_j) \mid 1 \leq j \leq q\}$ and $Z' = \{(c'_j, a'_j), (c'_j, b'_j) \mid 1 \leq j \leq q\}$.

Assume that $R$ and $R'$ are isomorphic. Then, there exist tree bijections $\varphi$, $\psi$ and $\gamma$ such that, for every $x, y \in V$, $x \neq y$, and for every $h \in \mathcal{H}$, Equations (1) and (2) of Definition 5 are satisfied. In a same way as we have developed the proof of Theorem 3, we can show that, for every $x \in V$, $\deg(x) = \deg(\varphi(x))$. This implies $\varphi(L) = L'$, $\varphi(W) = W'$. $\varphi(H \cup A) \subset H' \cup A'$, since vertices of the both sets $H$ and $A$ are of degree 2 (*idem* for $H'$ and $A'$). Let us prove that, actually, $\varphi(H) = H'$. Assume that there is a vertex $x \in H$ such that $\varphi(x) \in A'$. Consider then all communications from $\varphi(x)$ to the vertices of $L'$ that use the arc $(\varphi(x), y')$, for any given neighbor $y'$ of $\varphi(x)$. Since $\varphi(L) = L'$, the constraints of the arc $(\varphi(x), y')$ on $L'$ has to be a column of $M'$ or of $\overline{M'}$. However, this column must contain only one 1 or one 0 because the communications from $\varphi(x)$ to $v_1$ use a different edge than the communications from $\varphi(x)$ to all the other $v_i$'s, $i \neq 1$. This is a contradiction with the definition of $\mathcal{M}_{p,q}$ (see page 8). Therefore $\varphi(H) = H'$ and thus $\psi(Z) = Z'$.

We get that $[M\overline{M}]$ is a matrix of constraints of $Z$ on $L$. It is also a matrix of constraints of $Z'$ on $L'$. By applying Lemma 2 we obtain a contradiction with the fact that $M \neq M'$. Hence $R$ and $R'$ can not be isomorphic. Therefore $|\mathcal{M}_{p,q}| \leq |\mathcal{G}_{n,d}| \leq |RS(\mathcal{G})|$ and then, from Lemma 1, $|RS(\mathcal{G})| = 2^{\Omega(pq - p\log p - q\log q)} = 2^{\Omega(n)}$. This completes the proof of Theorem 4. □

### 4.3 Distributed *versus* Centralized Routing

A lower bound of $\Omega(n)$ bits for centralized routing follows from Theorem 4. However, one can do better. In particular, using a more sophisticated construction than the one of the proof of

Theorem 4, Gavoille[2] has recently shown in [18] that the following result holds:

**Theorem 5** *Let $d \geq 3$. The maximum, taken on all graphs $G$ of order $n$ and maximum degree $d$, of the minimum number of bits necessary to store any shortest path routing function $R$ on $G$ is $\Theta(n \log n)$ bits.*

Note that lower bound is tight since a bounded degree graph can always be described using a list of $n$ items of $O(\log n)$ bits. This yields a lower bound of $\Omega(n \log n)$ bits for the (distributed) global memory requirement on the class of bounded degree graphs. However, this bound is not necessarily tight since the best known upper bound on the global memory requirement is $O(n^2)$ bits, using routing tables[3].

# 5 Local Memory Requirement of Universal Routing Scheme

From Theorem 1, we can derive a lower bound on the local memory requirement for universal routing schemes. Indeed, since $\mathrm{MEM}_{global}(n, s) = \Omega(n^2)$, for every stretch factor $s < 2$, at least one router needs $\Omega(n)$ bits of routing information, i.e., for every $s < 2$, $\mathrm{MEM}_{local}(n, s) = \Omega(n)$. Unfortunately, this is not a tight lower bound because the upper bound is $O(n \log n)$. Let us first introduce several tools that will be useful for proving that the tight bound is $\Theta(n \log n)$.

## 5.1 Generalized Matrices of Constraints

In this section, we introduce a more general and powerful definition of *matrix of constraints* to improve the lower bound on the local memory requirement of universal routing schemes. Instead of specifying the routing constraints by a binary matrix (that indicates whether or not a given vertex $x$ has to use a given arc $e$), we will specify them using the label of the arc that $x$ must use to send a message of destination $y$ for any routing function of stretch factor at most $s$.

More formally, we generalize the notion of matrix of constraints as follows:

**Definition 10 (Generalized Matrix of Constraints)** *Let $G = (V, E)$ be a graph, and $s$ be any real $\geq 1$. A* matrix of constraints of $G$ of stretch factor $s$ *is a $p \times q$ integer matrix $M = (m_{i,j})$, $i \in \{1, \ldots, p\}$ and $j \in \{1, \ldots, q\}$, such that there exist two sets of vertices $A = \{a_i \mid 1 \leq i \leq p\} \subset V$ and $B = \{b_j \mid 1 \leq j \leq q\} \subset V$, and there exist $p$ functions $\psi_i : \{1, \ldots, \mid \cup_{1 \leq j \leq q} \{m_{i,j}\}\mid\} \rightarrow E$, $i \in \{1, \ldots, p\}$, satisfying: for every routing function $R = (I, H, P)$ on $G$ of stretch factor at most $s$, and for every $(i, j)$, $1 \leq i \leq p$, $1 \leq j \leq q$,*

$$\psi_i(m_{i,j}) = P(a_i, I(a_i, b_j))$$

Informally if $M = (m_{i,j})$ is a generalized matrix of constraints of a graph $G$, Definition 10 means that there exists two subsets of vertices $A$ and $B$ such that every near-shortest path between $a_i \in A$ to $b_j \in B$ starts with the arc $e_{i,j}$ which is locally labeled by the integer $m_{i,j}$. In the following, we

[2] Roughly speaking, the construction is based on the network whose a description can also be found in [21].
[3] It was recently proved in [21] that the real bound is $\Theta(n^2)$ bits.

call *constrained vertices* the vertices of the set $A$, and *target vertices* the vertices of the set $B$. The matrices of constraints defined in Section 3.1 can be seen as a restricted case of the generalized matrices of constraints whenever every constrained vertex has a degree 2. In order to simplify the notations in what follows, we will use the terminology *matrices of constraints* instead of *generalized matrices of constraints*.

Figure 4 shows a matrix of constraints of shortest path of the Petersen-graph, with the constrained vertices $A = \{a_1, \ldots, a_5\}$ and the target vertices $B = \{b_1, \ldots, b_5\}$. For example, on this graph, the shortest path from vertex $a_3$ to vertex $b_2$ has to start with the arc $(a_3, b_1)$. It means that it is possible to fix the labels of the incident arcs of the vertices of $A$ such that every shortest path routing function on the Petersen-graph uses port 1 to send a message from $a_3$ to $b_2$.

| ↗ | $b_1$ | $b_2$ | $b_3$ | $b_4$ | $b_5$ |
|---|---|---|---|---|---|
| $a_1$ | 1 | 1 | 2 | 2 | 3 |
| $a_2$ | 1 | 1 | 2 | 2 | 3 |
| $a_3$ | 1 | 1 | 2 | 2 | 3 |
| $a_4$ | 1 | 1 | 2 | 2 | 3 |
| $a_5$ | 1 | 2 | 2 | 1 | 3 |

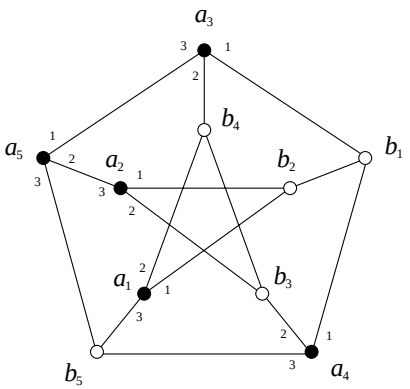



Figure 4: An example of generalized matrix of constraints of shortest path on the Petersen-graph.

If $M$ is a matrix of constraints of a graph $G$, then, from Definition 10, any matrix obtained from $M$ by permutation of rows and columns, and by entry permutation in each row is also a matrix of constraints of $G$. It simply corresponds to another labeling of the vertices and arcs of $G$. Of course, vertex and arc labeling of $G$ have significant implications on the size of the coding of a routing function $R$ on $G$ (see the example of the complete graph in Section 2). By hypothesis, we are looking for the best labeling so that we obtain the most compact coding of $R$.

**Definition 11 ($\sim$ Relation)** *Let $M = (m_{i,j})$, and $M' = (m'_{i,j})$ be two $p \times q$ integer matrices, $i \in I = \{1, \ldots, p\}$ and $j \in J = \{1, \ldots, q\}$. We set $M \sim M'$ if and only if there exist two permutations $r$ of $I$ and $c$ of $J$, and $p$ permutations $\psi_i$ of $\{1, \ldots, |\cup_{j \in J} \{m_{i,j}\}|\}$, $i \in I$, such that for every $i \in I$ and for every $j \in J$:*

$$m'_{i,j} = \psi_i(m_{r(i),c(j)}).$$

The relation $\sim$ is an equivalence relation because $r$, $c$ and $\psi_i$, $i \in \{1, \ldots, p\}$, are bijective functions. $M \sim M'$ means that $M'$ can be obtained from $M$ by rows and columns permutations, and by permutation of the entries of each rows. Permutations $\psi_i$ are related with arc labeling. Permutations $c$ and $r$ are related with vertex labeling (of constrained and target vertices).

As in Section 3.1, we can define a canonical representative of each equivalent class of integer matrices. In the following, we choose as canonical representative of a set of $p \times q$ integer matrices

$M = (m_{i,j})$ the matrix $M_0 \sim M$ that minimizes the integer $\sum_{i,j} m_{i,j} q^{(p-i)q+(q-j)}$. This integer is called the *index* of the matrix.

**Notation.** For every triple of integers $(p, q, d)$, we denote by $d\text{-}\mathcal{M}_{p,q}$ the set of canonical representatives of the different equivalence classes of $\sim$ for the $p \times q$ integer matrices with entries in $\{1, \ldots, d\}$. For example $3\text{-}\mathcal{M}_{2,3}$ is the set:

$$3\text{-}\mathcal{M}_{2,3} = \left\{ \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 2 \end{bmatrix}, \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 3 \end{bmatrix}, \begin{bmatrix} 1 & 1 & 2 \\ 1 & 1 & 2 \end{bmatrix}, \begin{bmatrix} 1 & 1 & 2 \\ 1 & 2 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 1 & 2 \\ 1 & 2 & 3 \end{bmatrix}, \begin{bmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \end{bmatrix} \right\} \quad (4)$$

This example shows that every $2 \times 3$ integer matrices with entries from $\{1, 2, 3\}$ is equivalent to one of the matrices of $3\text{-}\mathcal{M}_{2,3}$. For example matrix $\left[\begin{smallmatrix} 1 & 2 & 2 \\ 1 & 3 & 2 \end{smallmatrix}\right]$ is of index 479, and is equivalent (under $\sim$) to the matrix $\left[\begin{smallmatrix} 1 & 1 & 2 \\ 1 & 2 & 3 \end{smallmatrix}\right] \in 3\text{-}\mathcal{M}_{2,3}$ which is of minimum index 396.

**Lemma 5** *For every triple of integers $(p, q, d)$,*

$$\frac{d^{pq}}{p!\, q!\, (d!)^p} \leq |d\text{-}\mathcal{M}_{p,q}| = 2^{\Omega((pq-pd)\log d - q \log q - p \log p)}$$

**Proof.** There are $d^{pq}$ $p \times q$ integer matrices with entries in $\{1, \ldots, d\}$. At most $p!q!$ matrices are equivalent under row and column permutation, and, for each row, there are at most $d!$ distinct results obtained by permutation of entries inside the row (therefore at most $(d!)^p$ permutations for all the rows). Hence there are at least $d^{pq}/(p!q!(d!)^p)$ distinct classes of integer matrices. □

## 5.2 Generalized Graphs of Constraints

As in Section 3.2, we will see that for every integer matrix there exists a graph, called *graph of constraints*, that has $M$ as matrix of constraints.

**Lemma 6** *For every matrix $M \in d\text{-}\mathcal{M}_{p,q}$, there exists a graph $G$ of order at most $p(d+1)+q$ such that $M$ is a generalized matrix of constraints of $G$ of stretch factor $< 2$.*

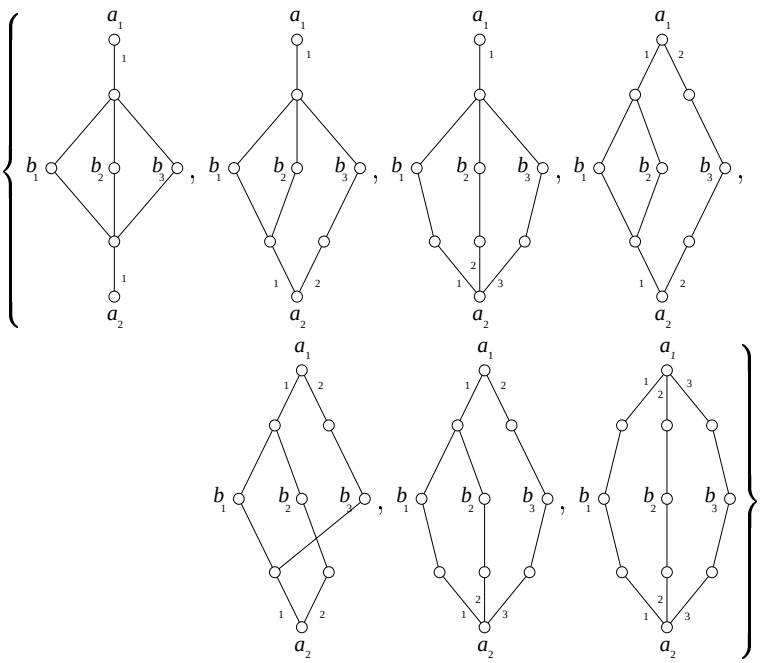
**Proof.** Let $(p, q, d)$ be a triple of integers, and $M = (m_{i,j})$ be any matrix in $d\text{-}\mathcal{M}_{p,q}$. We construct a graph $G = (V, E)$ composed of three distinct levels as follows: $V = A \cup B \cup C$, with $A = \{a_1, \ldots, a_p\}$, $B = \{b_1, \ldots, b_q\}$, and $C \subset \{c_{i,k} \mid 1 \leq i \leq p \text{ and } 1 \leq k \leq d\}$. Vertices are connected as follows: for every $a_i \in A$, $\{a_i, c_{i,k}\} \in E$ if and only if there exists $j$ such that $m_{i,j} = k$, and $\{b_j, c_{i,k}\} \in E$ if and only if $m_{i,j} = k$. No other arc belongs to $E$. Isolated vertices of $C$ are removed.

The local port labeling of vertex $a_i$ is: $(a_i, c_{i,k})$ is labeled $k$. One can check that $M$ is a matrix of constraints of $G$ where $A$ is the set of constrained vertices, and $B$ is the set of target vertices. Moreover, if $m_{i,j} = k$, then there exits a unique path of length 2 from $a_i$ to $b_j$: the path $a_i, c_{i,k}, b_j$. Moreover, the other paths from $a_i$ to $b_j$ that do not start with arc $(a_i, c_{i,k})$ are of length at least 4. Therefore, $M$ is a matrix of constraints of stretch factor $< 2$ in $G$. The order of $G$ is at most

$p(d+1)+q$, because $|A|=p$, $|B|=q$ and $|C|\leq pd$. □

The graphs constructed following the rules given in the proof of Lemma 6 are called *generalized graphs of constraints*, or *graphs of constraints* for short. For every triple of integers $(p,q,d)$, we denote by $d$-$\mathcal{G}_{p,q}$ the set of graphs of constraints of all the matrices in $d$-$\mathcal{M}_{p,q}$. Equation 5 shows the set 3-$\mathcal{G}_{2,3}$.

$$3\text{-}\mathcal{G}_{2,3} = \left\{ \right. \quad , \quad , \quad , \quad , \quad , \quad , \quad \left. \right\} \tag{5}$$




### 5.3 Proof of Theorem 2

Like in Section 3, it is not clear that all the graphs of constraints of the class $d$-$\mathcal{G}_{p,q}$ are pairwise non isomorphic, and support non isomorphic routing functions. However, this is not here the key point. Indeed, to prove Theorem 2, we will used a "local" argument that is, for any matrix $M \in d$-$\mathcal{M}_{p,q}$, the routers of the constrained vertices of a graph of constraints of the matrix $M$ are able to rebuild the matrix.

**Proof of Theorem 2.**
Let $(p,q,d)$ be a triple of integers. For every matrix $M \in d$-$\mathcal{M}_{p,q}$, let $G$ be its graphs of constraints, and let $n$ denote its order. Let $A=\{a_1,\ldots,a_p\}$ be the constrained vertices, and $B=\{b_1,\ldots,b_q\}$ be the target vertices. Let $R$ be a routing function on $G$ such that $s(R,G)<2$. $R$ uses particular vertex and arc labeling. Without loss of generality, one can assume that the output ports of vertices $a_i$ are labeled by the $m_{i,j}$ (otherwise consider another matrix $M'$ equivalent to $M$). By definition, every communication between $a_i \in A$, and $b_j \in B$ is sent out from $a_i$ by the output port labeled $m_{i,j}$. Let $M_B$ be the number of bits necessary to code the list of the labels of the vertices of $B$. Let $M_C$ be the number of bits necessary to describe a function that compute the canonical representative of any $p \times q$ integer matrix with entries from $\{1,\ldots,d\}$. Clearly, since we can do this reasoning for every matrix $M \in d$-$\mathcal{M}_{p,q}$, and since there exists at least one matrix $M \in d$-$\mathcal{M}_{p,q}$

that requires at least $\log_2 |d\text{-}\mathcal{M}_{p,q}|$ bits to be coded, we get that, for this matrix:

$$\sum_{a \in A} \mathrm{MEM}(G, R, a) + M_B + M_C + O(\log n) \geq \log_2 |d\text{-}\mathcal{M}_{p,q}|.$$

Indeed, to rebuild $M$, it is sufficient to test every routers of the vertices in $A$ on all the labels of the target vertices, and to store the results in a matrix $M'$. To do that, it is enough to know the routing functions $R_a$, $a \in A$, the labels of the vertices in $B$, and a way to find the canonical representative of the equivalence class of the matrix $M'$ obtained. To simplify the process, one can even assume that the integers $p$, $q$ and $d$ are given on $O(\log n)$ bits.

Since the vertex-labels are taken from $\{1, \ldots, n\}$, the list of labels of $B$ can be described with $M_B = \log_2 \binom{n}{q} + O(\log n)$ bits. Indeed, there are $\binom{n}{q}$ ways of choosing $q$ labels among the $n$ possible choices. Moreover $M_C = O(\log n)$ because a simple algorithm depending of $p$, $q$, and $d$ can compute all the permutations of rows, columns, and entries of each rows of $M$ in order to minimize the index. Since, for every $q \in \{1, \ldots, n\}$, $\log_2 \binom{n}{q} \leq n = O(pd + q)$, it follows that

$$\sum_{a \in A} \mathrm{MEM}(G, R, a) = \Omega((pq - pd) \log d - q \log q - p \log p) \tag{6}$$

by applying Lemma 5.

For every $n \geq 8$, let us choose $p = \lfloor \sqrt{n/2} \rfloor$, $q = \lfloor n/2 \rfloor$, and $d = p - 1$. For every matrix $M \in d\text{-}\mathcal{M}_{p,q}$, applying Lemma 6, we obtain a graph of constraints $G'$ of $M$ of order $n' \leq p(d+1)+q \leq n$. We can transform $G'$ in a graph $G$ by adding a path of $n - n'$ vertices to a vertex that is not a constrained vertex nor a target vertex. Clearly, $M$ is still a matrix of constraints with stretch factor $< 2$ of $G$, and $G$ is of order $n$. Therefore, by Equation 6,

$$\sum_{a \in A} \mathrm{MEM}(G, R, a) = \Omega(n^{3/2} \log n) \tag{7}$$

Therefore, since $|A| = p = \Theta(\sqrt{n})$, there exists at least one router $a_0 \in A$ that requires $\Omega(n \log n)$ bits to code any routing function $R$ on $G$ of stretch $< 2$. Note that $O(n \log n)$ bits are sufficient using routing tables. Thus $\mathrm{MEM}_{local}(n, s) = \Theta(n \log n)$, for every $s < 2$. □

Theorem 2 implies that routing tables can not be compressed (asymptotically) in the worst-case and in some routers.

**Remark.** Theorem 2 states that at least one router of a graph of constraints $G$ requires $\Omega(n \log n)$ bits to code any routing function $R$ on $G$ of stretch less than 2. However, since $O(n \log n)$ bits per router are sufficient, it its easy to see from Equation 7 that, in fact, $\Omega(\sqrt{n})$ routers (the number of constrained vertices) needs $\Theta(n \log n)$ bits. Using Equation 6 and 7, we can prove a stronger result. For a fixed constant $\varepsilon$, $0 < \varepsilon < 1$, and choosing $p = \Theta(n^{\varepsilon})$, $d = \Theta(n^{1-\varepsilon})$ and $q = \Theta(n)$, we get, applying Equation 6:

$$\sum_{a \in A} \mathrm{MEM}(G, R, a) = \Omega((1 - \varepsilon) n^{1+\varepsilon} \log n) = \Omega(n^{1+\varepsilon} \log n).$$

Therefore, for every constant $\varepsilon < 1$, at least $\Omega(n^{\varepsilon})$ routers require $\Theta(n \log n)$ bits each to code any routing function on $G$ of stretch factor $< 2$.

# 6 Conclusion

The following table summarizes the lower bounds on the local and global memory requirement, and the best known complexity of universal routing schemes on networks of order $n$ for a given stretch factor $s$. Note that, for the lower bounds on the memory requirement, we assumed that universal routing schemes work with uniform cost on the arcs of the graphs, that is we consider a particular case of weighted graphs. However, these lower bounds apply in the general case. Our contribution is indicated in bold. For each result, the reference where it can be found is indicated. When no reference is indicated, the result corresponds to the use of routing tables.

| stretch | local memory requirement | global memory requirement |
|---|---|---|
| $1 \leq s < 2$ | $\mathbf{\Theta(n \log n)}$ [Theorem 2] | $\mathbf{\Omega(n^2)}$ [Theorem 1]<br>$O(n^2 \log n)$ |
| $2 \leq s < 3$ | $\Omega(n^{1/(2s+4)})^\dagger$ [31]<br>$O(n \log n)$ | $\Omega(n^{1+1/(2s+4)})$ [31]<br>$O(n^2 \log n)$ |
| $3 \leq s < 16$ | $\Omega(n^{1/(2s+4)})^\dagger$ [31]<br>$O(n \log n)$ | $\Omega(n^{1+1/(2s+4)})$ [31]<br>$O(n^{1+1/\lfloor \log_2(s+1) \rfloor} \log^2 n)$ [1] |
| $16 \leq s < 87$ | $\Omega(n^{1/(2s+4)})^\dagger$ [31]<br>$O(n^{1/\lfloor \sqrt{s}/4 \rfloor} \log^2 n)$ [2] | $\Omega(n^{1+1/(2s+4)})$ [31]<br>$O(n^{1+1/\lfloor \log_2(s+1) \rfloor} \log^2 n)$ [1] |
| $s \geq 87$ | $\Omega(n^{1/(2s+4)})^\dagger$ [31]<br>$O(\sqrt{s} n^{1/\lfloor \sqrt{s}/4 \rfloor} \log^2 n)$ [2] | $\Omega(n^{1+1/(2s+4)})$ [31]<br>$O(s^3 n^{1+1/\lfloor (s-3)/12 \rfloor} \log n)$ [31] |
| $s = O(\log n)$ | $\Omega(1)^\dagger$<br>$O(e^{\sqrt{\log n}} \log^{3/2} n)$ [2] | $\Omega(n)$ [31]<br>$O(n \log^4 n)$ [31] |
| $s = O(\sqrt{n})$ | $\Omega(1)^\dagger$<br>$O(e^{\sqrt{\log n}} \log^{3/2} n)$ [2] | $\Omega(n)$ [31]<br>$O(n \log n)$ [30] |

$^\dagger$Bounds derived from the lower bound of the global memory requirement.

Table 1: Best known bounds on the memory requirement.

**Comments.**
In the referenced papers, the bounds on the memory requirement of the routing schemes are always given in term of $n^{1/k}$ where $k \geq 1$ is an integer function raising with the stretch factor. Table 1 presents results in another way so that one can compare them as a function of the stretch factor.

- In [1], the stretch factor is at most $2^k - 1$, and the routing scheme allows non uniform cost on the arcs. Another routing scheme is proposed in [1]. It locally requires at most $O(k(d + n^{1/k}) \log n)$ memory bits on any vertex of degree $d$ for every stretch factor at most $2 \cdot 3^k - 1$. Therefore, this routing scheme is efficient in the case of bounded degree networks.
- In [2], the stretch factor is at most $16k^2$, and the routing scheme allows non uniform cost on the arcs. The bound on the local memory requirement is given as a function of the diameter of the network: $O(kn^{1/k} \log n \log D)$ bits per router. In order to compare all the results, we

consider the worst-case, that is $D = O(n)$. Headers, and vertex labeling are of size $O(\log n)$ bits. Note that $n^{1/\sqrt{\log n}} = e^{\sqrt{\log n}}$.

- In [31], the stretch factor is at most $12k + 3$. However, the routing scheme does not support non uniform cost on the arc. Moreover, the routing scheme proposed in this paper is not evenly balanced on all the routers. This is why we can not use this routing scheme for the upper bound on the local memory requirement. Headers are of size $O(\log n)$ bits, but this routing scheme uses a vertex labeling on $O(\log^2 n)$ bits. This means that the computations of the lower and upper bounds are not performed with the same hypotheses because, for the lower bound, the vertex-labels are in the space $\{1, \ldots, n\}$. However, for large value of the stretch factor, this scheme gives a tight upper bound on the global memory requirement.

One can see that, for stretch factor $s < 16$, and for the local memory requirement, the best known universal routing scheme is the routing table. Universal routing schemes of near-shortest paths are not compact routing schemes. Looking for very compact near-shortest paths routing schemes can just be done in "small" class of networks.

In this paper, we gave bounds on the ROM-memory worst-case complexity of a router. However, the memory requirement is just a lower bound on the *surface* of a router. Indeed, some graphs may need a very small number of bits to store the routing function, but may require a large *space* of memory (RAM) to compute the routing function. In this situation, a compact router should realize a trade-off between the memory requirement to store the routing function, and the memory requirement to compute it. A similar trade-off could be searched between the *space*, and the *time* to compute the routing function.

Recently, Gavoille and Prenns give in [21] a tight lower bound of $\Theta(n^2 \log n)$ bits for the global memory requirement for the particular case of stretch factor 1. An interesting problem would be to close the gap between our lower bound of $\Omega(n^2)$ bits and the upper bound of $O(n^2 \log n)$ bits for the the global memory requirement of a near-shortest path routing function with stretch factor strictly greater than 1.

**Acknowledgments:** both authors are thankful to Stéphane Pérennès who helped them to derive the statement of Theorem 2.

**Note:** All results presented in this paper have been partially published in the proceedings of PODC'95 [12].

## References

[1] B. Awerbuch, A. Bar-Noy, N. Linial, and D. Peleg, *Improved routing strategies with succint tables*, Journal of Algorithms, 11 (1990), pp. 307–341.

[2] B. Awerbuch and D. Peleg, *Routing with polynomial communication-space trade-off*, SIAM Journal on Discrete Mathematics, 5 (1992), pp. 151–162.

[3] E. M. Bakker, J. van Leeuwen, and R. B. Tan, *Linear interval routing*, Algorithms Review, 2 (1991), pp. 45–61.

[4] W. J. Dally and C. L. Seitz, *The torus routing chip*, Distributed Computing, 1 (1986), pp. 187–196.

[5] ——, *Deadlock-free message routing in multiprocessor interconnection networks*, IEEE Transaction on Computers, C-36 (1987), pp. 547–553.

[6] F. Desprez, E. Fleury, and M. Loi, *T9000 et C104 : La nouvelle génération de transputers*, Research Report 93-01, LIP, École Normale Supérieure de Lyon, 69364 Lyon Cedex 07, France, Feb. 1993.

[7] E. W. Dijkstra, *A note on two problems in connexion with graphs*, Numerische Mathematik, 1 (1959), pp. 269–271.

[8] M. Flammini, J. van Leeuwen, and A. Marchetti-Spaccamela, *Lower bounds on interval routing*, Tech. Rep. 69, Università di L'Aquila, Dipartimento di matematica Pura ed Applicata, Oct. 1994.

[9] R. W. Floyd, *Shortest path*, Comm. ACM, 5 (1962), p. 345.

[10] P. Fraigniaud and C. Gavoille, *A characterization of networks supporting linear interval routing*, in $13^{th}$ Annual ACM Symposium on Principles of Distributed Computing (PODC), ACM PRESS, ed., Aug. 1994, pp. 216–224.

[11] ——, *Optimal interval routing*, in Parallel Processing: CONPAR '94 - VAPP VI, B. Buchberger and J. Volkert, eds., vol. 854 of Lecture Notes in Computer Science, Springer-Verlag, Sept. 1994, pp. 785–796.

[12] ——, *Memory requirement for universal routing schemes*, in $14^{th}$ Annual ACM Symposium on Principles of Distributed Computing (PODC), ACM PRESS, ed., Aug. 1995, pp. 223–230.

[13] P. Fraigniaud and E. Lazard, *Methods and problems of communication in usual networks*, Discrete Applied Maths, 53 (1994), pp. 79–133.

[14] P. Fraigniaud, A. Liestman, and D. Sotteau, *Open problems*, Parallel Processing Letters, 3 (1993), pp. 507–524.

[15] G. N. Frederickson and R. Janardan, *Designing networks with compact routing tables*, Algorithmica, 3 (1988), pp. 171–190.

[16] ——, *Efficient message routing in planar networks*, SIAM Journal on Computing, 18 (1989), pp. 843–857.

[17] ——, *Space-efficient message routing in c-decomposable networks*, SIAM Journal on Computing, 19 (1990), pp. 164–181.

[18] C. Gavoille, *Memory requirement of centralized routing scheme for bounded degree graphs.* Manuscript, Dec. 1995.

[19] C. Gavoille and E. Guévremont, *Worst case bounds for shortest path interval routing*, Research Report 95-02, LIP, École Normale Supérieure de Lyon, 69364 Lyon Cedex 07, France, Jan. 1995.

[20] C. Gavoille and S. Pérennès, *Lower bounds for interval routing on 3-regular networks*, in $3^{rd}$ International Colloquium on Structural Information & Communication Complexity (SIROCCO), N. Santoro and P. Spirakis, eds., Carleton University Press, June 1996, pp. 88–103.

[21] ——, *Memory requirement for routing in distributed networks*, in $15^{th}$ Annual ACM Symposium on Principles of Distributed Computing (PODC), ACM PRESS, ed., May 1996, pp. 125–133.

[22] M. Gondran and M. Minoux, *Graphs and Algorithms*, John Wiley, 1984.

[23] M. A. Harrison, *On the number of classes of binary matrices*, IEEE Transactions on Computers, C-22 (1973), pp. 1048–1051.

[24] M.-C. Heydemann, J. Meyer, and D. Sotteau, *On forwarding indices of networks*, Discrete Applied Mathematics, 23 (1989), pp. 103–123.

[25] A. N. Kolmogorov, *Three approaches to the quantitative definition of information*, Problems Inform. Transmission, 1 (1965), pp. 1–7.

[26] E. Kranakis and D. Krizanc, *Lower bounds for compact routing*, in $13^{th}$ Annual Symposium on Theoretical Aspects of Computer Science (STACS), C. Puech and R. Reischuk, eds., vol. 1046 of Lecture Notes in Computer Science, Springer-Verlag, Feb. 1996, pp. 529–540.

[27] E. Kranakis, D. Krizanc, and S. S. Ravi, *On multi-label linear interval routing schemes*, in $19^{th}$ International Workshop on Graph - Theoretic Concepts in Computer Science - Distributed Algorithms (WG), vol. 790 of Lecture Notes in Computer Science, Springer-Verlag, June 1993, pp. 338–349.

[28] F. T. Leighton, *Introduction to Parallel Algorithms and Architectures: Arrays - Trees - Hypercubes*, Morgan Kaufmann, 1992.

[29] D. Peleg and A. A. Schäffer, *Graph spanners*, Journal of Graph Theory, 13 (1989), pp. 99–116.

[30] D. Peleg and E. Upfal, *Efficient message passing using succint routing tables*, Research Report RJ 5768, IBM, 1987.

[31] ——, *A trade-off between space and efficiency for routing tables*, in $20^{th}$ Annual ACM Symposium on Theory of Computing (STOC), Chicago, IL, May 1988, pp. 43–52.

[32] N. Santoro and R. Khatib, *Labelling and implicit routing in networks*, The Computer Journal, 28 (1985), pp. 5–8.

[33] J. van Leeuwen and R. B. Tan, *Interval routing*, The Computer Journal, 30 (1987), pp. 298–307.

[34] E. M. Wright, *Graphs on unlabelled nodes with a given number of edges*, Acta Math., 126 (1971), pp. 1–9.