Compressing Two-dimensional Routing Tables with Order

Frédéric Giroire, Frédéric Havet, Joanna Moulierac

Compressing Two-dimensional Routing Tables with Order

Frédéric Giroire, Frédéric Havet , Joanna Moulierac

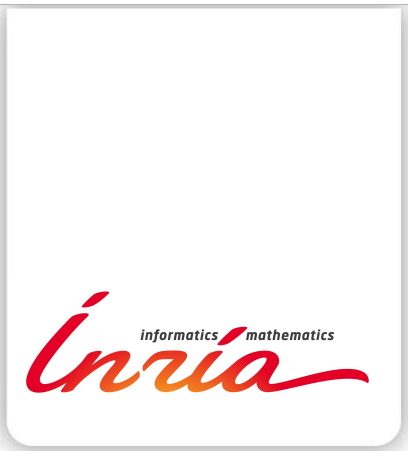
RESEARCH REPORT

N° 8658

December 2014

Project-Team COATI

ISSN 0249-6399 ISRN INRIA/RR--8658--FR+ENG



Compressing Two-dimensional Routing Tables with Order*

Frédéric Giroire†, Frédéric Havet †, Joanna Moulierac‡

Project-Team COATI

Research Report n° 8658 — December 2014 — 30 pages

Abstract: A *communication* in a network is a pair of nodes (s,t). The node s is called the source *source* and t the *destination*. A *communication set* is a set of distinct communications, i.e. two communications might have the same source or the same destination, but they cannot have both same source and same destination. A *routing* of a communication (s,t) is a path in the network from s to t. A *routing* of a communication set is a union of routings of its communications.
At each node, there is a set $\mathcal{X}$ of communications whose routing path goes through this node. The node needs to be able to find for each communication (s,t) in $\mathcal{X}$, the port that the routing path of (s,t) uses to leave it. An easy way of doing it is to store the list of all triples (s,t,k), where $(s,t) \in \mathcal{X}$ and k is the port used by the (s,t)-path to leave the node. Such triples are called *communication triples*.
However, such a list might be very large. Motivated by routing in telecommunication network using Software Defined Network Technologies, we consider the problem of compacting this list using aggregation rules. Indeed, SDN routers use specific memory which is expensive and of small capacity. Hence, in addition, we can use some additional triples, called $*$*-triples*. As an example, a t*-destination triple* $(*,t,p)$, means that every communication with destination t leaves on port p.
We carry out in this work a study of the problem complexity, providing results of NP-completeness, of Fixed-Parameter Tractability and approximation algorithms.

Key-words: routing, routing tables, software defined networks, complexity, FPT, approximation algorithm, compact tables.

* This work has been partially supported by ANR project Stint under reference ANR-13-BS02- 0007, ANR program Investments for the Future under reference ANR-11-LABX-0031-01, ANR VISE, ANR Skyflow, CNRS-FUNCAP project GAIATO, the associated Inria team AlDyNet, the project ECOS-Sud Chile and Paca Region.
† CNRS, Laboratoire I3S, UMR 7172, UNS, CNRS, Inria, COATI, 06900 Sophia Antipolis
‡ Université de Nice Sophia Antipolis, Laboratoire I3S, UMR 7172, UNS, CNRS, Inria, COATI, 06900 Sophia Antipolis

RESEARCH CENTRE
SOPHIA ANTIPOLIS – MÉDITERRANÉE

2004 route des Lucioles - BP 93
06902 Sophia Antipolis Cedex

Compression de tables de routage bidimensionnelles avec ordre

Résumé : Motivés par le routage dans les réseaux de télécommunications utilisant les technologies des réseaux définis en logiciel, de l'anglais Software Defined Networks (SDN), nous considérons le problème de compresser des tables de routage. En effet, les routeurs SDN utilisent une mémoire spécifique, à la fois chère et de petite capacité. Notre problème est de trouver des tables de routage de tailles minimums en transformant des règles (s,t,p), avec s l'adresse source, t l'adresse de destination et p le port de sortie du routeur, en règles agrégées (*,t,p) (s,*,p) et (*,*,p). Par exemple, la règle (*,s,t) signifie que toutes communications avec adresse de destination t utilisent le port p comme port de sortie du routeur.

Nous présentons une étude de la complexité du problème, à savoir des résultats de NP-complétude, de complexité paramétrique (FPT) et des algorithmes d'approximation.

Mots-clés : routage, tables de routage, software defined networking, complexité, FPT, algorithme d'approximation, tables compactes

1 Introduction

Motivation. Software Defined Technology (SDN), e.g. OpenFlow [14] is a new promising approach to operate telecommunication networks. Its promise is to allow to take dynamic routing decisions by decoupling the control plane (the system making decisions) from the data plane (which forwards the packets). This way, a centralized controller receives the data monitored in the system (e.g. load, delay, ...) and then, based on this information, computes appropriate routing decisions. Each time a new flow arrives, the router contacts the controller and waits for the decisions to be pushed into its forwarding table. The routing tables thus are populated with flow-based rules with header informations (source IP, destination IP, ...) $\rightarrow$ exit port.

However, SDN hardware uses specific memory, e.g. TCAM memory [11, 12], which is very expensive and of small size. Thus, the number of entries of the routing tables is limited to only a few thousands [15, 16] and grows linearly with the number of flows passing through a router, causing a problem of scalability. It is thus an important area of research to obtain routing using only a limited number of rules per router. [3] studies the problem of choosing routing with a limited number of entry per router using linear programming. Another way to compact the forwarding tables is to use aggregated rules. With such an aggregation, we can set routing entries such as "(*,destination)$\rightarrow$ port" or "(source,*)$\rightarrow$ port" or also a default entry such as "(*,*)$\rightarrow$ port". For example, [7] studies how to use default ports to reduce the size of routing tables. In this work, we consider the problem of compacting a routing table using aggregated rules.

We consider here *bi-dimensional routing tables* in which the routing decision is not done exclusively on the destination IP addresses, but on the source and destination IP address. Indeed, the commonly implemented destination-based routing has its limitations, especially in delivering quality of service which is a goal of SDN paradigm. One suggested remedy is to base the routing decision on additional fields in the packet header. One of the most important field is the source host. For instance, this would permit selective routing to provide a high bandwidth connection between two different sites of a company. Such refined forwarding is part of the next generation Internet design, and falls within the broader scope of layer four packet classification, where packets are routed using arbitrary fields of the packet header [9], [4], [13], [2]. Routers capable of packet classification can implement many advanced services, such as firewall access control, Virtual Private Networks, and quality of service routing, which are all promises of the SDN paradigm.

Modeling. A *communication* in a network is a pair of nodes (s, t). The node s is called the source *source* and t the *destination*. We use the source and destination fields in our examples, although our ideas apply to any two prefix fields in Internet protocol networks. A *communication set* is a set of distinct communications, i.e. two communications might have the same source or the same destination, but they cannot have both same source and same destination. A *routing* of a communication (s, t) is a path in the network from s to t. A *routing* of a communication set is a union of routings of its communications.

At each node, there is a set $\mathcal{X}$ of communications whose routing path goes through this node. The node needs to be able to find for each communication (s, t) in $\mathcal{X}$, the port that the routing path of (s, t) uses to leave it. An easy way of doing it is to store the list of all triples (s, t, k), where $(s, t) \in \mathcal{X}$ and k is the port used by the (s, t)-path to leave the node. Such triples are called *communication triples*.

However, such a list might be very large. So we want to reduce it as much as possible using the $*$ symbol. Hence, in addition, we can use some additional triples, called $*$*-triples*. There are two kinds of $*$-triples:

- *t-destination triple* $(*, t, p)$, meaning that every communication with destination t leaves

on port p.

- *s-source triple* $(s, *, p)$, meaning that every communication with source t leaves on port p.

A *routing list* is an ordered list $T_1, \ldots, T_r$ of triples (either communication, or source, or destination ones). A communication is then assigned the port of the *first* triple in the list, that applies to it. It is crucial to remark that *using $*$-triples introduces an order of the rules in the routing list.*

Let $\mathcal{C}$ be a set of communication triples. A routing list $\mathcal{R}$ *emulates* $\mathcal{C}$ if each communication of $\mathcal{C}$ is assigned the same port by $\mathcal{C}$ and $\mathcal{R}$. Observe that $\mathcal{R}$ may route more communications than $\mathcal{C}$. For example, if the port of all triples of $\mathcal{C}$ have source s and port p, then the singleton list made of the global triple $(s, *, p)$ emulates $\mathcal{C}$, even if there is not a triple in $\mathcal{C}$ for all communications.

Problems. The problem is then to find the shortest routing list that emulates $\mathcal{C}$. We denote by $\text{rmin}(\mathcal{C})$ the minimum number of triples in a routing list emulating $\mathcal{C}$.

Routing List:
Input: A set $\mathcal{C}$ of communication triples with at most k ports, and an integer r.
Question: $\text{rmin}(\mathcal{C}) \leq r$?

The *number of saved triples* is $\text{sav}(\mathcal{C}) = |\mathcal{C}| - \text{rmin}(\mathcal{C})$. The complementary problem to Routing List is the following.

List Reduction
Input: A set $\mathcal{C}$ of communication triples and an integer z
Question: $\text{sav}(\mathcal{C}) \geq z$?

We shall also consider a variation of the problem in which another kind of triples, called *global triples*, may be used. Such a triple is of the form $(*, *, p)$ and mean that all communications leave on port p. Hence, given a set of communication triples $\mathcal{C}$, the problem is to find the shortest routing list with global triples that emulates $\mathcal{C}$. Let us denote by $\text{rmin}^*(\mathcal{C})$ the minimum number of triples in a routing list with global triples emulating $\mathcal{C}$, and let $\text{sav}^*(\mathcal{C}) = |\mathcal{C}| - \text{rmin}^*(\mathcal{C})$. We study the following variations of Routing List and List Reduction.

With-global Routing List:
Input: A set $\mathcal{C}$ of communication triples with at most k ports, and an integer r.
Question: $\text{rmin}^*(\mathcal{C}) \leq r$?

With-global List Reduction
Input: A set $\mathcal{C}$ of communication triples and an integer z
Question: $\text{sav}^*(\mathcal{C}) \geq z$?

Note that rmin and rmin^* (resp. sav and sav^*) are closely related parameters, in the sense of the two following lemmas.

Lemma 1. *Let $\mathcal{C}$ be a set of communication triples in which n sources appear.*

$$\text{rmin}^*(\mathcal{C}) \leq \text{rmin}(\mathcal{C}) \leq \text{rmin}^*(\mathcal{C}) + n - 1.$$

Proof. Clearly, $\text{rmin}^*(\mathcal{C}) \leq \text{rmin}(\mathcal{C})$.

Let $\mathcal{R}$ be a shortest routing list with global triples. Trivially, $\mathcal{R}$ contains at most one global triple, because all communication triples are routed by a global triple. Let $s_1, \ldots, s_n$ be the sources appearing in $\mathcal{C}$. If $\mathcal{R}$ has a global triple $(*, *, p)$, then it can be replaced by the n source triples $(s_i, *, p)$, $1 \leq i \leq n$, to obtain a routing list with no global triples. Hence, $\text{rmin}(\mathcal{C}) \leq \text{rmin}^*(\mathcal{C}) + n - 1$. □

Let $M(\mathcal{C})$ be the maximum number of triples of $\mathcal{C}$ with the same port.

Lemma 2. *Let $\mathcal{C}$ be a set of communication triples.*

$$\mathrm{sav}(\mathcal{C}) \leq \mathrm{sav}^*(\mathcal{C}) \leq \mathrm{sav}(\mathcal{C}) + M(\mathcal{C}) - 1.$$

Proof. Clearly, $\mathrm{sav}(\mathcal{C}) \leq \mathrm{sav}^*(\mathcal{C})$.

Let us now prove that $\mathrm{sav}^*(\mathcal{C}) \leq \mathrm{sav}(\mathcal{C}) + M(\mathcal{C}) - 1$. Let $\mathcal{R}$ be a shortest routing list with global triples. We have $|\mathcal{R}| = |\mathcal{C}| - \mathrm{sav}^*(\mathcal{C})$. Trivially, $\mathcal{R}$ contains at most one global triple.

If $\mathcal{R}$ contains no global triple then $\mathrm{sav}(\mathcal{C}) \geq |\mathcal{C}| - |\mathcal{R}|$, and so $\mathrm{sav}^*(\mathcal{C}) \leq \mathrm{sav}(\mathcal{C})$.

Assume now that $\mathcal{R}$ contains a global triple, say τ. Let $\mathcal{R}_\tau$ be the set of triples of $\mathcal{C}$ that are routed by τ. Let $\mathcal{R}'$ be the list obtained from $\mathcal{R}$ by replacing τ by $\mathcal{R}_\tau$ (in any order). Clearly, $\mathcal{R}'$ emulates $\mathcal{C}$ and has no global triple. Hence

$$\mathrm{sav}(\mathcal{C}) \geq |\mathcal{C}| - |\mathcal{R}'| = |\mathcal{C}| - (|\mathcal{R}| + |\mathcal{R}_\tau| - 1) = \mathrm{sav}^*(\mathcal{C}) - (|\mathcal{R}_\tau| - 1).$$

But, by definition, $|\mathcal{R}_\tau| \leq M(\mathcal{C}) - 1$. Thus $\mathrm{sav}^*(\mathcal{C}) \leq \mathrm{sav}(\mathcal{C}) + M(\mathcal{C}) - 1$. □

Contributions. In this work, we first study the complexity of the above problems. We provide NP-completeness results in Section 3, Fixed Parameter Tractability in Section 4 and, finally approximation algorithms in Section5.

Our work answers an open question of [17]. Similarly to us, the authors consider the problem of determining a compact routing table using aggregation rules that has the same behavior as the original routing table. The difference with our problem is that their goal is to find what they call a *conflict-free* routing table in which the rules can be taken in any order. On the contrary, as noted above, the order is crucial in our problems. We quote: "The filter compression problem with inconsistent filters [rules], where one uses priority to define best matching filter [rule], is open, and we conjecture that it is NP-complete."

2 Preliminaries

2.1 Standard and canonical routing lists

Let $\mathcal{R} = T_1, \ldots, T_r$ be a routing list, possibly with a global triple. It is *standard* if there exists i such that T_j is a $*$-triple if and only if $j > i$. The following lemma is easy and left to the reader.

Lemma 3. *Let $\mathcal{C}$ be a set of communication triples and $\mathcal{R}$ be a routing list emulating $\mathcal{C}$. Then the routing list $\mathcal{R}'$ obtained from $\mathcal{R}$ by*

- *deleting the useless communication triples (the ones that route no triples),*
- *putting all the communication triples of $\mathcal{R}$ at the beginning and all the $*$-triples of $\mathcal{R}$ at the end, keeping the same order as in $\mathcal{R}$ for the $*$-triples,*

also emulates $\mathcal{C}$.

A routing list is *canonical* if it is the concatenation of sublist $\mathcal{B}_1, \ldots, \mathcal{B}_q$, called *blocks* having the following properties for every $1 \leq \ell \leq q$:

(i) in $\mathcal{B}_\ell$, there is a unique $*$-triple and it is the last one;

(ii) if the $*$-triple of $\mathcal{B}_\ell$ is an s-source triple (resp. t-destination triple), then all triples of $\mathcal{B}_\ell$ have source s (resp. destination t);

(iii) if $\ell \neq q$, then $\mathcal{B}_\ell$ has no global triple.

Lemma 4. *Let $\mathcal{C}$ be a set of communication triples and $\mathcal{R}T_1, \ldots, T_r$ be a routing list emulating $\mathcal{C}$. One can obtain from R a canonical routing list $\mathcal{R}'$ emulating $\mathcal{C}$ not longer than $\mathcal{R}$ by successive applications of the following operations:*

- *deleting some triple,*
- *replacing some communication triple (s,t,p) by the triple $(*,t,p)$,*
- *reordering the triples.*

Proof. We shall use three operations described in the statement according to the following rules on a routing list $\mathcal{L} = T_1, \ldots, T_r$.

(R1) If there are triples after a global triple, then they are useless, so we delete them.

(R2) If the last triple is a communication triple (s,t,p), then we replace it by $(*,t,p)$.

(R3) Let $i_1, i_2, \ldots, i_q$ be the indices in increasing order of the $*$-triples of $\mathcal{L}$. If for some ℓ, T_{i_ℓ} is an s-source triple (resp. t-destination triple) and there is a communication triple T_i with $i_\ell - 1 < i < i_\ell$ with source distinct from s (resp. destination distinct from t), then we move T_i after T_{i_ℓ}.

It is simple matter to check that if any of the above rules applies to a routing list emulating $\mathcal{C}$, then we obtain a new routing list emulating $\mathcal{C}$ which is not longer that the original one.

Therefore starting from $\mathcal{R}$, as long as one of the rules (R1), (R2) or (R3) applies, we do it. This process must end, because each rule can be applied only a finite number of times. Indeed (R1) decreases the size of the list, (R2) increases the number of $*$-triples, and (R3) decreases the vector $(i_1, i_2, \ldots, i_q)$ of the indices of $*$-triples in the lexicographic order.

Let $\mathcal{R}' = T'_1, \ldots, T'_r$ be the routing list obtained at the end of the process. Note that at each time of the process, we have a routing list emulating $\mathcal{C}$ not longer than $\mathcal{R}$ by the above remark. In particular $\mathcal{R}'$ emulates $\mathcal{C}$ and $|\mathcal{R}'| \leq |\mathcal{R}|$. Moreover, none of (R1), (R2), and (R3) applies. Let $i_1, i_2, \ldots, i_q$ be the indices in increasing order of the $*$-triples of $\mathcal{R}'$, and for $1 \leq \ell \leq q$, let $\mathcal{B}_\ell$ be the subsequence $T_{i_{\ell-1}+1}, \ldots, T_{i_\ell}$ (with $i_0 = 0$). By definition, the unique $*$-triple in $\mathcal{B}_\ell$ is its last triple. The concatenation $\mathcal{B}_1, \ldots, \mathcal{B}_q$ is $\mathcal{R}'$ for otherwise (R2) would apply. If $\ell \neq q$, then $\mathcal{B}_\ell$ has no global triple, for otherwise (R1) would apply. Finally, if the $*$-triple of $\mathcal{B}_\ell$ is an s-source triple (resp. t-destination triple), then all triples of $\mathcal{B}_\ell$ have source s (resp. destination t), for otherwise (R3) would apply. Hence, $\mathcal{R}'$ is canonical. □

2.2 The Direction-based Heuristic

Let $\mathcal{C}$ be a set of communication triples with destination set S and destination set T. Set $n = |S|$ and $m = |T|$. For any port p, let $\mathcal{C}(p)$ be the set of triples of $\mathcal{C}$ with port p. For a source s (resp. destination t) and a port p, let $\mathcal{C}(s,p)$ (resp. $\mathcal{C}(s,t)$) be the set of triples of $\mathcal{C}$ with source s (resp. destination t) and port p. For every source s, let $M(s) := \max_p |\mathcal{C}(s,p)|$ be the maximum number of triples in $\mathcal{C}$ with source s and same port, and for every destination t, let $M(t) := \max_p |\mathcal{C}(t,p)|$

be the maximum number of triples in $\mathcal{C}$ with destination t and same port. Set

$$Z^-(\mathcal{C}) = \sum_{s\in S}(M(s)-1) = \sum_{s\in S} M(s) - n \qquad \text{and}$$
$$Z^|(\mathcal{C}) = \sum_{t\in T}(M(t)-1) = \sum_{t\in T} M(t) - m.$$

Any routing list emulating $\mathcal{C}$ yields an upper bound on $\mathrm{rmin}(\mathcal{C})$. One such list can be obtained by routing source by source. One source s after another we route all triples of $\mathcal{C}$ with source s. This can be done by using the triple $(s,*,p)$ for p a port such that there are $M(s)$ triples with source s and port p after all triples with source s and port distinct from p. Doing so, we save $M(s)-1$ triples when routing the triples with source s. Hence, we obtain a routing list of size $|\mathcal{C}| - Z^-(\mathcal{C})$. Such a list is called a *source-based routing list.*

Proceeding similarly according to the destinations, we obtain a routing list, called *destination-based* of size $|\mathcal{C}| - Z^|(\mathcal{C})$.

Setting $Z(\mathcal{C}) = \max\{Z^-(\mathcal{C}), Z^|(\mathcal{C})\}$, we have

$$\mathrm{sav}(\mathcal{C}) \geq Z(\mathcal{C}) \text{ and } \mathrm{rmin}(\mathcal{C}) \leq |\mathcal{C}| - Z(\mathcal{C}). \tag{1}$$

The algorithm consisting in computing a source-based routing list and a destination-based routing list and taking the shortest of the two, is called the *Direction-based Heuristic.* It provides a routing list emulating $\mathcal{C}$ of size $Z(\mathcal{C})$. As we shall see in Corollary 25, the Direction-based Heuristic is a 2-approximation for List Reduction.

3 Complexity results

3.1 Solving 1-Port Routing List in polynomial time

Theorem 5. 1-Port Routing List *can be solved in polynomial time.*

Proof. Let $\mathcal{C}$ be a set of communication triples, all having the same port p. Let S and T be the set of sources and destinations, respectively, of $\mathcal{C}$.

Observe that a routing list with no global triples emulating $\mathcal{C}$ can always be transformed into a routing list with no global triples and no communication triples. Indeed each communication triple (s,t,p) can be replaced by the source triple $(s,*,p)$ (or the destination triple $(*,t,p)$). Therefore, we may only search for a shortest routing list with no global triples and no communication triples. Such a list is a set $\mathcal{R}$ of source triples and destination triples such that, for every triple (s,t,p) of $\mathcal{C}$, either $(s,*,p) \in \mathcal{R}$ or $(*,t,p) \in \mathcal{R}$. This corresponds to finding a minimum vertex cover in the bipartite graph G defined as follows:

- $V(G) = S \cup T$;
- $E(G) = \{st \mid (s,t,p) \in \mathcal{C}\}$.

Recall that a *vertex cover* in G is a subset W of $V(G)$ such that each edge $e \in E(G)$ has an endvertex in W. Let $\rho(G)$ be the size of a minimum vertex cover in G. A well-known theorem of Gallai [5] asserts that $|V(G)| = \rho(G) + \mu(G)$, where $\mu(G)$ is the size of a maximum matching in G. Now, finding a maximum matching in a bipartite graph can be done in polynomial time, by the Hungarian Method for example. Hence 1-Port Routing List can be solved in polynomial time. □

3.2 NP-hardness of 2-Port Routing List

Theorem 6. 2-Port Routing List *is NP-complete.*

Proof. We use a reduction from the Feedback Arc Set problem.

Let D be a bipartite graph with bipartition (A, B). Let $A = \{a_1, \ldots, a_n\}$ and $B = \{b_1, \ldots, b_m\}$. Consider the set of communication triples

$$\begin{aligned}\mathcal{C} \quad = \quad & \{(s_i, t_j, 1) \mid a_i b_j \in A(D)\} \cup \{(s_i, t_j, 0) \mid b_j a_i \in A(D)\} \\ & \cup \ \{(s_i, t'_{(m+1)i+r}, 1) \mid 1 \leq i \leq n,\ 1 \leq r \leq m+1\} \\ & \cup \ \{(s'_{(n+1)j+r}, t_j, 0) \mid 1 \leq j \leq m,\ 1 \leq r \leq n+1\}\end{aligned}$$

Set $r = n + m + k$. We claim that $\text{rmin}(\mathcal{C}) \leq r$ if and only if $\text{fas}(D) \leq k$.

Assume first that $\text{fas}(D) \leq k$. Let $\sigma = v_1, \ldots, v_{n+m}$ be an ordering of $V(D)$ with at most k feedback arcs.

Let $\mathcal{R}$ be the routing list that is the succession of the sublists $\mathcal{B}_1, \ldots, \mathcal{B}_{n+m}$ defined as follows. For $\ell = 1$ to $n + m$, if $v_\ell = a_i$ then let $\mathcal{B}_\ell$ be a list consisting of the communication triples in $\{(s_i, t_j, 0) \mid b_j \in N^-(a_i)\} \setminus \bigcup_{\ell'=1}^{\ell-1} \mathcal{B}_{\ell'}$ (in any order), followed by $(s_i, *, 1)$, and if $v_\ell = b_j$ then let $\mathcal{B}_\ell$ be a list consisting of the communication triples of $\{(s_i, t_j, 1) \mid a_i \in N^-(b_j)\} \setminus \bigcup_{\ell'=1}^{\ell-1} \mathcal{B}_{\ell'}$, followed by $(*, t_j, 0)$. Intuitively, when $v_\ell = a_i$, $\mathcal{B}_l$ routes the communications with source s_i that were not previously routed. The one with port 1 are routed by the source triple $(s_i, *, 1)$ while the ones with port 0 are routed by the preceding communication triples. Similarly, when $v_\ell = b_j$, $\mathcal{B}_l$ routes the communications with destination t_j that were not previously routed.

Clearly, $\mathcal{R}$ emulates $\mathcal{C}$. Let us now calculate its size. $\mathcal{R}$ contains n source triples and m destination triples.

Let us now compute the number of communication triples in each $\mathcal{B}_l$. Assume that $v_\ell = a_i$. A triple $(s_i, t_j, 0)$ is in $\{(s_i, t_j, 0) \mid b_j \in N^-(a_i)\} \setminus \bigcup_{\ell'=1}^{\ell-1} \mathcal{B}_{\ell'}$, if and only if, $b_j a_i \in A(D)$ and a_i precedes b_j in σ. Hence $(s_i, t_j, 0)$ is in $\mathcal{B}_\ell$ if and only if $b_j a_i$ is a σ-feedback arc.

Similarly, if $v_\ell = b_j$, then $(s_i, t_j, 1)$ is in $\mathcal{B}_\ell$ if and only if $a_i b_j$ is a σ-feedback arc. In both cases, the number of communication triples in $\mathcal{B}_l$ is equal to the number of σ-feedback arcs with head v_ℓ. Thus, the number of communication triples in $\mathcal{R}$ is equal to the number of σ-feedback arcs, and so it is at most k. Hence, adding the source triples and destination triples, the set $\mathcal{R}$ has size at most $n + m + k$.

Reciprocally, assume that $\text{rmin}(\mathcal{C}) \leq r$. Let $\mathcal{R}$ be a shortest canonical routing list that emulates $\mathcal{C}$. Then $|\mathcal{R}| \leq r$.

Observe that $\mathcal{R}$ contains exactly n source triples, $\{(s_i, *, 1) \mid 1 \leq i \leq n\}$, and m destination triples $\{(*, t_j, 0) \mid 1 \leq j \leq m\}$. Otherwise, assume that $(s_i, *, 1) \notin \mathcal{R}$ (The proof is similar if $(*, t_j, 0) \notin \mathcal{R}$.). This means that the communication triples $\{(s_i, t'_{(m+1)i+r}, 1) \mid 1 \leq i \leq n,\ 1 \leq r \leq m+1\} \in \mathcal{C}$ have to be routed by $m+1$ rules. If $(s_i, *, 0) \notin \mathcal{R}$, we add the source triple $(s_i, *, 1)$ and remove these $m + 1$ rules, saving m rules without changing the routing. If $(s_i, *, 0) \in \mathcal{R}$, we replace $(s_i, *, 0)$ by $(s_i, *, 1)$ and again remove the $m + 1$ rules. We have to add at most m rules to route all the communication triples with port 0 that were routed by $(s_i, *, 0)$. In both cases, we obtain a routing list emulating $\mathcal{C}$ of smaller size then $\mathcal{R}$, a contradiction.

Now $\mathcal{R}$ is canonical, so every communication triple (s, t, p) is either followed by an s-source triple or a t-destination triple. Therefore $\mathcal{R}$ can be decomposed into the blocks $\mathcal{B}_1, \ldots, \mathcal{B}_{n+m}$, where each $\mathcal{B}_\ell$, $1 \leq \ell \leq n + m$, is either a list of communication triples with source s_i followed by $(s_i, *, 1)$ for some $1 \leq i \leq n$, or a list of destination triples with destination t_j followed by $(*, t_j, 0)$ for some $1 \leq j \leq m$.

Let $\sigma = v_1, \ldots, v_{n+m}$ be the ordering of $V(D)$ where for $1 \leq \ell \leq n+m$, $v_\ell = a_i$ if all triples of $\mathcal{B}_\ell$ have source s_i, and $v_\ell = b_j$ if all triples of $\mathcal{B}_\ell$ have destination t_j. If an arc a_ib_j of D is not σ-feedback, then the block finishing with $(s_i, *, 1)$ precedes the one finishing with $(*, t_j, 0)$. Therefore, the communication triple $(s_i, t_j, 1)$ is routed by $(s_i, *, 1)$ and is not in $\mathcal{R}$. Conversely, if an arc a_ib_j of D is σ-feedback, then the block finishing with $(*, t_j, 0)$ precedes the one finishing with $(s_i, *, 1)$. Therefore, the communication triple $(s_i, t_j, 1)$ must be routed before $(*, t_j, 0)$ and thus is in $\mathcal{R}$. Hence an arc a_ib_j is σ-feedback if and only if $(s_i, t_j, 1)$ is in $\mathcal{R}$. Similarly, an arc b_ja_i is σ-feedback if and only if $(s_i, t_j, 0)$ is in $\mathcal{R}$.

Consequently, the number of σ-feedback arcs is equal the number of communication triples of $\mathcal{R}$, which is at most $r - n - m$, because $\mathcal{R}$ contains n source triples and m destination triples as observed above. Thus $\text{fas}(D) \leq k$. □

3.3 NP-hardness of 2-Port With-global Routing List

Theorem 7. 2-Port With-global Routing List *is NP-complete.*

Proof. The proof is based on the one of Theorem 6.

Let D be a bipartite graph with bipartition (A, B). Let $A = \{a_1, \ldots, a_n\}$ and $B = \{b_1, \ldots, b_m\}$. Consider the set of communication triples

$$\begin{aligned} \mathcal{C}^* &= \mathcal{C} \cup \{(s''_i, t''_i, 1) \mid 1 \leq i \leq 2nm + n + m + 3\} \\ &\quad \cup \{(s_i, t_0, 1) \mid 1 \leq i \leq n\} \\ &\quad \cup \{(s'''_i, t_0, 0) \mid 1 \leq i \leq n + 1\} \end{aligned}$$

where $\mathcal{C}$ is the set of communication triples defined in the proof of Theorem 6.

Set $r = 1 + n + (m + 1) + k$. We claim that $\text{rmin}^*(\mathcal{C}^*) \leq r$ if and only if $\text{fas}(D) \leq k$.

Assume first that $\text{fas}(D) \leq k$. We consider the routing list $\mathcal{R}^* = \mathcal{R} \| (*, t_0, 0) \| (*, *, 1)$, where $\mathcal{R}$ is defined in the proof of Theorem 6. Clearly $\mathcal{R}^*$ emulates $\mathcal{C}^*$ and has a size $1+n+(m+1)+k$.

Reciprocally, assume that $\text{rmin}^*(\mathcal{C}^*) \leq r$. Let $\mathcal{R}^*$ be a canonical shortest routing list that emulates $\mathcal{C}^*$.

Observe first that $(s''_i, t''_i, 1)$, $1 \leq i \leq 2nm + n + m + 3$, is the unique communication triple with source s''_i and the unique one with destination $t"_i$. Therefore unless it is $(*, *, 1)$, the triple routing $(s''_i, t''_i, 1)$ routes only that triple.

Let us first prove that the global triple $(*, *, 1) \in \mathcal{R}^*$. Suppose not. We have to use $2nm + n + m + 3$ triples to route the communication triples of $\{(s''_i, t''_i, 1) \mid 1 \leq i \leq 2nm + n + m + 3\}$. Two cases may happen depending on whether the global triple $(*, *, 0)$ is in $\mathcal{R}^*$ or not. In the second case, we add the global triple $(*, *, 1)$ and remove all triples routing $\{(s''_i, t''_i, 1) \mid 1 \leq i \leq 2nm + n + m + 3\}$ to $\mathcal{R}^*$, we save at least $2nm + n + m + 2$ triples. In the first case, we replace the global triple $(*, *, 0)$ by $(*, *, 1)$, and remove the $2nm + n + m + 3$ triples routing $\{(s''_i, t''_i, 1) \mid 1 \leq i \leq 2nm+n+m+3\}$, and we add the communication triples that were routed by $(*, *, 0)$. There are at most $nm+m(n+1)+(n+1) = 2nm+m+n+1$ such triples, because this is the total number of communication triples with port 0. Therefore, in both cases, we obtain a routing list emulating $\mathcal{C}^*$ shorter than $\mathcal{R}^*$, a contradiction.

Let us now prove that $\mathcal{R}^*$ contains the $m + 1$ destination triples $\{(*, t_j, 0) \mid 0 \leq j \leq m\}$. Suppose not. Then there exists j such that $(*, t_j, 0) \notin \mathcal{R}^*$. This means that the communication triples $\{(s'_{(n+1)j+r}, t_j, 0) \mid 1 \leq j \leq m,\ 1 \leq r \leq n+1\} \cup \{(s'''_i, t_0, 0) \mid 1 \leq i \leq n+1\} \in \mathcal{C}^*$ have to be routed by $n + 1$ rules. If $(*, t_j, 1) \notin \mathcal{R}^*$, we add the source triple $(*, t_j, 0)$ and remove these $n + 1$ rules, saving n rules without changing the routing. If $(*, t_j, 1) \in \mathcal{R}^*$, we replace $(*, t_j, 1)$

by $(*, t_j, 0)$ and again remove the $n+1$ rules. We have to add at most n rules to route all the communication triples with port 1 that were routed by $(*, t_j, 1)$. In both cases, we obtain a routing list emulating $\mathcal{C}^*$ shorter than $\mathcal{R}^*$, a contradiction.

We now denote by α the number of source triples $(s_i, *, 1)$ and by β the number of communication triples $(s_i, t_0, 1)$, for $1 \leq i \leq n$. We have $\alpha + \beta = n$. Indeed, as $(*, t_0) \in \mathcal{R}^*$, on a line s_i, the communication triples $(s_i, t_0, 1)$ is necessary when $(s_i, *, 1)$ is not in $\mathcal{R}^*$, and superfluous in the opposite case.

Now $\mathcal{R}^*$ is canonical, so every communication triple (s, t, p) is either followed by an s-source triple or a t-destination triple or the global triple $(*, *, 1)$. Therefore $\mathcal{R}^*$ can be decomposed into the blocks $\mathcal{B}_1, \ldots, \mathcal{B}_{m+\alpha+2}$, where each $\mathcal{B}_\ell$, $1 \leq \ell \leq m+\alpha+1$, is either a list of communication triples with source s_i followed by $(s_i, *, 1)$ for some $1 \leq i \leq n$, or a list of destination triples with destination t_j followed by $(*, t_j, 0)$ for some $1 \leq j \leq m$. The last block $\mathcal{B}_{m+\alpha+2}$ is a list of communication triples followed by the global triple $(*, *, 1)$.

We know show that there exists a canonical routing $\mathcal{R}''$ emulating $\mathcal{C}^*$ in which the rule $(*, t_0, 0)$ is in the penultimate block, $\mathcal{B}_{m+\alpha+1}$. Note first that the rule $(*, t_0, 0)$ appears after any source triple$(s_i, *, 1)$. Otherwise, we may switch the two corresponding blocks and save the communication triple$(s_i, t_0, 1)$. To conclude, we know that two blocks corresponding to two destination triples can be switched without affecting the routing. Henceforth, in the following we assume that $(*, t_0, 0)$ is the last triple of the block $\mathcal{B}_{m+\alpha+1}$.

Let $\sigma = v_1, \ldots, v_{n+m}$ be the ordering of $V(D)$ where for $1 \leq \ell \leq \alpha + m$, $v_\ell = a_i$ if all triples of $\mathcal{B}_\ell$ have source s_i, and $v_\ell = b_j$ if all triples of $\mathcal{B}_\ell$ have destination t_j. We then arbitrary assign the remaining vertices of $\mathcal{A}$ to the v_ℓ, for $\alpha + m + 1 \leq \ell \leq n + m$.

If an arc $a_i b_j$ of D is not σ-feedback, then the block finishing with $(s_i, *, 1)$ precedes the one finishing with $(*, t_j, 0)$. Therefore, the communication triple $(s_i, t_j, 1)$ is routed by $(s_i, *, 1)$ and is not in $\mathcal{R}^*$. Conversely, if an arc $a_i b_j$ of D is σ-feedback, then the block finishing with $(*, t_j, 0)$ precedes the one finishing with $(s_i, *, 1)$ if it exists, or the global triple $(*, *, 1)$. Therefore, the communication triple $(s_i, t_j, 1)$ must be routed before $(*, t_j, 0)$ and thus is in $\mathcal{R}^*$. Hence an arc $a_i b_j$ is σ-feedback if and only if $(s_i, t_j, 1)$ is in $\mathcal{R}^*$. Similarly, an arc $b_j a_i$ is σ-feedback if and only if $(s_i, t_j, 0)$ is in $\mathcal{R}^*$.

To summarize, the number of communication triples of $\mathcal{R}^*$ is equal to the number of σ-feedback arcs plus β, the number of communication triples of the column 0. $\mathcal{R}^*$ contains $m+1$ destination triples and α source triples, with $\alpha+\beta = n$. Thus, $\text{rmin}^*(\mathcal{C}^*) \geq 1+n+(m+1)+\text{fas}(D)$. Since $\text{rmin}^* \leq r$, it follows $\text{fas}(D) \leq k$. □

3.4 Short routing list for full sets of communication triples

Recall that a communication set is *full* if it is of the form $S \times T$, and that a set of communication triples is *full* if it is a set of communication triples on a full set of communications.

Theorem 8. Routing List *is NP-complete even if $\mathcal{C}$ is restricted to be a full set of communication triples.*

Proof. Reduction from Routing List.

Let $\mathcal{C}$ be a set of communication triples. Let S be the set of sources of $\mathcal{C}$ and T be the set of destinations of $\mathcal{C}$. Let Q be the set of communications of $S \times T$ not routed by $\mathcal{C}$. For each communication (s, t) in Q, let $p(s, t)$ be a port dedicated to this communication, and set $\mathcal{C}^* = \mathcal{C} \cup \{p(s, t) \mid (s, t) \in Q\}$. The set $\mathcal{C}^*$ is a full set of communication. Moreover, in any shortest routing list emulating $\mathcal{C}^*$, for each $(s, t) \in Q$, there is a unique triple with port $p(s, t)$, because it is only used for the communication (s, t). Therefore $\text{rmin}(\mathcal{C}^*) = |Q| + \text{rmin}(\mathcal{C})$. □

Similarly to Theorem 8, one can prove the following.

Theorem 9. With-global Routing List *is NP-complete even if $\mathcal{C}$ is restricted to be a full set of communication triples.*

In the proof of Theorem 8, the number of ports is not bounded. However, we believe that the problem remain NP-complete as soon as they are at least two ports.

Conjecture 10. For every $k \geq 2$, k-Port Routing List and k-Port With-global Routing List are NP-complete even if $\mathcal{C}$ is restricted to be a full set of communication triples.

4 Parameterized complexity

4.1 List Reduction and variants

Theorem 11. *For every $k \geq 1$, k-port List Reduction parameterized by z admits a linear kernel, and so is FPT.*

Proof. Let us describe a kernelization algorithm that given an instance $(\mathcal{C}, z)$ of k-port List Reduction, returns either 'Yes' (or a small yes instance, e.g. $(\emptyset, 0)$) only if $\text{sav}(\mathcal{C}) \geq z$, or an instance $(\mathcal{C}', z)$ equivalent to $\mathcal{C}$ (that is such that $\text{sav}(\mathcal{C}) \geq z$ if and only if $\text{sav}(\mathcal{C}') \geq z$) of size at most $(4z-4)k$.

A triple $\tau = (s, t, p)$ of $\mathcal{C}$ is *source-linked* (resp. *destination-linked*) is $\mathcal{C}$ contains another triple with port p and source s (resp. destination t). It is *isolated*, if it is neither source-linked nor destination-linked. Let τ be an isolated triple. Observe that whatever triple is used to route τ, it is the only one routed by this triple. Hence $\text{sav}(\mathcal{C}) = \text{sav}(\mathcal{C} \setminus \{\tau\})$.

Therefore our kernelization algorithm removes all isolated triples of $\mathcal{C}$. Let $\mathcal{C}'$ be the resulting set of communication triples. We then run the Direction-based Heuristic on $\mathcal{C}'$ to compute $Z(\mathcal{C}')$. If $Z(\mathcal{C}') \geq z$, then we return 'Yes', because $\text{sav}(\mathcal{C}) \geq \text{sav}(\mathcal{C}') \geq Z(\mathcal{C}')$. If not, then we return $\mathcal{C}'$.

Clearly, the returned instance is equivalent to $\mathcal{C}$. Let us now prove that it has size at most $(4z-4)k$. We know that $Z^-(\mathcal{C}') \leq z-1$.

Let $\mathcal{C}^-$ (resp. $\mathcal{C}^|$) be the set of source-linked (resp. destination-linked) communication triples in $\mathcal{C}'$. Because $\mathcal{C}'$ has no isolated vertices, we have $\mathcal{C}' = \mathcal{C}^- \cup \mathcal{C}^|$. Let S' be the set of sources s such that $M(s) \geq 2$. Observe that

$$|S'| \leq \sum_{s \in S'} (M(s) - 1) = Z^-(\mathcal{C}') \leq z - 1 \tag{2}$$

and all source-linked communication triples have source in S'. But for a source s there are at most $kM(s)$ triples with source s. Hence $|\mathcal{C}^-| \leq k \sum_{s \in S'} M(s)$. Now $\sum_{s \in S'} M(s) = \sum_{s \in S'} (M(s) - 1) + |S'| \leq 2z - 2$ by Equation (2). Therefore $|\mathcal{C}^-| \leq (2z-2)k$. Similarly, $|\mathcal{C}^|| \leq (2z-2)k$, thus $|\mathcal{C}'| \leq (4z-4)k$. □

Theorem 12. List Reduction *parameterized by z admits a quadratic kernel, and so is FPT.*

Proof. Let us describe a kernelization algorithm that given an instance $(\mathcal{C}, z)$ of List Reduction, returns either 'Yes' only if $\text{sav}(\mathcal{C}) \geq z$, or an instance $(\mathcal{C}', z)$ equivalent to $\mathcal{C}$ (that is such that $\text{sav}(\mathcal{C}) \geq z$ if and only if $\text{sav}(\mathcal{C}') \geq z$) of size at most $z(4z-4)$.

We first remove all isolated triples of $\mathcal{C}$, and denote the resulting set of communication triples $\mathcal{C}'$. Let S' be the set of sources s such that $M(s) \geq 2$, and T' be the set of destinations t such that $M(t) \geq 2$. We have $|S'| \leq z-1$ and $|T'| \leq z-1$. For any source s, let $P(s)$ be the set of ports such that do not appear on $\{s\} \times T'$.

Assume that $P(s) \neq \emptyset$. Let $M_P(s)$ be the maximum number of triples with source s and same port in $P(s)$, and let $p_1(s)$ be a port in $P(s)$ appearing on $M_P(s)$ triples with source s. For any $p \in P(s)$, let $\mathcal{C}'(p,s)$ be the set of triples of $\mathcal{C}'$ with source s and port p. Let $\mathcal{R}$ be a shortest routing list emulating $\mathcal{C}'$. Suppose that $(s,*,p) \in \mathcal{R}$ for some $p \in P(s) \setminus \{p_1(s)\}$. Necessarily, the triples of $\mathcal{C}'(p_1(s),s)$ precede $(s,*,p)$ in $\mathcal{R}$. Hence replacing $(s,*,p)$ by $(s,*,p_1(s))$ and the $M_P(s)$ triples of $\mathcal{C}'(p_1(s),s)$ by the ones of $\mathcal{C}'(p,s)$, we obtain a routing list $\mathcal{R}'$ emulating $\mathcal{C}'$ of length not smaller than $\mathcal{R}$. By minimality of $\mathcal{R}$, $\mathcal{R}'$ is also a shortest routing list emulating $\mathcal{C}'$. Moreover all triples of $\mathcal{C}'(p,s)$ appear in $\mathcal{R}$. Therefore

$$\text{sav}(\mathcal{C}') = \text{sav}(\mathcal{C}' \setminus \mathcal{C}'(p,s)) \text{ for all } p \in P(s) \setminus \{p_1(s)\}. \tag{3}$$

Similarly, for every destination t, we define $P(t)$ as the set of ports such that do not appear on $S' \times \{t\}$. We let $M_P(t)$ be the maximum number of triples with destination t and same port in $P(t)$, and let $p_1(t)$ be a port in $P(t)$ appearing on $M_P(t)$ triples with destination t. Finally, let for any $p \in P(t)$, let $\mathcal{C}'(t,s)$ be the set of triples of $\mathcal{C}'$ with destination t and port p. A symmetrical argument as above shows

$$\text{sav}(\mathcal{C}') = \text{sav}(\mathcal{C}' \setminus \mathcal{C}'(p,t)) \text{ for all } p \in P(t) \setminus \{p_1(t)\}. \tag{4}$$

Let $\mathcal{C}''$ be the the set of communication triples obtained from $\mathcal{C}'$ by removing $\mathcal{C}'(p,s))$ for every source $s \in S'$ and every port $p \in P(s) \setminus \{p_1(s)\}$, and removing $\mathcal{C}'(p,t))$ for all destination $t \in T'$ and all port $p \in P(t) \setminus \{p_1(r)\}$. Applying Equations 3 and 4 many times, we obtain $\text{sav}(\mathcal{C}') = \text{sav}(\mathcal{C}'')$, i.e. $\mathcal{C}''$ is equivalent to $\mathcal{C}'$ and so to $\mathcal{C}$.

By construction of $\mathcal{C}''$, for any $s \in S'$, there are at most z ports appearing on the triples with source s: at most $z-1$ on the triples with destination in T' and $p_1(s)$. Hence, there are at most $z\sum_{s\in S'} M(s)$ triples with source in S'. Since $\sum_{s\in S'} M(s) \leq |S'| + \sum_{s\in S'}(M(s)-1) \leq 2z-2$, there are at most $z(2z-2)$ triples of $\mathcal{C}''$ with source in S'. Similarly, there are at most $z(2z-2)$ triples of $\mathcal{C}''$ with destination in T'. Thus, $|\mathcal{C}''| \leq z(4z-4)$. Hence, returning $\mathcal{C}''$ yields the desired kernelization. □

Problem 13. Does List Reduction parameterized by z admits a linear kernel?

Theorem 14. *For every $k \geq 1$, k-port With-global List Reduction parameterized by z admits a linear kernel, and so is FPT.*

Proof. Let $(\mathcal{C},z)$ be an instance of k-port With-global List Reduction. If a port p appears at least $z+1$ times in $\mathcal{C}$, then $\text{sav}^*(\mathcal{C}) \geq z$ since we can first route each triple of $\mathcal{C}$ with a port distinct from p by itself, and then route all the triples with ports p with $(*,*,p)$. Therefore, we have the following easy kernelization algorithm. If a port appears at least $z+1$ times, then we return 'Yes'. If not, we return $\mathcal{C}$, which has size at most $k \cdot z$. □

Theorem 15. With-global List Reduction *parameterized by z admits a cubic kernel, and so is FPT.*

Proof. Let us describe a kernelization algorithm that given an instance $(\mathcal{C},z)$ of With-global List Reduction, returns either 'Yes' only if $\text{sav}^*(\mathcal{C}) \geq z$, or an instance $(\mathcal{C}',z)$ equivalent to $\mathcal{C}$ (that is such that $\text{sav}^*(\mathcal{C}) \geq z$ if and only if $\text{sav}^*(\mathcal{C}') \geq z$) of size at most $3z^3$.

We first check if a port appears at least $z+1$ times in $\mathcal{C}$. If yes, then we return 'Yes'. Henceforth, we may assume that every port appears at most z times.

A source s (resp. destination t) is *rainbow* if the communication triples of $\mathcal{C}$ with source s (resp. destination t) have pairwise distinct ports, that is if $M(s) = 1$. If $\mathcal{C}$ has more than z non-rainbow sources or more than z non-rainbow destinations, then we return 'Yes'. This is

valid because $\mathrm{sav}^*(\mathcal{C}) \geq \mathrm{sav}(\mathcal{C}) \geq Z(\mathcal{C})$. Henceforth we may assume that the numbers a and b of non-rainbow sources and non-rainbow destinations, respectively, are both less than z. Let $S' = \{s_1, \ldots, s_a\}$ be the non-rainbow sources, and $T' = \{t_1, \ldots, t_b\}$ be the set of non-rainbow destinations.

Let p be a port. It is loose there is no triple of $\mathcal{C}$ with source in S', destination in T' and port p. For every source s (resp. destination t), we denote by $\mathcal{C}(s,p)$ (resp. $\mathcal{C}(t,p)$) the set of triples of $\mathcal{C}$ with source s (resp. destination t) and port p. Let $\mathcal{T}(p)$ the set of triples of $\mathcal{C}$ with port p, source not in S' and destination not in T'. The *trace* of p is the $(a+b+1)$-uple

$$(|\mathcal{C}(s_1,p)|, \ldots, |\mathcal{C}(s_a,p)|, |\mathcal{C}(t_1,p)|, \ldots, |\mathcal{C}(t_b,p)|, |\mathcal{T}(p)|).$$

Note that if p is loose, then $(\mathcal{C}(s_1,p), \ldots, \mathcal{C}(s_a,p), \mathcal{C}(t_1,p), \ldots, \mathcal{C}(t_b,p), \mathcal{T}(p))$ is a partition of $\mathcal{C}(p)$. As long as two loose ports p_1 and p_2 have the same trace, then we remove the triples with ports p_1. This is valid according to the following claim whose proof is postponed to the end of this proof.

Claim. *If two loose ports p_1 and p_2 have the same trace, then* $\mathrm{sav}^*(\mathcal{C}) = \mathrm{sav}^*(\mathcal{C} \setminus \mathcal{C}(p_1))$.

We return the resulting set of communication triples $\mathcal{C}'$. Observe that in $\mathcal{C}'$ loose ports have different traces. Now $a+b+1 \leq 2z$ and every element of the trace is at most z.Therefore there are at most $2z^2$ possible traces, so there are at most $2z^2$ loose ports. Moreover, the number of non-loose ports is at most $S' \times T' \leq z^2$. Thus there are at most $3z^2$ ports appearing in $\mathcal{C}'$. But each port appears at most z times in $\mathcal{C}$ and thus also in $\mathcal{C}'$. Therefore $|\mathcal{C}'| \leq 3z^3$.

It remains to prove the claim.
Clearly, $\mathrm{sav}^*(\mathcal{C}) \geq \mathrm{sav}^*(\mathcal{C} \setminus \mathcal{C}(p_1))$.

Let us now prove the opposite inequality. Let $\mathcal{R}$ be a shortest routing with global triples emulating $\mathcal{C}$ that contains the minimum number of $*$-triples with port p_1.

Observe first that if $\mathcal{R}$ contains a source triple whose source is rainbow, then it routes a unique triple by which it can be replaced. Free to make such replacements, we may assume that the source of every source triple of $\mathcal{R}$ is in S'. Similarly, we may assume that the destination of every destination triple of $\mathcal{R}$ is in T'. Moreover, by Lemma 3, we can assume that in $\mathcal{R}$ all the $*$-triples are at the end.

Assume for a contradiction that $\mathcal{R}$ contains a source triple with port p_1, say $(s, *, p_1)$. Necessarily, $s \in S'$. Because p_1 and p_2 have the same trace, $|\mathcal{C}(s,p_1)| = |\mathcal{C}(s,p_2)|$. The triples of $\mathcal{C}(s,p_2)$ are routed by $\mathcal{R}$. But since p_2 is loose, their destinations are not in T', so they cannot be routed by a destination triple. Therefore $\mathcal{C}(s,p_2) \subseteq \mathcal{R}$. Furthermore, the destination of the triples of $\mathcal{C}(s,p_1)$ are not in T'. Thus there is no destination triple with their destination. Therefore no triple τ of $\mathcal{C}(s,p_1)$ appears in $\mathcal{R}$, for otherwise $\mathcal{R} \setminus \{\tau\}$ would also emulate $\mathcal{C}$, a contradiction to the fact $\mathcal{R}$ is a shortest list emulating $\mathcal{C}$. Consider the list $\mathcal{R}'$ obtained from $\mathcal{R}$ by replacing $\mathcal{C}(s,p_2)$ by $\mathcal{C}(s,p_1)$ and $(s,*,p_1)$ by $(s,*,p_2)$. Clearly, $\mathcal{R}'$ emulates $\mathcal{C}$, $|\mathcal{R}'| = |\mathcal{R}|$ and it has one $*$-triple with port p_1 less than $\mathcal{R}$. This is a contradiction. Therefore $\mathcal{R}$ contains no source triple with port p_1. Similarly, $\mathcal{R}$ contains no destination triple with port p_1.

Assume for a contradiction that $\mathcal{R}$ contains the global triple $(*,*,p_1)$.

We assert that there is no source or destination triple with port p_2. Suppose there were one, say, without loss of generality, a source triple $\tau = (s,*,p_2)$. Necessarily, $s \in S'$. Consider now a triple (s,t,p_1) of $\mathcal{C}(s,p_1)$. Because p_1 is loose, its destination t is rainbow and thus there is no destination triple with destination t in $\mathcal{R}$. Therefore (s,t,p_1) must be in $\mathcal{R}$ to route itself. Thus $\mathcal{C}(s,p_1) \subseteq \mathcal{R}$. Consider the concatenation $\mathcal{R}'$ of $\mathcal{C}(s,p_2)$ and $\mathcal{R} \setminus (\mathcal{C}(s,p_1) \cup \{\tau\})$. Any triple

of $\mathcal{C} \setminus (\mathcal{C}(s,p_1) \cup \mathcal{C}(s,p_2))$ is clearly assigned the same port by $\mathcal{R}'$ and $\mathcal{R}$, and every elements of $\mathcal{C}(s,p_2)$ is assigned p_2 by itself. Let $(s,t,p_1) \in \mathcal{C}(s,p_1)$. There is no destination triple with destination t in $\mathcal{R}'$ because there were no in $\mathcal{R}$, and there is no source triple with source s in $\mathcal{R}'$ because there the unique one of $\mathcal{R}$ was τ. Therefore $\mathcal{R}'$ assigns to (s,t,p_1) the port p_1 by $(*,*,p_1)$. Consequently, the routing list $\mathcal{R}'$ emulates $\mathcal{C}$, and it is shorter than $\mathcal{R}$, a contradiction. This proves our assertion.

This implies that $\mathcal{C}(p_2) \subseteq \mathcal{R}$. Now consider the routing list $\mathcal{R}''$ which is the concatenation of $\mathcal{C}(p_1)$, $\mathcal{R} \setminus (\mathcal{C}(s,p_2) \cup \{(*,*,p_1)\})$, and $(*,*,p_1)$. As above for $\mathcal{R}'$, one checks that $\mathcal{R}''$ emulates $\mathcal{C}$. But $\mathcal{R}''$ has the same size as $\mathcal{R}$ and but one $*$-triple with port p_1 less than $\mathcal{R}$, a contradiction.

Consequently, $\mathcal{R}$ has no $*$-triples with port $\mathcal{R}$, hence $\mathcal{C}(p_1) \subseteq \mathcal{R}$. Now the routing list $\mathcal{R} \setminus \mathcal{C}(p_1)$ emulates $\mathcal{C} \setminus \mathcal{C}(p_1)$, and so

$$\mathrm{sav}^*(\mathcal{C} \setminus \mathcal{C}(p_1)) \geq |\mathcal{C} \setminus \mathcal{C}(p_1)| - \mathcal{R} \setminus \mathcal{C}(p_1) = |\mathcal{C}| - |\mathcal{R}| = \mathrm{sav}(\mathcal{C}).$$

This completes the proof of the claim, and that of the theorem. □

Problem 16. Does With-global List Reduction parameterized by z admits a linear or quadratic kernel?

4.2 Routing List and variants

Theorem 17. Routing List *parameterized by r admits a cubic kernel, and so is FPT.*

Proof. Let us describe a kernelization algorithm that given an instance $(\mathcal{C}, r)$ of Routing List, returns either 'No' only if $\mathrm{rmin}(\mathcal{C}) \geq r$, or an instance $(\mathcal{C}", r)$ equivalent to $\mathcal{C}$ (that is such that $\mathrm{rmin}(\mathcal{C}) \geq r$ if and only if $\mathrm{rmin}(\mathcal{C}") \geq r$) of size at most $2r^3 + r^2 + r$.

We first check whether there are sets $S' \subseteq S$ and $T' \subseteq T$ such that $|S'| + |T'| \leq r$ and for every triple $(s,t,p) \in \mathcal{C}$ either $s \in S'$ or $t \in T'$. This is equivalent to decide whether there is a vertex cover of cardinality at most r in the bipartite graph G defined as follows:

- $V(G) = S \cup T$;
- $E(G) = \{st \mid \text{there exists } p \text{ such that} (s,t,p) \in \mathcal{C}\}$.

If no such sets S', T' exist, then we return 'No'. This is valid because in a canonical list of size at most r, there are at most r blocks and all triples in a block have either same source or same destination.

Therefore, we may assume that we have two such sets S' and T'. (They can also be found in polynomial time.)

A source $s \in S'$ (resp. destination $t \in T'$) is *forced* if $M(s) \geq r+1$ (resp. $M(t) \geq r$) and *loose* otherwise.

Claim 17.1. *Assume that a routing list $\mathcal{R}$ emulating $\mathcal{C}$ has size at most r.*

(i) If s is a forced source, then there is a unique port p_s such that $|\mathcal{C}(s,p_s)| \geq r$. Moreover $(s,*,p_s) \in \mathcal{R}$ and at most $r-1$ triples of $\mathcal{C}$ have source s and port different from p_s.

(ii) If s is a loose source, then $|\mathcal{C}(s)| \leq 2r-1$.

Proof. We may assume that for each source s, $\mathcal{R}$ contains at most one source triple with source s, for otherwise the last source triple of source s is useless and can be deleted.

Observe that for a given source s, all triples of $\mathcal{C}(s)$ that are not routed by the (possibly inexistant) source triple with source s are routed by different triples of $\mathcal{R}$.

Assume that s is a forced source. Let p_s a port such that $|\mathcal{C}(s,p_s)| \geq r$, then, by the above observation, $(s,*,p_s) \in \mathcal{R}$. Moreover all triples of $\mathcal{C}(s) \setminus \mathcal{C}(s,p_s)$ are routed by different triples. There are at most $r-1$ such triples, so $|\mathcal{C}(s) \setminus \mathcal{C}(s,p_s)| \leq r-1$. This proves (i).

Assume now that s is a loose source. If there is no $*$-triple with source s in $\mathcal{R}$, then, by the above observation, all triples of $\mathcal{C}(s)$ are routed by different triples, so $|\mathcal{C}(s)| \leq r$. If there is a triple $(s,*,p)$ in $\mathcal{R}$, then all triples of $\mathcal{C}(s) \setminus \mathcal{C}(s,p)$ are routed by different triples, so $|\mathcal{C}(s) \setminus \mathcal{C}(s,p)| \leq r-1$, and so $|\mathcal{C}(s)| \leq 2r-1$ because $|\mathcal{C}(s,p)| \leq r$ since s is a loose source. This proves (ii). ◊

Let s be a source in S'. Let $T_1(s)$ be the set of destinations of the triples in $C(s) \setminus \mathcal{C}(s,p_s)$ if s is forced, and in $C(s)$ if s is loose. Set $T_1 = \bigcup_{s \in S'} T_1(s)$ and $T_2 = T \setminus (T' \cup T_1)$. The kernelization algorithm computes $T_1(s)$ for all $s \in S'$. If a source s is forced and $|T_1(s)| \geq r$, then by Claim 17.1-(i), $\mathrm{rmin}(\mathcal{C}) > r$, and we return 'No'. In the same way, If a source s is forced and $|T_1(s)| \geq 2r$, then by Claim 17.1-(ii), $\mathrm{rmin}(\mathcal{C}) > r$, and we return 'No'. Henceforth, we may assume that $|T_1(s)| \leq 2r-1$ for every $s \in S'$ and so $|T_1| \leq (2r-1)|S'|$.

Similarly, for a destination $t \in T'$, we define $S_1(t)$ as the set of destinations of the triples in $C(s) \setminus \mathcal{C}(t,p_s)$ if t is forced, and in $C(t)$ if t is loose, and we set $S_1 = \bigcup_{t \in T'} S_1(t)$ and $S_2 = S \setminus (S' \cup S_1)$. The kernelization algorithm computes $S_1(t)$ for all $t \in T'$, and returns 'No' if $|S_1(t)| \geq 2r$. Henceforth, we may assume $|S_1| \leq (2r-1)|T'|$.

Claim 17.2. *Let $\mathcal{R}$ be a shortest routing list emulating $\mathcal{C}$. There is no destination triple with destination in T_2.*

Proof. Suppose for a contradiction that a shortest routing list $\mathcal{R}$ emulating $\mathcal{C}$ contains a destination triple τ_2 with destination $t_2 \in T_2$. Let $\mathcal{R}'$ be list obtained from $\mathcal{R}$ by deleting τ_2. We shall prove that $\mathcal{R}'$ emulates $\mathcal{C}$ which implies that $\mathcal{R}$ is not a shortest routing list emulating $\mathcal{C}$, a contradiction.

Since we only remove τ_2 from $\mathcal{R}$ it suffices to prove that all triples with destination t_2 are properly routed. Observe that since $t_2 \in T_2$, all triples with destination t_2 are of the form (s,t_2,p_s) for a forced source s. Now for such a source, the source triple $(s,*,p_s)$ exists. Moreover, in $\mathcal{R}'$, there is no destination triple with destination t_2, because $\mathcal{R}$ was a shortest routing list containing τ_2, and communication triple of the form (s,t_2,p) for such a triple would have been useless in $\mathcal{R}$. Therefore, $\mathcal{R}'$ properly routes (s,t_2,p_s) using $(s,*,p_s)$. ◊

Let $\mathcal{C}^\#$ be the set of communication triples of $\mathcal{C}$ with source in $S \setminus S_2$ and destination in $T \setminus T_2$. Let S'' be a set of $r+1$ sources disjoint from $S \setminus S_2$ and T'' be a set of $r+1$ destinations disjoint from $T \setminus T_2$. Set

$$\begin{aligned} \mathcal{C}'' \;=\; \mathcal{C}^\# \;&\cup\; \{(s,t'',p_s) \mid t'' \in T'' \text{and } s \text{ forced source of } \mathcal{C}\} \\ &\cup\; \{(s'',t,p_t) \mid s'' \in S'' \text{and } t \text{ forced destination of } \mathcal{C}\} \end{aligned}$$

Claim 17.3. $\mathrm{rmin}(\mathcal{C}) \leq r$ *if and only if* $\mathrm{rmin}(\mathcal{C}'') \leq r$.

Proof. $\mathcal{C}$ and $\mathcal{C}''$ coincide on $\mathcal{C}^\#$, and, by construction of $\mathcal{C}''$, they have the same forced sources and destinations. Moreover, by Claim 17.2, for a shortest routing list emulating $\mathcal{C}$ (resp. $\mathcal{C}''$) the triples not in $\mathcal{C}^\#$ are routed by the source or destination triples of the forced sources and destinations. Therefore a shortest routing list emulating $\mathcal{C}$ is also a routing list emulating $\mathcal{C}''$ (provided it has size at most r), and vice versa. ◊

Claim 17.3 states that $\mathcal{C}$ and $\mathcal{C}''$ are equivalent. Now $\mathcal{C}^\#$ has size at most $|S' \times T'| + |S' \times T_1| + |S_1 \times T'| \leq |S'| \cdot |T'| + (2r-1)|S'|^2 + (2r-1)|T'|^2 \leq (2r-1)r^2$, because $|S'| + |T'| \leq r$. Thus $|\mathcal{C}''| = |\mathcal{C}^\#| + 2r(r+1) \leq 2r^3 + r^2 + r$. Hence the algorithm returns $\mathcal{C}''$. □

In order to keep the above proof simple, we made no attempt to make the size of $\mathcal{C}''$ as small as possible. One can easily get smaller bounds than $2r^3 + r^2 + r$, but we did not find a way to have a quadratic bound. This leaves the following question open.

Problem 18. Does Routing List parameterized by r admits a linear or quadratic kernel?

Theorem 19. With-global Routing List *parameterized by r admits a cubic Turing kernel, and so is FPT.*

Proof. Let us describe a kernelization algorithm that given an instance $(\mathcal{C}, r)$ of Routing List, returns either 'No' only if $\text{rmin}(\mathcal{C}) \geq r$, or an set of r instances $(\mathcal{C}_i, r)$, $1 \leq i \leq r$, of size at most $???r^3$, such that $\text{rmin}(\mathcal{C}) \geq r$ if and only if $\text{rmin}(\mathcal{C}_i) \geq r$ for some i.

□

Problem 20. Does With-global Routing List parameterized by r admits a linear or quadratic Turing kernel?

5 Approximation algortihms

5.1 Relationship between the problems with and without global triple

Lemma 21. *If* rmin^* *can be approximated with ratio α in polynomial time, then* rmin *can be approximated with ratio $\alpha + 1$ in polynomial time.*

Proof. Assume that there is an algorithm $\mathbb{A}^*$, that given a set of communication triples $\mathcal{C}$, returns a list $\mathcal{R}^*$ emulating $\mathcal{C}$ with global triples such that $|\mathcal{R}^*| \leq \alpha\, \text{rmin}^*(\mathcal{C})$.

Let $\mathbb{A}$ be the algorithm that does the following: It first runs $\mathbb{A}^*$ to get a list $\mathcal{R}^*$ emulating $\mathcal{C}$ such that $|\mathcal{R}^*| \leq \alpha\, \text{rmin}^*(\mathcal{C})$. If $\mathcal{R}^*$ contains no global triple, then it returns $\mathcal{R} := \mathcal{R}^*$. Then $|\mathcal{R}| \leq \alpha\, \text{rmin}^*(\mathcal{C}) \leq \alpha\, \text{rmin}(\mathcal{C})$. Assume now that $\mathcal{R}^*$ contains a global triple. Free to delete, the useless triples after the first global triple, we may assume that $\mathcal{R}^*$ has a unique global triple, and that it is the final one. Let $\tau = (*, *, p)$ be this triple, and let $\mathcal{C}_\tau$ be the set of triples of $\mathcal{C}$ that are routed by it when routing according $\mathcal{R}^*$. By definition, all the triples of $\mathcal{C}_\tau$ have port p. Therefore, by Theorem 5, one can find in polynomial time a shortest routing list $\mathcal{R}_\tau$ emulating $\mathcal{C}_\tau$. We then return the concatenation $\mathcal{R}$ of $\mathcal{R}^* \setminus \tau$ and $\mathcal{R}_\tau$. Now $|\mathcal{R}| \leq |\mathcal{R}^*| + |\mathcal{R}_\tau| \leq \alpha\, \text{rmin}^*(\mathcal{C}) + \text{rmin}(\mathcal{C}_\tau) - 1$. But $\text{rmin}^*(\mathcal{C}) \leq \text{rmin}(\mathcal{C})$ and $\text{rmin}(\mathcal{C}_\tau) \leq \text{rmin}(\mathcal{C})$ because $\mathcal{C}_\tau \subseteq \mathcal{C}$. Hence $|\mathcal{R}| \leq (\alpha + 1)\, \text{rmin}(\mathcal{C}) - 1$. □

Lemma 22. *If* sav *can be approximated with ratio α in polynomial time, then* sav^* *can be approximated with ratio $\alpha + 1$ in polynomial time.*

Proof. Assume that there is an algorithm $\mathbb{A}$, that given a set of communication triples $\mathcal{C}$, returns a list $\mathcal{R}$ emulating $\mathcal{C}$ such that $\text{sav}(\mathcal{C}) \leq \alpha(|\mathcal{C}| - |\mathcal{R}|)$.

Let $\mathbb{A}^*$ be the algorithm that does the following:

1. It first runs $\mathbb{A}$ to get a list $\mathcal{R}$ emulating $\mathcal{C}$ such that $\text{sav}(\mathcal{C}) \leq \alpha(|\mathcal{C}| - |\mathcal{R}|)$.
2. Next, it finds the port p that appears the most in $\mathcal{C}$. Let $\mathcal{C}_p$ be the set of triples of $\mathcal{C}$ with port p. It constructs a list $\mathcal{R}'$ made of all elements of $\mathcal{C} \setminus \mathcal{C}_p$ (in any order) followed by the global triple $(*, *, p)$.
3. It returns the list $\mathcal{R}^*$ which is the shortest among $\mathcal{R}$ and $\mathcal{R}'$.

Since $\mathbb{A}$ runs in polynomial time, so does $\mathbb{A}^*$, because the steps 2 and 3 above can trivially be performed in polynomial time. Let us now prove that $\mathbb{A}^*$ gives an $(\alpha+1)$-approximate solution, that is $\mathrm{sav}^*(\mathcal{C}) \leq (\alpha+1)(|\mathcal{C}| - |\mathcal{R}^*|)$. By Lemma 2, we have $\mathrm{sav}^*(\mathcal{C}) \leq \mathrm{sav}(\mathcal{C}) + M(\mathcal{C}) - 1$. But $\mathrm{sav}(\mathcal{C}) \leq \alpha(|\mathcal{C}| - |\mathcal{R}|) \leq \alpha(|\mathcal{C}| - |\mathcal{R}^*|)$, and by definition, $|\mathcal{C}| - |\mathcal{R}^*| \geq |\mathcal{C}| - |\mathcal{R}'| = M(\mathcal{C}) - 1$. It follows that $\mathrm{sav}^*(\mathcal{C}) \leq (\alpha+1)(|\mathcal{C}| - |\mathcal{R}^*|)$. □

5.2 Approximate upper bounds on sav and sav*

Let $\mathcal{C}$ be a set of communication triples with destination set S and destination set T. Set $n = |S|$ and $m = |T|$. We can order S and T by decreasing order according to the function M. That is $M(s_1) \geq M(s_2) \geq \cdots \geq M(s_n)$ and $M(t_1) \geq M(t_2) \geq \cdots \geq M(t_m)$.

The directed $\{0,1,\ldots,n\} \times \{0,1,\ldots,m\}$-grid, denoted by $G_{n,m}$ is the digraph defined by

$$\begin{aligned} V(G_{n,m}) &= \{(i,j) \mid 0 \leq i \leq n \text{ and } 0 \leq j \leq m\} \\ A(G_{n,m}) &= \{((i-1,j),(i,j)) \mid 1 \leq i \leq n \text{ and } 0 \leq j \leq m\} \\ &\quad \cup \{((i,j-1),(i,j)) \mid 0 \leq i \leq n \text{ and } 1 \leq j \leq m\}. \end{aligned}$$

For convenience, we set $a^h(i,j) = ((i-1,j),(i,j))$ and $a^v(i,j) = ((i,j-1),(i,j))$. The set $A^h(G_{n,m} = \{a^h(i,j) \mid 1 \leq i \leq n \text{ and } 0 \leq j \leq m\}$ is the set of *horizontal arcs*, and the set $A^v(G_{n,m} = \{a^v(i,j) \mid 0 \leq i \leq n \text{ and } 1 \leq j \leq m\}$ is the set of *vertical arcs*.

A *edge-weighted digraph* is a pair (G,w) where G is a digraph and w a *weight function* on its arcs, that is a function from $A(D)$ into $\mathbb{R}$. The *length* of a path U in an edge-weighted digraph (G,w) is the sum of the weights of its arcs: $w(U) = \sum_{a \in A(U)} w(a)$.

Let $w_{\mathcal{C}}$ be the weight function defined on $A(G_{n,m})$ by $w_{\mathcal{C}}(a^h(i,j)) = \min\{M(s_i), m-j\} - 1$ and $w_{\mathcal{C}}(a^v(i,j)) = \min\{M(t_j), n-i\} - 1$.

Let $W(\mathcal{C})$ be the maximum length of a path in $(G_{n,m}, w_{\mathcal{C}})$. Observe that it is attained by a path from $(0,0)$ to one of the sides $\{n\} \times \{0,1,\ldots,n\}$ and $\{0,1,\ldots,n\} \times \{n\}$ because the weight of an arc is negative if and only if it is in $\{a^h(i,m) \mid 1 \leq i \leq n\} \cup \{a^v(n,j) \mid 1 \leq j \leq m\}$.

Theorem 23. *$\mathrm{sav}(\mathcal{C}) \leq W(\mathcal{C})$ and $\mathrm{rmin}(\mathcal{C}) \geq |\mathcal{C}| - W(\mathcal{C})$, for all set of communication triples $\mathcal{C}$.*

Proof. Let $\mathcal{R}$ be a canonical routing list with no global triple that emulates $\mathcal{C}$, and let $\mathcal{B}_1, \ldots, \mathcal{B}_q$ be the ordered list of its blocks. For $1 \leq k \leq q$, let T_{ℓ_k} be the $*$-triple of $\mathcal{B}_k$, let r_k be the number of triples of $\mathcal{C}$ routed by T_{ℓ_k}, and let i_k (resp. j_k) be the number of source triples (resp. destination triples) with index at most ℓ_k. We have

$$|\mathcal{R}| = |\mathcal{C}| - \Sigma \quad \text{with} \quad \Sigma = \sum_{k=1}^{q} (r_k - 1).$$

Let K_s (resp. K_t) be the set of indices k such that T_{ℓ_k} is a source (resp. destination) triple. For $k \in K_s$, s'_k be the source of T_{ℓ_k}, and for $k \in K_t$, t'_k be the destination of T_{ℓ_k}.

Assume $k \in K_s$. When routing according to $\mathcal{R}$, before considering the block $\mathcal{B}_k$, there are at most $m - j_k$ triples of $\mathcal{C}$ with source s'_k and at most $M(s'_k)$ triples of $\mathcal{C}$ with source s'_k. Therefore, $r_k \leq \min\{M(s'_k), m - j_k\}$.
Similarly, if $k \in K_t$, then $r_k \leq \min\{M(t'_k), n - i_k\}$. Thus

$$\Sigma \leq \sum_{k \in K_s} (\min\{M(s'_k), m - j_k\} - 1) + \sum_{k \in K_t} (\min\{M(t'_k), n - i_k\} - 1).$$

Set $u_0 = (0,0)$ and $u_k = (i_k, j_k)$ for $1 \leq k \leq q$. Note that $i_k + j_k = k$ for all $0 \leq k \leq q$. The sequence $U = u_0, u_1, \ldots, u_q$ can be seen as a path in the directed grid $G_{n,m}$. Observe that if $k \in K_s$ then $u_{k-1}u_k$ is a horizontal arc and $w_{\mathcal{C}}(u_{k-1}u_k) = \min\{M(s_{i_k}), m - j_k\} - 1$, and if $k \in K_t$ then $u_{k-1}u_k$ is a vertical arc and $w_{\mathcal{C}}(u_{k-1}u_k) = \min\{M(t_{j_k}), n - i_k\} - 1$. Hence

$$w_{\mathcal{C}}(U) = \sum_{k \in K_s} (\min\{M(s_{i_k}), m - j_k\} - 1) + \sum_{k \in K_t} (\min\{M(t_{j_k}), n - i_k\} - 1).$$

Now since $M(s_1) \geq M(s_2) \geq \cdots \geq M(s_n)$ and $j_1 \leq j_2 \leq \cdots \leq j_q$, we have $\sum_{k \in K_s} (\min\{M(s'_k), m - j_k\} - 1) \leq \sum_{k \in K_s} (\min\{M(s_{i_k}), m - j_k\} - 1)$. Similarly, $\sum_{k \in K_t} (\min\{M(t'_k), n - i_k\} - 1) \leq \sum_{k \in K_t} (\min\{M(t_{j_k}), n - i_k\} - 1)$. Hence $\Sigma \leq w_{\mathcal{C}}(U)$. Thus $\Sigma \leq W(\mathcal{C})$, and so $\text{sav}(\mathcal{C}) \leq W(\mathcal{C})$ and $|\mathcal{R}| \geq |\mathcal{C}| - W(\mathcal{C})$. □

From Theorem 23 we now derive that computing a source-based routing list and a destination-based one, and taking the smaller one, yields a list reduces the list of at least one half of the optimum.

Theorem 24. *Let $\mathcal{C}$ be a set of communication triples. Then*

$$Z(\mathcal{C}) \leq \text{sav}(\mathcal{C}) \leq W(\mathcal{C}) \leq 2Z(\mathcal{C}).$$

Proof. By Equation (1) and Theorem 23, we have $Z(\mathcal{C}) \leq \text{sav}(\mathcal{C}) \leq W(\mathcal{C})$.

Let us now prove $W(\mathcal{C}) \leq 2Z(\mathcal{C})$. Let U be a longest path in $(G_{n,m}, w_{\mathcal{C}})$. We have

$$\begin{aligned} W(\mathcal{C}) &= w_{\mathcal{C}}(U) \\ &= \sum_{a^h(i,j) \in A(U)} (\min\{M(s_i), m - j\} - 1) + \sum_{a^v(i,j) \in A(U)} (\min\{M(t_j), n - i\} - 1) \\ &\leq \sum_{a^h(i,j) \in A(U)} (M(s_i) - 1) + \sum_{a^v(i,j) \in A(U)} (M(t_j) - 1) \end{aligned}$$

Now for every $1 \leq i \leq n$, U contains at most one arc in $\{a^h(i,j) \mid 1 \leq j \leq m\}$ and for every $1 \leq j \leq m$, U contains at most one arc in $\{a^v(i,j) \mid 1 \leq i \leq n\}$. Therefore

$$\begin{aligned} W(\mathcal{C}) &\leq \sum_{i=1}^{n} (M(s_i) - 1) + \sum_{j=1}^{m} (M(t_j) - 1) \\ &\leq Z^-(\mathcal{C}) + Z^|(\mathcal{C}) \\ &\leq 2Z(\mathcal{C}) \end{aligned}$$

□

Since the source-based and destination-based routing lists can be computed in polynomial time, computing $Z(\mathcal{C})$

Corollary 25. *The Destination-based Heuristic is a 2-approximation for* List Reduction.

Together with Lemma 22, this corollary immediately yields the following.

Corollary 26. *There is a 3-approximation for* With-global List Reduction.

Observe that $W(\mathcal{C})$ is often smaller than $2Z(\mathcal{C})$. Therefore the Destination-based Heuristic often returns a routing list that saves more than a half of $\text{sav}(\mathcal{C})$ triples. This leads to think that the approximation of 2 is not best possible.

Problem 27. What is the best approximation ratio for List Reduction?

In addition, $W(\mathcal{C})$ can be computed in polynomial time. Indeed the longest path from a vertex u to a set X of vertices in a edge-weighted acyclic digraph (D, w) can be computed as follows. First we find a topological order of D, that is an order $v_1, \ldots, v_n$ such that if $v_i v_j \in A(D)$ then $i < j$. We initialize $p(u) = 0$ and $p(v) = -\infty$ for all $v \in V(D) \setminus \{u\}$. Then for $i = 1$ to n, we compute $p(v_i) := \max\{p(v_j) + w(v_j v_i) \mid v_j v_i \in A(D)\}$ and we return $\max\{p(x), x \in X\}$. Finding a topological order of an acyclic digraph can be done in linear time, that is in time $O(|V(D)| + |A(D)|)$. Thus the above algorithm computes the longest path from u to X in linear and thus can be used to compute $W(\mathcal{C})$ in $O(n + m)$ time. Therefore computing the ratio $W(\mathcal{C})/Z(\mathcal{C})$ gives in polynomial an upper bound on the approximation ratio of the above heuristic for each $\mathcal{C}$.

Theorem 23 is also useful because we can sometimes estimate $W(\mathcal{C})$ and thus obtain good lower bounds on rmin. Forthwith is an example.

Theorem 28. *Let S be a set of n sources, T be a set of n destinations. Let $\mathcal{C}$ be a set of communication triples for $S \times T$. If for every source, at most M communications with source s are assigned the same port, and for every destination t, at most M of the communications with destination t are assigned the same port, then* $\mathrm{rmin}(\mathcal{C}) \geq (n - M)^2 + 2n - M$ *and* $\mathrm{rmin}^*(\mathcal{C}) \geq (n - M)^2 + n - M + 1$.

Proof. We shall prove $\mathrm{rmin}(\mathcal{C}) \geq (n - M)^2 + 2n - M$. By Lemma 1, this implies $\mathrm{rmin}^*(\mathcal{C}) \geq (n - M)^2 + n - M + 1$.

Since $|\mathcal{C}| = n^2$, by Theorem 23, it suffices to prove that $W(\mathcal{C}) = (2n - M)(M - 1)$.

Let $U = u_0, u_1, \ldots, u_q$ be a longest path in $(G_{n,m}, w_{\mathcal{C}})$, with $u_k = (i_k, j_k)$. Let k_0 be the greater index such that $i_{k_0} \leq n - M$ and $j_{k_0} \leq n - M$. By symmetry of this edge-weighted digraph, we may assume that $i_{k_0+1} = n - M + 1$. Let us prove that U has the following properties.

(a) U contains no subpath $(i, j), (i+1, j), (i+1, j+1)$ for $n - M \leq i \leq n$ and $1 \leq j < n - M$.

(b) U does not end with the subpath $(n - M, j), \ldots, (n, j)$ for $1 \leq j < n - M$.

(c) U does not end with the subpath $(i, j), (i, j+1), (i+1, j+1), \ldots, (n, j+1)$ for $n - M \leq i \leq n$ and $n - M \leq j \leq n - 1$.

If U has a subpath $(i, j), (i + 1, j), (i + 1, j + 1)$ for some i and j such that $n - M \leq i \leq n$ and $1 \leq j < n - M$, then the path U' obtained from U by replacing $(i, j), (i + 1, j), (i + 1, j + 1)$ by $(i, j), (i, j + 1), (i + 1, j + 1)$ satisfies $w_{\mathcal{C}}(U') = w_{\mathcal{C}}(U) + 1$, a contradiction to the maximality of $w_{\mathcal{C}}(U)$. This proves (a).

If U ends with the subpath $(n - M, j), \ldots, (n, j)$ for some j such that $1 \leq j < n - M$, then the sequence U' obtained from U by replacing $(n - M, j), \ldots, (n, j)$ by $(n - M, j), (n - M, j + 1), \ldots, (n, j + 1)$ satisfies $w_{\mathcal{C}}(U') = w_{\mathcal{C}}(U) + n - M$, a contradiction to the maximality of $w_{\mathcal{C}}(U)$. This proves (b).

If U ends with the subpath $(i, j), (i, j + 1), (i + 1, j + 1), \ldots, (n, j + 1)$ for some a and b such that $n - M \leq i \leq n$ and $n - M \leq j \leq n - 1$, then the path U' obtained from U by replacing $(i, j), (i+1, j), (i+1, j+1), \ldots, (n, j+1)$ by $(i, j), (i+1, j), \ldots, (n, j)$ satisfies $w_{\mathcal{C}}(U') = w_{\mathcal{C}}(U)+1$, a contradiction to the maximality of $w_{\mathcal{C}}(U)$. This proves (c).

Now (a) and (b) imply that $u_{k_0} = (n - M, n - M) = u_{2n-2M}$ and (c) implies that U ends with $(n - M, n - M), \ldots, (n - M, n)$. In particular, the path U has $2n - M$ arcs. Moreover, each arc of U has weight $M - 1$. It follows that $W(\mathcal{C}) = w_{\mathcal{C}}(U) = (2n - M)(M - 1)$. □

For every M, the bound given by Theorem 28 is tight.

Proposition 29. *For every integer M and n such that $M \leq n$, there exists a set $\mathcal{C}$ of triples with n sources and n destinations such that*

(i) for every source, at most M communications with source s are assigned the same port,

(ii) for every destination t, at most M of the communications with destination t are assigned the same port, and

(iii) $\text{rmin}(\mathcal{C}) = (n-M)^2 + 2n - M$.

Proof. Let $\mathcal{C}$ be the set of triples $(s, t, p(s,t))$ for $1 \leq s \leq n$ and $1 \leq t \leq n$ such that

$$p(s,t) = \left\lceil \frac{s}{M} \right\rceil + \left\lceil \frac{t}{M} \right\rceil \mod \left\lceil \frac{n}{M} \right\rceil.$$

Conditions (i) and (ii) are clearly satisfied. Consider now the routing list $\mathcal{R}$ which is the concatenation of $\{(s,t,p(s,t)) \mid 2 \leq s \leq n \text{ and } 2 \leq t \leq n\}$, $\{(*,t,p(1,t) \mid 2 \leq t \leq n\}$ and $\{(s,*,p(s,1) \mid 1 \leq s \leq n\}$. One can check that $\mathcal{R}$ emulates $\mathcal{C}$ and $|\mathcal{R}| = (n-M)^2 + 2n - M$. Thus $\text{rmin}(\mathcal{C}) \leq (n-M)^2 + 2n - M$, and so by Theorem 28, $\text{rmin}(\mathcal{C}) = (n-M)^2 + 2n - M$. □

6 Problem with two ports

6.1 Relation with the Feedback Arc Set problem

Let $\mathcal{C}$ be a set of communication triples, with source set $S = \{s_1, \ldots, s_n\}$, destination sets $T = \{t_1, \ldots, t_m\}$ and port set $\{p_1, p_2\}$. We associate to $\mathcal{C}$ the $n \times m$ matrix $A = A_{\mathcal{C}}$ defined by $a_{i,j} = 1$ if $(s_i, t_j, p_1) \in \mathcal{C}$, $a_{i,j} = -1$ if $(s_i, t_j, p_2) \in \mathcal{C}$, and $a_{i,j} = 0$ otherwise. We also associate to $\mathcal{C}$ the bipartite digraph $D_{\mathcal{C}}$ with vertex set $S \cup T$ in which for all $s \in S$ and all $t \in T$, st is an arc if and only if $(s,t,p_1) \in \mathcal{C}$, and ts is an arc if and only if $(s,t,p_2) \in \mathcal{C}$. Hence, $A_{\mathcal{C}}$ is the biadjacency matrix of the bipartite digraph $D_{\mathcal{C}}$.

Observe that $\mathcal{C} \to A_{\mathcal{C}}$ is a one-to-one correspondence between the sets of communication triples, with source set $S = \{s_1, \ldots, s_n\}$, destination sets $T = \{t_1, \ldots, t_m\}$ and port set $\{p_1, p_2\}$, and the $\{-1, 0, 1\}$-entry $n \times m$ matrices. Similarly, $\mathcal{C} \to D_{\mathcal{C}}$ is a one-to-one correspondence between the sets of communication triples, with source set $S = \{s_1, \ldots, s_n\}$, destination sets $T = \{t_1, \ldots, t_m\}$ and port set $\{p_1, p_2\}$ and the labelled bipartite oriented graphs (an oriented graph is a digraph with no 2-cycles) with vertex set $S \cup T$.

Let us make some easy observations on $\mathcal{D}_{\mathcal{C}}$. In a digraph D, for any vertex v, we denote by $A^+_D(v)$, or simply $A^+(v)$ when D is clear from the context, the set of arcs leaving v. Similalry, we denote by by $A^-_D(v)$, or simply $A^-(D)$, the set of arcs entering v.

Fact 30. *1. The communication triples of $\mathcal{C}$ are in one-to-one correspondence to the arcs of $\mathcal{D}_{\mathcal{C}}$.*

2. If $s \in S$, then $A^+(s)$ corresponds to the set of communication triples with source s and port p_1 and $A^-(s)$ corresponds to the set of communication triples with source s and port p_2.

3. If $t \in T$, then $A^+(t)$ corresponds to the set of communication triples with destination t and port p_1 and $A^-(s)$ corresponds to the set of communication triples with destination t and port p_2.

In view of Fact 30.1, for sake of clarity, we often identify the arcs of $\mathcal{D}_\mathcal{C}$ with their corresponding communication triples.

Lemma 31. *Let $\mathcal{C}$ be a set of communication triples and $\mathcal{R}$ be a routing list of $\mathcal{C}$. The set of communication triples in $\mathcal{R}$ corresponds to a feedback arc set in $D_\mathcal{C}$.*

Proof. Since in the operations described in Lemma 4 to obtain a canonical routing list, no communication triple is added, it suffices to prove the results for canonical routing lists. Moreover, if $\mathcal{R}$ contains a global triple $(*,*,p)$, then we can replace it by the $(s,*,p)$ for all sources s of $\mathcal{R}$ without increasing the number of

We prove the result by induction the sum of the numbers of sources and destinations of $\mathcal{C}$, the result vacuously true if $\mathcal{C}$ is empty.

Let $\mathcal{C}$ be a non-empty set of communication triples and set $D = D_\mathcal{C}$. Let $\mathcal{R}$ be a canonical routing list with no global triple emulating $\mathcal{C}$ and let $\mathcal{R}'$ be the set of communication triples of $\mathcal{R}$. For each block $\mathcal{B}_i$ of $\mathcal{R}$, let T_i be the last triple of $\mathcal{B}_i$. Set $\mathcal{B}'_i = \mathcal{B}_1 \cap \mathcal{R}'$. We have $\mathcal{B}'_i = \mathcal{B}_i \setminus \{T_i\}$.

Suppose that T_1 is a source triple, say $(s,*,p)$. Set $p' = \{p_1,p_2\} \setminus \{p\}$. Then all the communication triples with source s and port p' must be routed before T_1, and thus belong to $\mathcal{B}_1$. Hence if $p' = p_1$ then $A^+_D(s) \subseteq \mathcal{B}'_1$, and if $p' = p_2$ then $A^-_D(s) \subseteq \mathcal{B}'_1$. In particular, there is no cycle through s in $D \setminus \mathcal{B}'_1$. Let $\mathcal{C}'$ be the set of triples obtained from $\mathcal{C}$ by removing the triples with source s. Then $D' = D_{\mathcal{C}'}$ is the digraph $D - s$ and $\mathcal{R} \setminus \mathcal{B}_1$ emulates $\mathcal{C}'$. By the induction hypothesis, $\mathcal{R}' \setminus \{\mathcal{B}'_1\}$, which is the set of communication triples of $\mathcal{R} \setminus \mathcal{B}_1$ corresponds to a feedback arc set in $D - s$. Therefore, $\mathcal{R}'$ is a feedback arc set in D.

Similarly, we get the result if T_1 is a destination triple. □

The following lemma is a kind of reciprocal to Lemma 31.

Lemma 32. *Let $\mathcal{C}$ be a set of communication triples with n sources and m destinations. If $D_\mathcal{C}$ is acyclic, then there is a routing list emulating $\mathcal{C}$ containing at most $n+m-1$ source or destination triples and no other triples.*

Proof. By induction on $n+m$, the result holding trivially when $\mathcal{C}$ has one source and one destination.

Suppose that $\mathcal{C}$ is a set of communication triples with n sources and m destinations, where $n+m \geq 3$. If $D_\mathcal{C}$ is acyclic, then it contains a vertex v with in-degree 0. If this vertex is a source (resp. destination), then all the communication triples with source (resp. destination) v have port p_1 (resp. p_2). Hence using first the triple $(v,*,p_1)$ (resp. $(*,v,p_2)$), all communication triples of $\mathcal{C}$ with source (resp. destination) v are routed. Therefore, the set $\mathcal{C}'$ of communication triples that remain to be routed has one source or destination less than $\mathcal{C}$. Moreover $D_{\mathcal{C}'}$ is $D_\mathcal{C} - v$, so it is acyclic. Thus, by the induction hypothesis, there is a routing list emulating $\mathcal{C}'$ containing $n+m-2$ source or destination triples. and no other triples. The concantenation of $(v,*,p_1)$ (resp. $(*,v,p_2)$) with this list gives the desired routing list emulating $\mathcal{C}$. □

The previous two lemmas implies that $\mathrm{rmin}^*(\mathcal{C})$ and $\mathrm{rmin}(\mathcal{C})$ are closely related to $\mathrm{fas}(D_\mathcal{C})$.

Corollary 33. *If $\mathcal{C}$ be a set of communication triples with n sources and m destinations, then* $\mathrm{fas}(D_\mathcal{C}) + 1 \leq \mathrm{rmin}^*(\mathcal{C}) \leq \mathrm{rmin}(\mathcal{C}) \leq \mathrm{fas}(D_\mathcal{C}) + n + m - 1$.

Proof. Let $\mathcal{R}$ be a canonical routing list emulating $\mathcal{C}$. By Lemma 31, there are at least $\mathrm{fas}(D_\mathcal{C})$ communication triples in $\mathcal{R}$. Moreover, $\mathcal{R}$ contains at least one $*$-triple because it is canonical. Hence $|\mathcal{R}| \geq \mathrm{fas}(D_\mathcal{C}) + 1$. Therefore, $\mathrm{rmin}^*(\mathcal{C}) \geq \mathrm{fas}(D_\mathcal{C}) + 1$.

Let $\mathcal{C}_0$ be the set of communication triples corresponding to a minimum feedback arc set F of $\mathcal{D}_\mathcal{C}$. Set $\mathcal{C}' = \mathcal{C} \setminus \mathcal{C}_0$. The digraph $D'_\mathcal{C}$ is $D_\mathcal{C} \setminus F$ and so is acyclic. Therefore, by Lemma 32,

there is a list $\mathcal{R}'$ of size at most $n+m-1$ emulating $\mathcal{C}'$. Now the list $\mathcal{C}_0, \mathcal{R}'$ has size at most $\text{fas}(D_\mathcal{C}) + n + m - 1$ and emulates $\mathcal{R}$. □

Lemma 34. *If $\mathcal{C}$ is a full set of communication triples with n sources and m destinations, then* $\text{rmin}(\mathcal{C}) \geq \text{fas}(D_\mathcal{C}) + \min\{n, m\}$.

Proof. Let $\mathcal{R}$ be a canonical routing list with no global triple. Since a block routes only the communication with a single source or communication with a single column, the number of blocks is at most $\max\{n, m\}$ communications. Since the set of communications is full, there are nm communications. Hence, there must be at least $nm/\max\{n, m\} = \min\{n, m\}$ blocks. Since a blocks contains a $*$-triple, they are at least $\min\{n, m\}$ $*$-triples in $\mathcal{R}$. As they are at least $\text{fas}(D_\mathcal{C})$ communication triples in $\mathcal{R}$. Hence $\mathcal{R}$ s of size at least $\text{fas}(D_\mathcal{C}) + \min\{n, m\}$. □

6.2 Deriving approximation algorithms

Van Zuylen [18] gave a 4-approximate polynomial-time algorithm for Feedback Arc Set in complete bipartite tournaments.

We can use this approximation algorithm for Feedback Arc Set to build polynomial-time approximation algorithms for some sets ofcommunication triples.

Theorem 35. *There is a polyomial-time 4-approximate algorithm for* Routing List *with no global triple restricted to the full sets of communication triples with n sources, m destinations and two ports such that $n + m - 1 \leq 4 \cdot \min\{n, m\}$.*

Proof. Let $\mathcal{C}$ be a set of communication triples with n sources and m destinations such that $n + m - 1 \leq 4 \cdot \min\{n, m\}$.

The algorithm proceeds as follows. Using Gupta's algorithm it finds a feedback arc set F of $D_\mathcal{C}$ of size at most $4\,\text{fas}(D_\mathcal{C})$. Then, as in the proof of Lemma 32, it derives a routing list $\mathcal{R}$ with no global triple of size at most $4\,\text{fas}(D_\mathcal{C}) + n + m - 1$ which is at most $4\,\text{fas}(D_\mathcal{C}) + 4 \cdot \min\{n, m\}$ by hypothesis. Now by Lemma 34, $\text{rmin}(\mathcal{C}) \geq \text{fas}(D) + \min\{n, m\}$. Therefore, the size of $\mathcal{R}$ is at most $4 \cdot \text{rmin}(\mathcal{C})$. □

We can also derive approximation algorithms for almost full sets of communication triples. For convenience, we present the result when the number of sources equals the number of destinations. They can easily be extended to the case when those numbers differ.

Theorem 36. *Let α be a positive real number. There is a polyomial-time* $\max\{4, 2 + 2\alpha\}$*-approximate algorithm for* Routing List *with no global triple restricted to the sets of at least $n^2 - \alpha \cdot n$ communication triples with n sources, n destinations and two ports.*

Proof. Let $\mathcal{C}$ be a set of at least $n^2 - \alpha \cdot n$ communication triples with n sources and n destinations and two ports.

Let D_1 (resp. D_2) be the digraph obtained by adding an arc st (resp. ts) for each communication (s, t) not appearing in $\mathcal{C}$. Observe that D_i corresponds to the set of communication triples obtained from $\mathcal{C}$ by routing all missing communications to the port p_i. Both D_1 and D_2 are bipartite tournaments.

Consider a minimum feedback arc set S of $D_\mathcal{C}$, and let $\sigma = v_1, \ldots, v_{2n}$ be an ordering such that S is σ-feedback. For each missing communication (s, t) in $\mathcal{C}$, the arc st is *backward* if $s = v_i$, $t = v_j$ and $j < i$, otherwise it is *forward*. Observe that st is forward (resp. backward) if and only if ts is backward (resp. forward). Let $B_1 = \{st \mid st \text{ is backward}\}$ and $B_2 = \{ts \mid$

st is backward}. Then for $i = 1, 2$, $S \cup B_i$ is a feedback arc set of D_i. Moreover $|B_1| + |B_2| = \alpha \cdot n$, so $\min\{|B_1|, |B_2|\} \leq \frac{\alpha}{2} \cdot n$. Hence

$$\min\{\text{fas}(D_1), \text{fas}(D_2)\} \leq |S| + \min\{|B_1|, |B_2|\} \leq \text{fas}(D_{\mathcal{C}}) + \frac{\alpha}{2} \cdot n. \quad (5)$$

Now our approximation algorithm proceeds as follows. It constructs D_1 and D_2, compute a feedback arc set of both and takes the smallest one S_m. It then returns the routing list $\mathcal{R}$ consisting of the communication triples corresponding to the arcs of S_m, by adding at most $2n-1$ appropriate destination of source triples following the proof of Lemma 32.

Now $|R| < |S_m| + 2n \leq 4 \cdot \min\{\text{fas}(D_1), \text{fas}(D_2)\} + 2n \leq 4 \cdot \text{fas}(D_{\mathcal{C}}) + (2\alpha + 2)n$ by Equation (5). But by Lemma 34, $\text{rmin}(\mathcal{C}) \geq \text{fas}(D_{\mathcal{C}}) + n$. Therefore $|R| \leq \max\{4, 2 + 2\alpha\} \cdot \text{rmin}(\mathcal{C})$. □

Problem 37. Can we derive from the fact that Feedback Arc Set is APX-complete, that our problem is also APX-complete? For which approximation ratio?

The complementary problem to Feedback Arc Set is the following. Maximum Acyclic Subdigraph:
Input: A digraph D, and an integer k.
Question: Does D have an acyclic subgraph with at least at k edges?

The number of edges of an acyclic subdigraph of D is denoted $\text{mas}(D)$. Clearly

$$\text{mas}(D) + \text{fas}(D) = |A(D)| \text{ for all digraph } D.$$

Arora et al. [1] proved that Maximum Acyclic Subdigraph admits a polynomial-time approximation scheme on dense instances. A *dense* digraph is one in which the number of arcs is $\Omega(n^2)$.

Theorem 38 (Arora et al. [1])**.** Maximum Acyclic Subdigraph *has an* $n^{O(1/\epsilon^2)}$ *time approximation scheme on dense graphs.*

6.3 Relation with the Star Cover problem

A routing list is *convenient* if it contains only source triples and destination triples. By Lemmas 31 and 32, there is a convenient routing list emulating a set of communication triples $\mathcal{C}$ if and only if $D_{\mathcal{C}}$ is acyclic.

In view of the results of Subsection 6.1, a general method to compute a routing list emulating a given set of communication triples $\mathcal{C}$ is to first compute a set $\mathcal{R}'$ of communication triples corresponding to a feedback arc set of $D_{\mathcal{C}}$, and then to compute a convenient routing list for $\mathcal{C}' = \mathcal{C} \setminus \mathcal{R}'$. The concatenation of $\mathcal{R}'$ and $\mathcal{R}''$ would then be a routing list emulating $\mathcal{C}$. In fact, the approximation algorithm described in Subsection 6.2 use this approach. But they use a non-optimal algorithm for finding the convenient routing list in the second phase.

In this subsection, we show how to optimize the second phase. That is, once the first phase is done, and we are left with the routing list $\mathcal{C}'$, find a shortest convenient routing list emulating $\mathcal{C}'$.

Convenient Routing List:
Input: A set $\mathcal{C}$ of communication triples with two ports, such that $D_{\mathcal{C}}$ is acyclic.
Find: a convenient routing list emulating $\mathcal{C}$ with minimum size?

We first show how this problem is equivalent to the Star Cover Problem for $\mathcal{D}_C$. We then show how to solve the Star Cover Problem for an arbitrary graph in polynomial time.

An *out-star* is a digraph consisting of one vertex, called the *centre*, dominating all the others. An *in-star* is the converse of an out-star, that is a digraph consisting of one vertex, called the *centre*, dominated by all the others. A *star* is either an out-star or an in-star. A *star cover* of a digraph D is a set of stars $\mathcal{S}$ with disjoint centres such that $A(D) = \bigcup_{S\in\mathcal{S}} A(S)$.

We are interested in finding the minimum size of a star cover of D, denoted by $\mathrm{sc}(D)$, and a minimum star cover of D, that is a star cover with $\mathrm{sc}(D)$ stars.

Star Cover
Input: a digraph D.
Find: a minimum star cover.

Observe that if $\mathcal{S}$ is a star cover, then the pair (V^+, V^-), where V^+ and V^- is the set of centres of out-stars and in-stars, respectively, in $\mathcal{S}$, satisfies the following property:

For every arc uv, either $u \in V^+$ or $v \in V^-$. $(\star)$

Reciprocally, assume that (V^+, V^-) is a pair of disjoint sets of vertices satisfying $(\star)$. For every vertex v, let S_v^+ be the out-stars with centre v and arcs $A^+(v)$, and S_v^- be the in-star with centre v and arcs $A^-(v)$. Then $\{S_v^+ \mid v \in V^+\} \cup \{S_v^- \mid v \in V^-\}$ is a star cover of D of size $|V^+|+|V^-|$. Therefore, for convenience, we shall consider that a star cover is pair (V^+, V^-) of disjoint sets of vertices satisfying $(\star)$.

Lemma 39. *Let $\mathcal{C}$ be a set of communication triples such that $D_{\mathcal{C}}$ is acyclic. There is a convenient routing list of size r emulating $\mathcal{C}$ if and only if $D_{\mathcal{C}}$ has a star cover of size r.*

Proof. Let $\mathcal{R}$ be a convenient routing list emulating $\mathcal{C}$. Set

$$\begin{aligned} V^+ &= \{s \mid (s,*,p_1) \in \mathcal{R}\} \cup \{t \mid (*,t,p_2) \in \mathcal{R}\} \quad \text{and} \\ V^- &= \{s \mid (s,*,p_2) \in \mathcal{R}\} \cup \{t \mid (*,t,p_1) \in \mathcal{R}\}. \end{aligned}$$

Clearly $|V^+| + |V^-| = |\mathcal{R}|$.

Let st be an arc of $D_{\mathcal{C}}$ with $s \in S$ and $t \in T$. The communication triple (s,t,p_1) of $\mathcal{C}$ is routed by $\mathcal{R}$. So either $(s,*,p_1)$ or $(*,t,p_1)$ is in $\mathcal{R}$. Hence either $s \in V^+$ or $t \in V^-$. Similarly, if ts is an arc of $D_{\mathcal{C}}$ with $s \in S$ and $t \in T$, then either $t \in V^+$ or $s \in V^-$. Therefore, (V^+, V^-) is a star cover of $D_{\mathcal{C}}$.

Reciprocally, let (V^+, V^-) be a star cover of $D_{\mathcal{C}}$. Set

$$\begin{aligned} \mathcal{R}^+ &= \{(s,*,p_1) \mid s \in V^+\} \cup \{(*,t,p_2) \mid t \in V^+\} \quad \text{and} \\ \mathcal{R}^- &= \{(s,*,p_2) \mid s \in V^-\} \cup \{(*,t,p_1) \mid t \in V^-\}. \end{aligned}$$

Futhermore, because $D_{\mathcal{C}}$ is acyclic, we can order $\mathcal{R}^+$ in such a way that a triple $(s,*,p_1)$ is in front of (resp. goes after) $(*,t,p_2)$ whenever $(s,t,p_1) \in \mathcal{C}$ (resp. $(s,t,p_2) \in \mathcal{C}$). It suffices to take the triples of $\mathcal{R}^+$ according to a linear ordering $v_1, \ldots, v_{n+m}$ of the vertices of $D_{\mathcal{C}}$ with no arcs vjv_i with $i < j$ in D. Similarly, we can order $\mathcal{R}^-$ in such a way that a triple $(s,*,p_2)$ is in front of (resp. goes after) $(*,t,p_1)$ whenever $(s,t,p_2) \in \mathcal{C}$ (resp. $(s,t,p_1) \in \mathcal{C}$).

Now it is easy to check that the concatenation of $\mathcal{R}^+$ and $\mathcal{R}^-$ is a convenient routing list of size $|V^+| + |V^-|$ that emulates $\mathcal{C}$. □

6.3.1 Solving Star Cover by a purely combinatorial algorithm

Lemma 40. *Let D be a digraph and (V^+, V^-) a star cover of D. Let P be a path in D. If the initial vertex of P is in V^-, then all vertices of P are in V^-.*

Proof. By induction on the length of P, the result holding trivially if it has length 0. Let $(v_0, \ldots, v_p)$ be a path of length $p \geq 1$ such that $v_0 \in V^-$. By the induction hypothesis applied to the path $(v_0, \ldots, v_{p-1})$, $v_i \in V^-$ for all $1 \leq i \leq p-1$. In particular, $v_{p-1} \in V^-$, so $v_{p-1} \notin V^+$. Thus v_p must be in V^-. □

If u and v are two vertices, then a *(u,v)-path* is a path with initial vertex u and terminal vertex v. If U is a set and v a vertex, then a a *(U,v)-path* is a path with initial vertex in U and terminal vertex v. A digraph is *strongly connected* or strong, if for any pair (u,v) of vertices, there is a (u,v)-path. A *strong component* of a digraph is a strong subdigraph that is inclusion-wise maximal. A strong component C is *trivial* if it has order 1. It is *minimal* if no arc enters C, and *maximal* if no arc leaves C.

Corollary 41. *If D is strong, then $(V(D), \emptyset)$ and $(\emptyset, V(D))$ are the sole star covers of D. In particular,* $\mathrm{sc}(D) = |V(D)|$.

Proof. Let (V^+, V^-) be a star cover of D.

Assume that V^- contains a vertex u. For any vertex v of D, there is a (u,v)-path, because D is strong. Thus, by Lemma 40, $v \in V^-$. Hence $V^- = V(D)$.

By directional duality, if V^+ is not empty, then $V^+ = V(D)$. □

The *transitive closure* of an acyclic digraph D is the digraph $\vec{T}_D$ with vertex set $V(D)$ and arcs the pairs (u,v) of distinct vertices such that there exists a (u,v)-path in D. A *stable set* in a digraph is a set of pairwise non-adjacent vertices. The *stability number* of a digraph D, denoted $\alpha(D)$, is the maximum size of a stable set.

Lemma 42. *If D is an acyclic digraph, then* $\mathrm{sc}(D) = |V(D)| - \alpha(\vec{T}_D)$.

Proof. Let D be an acyclic digraph.

Let (V^+, V^-) be a minimum star cover of D. Set $U = V(D) \setminus (V^+ \cup V^-)$.

We claim that U is a stable set in $\vec{T}_D$. Indeed, suppose for a contradiction, that U is not a stable set. Then there are two vertices u and v in U such that (u,v) is an arc in $\vec{T}_D$. Hence, by definition of transitive closure, there is a (u,v)-path P in D. Let w be the second vertex of P. Since u is in U, vertex w must be in V^-. But in D, there is a (w,v)-path, namely $P - u$. So, by Lemma 40, $v \in V^-$, a contradiction. This proves our claim that U is a stable set of $\vec{T}_D$.

Hence $|U| \leq \alpha(\vec{T}_D)$. But $|V^+| + |V^-| = |V(D)| - |U|$. Thus $\mathrm{sc}(D) = |V^+| + |V^-| \geq |V(D)| - \alpha(\vec{T}_D)$.

Reciprocally, let U be a maximum stable set in $\vec{T}_D$. Since U is maximal, in $\vec{T}_D$, every vertex v of $V(D) \setminus U$ is either dominated by a vertex of U or dominates a vertex of U. Let V^+ (resp. V^-) be the set of vertices of $V(D) \setminus U$ that dominates (resp. is dominated by) a vertex of U in $\vec{T}_D$.

We first claim that V^+ and V^- are disjoint. Indeed, assume for a contradiction that a vertex v is in $V^+ \cap V^-$. Then there is a vertex u_1 in U dominating v in $\vec{T}_D$, and a vertex u_2 in U dominated by v (possibly $u_1 = u_2$). In D, there is a (u_1, v)-path P_1 and a (v, u_2)-path. The concatenation of P_1 and P_2 contains a (u_1, u_2)-path in D, which contradicts the fact that U is a stable set in D if $u_1 \neq u_2$, or that D is acyclic if $u_1 = u_2$. This proves our first claim.

We also claim that if uv is an arc of D, then either $u \in V^+$ or $v \in V^-$. Indeed, if $u \notin V^+$, then $u \in U \cup V^-$. Thus there is a (U,u)-path from U to u in D (reduced to the vertex u when $u \in U$). Adding the arc uv to this path we obtain a (U,v)-path in D. So $v \in V^-$. This proves our second claim.

The two claims shows that (V^+, V^-) is a star cover Now $|V^+| + |V^-| = |V(D)| - |U| = |V(D)| - \alpha(\vec{T}_D)$. Therefore $\mathrm{sc}(D) \leq |V(D)| - \alpha(\vec{T}_D)$. □

It is well-known that the transitive closure of any acyclic digraph corresponds to a strict partial order. Therefore, the underlying graph of a transitive closure is a comparability graph. Comparability graphs are perfect, so a maximum stable set can be found in polynomial time in such graphs. Hence, for a given acyclic digraph D, a maximum stable set of $\overrightarrow{T}_D$ can be computed in polynomial time, and so by Lemma 42 (and its proof), one can find in polynomial time a minimum star cover.

Theorem 43. Star Cover *is polynomial-time solvable.*

Proof. Let us present a procedure `star-cover`(D), that given a digraph D returns a minimum star cover of D. The idea is to reduce to the case when D is acyclic. When D is acyclic, the procedure runs as we just explained, so we only detail here how to reduce to this case.

We first compute the strong components of D. If D is not acyclic, one of these components, say C, is not trivial. Let U^+ (resp. U^-) be the set of vertices of $V(D-C)$ having an out-neighbour (resp. in-neighbour) in C. Let D_1 be the digraph obtained form $D-C$ by adding all the arcs from U^+ to U^-.

Let (V^+, V^-) be a star cover of D. By Corollary 41, $(V^- \cap V(C), V^+ \cap V(C))$ is either $(V(C), \emptyset)$ or $(\emptyset, V(C))$. In the first case, all arcs entering C must have their tail in V^+, so $U^+ \subseteq V^+$. In the second case, $U^+ \subseteq V^-$. In both cases, it implies that every arc from U^+ to U^- has its tail in V^+ or its head in V^+. Hence $(V^+ \setminus V(C), V^- \setminus V(C))$ is a star cover of D_1.

Reciprocally, assume that $\mathcal{S}_1 = (V_1^-, V_1^+)$ is a star cover of D. Then necessarily $U^+ \subseteq V^+$ or $U^- \subseteq V^-$. If $U^+ \subseteq V_1^+$, then $(V_1^+ \cup V(C), V_1^-)$ is a star cover of D. If $U^- \subseteq V_1^-$, then $(V_1^+, V_1^- \cup V(C))$ is a star cover of D.

To summarize, the mapping θ that associates to each star cover $\mathcal{S} = (V^+, V^-)$ of D, the pair $(V^- \cap V(C), V^+ \cap V(C))$ is a surjection into the set of star covers of D_1. Moreover, $\mathcal{S}| = |\theta(\mathcal{S})| + |C|$.

Therefore the algorithm makes a recursive call to `star-cover`(D_1), which returns a minimum star cover of (V_1^-, V_1^+) of D_1. If $U^+ \subseteq V_1^+$, then it returns $(V_1^+ \cup V(C), V_1^-)$, otherwise it returns $(V_1^+, V_1^- \cup V(C))$. □

6.3.2 Solving Star Cover via linear programming

The Star Cover problem can be formulated as the following ILP.

$$\begin{array}{lrll} \text{Minimize} & \sum_{v \in V(D)} (x_v^+ + x_v^-) & & \\ \text{Subject to:} & x_u^+ + x_v^- \geq 1 & \text{for all } uv \in A(D) & (6) \\ & x_v^+ + x_v^- \leq 1 & \text{for all } v \in V(D) & \\ & x_v^+, x_v^- \in \{0,1\} & \text{for all } v \in V(D) & \end{array}$$

Let $\mathbf{A}_D$ be the matrix associated to this ILP, and let B_D be the bipartite graph defined by

$$\begin{aligned} V(B_D) &= \bigcup_{v \in V(D)} \{x_v^+, x_v^-\} \\ E(B_D) &= \{x_u^+ x_v^- \mid uv \in A(D)\} \cup \{x_v^+ x_v^- \mid v \in V(D)\} \end{aligned}$$

After multiplying the rows of $\mathbf{A}_D$ corresponding to the second constraint by -1, the resulting matrix $\mathbf{A}'_D$ is the incidence matrix of B_D. It is well known that the incidence matrix of a bipartite graph is totally unimodular. Thus $\mathbf{A}'_D$ is totally unimodular and so is $\mathbf{A}_D$.

Theorem 44 (Hoffman and Kruskal [10])**.** *If* $\mathbf{A}$ *is totally unimodular and* $\mathbf{b}$ *is an integral vector, then the linear programme*

$$\textit{Minimize} \quad \mathbf{c^T x} \quad \textit{subject to :}$$

$$\begin{array}{rcl} \mathbf{Ax} & \geq & \mathbf{b} \\ \mathbf{x} & \geq & 0 \end{array}$$

has an integral optimal solution (if it has one).

Corollary 45. $\mathrm{sc}(D)$ *is equal to the optimal solution of the fractional relaxation of the ILP* (6) *and thus can be computed in polynomial time using linear programming.*

Remark 46. In fact, Theorem 44 derives from the fact that $\{\mathbf{x} \mid \mathbf{Ax} \geq \mathbf{b}, \mathbf{x} \geq 0\}$ is an integral polyhedron. This implies in particular that an integral optimal solution can be found in polynomial time.

6.4 Polynomial-time algorithm when the associated digraph is acyclic

Theorem 47. *There is a polynomial-time algorithm that computes* $\mathrm{rmin}\,\mathcal{C}$ *when* $D_{\mathcal{C}}$ *is acyclic.*

The proof of Theorem 47 is based on the following integer linear programming formulation.

$$\begin{array}{lrl} \text{Minimize} & \sum x_{(s,t,p)} + \sum x_{(*,t,p)} + \sum x_{(s,*,p)} & \\ \text{Subject to:} & x_{(s,t,p)} + x_{(*,t,p)} + x_{(s,*,p)} \geq 1 & \text{for all } (s,t,p) \in \mathcal{C} \\ & \sum_p x_{(*,t,p)} \leq 1 & \text{for all } t \in T \\ & \sum_p x_{(s,*,p)} \leq 1 & \text{for all } s \in S \\ & x_{(s,t,p)}, x_{(*,t,p)}, x_{(s,*,p)} \in \{0,1\} & \end{array} \tag{7}$$

This translates in the following ILP on $D = \mathcal{D}_{\mathcal{C}}$, which extends (6)

$$\begin{array}{lrl} \text{Minimize} & \displaystyle\sum_{a \in A(D)} x_a + \sum_{v \in V(D)} (x_v^+ + x_v^-) & \\ \text{Subject to:} & x_{uv} + x_u^+ + x_v^- \geq 1 & \text{for all } uv \in A(D) \\ & x_v^+ + x_v^- \leq 1 & \text{for all } v \in V(D) \\ & x_a, x_v^+, x_v^- \in \{0,1\} & \end{array} \tag{8}$$

Let us prove that this ILP computes $\mathrm{rmin}(\mathcal{C})$ if $\mathcal{D}_C$ is acyclic.

Proposition 48. *If* $\mathcal{D}_C$ *is acyclic, then the ILP* (7) *computes* $\mathrm{rmin}(\mathcal{C})$.

Proof. Let x^* be an optimal solution of the ILP (7) for $\mathcal{C}$ and let r^* be the optimal value of the objective function. Let $\mathcal{R}$ be a shortest routing list with no global triples emulating $\mathcal{C}$.

Let $x^{\mathcal{R}}$ be the characteristic function of R, that is $x^{\mathcal{R}}_\tau = 1$ if the the triple $\tau \in R$, and $x^{\mathcal{R}}_\tau = 0$ otherwise. For all $(s,t,p) \in \mathcal{C}$, the triple is routed by $\mathcal{R}$ so $x^{\mathcal{R}}_{(s,t,p)} + x^{\mathcal{R}}_{(*,t,p)} + x^{\mathcal{R}}_{(s,*,p)} \geq 1$. Now since R is a shortest list, there is at most one destination triple per destination t, so for all $t \in T$, $\sum_p x^{\mathcal{R}}_{(*,t,p)} \leq 1$. Similarly, there is at most one source triple per source s, so for all $s \in S$, $\sum_p x^{\mathcal{R}}_{(s,*,p)} \leq 1$. Therefore, $x^{\mathcal{R}}$ satisfies the constraints of (7). Moreover, by definition $|\mathcal{R}| = \sum x^{\mathcal{R}}_{(s,t,p)} + \sum x^{\mathcal{R}}_{(*,t,p)} + \sum x^{\mathcal{R}}_{(s,*,p)}$. Therefore $r^* \leq |\mathcal{R}| = \mathrm{rmin}(\mathcal{C})$.

Reciprocally, let us construct a routing list $\mathcal{R}^*$ with no global triples emulating $\mathcal{C}$ from x^*. Let $\mathcal{C}'$ be the set of communication triples (s,t,p) such that $x^*_{(s,t,p)} = 1$, let $\mathcal{R}'$ be any list over $\mathcal{C}'$, and set $\mathcal{C}'' = \mathcal{C} \setminus \mathcal{C}'$. Since $D_{\mathcal{C}}$ was acyclic, $D_{\mathcal{C}''}$ is also acyclic. Moreover. the set

Z of destination and source triples τ such that $x^*_\tau = 1$ corresponds to a star cover of $D_{\mathcal{C}''}$. Indeed, the first constrainst implies that for every arc uv of $D_{\mathcal{C}''}$, $u \in V^+$ or $v \in V^-$ because $x^*_{(s,t,p)} = 0$, and the second and third constraints imply that two stars have distinct centres. Hence, as in Lemma 39, there is a convenient routing list $\mathcal{R}''$ of size $|Z|$ emulating $\mathcal{C}''$. The concatenation of $\mathcal{R}'$ and $\mathcal{R}''$ is then a routing list with no global triples emulating $\mathcal{C}$. Hence $\text{rmin}(\mathcal{C}) \leq |\mathcal{R}'| + |\mathcal{R}''| = |\mathcal{C}'| + |\mathcal{C}''| = |\mathcal{C}|$. □

Lemma 49. *ILP* (7) *can be solved in polynomial time.*

Proof. Let $\mathbf{M}_D$ be the matrix associated to ILP *(7)*. Then $\mathbf{M}_D = [\mathbf{I}\ \mathbf{A}_D]$. Since $\mathbf{A}_D$ is totally unimodular, $\mathbf{M}_D$ is also totally unimodular. Thus, by Theorem 44, ILP (7) can be solved in polynomial time. □

Proof of Theorem 47. It follows directly from Proposition 48 and Lemma 49. □

7 Conclusion

We studied the complexity of several variants Routing List and of List Reduction. We provide results of NP-completeness, of Fixed-Parameter Tractability and approximation algorithms.

We leave several questions as open problems:

- We describe several kernels for our problems. Is it possible to find smaller kernels? In particular, we obtained a quadratic kernel for List Reduction parameterized by z.

  Problem 13. Does List Reduction parameterized by z admits a linear kernel?

- The Destination-based Heuristic is a 2-approximation for List Reduction. However, observe that the heuristic often returns a routing list that saves more than a half of $\text{sav}(\mathcal{C})$ triples. This leads to think that the approximation of 2 is not best possible.

  Problem 27. What is the best approximation ratio for List Reduction?

- We proved the NP-Completeness of Routing List by a reduction to Feedback Arc Set. Knowing that Feedback Arc Set is APX-complete,

  Problem 37. Is Routing List also APX-complete? For which approximation ratio?

References

[1] Sanjeev Arora, Alan Frieze, and Haim Kaplan. A new rounding procedure for the assignment problem with applications to dense graph arrangement problems. *Mathematical programming*, 92(1):1–36, 2002.

[2] Milind M Buddhikot, Subhash Suri, and Marcel Waldvogel. *Space decomposition techniques for fast layer-4 switching*. Springer, 2000.

[3] Rami Cohen, Liane Lewin-Eytan, Joseph Naor, and Danny Raz. On the effect of forwarding table size on SDN network utilization. In *2014 IEEE Conference on Computer Communikations, INFOCOM 2014, Toronto, Canada, April 27 - May 2, 2014*, pages 1734–1742, 2014.

[4] David Eppstein and S Muthukrishnan. Internet packet filter management and rectangle geometry. In *Proceedings of the twelfth annual ACM-SIAM symposium on Discrete algorithms*, pages 827–835. Society for Industrial and Applied Mathematics, 2001.

[5] Tibor Gallai. Maximum-minimum sätze über graphen. *Acta Mathematica Hungarica*, 9(3):395–434, 1958.

[6] Frédéric Giroire, Dorian Mazauric, Joanna Moulierac, and Brice Onfroy. Minimizing routing energy consumption: from theoretical to practical results. In *Green Computing and Communications (GreenCom), 2010 IEEE/ACM Int'l Conference on*, pages 252–259. IEEE, 2010.

[7] Frédéric Giroire, Joanna Moulierac, and Truong Khoa Phan. Optimizing Rule Placement in Software-Defined Networks for Energy-aware Routing. In *IEEE GLOBECOM*, Austin Texas, United States, December 2014. IEEE.

[8] Frédéric Giroire, Joanna Moulierac, Truong Khoa Phan, and Frédéric Roudaut. Minimization of network power consumption with redundancy elimination. In *NETWORKING 2012*, pages 247–258. Springer, 2012.

[9] Adiseshu Hari, Subhash Suri, and Guru Parulkar. Detecting and resolving packet filter conflicts. In *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 3, pages 1203–1212. IEEE, 2000.

[10] Alan J Hoffman and Joseph B Kruskal. Integral boundary points of convex polyhedra. In *50 Years of Integer Programming 1958-2008*, pages 49–76. Springer, 2010.

[11] Nanxi Kang, Zhenming Liu, Jennifer Rexford, and David Walker. Optimizing the "one big switch" abstraction in software-defined networks. In *Proceedings of the Ninth ACM Conference on Emerging Networking Experiments and Technologies*, CoNEXT '13, pages 13–24, New York, NY, USA, 2013. ACM.

[12] Y. Kanizo, D. Hay, and I. Keslassy. Palette: Distributing tables in software-defined networks. In *INFOCOM, 2013 Proceedings IEEE*, pages 545–549, April 2013.

[13] TV Lakshman and Dimitrios Stiliadis. High-speed policy-based packet forwarding using efficient multi-dimensional range matching. *ACM SIGCOMM Computer Communication Review*, 28(4):203–214, 1998.

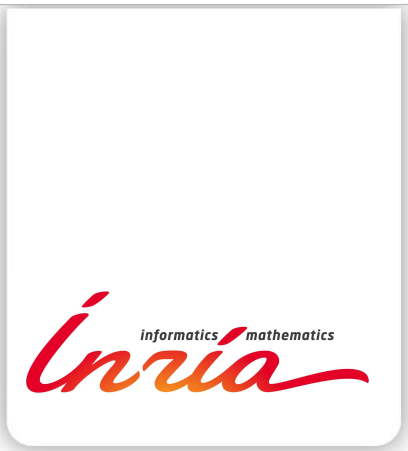
[14] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner. Openflow: Enabling innovation in campus networks. *SIGCOMM Comput. Commun. Rev.*, 38(2):69–74, March 2008.

[15] R. Narayanan, S. Kotha, G. Lin, A. Khan, S. Rizvi, W. Javed, H. Khan, and S.A. Khayam. Macroflows and microflows: Enabling rapid network innovation through a split sdn data plane. In *Software Defined Networking (EWSDN), 2012 European Workshop on*, pages 79–84, Oct 2012.

[16] Brent Stephens, Alan Cox, Wes Felter, Colin Dixon, and John Carter. Past: Scalable ethernet for data centers. In *Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies*, CoNEXT '12, pages 49–60, New York, NY, USA, 2012. ACM.

[17] Subhash Suri, Tuomas Sandholm, and Priyank Warkhede. Compressing two-dimensional routing tables. *Algorithmica*, 35(4):287–300, 2003.

[18] Anke Van Zuylen. Linear programming based approximation algorithms for feedback set problems in bipartite tournaments. In *Theory and Applications of Models of Computation*, pages 370–379. Springer, 2009.



RESEARCH CENTRE
SOPHIA ANTIPOLIS – MÉDITERRANÉE

2004 route des Lucioles - BP 93
06902 Sophia Antipolis Cedex

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-6399