Algorithms for network topology discovery using end-to-end measurements

Laurent Bobelin (1,2,3), Traian Muntean (2)

(1) British Telecom 200 rue Pierre Duhem BP 389
13799 Aix-en-Provence Cedex 3 France
laurent.bobelin@univmed.fr

(2) University of Marseilles, Parc Scientifique de Luminy, ESIL F-13288 Marseille Cedex France
traian.muntean@univmed.fr

(3) CPPM - Centre de Physique des Particules de Marseille 163 avenue de Luminy
- case 902 - 13288 Marseille Cedex 09 France

Abstract

Identifying and inferring performances of a network topology is a well known problem. Achieving this by using only end-to-end measurements at the application level is a method known as network tomography. When the topology produced reflects capacities of sets of links with respect to a metric, the topology is called a Metric-Induced Network Topology (MINT). Tomography producing MINT has been widely used in order to predict performances of communications between clients and server.

Nowadays grids connect up to thousands communicating resources that may interact in a partially or totally coordinated way. Consequently, applications running upon this kind of platform often involve massively concurrent bulk data transfers. This implies that the client/server model is no longer valid. In this paper, we introduce new algorithms that reconstruct a novel representation of the knowledge inferred from the network which is able to deal with multiple sources/multiple destinations transfers.

1 Introduction

Nowadays grid testbeds often aim to link together up to thousands of computing and data storage resources over the world. Connectivity is ensured using either the Internet, or high bandwidth, delay networks such as GEANT in Europe [2] or TeraGrid [4] in US.

Upon such a kind of testbed, applications usually deploy software and resources dedicated to bulk data transfer. For example, EGEE project [1] uses a notion of a hierarchy of tiers, as illustrated in figure 1. In such a hierarchy, each tier is a data storage center physically located in one of the project partner’s lab. Level of each tier reflects the contribution of the project partner owning that resource. Tier-0 is located close to the experiment place (for EGEE, at CERN). Tier-0 communicates to tier-1, tier-1s can communicate with every tier-1s and to a subset of tier-2s and tier-2s communicate to a subset of tier-2s and a subset of tier-3s. Tier-1 are national or institutional centers, tier-2 are located close to large computing centers, tier-3 are located in labs. In such a case, the data transfer paradigm is no more a client/server one: each host is a source, a destination, or both, and each source communicates to a subset of destinations.

Figure 1. Overview of a n-Tier organization

This logical organization is mapped into the physical existing network as illustrated in figure 2. The example of the physical topology here is GEANT. As we can see, this mapping can imply that logically separated links are physically the same. For example, links between Italian and French tier-1 and between French tier-1 and tier-0 are logically separated but have physically a common sub-path.

Therefore, it is mandatory to know capacity and topology of the underlying network in order to optimize communications between tiers. If not, some logically independent transfers may compete for the same physical network re-
source while optimal performances would require transfers not to be scheduled simultaneously. Unfortunately, most of the time, physical topology is unknown. Moreover, existing monitoring tools like NWS [17] or WREN [13] allow to model only basic interactions between transfers. In their model, either transfers occur between hosts belonging to the same group (called clique) and then share a common link, or not. If not, transfers are considered as not interfering with each other.

Most of the time, the topology discovery can be done using tools like traceroute [12]. The resulting topology is unlabeled. It is formed by matching IP address of network equipments belonging to the different observed paths. Moreover, these tools use information that can only be obtained if network administrators allow doing so. As a grid application runs on hosts owned by organizations applying different security policies, using such tools is most of the time not realistic. In order to infer a topology one must use only application level measurements. Such a method is known in the literature as network tomography [18].

Since a decade, network tomography has been widely studied. Different approaches have been used, depending both on the needs expressed and on targeted network (see [9] for a state of the art). Most of the time, topology is inferred using values of links for a given metric. This metric can be for example maximum achievable bandwidth or delay. Such a topology is an oriented graph where each edge is labeled with the set of physical objects it represents. In client/server case, this topology is a tree. The root is the server, the leaves are client and inner nodes are disjunction point of paths between the server and clients. Vertices are labeled with the capacity (in respect to the metric considered) of routers and wires belonging to the sub-path considered. Such inferred topologies are known as Metric Induced Network Topologies (MINT).

This kind of topology inference is an inverse problem. Most of the time, it is solved using statistical techniques that aim to estimate likelihood. Roughly speaking, it consists to collapse inferred points into one when capacities of the paths leading to those inferred points are similar. These methods have drawbacks. Most of all, it relies on the assertion that the resulting topology is a tree. But as mentioned in [6], a tree cannot characterize the network when multiple sources and multiple destinations are involved.

Solving an inverse problem mainly consist in three steps: 1) find an accurate model for solutions, which enable to pose the problem as a well-defined one, 2) Find a way to retrieve an initial set of observed data which enables reconstruction 3) Reconstruction of the solution, given this initial set of data.

This paper mainly focuses on the third point, i.e. reconstruction of the problem for a given model and a given initial data set. In this paper we introduce new algorithms dedicated to a specific model of the network, the Metric Induced Network Poset (MINP), described in an earlier paper [10].

The remainder of this paper is organized as follows. First, we give an overview of existing work dealing with topology discovery in section 2. We then define the terms and notations used in this paper in section 3. Then, we shortly introduce MINP in section 4 and useful properties in order to reconstruct MINP from end to end measurements in section 5. After giving a basic way to probe the network in order to retrieve an initial set of data in section 6, we give a basic algorithm in order to reconstruct it in section 7. We then refine this algorithms in order to minimize its cost in section 8. We finally conclude in section 9.

2 Related work

As stated before, the first formalization of necessary conditions that a metric must satisfy to allow a metric-induced topology reconstruction has been given in [5]. Nevertheless, those definitions are no longer valid when the problem shifts from client/server to a multiple sources, multiple destinations.

For the classical case of a single source communicating to a set of destination, the problem has been widely studied. Different approaches have been tested. Both passive [14] and active [5] measurements have been used. It has been applied to cases such as one source communicating to many destination or many sources communicating to a single destination. Reconstruction techniques are most of the time similar : they are based on statistical methods (see [9] for a state of the art). The main differences occur in the measurements procedure. Measurements are mainly realized using packet train techniques but can also be based for example on multicast trees [6].

Up to now, a few studies have focused on finding a topology for the multiple source/ multiple destination. In [15], authors use the existing MINT model to induce tree topolo-
gies. Then, they infer sub-paths common to two trees. And by this mean, they infer conjunction points between trees. The main drawback relies in the fact that "having a common sub-path" is not transitive. Indeed, if a path a has a common sub-path with a path b, and if b has a common sub-path with a path c, that does not mean that a has a common sub-path with c. Even if a has a common sub-path with c, it does not mean that there is a sub-path common to all paths a, b, and c. Therefore, conjunction point exists only between two trees. The method used is close to the one used in [6] where identification of common sub-path is done on edges belonging to multicast trees. Other authors [7] does such a matching between trees by using harsh assumptions about the network properties, such as routing symmetry and capacity symmetry.

In [11], authors formalize a problem close to our. The idea is to reconstruct a topology by detecting the sub-paths common to flows by using a metric related to bandwidth without labeling the edges. The notion of interference used there is close to the notion of having common detectable sub-path. Moreover, the metric used avoids any labeling.

Other authors rely on active but "stealth" measurements (i.e. without requiring the collaboration of destinations) in order to reconstruct unlabeled topologies [16]. They use Round Trip Time in order to infer common links between flows. Anyhow, their method cannot infer labeled topologies, and is thus useless in our case.

3 Notations

3.1 Vocabulary

A probe is the atomic action of injecting messages into the network in order to determine its properties. The complete process of injecting probes in order to discover the entire targeted network is the measurement procedure. Except when explicitly stated, we will assume that there is no cross traffic. Hereafter in this article, we will similarly assume that routing is consistent and stable. By the former, we suppose that routing function does not allow routing paths to join, fork, and join again. By the latter, we suppose that routing paths will not change during the whole probing process.

We consider the network as an oriented graph \( G = (V \cup S \cup R, E) \) where vertices \( V \) are network equipments such as routers, hub, etc., \( S \) the set of hosts which will behave like senders, \( R \) the set of hosts which will behave as receivers and \( E \) physical links between them (\( E \subset V \cup S \times V \cup R \)). We will note \( l_{ij} \) a directed edge from \( i \) to \( j \). A host that is both a source and a destination will be considered as two different hosts, one source and one destination.

Upon this graph, routing function defines a set of paths. If routing is consistent, there is a unique path between each source \( a \) and destination \( b \). Indeed if two paths exists between \( a \) and \( b \), that means that they have joined in \( a \), then fork, and join again in \( b \). \( p_{ab} \) is the path from \( a \in S \) to \( b \in R \). This path is an ordered sequence \( p_{ab} = \{l_{ai}, l_{ij}, l_{jk}, ..., l_{bk}\} \) of directed edges \( l_{ij} \in E \). We will use either link or edge in order to name such \( l_{ij} \). Each directed edge of any path starts from the destination of the edge preceding it (if such an edge exists). A sub-path of \( p_{ab} \) is a sub-sequence of this sequence that satisfies the path definition between a source \( a' \in S \cup V \) and a destination \( b' \in V \cup R \). This sub-path is contained by \( p_{ab} \). The length of a path is the number of directed edges in the sequence. The set of all paths defined by the routing function between each source \( s \in S \) and each destination \( r \in R \) will be noted \( P_{s2r} \). It is the set of end-to-end paths. We will call flow probes packets going through a path or sub-path.

A common sub-path to a set of paths \( P_s \) a sub-path contained by each element of \( P_s \). The common maximum sub-path of a set of paths \( P_s \) the longest common sub-path of \( P_s \). If consistency holds, the longest common maximum sub-path is unique for a given \( P_s \). This sub-path will be noted \( \hat{P} \). We will say that paths contained in \( P_s \) admit a common maximum sub-path. The set of common maximum sub-path admitted by at least one subset of \( P \) will be noted \( \text{Max}^P \).

3.2 Metric

A metric is a function whose initial domain is the set of flows and whose range is reals. As flows are defined over paths, the value obtained for a flow can label a path. We will note \( c_m \) the capacity of a path for the metric. For example, if the metric \( m \) is the delay, the capacity \( c_m \) of a path will be equal to the sum of the delay induced by each directed edge composing it.

A capacity of a path \( p \) will be detectable if there exists a set of paths containing \( p \) such that probing over those paths can exhibit capacity of \( p \). For example, if the metric is the throughput achievable by TCP flows on steady-state, then the capacity of a sub-path can be detected only if it is feasible to saturate this path. An undetectable capacity of a path can be for example a path inducing no delay for the delay metric, or a path with infinite capacity if the metric is the bandwidth.

A metric will be constant with respect to measurement if the capacity \( c_m \) does not depend on the paths followed by probes that detect it. For example, if the metric is the delay induced by a path, the capacity of a sub-path common to a set of path will be the same for each of these paths. The ratio of achievable bandwidth between two co-occurring TCP flows on a same sub-path is a non-constant metric. Indeed, two TCP concurrent flows will share bandwidth according to their respective round trip time. Therefore, two pairs of
TCP flows admitting the same common sub-path will not share the achievable bandwidth the same way, and will exhibit a different ratio. A metric will be said to be bounded if the capacity of a (sub)path is determined by the lowest (or highest) capacity of all the links composing it. Most of bandwidth-based metric are bounded.

4 Metric Induced Network Poset

This section is devoted to the definition of the model used to describe the network. For further details, see [10].

4.1 Definition

A metric induced network poset is a poset \( P^m = (X, \prec) \) formed from \( \text{Max}^{P_{2e}} \).

- \( X \) is defined by the relation \( \forall i \in \text{Max}^{P_{2e}}, i \text{ detectable} \) for the metric \( m \) \( \iff \) \( i \in X \),
- \( \prec \) is defined by the relation \( \forall i, j \in X, i \subset j \iff j \prec i \),
- Every element of \( p \in X \) is labeled by its capacity \( c^p_m \).

Roughly speaking, this model does not anymore represent the topology, but detectable common sub-paths and the set of longer (sub)paths in which they are included.

4.2 Drawing

We will use a Hasse diagram graphical rendering of partially ordered sets. We display the poset via the cover relation (the transitive reflexive reduction of the partial order) of the partially ordered set with an implied upward orientation. In a Hasse Diagram a point is drawn for each element of the poset, and arcs are drawn between these points according to the following two rules:

- If \( x \prec y \) in the poset, then the point corresponding to \( x \) appears lower in the drawing than the point corresponding to \( y \).
- The line segment between the points corresponding to any two elements \( x \) and \( y \) of the poset is included in the drawing iff \( x \) covers \( y \) or \( y \) covers \( x \).

We will display \( P_{2e} \) elements at the bottom of the drawing.

In figures below, white circles depicts sources and black ones destinations. Squares are physical router or hubs that are conjunction/disjunction points for paths of \( P_{2e} \). Hereafter parameters \( c^e_m, c^{e'}_m, c^f_m, c^{f'}_m, c^g_m, c^h_m \) and \( c^i_m \) depicts capacities of edges. Dotted lines depicts various routes of flows between sources and destinations. We consider that the logical organization of transfers only allows to communicate from \( a \) to \( a' \), \( b \) to \( b' \), and \( c \) to \( c' \). As we focus mainly on achievable bandwidth, we will consider hereafter this metric.

4.3 Examples

Figure 3 (1) represents two 3 sources 3 destinations topologies containing only 3 paths. We consider that all links have higher capacity than those of \( e \) and \( f \). These topologies will have similar impact on communications performances if \( c_{\text{Bandwidth}}^e = c_{\text{Bandwidth}}^f \) and \( c_{\text{Bandwidth}}^f = c_{\text{Bandwidth}}^f \). This is explained by the fact that path \( a \rightarrow a' \) and path \( b \rightarrow b' \) will share in both case a narrow link of available bandwidth \( c_{\text{Bandwidth}}^f \) and that the common narrow link to all paths will have a capacity of \( c_{\text{Bandwidth}}^f \).

![Figure 3. Simple topologies and their representation as MINP](image)

The two possible MINP depicted on the right corresponds to two different relations between the values of \( c_{\text{Bandwidth}}^e \) and \( c_{\text{Bandwidth}}^f \). The MINP on the left depicts the case where \( c_{\text{Bandwidth}}^e < c_{\text{Bandwidth}}^f \). The upper node represents the sub-path \( f \), the middle one the sub-path formed by the links \( e \) and \( f \), and finally the lower nodes represents, from left to right paths \( a \rightarrow a' \), \( b \rightarrow b' \) and \( c \rightarrow c' \). The MINP on the right depicts the case where \( c_{\text{Bandwidth}}^e \geq c_{\text{Bandwidth}}^f \). In such a case, the sub-path \( e \) is not detectable as no subset of \( P_{2e} \) can saturate this link while probing simultaneously. The difference between the two possible cases in real life can be caused by cross-traffic, when dealing with available bandwidth. If we consider that the wire capacity of path \( e \) is constant, the former stands for a case when cross traffic make the available bandwidth decrease on \( e \) so that capacity of \( e \) appears detectable. This is important, as it means that a MINP representation of a network depends not only on network topology and paths included in \( P_{2e} \), but also on cross-traffic.
4.4 k-detectability

Detectability as we have defined in section 3 is a property of a sub-path $p$. $p$ is detectable if there is at least a subset $P$ of $P_{e2e}$ such that a probe applied to $P$ can exhibit the capacity of $p$.

Hereafter we need a more restrictive definition of detectability in order to enlighten possibilities of reconstruction of a MINP representation from end-to-end measurements. We enhance this notion by defining $k$-detectability.

**k-detectability** A sub-path $p$ is $k$-detectable if there is at least one subset of $P' \subseteq P$, $|P'| \leq k$ such that a probe applied to $P$ can exhibit the capacity of $p$.

This is important, as practical algorithms cannot rely on a probe involving all paths in $P_{e2e}$. Given this property, one can define a $k$-MINP as a MINP representing only $k$-detectable sub-paths. In [10], we have demonstrated that the problem of reconstructing either a MINP or a $k$-MINP is well defined.

5 MINP properties

In order to reconstruct a (k-)MINP, the algorithms given in section 7 rely on the following properties.

5.1 Covering rule

Given a set of paths $P$ such that $|P_{\text{max}}| \neq 0$ no probe can exhibit two different $|P_{\text{max}}|$. This property trivially holds only if the metric is constant and paths are stable.

5.2 Grouping rule for bounded metrics

Let two set of paths $P = P_{\text{core}} \cup \{p_a\} \cup P_{\text{pivot}}$, $P' = P_{\text{core}} \cup \{p_b\} \cup P_{\text{pivot}}$. If $P_{\text{core}} \geq 0$, suppose that each elements of $P$ all share a common maximum sub-path, as well as elements of $P'$ and $P_{\text{pivot}}$. If $\{p_a, p_b\} \cup P_{\text{pivot}}$ share a common maximum sub-path, then $P'' = P_{\text{core}} \cup \{p_a\} \cup \{p_b\} \cup P_{\text{pivot}}$ share an unique common maximum sub-path. Moreover if the metric is bounded, then one can reorder the different common maximum sub-paths of each of those sets.

**Proof** If $|P_{\text{core}}| = 0$, it is trivially true. Let suppose $|P_{\text{core}}| = 1$, i.e. $P_{\text{core}} = \{p_d\}$. For an easier writing, let’s note $p_c$ the common maximum sub-path of $P_{\text{pivot}}$. The property above mean that sets $\{p_a, p_b, p_c\}$, $\{p_a, p_c, p_d\}$ and $\{p_b, p_c, p_d\}$ all have a common maximum sub-path. It means that $\{p_a \cap p_c\} \cap p_d$ and $\{p_a \cap p_c\} \cap p_b$ are not null. As sub-paths between paths are unique, the common maximum sub-path $\{p_a, p_c\}$ have two common sub-paths, one with $p_d$ and one with $p_b$. Either those sub-paths $\{p_a \cap p_c\} \cap p_b$ and $\{p_a \cap p_c\} \cap p_d$ are disjoints, or they share a common sub-path. As $\{p_b, p_c, p_d\}$ admit a non-empty common sub-path, common sub-paths of $\{p_a \cap p_c\}$ with $p_b$ and with $p_d$ admit a non-empty common sub-path. If not, it would mean that there are multiple common sub-path for $\{p_a, p_c\}$ and $\{p_c, p_d\}$. So, it exists a commun sub-path for the set $\{p_a, p_b, p_d\} \cup P_{\text{pivot}}$. When $|P_{\text{core}}| > 1$, we only have to replace paths of $P_{\text{core}}$ with the common maximum sub-paths of paths in $P_{\text{core}}$. The upper reasoning still holds. So there is an unique common maximum sub-path to the set $\{p_a, p_b\} \cup P_{\text{core}} \cup P_{\text{pivot}}$.

Please note that if $P_{\text{core}} = P_{\text{core}} \setminus \{p_x\}$, we can use the same reasoning by using $p_x$ in place of $p_{\text{pivot}}$ and use the same reasoning. By this way, we obtain a way to construct recursively the knowledge of the existence of a common maximum sub-path for an arbitrary set of paths $P$.

Let suppose again $|P_{\text{core}}| = 1$, i.e. $P_{\text{core}} = \{p_d\}$. As there is a sub-path $p_z$ for the set $\{p_a, p_b, p_c, p_d\}$, each triplet of path has an upper bound (for bandwidth for example for which the associated relation is $\leq$) which is by definition the capacity of this common maximum sub-path, as it is contained in each common sub-path of each triplet. If the capacity of one (or many) of those sub-paths $p_z$ is lesser than $\pm s$ (s sensibility parameter) the capacity of this sub-path, then it means that $p_c$ contains a link of a lesser capacity than the common maximum sub-path of the 4 paths together. Path $p_z$ contains then the sub-path $p_z$, so one can reorder those two sub-path into a tree where $p_z$ is the root. If the capacity of $p_z$ is similar to the capacity of one of the triplet’s common maximum sub-path, then one can confuse $p_z$ and this path.

Similarly to the upper reasoning, when $|P_{\text{core}}| > 1$ one can replace paths of $P_{\text{core}}$ by the common maximum sub-path of $P_{\text{core}}$.

6 Measurement procedure

This section deals about feasible measurements procedure in order to detect elements of $X$. We present here a “naive” procedure that can effectively state if a number of flows share detectable common sub-paths. As our aim is to infer knowledge about bandwidth, this measurement procedure is designed in order to measure this metric. It is based on bulk data transfers. We used it to validate our measurement procedure and implement our algorithm given in section 7 upon large platform using simGrid [3] simulator. As it does not rely on packet-level network properties, and as simGrid bandwidth share model is INV-RTT-BOUNDED [8], this approach was compatible with our initial hypothesis. As simGrid experiments are less costly in terms of time than packet-level simulations, we used that one for large platform. This measurement procedure is "naive", be-
cause it is based on steady state properties of TCP flows. As steady-state might not be achievable in reasonable time for large platforms, it is unrealistic to use it for real life experiments. For real-life test cases, we use a procedure based on packet train dispersion techniques. This procedure has been implemented and tested and works well. As many issues must be considered for such packet-based techniques, we will not develop this approach here, but in a later paper.

6.1 Procedure based on bulk data transfers

The main principle of the technique relies on the fact that with INV-RTT-BOUNDED sharing model, the sum of bandwidth allocated to each flow sharing a common bottleneck is equal to the capacity of this bottleneck.

Suppose we have 3 paths \( p_a, p_b \) and \( p_c \in P_{\text{probes}} \), and that flows \( f_{p_a}^{\{p_a\}}, f_{p_b}^{\{p_b\}} \) et \( f_{p_c}^{\{p_c\}} \) are injected along those paths. One can establish bandwidth for each \( f_{p_j}^{P_j} \) for all sets \( P_i \subseteq P_{\text{probes}} \) et \( p_j \in P_{\text{probes}} \). It means that one establish achievable bandwidth for steady state for each flow independently, then competing two by two, and then all together.

As the capacity is equal to the sum of achievable bandwidth for all the flows, then the sum of the bandwidth for a given \( P_i \) is lesser than the sum of the achieved bandwidth of all flows when are measured separately. It means that :

\[
\forall P.t.q.|P| = 2 \sum_{i=O}^{P} \text{bw}(f_{P_i}^{P}) > \sum_{i=O}^{P} \text{bw}(f_{P_i}) \iff P_{\text{maximum}} \text{detectable and } \sum_{i=O}^{P} \text{bw}(f_{P_i}) = \sum_{i=O}^{P} \text{bw}(f_{P_i}^{P}) \tag{1}
\]

So one can determine if two flows share a common detectable sub-path, and what its capacity is. For the triplet \( \{p_a, p_b, p_c\} \), it is detectable only if bandwidth obtained by the 3 flows injected simultaneously is lesser than all possible combinations of interference between flows :

\[
\sum_{i=O}^{P} \text{bw}(f_{P_i}^{P}) < \sum_{p_j \in P'} \text{bw}(f_{P_j}) + \sum_{p_j \in P' \backslash P}^{P'} \text{bw}(f_{P_j}^{P'} \forall P' \subset P) \iff P_{\text{maximum}} \text{ detectable and } \sum_{i=O}^{P} \text{bw}(f_{P_i}^{P}) = \sum_{i=O}^{P} \text{bw}(f_{P_i}) \tag{2}
\]

By establishing each values and by using those properties, one can establish if it exists a common detectable sub-path for each \( P' \subseteq P \) and reconstruct a partial MINP for \( P \).

7 A reconstruction algorithm

In this section we present an algorithm in order to reconstruct a MINP from an initial set of data obtained using for example the measurement procedure described in section 6.

7.1 Algorithm description

The algorithm relies on the MINP properties given before, namely grouping and covering rules. The basic idea is to reconstruct a complete network representation using measurement done on subset of \( P_{2e} \). Once all the initial data are collected, partial views of the network are reconstructed. The algorithm then applies iteratively the grouping and covering rules in order to reconstruct the whole representation.

The algorithm contains 3 key steps. The first step is to sort the partial MINP contained in \( M_k \). We sort them by using the value of the maximum common sub-path of all flows contained in \( P_i \) where \( P_i \) is the set of paths represented by \( x_i = (X, \leq) \in M_k \). If there is a maximum common detectable sub-path common to all flow, then \( x_i \) is sorted by its capacity increasingly ; if not, it is considered to be greater than any of the \( x_i \) having a maximum common detectable sub-path.

We do so because of the following property :

**Lemma** If the metric \( m \) is bounded, then the lowest element \( x_{\text{min}} \) of the sorted list of \( M_k \) is completely defined. No other \( k \)-detectable sub-path common to the elements of \( x_{\text{min}} \) can exist.

**Proof** If the metric is bounded, no sub-path can exist between elements of \( x_{\text{min}} \), as it should have a lower capacity than the supremum of \( x_{\text{min}} \).

The second step is a loop upon the set of partial MINP \( M_k \) obtained via \( k \)-probes for each subset of \( P_{2e} \) of cardinality \( k \) described formally by algorithm 7.1. Please note that the grouping property holds only when \( k \geq 3 \). That means that in practice the best way to use it is to choose \( k = 3 \). This loop executes a search upon the set of partial MINP in order to know if either the grouping or the covering rule is applicable. If so, the third step of the algorithm consists in applying the correct rule, either the covering one (algorithm 3) or the grouping one (algorithm 2).

Please note that it is not necessary to keep the nodes of the partial MINP \( x = (X, \leq) \) added by the covering or the grouping rule. It is because subsets of \( X \) must belong to \( MINTotal \) in order to find common detectable sub-paths necessary to the application of the grouping rule for bounded k-MINP.
Algorithm 1 Main algorithm

Require: Sensibility parameter \( s \), list \( \text{sort}_{M_k}[0] \) of \( M_k \) sorted as described above.

Ensure: reconstruction of a k-MINP for the whole network.

\[ \text{MINP}_{\text{total}} \leftarrow \text{sort}_{M_k}[0] \]
\[ i \leftarrow 1 \]
while \( i \leq \text{length(\text{sort}_{M_k})} \) do
   \[ \text{currentMINP} \leftarrow \text{sort}_{M_k}[i] \]
   if \( \text{currentMINP} \) admits a detectable sub-path for at least 3 of the paths then
      Search into \( M_k \) of common detectable sub-paths in order to apply the grouping rule for bounded metric (algorithm 2). It means find if all triplet of the paths covered by nodes covered by the common detectables sub-paths in \( \text{MINP}_{\text{Total}} \) and \( \text{currentMINP} \) admit a common detectable sub-path.
   end if
   if Grouping rule not applicable then
      fusion of \( \text{currentMINP} \) and \( \text{MINP}_{\text{Total}} \) respecting the unique coverage rule (algorithm 3).
   end if
end while
return \( \text{MINP}_{\text{Total}} \).

Algorithm 2 Fusion of MINP using grouping rules

Require: \( \text{MINP}_{\text{Total}},x,y \) common detectable sub-paths of \( \text{MINP}_{\text{Total}} \), \( \text{currentMINP} \) and it common detectable sub-path \( z \), sensibility parameter \( s \)

Ensure: MINP fusion

Sort detectable sub-paths according to the relation \( \leq \) associated with the metric.
Create a new node \( w \) in \( \text{MINP}_{\text{Total}} \) which capacity is equal to the greater capacity (according to \( \leq \) ) in the list of common detectable sub-paths.
for all \( \text{path} \in \{x, y, z\} \) do
   if \( c_{\text{path}} - c_{w} = 0 \pm s \) then
      Confuse \( \text{path} \) and \( w \).
   else
      Add a new node \( v \) son of \( w \) covering the same (sub)paths that \( \text{path} \) in \( \text{MINP}_{\text{Total}} \).
   end if
end for
return \( \text{MINP}_{\text{Total}} \).

Algorithm 3 Fusion using the covering rule

Require: \( \text{currentMINP}, \text{MINP}_{\text{total}}, \) sensibility parameter \( s \).

Ensure: Fusion of the 2 MINP respecting the covering rule.

if \( \text{MINP}_{\text{Total}} \supseteq \text{currentMINP} \) then
   Do not do anything (relations in \( \text{MINP}_{\text{partiel}} \) are already represented).
else
   Add nodes not already into \( \text{MINP}_{\text{Total}} \) according to their coverage.
end if
Return \( \text{MINP}_{\text{Total}} \).

7.2 Performances

Complexity of this algorithm is polynomial, but this is not a key point. Unfortunately, the most expensive part of the whole MINP reconstruction is the establishment of the initial set of partial MINPs by probing the network, which is really costly in comparison to the algorithm execution. The algorithm in this section necessitate all triplets of flows to be tested, and so do \( \frac{1}{6} n(n-2)(n-1) \) probes. This seems to us an unrealistic approach. So, we need to further optimize this initial algorithm by reducing the needed number of probes.

8 Faster algorithms

As stated before, the number of measurements is by far the most costly part of the MINP reconstruction. Unfortunately, the previous algorithm given in section 7 requires that all the triplet of flows have been probed, leading to probe all \( o(n^3) \) possibilities.

An easy way to reduce the number of triplet that have to be probed is twofold. The first step is to transform the algorithm in order to make it an online algorithm, i.e. an algorithm that will depend of already probed partial MINP, in order to test only relevant subset of flows. In order to do this, we have to no more consider k-MINP but 1-MINP. The second step is to cut the search phase of every triplet covered by the nodes candidates to merging by using grouping rule.

8.1 An online algorithm

If we consider 1-MINP, one can measure the achievable bandwidth for each path. After having an estimate of it for every \( p \in P_{e2e} \), one can sort them increasingly. Doing so, one can rely on the lemma given in section 7.1, as if a supremum exists in any of the partial 1-MINP containing the path having the lower capacity, no other supremum can exist between this supremum and the path. It allows us to sort the
paths instead of the partial MINP, and consequently to reconstruct partial MINP only when needed, instead of having to test the whole set of triplet subsets of $P \times 2^e$.

### 8.2 Cutting the search phase

Suppose that, at a given moment of the reconstruction, a set $P$ of paths of cardinality $n$ admitting a common detectable sub-path. Suppose that a path $p_e$ admit a common detectable sub-path with $p_a, p_b \in P$. By definition, either a detectable sub-path exists for all paths $P \cup \{p_e\}$, or it exists a common detectable sub-path for paths $P' \subset P$ and $p_e$, and a set of paths $P'' = P \setminus P'$ for which it does not exist a common detectable sub-path for $p_e$ and $p_a, p_b \in P''$.

If we consider that are no specific properties existing for the underlying network, sets $P'$ and $P''$ are a partition of the set $P$. As this partition has no specific properties, one can infer that the probability that a path belong to either $P'$ or $P''$ is $\frac{1}{2}$. As a false-true detection is done when at least one of the two paths of $P$ randomly chosen from the $n - 2r$ paths that we still have to test at the $r$-th measure confirming that the grouping rule can apply belong to $P''$, one can estimate the probability $P(A)$ of having not detected a false true at the $r$-th measure as follow:

$$P(A) = \prod_{i=0}^{r-1} \frac{\left(\frac{1}{2}\right)^{n-2i}}{2^{i-2i}} \approx \left(\frac{1}{4}\right)^r \quad (3)$$

So, we can easily cut the search phase in order to have confidence in a result for a certain degree $d$, by choosing $r$ so that $\left(\frac{1}{2}\right)^r \leq 1 - d$. It means that the number of probes drops drastically, as each use of the grouping rules only implies $r$ subset to be probed.

### 9 Conclusion and discussion

The problem of modeling network performances of an unknown platform upon which actors transfer bulk data in a many-to-many paradigm and on a coordinated scheme has arose from the offspring of network intensive grid applications. So, inferring, modeling and representing knowledge about performances of a network is a challenging new problem.

In this paper, we presented simple algorithms in order to reconstruct a representation of interactions between paths into a adequate model. It is an important shift in methodology for the grid because the way we represent the network topology when dealing with classic distributed applications was no more valid. By providing these algorithms, we have proved that it is at least feasible to use this model.

Our work is to build a prototype of a tool that can reconstruct and depict such a kind of knowledge, in order to bring more precise model to optimization processes.

### References