Optimized P2P Streaming for Wireless Distributed Networks[☆]

Leonardo Maccari, Nicolò Facchi, Luca Baldesi, Renato Lo Cigno[1]

Abstract

The future Internet will support pervasive applications and communications models that require end-nodes cooperation, such as fog computing and machine-to-machine communications. Among the many applications, also video streaming can be provided with a cooperative and peer-to-peer approach. Cooperative distribution requires building a distribution overlay on top of the physical underlay topology, and this work proposes an optimized, cross-layer approach to build this overlay minimizing the impact on the underlay. We design an optimal strategy, which is proven to be NP-complete, and thus not solvable with a distributed, lightweight protocol. The optimal strategy is relaxed exploiting the knowledge on the betweenness centrality of the underlay nodes, obtaining two easily implementable solutions applicable to any link-state protocol for distributed wireless mesh network. We then introduce heuristic improvements which further optimizes the performance in real network scenarios. Extensive simulation results support the theoretical findings using three different network topologies. They show that the relaxed implementations are reasonably close to the optimal solution, and provide vast gains compared to the traditional random overlay topology that a peer-to-peer application would build.

Keywords: P2P video streaming, Mesh networks, ad-hoc networks

1. Introduction

Fog computing, machine-to-machine communications, Direct WiFi and LTE, the same architecture of the Internet of Things, are all indicating that the future trend of the Internet evolution is toward decentralized systems, where peer-to-peer communications will play a mayor role, if not for else, to relieve the backbones and data-centers that are the bottleneck of today centralized

[☆]This work was partially financed by the European Commission, H2020-ICT-2015 Programme, Grant Number 688768 'netCommons' (Network Infrastructure as Commons).

[1]Department of Information Engineering and Computer Science (DISI), University of Trento, Italy

architectures. Decentralization for sustainability is probably one of the key factors that will enable pervasive computing and communications.

Streamlined communications, video but not only, will remain among the largest bandwidth guzzlers, as they are today, and their efficient support is of paramount importance for the scalability of the entire system. Currently, the majority of the Video Service Providers (VSPs) deliver video streams using unicast traffic and leveraging centralized platforms supported by world-wide Content Delivery Networks (CDNs). A CDN is a very costly resource that replicates the content as close to the final user as possible. CDNs are approximately co-located with Internet exchange points and replicate videos based on the estimated popularity of the video itself. Users access replicas cached in CDNs, so that the total load on the VSP infrastructure is limited. This approach does not fit well the advent of pervasive communications, in which the video streams will need to be served with minimal delay and on a personal basis. Each user may have potentially tens of video flows and/or real-time data streams generated by the smart devices in his house, friends, co-workers, relatives, which will need to be accessed by different locations. Each stream will be useful to a small number of people, so the convenience of CDNs and cloud-based caching will lower. Pervasive computing instead will be based on a distributed mesh of wireless (or wired) devices, which will form an *underlay* network over which live video will be produced, locally served and possibly globally distributed. An already relevant example of this future trend is represented by Community Networks (CN from now on). CNs are large mesh networks (primarily made of wireless links) that are flourishing in many different scenarios, from the developing countries where there is no other connectivity means, to the urban areas of Western cities where they compete with other network providers. The steep decrease of the prices of outdoor wireless equipment makes it possible to build wireless mesh networks with links that can achieve tens of Mbit/s and support CNs made of hundreds of nodes. Today, CNs made of thousands of devices that cover large cities or even regions exists, like the Guifi network in Spain or the Freifunk network in Germany[2]. These networks already connect in a peer-to-peer (P2P) way thousands of people, and potentially, tomorrow, hundreds of sub-networks made of smart devices. These networks are a key element of future pervasive applications, especially in developing countries or underserved regions.

In this trend, there is room for a renewed interest in P2P systems. Despite a very large interest from the academia, in the last 15 years P2P applications could not compete with centralized, cloud-based applications. The key factor hampering P2P development, specially for video distribution, has been the difficulty to realize P2P overlays optimized from the point of view of the Internet Service Providers (ISPs), mostly due to lack of information on the physical network. In mesh networks, the underlay is normally known, since the routing protocols exports it to each node (as long as a link-state routing protocol is used), which removes one of the technical barriers that blocked the deployment

[2]See www.guifi.net, www.frefunk.org

of P2P video streaming on the Internet.

In this paper, we propose a cross-layer optimization scheme to perform *live* video streaming (i.e., with a strict deadline on the arrival delay) in mesh networks [1]. The optimization minimizes the impact of the streaming overlay on the underlay network exploiting information on the topology and the routing of the underlay. The optimization is based on the concept of *centrality*, which is also at the base of successful algorithms as Google PageRank [2]. Taking into account the centrality of peers in the underlay graph, the optimized overlay topology greatly improves the efficiency of the video distribution and maintains high performance.

More specifically, our key contributions are as follows:

- We formalize an optimization problem that, given the underlay of the P2P network and the peers in the overlay, finds the overlay topology whose cost on the underlay is minimum. We define the cost of the overlay as a combined metric composed by the load and the fairness imposed by the overlay on the underlay;
- We prove that the optimal strategy is NP-complete by reducing it to a quadratic knapsack problem;
- We propose two techniques for relaxing the optimal strategy which relies on the concept of betweenness centrality of the nodes of the underlay. The relaxed strategies are applicable to any wireless mesh network, as long as they use a link-state routing protocol;
- We further improve the performance of the relaxation strategies by introducing a *neighborhood pruning* heuristic which exploits characteristics often found in real network scenarios;
- We evaluate the proposed relaxation strategies and pruning heuristics through extensive simulations that compute the best overlay, according to each proposed technique, on synthetic network topologies. Simulation results show that the proposed relaxations and heuristics are reasonably close to the optimal solution and largely outperform random overlay building strategies.

This paper builds on a previous work in which we first formalized the optimization problem and, introduced the relaxation strategy and evaluated its results [3]. This paper extends these results with further optimizations to make the proposal even more efficient in overlays with a realistic topology, i.e. comprising many leaf-node in the network graph.

The rest of this paper is organized as follows: in Section 2 we formalize the optimization problem. The related work on this area is summarized in Section 3. Section 4 introduces the intersection graph notation we use for defining the overlay. In Section 5 we describe the strategies used for relaxing the optimization problem, and the heuristic used for further improving the relaxation strategies is described in Section 6. We analyze the performance of the proposed strategies and heuristics in Section 7. Finally, Section 8 concludes the paper.

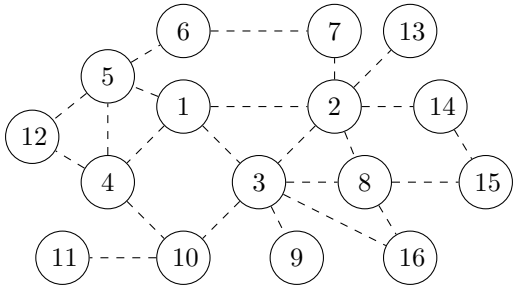


Figure 1: Example of wireless mesh underlay graph. Hosts are numbered with an arbitrary ordering. Dashed lines represent the wireless links.

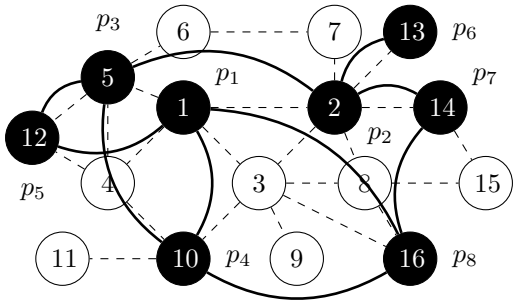


Figure 2: Possible overlay graph (black vertexes and edges) over the underlay graph of Fig. 1.

2. Motivation and Problem Statement

We consider a communication system where the cooperative distribution exploits a logical topology called the *overlay* built on top of a meshed routing network called the *underlay*. Nodes in the *overlay*, that from now on we call peers, do have access to information concerning the *underlay*, including details on the topology and quality of its links. This is true for a wireless mesh network using link-state routing protocols, in which every wireless router needs to know the whole topology of the network to perform routing. For instance, standard implementations as the OLSRd daemon implementing the Optimized Link State Routing (OLSR) protocol [4] export the topology with a simple API. The optimization we propose regards the choices of the edges in the *overlay* so that the impact on the *underlay* is minimized and evenly distributed.

We model the *underlay* with an undirected graph $U(H, L)$ with vertexes $h \in H$ called hosts or nodes, and edges $l \in L$ called links. The size of H is between a few tens up to a thousand hosts, which corresponds to the realistic size of a CN [5]. Fig. 1 shows the graph representation of a sample underlay with 16 hosts.

The peers form an *overlay* that is also modelled as an undirected graph

Table 1: Summary of the main symbols used through the paper and their meaning.

Peers, Hosts, Links and Virtual Links sets	P, H, L, and E
Overlay and underlay graphs	$U(H,L)$, $O(P,E)$
Fairness of $O(P,E)$ over $U(H,L)$	$\mathcal{F}$
Network load of $O(P,E)$ over $U(H,L)$	$\mathcal{L}$
kth undirected overlay edge in $P \times P$	e_k
Cross-layer overlay edge descriptor of e_k	$\bar{e}_k$
ith overlay peer	p_i
Degree of p_i in the overlay	d_i
Set of the overlay edges in $O(P,E)$ linking p_i	S'_i
Family of all $S'_i,\ i = 1, \ldots, \lvert P \rvert$	F'
$O(P,E)$ as an intersection graph	$\Omega(F')$
Target function on $\Omega(F')$ addressing $\mathcal{L}$ and $\mathcal{F}$	O_c
Binary variable representing of whether $e_k \in E$	z_k
Estimation of link usage in $O(P,E)$	b

$O(P,E)$ with vertexes $p \in P$ called peers, and edges $e \in E$ called virtual or logical links. Each peer resides in one host only, and it has access to information pertaining to U, including the association between peers and hosts. Fig. 2 depicts a possible overlay graph on the underlay of Fig. 1.

The goal is to find a viable (meaning that can be implemented as a distributed system with limited signaling overhead and acceptable computational overhead) methodology to select virtual links between peers to build $O(P,E)$ given $U(H,L)$ and P so that the load imposed by the video streaming on the *underlay* links is minimized, and links are loaded as fairly as possible. O is dynamically created and maintained because both O and U can change frequently, so the modification of O must be fast and efficient.

2.1. Formal Problem Definition

Tab. 1 reports the main notation we use in the paper. Given an edge e connecting $p_i, p_j \in P$, we call $\mathrm{D}(e)$ the Dijkstra function returning an (ordered) set of links in the underlay that form the shortest path from the host where p_i resides and the host where p_j resides. For example in Fig. 2 we have:

$$e = (p_1, p_4) \mapsto \mathrm{D}(e) = \{(1,3),(3,10)\}$$

where (i,j) is the link e connecting hosts i and j. A generic weight $w(l)$ is assigned to any link in the *underlay*, so that the load $\mathcal{L}$ imposed by O on U is

$$\mathcal{L} = \sum_{e \in E} \sum_{l \in \mathrm{D}(e)} \frac{w(l)}{s(e)} \tag{1}$$

The $s(e)$ denominator represents a scaling factor for the edge, we detail it and use it in rest of the paper to express the fact that certain edges can convey

more traffic than others. This representation perfectly fits the routing protocols that use the ETX metric [6], ETX is the expected average number of frames sent on the link to correctly deliver one frame, and it is used by OLSR and other protocols.

Every link l is loaded by a number of virtual links. To measure fairness, we use Jain's fairness index on the distribution of the number of logical links insisting on every l. Let $\mathcal{H}(l)$ be the number of logical links loading l:

$$\mathcal{H}(l) = |\{e \in E \ : l \in \mathrm{D}(e)\}| \tag{2}$$

where $|\cdot|$ is the size of a set. The Jain's fairness is defined as

$$\mathcal{F} = \frac{\left(\sum_{l \in L} \mathcal{H}(l)\right)^2}{|L| \sum_{l \in L} \mathcal{H}(l)^2} \tag{3}$$

Jain's fairness is maximal if $\mathcal{F} = 1$ and minimal when $\mathcal{F} = \frac{1}{|L|}$, but we do not expect that maximal fairness can be reached, as in general there are links in the *underlay* that do not support any edge in the *overlay*.

The contribution of this paper is twofold:

- First we derive a formal framework wherein it is possible to define an optimization problem that allows finding the topology of $O(P, E)$ given $U(H, L)$ and P that minimizes a metric composed of $\mathcal{L}$ and $\mathcal{F}$. The problem is NP-complete and we'll show that it can be reduced to a quadratic knapsack problem [7];
- Second we propose two relaxations of the optimization in decreasing complexity order and we show, with numerical solutions and with an implementation in a real P2P streaming platform, that the two relaxations are close to the global optimum and that they vastly outperform the traditional P2P *overlay* building based on selecting uniformly neighboring peers to build an Erdös-Rényi graph.

The paper focuses on building a mesh overlay and not on how video chunks are distributed on it. On mesh topologies this latter problem can be tackled with several different strategies. Even if this is not the focus of the paper, Sec. 3 briefly refers some relevant works on this topic, justifying our choice for the distribution strategy we use in Sec. 7 and Sec. 7.1. The assumption we start from, which becomes a constraint of the optimization problem, is that each peer should receive exactly one copy of every chunk. In our previous work [3] we also assumed that each peer contributes to the dissemination serving chunks to other peers proportionally to its degree in O. This was motivated by the effort to keep the number of neighbors per peer constant (with a quasi-regular topology), so that the overall load was fairly distributed. In this paper we introduce a new model, in which we assume that every peer will generate a constant amount of traffic, evenly distributed among its neighbors. This way we support non quasi-regular topologies that match better with the underlay topologies.

In our previous paper we used two evaluation strategies, first we used graph analysis with Matlab and the Python Networkx graph library, then we validated the results implementing the strategies on a real P2P video streaming platform, PeerStreamer [8]. We showed that the first methodology approximates extremely well the real implementation. In this work we focus on further insights on the theoretical model and its validation and thus we rely on the first instrument and we leave the implementation as future work.

3. Related Works

Cooperative video streaming (including P2P) is an established research area. We focus on unstructured and mesh-based approaches, in which, differently from structured P2P networks [9], there is no specific structure (like a tree) in the topology. This approach has been shown to be particularly robust even in networks with churn (i.e., peers leaving and joining the swarm), and the overlay topology design is of paramount importance [10].

We do not consider here papers that perform streaming optimization on mesh networks requiring modifications to the lower layers (they cannot be applied to existing CNs) or that are not tailored for live video streaming (e.g., using large chunks that imply several seconds of buffering delay). We also do not consider techniques (e.g., like cloud-assisted or SDN based), where the role of the peers is, in one way or another, not fundamental, and we assume that security [11] and collusion [12] issue need not be solved by the application itself. The following discussion is focused on two parts: *topology management*, which is directly related to our contribution, and *chunk/information scheduling*, which justify the choice of the chunk selection strategy in Sec. 7 and Sec. 7.1.

3.1. Topology Management

As we already mentioned, *overlay* optimization on the Internet is not feasible due to lack of information on the *underlay* details; however several efforts have been done to adapt and improve the *overlay* topology to some measured *underlay* characteristics.

The first approach to mention is the use of "network coordinates" as a means to compute distances between hosts in a certain space. Several algorithms were proposed [13, 14, 15], that are designed to work in the heterogeneous environment of Internet. In all of them the goal is clearly to find a method to infer details on the *underlay* (the Internet), a problem that we do not have, as we take advantage of the available information on the network topology provided by routing protocols in Wireless Mesh Networks (WMNs). We believe that our solution can be adapted, albeit not straightforwardly, to situations where the *underlay* is not known but approximated with network coordinates.

A second line of research has been concerned with the adaptation of the *overlay* based on bandwidth [16] or delay (normally the round trip time between peers) [17] measures, but also on a mix of the two [18]. The solutions found in these works are, once again, tailored to the Internet, where delays

can be large (CNs spans a few tens of hundreds of km at most), and bandwidth asymmetry at the edges impose hard limits to the capacity of peers to contribute to dissemination.

Extremely interesting and promising for topology management is the adoption of centrality metrics as means to better understand the topology characteristics of a network graph as it emerges from the routing protocol. Centrality metrics in graphs have been used in social science since the 70s to identify the most influential elements in social networks. Quite surprisingly, they were not applied to multi-hop networks up to recent times [2]. Centrality metrics can be used to enhance network monitoring and routing [19], as well as the resilience of routing algorithms to networks failures [20] intrusion detection and firewalling [21, 22], and topology control [23]. There are several metrics based on different centrality "concepts". In this paper we use the betweenness centrality (see [22] for a definition tailored to our problem) to relax the optimization problem as it is strictly related to shortest path routing.

3.2. Chunk Scheduling

In a mesh-based *overlay* the problem of chunk scheduling is the selection of the neighbors to send/receive information to/from while contextually choosing the right information (chunk) to send/retrieve. This problem has been extensively studied [24, 25, 26], and in some specific contexts with restrictive assumptions, the existence of an optimal scheduling strategy has been proven [27].

Those works show that an efficient and robust chunk scheduling technique that works well in most environments consists in selecting a neighbor with a random (possibly weighted) strategy, and push the most recent chunk that is still missing at the receiving peer (Latest Useful Chunk).

4. Overlay Model

The links in U are bidirectional and assumed quasi-symmetric: the most common routing protocols take care of excluding unidirectional or highly asymmetric links. Thus both O and U are undirected, and the maximum number of edges they can have is $m_O = \frac{|P|^2-|P|}{2}$ and $m_U = \frac{|H|^2-|H|}{2}$ respectively.

Let $r \in 1 \dots m_U$ be an arbitrary ordering on the links; r is also a mapping from the two hosts h_i and h_j that are the endpoints of the link: $r(h_i, h_j)$. Similarly we define an ordering $k \in 1 \dots m_O$ on the edges of O, and k is a mapping $k(p_i, p_j)$.

Every link l_r is represented by a binary array of size m_U with the rth element set to one and all other elements set to zero (we use the bar sign to refer to the array representation of a link):

$$\bar{l}_r = (0, \dots, 0, 1, 0, \dots, 0)$$

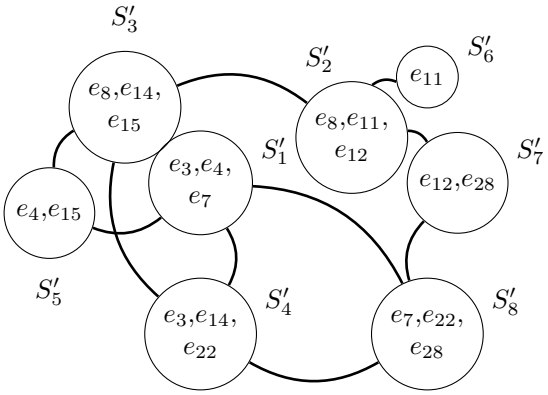


Figure 3: Example of overlay intersection graph. Elements e_k are the cross-layer overlay edge descriptors between the nodes.

Equivalently each virtual link e_k is represented by the sum of the link arrays in the corresponding shortest path:

$$\bar{e}_k = \sum_{l_r \in \mathrm{D}(e_k)} \bar{l}_r$$

We call $\bar{e}_k$ the cross-layer overlay edge descriptor, since it bonds the overlay to the underlay. Link weights $w(l)$ in U can be easily taken into account: let $W \in \mathbb{R}^{m_U \times m_U}$ be a diagonal matrix such that $W_{r,r} = w(l_r)$, then $\bar{e}_k W$ is the weighted representation of the overlay edge.

4.1. Overlay re-definition as an intersection graph

Given all $\bar{e}_k$ for a set of peers P, we can take advantage from a transformation into the intersection graph space (see [28] for the complete definitions and properties of intersection graphs) in order to formulate the optimization problem of overlay construction.

Let S be a set and $F = \{S_1, \ldots, S_p\}$ a nonempty family of distinct nonempty subsets of S whose union is S. The intersection graph of F is denoted $\Omega(F)$, with S_i and S_j adjacent whenever $i \neq j$ and $S_i \cap S_j \neq \emptyset$. It is easily shown that if $O(P, E)$ is a full mesh then it is isomorphic with the intersection graph space $\Omega(F)$ where

$$S_i = \{\bar{e}_{k(p_i,p_j)}, \forall p_j \in P\} \ \forall p_i \in P; \quad S = \cup_{i=1}^{|P|} S_i \tag{4}$$

and each S_i is the set of all the possible virtual links built on shortest paths from peer p_i to all other peers. As a consequence, given an underlay $U(H, L)$ and a set of peers P, any overlay $O(P, E)$ over $U(H, L)$ can be defined as the intersection graph $\Omega(F')$ with $F' = \{S'_1, ..., S'_{|P|}\}$ where $S'_i \subseteq S_i \ \forall S_i \in F$. If each peer chooses only a subset $S'_i \subseteq S_i$ and activates only a subset of possible

edges, then the resulting overlay is isomorphic with some $\Omega(F')$. The overlay depicted in Fig. 2 is isomorphic with the intersection graph shown in Fig. 3: every S'_i in Fig. 3 corresponds to a peer p_i in Fig. 2.

4.2. Performance Measures

We can now redefine in terms of intersection graphs also the performance metrics expressed by Eq. (1) and Eq. (3) in Sec. 2:

$$\mathcal{L} = O_I(\Omega(F')) = \vec{1} \cdot L(F'); \quad L(F') = \sum_{\bar{e}_k \in S'} \bar{e}_k \frac{W}{s(e_k)} \tag{5}$$

where $L(F')$ is the array that associates the traffic potentially produced by the overlay to each link in the underlay, $\vec{1}$ is the array of size m_U made of all ones, and $\cdot$ is the dot product. Given the assumption on the traffic generated by every peer that we made in the end of Sec. 2.1 we can set $s(e_k) = (d_i d_j)/(d_i + d_j)$, where d_i, d_j are the degrees of the peers that constitute the endpoints of e_k. This formulation of $s(e_k)$ takes into consideration the distribution of the unitary load generated by a peer on its neighbors. If all the peers have the same degree $s(e_k)$ is constant and can be omitted from the formulation. Else, it takes into account that a peer with many neighbors will load each of its edges less than a peer with few neighbors.

Eq. (5) redefines Eq. (1) through operations done in the intersection graph space. Similarly we re-define the Jain's fairness index as

$$\mathcal{F} = O_f(\Omega(F')) = \frac{(\sum_{k=1}^{m_U} L(F')_i)^2}{m_U(\sum_{k=1}^{m_U} L(F')_i^2)} \tag{6}$$

5. Overlay Optimization

For the sake of simplicity, but without loss of generality, we take as weighting matrix W the identity matrix. In this section we consider regular overlays with constant degree, so we omit $s(e_k)$. We will re-introduce it in Sec. 6 where we show that regular overlays are necessarily suboptimal over certain realistic underlays. Let's say we want to build an overlay O determined by a choice of $F' = \{S'_1, ..., S'_{|P|}\}$ that minimizes the load on the underlying edges and guarantees a fairness as close as possible to 1. We have to choose the sets $S'_i \subseteq S_i$ with $S' = \cup_{i=1}^{|P|} S'_i$ such that both O_I and O_f are minimal, which is a multi-objective combinatorial optimization problem. The problem allows the definition of a combined metric O_c that expresses the *cost* of the overlay. Since each array $\bar{e}_k \in S'$ corresponds to a set of links the cost of the overlay is defined as:

$$O_c(\Omega(F')) = \left|\left| \sum_{k=1}^{|S'|} \bar{e}_k \right|\right|_2$$

The creation of an efficient overlay $\Omega(F')$ can now be formulated as a minimization problem as follows. Select F' in order to minimize the expression:

$$\arg\min_{z} \left|\left| \sum_{k=1}^{|S|} z_k \bar{e}_k \right|\right|_2 \tag{7}$$

where

$$z_k = \begin{cases} 1 & \text{if } \bar{e}_k \in S' \\ 0 & \text{otherwise} \end{cases}$$

In order to avoid a trivial solution we impose on each peer a minimum node degree $d > \log_2(|P|)$, which also guarantees that the resulting overlay is connected with high probability[3].

This problem can be rephrased as: find the overlay graph $\Omega(F')$ with minimum degree d defined by $F' = \{S'_1, ..., S'_{|P|}\}$ so that the norm of $L(F')$ is minimized subject to the following constraint

$$\left(\sum_{\bar{e}_k \in S_i} z_k \right) \geq d; \quad \forall i = 1...|E| \tag{8}$$

In Eq. (8) we did a small abuse of notation to improve the readability: the sum spans all the edges $e_k \in S_i$, but it is indeed a sum over $k(p_i, p_j)$ to correctly identify the indication function z_k. We will use this notation also in several other equations.

If, instead of the norm, in Eq. (7) we use the sum of the elements, this would simply minimize O_I. The norm instead prefers solutions that are close to the minimum O_I and, among two potential solutions with the same O_I, it prefers the one in which the weights of $L(F')$ are more fairly distributed.

This problem is a zero-one quadratic programming problem [29], similar to a quadratic knapsack problem [7]. In this kind of problems one wants to minimize the value of an expression $c^T x + x^T Q x$ where $x = \{0, 1\}^n$ is an array of binary variables, $c \in R^n$ and Q is a symmetric matrix of size $n \times n$. The minimization is subject to a constraint of the kind $h^T x + x^T G x > g$ where h is an array of size n, G a symmetric matrix of size $n \times n$ and g some real value. If we call $\bar{A}$ the matrix made of columns corresponding to the arrays $\bar{e}_k$ and z the array made of z_k elements then:

$$\arg\min_{z} \left|\left| \sum_{k=1}^{|S|} z_k \bar{e}_k \right|\right|_2 = \arg\min_{z} z^T \bar{A}^T \bar{A} z$$

Our problem is thus a zero-one quadratic problem with $Q = \bar{A}^T \bar{A}$, G and c made of all zeros, $h = \vec{1}$. This family of problems is known to be NP-hard, but there are algorithms in literature that make them tractable up to a certain size

[3]Using simple P2P techniques the probability actually converges to 1 [18].

using branch-and-bound techniques. Still, when $|S|$ grows beyond a few hundreds the problem can not be solved on commodity hardware. $|S|$ corresponds to the number of possible edges in the overlay m_O, so it scales quadratically with the number of peers, which quickly makes the problem intractable.

5.1. Betweenness Centrality-based Relaxation

We need to find a relaxation in which each p_j can solve a portion of the problem, making some assumptions on the behaviour of the other peers. This corresponds to a scenario in which every peer is aware of the other peers, independently selects its own neighbors, and it communicates them its choice. This is the way P2P streaming protocols based on peer sampling typically work (including the already mentioned PeerStreamer).

Let us first separate the contribution to the overall cost of the edges chosen by p_j and all the other peers in Eq. (7)

$$\left|\left|\sum_{k=1}^{|S|} z_k \bar{e}_k\right|\right|_2 = \frac{1}{2}\left|\left|\sum_{S_i \in F, S_i \neq S_j} \sum_{\bar{e}_k \in S_i} z_k \bar{e}_k + \sum_{\bar{e}_k \in S_j} z_k \bar{e}_k\right|\right|_2 \tag{9}$$

The value $\frac{1}{2}$ comes from the observation that when we separately count each link, every link is counted twice in the sum. Let's call b_j the vector representing the choices of the peers in $P \setminus \{p_j\}$:

$$\left|\left|\sum_{k=1}^{|S|} z_k \bar{e}_k\right|\right|_2 = \frac{1}{2}\left|\left|b_j + \sum_{\bar{e}_k \in S_j} z_k \bar{e}_k\right|\right|_2 \tag{10}$$

and we can say that:

$$\arg\min_z \left|\left|\sum_{k=1}^{|S|} z_k \bar{e}_k\right|\right|_2 = \arg\min_z \left|\left|b_j + \sum_{\bar{e}_k \in S_j} z_k \bar{e}_k\right|\right|_2 \tag{11}$$

Our goal is now to find a relaxation of the problem in which every peer chooses its own neighborhood making some assumptions about b_j, which represents the choice that the other peers $p_i \neq p_j$ do. We need to find a reasonable approximation $\bar{b} \simeq b_j$ that node p_j can use in (11).

Let us now introduce a notion that helps us in this task. In graph theory, the notion of *betweenness centrality* is a property of the edges (or of the nodes) of a graph defined as the fraction of the total number of shortest paths that passes through that edge (or node). It is a metric used to identify the edges (or nodes) that are more involved in multi-hop interactions between the vertexes of a graph, so for some applications they can be considered more important than the others.

We call $b = \sum_{\bar{e}_k \in S} \bar{e}_k$ the summation of all the cross-layer overlay edge descriptors of the complete overlay. Recall that $\bar{e}_k$ corresponds to a shortest path in U between two hosts on which a peer resides, thus, each element of b

corresponds to a link in U and expresses the number of shortest paths in the set S that insist on that link.

Consider the limit case in which every host in the underlay contains a peer ($|P| = |H|$) and let b^* be the value of b normalized to the total number of shortest paths:

$$b^* = b\frac{2}{|P|^2-|P|}$$

b^* is exactly the vector corresponding to the betweenness centrality of each link in L.

If $|H| > |P|$, b^* is an approximation of the real array of centralities. This is a known fact that is used to approximate centrality in large networks: if the number of nodes is too large to compute all the shortest paths, centrality can be estimated using a subset of the paths chosen from a random set of nodes [30]. A key fact is that the convergence to a solution close to the real one is pretty fast in power-law graphs, that are extremely frequent in real communication networks, and also in some large CNs [31]. Thus, even if p_j ignores b_j, a reasonable assumption is that whatever the choice of each other peer is, the elements of b_j (that represent the sum of all choices) have a shape similar to the centrality expressed by the normalized value b^*. This, on power-law underlays is true even for values of $|P|$ one order of magnitude smaller than $|H|$.

Given that we impose each peer to have (at least) d neighbors, the number of edges in E will be approximately $\frac{d|P|}{2}$ and finally the best approximation for b_j turns to be

$$\bar{b} = b^*(|P|-1)\frac{d}{2} = b\frac{d(|P|-1)}{|P|^2-|P|} = b\frac{d}{|P|} = \frac{d\sum_{k=1}^{|S|}\bar{e}_k}{|P|} \tag{12}$$

The complexity of Eq. (12) is polynomial with P [30], that allows the computations of its solution for overlays of hundreds of peers using commodity hardware. Moreover in communication networks, that are sparse graph, betweenness can be computed quickly using heuristics [32], which explains why we introduced this formulation based on centrality. Thus, replacing b_j with $\bar{b}$ in (11) each peer p_j resolves the following optimization problem:

$$\arg\min_z \left|\left|\bar{b} + \sum_{\bar{e}_k \in S_j} z_k \bar{e}_k\right|\right|_2 \tag{13}$$

$$\text{conditioned to: } \sum_{\bar{e}_k \in S_j} z_k \geq d \tag{14}$$

The formulation of (13) is another zero-one quadratic minimization problem, but the dimension of the problem is now bounded by $|P| < |S|$ (the maximum number of neighbors for p_j), and can be effectively solved up to hundreds of peers. In the rest of the paper we will use the branch-and-bound solver given by the YALMLIP library [33] which solves the problem (13) for a network of 100 nodes in few seconds.

If still the dimension of the problem or the available hardware do not allow the solution of the optimization, we can apply a greedy search algorithm, ranking each possible $\bar{e}_k$ for its weight and choosing the ones that minimize the sum. This corresponds to relax (13) to:

$$\arg\min_{z} \sum_{\bar{e}_k \in S_j} z_k \left|\left|\bar{b} + \bar{e}_k\right|\right|_2 \tag{15}$$

$$\text{conditioned to: } \sum_{\bar{e}_k \in S_j} z_k \geq d \tag{16}$$

which of course captures only a part of the original problem but greatly simplifies the computation. Our results show that the solutions generated by (7), (13) and (15) in the case of realistic network topologies are reasonably close one to each other.

It is worth nothing that if we set $\bar{b} = \vec{0}$ in Eq. (15), then the l-2 norm yields the same order of the l-1 norm. In practice each p_j chooses the neighbors that are closer in terms of hops in the underlay. The $\bar{b}$ terms instead introduce a bias in the choice towards the neighbors connected through links that are less overloaded and introduce a higher fairness. In the comparison we include also a strategy in which $\bar{b} = \vec{0}$ because this strategy can be used also in absence of full information from the underlay topology, since the distance from another peer can be measured with probing tool (like the `traceroute` application) or can be inferred by the time-to-live field in IP packets. Results show that the performance of this simple ranking function is sensibly lower compared to the proposed strategies.

Tab. 2 summarizes the different optimization strategies and labels them with the names we use.

Name	Symbol	Formula
Global Optimization	G_o	$\arg\min_z \parallel \sum_{k=1}^{\lvert S\rvert} z_k \bar{e}_k \parallel_2$
Local Optimization	L_o (L_{op})	$\arg\min_z \parallel \bar{b} + \sum_{k=1}^{\lvert S_j\rvert} z_k \bar{e}_k \parallel_2$
Local Equalized Ranking	E_r (E_{rp})	$\arg\min_z \sum_{k=1}^{\lvert S_j\rvert} z_k \parallel \bar{b} + \bar{e}_k \parallel_2$
Local Ranking	L_r (L_{rp})	$\arg\min_z \sum_{k=1}^{\lvert S_j\rvert} z_k \parallel \bar{e}_k \parallel_2$

Table 2: A summary of the optimization functions. Symbols between parentheses identify the optimization strategies that use the *neighborhood pruning* operation discussed in Sec. 6

6. Neighborhood Pruning

So far we considered only a sort of "blind" optimization, that can be largely improved in some specific, but not unrealistic, situations. Consider for instance

the case of a peer p_j that is placed on a leaf node of the underlay topology, and assume that its only underlay neighbor also contains a peer p_i. Does it make sense for p_j to choose more than one neighbor in the overlay? Under a performance point of view it makes no sense, all the traffic that p_j receives will pass through p_i. Under a robustness point of view it still makes sense, because p_i may suddenly leave the overlay and p_j would be logically disconnected. Yet it is intuitive to understand that it is convenient to reduce the neighborhood size of p_i, compared with another peer that resides in the center of the topology.

In general we can state that, regardless of the underlay topology, when p_j selects its neighbors, if there exists a peer p_y that lies on the shortest path between p_j and another p_i, then p_y should be preferred to p_i. In general, any node p_j should prefer as neighbors the peers that are the first encountered along any shortest path route, because otherwise the data packets will "jump" some peer that are in the path to their destination, which is a waste of resources. This situation is not frequent if underlays are random graphs and if the number of peers is small compared to the number of underlay nodes, but it can be frequent, or even dominant, in realistic topologies with many leaf nodes and when the number of peers becomes comparable with the number of nodes in the underlay.

Based on the consideration above, we can re-define the objective of building an optimal overlay limiting the neighborhood selection to peers first encountered on shortest path routes. We call this operation (of limiting the selection) *neighborhood pruning* because the search for a neighbor along a branch is stopped immediately at the first peer encountered, and thus the branch is pruned.

The pseudocode of the steps executed by a generic peer p_j for pruning the neighborhood are summarized in Algorithm 1 and detailed next.

1: $P^{tmp} = \{p_i | p_i \in P, i \neq j\}$, $P^j = \emptyset$
2: **repeat**
3: $\quad p_y \leftarrow \min_{p_i \in P^{tmp}} |\mathrm{D}(\bar{e}_{k(p_j,p_i)})|$
4: $\quad P^j = P^j \cup \{p_y\}$, $P^{tmp} = P^{tmp} \setminus \{p_y\}$
5: $\quad P^{tmp} = P^{tmp} \setminus \{p_i | p_i \in P^{tmp}, p_y \in l, l \in \mathrm{D}(\bar{e}_{k(p_j,p_i)})\}$
6: **until** $P^{tmp} \neq \emptyset$
7: $S_j = \{\bar{e}_{k(p_j,p_y)} | p_y \in P^j\}$

Algorithm 1: Neighbood pruning algorithm for peer p_j

In the initialization step (line 1) two sets are created: P^{tmp} is initialized to $P \setminus \{p_j\}$ and is used to hold the available peers; instead, P^j, initialized as an empty set, is used for holding the possible candidate neighbors of p_j. When a peer p_y is selected as a candidate neighbor and inserted into P^j, all the other peers p_i still available and which include p_y in their shortest path from p_j are removed from P^{tmp} and not considered as possible candidate neighbors anymore. This is what happens in the main loop of the algorithm (lines 2–6): at the beginning of every iteration the peer p_y, which is the closest to p_j among all the peers still available in P^{tmp} (line 3), is inserted in P^j and removed from

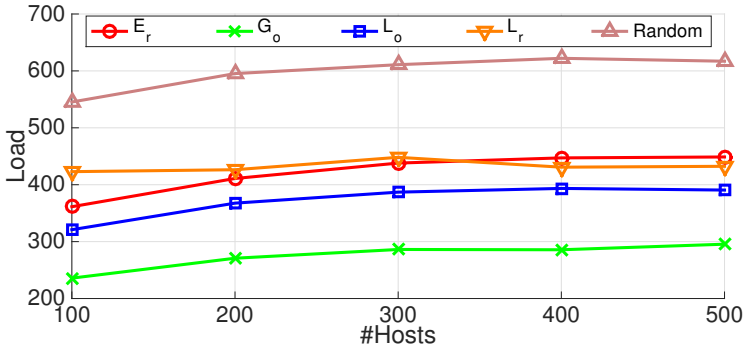
P^{tmp} (line 4). Finally, any peer p_i which includes p_y in its shortest path from p_j is removed from P^{tmp} (line 5). The loop stops when there are no more peers available ($P^{tmp} = \emptyset$). After the loop, the algorithm builds the set S_j used by the optimization strategies, which includes only the virtual links $\bar{e}_k$ that connect p_j to the peers p_y selected as candiate neighbors (line 7).

The computational complexity of the neighborhood pruning algorithm is linear in $|P|$ if we assume that the shortest paths computed by the Dijkstra functions are already available and that p_j has enough memory to maintain a data structure for easily connecting every peer p_i to the shortest paths that include p_i. If instead only the shortest paths computed by Dijkstra are available, the pruning algorithm is $O(|P|D)$, where D is the diameter of the network. In any case, the neighborhood pruning algorithm can be easily integrated into the Dijkstra functions which could be modified for returning only the shortest paths originating from p_j and terminating to the peers in S^j. Finally, note that when $|S^j| \leq d$ all the peers in P^j are automatically selected as neigbors by p_j without the need of executing any optimization. This means that when the number of peers becomes comparable with the number of nodes in the underlay the computational cost of the optimization strategy becomes almost negligible for most of the peers.

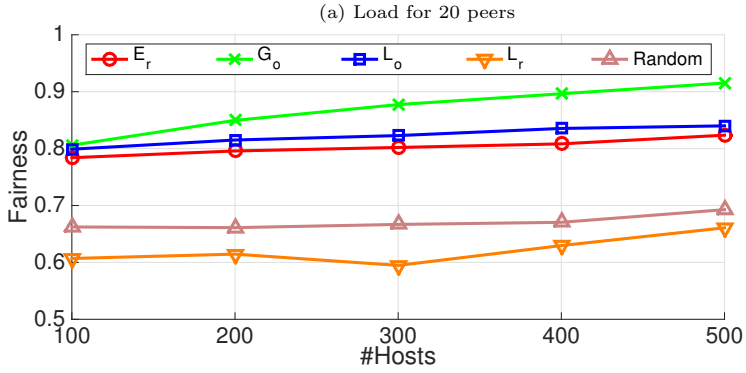
The *neighborhood pruning* operation can be easily applied to all the local optimization strategies discussed in Sec. 5 with the difference that the scaling factor $s(e_k)$ introduced in Eq. (5) cannot be ignored, because every peer now potentially has a different degree. In particular, for computing the final performance metrics of the overlay, we set $s(e_k) = (d_i d_j)/(d_i + d_j)$, where d_i, d_j, computed with the knowledge of the local solutions of all the peers, are the real degrees of the endpoint peers of e_k. Instead, to compute the solution of the local optimization strategies, for each peer p_j, we set $s(e_k) = d_j$, where $d_j = \min(d, |S^j|)$. In this way the computational complexity is reduced because p_j does not have to compute the pruned neighborhood of all the possible candidate neighbors in S^j.

7. Results Evaluation

We evaluated the proposed strategies implementing them in a simulator that computes the best overlay according to each strategy on synthetic network topologies. Simulations have been performed using the Networkx library, a powerful library for the generation and analysis of graphs realized in the Python language. Given an underlay topology we have implemented the proposed strategies and for each one we compared the measures of load and fairness of the generated overlay graph. The global and the local optimization problems are solved with the YAMLIP library, while the others have been implemented directly in Python. We used three different kinds of underlay graphs, the well known Erdős-Rényi (ER) and the Barabási-Albert (BA) model with preferential attachment and a model from Cerdá-Alabern (CE) [31] derived from the analysis of a number of real mesh networks. The CE algorithm uses a preferential attachment algorithm to create a core of interconnected hosts, and then



(a) Load for 20 peers



(b) Fairness for 20 peers

Figure 4: Load and fairness for an ER underlay from 100 to 500 hosts and 20 peers

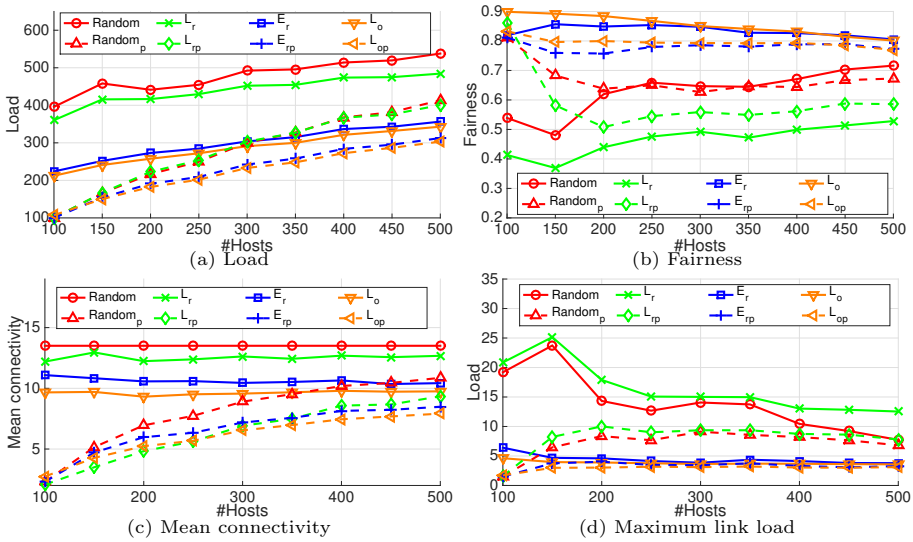
adds leaf hosts using a Gamma distribution. Finally we take advantage from real-life WCN topologies, taken from the Ninux and the FFWien[4] networks [34] respectively made of 131 and 236 nodes.

7.1. Results

Fig. 4 reports the comparison of all the described strategies in a small (20 peers) scenario, increasing the size of an ER underlay. With 20 peers we are able to solve all the optimization problems, so this is a good benchmark to outline the differences between each strategy. The optimized results vastly outperform the random strategy, which is not surprising since they use available information on the underlay. What is most significant is that even the strategies that are less costly to compute, namely L_o and E_r achieve results that are close to the optimal strategy G_o especially in terms of fairness.

This means that even if the quadratic optimization remains NP, in the analysed graphs the number of available disjoint paths between two peers is low and the space of the solutions of the optimization problem is small enough for all

[4]See `http://ninux.org` and `http://www.funkfeuer.at/`



(a) Load (b) Fairness (c) Mean connectivity (d) Maximum link load

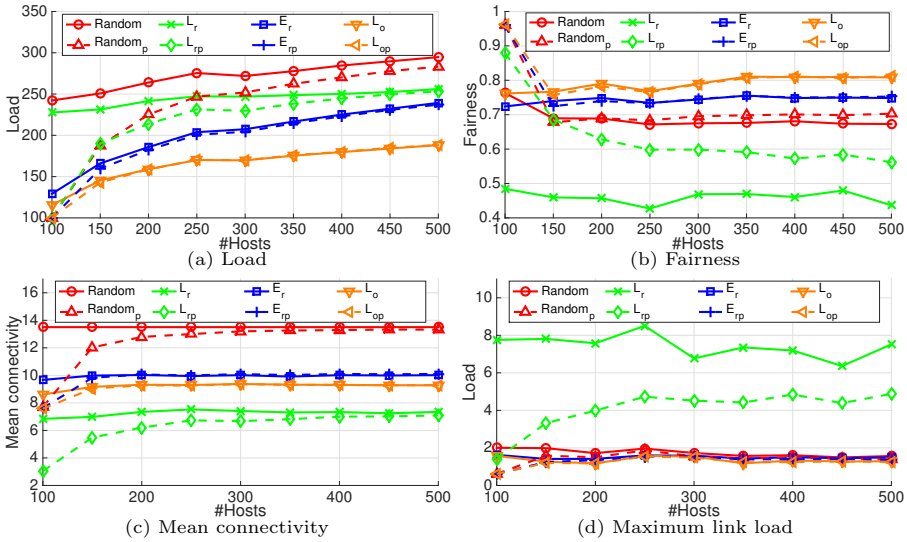
Figure 5: CE underlay, 100 peers, from 100 to 500 hosts

the optimization strategies to be very close. More results on this analysis can be found in our previous publication [1].

We performed experiments with overlay size up to 100 peers, since there is no qualitative difference in the results we report only the results for 100 peers. With that size we are not able to use the G_o strategy, so the comparison is done only with the remaining strategies and their "pruned" version.

Fig. 5 reports the results on a 100 peers overlay computed on CE underlays and shows two main results. The first is that the L_o and E_r strategies are very close and achieve a substantial improvement compared to the random strategy, which confirms the effectiveness of our approach. The L_r strategy produces results that are closer to the random strategy. This is again an interesting result because it shows that the naive neighbor choice based only on distance (L_r) produces a higher load than one can have if some long edges are added. Intuitively, choosing the peers that are the closest ones in the underlay seems the most reasonable solution. Instead, the shortest path between two peers computed on the underlay can be shorter than the sum of the shortest paths that pass through some other peers. The betweenness-based strategies try to avoid central peers which in some cases leads to the introduction of longer edges. These long edges may be network-wise more efficient than choosing neighbors only among the closest peers. The second important result is that every approach is substantially improved by the "pruning" strategy in terms of load on the underlay. This confirms that pruning neighbors is effective in excluding potential neighbors that are inherently inefficient.

If we look at the fairness, results are similar. Fig. 5b shows that L_o produces the highest fairness followed by E_r. They both fall in the $[0.8, 0.9]$ range, which is



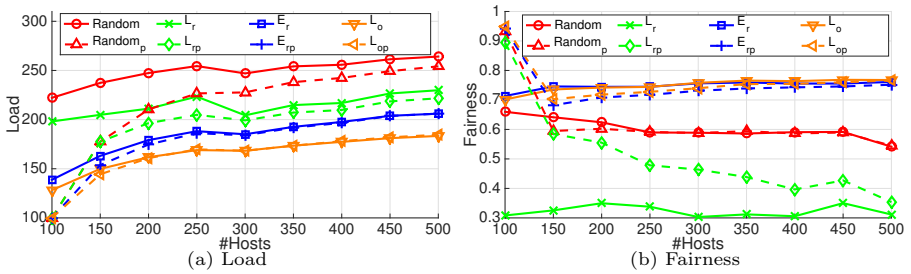
(a) Load (b) Fairness (c) Mean connectivity (d) Maximum link load

Figure 6: ER underlay, 100 peers, from 100 to 500 hosts

extremely high. Random placement performs better than L_r. This confirms that the choice of the closest peer is not a good strategy because it stresses the peers (and the corresponding underlay hosts and links) that are more central. The pruned strategies lightly decrease the fairness due to a similar effect: pruning removes from S_j the peers that are further away, thus it concentrates the choice on the closest peers. In this case the negative effect is very small. To further investigate this issue Fig. 5d shows the maximum load on any underlay link averaged on all the runs. The maximum load is reduced by pruning, so the load distribution, even if slightly less fair is more sustainable. Note also that, for an increasing number of hosts in the underlay, the maximum link load tends to keep a constant value or even decrease. This behavior is expected, in fact, when the number of hosts increases, the load of the underlay is distributed over a bigger number of links, thus reducing the maximum link load.

Finally in Fig. 5c we report the mean connectivity of the overlay computed by each strategy. The non-pruned strategies maintain a certain connectivity independently of the growth of the underlay. The pruned strategies instead increase the average connectivity with the growth of the underlay. To understand this effect, consider the point with 100 peers and 100 hosts. In this case the overlay corresponds to the underlay. The pruned strategies converge to almost the same average, because each peer can do nothing better than choosing as neighbors in the overlay the physical neighbors in the underlay. As the underlay grows the pruning algorithm leaves to each peer more neighbor candidates and thus, the average degree increases.

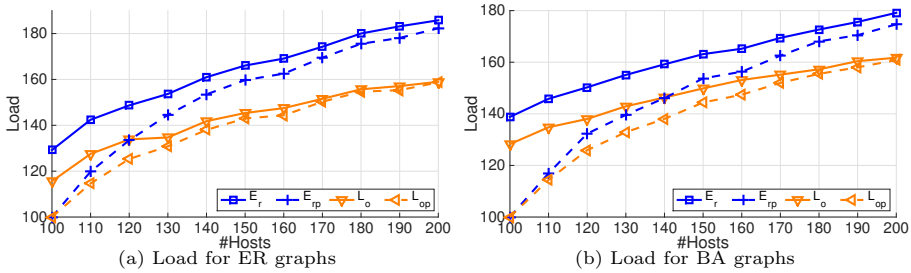
Fig. 6 and Fig. 7 report the same set of results computed on ER and BA topologies. The comparison among the non-pruned strategies confirms the same



(a) Load (b) Fairness

(c) Mean connectivity (d) Maximum link load

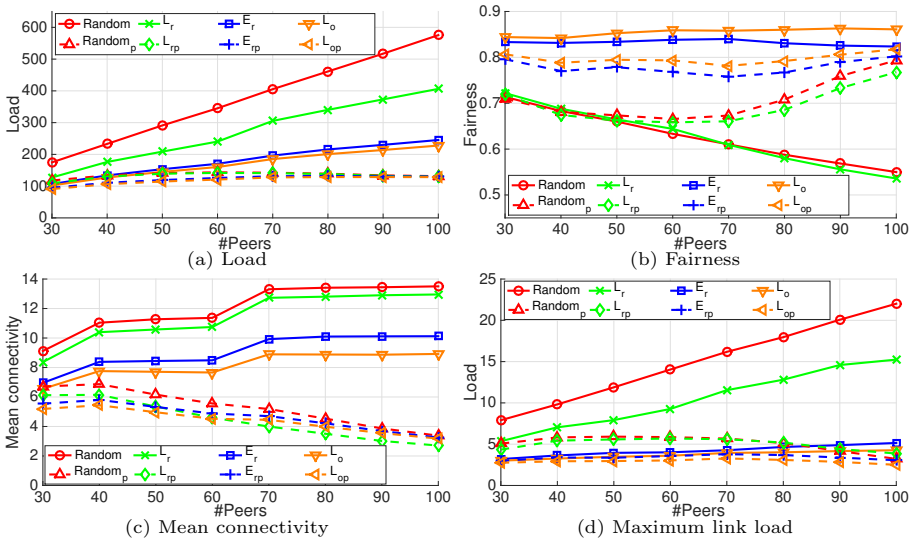
Figure 7: BA underlay, 100 peers, from 100 to 500 hosts



(a) Load for ER graphs (b) Load for BA graphs

Figure 8: ER and BA underlay, 100 peers, from 100 to 200 hosts

results we obtained in the CE topologies, the local optimization and the equalized ranking strongly improves both the random and the local ranking. This again confirms the effectiveness of our relaxation strategies. The effect of pruning is instead reduced. This is natural if we consider that by construction ER and BA underlays do not have leaf nodes, so the pruning effect is less visible. Indeed, even in this case, for the most dense scenarios there is a positive effect of pruning, which can be noticed in Fig. 8 where we report the load generated by the two heuristic strategies, with and without pruning, in the range between 100 and 200 peers. The figure shows that in dense overlays the improvement of pruning is still significant. As a general remark it also shows that it is extremely hard to optimize any distributed strategy for a generic network topology, and the choice of realistic topologies (as we do with the CA topologies and with the next section) is essential to achieve sound results.

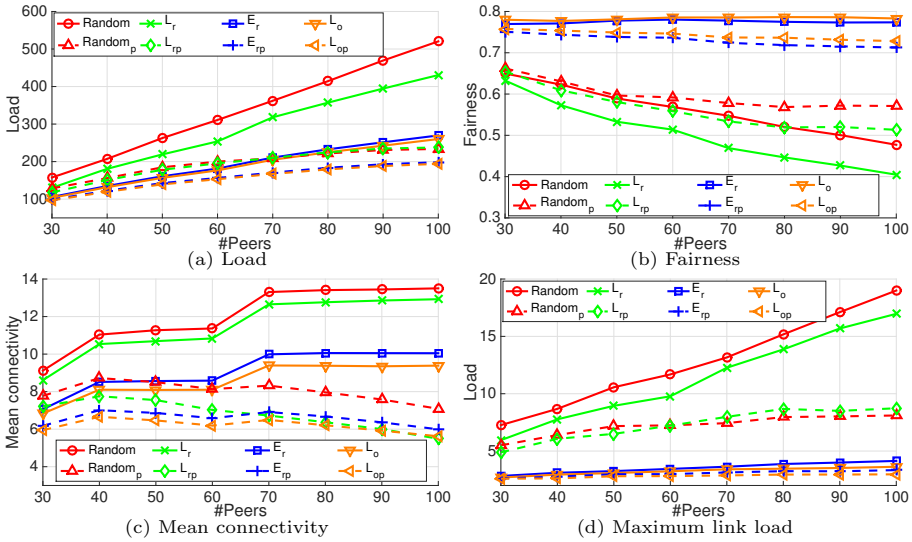


(a) Load (b) Fairness (c) Mean connectivity (d) Maximum link load

Figure 9: Ninux network, from 30 to 100 peers

7.2. Results on Real Topologies

Figs. 9 and 10 report the results computed on two topologies derived from two real networks, the ninux and FunkFeuer networks respectively made of 131 and 236 nodes. In this case, given the fixed underlay, we produced the results for an increasing size of the overlay which goes from a minimum of 30 peers to a maximum of 100 peers. The results are perfectly compatible with the ones obtained on the CE graphs. In real networks in which the number of leaf nodes is substantial, the pruned algorithms strongly improve the performance of the non pruned algorithms. As already shown for the CE underaly scenario, the improvement given by the "pruning" strategies in term of load on the underlay increases when the peers density increases (Figs. 9a and 10a) which confirms that the "pruning" is effective in excluding inefficient neighbors also in real network toplogies. This observation is confirmed also by the results reported in Figs. 9c and 10c that show how the mean connectivity decreases when the number of peers in the overlay becomes closer to the number of host in the underlay. In fact, in the scenario with 100 peers, most of them can do nothing better than choosing as neighbors in the overlay the physical neighbors in the underlay. This is not true for the case with 30 peers where the mean connectivity of the "pruning" strategies is more similar to the one obtained by the non-pruned strategies. Also with real network topologies the E_{rp} and L_{op} strategies lightly decreases the fairness if compared with the corresponding non-pruned strategies, but the negative effect is very small and negligible from a practical point of view.



(a) Load (b) Fairness (c) Mean connectivity (d) Maximum link load

Figure 10: FFW network, from 30 to 100 peers

8. Conclusions and Future Work

Video streaming is one of the most popular Internet applications, thus achieving an efficient video streaming in a Wireless Mesh Network (WMN) with a small footprint on the precious wireless resources is of paramount importance. Starting from the observation that usually WMNs allow symmetrical bandwidth and a complete knowledge of the topology, we designed a novel strategy for cooperative (live) video streaming which can be successfully deployed on any multi-hop network that provides to the network nodes a complete view of the topology. Exploiting a novel mapping of the problem onto intersection graphs, we formulated an optimization problem to build an overlay that not only reduces the total load on the underlay, but also increases the fairness in the distribution of the load on the underlay links. Given the NP-completeness of the problem in its general formulation, we proposed two relaxations based on betweenness centrality. Simulation based results show that these strategies largely outperform random overlay building techniques. Even better performances can be obtained if the topological properties of real networks are taken into consideration. Thus, we proposed a neighbour pruning strategy that takes into account the different topological positions of nodes in real networks, that are very different from purely random graphs.

In our future works we will extend our proposal to networks and applications with a larger number of hosts and peers. Currently, in our simulations, we focused on underlays made of hundreds of hosts, so that the local solutions can be easily computed on low-power devices. However, Recently introduced heuristics [35] make the computation of centrality metrics fast enough even on topologies made of thousands of nodes. We also plan to extend our research towards a

more experimental approach. The first step in this direction will be to evaluate the proposed relaxation strategies and pruning heuristics in an emulated environment (e.g., Mininet). This would allow us to evaluate our work in large-size networks and with real video streaming application (e.g., PeerStreamer). Then, for small/middle-size networks we also plan to test our algorithm through experiments in real testbeds. This would allow us to evaluate how our algorithms behaves on real embedded devices with constrained resources. Furthermore, we will study the way to approximate the centrality of nodes even when the network uses a routing protocol that does not export all the network topology but just an approximation of it, such as a distance-vector protocol. Finally, we are also interested in evaluate how the proposed strategies can be applied to different application areas that face similar problems and would benefit from a more efficient use of the underlay (e.g., surveillance in urban sensor networks [36]).

References

[1] L. Baldesi, L. Maccari, R. Lo Cigno, Improving P2P streaming in Wireless Community Networks, Computer Networks 93, Part 2 (2015) 389–403.

[2] D. Katsaros, N. Dimokas, L. Tassiulas, Social network analysis concepts in the design of wireless ad hoc network protocols, IEEE Network 24 (6) (2010) 23–29.

[3] L. Baldesi, L. Maccari, R. Lo Cigno, Optimized cooperative streaming in wireless mesh networks, in: 15th IFIP Networking Conference NETWORKING, Vienna, AT, May 2016, pp. 350–358.

[4] T. Clausen, P. Jaquet, Optimized link state routing protocol (OLSR), RFC 3626 (Oct. 2003).

[5] B. Braem, C. Blondia, C. Barz, H. Rogge, F. Freitag, L. Navarro, J. Bonicioli, S. Papathanasiou, P. Escrich, R. Baig Viñas, A. L. Kaplan, A. Neumann, I. Vilata i Balaguer, B. Tatum, M. Matson, A Case for Research with and on Community Networks, ACM SIGCOMM Comput. Commun. Rev. 43 (3) (2013) 68–73.

[6] M. E. M. Campista, P. M. Esposito, I. M. Moraes, L. H. M. k. Costa, O. C. M. b. Duarte, D. G. Passos, C. V. N. D. Albuquerque, D. C. M. Saade, M. G. Rubinstein, Routing metrics and protocols for wireless mesh networks, IEEE Network 22 (1) (2008) 6–12.

[7] D. Pisinger, The quadratic knapsack problem–a survey, Discrete Applied Mathematics, Elsevier 155 (5) (2007) 623–648.

[8] R. Birke, E. Leonardi, M. Mellia, A. Bakay, T. Szemethy, C. Kiraly, R. L. Cigno, F. Mathieu, L. Muscariello, S. Niccolini, J. Seedorf, G. Tropea, Architecture of a Network-Aware P2P-TV Application: The NAPA-WINE Approach, IEEE Communications Magazine 49 (2011) 154–163.

[9] N. Shah, D. Qian, An Efficient Unstructured P2P Overlay over MANET Using Underlying Proactive Routing, in: Seventh International Conference on Mobile Ad-hoc and Sensor Networks, 2011, pp. 248–255.

[10] D. Carra, R. Lo Cigno, E. W. Biersack, Graph Based Analysis of Mesh Overlay Streaming Systems, IEEE Jou. on Selected Areas in Communications 25 (9) (Dec. 2007) 1667–1677.

[11] G. Gheorghe, A. Montresor, R. Lo Cigno, Security and Privacy Issues in P2P Streaming Systems: A Survey, Springer Peer-to-Peer Networking and Applications 4 (Apr. 2011) 75–91.

[12] G. Ciccarelli, R. Lo Cigno, Collusion in Peer-to-Peer Systems, Elsevier Computer Networks 55 (15) (Oct. 2011) 3517–3532.

[13] T. E. Ng, H. Zhang, Predicting internet network distance with coordinates-based approaches, in: IEEE INFOCOM, New York, NY, USA, June 2002, pp. 170–179.

[14] F. Dabek, R. Cox, F. Kaashoek, R. Morris, Vivaldi: A decentralized network coordinate system, ACM SIGCOMM Computer Communication Review 34 (4) (2004) 15–26.

[15] Y. Chen, X. Wang, C. Shi, E. K. Lua, X. Fu, B. Deng, X. Li, Phoenix: A weight-based network coordinate system using matrix factorization, Network and Service Management, IEEE Trans. on 8 (4) (2011) 334–347.

[16] R. Fortuna, E. Leonardi, M. Mellia, M. Meo, S. Traverso, QoE in Pull Based P2P-TV Systems: Overlay Topology Design Tradeoffs, in: 10th IEEE Int. Conf. on Peer-to-Peer Computing (P2P-10), Delft, NL, Aug. 2010, pp. 1–10.

[17] A. Russo, R. Lo Cigno, Delay-Aware Push/Pull Protocols for Live Video Streaming in P2P Systems, in: IEEE ICC, Cape Town, ZA, May 2010, pp. 1–5.

[18] S. Traverso, L. Abeni, R. Birke, C. Kiraly, E. Leonardi, R. L. Cigno, M. Mellia, Neighborhood Filtering Strategies for Overlay Construction in P2P-TV Systems: Design and Experimental Comparison, IEEE/ACM Trans. on Networking 23 (3) (2015) 741–754.

[19] S. Dolev, Y. Elovici, R. Puzis, Routing betweenness centrality, J. ACM 57 (4) (2010) 25:1–25:27.

[20] L. Maccari, R. Lo Cigno, Pop-routing: Centrality-based tuning of control messages for faster route convergence, in: 35th Annual IEEE Int. Conf. on Computer Communications (INFOCOM), San Francisco, CA, Apr. 2016, pp. 1–9.

[21] L. Maccari, R. Lo Cigno, Waterwall: a cooperative, distributed firewall for wireless mesh networks, EURASIP Jou. on Wireless Communications and Networking 2013 (225) (Sept. 2013) 1–12.

[22] L. Maccari, R. Lo Cigno, Betweenness estimation in OLSR-based multi-hop networks for distributed filtering, Elsevier Jou. of Computer and System Sciences 80 (3) (May, 2014) 670–685.

[23] A. Vzquez-Rodas, L. J. de la Cruz Llopis, A centrality-based topology control protocol for wireless mesh networks, Ad Hoc Networks 24, Part B (2015) 34–54.

[24] Y. Sakata, K. Takayama, R. Endo, H. Shigeno, A Chunk Scheduling Based on Chunk Diffusion Ratio on P2P Live Streaming, in: IEEE NBiS, Sept. 2012, pp. 74–81.

[25] K.-L. Hua, G.-M. Chiu, H.-K. Pao, Y.-C. Cheng, An efficient scheduling algorithm for scalable video streaming over P2P networks, Elsevier, Computer Networks 57 (14) (Oct. 2013) 2856–2868.

[26] J. Zhang, W. Xing, Y. Wang, D. Lu, Modeling and performance analysis of pull-based live streaming schemes in Peer-to-Peer network, Elsevier Computer Communications 40 (Mar. 2014) 22–32.

[27] L. Abeni, C. Kiraly, R. Lo Cigno, On the Optimal Scheduling of Streaming Applications in Unstructured Meshes, in: IFIP Networking, Aachen, DE, May 2009, pp. 117–130.

[28] F. Harary, Graph theory, Addison-Wesley, Reading, MA, 1969.

[29] H. D. Sherali, J. C. Smith, An improved linearization strategy for zero-one quadratic programming problems, Optimization Letters 1 (1) (2007) 33–47.

[30] U. Brandes, C. Pich, Centrality estimation in large networks, Int. Jou. of Bifurcation and Chaos; Special Issue on Complex Networks' Structure and dynamics 17 (7) (2007) 2303–2318.

[31] L. Cerda-Alabern, On the topology characterization of Guifi.net, in: IEEE 8th Int. Conf. on Wireless and Mobile Computing, Networking and Communications (WiMob), 2012, pp. 389–396.

[32] R. Puzis, P. Zilberman, Y. Elovici, S. Dolev, U. Brandes, Heuristics for Speeding Up Betweenness Centrality Computation, in: Privacy, Security, Risk and Trust (PASSAT), Int. Conf. on and Int. Conf. on Social Computing (SocialCom), 2012, pp. 302–311.

[33] J. Lofberg, YALMIP: A toolbox for modeling and optimization in MATLAB, in: Computer Aided Control Systems Design, IEEE Int. Symposium on, IEEE, 2004, pp. 284–289.

[34] L. Maccari, R. Lo Cigno, A week in the life of three large Wireless Community Networks, Ad Hoc Networks 24, Part B, Elsevier (2015) 175–190.

[35] L. Maccari, Q. Nguyen, R. Lo Cigno, On the Computation of Centrality Metrics for Network Security in Mesh Networks, in: IEEE Global Communications Conf. (Globecom), Washington, DC, Dec. 2016, pp. 1–6.

[36] B. Rashid, M. H. Rehmani, Applications of wireless sensor networks for urban areas, J. Netw. Comput. Appl. 60 (C) (2016) 192–219.