

A Mathematical Foundation for Topology Awareness of P2P Overlay Networks

Habib Rostami and Jafar Habibi

Computer Engineering Department,
Sharif University of Technology, Tehran, Iran
{Habib, Habibi}@sharif.ir

Abstract. In peer-to-peer (P2P) overlay networks, the mechanism of a peer randomly joining and leaving a network, causes a topology mismatch between the overlay and the underlying physical topology. This causes a large volume of redundant traffic in the underlying physical network as well as an extra delay in message delivery in the overlay network. Topology mismatch occurs because overlay networks are not aware of their underlying physical networks. In this paper we present a mathematical model for topology awareness of overlay networks (degree of matching between an overlay and its underlying physical network) and the efficiency of message delivery on them. We also after determining the computational complexity of the model, propose an optimization heuristic algorithm to increase topology awareness of P2P overlay networks. Then we present the results of running the algorithm on different kinds of random graphs and show, how we can implement the algorithm over P2P networks.

1 Introduction

A peer-to-peer (P2P) overlay is a logical network on top of a physical network. This means that an overlay organizes the computers in a network in a logical way so that each node (computer) connects to the overlay network just through its neighbors.

In many existing P2P overlays like HyperCup [1], Chord [2], CAN [3], KazaA [4], Gnutella [5], Pastry [6] and P2PSLN [7] when a node joins the network, it is not optimally positioned in the overlay in respect of the underlying network such as IP network, however some of them like Coral [8], CAN [3] and Pastry [6] consider the topology that they ride. Not caring about network topology, in addition to losing performance, increases the network link stress and causes a large amount of unnecessary traffic over the physical links. Studies in [9] and [10] show that P2P traffic contributes the largest portion of the Internet traffic on some popular P2P systems such as Gnutella [5]. There are some reasons for this problem. The most important is the mechanism of a peer randomly choosing overlay neighbors without any knowledge about the underlying physical network, which causes topology mismatch between the overlay and the underlying physical network. Because of this problem, the same message may traverse the same physical underlying link multiple times, causing redundant traffic and extra delay.

For example, assume that nodes A , B , C and D are connect through a physical network shown in Fig. 1. If these nodes participate in an overlay network according

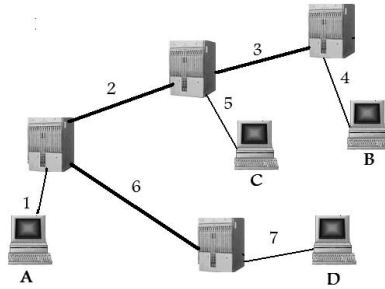


Fig. 1. Physical network which connects peers *A*, *B*, *C* and *D*

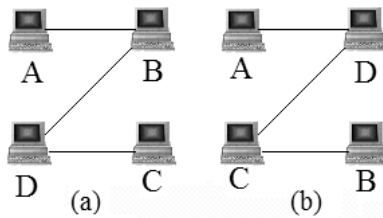


Fig. 2. Position of peers in the overlay network

to one of the two positions presented in Fig. 2(a) and 2(b) then we will have different performances.

In the overlay of Fig. 2(a), node *A* sends messages to *D* through node *B*. In this way, a message will pass across the following sequence of physical links (Fig. 1) to reach node *D* from node *A*: $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 4 \rightarrow 3 \rightarrow 2 \rightarrow 6 \rightarrow 7$. In the overlay of Fig. 2(b), node *A* sends messages to *C* through node *D*. In this way, a message will pass across the following sequence of physical links (Fig. 1): $1 \rightarrow 6 \rightarrow 7 \rightarrow 7 \rightarrow 6 \rightarrow 2 \rightarrow 5$.

If we assume that delay of sending a fixed size message over each physical link is t milliseconds, then in the overlay of Fig. 2(a), the transfer cost of edge *AB* is $4t$ milliseconds, the cost of *BD* is $5t$, the cost of *DC* is $4t$ milliseconds and the cost of the overlay (sum of the costs of all edges of the overlay) is $13t$ milliseconds. With the same assumptions, the cost of overlay 2(b) is $10t$. Therefore the overlay 2(b) is more congruent with the underlying physical network than overlay 2(a).

Many existing overlay networks are not congruent with their underlying physical network topology. Thus, a fundamental challenge in using large-scale overlay networks is to incorporate underlying topological information in the construction of the overlay to improve message delivery performance [11].

Some examples of overlay networks which introduce topology awareness are Coral [8], TOPLUS [12], SkipNet [13], Pastry [6] and CAN [3]. They use topological information to reduce the latency of sending messages in the overlays.

The contribution of this paper is the presentation of a mathematical model for topology awareness of overlay networks (degree of matching between overlay networks and

their underlying physical networks) as well as a heuristic optimization algorithm to increase topology awareness of overlay networks. In the next section we review some related works. In section 3 we present the model. Then we show the computational complexity class of the model in section 4. In section 5 we represent some definitions and results about the model. In section 6 we study the model based on a distance prediction mechanism and in section 7 we propose a heuristic algorithm to improve topology awareness of overlay networks in terms of reducing the latency of message delivery in the overlay networks. Then, in section 8 we show results of running the algorithm. In section 9, we present issues about the implementation of the algorithm over P2P networks and finally in section 10 we conclude this study.

2 Related Work

Some overlay networks like Coral [8] and CAN [3] incorporate underlying topological information in the construction of the overlay networks to improve message delivery performance. In this section we provide an overview of the mentioned overlay networks.

Coral is a peer-to-peer content distribution system which is based on a distributed sloppy hash table (DSHT) [8]. Coral uses the following technique to enable distance-optimized object lookup and retrieval. In order to restrict queries to nearby machines, Coral gathers nodes in groups called clusters. The diameter of a cluster is the maximum desired round-trip time (RTT) between any two nodes it contains. Therefore Coral uses round-trip time as distance metric obtained from the underlying topology to gain better performance [8].

Content Addressable Network (CAN) is a peer-to-peer overlay network which can be described as a distributed, Internet-scale hash table. To enable distance-optimized communication between nodes, CAN assumes the existence of a well known set of machines (for example, DNS root name servers) that act as landmarks on the Internet. It achieves a form of "distributed binning" of CAN nodes based on their relative distances from this set of landmarks. Every CAN node measures its round-trip time to each of these landmarks and orders the landmarks in order of increasing RTT. According to these distances, topologically close nodes are likely to have the same ordering and hence neighbors in the overlay are likely to be topologically close on the Internet [3].

In [14] a method is presented to enhance topology awareness of unstructured peer-to-peer systems. And [15] proposed a method to enhance topology awareness of structured peer-to-peer systems.

We saw that the mentioned overlays use underlying topological information to improve their communication performance. In other words these overlays are aware of their underlying network and use this awareness to improve their performance. Also [14] and [15] presented some techniques to enhance topology awareness of some kinds of overlay networks. But can we compare the degree of awareness of these overlays? or can we say which overlay network is more congruent with its physical underlying network? or can we increase awareness of an overlay to increase its performance?

In this paper we are going to answer these questions by modelling the topology awareness of overlay networks. Therefore our model provides a metric for comparison of overlay networks in terms of topology awareness.

3 Topology Awareness Model

In this section we define a mathematical model which models the topology awareness of overlay networks.

As we mentioned before, topology mismatch between an overlay network and its underlying physical topology, causes redundant traffic in the physical topology and extra latency in the delivery of messages in the overlay network. So the topology mismatch problem can be studied based on both redundant traffic and extra latency. We will study the problem based on extra latency in message transferring. Although there is a positive correlation between these factors. To have a sense about this correlation, roughly assume that all physical links have the same properties in terms of propagation delay and bandwidth. And assume that when a message passes across a link, causes f volume of traffic and consumes t units of time. So if a message that is sent in the overlay network, passes across n physical links, then it causes $n \times f$ volume of traffic in the physical network and consumes $n \times t$ units of time. So the volumes of traffic and latency have positive correlation.

Hereafter, we study the degree of matching between an overlay network and its underlying physical network (topology awareness of the overlay network) based on extra latency in message delivery and build our model based on it.

We define $L(IP_u, IP_v)$ as the communication cost of sending a message between two neighboring nodes u and v in terms of latency ($L(IP_u, IP_v)$ approximates the distance of nodes u and v in a physical topology in terms of delivery time). We assume that $L(\cdot)$ is symmetric¹. This means that $L(IP_u, IP_v) = L(IP_v, IP_u)$. Also the value of $L(IP_u, IP_v)$ is not constant and may change by time. So we assume that $L(IP_u, IP_v)$ is the average communication cost of sending a message between two neighboring nodes u and v . We call $L(IP_u, IP_v)$ the weight of the edge which connects u and v . Also $\sum_{uv \in E(overlay)} L(IP_u, IP_v)$ is the sum of weights of all the edges.

There are several metrics to estimate the distance between two internet hosts in terms of latency. For example IP path length, autonomous system path length, actual geographic distance and previously measured round trip times (RTT) [16]. Several efforts to obtain various sorts of distance information and other Internet characteristics were done in [17], [18], [19], [20], [21] and [22]. We can compute $L(IP_u, IP_v)$ using any of the mentioned metrics. So we will not go through the computation of $L(IP_u, IP_v)$ more. Instead we assume that the function is given and we build our model based on it. Although in section 6 we will verify our model based on computing $L(\cdot)$ according to the GNP [21] mechanism. In the rest of this paper we assume that the range of $L(\cdot)$ is positive integers (including zero). This assumption is correct because in the real world we can use smaller units like nanoseconds instead of milliseconds to free from decimal values.

Definition 1. If $G = (E, V)$ is the representative graph of an overlay, we define layout function ϕ as follows: $\phi : V \rightarrow \{IP_1, \dots, IP_{|V|}\}$, $IP_i \geq 0$, if $i \neq j \rightarrow IP_i \neq IP_j$.

Definition 2. If $G = (E, V)$ is the representative graph of an overlay with layout ϕ , we define the overlay cost as follow: $cost(G, \phi) = \sum_{uv \in E} L(\phi(u), \phi(v))$.

¹ If we assume that $L(\cdot)$ is not symmetric then we can model an overlay using a directed graph.

The goal is to find a layout ϕ that minimizes $cost(G, \phi)$. It means that nodes are positioned in the overlay in such a way that the latency of a message delivery becomes minimum in average. We define $opt(G) = \min_{\forall \phi} cost(G, \phi)$. In fact we model the topology awareness of an overlay network with representative graph G , by $cost(G, \phi)$ where the function ϕ shows positions of nodes in the overlay.

When $opt(G) = cost(G, \phi(v))$, it is expected that messages deliver over paths in optimal way in terms of latency. It is clear that in the case of $opt(G) = cost(G, \phi(v))$, according to definition of $opt(G)$, messages are transferred over edges optimally in average. It means that if we send a message over all edges of the graph, then the sum of costs (latencies) is minimized.

On the other side, if we consider all paths with $n > 2$ nodes (probably with common edges) and send a message over each of them, then the total cost is $\sum_{uv \in E(G)} \alpha_i L(u, v)$. In the optimum case, we have $\sum_{uv \in E(G)} \alpha_i L(u, v) = \alpha opt(G) + \sum_{uv \in E(G)} \beta_i L(u, v)$ where $\beta_i = \alpha_i - \alpha$ and $\beta_i \geq 0$. For graphs that all edges participate equally in paths with a given size ², $\sum_{uv \in E(G)} \beta_i L(u, v)$ becomes zero. Therefore the sum of transfer costs over paths with $n > 2$ nodes is minimized. For graphs that all edges don't participate equally in paths with a given size, the value of $\sum_{uv \in E(G)} \beta_i L(u, v)$ is not zero. Therefore for these kinds of graphs we will not reach the exact minimum values for some paths with a given size, but yet we have a reasonably accurate model for these special paths. Because the cost of each edge of the graph is optimal in average, according to definition of $opt(G)$.

Hereafter we call the problem of finding $opt(G)$, IP labelling. If we define the distance of u and v as $L(\phi(u), \phi(v)) = |\phi(u) - \phi(v)|$, it can be seen that IP labelling is a generalization of the Minimum Linear Arrangement problem [23]. The Minimum Linear Arrangement (MINLA) problem was stated in 1964 by Harper [23]. Harper's aim was to design error-correcting codes with minimal average absolute errors on certain classes of graphs.

To provide a metric for topology awareness of overlay networks we define $\alpha(G, \phi)$, which shows topology awareness of an overlay with representative graph G and layout ϕ .

Definition 3. *Topology awareness of overlay G with layout ϕ is $\alpha(G, \phi)$ where*

$$\alpha(G, \phi) = \begin{cases} 1, & cost(G, \phi) = 0 \\ \frac{opt(G)}{cost(G, \phi)}, & otherwise \end{cases} .$$

According to definition 3 and the definitions of $cost(G, \phi)$ and $opt(G)$, we can see that $\alpha(G, \phi) \in (0, 1]$ (greater values exhibit more topology awareness). Our metric has at least three applications: comparison of different overlay networks in terms of matching with their underlying physical network, evaluation of algorithms and techniques that enhance matching between an overlay and its underlying physical network ([14] and [15] presented some algorithms and we also offer such an algorithm in section 7) and using it as a guideline to construct topology aware overlay networks.

² Many overlays have this property.

4 Computational Complexity of the Problem

In respect to the point that MINLA optimization is a NP hard problem, it is natural that IP labelling is NP hard also. The following theorem shows this fact.

Theorem 1. *IP labelling optimization problem is NP hard.*

Proof. We give a reduction from MINLA to IP labelling. In graph $G = (V, E)$, if $p \in MINLA$ we define $p' \in IP$ labelling such that $\phi : V \rightarrow \{IP_1, \dots, IP_{|V|}\}$, $IP_i = i$ and $L(\phi(u), \phi(v)) = |\phi(u) - \phi(v)|$. By this statement the reduction is complete and we have the proof.

We showed that IP labelling is a NP hard problem so in general cases we should not expect to have an efficient algorithm to solve the problem. Therefore we should seek approximation or heuristic algorithms or solve the problem for particular cases.

5 Preliminary Definitions and Results

In this section we present a lower bound and an upper bound for $cost(G, \phi)$. Before presenting these bounds, we define some concepts formally.

Definition 4. *Given a graph $G = (V, E)$ and $u \in V$ and $\phi : V \rightarrow \{IP_1, \dots, IP_{|V|}\}$, $IP_i \geq 0$, if $i \neq j \rightarrow IP_i \neq IP_j$, then $cost(u, \phi) = \sum_{uv \in E} L(\phi(u), \phi(v))$.*

It is clear that $cost(G, \phi) = \frac{1}{2} \sum_{u \in V} cost(u, \phi)$.

Definition 5. *Given a graph $G = (V, E)$ and $\phi : V \rightarrow \{IP_1, \dots, IP_{|V|}\}$, $IP_i \geq 0$, if $i \neq j \rightarrow IP_i \neq IP_j$, then $L_{min} = \min_{u,v \in V} L(\phi(u), \phi(v))$ and $L_{max} = \max_{u,v \in V} L(\phi(u), \phi(v))$.*

Proposition 1. *If $G = (V, E)$ is a graph then $|E|L_{min} \leq cost(G, \phi) \leq |E|L_{max}$.*

Proof. The graph has $|E|$ edges, the weight of each edge is at least L_{min} and at most L_{max} , so the cost of G is at least $|E|L_{min}$ and at most $|E|L_{max}$.

Proposition 1 shows that the lower bound of $cost(G, \phi)$ is proportional to the number of the graph edges and the communication latency between nearest nodes. Also the upper bound denotes that if the nodes of the overlay are near in terms of latency, the communication becomes faster even for badly formed overlays. This conclusion is natural and makes sense.

6 Topology Awareness Model Based on GNP Distance Mechanism

Global network positioning (GNP) is based on absolute coordinates from modelling the Internet as a d -dimensional geometric space [21]. It is an architecture for network distance prediction that is based on peer-to-peer computing. In this mechanism, each node is assigned an absolute d -dimensional coordinate and each node maintains its own coordinates. Since nodes maintain their own coordinates, these approaches allow nodes to compute their distance from other nodes as soon as they discover each other [21].

In this section, we study the computation of $opt(G)$ if we compute $L(\cdot)$ according to the d -dimensional GNP geometric space. We study the problem for $d = 1$ in subsection 6.1 and then in subsection 6.2 we discuss the problem for arbitrary d .

6.1 IP Labelling Based on One Dimensional GNP

In the case of a one dimensional geometric space, computing $opt(G)$ (IP Labelling), becomes a version of MINLA [23] which instead of assigning numbers $1 \dots n$ to vertices, we assign n positive integers to the vertices. Therefore

$$opt(G) = \min_{\forall \phi} \sum_{uv \in E(G)} |c(\phi(u)) - c(\phi(v))| \text{ where}$$

$$\phi : V(G) \rightarrow \{IP_1, \dots, IP_n\}, IP_i \geq 0, \text{ if } i \neq j \rightarrow IP_i \neq IP_j,$$

$$c : \{IP_1, \dots, IP_n\} \rightarrow \{k_1, \dots, k_n\}, k_i \geq 0 \text{ and } L(u, v) = |c(\phi(u)) - c(\phi(v))|.$$

In the rest of this subsection we show that how we can compute the exact value of $opt(G)$, when G is a hypercube based overlay network.

Harper has considered the problem of assigning the integers $1, \dots, 2^n$ to the vertices of an n -cube so as to minimize $\sum \Delta_{ij}$, where the sum runs over all neighboring pairs of vertices and Δ_{ij} is the absolute value of difference of the numbers assigned to the vertices. He showed that the following polynomial time algorithm produces all such assignments: having assigned $1, \dots, l$, assign $l + 1$ to an unnumbered vertex (not necessarily unique) which has the most numbered nearest neighbors [23].

Steiglitz and Bernstein proved that Harper’s algorithm is correct even when we use 2^n positive integers ($k_1 \leq \dots \leq k_{2^n}$) instead of $\{1, \dots, 2^n\}$ [24]. They also showed that the minimum value is

$$\sum \Delta_{ij} = (2r_1 - n)(k_1 - k_{2^n}) + \sum_{i=2}^{2^{n-1}} (2r_i - n)(k_i - k_{2^{n-i+1}}) \tag{1}$$

where r_i is the number of ones in the binary expansion of $i - 1$.

It is clear that the problem that was solved by Steiglitz and Bernstein is IP Labelling problem based on one dimensional GNP. So we can compute the exact value of $opt(G)$ based on one dimensional GNP for overlay networks like HyperCup [1] which use hypercube as their overlay topology.

Corollary 1. *If G is a n -dimensional hypercube based overlay and we compute $L(\cdot)$ according to one dimensional GNP mechanism so that $k_1 \leq \dots \leq k_{2^n}$ are coordinates of nodes then we have $opt(G) = (2r_1 - n)(k_1 - k_{2^n}) + \sum_{i=2}^{2^{n-1}} (2r_i - n)(k_i - k_{2^{n-i+1}})$.*

Therefore to compute $\alpha(G, \phi)$ for a hypercube based overlay G , we only need to compute $cost(G, \phi)$ in steady state, which can be done using simulation techniques.

6.2 IP Labelling Based on d -Dimensional GNP

In the case of d -dimensional geometric space, computing $opt(G)$ (IP Labelling), becomes a version of MINLA [23] which instead of assigning numbers $1 \dots n$ to vertices, we assign n d -tuples of positive integers (d -dimensional vectors) to vertices. Therefore

$$opt(G) = \min_{\forall \phi} \sum_{uv \in E(G)} |\vec{c}(\phi(u)) - \vec{c}(\phi(v))| \text{ where } \phi : V \rightarrow \{IP_1, \dots, IP_n\},$$

$$IP_i \geq 0, \text{ if } i \neq j \rightarrow IP_i \neq IP_j, c : \{IP_1, \dots, IP_n\} \rightarrow \{\vec{k}_1, \dots, \vec{k}_n\}, \vec{k}_i \in \mathbb{Z}^{+d} \text{ and}$$

$$L(u, v) = |\vec{c}(\phi(u)) - \vec{c}(\phi(v))|.$$

In the following we present a lower bound for $opt(G)$, so we can have a lower bound for $\alpha(G, \phi)$.

Lemma 1. *If $P = v_1v_2 \dots v_n$ is a path, $\phi : V(P) \rightarrow \{IP_1, \dots, IP_n\}$ and $c : \{IP_1, \dots, IP_n\} \rightarrow \{\vec{k}_1, \dots, \vec{k}_n\}$ and $opt(G) = cost(G, \phi)$ then $opt(P) \geq |\vec{c}(\phi(v_n)) - \vec{c}(\phi(v_1))|$.*

Proof. $opt(P) = cost(P, \phi) = \sum_{i=1}^{n-1} |\vec{c}(\phi(v_i)) - \vec{c}(\phi(v_{i+1}))| \geq |\sum_{i=1}^{n-1} \vec{c}(\phi(v_i)) - \vec{c}(\phi(v_{i+1}))| = |\vec{c}(\phi(v_n)) - \vec{c}(\phi(v_1))|$.

Theorem 2. *If $G = (V, E)$ is a k -connected graph, $\phi : V \rightarrow \{IP_1, \dots, IP_{|V|}\}$ and $c : \{IP_1, \dots, IP_{|V|}\} \rightarrow \{\vec{k}_1, \dots, \vec{k}_{|V|}\}$ and $\vec{k}_1 \leq \dots \leq \vec{k}_{|V|}$ then*

$$opt(G) \geq k \times |\vec{k}_{|V|} - \vec{k}_1|.$$

Proof. In a k -connected graph, there are k disjoint paths between each two nodes [25]. Now consider two nodes u and v such that $\vec{c}(\phi(u)) = \vec{k}_1$ and $\vec{c}(\phi(v)) = \vec{k}_{|V|}$ (ϕ is a layout that $opt(G) = cost(G, \phi)$). Therefore there are k disjoint paths between u and v . According to lemma 1, weight of each path is greater than $|\vec{k}_{|V|} - \vec{k}_1|$ so the weight of the k paths is at least $k \times |\vec{k}_{|V|} - \vec{k}_1|$.

Corollary 2. *If G is a hypercube based overlay network (n -dimensional) then because a n -dimensional hypercube is n -connected, based on GNP mechanism we have:*

$$opt(G) \geq n \times |\vec{k}_{2^n} - \vec{k}_1|, \text{ where } |\vec{k}_1| \leq \dots \leq |\vec{k}_{2^n}|.$$

7 Optimization Algorithm

In this section we present a heuristic optimization algorithm to increase $\alpha(G, \phi)$ by reducing $cost(G, \phi)$ in graph G . Given that the function $cost$ is a model for latency of message delivery, the algorithm reduces message delivery latency in an overlay network. Also according to definition 3, our algorithm increases topology awareness of overlay networks. The following algorithm uses a local optimization to reduce the total cost of its input graph. According to the algorithm, each node tries to stand in a position in such a way that the communication cost of the overlay becomes minimum. Therefore each node checks whether swapping its node with a neighbor reduces the cost function. Also according to the algorithm, the overlay topology remains fix and only nodes swap their positions.

Algorithm 1 is highly scalable because each node only needs to know about its neighbors and IP of neighbors of neighbors, which receives from its neighbors.

Algorithm 1. Input. $G=(V, E)$.

```

repeat
  cond = false
  for all edges uv of G do
    if (cost(u, phi) + cost(v, phi) >
        cost(u, phi') + cost(v, phi')) then

```

```

    swap positions of u and v
    cond = true
while(cond = true)

```

In algorithm 1, alpha' and phi' stand for α and ϕ functions if we swap positions of u and v . The following theorem reveals the correctness of the mentioned algorithm.

Theorem 3. *In the algorithm 1, $cost(G, \phi)$ is reduced by each swap.*

Proof. After swap of u and v we name the mapping function, ϕ' . So

$$cost(G, \phi') = cost(G, \phi) - \underbrace{cost(u, \phi) - cost(v, \phi) + cost(u, \phi') + cost(v, \phi')}_{<0}.$$

Therefore $cost(G, \phi') < cost(G, \phi)$ and the proof is complete.

Proposition 2. *Algorithm 1 on graph G , terminates with at most $|E|(L_{max} - L_{min})$ node swaps.*

Proof. According to proposition 1, the difference between the worst case and the best case is $|E|(L_{max} - L_{min})$. If each swap reduces the cost only by one unit ($L(\cdot)$ and $cost(G, \phi)$ are integers), then after at most $|E|(L_{max} - L_{min})$ swaps, we will reach the minimum value and the algorithm terminates.

8 Experimental Results

In this section we describe the methodology and results of running the algorithm 1 on some different random graphs. In this simulation we use one dimensional GNP mechanism [21] to compute the distances of nodes. So we assign a random positive integer to each node as its coordination in GNP space. And therefore $L(\phi(u), \phi(v))$ equals to the absolute difference of the assigned numbers to the nodes. For example if 1000 is assigned to u and 300 is assigned to v then $L(\phi(u), \phi(v))$ is 700. During the simulation the graph is static. It means that there is no dynamic node arrival and departure during the simulation. We aim to compute the percent of optimization in topology awareness of the tested graphs by running the algorithm 1 on them. We use the following formula to compute optimization rate:

$$optimization\ rate = 100 \times \frac{\alpha(G, \phi') - \alpha(G, \phi)}{\alpha(G, \phi')} = 100 \times \frac{cost(G, \phi) - cost(G, \phi')}{cost(G, \phi)}.$$

Where ϕ is the layout of nodes in graph G before running the algorithm and ϕ' is the layout of nodes after running the algorithm.

We use the JUNG [26] library to write a java program for execution of alorithm 1. We generate simple random³, Kleinberg small world with clustering exponent 2 [27], Eppstin power law [28] and Watts beta small world with $\beta = 0^4$ [29] graphs with 1000

³ Simple graphs where $|V|$ vertices are generated and $|E|$ random edges are chosen pairwise uniformly.

⁴ β is the probability of an edge being rewired randomly.

nodes and run the algorithm on them⁵. Then we run the algorithm on the same kinds of graphs with 1500 nodes and then 2000 nodes and . . . 8000 nodes. In these graphs the number of edges is 10 times the number of nodes ($\frac{|E|}{|V|} = 10$). Then we generate the same kinds of graphs with the same number of nodes but this time the number of edges is 15 times the number of nodes. We Also repeat the process by graphs with $\frac{|E|}{|V|} = 20$. Also we run the algorithm on Kleinberg graphs with parameter zero. Fig. 3 shows the results of running the algorithm on mentioned graphs.

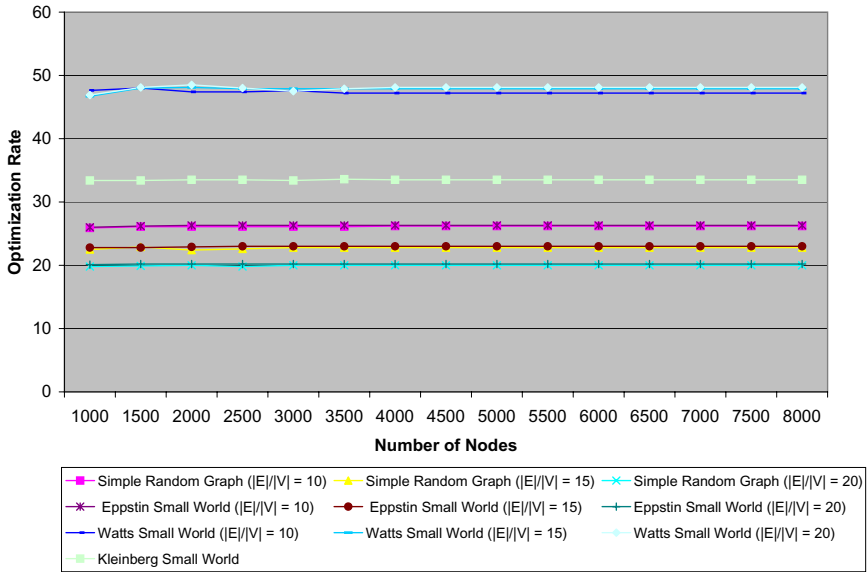


Fig. 3. The results of running the algorithm on the graphs

As it can be seen, in the tested graphs, increasing the number of nodes will not change the performance of the algorithm in terms of improvement of topology awareness. So the algorithm is highly scalable. Also the optimization rate is different for the different kinds of graphs. For example, the improvement of topology awareness in the Watts small world graphs with $\beta = 0$ is about 48 percent while this improvement for Eppstin power law graphs is about 22 percent. Therefore the optimization rate is dependent to the structure of the input graphs. On the other side when we increase ratio of edges to nodes, In the simple random graphs and the Eppstin power law graphs, the performance of the algorithm decreases while this performance increases for Watts beta small world graphs with $\beta = 0$. So the performance of the algorithm to increase topology awareness, in addition to the type of graphs, is dependent to density of graphs.

⁵ We generate each kind of graph with a fixed size 15 times and run the algorithm on them. Then the average of the results was represented as the result of running the algorithm on that kind with the given size.

9 Implementation over P2P Networks

To support algorithm 1, each node, keeps a list of its neighbors' IP as well as a boolean flag against each neighbor. The flag indicates that whether the node should check the swapping condition or the neighbor does it. It means that if u and v are neighboring nodes in the overlay network then if $u.flag_v = false$ and $v.flag_u = true$, then v should check the swapping condition (we say that v is an active node and u is a passive node). It is important that the two sides of a connection should not have the same flag, so we set that for a connection uv , the flag of the node with the greater IP (each IP can be assumed as a 32 bit positive integer) will be true and the flag of other node for the connection is false. Active nodes periodically (each t seconds) check the swapping condition. An active node v which connect to node u , sends its neighbors' IP to u , and waits for the response of u . Node u sends back its neighbors' IP as well as its distance to its neighbors and the neighbors of v (Node u can compute its distance to the neighbors of v using any of the mentioned methods in section 3, but the most straight forward method is using round-trip time (RTT) by pinging target nodes). After that, v has enough information to check the swap condition. If the swap condition is true then v informs u to start the swapping process. The algorithm 1, may never converge in a real P2P system because of the continuous joining and departure of nodes. But the algorithm causes a continuous improvement in the P2P network in terms of matching with its underlying physical network. And therefore we will have continuous improvement in message delivering performance and reduction of underlying physical network traffic.

In the swapping process between two neighboring nodes u and v , u connects to all neighbors of v (except itself) and v connects to the neighbors of u . Then they put their old connections (except uv connection) into a to-cut list (see Fig. 4). After this process, a peer will not send or forward messages to connections in its to-cut list, but these connections are kept alive because these links are needed when receiving previous query responses that have to be forwarded according to their inverse search path. A connection in a to-cut list will be cut after a certain time period. We can find the optimal value of the period using simulation techniques.

In some peer-to-peer networks, when two nodes swap, they should update some information and tables. We update by building temporary tables and information as we do for connections. In addition to the mentioned guidelines for peer swapping that we pointed out in this section, we should note that swapping process is a network dependent operation and should be designed independently for each overlay network.

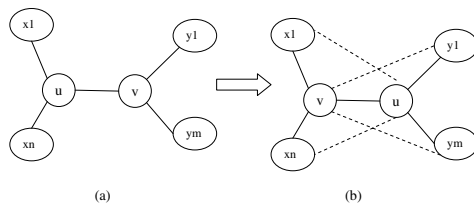


Fig. 4. Swapping process. In (b), the dashed lines show to-cut connections.

Node v before checking the swap condition with node u , should compute its distance with its neighbors and neighbors of u . Also u should know its distance with its neighbors and the neighbors of v . We assume that $L(\cdot)$ is dynamic. So node x have to communicate with node y to find its distance to it.

If we assume that in average, each node has σ neighbors then in the worst case, $O(\sigma)$ messages are needed to check the swapping condition and doing the swapping process. If the frequency of checking the swapping condition by nodes is f , then the extra messages that are caused by the algorithm is $O(n\sigma f)$, where n is the number of nodes in the network. Because f is a constant, the order of extra messages created by the algorithm in one minute (unit of time) is $O(n\sigma)$. As observed in [30], each peer issues 0.3 queries per minute in average. Thus, the traffic incurred by an unstructured P2P network with n peers which uses broadcast to search objects is $O(n^2)$ and the traffic caused by a structured P2P network which causes $\log(n)$ message per search is $O(n\log(n))$. Studies in [10] show that σ is very smaller than the number of nodes in the network (in some structured overlays like HyperCup [1] and Chord [2], σ is equal to $\log(n)$). Therefore in both cases, the algorithm will not increase the order of traffic of the network. As we saw before, the algorithm will decrease traffic by increasing topology awareness.

10 Conclusion

We introduced a mathematical metric for the degree of topology matching between an overlay network and its underlying physical network. We constructed the model based on an optimization problem that we called it IP labelling. Also we have shown that IP labelling optimization is a NP hard problem. Then we showed that this NP hard problem is solvable in polynomial time for some particular inputs. Also we proposed an optimization heuristic algorithm to solve the problem for general cases. Then we ran the algorithm on different kinds of random graphs and we presented the results. We also noted how the algorithm can be implemented on P2P networks.

References

1. Schlosser, M., Sintek, M., Decker, S., Nejd, W.: Hypercup - hypercubes, ontologies and efficient search on p2p networks. In: International Workshop on Agents and Peer-to-Peer Computing, Bologna, Italy (2002)
2. Stoica, I., Morris, R., Karger, D., Kaashoek, M.F., Balakrishnan, H.: Chord: A scalable p2p lookup service for internet applications. In: Proc. ACM SIGCOMM'01. (2001)
3. Ratnasamy, S., Francis, P., Handley, M., Karp, R., Shenker, S.: Scalable content-addressable networks. In: Proc. ACM SIGCOMM'01. (2001)
4. : Kazaa. <http://www.kazaa.com> (2003)
5. : Gnutella. <http://gnutella.wego.com> (2003)
6. Rowstron, A., Druschel, P.: Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In: International conference on Distributed Systems platforms (Middleware). (2001) 329–350
7. Zhuge, H., Liu, J., Feng, L., Sun, X., He, C.: Query routing in a peer-to-peer semantic link network. Computational Intelligence **21** (2005) 197–216

8. Freedman, M., Mazieres, D.: Sloppy hashing and self-organizing clusters. In: Proc. 2nd International Workshop on Peer-to-Peer Systems (IPTPS03). (2003)
9. Saroui, S., Gummadi, K.P., Dunn, R.J., Gribble, S.D., Levy, H.M.: An analysis of internet content delivery systems. In: Proc. IEEE Fifth Symp. Operating Systems Design and Implementation. (2002)
10. Sen, S., Wang, J.: Analyzing peer-to-peer traffic across large networks. In: Proc. ACM SIGCOMM Internet Measurement Workshop. (2002)
11. Ratnasamy, S., Handley, M., Karp, R., Shenker, S.: Topologically-aware overlay construction and server selection. In: Proc. IEEE INFOCOM'02. (2002)
12. Garces-Erice, L., Ross, K., Biersack, E., Felber, P., Urvoy-Keller, G.: Topology-centric lookup service. In: Proc. The 5th International Workshop on Networked Group Communications (NGC'03). (2003)
13. Harvey, N.J.A., Jones, M.B., Saroui, S., Theimer, M., Wolman, A.: Skipnet: A scalable overlay network with practical locality properties. In: Proc. The Fourth USENIX Symposium on Internet Technologies and Systems (USITS03). (2003)
14. Liu, Y., Xiao, L., Liu, X., Ni, L.M., Zhang, X.: Location awareness in unstructured peer-to-peer systems. *IEEE Transactions on Parallel and Distributed Systems* **16** (2005) 163–174
15. Ferreira, R.A., Jagannathan, S., Grama, A.: Enhancing locality in structured peer-to-peer networks. In: Proc. ICPADS04 Tenth International Conference on Parallel and Distributed Systems. (2004)
16. Huffaker, B., Fomenkov, M., Plummer, D., Moore, D., Claffy, K.: Distance metrics in the internet. In: IEEE International Telecommunications Symposium (ITS). (2002)
17. P. Francis, S. Jamin, V.P.L.Z.D.F.G.Y.J.: An architecture for a global internet host distance estimation service. In: Proc. IEEE INFOCOM'99. (1999)
18. Guyton, J.D., Schwartz, M.F.: Locating nearby copies of replicated internet servers. In: Proc. ACM SIGCOMM'95. (1995)
19. Savage, S., Collins, A., Homan, E.: The end-to-end effects of internet path selection. In: Proc. ACM SIGCOMM'99. (1999)
20. Paxson, V.: End-to-end routing behavior in the internet. In: Proc. ACM SIGCOMM'96. (1996) 25–38
21. Ng, T.E., Zhang, H.: Predicting internet network distance with coordinates-based approaches. In: Proc. IEEE INFOCOM'02. (2002)
22. Chen, Y., Katz, R.: On the placement of network monitoring sites. <http://www.cs.berkeley.edu/yanchen/vnms/> (2001)
23. Harper, L.: Optimal assignments of numbers to vertices. *J. Soc. Industrial Appl. Math.* **12** (1964) 131–135
24. Steiglitz, K., Bernstein, A.J.: Optimal binary coding of ordered numbers. *J. Soc. Industrial Appl. Math.* **13** (1965) 441–443
25. West, D.B.: *Introduction to Graph Theory*. second edn. Prentice Hall (2001)
26. White, S., O'Madadhain, J., Fisher, D., Boey, Y.B.: Jung-java universal network/graph framework. <http://jung.sourceforge.net/index.html> (2004)
27. Kleinberg, J.: The small-world phenomenon: An algorithmic perspective. In: Proc. 32nd ACM Symposium on Theory of Computing. (2000)
28. Eppstein, D., Wang, J.: A steady state model for graph power laws. *ACM Computing Research Repository* (2002)
29. Watts, D.: *Small world: The dynamics of networks between order and randomness*. Princeton Univ. Press (1999)
30. Sripanidkulchai, K.: The popularity of gnutella queries and its implications on scalability. <http://www2.cs.cmu.edu/knunwadee/research/p2p/gnutella.html> (2001)