A Bottleneck Eliminating Approximate Algorithm for PON Layout

Samee Ullah Khan and Munib Ahmed
Department of Computer Science and Engineering
University of Texas
Arlington, TX 76019, USA
{samee, munib.ahmed}@uta.edu

Abstract

To layout a passive optical network (PON) under the constraints of cost effectiveness is essentially a constraint optimization problem. In this paper, we first show that in general this problem is hard not solvable in polynomial time. We then follow it up by exploiting certain graph theoretical techniques and proposing an algorithm that produces an effective layout for PONs. We verify our results via an experimental methodology, where our proposed approach performs extremely well compared to randomized layouts.

1. Introduction

A PON consists of an optical line terminator (OLT), usually located at the central office (CO) and a set of associated optical network terminals (ONTs), usually located at the subscribers' premises. Between them lies the optical distribution network comprised of fibers and splitters.

The main component of PON is an optical splitter device. Depending on which direction the light is travelling, it splits the incoming light and distributes it to multiple fibers towards ONTs, or combines it into one towards the OLT. The PON when included in the fiber to the home (FTTH) architecture, runs an optical fiber from the CO to an optical splitter and towards an ONT which then distributes it to the subscribers' locations. Fiber to the cabinet (FTTCab) architecture runs an optical fiber from the CO to an optical splitter and then onto the optical network units (ONUs) where the signal is converted to feed the subscribers over a twisted copper pair. The PON technology uses a double–star architecture. The first star topology centers at the OLT, and the second at the optical splitter [3].

With PONs, all active components between the CO and the subscribers' premises are eliminated, and passive optical components are put into the network to guide traffic based on the splitting power of optical wavelengths to endpoints along the way. The splitters are mere devices working to pass or restrict light, and as such, have no power or processing requirements and have virtually unlimited mean time between failures thereby lowering the overall maintenance costs. In the near future we may witness a full blown PON deployment in every major city across the globe. Therefore, efficient and computationally feasible algorithms are needed to compute PON deployments [3].

2. Problem Description

The $p-1$ fault–tolerant PON Network Layout (PNL) problem can be formulated as a graph theoretical problem.

Consider a graph $G(V,E)$ such that V represents the physical locations of the subscribers, CO, and any other location acquired by the CO to expand its network, and E represents the communication lines between two V_i's. If there is no direct communication line $c(i,j)$ between V_i and V_j, we consider the shortest path between them. Without the loss of generality we assume that $c(i,j)=c(j,i)$. Furthermore, E satisfies triangle inequality, i.e., $c(i,j)+c(j,k)\geq c(i,k)$, for each i,j,k.

Notice that we can further classify V into the obvious categories that represent the locations of OLTs, ONTs and splitters. Formally, the PNL problem can be stated as:

"Given an undirected graph G, find the locations of ONTs and splitters such that the cost of the equipment is minimized and for QoS the maximum distance between: 1) ONTs and splitters, and 2) splitters and OLTs, is minimized, subject to the constraint that the network has tolerance to operate with $p-1$ PON equipment (ONTs or splitters or both) failures."

We assume that the OLT resides inside the CO. Furthermore, for simplicity, we treat ONT and ONU as more or less the same entities. The problem formulation does not consider the optimization of the distance between ONTs and the subscribers' computing devices, since the distance between them is negligible.

3. Our Approach

3.1 Preliminaries

Below we elucidate some necessary terms and conditions, which will form the basis of the proposed PNL algorithm.

Definition 1 *A dominating set of a graph G is a subset $(S \subseteq V)$ such that every vertex in $V - S$ is adjacent to a vertex in S. Computing a dominating set is NP–complete [2].*

Definition 2 *A square of a graph G, G^2, is a graph containing an edge (u, v) whenever G has a path of length two–hop between u and v, and $u \neq v$.*

Definition 3 *A star graph is a tree on $n + 1$ nodes with one node having vertex degree of n and the others having vertex degree 1.*

Definition 4 *An independent set I of a graph G is a subset of the vertices such that no two vertices in the subset represent an edge of G. The cardinality of the independent set $|I|$ is the measure of the number of vertices included in the set.*

Definition 5 *The maximal independent set M of a graph is the maximum cardinality of a set of vertices such that no two vertices in the subset represent an edge of G.*

Definition 6 *A p–independent set is a subset $S \subseteq V$ such that every node $v \in V - S$ has at most $p - 1$ neighbors in S.*

Definition 7 *A p–dominating set is a set $S \subseteq V$ such that each node $v \in V - S$ has at least p neighbors in S.*

Definition 8 *A clique in a graph G, is a set of vertices $S \subseteq V$ such that for every two vertices $(u, v) \in S$, there exists an edge connecting u and v.*

Reflecting back at the problem formulation, one can see that the PNL problem is a two–step optimization process.

1. Step 1: The first part concentrates on estimating the locations of ONTs. Since ONTs are not costly, the optimization process only focuses on the reduction of the fiber length. Moreover, due to the random incursion and removal of subscribers, it becomes necessary to spread the ONTs as much as possible in the system.
2. Step 2: The second part concentrates on estimating the locations of splitters. This step takes the locations of ONTs (as identified previously) as input, and uses this information to optimize the locations of splitters. Since splitters are not as cost effective as ONTs, this step focuses on minimizing: 1) the fiber length and 2) the deployment cost of splitters.

Notice that the two steps differ from each other only by the additional constraint of splitters' cost minimization — step 2. Therefore, for simplicity, we will only describe step 2 in the subsequent text. In either of the steps, our aim is the following. Suppose we wish to locate components at k out of n vertices of G such that the maximum distance of a vertex to its p–th closest component is minimized. Considering the p–th closest component is important when the components concerned are subject to failures and we wish to ensure that even if up to $p-1$ components fail, every vertex (by vertex we mean the locations of ONT, OLT, splitters, etc.) has a functional component close to it.

The PNL algorithm that will be derived in the subsequent text, gathers inspiration from [1], where the authors developed a general paradigm for approximating NP–hard problems. We describe their paradigm in the context of the PNL problem as follows. Let $e_1, e_2, \cdots, e_k$ be the edge weights (fiber cost) in increasing order and let G_i be the subgraph induced by edges of weight at most e_i. Using Definition 1, if we are able to find the minimum edge weight e_i such that G_i has a dominating set of size at most k, then we can effectively identify the locations for the PON components. While it is easy to generate the subgraphs $G_1, G_2, \cdots, G_k$, the problem of checking if these subgraphs have a dominating set of size at most k is NP–complete (see Definition 1). However, if in a subgraph G_i we can find an independent set I of size more than k such that no vertex in G_i is adjacent to vertices of I (Definition 4), then any dominating set in G_i has a unique vertex dominating each vertex of I. Note that any dominating set in G_i has a unique vertex dominating each vertex of I and therefore cannot be of size k or less.

In order to successfully attack the PNL problem using the concept of independent set I, we make use of power graphs. Given a graph $G = (V, E)$ the x–th power of G, G^x, is a graph with the same vertex set as G and an edge between two vertices if they are connected by a path of at most x edges in G. Based on this discussion, we say that I is an independent set of vertices in G_i^2 (see Definitions 2 and 4). To argue that G_i has no dominating set of size at most k, it suffices to find an independent set in G_i^2 of size larger than k. This argument can easily be guaranteed, if one assumes that the independent set in G_i^2, I, is maximal (denoted as M), i.e., the addition of any other vertex to M yields a set which is not independent. But this implies that every vertex not in M has a neighbor in M which means that M is a dominating set in G_i^2 (see Definition 5).

3.2 The Hardness of the Problem

Below we will iteratively gather results that will finally aggregate towards the formal proof that:

If $P \neq NP$, there exists no polynomial time algorithm to solve the PNL problem.

Lemma 1 *Let U be maximal k–independent set such that $|U| \geq k$, then U is a k–dominating set in G^2.*

Proof: If a node v belongs to $V - U$ is picked in U, then v has a k–th neighbor u, which is in fact adjacent to $k-1$ u nodes all belonging to U. By Definitions 1 and 2 the set of nodes u are neighbors in G^2.

Lemma 2 *Let V be a k–dominating set in G, then $|U| \leq |V|$ holds for any k–independent set U in G^2.*

Proof: For the two non–trivial cases of $U \subset V$,

- U is not contained in V.

  Pick a node u at random such that u belongs $U - V$. Thus a set of nodes S can be defined such that the neighborhood of u $N(u)$ is not contained in V, i.e., $S = N(u)$ intersection V. Let L define the set of nodes that are adjacent to S, then any node v in $S \cup L$ is contained in G^2 (Definition 2) and is adjacent to at most k vertices in $S \cup L$.

- U is contained in V.

  If U is contained in V, then we can define a graph G containing vertices $V-(S\cup L)$. Thus the lemma would hold, if $V' = V - (S \cup L) = V - S$, where V' is a k–dominating set in G. Pick a random node v in G not contained in V, then v belongs to $V - V'$, and v has at least k neighbors in G not present in V'. Since we assumed that V' is a k–dominating set in G, no neighbor belongs to G (by the definition of G). Thus $N(v) \cap V'$ is a subset of V'.

Theorem 1 *Assuming $P \neq NP$, for any arbitrary fixed $a \leq p$, there does not exist any polynomial time algorithm for PNL.*

Proof: Suppose we have an algorithm A, which gives a solution for the PNL problem, then a solution for dominating set can be obtained. We will now give a polynomial time reduction from PNL to the dominating set problem. Let $|V|$ be the pairwise neighborhood graph such that by picking any vertex v in V, $N(v)$ intersection V is null. Thus the graph to find PNL can be computed by picking vertices as follows:

- $(u, v) = 1$ if $u, v \in V$,
- $(u, v) = 1$ if $u \in V$ and $n \in N(v)$,
- $(u, v) = f(|V|) + \epsilon$ otherwise.

The choice of ϵ (epsilon), exhibits the epsilon approximation factor in the final layout. If G has the dominating set of size d, then the solution for PNL has a set J such that $d \cup N(v)$ is a set of nodes with cardinality $d + (a-1)$, where $a \leq p$. For $a = 1$, the problem reduces to minimum k–center problem, so we will consider $a \geq 2$. Pick any node v in $N(v)$. It is clear that v has only one neighbor in V', which is at a distance of 1 (triangle inequality). If it is not equal to 1, then it must be covered by a neighbors within a distance of 1. Let $Z = P$ intersection V such that Z is subset of V and contains d nodes such that $d = |P| - (a-1)|V|$, then any arbitrary node v belongs to $V - Z$ must have at least a nodes in P with a distance of 1, but by definition and previous argument, only $a - 1$ nodes can form the neighborhood. Thus, Z is a dominating set of size d, but intutively, G cannot contain a dominating set of size d.

3.3 PNL Algorithm

In essence, the PNL algorithm starts with a p–independent set. If this p–independent set is not a p–dominating set, then the algorithm finds a vertex that has less than p neighbors in the set and adds this vertex to the set to obtain another p–independent set with strictly higher potential. (This process of adding vertices continues till there are no vertices with neighbors less than p.)

We first describe the so called *Certificate Function* (which can be useful to terminate the PNL algorithm possibly earlier). This function takes as input a graph G, and returns either a maximal independent set M or a certificate of failure.

Certificate Function
Input: G.
Output: Certificate of failure or M.
1. If $(|M| \geq p)$.
2. Return Certificate of failure.
3. If $(|M| < p)$.
4. While $|M| <> p$.
5. Add vertices to M with less than p neighbors, till $|M| = p$.
6. Return M.
7. Endw.

Let P_o represent the optimal placement of the PON components at either of the two phases of the PNL. Let the optimal solution value be denoted by $\sigma_o = f(P_o)$. Function f is an assumed *Black Box*, which takes as an input a graph G and outputs a numerical value representing the solution quality of the PON component placements. (We will use this numerical value to derive the theoretical bounds.) Given a solution s the *Certificate Function* will either inform us that $\sigma_o > s$ (certificate of failure), or will deliver a solution.

Below we detail the working of the PNL algorithm. The PNL algorithm takes as input:

1. Graph G.
2. k which is the number of the clustered population. (The numerical value of k will become clear when we describe the experimental setup. For the time being assume that k has a specific value.)
3. p which carry two different meanings.
   (a) Step 1: p represents the physical locations of the subscribers. (These locations are approximated to the nearest vertex in graph G.)
   (b) Step 2: p represents the physical locations of the ONTs, as identified in Step 1.

In the PNL algorithm, first, the edges in G are sorted in the increasing order, i.e. $\sigma(e_1) \leq \sigma(e_2) \leq \ldots \leq \sigma(e_n)$. By making use of the *Certificate Function* the PNL algorithm performs a binary search to locate the minimum i (line 6 of the PNL algorithm) such that the *Certificate Function* returns a maximal independent set M (line 6 of the *Certificate Function*). We can be sure that such a set M would exist because of the construction of the *Certificate Function* which guarantees that $\sigma_o > \sigma(e_i)$. Lines 1–11 of the PNL algorithm would repeatedly call the *Certificate Function* to identify the locations of the ONTs, such that, each ONT is in the vicinity of at least $p-1$ ONTs. For the optimization process of Step 2 of the PNL problem formulation, i.e., the identification of the locations of the splitters, the PNL algorithm invokes lines 12-13, which reinvokes the *Certificate Function*. The only difference is that the *Certificate Function*, now, adds vertices to M (line 5 of the *Certificate Function*) in the increasing order of their weights $w(v_i)$ (cost of a splitter). We deliberately did not depict this difference when writing the *Certificate Function* in order to avoid confusion.

PNL Algorithm
Input: G, k and p.
Output: G' (Final PNL).
1. Sort G ($\sigma(e_1) \leq \sigma(e_2) \leq \ldots \leq \sigma(e_n)$).
2. L $= k-1$.
3. H $= |V|$.
4. While (H $-$ L) > 1.
5. $z =$ (H $+$ L)$/2$.
6. $i = \lceil z \rceil$
7. $G_i = \text{PNL}(G, k, p)$.
8. Result $=$ Certificate(G_i).
9. If Result is the certificate of failure; then L $= i$.
10. If Result is M; then H $= i$.
11. Endw.
12. $\{G', k', p'\} =$ Certificate(G_i).
13. PNL(G', k', p').
14. Repeat 1–11.
15. Output G'.

Lemma 3 *The PNL algorithm is complete and will identify a solution, if there exists one.*

Proof: Let D and D_m represent the dominating set and the minimum dominating set in a graph G, respectively. Step 1 of the algorithm lines 1–11 (optimizing the locations of ONTs), is based on the fact that $|M| \leq D$. Then, in the worst case G would contain $|D_m|$ stars spanning all vertices, and also there would be $|D_m|$ cliques (see Definition 8) spanning all the vertices. Therefore, at most one and only one vertex can be picked from each clique. Step 2 of the algorithm (optimizing the locations of the splitters) is the same optimization procedure as step 1 but with weighted constraints, and the only difference is that now in the weighted graph the neighbors are picked in G and not in G^2.

Lemma 4 *In the worst case, the PNL algorithm computes a solution in $\mathcal{O}(n^5 \log n)$.*

Proof: The repeated calls to the *Certificate Function* in the PNL algorithm is the most time consuming operation. The *Certificate Function* has to first construct G^2 in order to find the maximal independent set M. This construction can be performed in $\mathcal{O}(n^3)$ [4]. Fortunately, the call to the *Certificate Function* is at most made $\mathcal{O}(\log n)$ times (due to the binary search pattern), and the time complexity becomes $\mathcal{O}(n^3 \log n)$. The maximal independent set on the other hand can be identified in $\mathcal{O}(n^2)$ [4]. Since these two operations are interdependent, the overall running time now becomes $\mathcal{O}(n^5 \log n)$.

4 Experiments and Discussions

We evaluated our proposed algorithm from the point of view of saving fiber and equipment cost. We also identified the relationships between splitters, ONTs and the fiber length being used. We considered a 25.9 km^2 *Manhattan* grid with 50 m. apart streets. The choice of Manhattan grid was due to the fact that underlying graph should exhibit the property of triangle inequality.

The locale was populated with 20,000 personnel (random distribution), clustered into small, medium and large

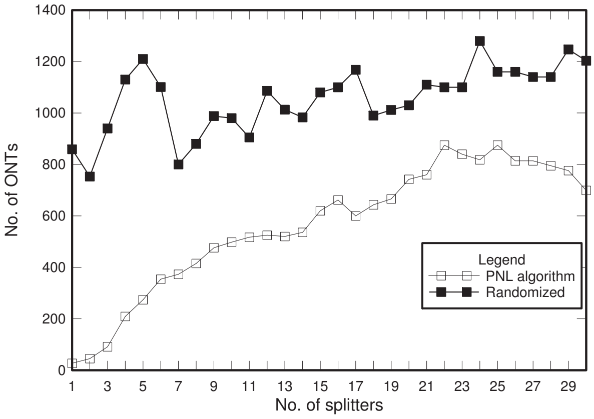

Figure 1. No. of splitters versus no. of ONTs.

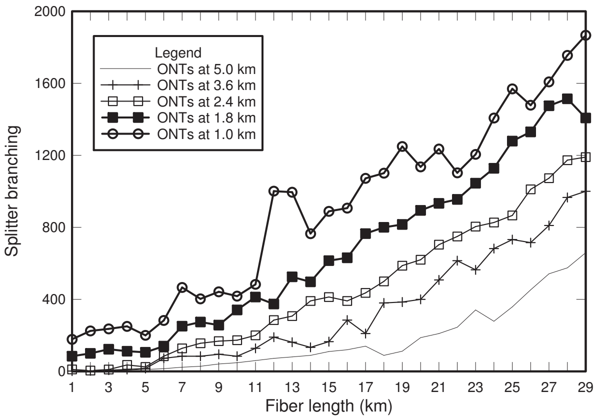

Figure 2. Fiber length versus splitter branching.

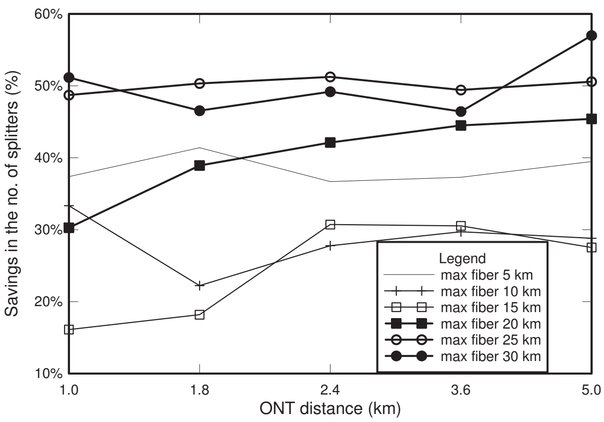

Figure 3. ONT distance versus splitters savings.

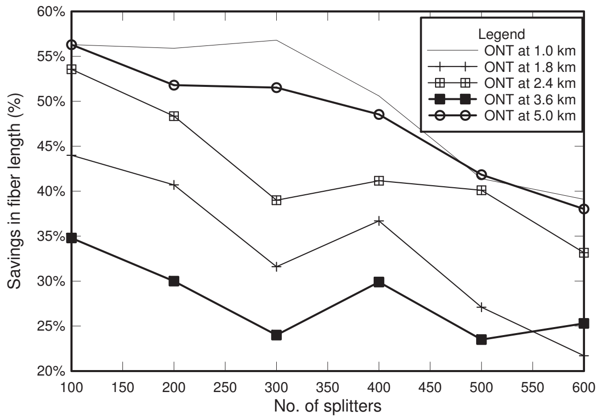

Figure 4. No. of splitters versus fiber savings.

businesses. We assume that small business has on average 6 personnel, medium has 60 and large franchise has 2000 employees on average [3]. The number of clusters in the system is represented by k, which is one of the inputs to the PNL algorithm.

Fiber cost was assumed to be \$100/m, ONT at \$50/unit, splitter at \$600/unit and OLT at \$250/unit. These costs are assumed only to drive the PNL algorithm. Realistic costs can be obtained from the market at the time of deployment.

To evaluate the solution quality of the PNL algorithm, we first generated random network layouts. The main purpose was to compare the PNL algorithm with arbitrary layouts and measure the difference (increase/decrease) in the solution quality brought by the PNL algorithm in terms of savings in the number of ONTs, splitters, and the fiber used. Note that for each experimental setup a total of 5 different random layouts were generated in order to suppress the pseudo–randomness phenomenon.

In the first experiment (Figure 1), we measured the relationship between the number of ONTs and the number of splitters. (The savings in ONTs reflect the appropriate utilization of splitters and the fiber.) PNL performed exceptionally well by reporting a reduced usage of ONTs. For instance, a complete coverage with random layout was achieved with 5 splitters and 1210 ONTs. (By complete coverage we mean that every subscriber had an access to the PON.) Using the same setup, the PNL algorithm achieved a complete coverage with 5 splitters and 274 ONTs — a saving of approximately 77%.

Next, we observed the relationship between the length of the fiber used and splitter branching. Splitter branching refers to the number of branches originating from a splitter to its associated ONTs. (The number of the splitters in the system is indirectly related to splitter branching. That is, it is more likely that the number of splitters in the system is higher when the splitter branching is higher. This relationship is more rigid when an upper bound on the number of branches originating from a splitter is maintained. However, in our experimental setup we do not enforce such a limit.) To observe the relationship between the length of the fiber used and the splitter branching (Figure 2), the maximum distance allowable for the placement of ONTs was varied between 1.0 km and 5.0 km. As expected a consistent increase was observed with the decrease in the allowable ONT distances.

Next, we observed the relationship between the maximum allowable distance for the placement of ONTs and the savings in percentage brought by the PNL algorithm in the number of splitters used subject to the constraint of the maximum allowable fiber usage compared to the random layouts (Figure 3). In most cases, savings in the splitter usage increased with the increase in the maximum allowable distance for the placement of ONTs. As much as 57% savings was observable under the constraints of the maximum allowable distance for the placement of ONTs as 5.0 km and the maximum allowable fiber usage as 30 km.

In the last experiment (Figure 4), we measured the relationship between the number of splitters and the savings in percentage brought by the PNL algorithm in the fiber length subject to the constraint of maximum allowable distance for the placement of ONTs compared to the random layouts. To observe the general overall trend, we hypothetically increased the number of splitters in the system regardless of the fact that most of them were of no use in the system. It was observed that with the increase in the number of splitters in the system, the fiber usage increased and less savings were reported.

5 Conclusions

We studied the passive optical network layout (PNL) problem as a graph theoretical problem and proposed a PNL algorithm based on this transformed problem. The algorithm delivered a solution of 4–approximation to the optimal solution with a guaranteed tolerance towards $p - 1$ equipment failures. We also identified via experimental studies the relationships between the splitters, ONTs and fiber used. The proposed framework for the PNL problem was kept as general as possible so that future research can be undertaken by extending the problem formulation to incorporate various other issues related to the passive optical networks (PONs) such as, variable upstream/downstream bandwidth, incorporating existing ducts and cabinets, extending the current work to formulate various product specific requirements for ATM based PONs, Broadband PONs, Ethernet based PONs, and Gigabit PONs.

References

[1] D. S. Hochbaum and D. B. Shmoys. A Unified Approach to Approximation Algorithms for Bottleneck Probelms. *Journal of ACM*, 33(3):533–550, 1986.

[2] R. W. Irving. On Approximating the Minimum Independent Dominating Set. *Information Processing Letters*, 37:197–200, 1991.

[3] S. U. Khan. Passive Optical Network Layout in Manhattan. *IEEE Photonics Technology Letters*, 15(10):1488–1490, 2003.

[4] S. U. Khan. Heuristics-based PON Deployment. *IEEE Communications Letters*, 9(9):847–849, 2005.