Heuristics-Based PON Deployment

Samee Ullah Khan

***Abstract*— This letter proposes a simple greedy algorithm and three of its variants for the passive optical network (PON) deployment on an arbitrary grid. The algorithms guarantee a result of 2–approximation to the optimal solution with worst case running time of $\mathcal{O}(n^2 + m^2)$. This improves upon the best previously known result of 3–approximation with worst case running time of $\mathcal{O}(n^5)$. We also improve the simulation study by introducing the concept of population density function embedded graphs. The experimental results reveal that the proposed algorithms save 45–65% of the deployment cost (fiber, equipment, etc.) on average.**

***Index Terms*— Optimization methods, optical fiber communication, passive optical network (PON), optical access network.**

I. Motivation

With PONs, all active components between the central office exchange and the customer premises are eliminated, and passive optical components are put into the network to guide traffic based on splitting the power of optical wavelengths to endpoints along the way. The Splitters are merely 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 near optimal deployments.

II. Problem Formulation

A PON consists of an optical line terminator (OLT) located at the central office (CO) and a set of associated optical network terminals (ONTs) located at the customer's premise. Between them lies the optical distribution network comprised of fibers and Splitters (see Fig. 1) [1].

The PON deployment (POND) problem can be formulated as a graph theoretical problem. Consider a graph $\mathcal{G}(V, E)$, such that V represents the physical locations of the subscribers, CO and another locations 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.

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.

Manuscript received February 8, 2005. The associate editor coordinating the review of this letter and approving it for publication was Prof. Maode Ma.

The author is with the Department of Computer Science and Engineering, The University of Texas at Arlington, Arlington, TX, USA (e-mail: sakhan@cse.uta.edu).

Digital Object Identifier 10.1109/LCOMM.2005.09002.

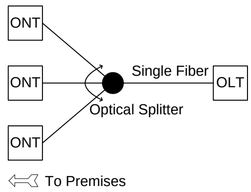


Fig. 1. Basic PON architecture.

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

"*Given an undirected graph $\mathcal{G}$, 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.*"

The problem formulation does not consider the optimization of the distance between ONTs and the subscribers' computing devices, since the distance between them is negligibly small.

In the generalized case, the POND problem is NP–hard [1].

III. Proposed Approach

A. Population Density Function Embedded Graphs (PDF–G)

First, we have to obtain a graph $\mathcal{G}$ that accurately represents the terrain map with population. Obtaining PDF–G involves finding a transformation $r \to \mathcal{G}(r)$ of a plane (grid) to another plane (PDF embedded graph) such that the Jacobian $\frac{\partial(\mathcal{G}_x, \mathcal{G}_y)}{\partial(x,y)}$ of the transformation is proportional to some PDF, thus:

$$\frac{\partial(\mathcal{G}_x, \mathcal{G}_y)}{\partial(x, y)} \equiv \frac{\partial \mathcal{G}_x}{\partial x}\frac{\partial \mathcal{G}_y}{\partial y} - \frac{\partial \mathcal{G}_x}{\partial y}\frac{\partial \mathcal{G}_y}{\partial x} = \frac{\rho(r)}{\overline{\rho}}, \qquad (1)$$

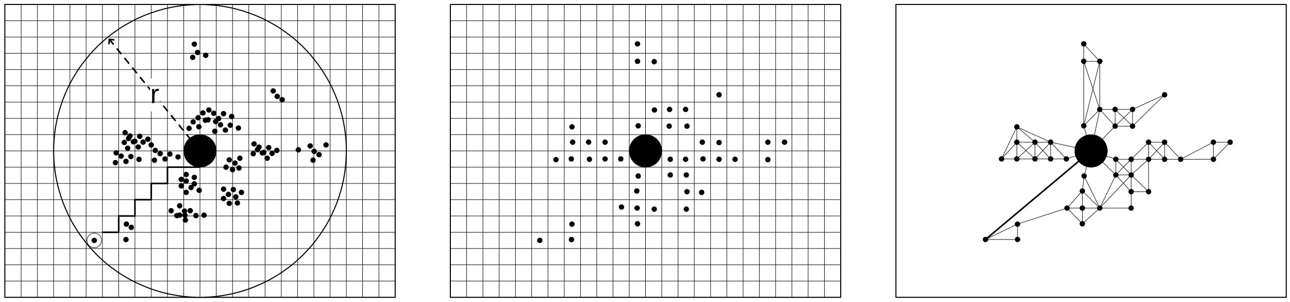
where $\mathcal{G}_x$ and $\mathcal{G}_y$ are the x and y transforms of $\mathcal{G}$, respectively, while $\overline{\rho}$ is the mean population density average over the area that is to be transformed into $\mathcal{G}$. The PDF that we use is the well–studied and highly cited empirical density function reported in [2], which is hypothesized to follow:

$$\ln \rho(r) = \ln \rho_0 + a\sqrt{r} \qquad (0 \leq r \leq L; a < 0), \qquad (2)$$

where $\rho(r)$ is the population density at distance r from the center, ρ_0 is the density at distance 0 (the center of the metropolitan area), a is the rate at which the logarithm of density decreases with the square root of distance from the center and L represents the city limits. Plugging back $\rho(r)$ into Eq. 1 would effectively transform the terrain map with population to a graph $\mathcal{G}$ having the population embedded in it. Fig. 2 briefly describes the process of obtaining the PDF–G.

B. User Access Requirements

This corresponds to the traffic generated by the users, while accessing data using the conventional networks. Here, we have



(a) $n \times n$ Manhattan grid is populated using Eq. 2, with population represented as dots. The large black dot denotes ρ_0. A zigzag path is also shown to represent the shortest path in terms of Manhattan distance.

(b) Clustering based on user access requirements (Sec. III-B) is performed, followed by solving the Jacobian. Notice, the change in the: 1) density of dots (due to clustering) and 2) location of the dots (due to the Jacobian).

(c) Nodes in the PDF–G represent the embedded population density and approximate major building location as (x, y). Shortest distance between the nodes is denoted by thin lines; the thick line represents the zigzag path.

Fig. 2. An overview of how to obtain PDF–G from an arbitrary terrain.

TABLE I
USER ACCESS REQUIREMENTS

Type	**Traffic**		
	Small Business	**Medium Business**	**Large Business**
Voice	0.5MB	1.0MB	1.5MB
Web	0.9MB	1.35MB	1.8MB
E–mail	15kB	150kB	500kB
FTP	5MB	7MB	7MB
Fax	4kB	4.8kB	4.8kB
Total/person	6.419MB	9.5048MB	10.8048MB
Total/busines	38.514MB	570.288MB	21609.6MB

assumed such a data (Table I); however, in reality, the user access requirements can easily be obtained by monitoring the network traffic. We assume that small, medium and large businesses employ 6, 60 and 2000 employees, respectively.

C. Proposed Algorithm(s)

POND is a two step optimization problem [1].

Step 1: (Locate ONT positions.)

a : Take as input (the PDF–G) graph $\mathcal{G}$.

b : Apply the Greedy algorithm (see Sec. III-C.1) to locate the positions of ONTs and transform $\mathcal{G}$ into $\mathcal{G}'$. $\mathcal{G}'$ has the positions of ONTs marked.

Step 2: (Locate Splitter positions.)

a : Take as input $\mathcal{G}'$.

b : Apply the Greedy algorithm to locate the positions of Splitters and transform $\mathcal{G}'$ into $\mathcal{G}''$. $\mathcal{G}''$ (has the positions of Splitters marked and) is the solution to the POND problem.

1) Greedy Algorithm: The algorithm works in an iterative fashion, so that the objective function (minimizing the maximum distance subject to the fiber length and equipment cost) is each time reduced as much as possible.

Algorithm Description—It is sufficient to consider only *Step 1* for the description purposes. *Step 2* works in a similar fashion. The algorithm takes as input an $n \times n$ symmetric distance ($c(i, j) = c(j, i)$) matrix that satisfies the triangle inequality. Iteratively, the algorithm builds the final solution in n (number of ONTs) steps so that, given a partial solution C_{i-1}, it forms a new partial solution C_i by extending C_{i-1} with the vertex V_i which is the closest to the C_{i-1}, i.e., the vertex V_i which minimizes the distance ($d(V_i, C_{i-1})$) between any vertex $V_k \in C_{i-1}$ and V_i at step i. Since there can be at most n possible candidates in each step, therefore, the entire process would take at most $\mathcal{O}(n^2)$.

The idea behind minimizing $d(V_i, C_{i-1})$, is to scatter the equipment as much as possible subject to the fiber usage and cost constraints. Note that each equipment has an operational range ($\mathcal{L}$). For instance, the CO management may require ONTs or Splitters not to be placed more than 10 km from each other, since the total operational range of PON is 20 km [1]. Therefore, for any vertex selection $V_i \subseteq V$ of designated ONTs, the *radius* of V_i is the minimum distance $\mathcal{L}$ between V_i and some vertex $V_k \in C_{i-1}$. Thus, the algorithm in each step i, ignores the vertices that lay beyond $2\mathcal{L}$. The bound of $2\mathcal{L}$ is due to the triangle inequality ($\mathcal{L} + \mathcal{L} \geq \mathcal{L} \Rightarrow 2\mathcal{L} \geq \mathcal{L}$).

Theorem 1 (Completeness): The Greedy algorithm is complete and will identify a solution, if there is one.

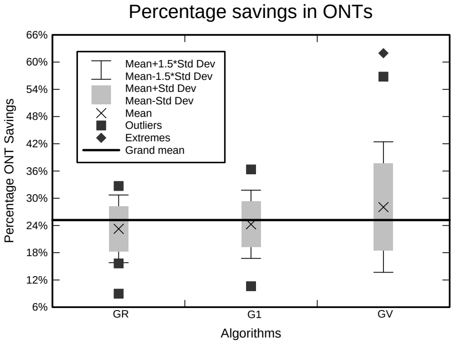
Proof: We must only show that if there is a partial solution C_{i-1} with operational radius at most $\mathcal{L}$, then the algorithm outputs a solution C_i after selecting V_i. Suppose there exists C_{i-1}, and assign each candidate vertex V_i^* to its closest vertex in C_{i-1}. Thus, we have partitioned the vertices into at most n parts, $C_{i_1}^*, \cdots, C_{i_n}^*$. The algorithm can select at most one vertex form each part $C_{i_i}^*$, since the selection of V_i causes all remaining vertices in $C_{i_i}^*$ to be ignored; thus, the algorithm correctly identifies a solution $C_i = C_{i_i}^*$ which minimizes $d(V_i, C_{i-1})$. ■

Theorem 2 (Timings): If a solution is identifiable, the Greedy algorithm computes it in $\mathcal{O}(n^2 + m^2)$.

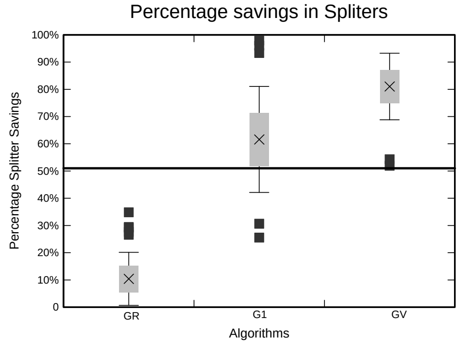
Proof: In *Step 1*, the algorithm applies a simple search over the graph $\mathcal{G}$, which is transformed into $\mathcal{G}'$; a worst case running time of $\mathcal{O}(n^2)$. In *Step 2*, notice that the number of vertices in $\mathcal{G}'$ are less than n, since o of them were marked with the positions of the ONTs, i.e., $m = n - o$. Thus, the overall running time is $\mathcal{O}(n^2 + m^2)$. ■

Theorem 3 (Impossibility): If there exists a POND algorithm of t–approximation with $t < 2$, then P=NP.

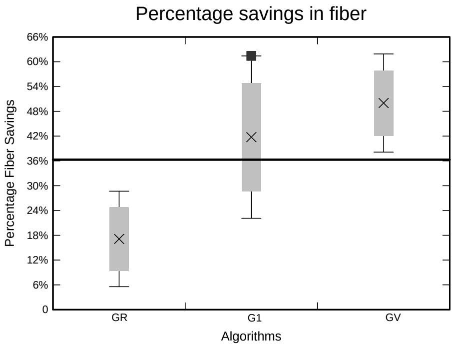
Proof: To see this, consider the *dominating set* problem [1, p. 1488], which is NP–hard. Given an instance of the dominating set problem, we can define an instance of the POND problem by setting the distance between the adjacent vertices to 1, and non–adjacent vertices to 2: there is dominat-



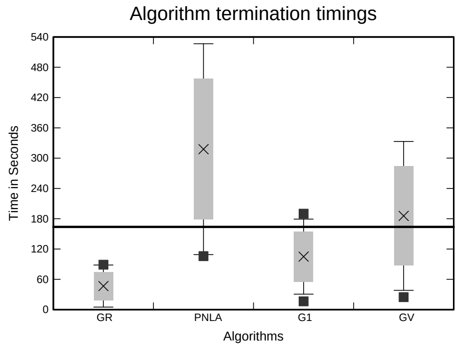
(a) Savings in the ONT usage.



(b) Savings in the Splitter usage.



(c) Savings in the fiber usage.



(d) Algorithm termination timings.

Fig. 3. Simulation results.

ing set if and only if the optimal radius for this POND instance problem is 1. Furthermore, any t–approximation algorithm with $t < 2$ must always produce a solution of radius of 1 if such a solution exists, since any solution of radius $t < 2$ must actually be of radius 1. Thus, unless $2\mathcal{L} = \mathcal{L}$, the algorithm cannot find a solution better than 2–approximation. ■

It is to be noted that the Greedy algorithm suffers from the dilemma of how to choose the very first location of both the ONT (*Step 1*) and the Splitter (*Step 2*). For the selection of the first location there may be several possibilities. However, we only focus on three natural choices.

2) Random: Choose the location randomly.

3) 1–center: Treat the problem as finding the location of only one ONT and one Splitter. This will most likely result in locations that are in the core of the PDF, i.e., ρ_0.

4) $|V|$–times: Apply the heuristic $|V|$ times, each time with different starting vertex, and then choose the best solution.

IV. Experiments and Discussion

We considered a 20-km^2 Manhattan (New York City) grid with streets 50 m apart. The choice of the Manhattan grid was due to the fact that the POND problem formulation will not hold if the deployment is performed on an arbitrary graph whose edges do not exhibit triangle inequality [1].

The Manhattan locale was populated using Eq. 2, with $\rho_0 = 62,900$, $a = -0.781$ and $L = 10$ km [2, p. 296] — a total of 2,41,182 personnel. After populating the grid, clustering was performed with cutoffs as: 1) small, 2) medium and 3) large businesses. This was followed by obtaining the PDF–G by solving Eq. 1. Fiber cost was assumed to be \$25/m, ONT at \$50/unit, Splitter at \$600/unit and OLT at \$250/unit. These costs also encapsulate the labor costs. Realistic costs can easily be obtained at the time of the POND from market resources.

We compared the three variants: Greedy Random (GR), 1–center (G1) and $|V|$–times (GV) with the PNL algorithm (PNLA) reported in [1]. The solution quality was measured in terms of the percentage savings in: 1) ONTs, 2) Splitters and 3) fiber used. Each algorithm's performance was recorded against the variable equipment operational range $\mathcal{L} = [1 - 10]$ km. Figs. 3(a)–3(c) summarize the results. The idea was to measure the deviation in terms of solution quality, compared to that of the PNLA. The Outliers and Extremes were limited to 2 and 3 standard deviations, respectively.

The plots are self–explanatory; it can be seen that GV and G1 clearly outperformed the PNLA, while GR suffered from the choice of randomized initial location (notice that the difference in the solution quality between the heavy–duty PNLA and GR is minute), however, GR is 6 times faster than PNLA (Figure 3(d)).

V. Conclusion

We proposed a 2–approximation algorithm for the generalized POND problem. The experimental results revealed significant savings in the usage of ONTs, Splitters and fiber used compared to the best previously known POND algorithms.

References

[1] S. U. Khan, "Passive optical network layout in Manhattan," *IEEE Photon. Technol. Lett.*, vol. 15, pp. 1488–1490, Oct. 2003.

[2] J. B. Parr, "A population–density approach to regional spatial structure," *Urban Studies*, vol. 22, pp. 289–303, Aug. 1985.