

Topology Abstraction Algorithms for Light-Mesh – An Alternate Model for PON

¹Anuj Agrawal, ¹Ashwin Gumaste, ¹Mohit Chamania and ²Nasir Ghani

¹Dept. CSE, Indian Institute of Technology, Bombay, ²Dept. of ECE, University of New Mexico at Albuquerque
Email: anuj.agrawal@jit.iitb.ac.in, ashwing@ieee.org, mohitc@iitb.ac.in and nghani@ece.unm.edu

Abstract: Light-mesh – an alternate solution for access networks is presented. Two heuristic topology algorithms are discussed and simulated showing cost and performance benefits.

© 2008 Optical Society of America

OCIS codes: 060.4250 Networks; 060.4250 Networks

1. Introduction and Light-mesh Concept

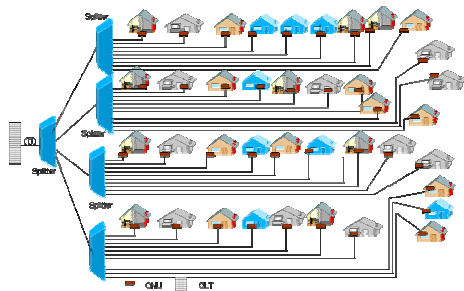


Fig. 1. Conventional PON

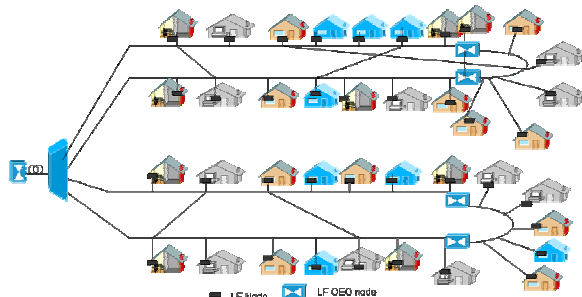


Fig. 2. Proposed Light-mesh network.

The growth of broadband services like *Video-on-Demand* (VoD), online-gaming, IPTV and Triple Play have propelled the need for high-speed broadband access using fiber technology. Passive optical networks (PONs) are currently emerging as optical layer solutions in the access and are based on a dual-wavelength star solution, interconnecting end-user ONUs to service-provider Central Office (CO) based OLT, in technology flavors that range from LAN (EPON, GEAPON) to TDM (GPON) variants. The primary deployment deterrent in the access area for PONs is the high-costs due to fiber laying. A technology solution that can alleviate this problem is the light-frame based mesh approach, first described in [1, 2] that uses a mesh topology for providing end-user connectivity. As can be seen from Fig. 1 and Fig. 2, mesh architecture is (1) more naturally suited for the access and (2) results in significantly lower fiber costs. A mathematical explanation of mesh approach resulting in lower fiber requirement is shown in [1] while intuitively is evident from Fig. 1 and Fig. 2. The mathematical constraint shown in [1] determines the condition when a mesh network results in lesser fiber requirements than a PON (star) solution, and is based on the theory that end-users are close to each-other with the average-distance between the end-users and the CO being greater than the *spread* of the network (diameter of area of coverage) as shown in Fig. 3. We define such a mesh-based access network as a *light-mesh* and characterize this as an N^2 connected network that has two types of nodes – *OOO* and *OEO*. The light-mesh is interconnected through *strings* and *threads*, where, strings are all-optical buses while threads are point-to-point all-optical paths that connect a node in one string to a node in another. In principle, the light-mesh is a graph which has multiple DAGs such that each DAG is further characterized by *OEO* nodes at the periphery and all other nodes being of *OOO* type. The light-mesh potentially reduces fiber costs in the access area by 45 % ~ 90 % for networks with 16 ~ 64 nodes [1]. This potential fiber savings is significant from a network planning perspective and potentially offsets all network-element costs. Technically the light-mesh is based on pragmatic optical packet transport technology that enables all-optical sub-wavelength access using a smart collision detection algorithm and supporting node-architecture. *OOO* nodes are purely passive nodes with electronics for client/local access only. The idea is such that when an access network is configured as a mesh and nodes are switched *OFF*, the network would still continue to support connectivity. Architecture of such nodes as well as more “intelligent” *OEO* nodes is described in the next Section. The entire network supports N^2 connectivity with the added constraint that there is no-all-optical cycle in the design. So one interesting problem is to place *OEO* nodes at smart locations leading to a cycle-free design, while maintaining a *fully connected* network. This problem is shown to be *NP-complete* [1]. In this paper our objective is to propose a topology abstraction algorithm that defines the

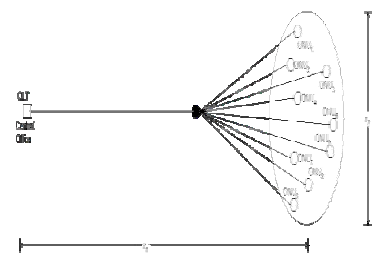


Fig. 3. Spread

virtual topology of the light-mesh to be able result in a low-cost, service-aware access network [3]. Section 2 describes the light-mesh node architecture while Section 3 details heuristic algorithms for topology design. Section 4 presents simulation results.

2. Node architecture and Problem Definition

The node architecture of *OOO* nodes is shown in Fig. 4. The node is in a 3x3 configuration with 2-ports allocated for local access (I/O) and the remaining 4-ports for network access. Two of the network ports are connected to a string while the remaining 2 are *potentially* connected to threads. Optical signal entering the node (at the string input) in form of optical frames is split into 3 copies at the first splitter called the DBC (*drop and bifurcate coupler*). The first-copy is locally dropped for possible electronic processing; the second copy is forwarded all-optically to the second coupler, while the third copy is possibly sent to a thread input. The second coupler (AAC – *add and admittance coupler*) is a 3x1 combiner that can possibly receive packets from (1) local input – a burst-mode transmitter, (2) the DBC and (3) from a thread. A thread that enters the node is split into two parts using a splitter, with one copy of the input signal being locally processed using a burst-mode receiver. The *OEO* variant of the node has a disconnect between the string input and string output as well as between the thread input and the AAC as compared to the *OOO* version.

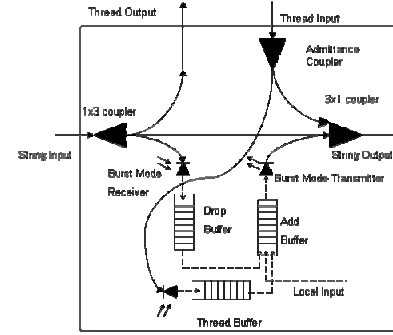


Fig. 4. Light-mesh node architecture

Transmission in the light-mesh is based on optical packets using a CSMA-like principle, i.e. a node senses the channel (string) and if no upstream node on the string is transmitting data then it inserts its own packet (locally). Collisions in the light-mesh can happen and are dealt with using a novel collision detection scheme now described. The principle of this scheme is that any two inputs that receive a packet in overlapping time-interval, do so, such that the packet is split into two copies – one stored locally and the other undergoes collision. Hence, if two local buffers receive packets in overlapping time-intervals then the node *knows* that a collision occurred at the optical layer and has also saved electronically a copy of the two packets that collided. The node then transmits the two copies that were locally saved and in this way collision detection and recovery is performed.

While the cost benefit of a mesh in access is evident from Fig. 1 and Fig. 2, the optical bus property of strings and opportunistic scheduling mechanism of CSMA result in a higher average delay as compared to a star. However, by efficiently designing the logical topology (that leads to the creation of strings and threads), it is possible to keep a service-aware delay bound while maintaining N^2 connectivity essential for a service provider access network [4].

3. Heuristic Algorithms for Topology Design

```

select an OEO node ( $C$ ) connected to a passive splitter
  add it to  $Y$ 
while ( $Y \neq V$ )
  find OOO members ( $M$ ) which satisfy the delay constraints and power constraints
  add the OOO nodes to  $Y$ 
  find an available OEO node ( $E$ ) to be the end node which satisfy the delay constraints and power constraints
  add the OEO end node to  $Y$ 
  create a string with  $C$ ,  $M$  and  $E$ 
  add the string to the stringList
  let  $C = E$ 
end while
  
```

Main algorithm

We take as input – the set of all nodes denoted by N : $\{N_1, \dots, N_N\}$; the set of all *OEO* nodes denoted by N_E : $\{N_{1e}, \dots, N_{ve}\}$; and the set of all *OOO* nodes denoted by N_O : $\{N_{1o}, \dots, N_{vo}\}$. Thus, $N = (N_E \cup N_O)$. We want to produce as an output a graph $G(V, E)$ which is obtained by connecting the nodes such that the end-to-end delay between any two nodes does not exceed a service maxima of Δ_{max} and the power of the optical signal at any point does not drop below a threshold P_{th} .

To design the topology, we populate the graph $G(V, E)$ by first creating a set of strings such that every node N_i is part of a string. Once all the strings are formed, we create threads between these strings to achieve N^2 connectivity while ensuring that the end-to-end delay and the optical power loss constraints are satisfied.

We begin by taking an empty set Y , to which we will assign nodes as they are included in a string. The algorithm works as follows: We start by selecting an *OEO* node that is connected to the CO. This node becomes the start-node of the first string. The end-node and *OOO* members of this string are determined by the power threshold and delay constraint as shown in *Main algorithm*. Once a string is formed, all the nodes of the string are added to Y . The next string is formed by making the last node of the previous string as the start-node of the next string. We continue in this fashion until all nodes have become a part of a string, that is, until $Y = V$. Thus, we now have a graph that contains all the nodes, with each being part of a string and is denoted by *stringList*. To create threads between any two nodes N_i and N_j we assume the following constraints – $C_T(N_i, N_j)$:

1. Thread output of N_i is free (available).
2. Thread input of N_j is free (available).
3. Connecting N_i and N_j does not lead to a cycle.
4. N_i and N_j are not part of the same string.
5. If S_1 and S_2 are two strings such that $N_i \in S_1$ and $N_j \in S_2$, and N_{E2} is the end-node of S_2 and N_{E1} is the convener node of S_1 , then all optical path from N_{E1} to N_{E2} (assuming N_i and N_j connected) satisfies the optical power loss constraint
6. N_i is not a start-node of any string

To obtain the final graph $G(V, E)$ we now need to create threads. There are two approaches we consider: (1) random-threading and (2) maximal-threading. In random threading, a thread is formed between any-two nodes such that the thread satisfies the $C_T(N_i, N_j)$ constraints. In maximal threading, the idea is to create threads that stretch the all-optical path to the maximum allowable power-threshold and delay limit while satisfying the $C_T(N_i, N_j)$ constraints.

```

Approach 1 - Random threading:
while ((G does not have  $N^2$  connectivity) and (end-to-end delay <  $\Delta_{max}$ ))
    for every node  $N_i$ 
        find a node  $N_j$  satisfying the constraints  $C_T(N_i, N_j)$ 
        create a thread from  $N_i$  to  $N_j$ 
    end for
end while

Approach 2: Maximal threading:
while ((G does not have  $N^2$  connectivity) or (end-to-end delay >  $\Delta_{max}$ ))
    for every node  $N_i$ 
        find all nodes  $N_j'$  satisfying the constraints  $C_T(N_i, N_j')$ 
        select a node  $N_j$  from  $N_j'$  which maximizes the number of nodes between  $N_{E1}$  and  $N_{E2}$ 
        create a thread from  $N_i$  to  $N_j$ 
    end for
end while

```

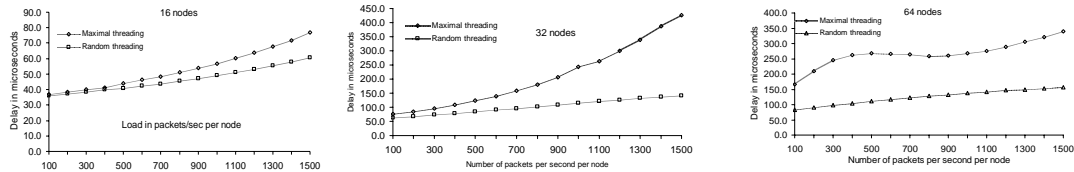


Fig. 5. Arrival rate versus delay for 16 nodes. Fig. 6. Arrival rate versus delay for 32 nodes. Fig. 7. Arrival rate versus delay for 64 nodes

4. Simulation

We simulated light-mesh network with 16, 32 and a 64-node configurations to observe delay as a function of load for the two topology design schemes proposed. λ is the rate at which packets are generated at a node. In the simulation, we varied λ from 100 packets/sec to 1500 packets/sec in increments of 100. Packet size was fixed at 1500 bytes and line-rate was 1 Gbps. Maximum power-threshold was 28 dB and allowable end-to-end delay was pegged at 5 ms. We simulated the two heuristics to design the topology and computed delay statistics.

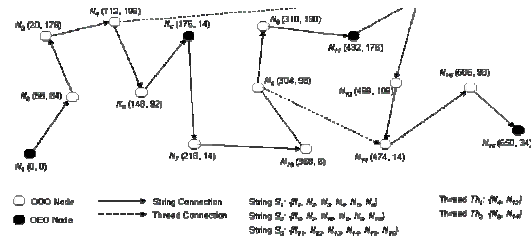


Fig. 8. Sample topology (approach 1)

The results shown in Fig. 5 through Fig. 7 for 16, 32 and 64 nodes respectively show that random threading outperforms maximal threading in terms of delay but has a higher convergence time (of approximately one-order of magnitude). The case of 64 nodes (Fig. 7) shows a peculiar phenomenon (with maximal threading) at high loads. Here, the average delay rises with the load but drops between $\lambda=600$ to $\lambda=800$. As we further increase the load the average delay rises again. This is due to the fact that at any given load, the number of collisions experienced by a packet from a node far from the CO is higher than a packet sent by a node close to the CO. As the load is increased, this phenomenon is more prominent. Since the average delay is computed on the packets that reach the CO, at high loads, this average delay is governed primarily by packets reaching the CO from closer nodes. Finally, shown in Fig. 8 is a graphical illustration of how random threading results in creation of strings and threads as part of light-mesh topology.

5. Conclusion

In this paper we have presented the light-mesh concept as an alternative technology to PON in access networks. Two heuristic algorithms for topology design keeping delay into consideration as well as adhering to other design constraints are presented and simulated.

6. References

1. A. Gumaste and S. Zheng, *IEEE Journ. of Lightwave Tech.*, Oct 2006. Vol. XX. No. YY.
2. A. Gumaste, I. Chlamtac and J. Jue, *IEEE Int'l Conf. on Commun.* Paris, France, June 2004.
3. R. Mochida, *IEEE Commun Mag.* Vol. 32. Feb 1995 pp 64-73
4. G. Kramer and G. Pesavento, *IEEE Communications Magazine*, vol.40, no.2, pp.66-73, Feb 2002