JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 33, 305-327 (2017)

DOI:10.1688/JISE.2017.33.2.3

Heuristic and Genetic Algorithms for Solving the Virtual Topology Design Problem on Elastic Optical Networks

DER-RONG DIN
Department of Computer Science and Information Engineering
National Changhua University of Education
Changhua City, 500 Taiwan
E-mail: deron@cc.ncue.edu.tw

Elastic optical networks (EONs) are considered as very promising architectures for future optical transport networks, since they efficiently use the spectrum resources and provide high bandwidth scalability and granularity. In this paper, the virtual topology design (VTD) problem on EON is considered. Given the physical network and the traffic demand matrix, the goal of VTD is to find the routing paths and the allocated frequency slots of the demand so that the objective cost can be minimized. In this paper, multiple-path routing is allowed and delay-variation between lightpaths for same demand is considered. In this paper, an integer linear programming (ILP) model is used to define the VTD problem on EONs. Moreover, a genetic algorithm (GA) and two heuristic algorithms are proposed to solve this problem. Simulations show that the proposed algorithms can achieve good results.

Keywords: elastic optical network (EON), virtual topology design, multi-path routing, delay-variation, heuristic algorithm, genetic algorithm

1. INTRODUCTION

Elastic optical networks (EONs), which employ *optical orthogonal frequency division multiplexing* (O-OFDM), have been proposed to scale the demands by efficiently utilizing the spectrum as they provide finer spectrum granularity and distance adaptive modulation formatting. The spectrum of a link on EONs is divided into small unit *frequency slots* (*FS*s) and necessary amount of consecutive frequency slots for a given data rate are assigned to support the connection request. Besides, more efficient spectrum allocation is achieved in these networks due to flexible grid and elastic line rates providing finer granularity. Therefore, EON performs higher scalability and higher flexibility than *wavelength division multiplexing* (WDM) network [1, 2]. EONs provide a super-channel connectivity for accommodating ultra-high capacity demands and a sub-wavelength granularity for low-rate transmissions [1, 2].

Due to the *spectrum continuity constraint* [1, 2], there is a tight coupling between spectrum allocation and routing of a demand. Consequently, *routing and spectrum assignment* (RSA) [1, 2] has emerged as the essential problem for spectrum management on EONs. A connection requiring a certain capacity should be satisfied by assigning a number of contiguous frequency slots. For a given connection request, the goal of the RSA problem is to find a lightpath on the network and assign the frequency slots. Due to the multi-layer nature of IP over EON, virtual topology design (VTD) is an inevitable and important issue in IP over EON to optimize network performance.

Received February 16, 2016; revised April 19 & June 19, 2016; accepted June 28, 2016.
Communicated by Xiaohong Jiang.

In this paper, the VTD problem on EONs is considered. Given the physical network and the traffic demand matrix, the goal of VTD is to find the routing paths and the allocated frequency slots of the demand such that the total cost of transponders can be minimized. In this paper, multiple-path routing is allowed and delay-variation between lightpaths for same demand is considered. Since the VTD problem is a hard problem. In this paper, an integer linear programming (ILP) model is used to define the VTD problem on EONs. In this paper, two heuristic algorithms are proposed, they are *Minimum Delay Path First* (MDPF) and *Maximal Allocates First* (MAF). Moreover, a genetic algorithm (GA) and two heuristic algorithms are proposed to solve this problem. In GA, a smart coding method of GA is constructed for the VTD problem on EONs. Then, several problem-depended heuristics for solving VTD problem are encoded into the crossover and mutation operations to improve the performance of the GA. Moreover, several heuristics are designed and used to speed up the convergence of the GA.

The contribution of this paper is the using the delay-variation multipath routing scheme to design the virtual topology of EONs. We take advantage of the load balancing potential of multipath and minimize the required TRs by two heuristic algorithms and a genetic algorithm. The proposed multipath routing algorithms are significantly better than a general one, which only performs a simple searching from a candidate set of (disjoint or non-disjoint) paths. Moreover, an ILP model is also proposed to define the problem and the AMPL/CPLEX package is used to find the optimal solution. The results showed that the proposed GA can get better results than MAF and MDPF algorithms on median size networks. The propose GA also can get near optimal results on the small size networks.

The rest of this paper is organized as follows. In Section 2, the assumptions and definitions of the VTD problem are described. In Section 3, the related works are described. In Sections 4 and 5, two heuristic algorithms and a genetic algorithm (GA) are proposed to solve this problem. Experimental results are given in Section 6. Finally, conclusions are given in Section 7.

2. PROBLEM DEFINITION

2.1 Assumptions

The assumptions of the VTD on EONs are given as follows.

- For each link, there is a fiber connecting the end-nodes and signal can be transmitted bidirectionally.
- All nodes in the network are equipped with *bandwidth variable wavelength cross-connects* (BV-WXC).
- For simplicity, the numbers of *FS*s provided by links are all equal.
- The bandwidth requirement between nodes can be transmitted by using multiple light-paths with same or different routes and numbers of frequency-slots.
- A guard band(GB) or guard subcarrier should be allocated between two lightpaths.
- The cost of the BV-WXC is known and fixed.
- The number of *FS*s that a BV-WXC switching node can support should be less than or equal to *F*.

2.2 Notations

- $G = (V, E, d)$: The physical topology of the network, where $V = \{v_1, v_2, \ldots, v_n\}$ is the set of nodes ($|V| = n$), $E = \{e_1, e_2, \ldots, e_m\}$ is the set of links ($|E| = m$), and $d(e_l)$ is delay of the link $e_l \in E$.
- $G^{new} = (V, LP)$: The virtual topology, which consists of the set of lightpaths and is denoted as $LP = \{lp_i = (s_i, d_i, r_i, t_i, fs_i) \mid v_s \in V, v_d \in V\}$, of the network. For the lightpath $lp_i = (s_i, d_i, r_i, t_i, fs_i)$, where $s_i \in V$ and $d_i \in V$ are the source and destination nodes of the lightpath, r_i is the actual route of the lightpath, t_i is the number of allocated *FS*s, and fs_i is the starting index of the *FS*s.
- B: The total number of *FS*s provided by each fiber.
- F: The maximum number of *FS*s can be used by a lightpath.
- $T = [T_{sd}]_{n \times n}$: The traffic matrix from the upper layers, where element T_{sd} represents the traffic demand from node $v_s \in V$ to node $v_d \in V$.
- Δ_{sd}:The maximum delay variation of the lightpaths between nodes $v_s \in V$ and $v_d \in V$.
- C: The cost of the transponder.
- $b_l(j)$: The bit-mask $b_l(j)$ of the link $e_l \in E$ ($j = 1, 2, \ldots, B$), which represents the status of the jth *FS* of the link e_l. If the jth *FS* is occupied, then $b_l(j) = 1$; otherwise, $b_l(j) = 0$.
- $sum(b_{el})$: The summation of the B-bit bit-mask of the link e_l for all $j = 1, 2, \ldots, B$, that is, $sum(b_{e_l}) = \sum_{j=1}^{B} b_l(j)$.

2.3 Constraints

On EONs, several constraints should be satisfied, these constraints are listed as follows.

- *Spectrum continuity constraint*: This constraint requires that, for a given lightpath, the same block of *FS*s of every link along the light-tree is allocated.
- *Subcarrier consecutiveness constraint*: Due to the very nature of O-OFDM, sub-carriers of the same data stream must be consecutive along the frequency domain. Hence, all frequency slots assigned in a link for a given request should be adjacent in the spectrum.
- Delay variation constraint: Normally, in multi-path provisioning, the maximal differential delay between lightpaths of same demand should less than or equal to an upper-bound [3].
- *Non-overlapping spectrum constraint*: Allocated *FS*s for paths must be separated by GBs in order to prevent interfering, *i.e.*, at least one frequency slot must be assigned as a GB between the frequency slot set of every lightpaths. Likewise, this constraint also implies that one frequency slot can be employed by only a single lightpath at a time.

For the required bandwidth T_{sd}, let N_{sd} be the number of *FS*s allocated for the demand T_{sd}. The value of N_{sd} can be computed by the formula (1), where C_f is the bandwidth provided by a single *FS*. If the request can be supported by a single lightpath, the minimal required number of *FS*s can be computed by $\lceil T_{sd}/C_f \rceil$ plus a guard band (GB). That is, if $\lceil T_{sd}/C_f \rceil \leq F$, then the second part of the formula (1) is equal to 1. If $\lceil T_{sd}/C_f \rceil$ is greater than the upper bound F, or a block $\lceil T_{sd}/C_f \rceil$ of continuous frequency-slots cannot be allocated on all links of the selected lightpath, the demand T_{sd} will be supported by establishing multiple lightpaths. In this case, more *GB*s will be introduced and the delay variation between selected lightpaths should be satisfied. The minimal required number of *FS* for the demand T_{sd} by using multipath routing can be computed by the formula:

$$N_{sd} = \lceil T_{sd}/C_f \rceil + \lceil \lceil T_{sd}/C_f \rceil / F \rceil \times GB. \tag{1}$$

2.4 Objective Cost

In this paper, given an EON physical network $G=(V, E, d)$ and a demand matrix T, the goal is to design a virtual topology G^{new} and assign *FS*s for lightpaths such that the total cost ($C\times TR$) of transponders can be minimized, where TR is the total number of transponders used to construct the virtual topology. Since the VTD is an NP-hard problem, in this paper, a genetic algorithm and two heuristic algorithms are proposed to solve it. The details of the proposed algorithm are described is following sections.

2.5 ILP Model

In this subsection, an ILP formulation is used to describe the VTD problem on EON. For each connection request (v_s, v_d, N_{sd}) from source v_s to destination v_d for N_{sd} *FS*s, a set of candidate paths PS_{sd} from v_s to v_d is pre-computed. Paths in PS_{sd} are sorted in increasing order according to the delay of paths.

Notations:
- p^k_{sd}: the kth minimal delay path in PS_{sd}, $1 \le k \le |PS_{sd}|$.
- x^l_{sdk}: equal 1 if path p^k_{sd} uses link $e_l \in E$, 0 otherwise.
- $delay(p^k_{sd})$: the delay of the path p^k_{sd}.
- K_{sd}: the number of paths in PS_{sd}, $K_{sd} = |PS_{sd}|$.
- P: the set $P = \cup_{\forall(s,d)} PS_{sd}$.
- $|P|$: the number of paths in set P.
- g: the number of guard subcarriers.

Variables:
- y_{sdk}: equal 1 if path p^k_{sd} is selected to route the demand (v_s, v_d, N_{sd}), 0 otherwise.
- c^w_{sdk}: boolean variable that denotes if path p^k_{sd} uses frequency slot w, *i.e.*, $c^w_{sdk} = 1$ if path p^k_{sd} uses frequency slot $w(1 \le w \le B)$; 0 otherwise.
- u^w: boolean variable that equals 1 if there exists a spectrum path using frequency slot w, 0 otherwise.

Objective:

$$Minimize: \quad C\times TR = C\times \sum_{\forall v_s\in V}\sum_{\forall v_d\in V}\sum_{k\in[1,K_{sd}]} y_{sdk}. \tag{2}$$

Eq. (2) obtains the total cost of transponders.

Constraints:
- Frequency slots constraint:

$$w \times u^w \leq B,\ \forall w \in [1, B], \tag{3}$$

$$\sum_{\forall v_s \in V} \sum_{\forall v_d \in V} \sum_{k \in [1, K_{sd}]} c^w_{sdk} \leq u^w,\ \forall w \in [1, B]. \tag{4}$$

Eq. (3) obtains the maximum index of the allocated frequency slots should be less than or equal to B. Eq. (4) ensures that u^w equals 1 when at least one path utilizes frequency slot w.

• Capacity allocation constraints:

$$\sum_{w \in [1, B]} \sum_{k \in [1, K_{sd}]} y_{sdk} c^w_{sdk} = N_{sd}, \forall (v_s, v_d) \in V \times V \tag{5}$$

$$\sum_{w \in [1, B]} y_{sdk} c^w_{sdk} \leq F, \forall (v_s, v_d) \in V \times V, k \in [1, K_{sd}] \tag{6}$$

Eq. (5) ensures that the total number of frequency slots allocated on all the routing paths of a connection (v_s, v_d) is equal to the required frequency slots N_{sd}. Eq. (6) ensures that the total number of frequency slots allocated on the routing path of a connection (v_s, v_d) is less than or equal to F.

• Non-overlapping spectrum constraints:

$$\sum_{\forall v_s \in V} \sum_{\forall v_d \in V} \sum_{k \in [1, K_{sd}]} y_{sdk} c^w_{sdk} x^l_{sdk} \leq 1, \forall e_l \in E, w \in [1, B]. \tag{7}$$

Eq. (7) ensures that each frequency slot on a link can be utilized by at most one spectrum path.

• Delay-variation constraints:

$$\max_{k \in [1, K_{sd}]} \{y_{sdk} \times delay(p^k_{sd})\} - \min_{k \in [1, K_{sd}]} \{y_{sdk} \times delay(p^k_{sd})\} \leq \Delta_{sd},\ \forall (v_s, v_d) \in V \times V. \tag{8}$$

Eq. (8) ensures that the differential delay between max-delay and min-delay paths used for a connection can not exceed Δ_{sd}.

• Guard subcarrier constraints:

$$2g\,|P|\,(c^w_{sdk} x^l_{sdk} - 1) + \sum_{w' \in [\max(1, w-g), \min(B, w+g)]} c^{w'}_{s'd'k'} x^l_{s'd'k'} \leq 0,$$
$$\forall w \in [1, B],\ e_l \in E,\ p_{sd}^k | x_{sdk}^l = 1,\ s \neq s',\ d \neq d',\ k \neq k'. \tag{9}$$

When two adjacent paths share a link, they have to be separated by g guard subcarriers. Eq. (9) ensures that if path p^k_{sd} uses subcarrier w on link e_l, then frequency slots [w–g, w+g] on e_l can't be occupied by other paths. Specifically, if path p^k_{sd} contains link e_l and uses frequency slot w, then the first term in the left hand side of the equation is 0. This forces the second term to be 0, which ensures that all the other paths that also use link e_l cannot occupy frequency slots [w–g, w+g] [4].

• Subcarriers connectiveness constraint:

$$(c_{sdk}^{w} + c_{sdk}^{w+1} - 1)(-B) \geq \sum_{w' \in [w+2,B]} c_{sdk}^{w'}, \forall w \in [1,B], k \in [1,K_{sd}], \forall (v_s, v_d) \in V \times V \quad (10)$$

Eq. (10) ensures that if path p^{k}_{sd} utilizes frequency slot w and doesn't utilize frequency slot w+1, it cannot occupy any frequency slot with index $w' \in [w+2, B]$. That is, a spectrum path has to be allocated a set of contiguous frequency slots.

3. RELATED WORKS

3.1 RSA Problem

Several variants of the RSA problem on EON have been studied in the literature, and take into account various design aspects. In [5], the authors proposed an ILP model to minimize the assigned spectrum with a static traffic matrix. In [6], the authors presented a comprehensive study on the RSA problem in SLICE networks and proved its NP-hardness. The paper also proposed an ILP model and two heuristics to find spectrum paths using minimal number of sub-carriers. Since the problem is intractable, therefore, an array of heuristic algorithms has been put forward to obtain reasonably good solutions efficiently [7]. Two-step RSA heuristic decomposes the problem into two subproblems, the routing problem and the spectrum assignment subproblem, which are then solved sequentially. In routing problem, a number of candidate physical paths for each source-destination node pair is computed and arranged in a path table. In the spectrum assignment subproblem, for the selected path from the path table, a spectrum allocation policy is used to assign a feasible path and set of contiguous *FS*s, if they exist, for the requested demand [7].

3.2 LG-RSA

On the other hand, one-step RSA heuristic solves the two subproblems simultaneously. In [8], authors incorporated a *layered graph approach* to design integrated *multicast-capable routing and spectrum assignment* algorithm for achieving efficient all-optical multicast on EONs. This approach can also be used to solve the RSA problem for a unicast on EON networks. The procedure is denoted as *layered graph RSA* (*LG-RSA*).

Recall that $b_l(j)$ be the status of the link $e_l \in E$ on *j*th *FS* which is a binary indicator; $b_l(j)$=0 means that the *j*th frequency slot of link e_l is free; otherwise, $b_l(j)$=1. For the network G=(V, E, d) and the given connection request (v_s, v_d, C), where C is the number of required *FS*s; the *layered graph*(*LG*) is a set of graphs LG={G^j(V^j, E^j), j=1, 2, …, B-C+1}. For the graph G^j(V^j, E^j), where V^j=V and E^j ={e_l|B_{lj}= $\sum_{z=j}^{j+C-1} b_l(z) = 0$ and $e_l \in E$}. On G^j(V^j, E^j), the link $e_l \in E^j$ represents that there are free continuous *FS*s within *j*th to (j+C-1)th on the link $e_l \in E$ for the connection request. If a lightpath can be found on G^j(V^j, E^j), it means that the lightpath can be allocated to network G and the starting index of *FS*s is *j* and for C continuous *FS*s. The details of the LG-RSA are described in Algorithm 1.

Algorithm 1 : LG-RSA

```
1: Input: $G(V, E, d)$, $(v_s, v_d, C)$;
```

```
2: Output: routing path and the starting index of allocated FSs;
3: for all ($j$ =1, 2, ..., $B + C - 1$) do
4:     Construct $G^j(V^j, E^j)$, where $V^j = V$ and $E^j = \{e_l \mid B_{lj} = \sum_{z=j}^{j+C-1} b_l(z) = 0$ and $e_l \in E\}$.
5:     Perform Dijkstra shortest path algorithm on $G^j(V^j, E^j)$ to find the shortest path $p$ from $v_s$
       to $v_d$.
6:     if (path $p$ can be found) then
7:         return path $p$ and starting index $j$;
8:     end if
9: end for
10: return path cannot be found;
```

3.3 Multi-Path Routing

Multi-path routing scheme has already demonstrated improved network performance in WDM (wavelength division multiplexing) networks [9]. Meanwhile, it has been a consensus that they need to address the differential delay between routing paths properly, due to the QoS constraints from network applications. In [3], the authors proposed a novel *dynamic multi-path service provisioning algorithm* that is specifically designed for EONs and considers the differential delay constraint. The proposed algorithm tried to set up dynamic connections with single-path routing in a best-effort manner. If this cannot be done, it used an auxiliary-graph based approach to calculate a multi-path provisioning scheme based on two parameters, *i.e.*, the differential delay upper-bound and the bandwidth allocation granularity. Simulation results demonstrate that the multi-path scheme outperforms two single-path ones by providing lower bandwidth blocking probability and achieves 10-18% improvement on average network throughput. In [10], the authors first attempted to address the spectrum fragmentation issue with multipath routing. The authors proposed a new method to effectively alleviate the spectrum fragmentation issue by multipath routing. An ILP based optimization model for multipath routing and spectrum assignment was proposed in [10]. They also proposed heuristic algorithms that are applicable in large networks where the ILP model may become infeasible due to the complexity issue.

In [11] the authors proposed a novel parallel transmission framework designed for EONs to support high-speed Ethernet services. The authors formulated an optimization ILP model, with consideration of various constraints, including spectrum fragmentation, differential delay and guard-band constraints. A heuristic RSA algorithm has been proposed to solve the parallel transmission problem, which was divided into two subproblems, *i.e.*, computation of fiber-level path(s) and spectrum assignment. The first phase of the proposed heuristic algorithm is to compute a set of fiber-level paths. The second phase selects suitable paths and finds the assigned *FS*s.

In [4], the authors considered the survival multipath routing and spectrum allocation problem on EON, they defined the *Static Survivable Multipath Routing and Spectrum Allocation* (SM-RSA) problem, which aims to accommodate a given set of demands with minimum utilized spectrum. They showed that the static SMRSA problem is NP-hard and provided an ILP formulation for it. Also, an efficient heuristic algorithm has been proposed to solve this problem.

3.4 VTD on WDM Network and EONs

For the WDM networks, the VTD problem can be divided into four sub-problems, and one of these sub-problems is the famous *routing and wavelength assignment* (RWA) problem which has been proven NP-complete [12]. The WDM VTD problem can thus be formulated as an optimization problem where the objective function of the problem is to minimize the average weighted-hop distance or minimize the network-wide average packet delay [12]. Researchers of optical networks have attempted such problems of designing virtual topologies and have obtained solutions using heuristic methods in polynomial time.

The previous related work of VTD has shown notable approaches and interesting conclusions [13] in WDM networks, with the objectives of minimizing the cost of transmission and switching equipment. In [14], not only the cost, but also the power consumption should be considered to achieve a balance in VTD of EONs. The authors studied an *energy-efficient virtual topology design* (EE-VTD) scheme on EONs, and formulated a mixed integer linear programming (MILP) model to exactly describe the problem and find the optimal solution [14]. The authors also proposed a heuristic algorithm for this scheme.

3.5 Genetic Algorithm

For an NP-hard problem, designing an algorithm to find an optimal solution is impractical, due to the exponential growth in execution time. *Genetic Algorithms* (*GAs*) proposed by John Holland [15] have been trusted as a class of general-purpose search strategies that strike a reasonable balance between exploration and exploitation. GAs have been constructed as robust stochastic search algorithms for various optimization problems. GAs are implemented as computer simulations to gain better solutions in a population of *chromosomes* of candidate solutions to an optimization problem. If the algorithm has terminated due to the maximum number of generations, a satisfactory solution may or may not have been reached. GAs find applications in computer science, engineering, and other fields [15]. The structure of the GA is shown in Algorithm 2 [15].

Algorithm 2 : Genetic Algorithm (GA)

```
1: generate an initial population;
2: evaluate the fitness and penalty function of each individual in the population;
3: while (not terminated) do
4:     select individuals form parent to reproduce;
5:     generate offspring through crossover and mutation;
6:     evaluate the individual fitness of the offspring;
7:     replace the worst ranked part of the population with offspring;
8: end while
```

GAs search by exploiting information sampled from different regions of the solution space. The combination of crossover and mutation helps GA escape from local optima. These properties of GA provide a better global search methodology for the virtual topology design problem. In this paper, a GA is proposed to find a good virtual topology on EONs so as to minimize the objective cost. The performance of the proposed algorithm is evaluated in terms of the costs associated with the transponder they construct.

4. HEURISTIC ALGORITHMS FOR VTD ON EON

Two heuristic algorithms are proposed and described in this section to solve the VTD problem on EONs. For the multiple-path routing, two path selecting schemes are used, they are *Minimum Delay Path First* (MDPF) and *Maximal Allocates First* (MAF). To select multiple paths to route the demand, the delay variation constraint of selected paths should be satisfied. The details of these two algorithms will be described in the following subsections. First, the *Preprocessing Algorithm* is presented in Section 4.1. Then, the details of the proposed algorithms are described in Sections 4.2 and 4.3.

4.1 Preprocessing Algorithm

All non-zero demands N_{sd}, $\forall v_s$, $v_d \in V$, are sorted in descending order according to the value of N_{sd} to form a set of requests $R=\{(v_s, v_d, N_{sd}, \Delta_{sd}) \mid \forall v_s, v_d \in V\}$. For the multi-path routing, first, the K-shortest paths algorithm [16] is used to find a set of candidate paths for the node-pair (v_s, v_d). The set of candidate paths of the node-pair (v_s, v_d) is denoted as $PS_{sd} = \{p^k{}_{sd}, k=1, 2, \ldots, K\}$, and the delay of the path $p^k{}_{sd}$ is denoted as $delay(p^k{}_{sd})$. The details of the Preprocessing Algorithm are described in Algorithm 3.

Algorithm 3 : Preprocessing Algorithm

```
1: Input: $G(V, E, d)$, traffic matrix $N_{sd}$, $\Delta_{sd}$, $\forall v_s, v_d \in V$;
2: Output: $R$, $PS_{sd}$;
3: All non-zero element $N_{sd}$, $\forall v_s, v_d \in V$ are sorted in descending order according to the value of
   $N_{sd}$ to form a set of requests $R = \{(v_s, v_d, N_{sd}, \Delta_{sd}) \mid \forall v_s, v_d \in V\}$.
4: For each pair of nodes $v_s$, $v_d$, perform $K$-shortest paths algorithm [16] to find a set of candidate
   paths from node $v_s$ to node $v_d$ on graph $G$. The set of paths is denoted as $PS_{sd} = \{p^k_{sd}, k =
   1, 2, ..., K\}$. The delay of the path $p^k_{sd}$ is denoted as $delay(p^k_{sd})$.
```

4.2 Minimum Delay Path First (MDPF)

In the subsection, the details of the Minimum Delay Path First (MDPF) algorithm are described (shown in Algorithm 4). After performing the Preprocessing Algorithm, if the set R is nonempty, then these requests in R are processed one-by-one in order. For each connection request $(v_s, v_d, N_{sd}, \Delta_{sd})$ selected from the set R, if $N_{sd} \le F$, then a single path is found by performing the *LG-RSA* algorithm on network G to route the demand. If a single path with suitable *FS*s can be found, then the lightpath is allocated. Otherwise, the multi-path routing scheme is applied to find a set of lightpaths with delay-variation constraint, the **MDPF Path Selecting Algorithm** is performed. If lightpaths for the demand cannot be satisfied, then an error is reported. It may be the reason that lacks of *FS*s or candidate paths in PS_{sd}.

Algorithm 4 : Minimum Delay Path First (MDPF)

```
1: Input: $G(V, E, d)$, $N_{sd}$, $\Delta_{sd}$, $\forall v_s, v_d \in V$;
2: Output: $G^{new}(V, LP)$;
3: $LP = \emptyset$, $TR = 0$.
4: Performing the Preprocessing Algorithm to generate set $R$ and $PS_{sd}, \forall v_s, v_d \in V$.
5: while ($R \neq \emptyset$) do
6:    {
7:    Select and remove a request $(v_s, v_d, N_{sd}, \Delta_{sd})$ from $R$.
8:    if ($N_{sd} \le F$) then
9:       //Try the single path routing scheme
10:      Perform the LG-RSA on $G$ to find a single path $p$ for the request $(v_s, v_d, N_{sd})$.
11:      if (path $p$ can be found) then
12:         Establish the lightpath $p$, allocate FSs, add $p$ to $LP$, update $G$ and $TR$.
13:         continue;
14:      end if
15:     end if
16:     Performing MDPF Path Selecting Algorithm to find a set $PQ$ of paths for request
        $(v_s, v_d, N_{sd}, \Delta_{sd})$ with delay variation constraint.
17:     if ($PQ$ can be found) then
18:        Establish the lightpaths in $PQ$, add paths in $PQ$ to $LP$, update $G$ and $TR$.
19:     else
20:        Stop and report an ERROR! //for not enough FSs or candidate paths.
21:     end if
22:     }
23: end while
24: return $G^{new}(V, LP)$.
```

4.2.1 MDPF path selecting algorithm

In the following, the details of the MDPF Path Selecting Algorithm are described. Let the path queue *PQ* be the set of paths which are selected as the lightpaths for routing the demand of nodes v_s and v_d. The set *PQ* is initially set as an empty set and $PS=PS_{sd}$ is the set of candidate paths. Let *min_delay*(*PQ*) be the minimal delay of the paths in the set *PQ*. In the MDPF, for the first path, it is clear that there is no need to consider the delay variation constraint, the path *p* in *PS* with minimum delay is checked whether it can be allocated to the current network. Let *FS*(*p*) denote the maximum allocatable *FS*s of the path *p* to the current network. If the path *p* cannot be allocated to the network (*i.e.*, $FS(p)=0$), the path *p* is removed from *PS*. Otherwise, if $N_{sd} \leq FS(p)$ and $N_{sd} \leq F$, the path *p* is allocated on network and added to *PQ*, then return. If $FS(p)<N_{sd}$, then the path *p* is added to *PQ* and $min\{N_{sd}, FS(p), F\}$ *FS*s are temporarily allocated on the current network. Then, N_{sd} is updated by subtracting maximal allocatable frequency slots, the network and the value of *min_delay*(*PQ*) are updated accordingly.

For the follow-up paths of the demand, the *path finding process* is performed and described as follows. When the path queue *PQ* is not empty, the minimum delay path *p* in *PS* is checked whether it can satisfy the delay variation constraint ($delay(p)-min\_delay(PQ) \leq \Delta_{sd}$) and can be allocated to the current network. If the *PS* is empty, then report an error; the network planner should increase the number of *FS*s or the value of $|PS_{sd}|$. If the path *p* can be allocated, the path *p* is added to the set *PQ* and the demand N_{sd} is updated by subtracting and allocate $min\{N_{sd}, FS(p), F\}$ *FS*s for the path temporarily. And then, the current network is also updated accordingly. If N_{sd} is equal to 0, then the set *PQ* is returned.

If path *p* cannot be found due to the delay variation constraint cannot be satisfied, then the paths *p*′ in *PQ* which with $delay(p)-delay(p')>\Delta_{sd}$ are removed from *PQ*. Then, the network is updated by restoring the *FS*s allocated to the paths *p*′ and the value of N_{sd} is restored by adding the demand allocated on the selected paths. Moreover, those paths whose delay are less than or equal to delay(*p*′) are removed from the set *PS*, since these paths cannot be selected as the lightpaths. The value of *min_delay*(*PQ*) is also updated accordingly. If N_{sd} is greater than 0, then the path finding process is repeated. If path *p* cannot be found due to the required *FS*s cannot be allocated, then the path is temporarily removed from the set PS_{sd}. The details of the MDPF Path Selecting Algorithm are described in Algorithm 5.

Consider the example shown in Table 1, for the node-pair v_s and v_d, K(=7) shortest paths are computed and listed in the first column of the table. The delay of each path and the maximal allocatable *FS*s is listed in the second and third column, respectively. CS_{sd} is

Algorithm 5 : MDPF Path Selecting Algorithm

```
1: Input: $G(V,E,d)$, request $(v_s, v_d, N_{sd}, \Delta_{sd})$, $PS_{sd}$;
2: Output: $PQ$;
3: Construct $PQ = \emptyset$, $PS = PS_{sd}$, $min\_delay(PQ) = 0$;
4: while ($N_{sd} > 0$) do
5:   {
6:   if ($PQ == \emptyset$) then
7:     { // find first path, there is no need to check the delay variation constraint
8:     Check the minimum-delay path $p$ in $PS$, whether it can be allocated to the current network
       or not.
9:     if (the path $p$ cannot be allocated) then
10:      Remove $p$ from $PS$.
11:    else
12:      {
13:      Let $FS(p)$ be the maximum allocatable number of FSs for the path $p$.
14:      Temporarily allocate $\min\{N_{sd}, FS(p), F\}$ FSs to the lightpath $p$.
15:      Add path $p$ to $PQ$, update $min\_delay(PQ)$, $N_{sd}$ and $G$.
16:      }
17:    end if
18:    }
19:  else if ($PS == \emptyset$) then
20:    return error;
21:  else
22:    {
23:    Check the minimum-delay path $p$ in $PS$, whether it can be added to $PQ$ (can satisfy the
       delay constraint $delay(p) - min\_delay(PQ) \le \Delta_{sd}$) and can be allocated to the current
       network or not.
24:    if (path $p$ cannot be allocated due to lack of FSs) then
25:      Remove $p$ from $PS$.
26:    else if (path $p$ can be allocated) then
27:      {
28:      Let $FS(p)$ be the maximum allocatable number of FSs for the path $p$.
29:      Temporarily allocate $\min\{N_{sd}, FS(p), F\}$ FSs to the lightpath $p$.
30:      Add path $p$ to $PQ$, update $N_{sd}$ and $G$.
31:      }
32:    else if (path $p$ cannot be allocated due to delay variation constraint) then
33:      {
34:      Remove all paths $p'$ from $PQ$, where $p'$ satisfies $delay(p) - delay(p') > \Delta_{sd}$.
35:      Release all FSs allocated to these paths $p'$.
36:      Update $N_{sd}$ by adding the allocated FSs and update $G$.
37:      }
38:    end if
39:    }
40:  end if
41:  }
42: end while
43: return $PQ$;
```

Table 1. Example of multi-path routing.

path p^k_{sd}	$delay(p^k_{sd})$	$FS(p^k_{sd})$	$u(k)$	CS_{sd}	$\sum_{\forall p^k_{sd} \in CS_{sd}} FS(p^k_{sd})$
p^1_{sd}	2	4	5	$\{p^1_{sd}, \ldots, p^5_{sd}\}$	14
p^2_{sd}	2	1	5	$\{p^2_{sd}, \ldots, p^5_{sd}\}$	10
p^3_{sd}	3	3	6	$\{p^3_{sd}, \ldots, p^6_{sd}\}$	10
p^4_{sd}	4	5	7	$\{p^4_{sd}, \ldots, p^7_{sd}\}$	10
p^5_{sd}	4	1	7	$\{p^5_{sd}, \ldots, p^7_{sd}\}$	5
p^6_{sd}	5	1	7	$\{p^6_{sd}, \ldots, p^7_{sd}\}$	4
p^7_{sd}	6	3	7	$\{p^7_{sd}\}$	3

the set of candidate paths which satisfy the delay variation constraint and $\sum_{\forall p^k_{sd} \in CS_{sd}} FS$ (p^k_{sd}) is the maximal possible number of *FS*s supported by the set CS_{sd}. For the demand

Δ_{sd}=2, N_{sd}=8 and F=4, the path p^1_{sd} is selected and added to PQ. Then N_{sd} is updated to N_{sd}=8–4=4. If paths p^2_{sd} and p^3_{sd} cannot be allocated after the path p^1_{sd} is allocated (due to lack of FSs), then p^4_{sd} is selected and allocated.

4.3 Maximal Allocates First (MAF)

In this subsection, the details of the Maximal Allocates First (MAF) algorithm are described (shown in Algorithm 6). After performing the Preprocessing Algorithm, if the R is nonempty, then these requests in R are processed one-by-one in order. For each connection request (v_s, v_d, N_{sd}, Δ_{sd}) selected from the set R, if $N_{sd} \leq F$, then a single path is found by performing the *LG-RSA* algorithm on network G to route the demand. If a single path with suitable FSs can be found, then the FSs are allocated. Otherwise, the multi-path routing scheme is applied and the **MAF Path Selecting Algorithm** (shown in Algorithm 7) is performed. If lightpaths for the demand cannot be satisfied, then an error is reported. It may be the reason that lacks of FSs or the candidate paths in PS_{sd}.

Algorithm 6 : Maximal Allocates First (MAF)

```
 1: Input: $G(V,E,d)$, $N_{sd}$, $\Delta_{sd}$, $\forall v_s, v_d \in V$;
 2: Output: $G^{new}(V,LP)$;
 3: $LP = \emptyset$, $TR = 0$;
 4: Performing Preprocessing Algorithm to generate set $R$ and $PS_{sd}, \forall v_s, v_d \in V$.
 5: while ($R \neq \emptyset$) do
 6:   {
 7:   Select and remove a request $(v_s, v_d, N_{sd}, \Delta_{sd})$ from $R$.
 8:   if ($N_{sd} \leq F$) then
 9:     //Try the single path routing scheme
10:     Perform the LG-RSA on $G$ to find a single path $p$ for the request $(v_s, v_d, N_{sd})$.
11:     if (path $p$ can be found) then
12:       Establish the lightpath $p$, allocate FSs, add $p$ to $LP$, update $G$ and $TR$.
13:       continue;
14:     end if
15:   end if
16:   Performing MAF Path Selecting Algorithm to find a set $PQ$ of paths for the request
      $(v_s, v_d, N_{sd}, \Delta_{sd})$ with delay variation constraint.
17:   if ($PQ$ can be found) then
18:     Establish the lightpaths in $PQ$, add paths in $PQ$ to $LP$, update $G$ and $TR$.
19:   else
20:     Stop and report an ERROR! //not enough FSs or candidate paths.
21:   end if
22:   }
23: end while
24: return $G^{new}(V,LP)$;
```

4.3.1 MAF path selecting algorithm

In the following, the details of the MAF Path Selecting Algorithm are described. For the MAF scheme, the path p with maximal allocatable FSs in the set PS_{sd} is selected as the first lightpath; if tie, the path with minimum delay is selected. For the demand N_{sd}, the candidate paths are sorted in descending order according to the maximal number of allocatable FSs on the current network and put into priority queue PQ. Paths in the PQ are examined one-by-one to find the set of paths. Initially, the selected paths are stored in the *selQ*, D_{max} and D_{min} are the maximal and minimal delay of the paths in the *selQ*.

Algorithm 7 : MAF Path Selecting Algorithm

```
1: Input: $G(V, E, d)$, request $(v_s, v_d, N_{sd}, \Delta_{sd})$, $PS_{sd}$;
2: Output: $selQ$;
3: Construct $PQ = \emptyset$, $selQ = \emptyset$, $G' = G$;
4: Let $D_{max}$ and $D_{min}$ be the minimal and maximal delay of paths in $selQ$. Initially, $D_{max} = -1$
   and $D_{min} = \infty$.
5: All paths in $PS_{sd}$ are sorted according to the allocatable FSs of the lightpath on the current
   network in descending order and added to $PQ$.
6: while $(N_{sd} > 0)$ do
7:    {
8:    while $(PQ \neq \emptyset)$ do
9:       {
10:      Select a path $p$ from $PQ$ and find the maximal allocatable FSs on the current network.
11:      if (path $p$ cannot be allocated to the network) then
12:         Remove path $p$ from $PQ$.
13:      else
14:         {
15:         // path $p$ can be allocated $FS(p)$ FSs on the current network
16:         if $((selQ == \emptyset)$ or (add path $p$ to $selQ$ can satisfy the delay variation constraint))
            then
17:            {
18:            Temporarily allocate $\min\{N_{sd}, FS(p), F\}$ FSs for the demand.
19:            Add path $p$ to $selQ$, update $N_{sd}$, $G$, $D_{max}$, and $D_{min}$.
20:            if $(N_{sd} == 0)$ return paths in $selQ$ end if.
21:            }
22:         else
23:            {
24:            //try to update the $selQ$ by releasing some paths in $selQ$
25:            Release resources allocated to paths which violates delay constraint after adding
               path $p$ to $selQ$.
26:            Restore the value of $N_{sd}$ from paths in $selQ$.
27:            Find the maximal allocatable $FS(p)$ FSs on the current network for path $p$.
28:            Temporarily allocate $\min\{N_{sd}, FS(p), F\}$ FSs for the demand.
29:            Add path $p$ to $selQ$, update $N_{sd}$, $G$, $D_{max}$, and $D_{min}$.
30:            if $(N_{sd} == 0)$ return paths in $selQ$ end if.
31:            }
32:         end if
33:         }
34:      end if
35:      }
36:   end while
37:   Release all resources allocated to paths in $selQ$.
38:   Restore the value of $N_{sd}$ from paths in $selQ$.
39:   return ERROR;
40:   }
41: end while
```

First, a path p is selected from PQ, the maximal allocatable FSs of the path p is found for the demand. If path p cannot be allocated on the current network, then path p is removed from PQ and the next path in PQ is examined. Otherwise, if path p is the first path for the demand or adding path to $selQ$ can satisfy the delay variation constraint, the path p is added to $selQ$ and $min\{N_{sd}, FS(p), F\}$ FSs are allocated to the path p. Then, the values of N_{sd}, D_{max}, D_{min}, and the network G' are updated. If all the demand can be allocated (*i.e.*, N_{sd} is equal to 0), then the set of paths in $selQ$ is returned.

If adding path p to $selQ$ violates the delay variation constraint, then the path p is added to $selQ$ and then those paths which violate the delay variation constraint (not the path p) are removed from $selQ$. The allocated resources of removed paths are released, the values of D_{max} and D_{min} and the network are updated accordingly. The $FS(p)$ is found for the selected path p, and $min\{N_{sd}, FS(p), F\}$ FSs are allocated to the path p. Finally, the N_{sd} is updated. If all the demand can be allocated, then the set of paths in *selQ* is returned. If all paths in *PQ* are examined, but the demand N_{sd} cannot be supported by the current network (N_{sd}>0), then return an error. The details of the MAF Path Selecting Algorithm are shown in Algorithm 7.

Consider the example shown in Table 1 for the demand N_{sd}=6 and Δ_{sd}=2, the path p^4_{sd} with maximal *FS*s in the set PS_{sd} is selected as the first lightpath. The p^4_{sd} is added to *selQ* and D_{max}=4 and D_{min}=4. Then, the next candidate path is p^1_{sd}, since adding path p^1_{sd} to *selQ* can satisfy the delay variation constraint. If $FS(p^1_{sd})$ is 4 (not affected by the allocation of path p^4_{sd}), then path p^1_{sd} is added to *selQ* and return.

5. GENETIC ALGORITHM

In this section, the details of the proposed genetic algorithm (GA) developed to solve the VTD problem on EONs are described. The development of GA requires: (1) an encoding scheme; (2) a population generation method; (3) crossover operators; (4) mutation operators; (5) a fitness function; (6) a replacement strategy; and (7) termination rules.

5.1 Preprocessing

First, for each source-destination pair of the network, *K* shortest paths are computed and stored as the set (path table) of candidate lightpaths by using the algorithm proposed in [16]. The path tables for all node-pairs can be found in $O(mn+n^2logn+Kn^2)$ [16]. For the node pair v_s, $v_d \in V$, the candidate lightpaths are stored in the *path-table* PS_{sd}. Paths in $PS_{sd}=\{p^1_{sd}, p^2_{sd}, \ldots, p^K_{sd}\}$ are sorted and indexed in increasing order according to the delay of the paths. An integers $u(k)$ is computed for the path $p^k_{sd} \in PS_{sd}$, k = 1, 2, …, K; $u(k)$ denotes the ending index of the path p^k_{sd}, as the path p^k_{sd} is selected as the starting index of the set of candidate paths. The candidate set of paths is denoted as $CS_{sd}=\{p^k_{sd}, p^{k+1}_{sd}, \ldots, p^{u(k)}_{sd}\} \subseteq PS_{sd}$ and with constraint $delay(p^{u(k)}_{sd})-delay(p^k_{sd}) \le \Delta_{sd}$. It means that all paths in CS_{sd} can be selected as the routing paths for the traffic demand between nodes v_s and v_d, and can satisfy the delay variation constraint. Consider the example shown in Table 1, if k=1, then $u(k)$=5 and $CS_{sd}=\{p^1_{sd}, p^2_{sd}, \ldots, p^5_{sd}\}$.

5.2 Chromosomal Coding

Since the design of the virtual topology requires determining the lightpaths of node-pairs, the routing paths of the lightpaths on the physical network, and the assigned *FS*s of the lightpaths. When a pair of nodes is selected to establish lightpaths, the information about the actual route and assigned *FS*s of each lightpath should be determined.

In the proposed GA, the virtual topology of EON is initially represented by a ($n \times n$) two-dimension array *matrix chromosome* (denoted as *MC*), which is randomly generated. The element $MC_{sd} \in MC$ represents the established lightpaths for the demand N_{sd}, and it consists of a list of indices of selected paths, the number of assigned *FS*s, and the starting indices of assigned frequency slots. For each path in the list, the path index, starting index of *FS*s and the number of allocated *FS*s are kept.

5.3 Initial Population

By using *MC* to represent the virtual topology, it should ensure that the virtual topology is constraint-satisfied. In other words, the *MC* represents a candidate virtual topology of the EON. However, if the contents of the *MC* are randomly generated, then the virtual topology may be infeasible with respect to the constraints on the delay variation between lightpaths. Thus, in GA, only virtual topologies which satisfy the constraint will be generated by performing the *Random Initialization Algorithm* (*RIA*). The details of RIA are described in Algorithm 8.

Algorithm 8 Random Initialization Algorithm (RIA)

```
Input: $G(V, E, d)$, $N = [N_{sd}]_{n \times n}$, $\Delta_{sd}, \forall v_s, v_d \in V$,
Output : $MC$.
All non-zero elements $N_{sd}$ in $N$ are sorted in descending order according to the demand of the
  node-pair and add to the priority queue $Q$.
while ($Q \neq \emptyset$) do
  {
  Select and remove a node-pair (say $v_s$, $v_d$) from $Q$.
  Random select an integer $k \in \{1, 2, ..., \lceil K/2 \rceil\}$
  RS: // reselect the starting index
  Let $CS_{sd}$ be $\{p_{sd}^{k}, p_{sd}^{k+1}, ..., p_{sd}^{u(k)}\}$ and all paths in $CS_{sd}$ are valid.
  while ($N_{sd} > 0$) do
    {
    while (there is at least a valid path in $CS_{sd}$) do
      {
      Random select a valid index $i$ within $[k, u(k)]$.
      Find the maximum allocatable $FS(p_{sd}^{i})$ FSs for the lightpath $p_{sd}^{i}$ on the network $G$.
      if ($FS(p_{sd}^{i}) > 0$) then
        Temporarily allocate $\min\{N_{sd}, F, FS(p_{sd}^{i})\}$ FSs for the selected lightpath $p_{sd}^{i}$.
        Add path $p_{sd}^{i}$, allocated FSs, and starting index of FSs to the list of $MC_{sd}$ .
        Update $N_{sd}$ and $G$.
      else
        Mark the lightpath $p_{sd}^{i}$ as invalid.
      end if
      }
    end while
    if (($N_{sd} > 0$) and ($k \geq 2$)) then
      Restore $G$ and reset the list of $MC_{sd}$.
      Let $k = \lceil k/2 \rceil$ and goto RS.
    else
      return ERROR.
    end if
    }
  end while
  }
end while
```

5.4 Crossovers

Two crossover operators are considered in the proposed GA:

- *Row-oriented crossover* (ROC): Two MCs (say MC^1 and MC^2) are randomly selected for crossover from the previous generation, and then two integers i and j are generated randomly in the range [1, n]. These numbers are used as the crossover points. The (i+1)~n rows of the MC^1 and MC^2 are switched. The example is shown in Fig. 1 (b).

- *Block-oriented crossover* (BOC): Two MCs (say MC^1 and MC^2) are randomly selected for crossover from previous generation and then two integers i and j are generated randomly in the range [1, n]. These numbers are used as the crossover points. The block (1, 1)–(i, j) of the MC^1 and MC^2 are switched. The example is shown in Fig. 1 (c).

It is worth noting that after performing crossover the children may not be feasible with respect to conflict of assigned *FS*s. Thus, the assigned *FS*s of lightpaths in the new block (or sub-matrix) should be reassigned. The first-fit approach is used to reassign the required *FS*s.

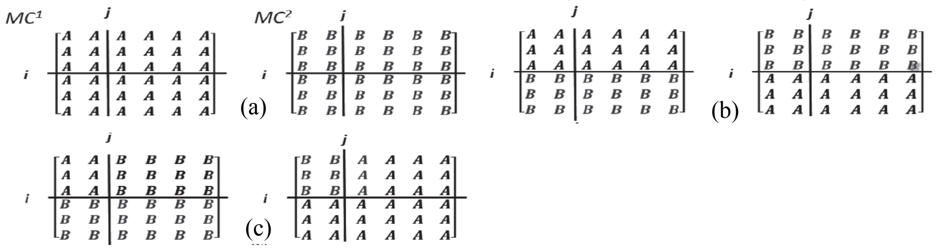


Fig. 1. (a) Before crossover; (b) after performing ROC; (c) after performing BOC.

5.5 Mutations

Three mutating operations are considered in the proposed GA: In these mutations, two integers i and j are generated randomly in the range [1, n]. These numbers are used as the mutation points.

- *Starting index mutation* (*SIM*): The selected element $MC_{ij} \in MC$ changes the value of the starting index, which is randomly generated between 1 to $\lceil k/2 \rceil$, where k is the current starting index. After performing mutation, the list of the lightpaths for demand N_{sd} should be changed accordingly and the assigned *FS*s should be recomputed.
- *Maximum allocates first mutation* (*MAFM*): The selected element $MC_{ij} \in MC$ changes the value of the starting index to 1 and the list of lightpaths is selected by applying the MAF heuristic algorithm for the demand N_{ij}.
- *Maximum delay path first mutation* (*MDPFM*): The selected element $MC_{ij} \in MC$ changes the value of the starting index to 1 and the list of lightpaths is selected by applying the MDPF heuristic algorithm for the demand N_{ij}.

5.6 Fitness Function Definition

Generally, genetic algorithms use fitness functions to map objectives to costs to achieve the goal of finding optimal solutions. If the content of the chromosome is known, the total cost of the chromosome can be determined. A fitness function value is associated with each chromosome, which is the same as the measure mentioned above. Since the goal is to minimize the total cost of transponders, the fitness function can be defined as $C \times TR$.

Since the best-fit chromosomes should have a probability of being selected as parents that is proportional to their fitness, they need to be expressed in a maximized form. This is done by subtracting the objective from a large number C_{max}. Hence, the fitness function becomes C_{max}–(C×TR), where C_{max} denotes the maximum value observed, so far, of the cost function in the population. Let C_{max} be the value of the cost function for the chromosome; C_{max} can be calculated by the following iterative equation:

$$C_{max}=\max\{C_{max}, C_{cost}\}, \tag{11}$$

where C_{max} is initialized to zero.

5.7 Replacement Strategy and Termination Rules

Initially, assume *pop* be the number chromosomes to be generated and N_{parent} connection chromosomes are randomly constructed. In the process of selection, *pop/*2 pairs of connections are randomly selected for crossover to generate the new generation of chromosomes. After crossover, chromosomes are sorted according to the fitness function in increasing order, N_{parent} chromosomes with smaller fitness is selected to construct the new generation. The values of *pop* and N_{parent} will be determined through experiments. Execution of GA can be terminated when the number of generations exceeds an upper bound specified by the user.

6. SIMULATION RESULTS

The proposed algorithms were coded by using C++ programming language. All simulations were run on a notebook computer with Intel i7-3610 QM 2.30 GHz CPU, 8.0 GB RAM and with Windows 8.1 operating system. The parameters of the GA are determined through simulations, the population size (*pop*) is 100, the number of selected parents (N_{parent}) is 50, the number of generations is 20, the crossover probability is 1.0, and the mutation probability is 0.1. In the following, the result of GA used for comparison is the average result of 5 simulations.

6.1 ILP Simulations

The ILP formulation stated in Section 2.5 is modeled in AMPL [17] and solved by CPLEX [18] in order to obtain optimal solutions as a reference for the proposed genetic algorithm and two heuristic algorithms. If the number of candidate paths for each node-pair does not limit by the formulation (denoted as $|PS|=\infty$), CPLEX can search the entire solution space thus provide lower bounds of the problem. For the larger problem instances, CPLEX is not able to find optimal solutions due to huge constraints (for the last case in Fig. 3) and the solution is marked as '*'.

The simulations for the ILP are run on a six-node network N1 and an eight-node network N2 shown in Figs. 2 (a) and (b), respectively. The number nears the line represents the delay of the link and the unit of the delay is millisecond (ms). The traffic demands matrix of N1 and N2 is the matrix Λ^1 and Λ^2, respectively. Each of which is the 6×6 and 8×8 sub-matrix of the matrix Λ shown in Fig. 4 (b), respectively. The total transponders provided by the heuristic algorithms and GA were compared with the results provided by CPLEX. For the CPLEX calculations were performed with CPLEX version 7. Fig. 3 shows a comparison of the objective costs and run-times (in seconds).

The column “%” represents the ratio of the cost of the algorithm to that of the cost of ILP method. In these simulations, the Δ_{sd} is set to 2.8, different values of B and $|PS|$ (the number of candidate paths for each node-pair). Fig. 3 shows that GA can find the same results as the CPLEX method for all cases run on the six-node network N1. The results of MDPF and MAF algorithms are the same and worst than GA and ILP.

For the eight-node network N2, the results found by performing the GA are very close to that of the CPLEX method, the average ratio of the cost of GA to that of the ILP is about 101.82%. The MAF can get better results than the MDPF, and the average ratio of the cost of MAF to that of the ILP is about 110.91%. For most of the cases in N2, CPLEX can finish in a reasonable time, but for the $|PS|=\infty$ case (means all possible paths of node-pair are considered for finding solutions), CPLEX fails to find solution due to so many constraints.

6.2 Comparisons

Two larger networks were used for simulations, the topologies of the networks are shown in Figs. 4 (a) and 5 (a), the number nears the line represents the delay of the link and the unit of the delay is millisecond (ms). The traffic demand matrices are shown in Figs. 4 (b) and 5 (b), the unit of the element in the matrix is the number of frequency slots.

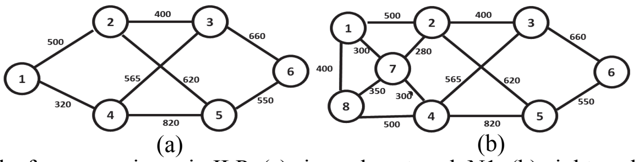
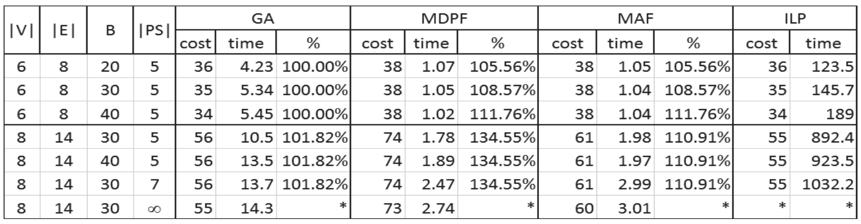
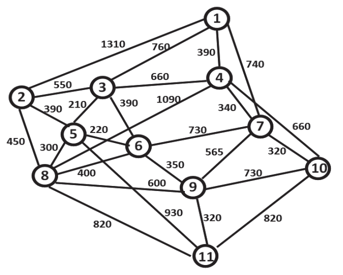


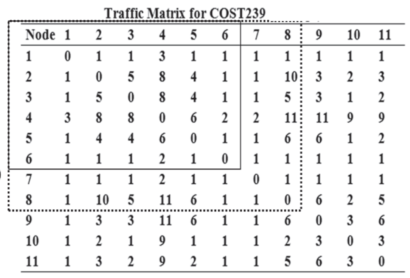
Fig. 2. Networks for comparisons in ILP; (a) six-node network N1; (b) eight-node network N2.



|V|	|E|	B	|PS|	GA cost	GA time	GA %	MDPF cost	MDPF time	MDPF %	MAF cost	MAF time	MAF %	ILP cost	ILP time
6	8	20	5	36	4.23	100.00%	38	1.07	105.56%	38	1.05	105.56%	36	123.5
6	8	30	5	35	5.34	100.00%	38	1.05	108.57%	38	1.04	108.57%	35	145.7
6	8	40	5	34	5.45	100.00%	38	1.02	111.76%	38	1.04	111.76%	34	189
8	14	30	5	56	10.5	101.82%	74	1.78	134.55%	61	1.98	110.91%	55	892.4
8	14	40	5	56	13.5	101.82%	74	1.89	134.55%	61	1.97	110.91%	55	923.5
8	14	30	7	56	13.7	101.82%	74	2.47	134.55%	61	2.99	110.91%	55	1032.2
8	14	30	∞	55	14.3	*	73	2.74	*	60	3.01	*	*	*

Fig. 3. Simulation results of GA, HAs and ILP on N1 and N2 networks.



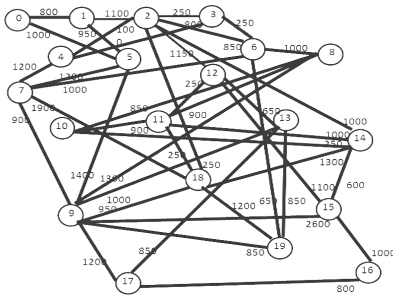


Traffic Matrix for COST239

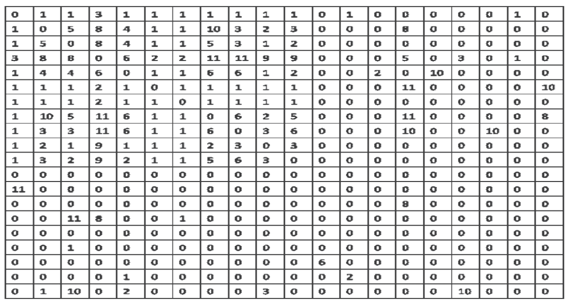
Node	1	2	3	4	5	6	7	8	9	10	11
1	0	1	1	3	1	1	1	1	1	1	1
2	1	0	5	8	4	1	1	10	3	2	3
3	1	5	0	8	4	1	1	5	3	1	2
4	3	8	8	0	6	2	2	11	11	9	9
5	1	4	4	6	0	1	1	6	6	1	2
6	1	1	1	2	1	0	1	1	1	1	1
7	1	1	1	2	1	1	0	1	1	1	1
8	1	10	5	11	6	1	1	0	6	2	5
9	1	3	3	11	6	1	1	6	0	3	6
10	1	2	1	9	1	1	1	2	3	0	3
11	1	3	2	9	2	1	1	5	6	3	0

(b)

Fig. 4. (a) COST239 network; (b) Traffic matrix for COST239 $\Lambda_{11\times11}$, for N1 $\Lambda^1_{6\times6}$, for N2 $\Lambda^2_{8\times8}$.



(a)



0	1	1	3	1	1	1	1	1	1	1	0	1	0	0	0	0	0	1	0
1	0	5	8	4	1	1	10	3	2	3	0	0	0	8	0	0	0	0	0
1	5	0	8	4	1	1	5	3	1	2	0	0	0	0	0	0	0	0	0
3	8	8	0	6	2	2	11	11	9	9	0	0	0	5	0	3	0	1	0
1	4	4	6	0	1	1	6	6	1	2	0	0	2	0	10	0	0	0	0
1	1	1	2	1	0	1	1	1	1	1	0	0	0	11	0	0	0	0	10
1	1	1	2	1	1	0	1	1	1	1	0	0	0	0	0	0	0	0	0
1	10	5	11	6	1	1	0	6	2	5	0	0	0	11	0	0	0	0	8
1	3	3	11	6	1	1	6	0	3	6	0	0	0	10	0	0	10	0	0
1	2	1	9	1	1	1	2	3	0	3	0	0	0	0	0	0	0	0	0
1	3	2	9	2	1	1	5	6	3	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	0	0	0	0	0
0	0	11	8	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	6	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0
0	1	10	0	2	0	0	0	0	3	0	0	0	0	0	0	10	0	0	0

(b)

Fig. 5. (a) Simulation network with 20 nodes; (b) traffic demand matrix.

The link fragmentation ratio is usually introduced to describe the spectrum occupation of link [7]. The *fragmentation ratio* (FR_{e_l}) of a link e_l is 1 subtract the ratio of the maximal number of continuous free frequency-slots (denoted as $MaxBlock(e_l)$) on link e_l to the number of free frequency-slots on link e_l (denoted as $B\text{-}sum(b_{e_l})$). It can be computed by the following formula:

$$FR_{e_l} = \begin{cases} 1-(MaxBlock(e_l)/(B-sum(b_{e_l}))), & sum(b_{e_l}) < B \\ 0, & sum(b_{e_l}) = B. \end{cases}$$

Moreover, let $MFR = \max\{FR_{e_l} | \forall e_l \in E\}$ and the *guard band ratio* (GBR) of a network is the ratio of the number of guard band frequency-slots on the network to the total number of frequency-slots on the network.

The simulation results of these algorithms on COST239 and NET networks are shown in Figs. 6 and 7, respectively. Simulations for the effect of different values of delay variation (Δ_{sd}) are examined, where the delay variation for all pairs of nodes is the same. In these simulations, the value of F is set to 4 and the value of B is set to 20. Figs. 6 (a) and 7 (a) show that the CPU time in seconds of MAF is higher than that of the MDPF, the GA is the most time-consuming algorithm, and the computation time slightly increases as the value of delay variation increases. Figs. 6 (b) and 7 (b) show that the GBR of MAF is smaller than that of the MDPF and the curve MDPF is relatively stable, GA can get the smallest value than other heuristic algorithms. Figs. 6 (c) and 7 (c) show that the number of TXs/RXs of MDPF is greater than that of the MAF, and GA can get the best solution. For the MAF algorithm, the number of TXs/RXs increases as the value of delay variation increases. Figs. 6 (d) and 7 (d) show that the MFR of the MDPF is greater than that of the MAF and GA, the curve MDPF is relatively stable. For the MAF algorithm, the MFR increases as the value of delay variation increases from 0.9 to 1.0. For the GA algorithm, the MFR increases as the value of delay variation increases from 1.1 to 1.2.

For different values of F, the simulation results of these algorithms for the case with Δ_{sd} = 0.8 are shown in Figs. 8 and 9, respectively. Figs. 8 (a) and 9 (a) show that the CPU time in seconds of MAF is higher than that of the MDPF, GA is the most time-consuming method, and the time decreases as the value of F increases. Since as the value of *F* increases, the demand can be allocated more quickly. As the *F* nears the maximal demand (11 *FS*s in matrix *N*), the computation time of two heuristic algorithms is very close. Figs. 8 (b) and 9 (b) show that the GBR of GA is smaller than that of the MAF and MDPF, and the GBR of the GA and MAF increases quicker than that of the MDPF, as the value of *F* increases. Figs. 8 (c) and 9 (b) show that the total number of TXs/RXs of GA is smaller than that of the MAF and MDFA, and the numbers of TXs/RXs for these methods decrease, as the value of *F* increases. Moreover, as the value of *F* increases, the difference between these algorithms decreases. Figs. 8 (d) and 9 (d) show that the MFR of the MDPF is greater than that of the MAF and GA, and the MFR value increases as the value of *F* increases.

7. CONCLUSIONS

In this paper, the virtual topology design problem on EONs has been studied. For a given EON and a traffic demand matrix, the goal is to design a virtual topology so as to minimize the total cost of transponders. In the studied problem, the demand can be transmitted by multiple lightpaths which satisfy the delay-variation constraint. Since the virtual topology design problem is a hard problem. Two heuristic algorithms and a genetic algorithm (GA) have been proposed in this paper to solve this problem. The proposed two heuristic algorithms are *Minimum Delay Path First* (MDPF) and *Maximal Allocates First* (MAF). Moreover, an ILP model has also been used to define the problem and the AMPL/CPLEX package has been used to find the solution. Simulations have been conducted to evaluate the cost of the transponders and MFR of these methods. The results showed that the proposed GA can get better results than MAF and MDPF algorithms on median size networks. The propose GA also can get near optimal results on the small size networks.

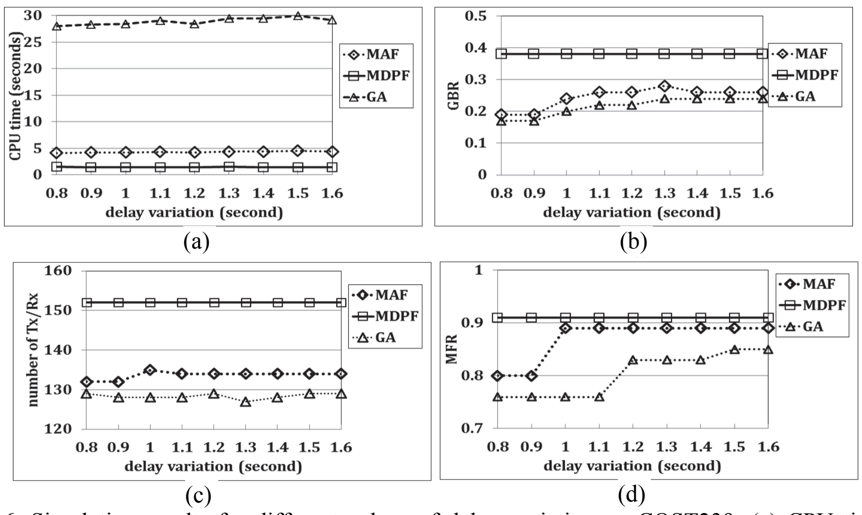


Fig. 6. Simulation results for different values of delay variation on COST239: (a) CPU time in seconds; (b) used ratio; (c) number of TXs/RXs, (d) MFR.

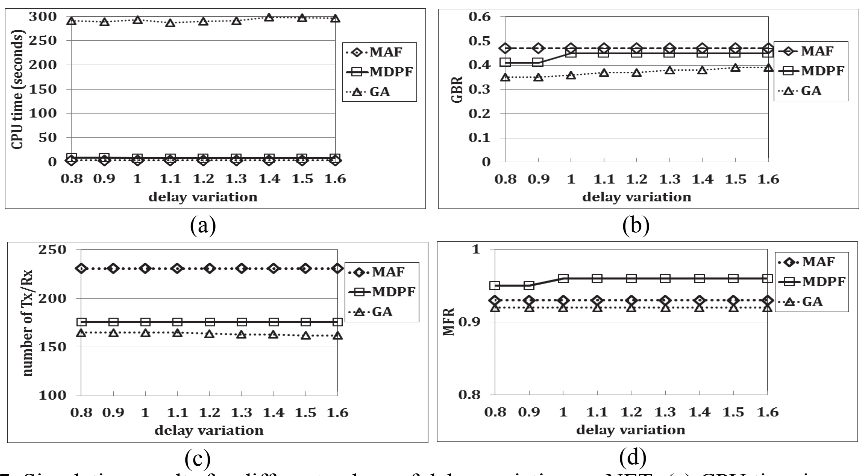


Fig. 7. Simulation results for different values of delay variation on NET: (a) CPU time in seconds; (b) used ratio; (c) number of TXs/RXs; (d) MFR.

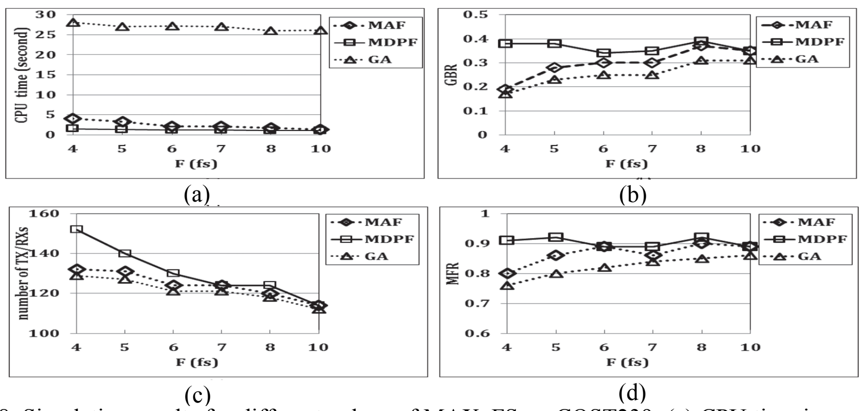


Fig. 8. Simulation results for different values of MAX_FS on COST239: (a) CPU time in seconds; (b) used ratio; (c) number of TXs/RXs, (d) MFR.

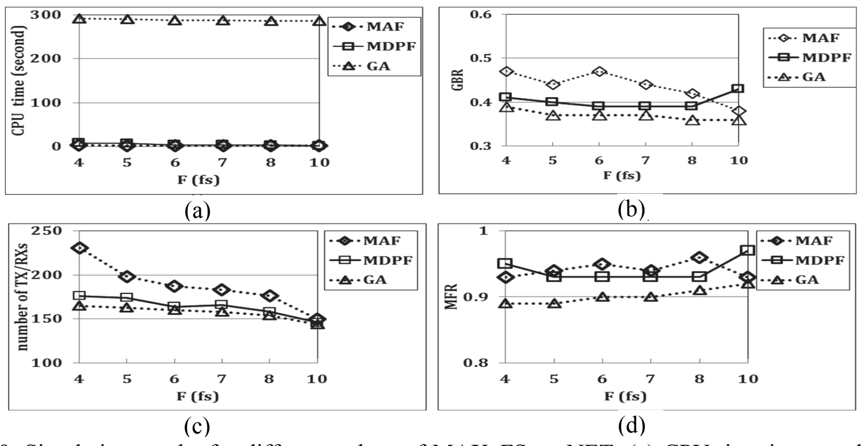


Fig. 9. Simulation results for different values of MAX_FS on NET: (a) CPU time in seconds; (b) used ratio; (c) number of TXs/RXs; (d) MFR.

REFERENCES

1. G. Zhang, M. D. Leenheer, A. Morea, and B. Mukherjee, "A survey on OFDM-based elastic core optical networking," *IEEE Communications Surveys and Tutorials*, Vol. 15, 2013, pp. 65-87.
2. M. Klinkowski and K. Walkowiak, "Routing and spectrum assignment in spectrum sliced elastic optical path network," *IEEE Communications Letters*, Vol. 15, 2011, pp. 884-886.
3. W. Lu, X. Zhou, L. Gong, M. Zhang, and Z. Zhu, "Dynamic multi-path service provisioning under differential delay constraint in elastic optical networks," *IEEE Communications Letters*, Vol. 15, 2013, pp. 158-161.
4. L. Ruan and N. Xiao, "Survivable multipath routing and spectrum allocation in OFDM-based flexible optical networks," *Journal of Optical Communications and Networking*, Vol. 5, 2013, pp. 172-182.
5. K. Christodoulopoulos, I. Tomkos, and E. Varvarigos, "Routing and spectrum allocation in OFDM-based optical networks with elastic bandwidth allocation," in *Proceedings of IEEE Global Telecommunications Conference*, 2010, pp. 1-6.
6. Y. Wang, X. Cao, and Y. Pan, "A study of the routing and spectrum allocation in spectrum-sliced elastic optical path networks," in *Proceedings of IEEE INFOCOM*, 2011, pp. 1503-1511.
7. S. Talebi, F. Alam, I. Katib, M. Khamis, R. Salama, G. N. Rouskas, "Spectrum management techniques for elastic optical networks: a survey," *Optical Switching and Networking*, Vol. 13, 2014, pp. 34-48.
8. X. Liu, L. Gong, and Z. Zhu, "Design integrated RSA for multicast in elastic optical networks with a layered approach," in *Proceedings of IEEE Global Telecommunications Conference*, 2013, pp. 2346-2351.
9. A. Shami, C. Assi, I. Habib, and M. A. Ali, "Performance evaluation of two GMPLS-based distributed control and management protocols for dynamic lightpath provisioning in future IP networks," in *Proceedings of Internaitonal Conference on Communications*, 2002, pp. 2289-2293.
10. X. Cheny, A. Jukany, and A. Gumaste, "Multipath de-fragmentation: achieving better spectral efficiency in elastic optical path networks," in *Proceedings of IEEE International Conference on Computer Communications*, 2013, pp. 390-394.
11. X. Chen, A. Jukan, and A. Gumaste, "Optimized parallel transmission in elastic optical networks to support high-speed ethernet," *Journal of Lightwave Technology*, Vol. 32, 2014, pp. 228-238.
12. R. Dutta and G. N. Rouskas, "A survey of virtual topology design algorithms for wavelength routed optical networks," *Optical Networks Magazine*, Vol. 1, 2000, pp. 73-89.
13. B. Wu, K. L. Yeung, and P.-H. Ho, "Virtual topology design for minimizing network diameter and average hop count in WDM networks," *IEEE/OSA Journal of Optical Communications and Networking*, Vol. 2, 2010, pp. 1077-1086.
14. Y. Yu, Y. Zhao, J. Zhang, H. Li, Y. Ji, and W. Gu, "Virtual topology design scheme with energy efficiency for IP over elastic optical networks," *Chinese Optics Letters*, Vol. 12, 2014, pp. 110602.

15. J. Holland, *Adaptation in Natural and Artificial Systems*, *An Introductory Analysis with Applications to Biology*, *Control and Artificial Intelligence*, MIT Press Cambridge, MA, 1992.
16. D. Eppstein, "Finding the *k* shortest paths," *SIAM Journal on Computing*, Vol. 28, 1998, pp. 652-673.
17. R. Fourer, D. M. Gay, and B. M. Kernighan, *AMPL: A Modelling Language for Mathematical Programming*, 2nd ed., Duxbury, MA, 2003.
18. *IBM ILOG CPLEX Optimization Studio CPLEX User's Manual*, IBM Corp., 2014.

Der-Rong Din (丁德榮) received Ph.D. degree in Computer and Information Science from National Chiao-Tung University, Hsinchu, Taiwan in 2001. Now, he is currently a Professor at National Changhua University of Education. His current research interests are in WDM elastic optical network, mobile communication, parallel compiler and algorithms.