On Joint Topology Design and Load Balancing in Free-Space Optical Networks

In Keun Son\textsuperscript{a}, Shiwen Mao\textsuperscript{a,b}, Sajal K. Das\textsuperscript{c}

\textsuperscript{a}Defense Acquisition Program Administration (DAPA), Republic of Korea.
\textsuperscript{b}Department of Electrical and Computer Engineering, Auburn University, Auburn, AL 36849-5201, USA
\textsuperscript{c}Department of Computer Science and Engineering, University Texas at Arlington, Arlington, TX 76019, USA

Abstract

Free space optical networks have emerged as a viable technology for broadband wireless backbone networks of the next generation. In this paper, we investigate the challenging problem of joint topology design and load balancing in FSO networks. We consider FSO link characteristics, cost constraints, traffic characteristics, traffic demand, and QoS requirements in the formulation, along with various objective functions including network-wide average load and delay. We apply the Reformulation-Linearization Technique (RLT) to obtain linear programming (LP) relaxations of the original complex problem, and then incorporate the LP relaxations into a branch-and-bound framework. The proposed algorithm can produce highly competitive solutions with performance guarantees in the form of bounded optimality gap. For reducing computation complexity, we also develop a fast heuristic algorithm to provide highly competitive solutions. The heuristic algorithm iteratively perturbs the current topology and computes network flows for the new topology, thus progressively improving the configuration and load balancing of the FSO network. The proposed algorithms are complementary to each other, since jointly applying the algorithms can make the FSO network dynamically adaptive to events occurring at both large and small timescales. The proposed algorithms are evaluated with extensive simulations. Our simulation results show that the heuristic algorithm can achieve an optimality gap close to that of the branch-and-bound algorithm, with significantly reduced computation time.

Key words: Free space optics, load balancing, multipath routing, topology design, optimization.

1. Introduction

Free space optics (FSO) have emerged as a promising technology for broadband wireless networks of the next generation [3]. FSOs are wireless systems that use free space as transmission medium to transmit optical data signals at high bit rates. FSOs have many advantages such as cost effectiveness, long transmission range, free license, interference immunity, and high-bandwidth, among others. In recent years, considerable advances have been made in understanding the FSO channel, and both experimental data and commercial FSO transceivers are now available [3]. For widespread deployment of FSO networks, several important network problems should be addressed, such as how to design an FSO network topology with rich connectivity (making it robust to link failures) and how to accommodate traffic demands and QoS requirements of the underlying wired or wireless access network.

The FSO network topology design problem has been addressed in several prior works. In [4], a distributed Minimum Spanning Tree (MST) algorithm was proposed to build degree-bounded tree topologies. In [5, 6], algorithms were developed to maximize network connectivity and make a mesh topology. The load balancing problem was addressed in [7, 8], where several topology design heuristics were developed to minimize network-wide average load. To maximize the potential of FSO networks, the unique characteristics of FSO links should be considered in the formulation, and the problems of topology design and routing of the traffic flows should be jointly considered and optimized [7, 8]. Usually such problems are highly complex. Consequently, an approximation algorithm with performance guarantees (i.e., in the form of a bounded optimality gap) would be highly appealing.

An FSO link is “pseudo-wired” in the sense that it has high bandwidth, narrow beam, and long distance as an optical fiber link. However, it is also like a radio frequency (RF) link that is flexibly steerable, in contrast to buried optical fiber links. Thus the network topology can be adaptively reconfigured on-the-fly in response to network dynamics. In this paper, we investigate the problem of joint topology design and load balancing in FSO networks, considering important design issues such as link reliability, cost constraints, traffic characteristics and demand, QoS requirements, routing policies, and network topology. We assume that a traffic matrix is known and
the total number of edges used for building a topology is given a priori (due to some cost constraint). We then formulate the joint topology design and load balancing problem with objectives to minimize network-wide average load or network-wide average delay. Since for the same offered load, different traffic models will yield very different delay performance, we consider both short-range dependent (SRD) and long-range dependent (LRD) traffic models when deriving network delays.

With the objective function of average load, the formulated problem is a Mixed Integer Linear Programming (MILP) problem. With the objective function of average delay, the formulated problem is a Mixed Integer Nonlinear Programming (MINLP) problem. These problems are NP-hard in general [7, 8]. In prior work [7, 8], effective heuristic algorithms are presented to minimize the network-wide average load. However, there was no guarantee on the optimality performance of the heuristic algorithms, and they do not apply to the more complex problem of minimizing network-wide average delay.

In this paper, we first develop a branch-and-bound algorithm incorporating the Reformulation-Linearization Technique (RLT) that can produce highly competitive solutions with bounded performance. RLT is a useful technique that can be applied to derive linear programming (LP) relaxations for an underlying non-linear non-polynomial programming problem [9]. We first adopt RLT to obtain LP relaxations for the complex MILP and MINLP problems. We then incorporate the LP relaxations into the branch-and-bound framework to compute (1-\epsilon)-optimal solutions, where 0 \leq \epsilon \ll 1 is a prescribed tolerance. When the algorithm terminates, it produces a feasible solution to the original MILP or MINLP problem, which is within the \epsilon range of the global optimum.

Although highly appealing, the RLT-based branch-and-bound algorithm has a relatively high computation complexity. We next present a fast heuristic algorithm to the joint topology design and load balancing problem. The heuristic algorithm consists of three components: (i) initial topology design, (ii) multipath routing for load balancing, and (iii) topology perturbation. Starting from an initial topology that is designed to minimize the network-wide average load, the heuristic algorithm iteratively perturbs the current topology and computes network flows for the new topology, thus progressively improving the configuration and load balancing of the FSO network.

The proposed algorithms are evaluated with extensive simulations, and are shown to be highly suitable for joint topology and load balancing optimization in FSO networks. Our simulation results also show that the heuristic algorithm can achieve an optimality gap close to that of the branch-and-bound algorithm, but with significantly reduced computation time.

The proposed fast heuristic algorithm and the branch-and-bound algorithm are complementary to each other. The latter is suitable for optimizing the FSO network design and operation at large timescales with guaranteed optimality. The former is suitable for dynamic reconfiguration of the FSO network in response to small timescale events. We envision that the branch-and-bound algorithm will be executed at relatively large time intervals when significant changes occur in the FSO network, while the heuristic algorithm will be kept running to continuously optimize the operation and configuration of the FSO network in response to small timescale events such as bad weather conditions or fluctuations in traffic demand.

The remainder of this paper is organized as follows. In section 2, we describe the system model and assumptions, and formulate the optimization problem. The RLT-based algorithm is presented in Section 3 and the fast heuristic algorithm is presented in Section 4. The algorithms are evaluated in Section 5. We discuss related work in Section 6, and Section 7 concludes this paper.

2. System Model and Problem Statement

2.1. Network Model

We consider an FSO network consisting of \( n \) base stations (BS), which provide mobile users with network access. Each BS could be the head of a cluster consisting of multiple access points. The aggregate traffic at the BS’s will be relayed through wireless optical links. We assume that each BS has multiple sets of wireless optical devices in order to support the aggregate traffic load and provide a rich mesh connectivity. The FSO links are point-to-point connections with narrow beam divergence, and are immune to electromagnetic interference [3, 6, 7, 8].

The FSO network can be modeled as a simple graph \( G(V, E) \), where each vertex \( v \in V \) represents a BS and each edge \( e \in E \) is an FSO link. Let \( n \) and \( m \) denote the cardinality of \( V \) and \( E \) respectively. We assume an \( n \times n \) traffic matrix \( \mathbf{F} \) that describes the traffic demand (measured, estimated, or projected) for the access network, where each element \( f_{sd} = [\mathbf{F}]_{sd} \) represents the mean data rate between each source and destination BS pair \( s-d \).

We characterize each FSO link \( e = (i, j) \in E \) with two parameters: i) link capacity \( c_{ij} \); and ii) link reliability \( \gamma_{ij} \). As in prior work [8], we assume that each FSO channel is full duplex with symmetric capacity and a nominal data rate \( c \) is achievable within a predefined transmission range, i.e., \( c_{ij} = c_{ji} = c \) for all \( i \neq j \). We also assume symmetric link reliability, i.e., \( \gamma_{ij} = \gamma_{ji} \) for all \( i \neq j \), due to the line-of-sight transmissions with narrow beam divergence. There is connectivity between two BS’s if the link reliability is larger than a threshold \( \gamma_{th} \).

2.2. FSO Channel Model

We adopt the log-normal model to characterize FSO link reliability under turbulent atmosphere [10, 11]. The marginal distribution of light intensity fading induced by atmospheric turbulence can be statistically modeled as [10]

\[
    f_{I}(I) = \frac{1}{2\sigma_X I \sqrt{2\pi}} \exp \left\{ -\frac{(\ln(I) - \ln(I_0))^2}{8\sigma_X^2} \right\}, \quad (1)
\]

\( I \) is the received light intensity, \( \sigma_X \) is the standard deviation of the log-normal random variable, and \( I_0 \) is the reference intensity at short distances.
where $\sigma_X^2$ is the variance and $I_0$ is the received average intensity without turbulence. The standard deviation $\sigma_X$ can be approximated as $\sigma_X = 0.30545 \left( \frac{2\pi}{\lambda} \right)^{7/6} C_n^2(L) z^{11/6}$, where $\lambda$ is the wavelength, $C_n^2(L)$ is the index of refraction structure parameter with constant altitude $L$, and $z$ is the distance. It is shown that for atmospheric channels near the ground (i.e., $L < 18.5m$), $C_n^2$ ranges from $10^{-13}\text{m}^{-2/3}$ to $10^{-17}\text{m}^{-2/3}$ for strong and weak atmospheric turbulence respectively. The common average is $10^{-15}\text{m}^{-2/3}$.

The link reliability $\gamma_{ij}$ is the probability that the intensity of received signal $I$ exceeds a threshold $I_{th}$, which can be computed using the error function $\text{erf}()$ as

$$\gamma_{ij} = P(I \geq I_{th}) = \frac{1}{2} - \frac{1}{2} \text{erf} \left( \frac{\ln(I_{th}/I_0)}{2\sqrt{2}\sigma_X} \right).$$

The ratio $I_{th}/I_0$ can be interpreted as transmittance according to Beer-Lambert Law [12], which is determined by the distance and absorption coefficient. For a fixed ratio $I_{th}/I_0$, $\gamma_{ij}$ depends on the standard deviation $\sigma_X$, which is strongly influenced by weather conditions and transmission distance [13]. We assume that the set of edges satisfying $\gamma_{ij} \geq \gamma_{th}$ forms the candidate link set for constructing the FSO network topology.

2.3. Performance Measures

2.3.1. Network-wide Average Load ($L$)

We adopt multipath routing for load balancing, where a flow $f_{sd}$ may be split into multiple subflows. Let $f_{ij}^{sd}$ be the subflow of $f_{sd}$ passing through a link $(i, j)$. We have the following flow-conservation condition.

$$\sum_{j=1}^{n} f_{ij}^{sd} - \sum_{j=1}^{n} f_{ji}^{sd} = \begin{cases} f_{sd}, & i = s, \forall i \in V \\ -f_{sd}, & i = d, \forall i \in V \\ 0, & \text{otherwise}, \forall i \in V. \end{cases}$$

Considering all the $s-d$ pairs, the average traffic load $\lambda_{ij}$ and the link utilization $\rho_{ij}$ are

$$\lambda_{ij} = \sum_{s,d \in V} f_{ij}^{sd} \text{ and } \rho_{ij} = \lambda_{ij}/c_{ij} < 1.$$ (4)

For the link to be stable, we have $\rho_{ij} < 1$, for all $(i, j) \in E$. We define the network-wide average load $L$ as

$$L \cong (1/\lambda) \cdot \sum_{(i,j) \in E} \lambda_{ij},$$

where $\lambda = \sum_{s,d \in V} f_{sd}$ is the sum of all traffic demands. Note that when a packet is forwarded, it is counted multiple times in $L$. When all the $s-d$ traffic are transmitted through direct links, $L$ achieves its minimum value 1.

2.3.2. Network-wide Average Delay ($T_1, T_2$)

We model each link $(i, j) \in E$ as a general queueing system with average input rate $\lambda_{ij}$ and service capacity $c_{ij}$. The average delay incurred at the link depends on the traffic auto-correlation structure. When the traffic constantly exhibits short-range dependent (SRD) characteristics (e.g., voice over IP (VoIP) traffic), we can model the link delay with an exponential distribution with parameter $c_{ij} - \lambda_{ij}$ [14]. Applying Little’s formula, the network-wide average delay $T_1$ can be computed as

$$T_1 \cong \sum_{(i,j) \in E} \frac{1}{\lambda} \frac{\lambda_{ij}}{c_{ij} - \lambda_{ij}}.$$ (6)

When the traffic exhibits long-range dependent (LRD) characteristics (e.g., computer data or variable-bit-rate video traffic), we can model each link as a fractional Brownian motion (fBm) queueing system, where the queue length has a heavy-tailed Weibull distribution [15], i.e.,

$$\Pr\{Q_{ij} > q\} \approx \exp \left\{ -\left(\frac{c_{ij} - \lambda_{ij}}{2\kappa^2(H)a\lambda_{ij}}\right)^{2H} q^{2-2H} \right\},$$ (7)

where $\kappa(H) = H^H(1-H)^{1-H}$, $H \in [0.5, 1)$ is the Hurst parameter, and $a$ is the index of dispersion. Applying Little’s formula, the network-wide average delay $T_2$ is

$$T_2 \cong \frac{1}{\lambda} \Gamma \left(1 + \frac{1}{2 - 2H}\right) \sum_{(i,j) \in E} \left(\frac{\lambda_{ij}}{2\kappa^2(H)a\lambda_{ij}}\right) \frac{1}{\gamma_{ij}}.$$ (8)

Defining $\tau = \frac{1}{\lambda} \Gamma \left(1 + \frac{1}{2 - 2H}\right) \left[2\kappa^2(H)a\right]^{-1/2}$, we have

$$T_2 = \tau \cdot \sum_{(i,j) \in E} \left(\frac{\lambda_{ij}}{(c_{ij} - \lambda_{ij})^{2H}}\right)^{1/2\gamma_{ij}}.$$ (9)

2.4. Problem Statement

Without loss of generality, we consider the case of fixed BS’s. The atmospheric condition can be known through weather forecast and past experiences. Then we can evaluate the edge reliabilities and determine the candidate edge set $E_c$. The number of links in the FSO network is upper bounded by $m_c = |E_c|$. The number of links is also lower bounded by $n - 1$, the minimum number of links that is needed to construct a connected network (i.e. a minimum spanning tree).

The problem is to select $m$ links from $m_c$ candidate edges to form a mesh topology. In addition, we also determine multipath routing for the $s-d$ flows, such that either the network-wide average load $L$ or the network-wide average delay $T_1$ or $T_2$ is minimized. Define the following index variables for each link $(i, j) \in V$ as

$$x_{ij} = \begin{cases} 1, & \text{if } (i,j) \in E_c \\ 0, & \text{otherwise}. \end{cases} \quad y_{ij} = \begin{cases} 1, & \text{if } (i,j) \text{ is chosen} \\ 0, & \text{otherwise}. \end{cases}$$

We have $x_{ij} \geq y_{ij}$ for all $(i,j) \in V$. The problem of joint topology design and load balancing, denoted as Problem
OPT-TDLB, can be formulated as follows.

\[
\begin{aligned}
& \text{minimize} & & L/T_1/T_2 \\
& \text{subject to:} & & \\
& & & \sum_{i=1}^{n} \sum_{j=x_{ij}}^{n} x_{ij} = m_c, \sum_{i=1}^{n} \sum_{j=y_{ij}}^{n} y_{ij} = m, \\
& & & \text{for } i, j \in V \\
& & & y_{ij} \leq x_{ij}, x_{ij} = x_{ji}, y_{ij} = y_{ji}, \text{ for } i, j \in V \\
& & & \theta_i^b \leq \sum_{j=1}^{n} y_{ij} \leq \theta_i^c, \text{ for } i \in V \\
& & & 0 \leq f_{ds}^c \leq y_{ij} f_{sd}, \text{ for } i, j, s, d \in V \\
& & & \lambda_{ij} = \sum_{s, d \in V} f_{ds}^c \leq c_{ij}, \text{ for } (i, j) \in E_c \\
& & & \text{flow conservation constraint (3)} \\
& & & x_{ij} \in \{0, 1\}, \ y_{ij} \in \{0, 1\}, \text{ for } i, j \in V.
\end{aligned}
\]

In Problem OPT-TDLB, the optimization variables include binary variables \( y = \{y_{ij} \mid \forall i, j \in V \} \) and continuous variables \( f = \{f_{ds}^c \mid \forall i, j, s, d \in V \} \). With objective function \( L \), Problem OPT-TDLB(\( L \)) is an MILP problem. With objective function \( T_1 \) or \( T_2 \), the corresponding Problems OPT-TDLB(\( T_1 \)) and OPT-TDLB(\( T_2 \)) are MINLP problems. Constraints (11) \~ (13) are for edge selection and topology design, while constraints (14) \~ (16) are for multipath routing and load balancing. In (13), \( \theta_i^b \) is the maximum degree (enforced by some cost constraints), and \( \theta_i^c \) is the minimum degree for BS \( i \), which is required to support the incoming and outgoing traffic from the BS and can be estimated as

\[
\theta_i^c = \max \{\lfloor \sum_{d=1}^{n} \lambda_{id}/c \rfloor, \lfloor \sum_{s=1}^{n} \lambda_{si}/c \rfloor \}.
\]

3. RLT-based Branch-and-Bound Algorithm

Our solution procedure for Problem OPT-TDLB is to incorporate an LP relaxation of the original problem into a branch-and-bound framework [9]. Problem OPT-TDLB(\( L \)) is an MILP. We can obtain its LP relaxation by allowing the binary variables \( y_{ij} \) to take real values in \([0, 1]\). In this section, we reformulate and linearize MINLP problems OPT-TDLB(\( T_1 \)) and OPT-TDLB(\( T_2 \)). The LP relaxations will then be incorporated into a branch-and-bound algorithm that can compute (1-\( \epsilon \))-optimal solutions.

3.1. Reformulation and Linearization: OPT-TDLB(\( T_1 \))

To linearize the objective function \( T_1 \), we define substitution variables \( t'_{ij} \) as \( t'_{ij} = \lambda_{ij}/(c_{ij} - \lambda_{ij}) \). Then we obtain a linear objective function \( \sum t'_{ij} \) and additional nonlinear constraints \( t'_{ij} \cdot c_{ij} - t'_{ij} \cdot \lambda_{ij} - \lambda_{ij} = 0 \).

We next linearize the nonlinear constraint by defining substitution variables for the quadratic term \( \mu_{ij} = t'_{ij} \cdot \lambda_{ij} \). Substituting \( \mu_{ij} \) into the additional nonlinear constraints, we make them linear, but with the additional RLT bound-factor product constraints for the new variable \( \mu_{ij} \). Since \( t'_{ij} \) and \( \lambda_{ij} \) are bounded by their respective lower and upper bounds as \( 0 \leq t'_{ij} \leq \bar{t} \) and \( 0 \leq \lambda_{ij} \leq c_{ij} \), we have

\[
\begin{aligned}
& \left( t'_{ij} - 0 \right) \cdot (\lambda_{ij} - 0) \geq 0 \\
& \left( t'_{ij} - 0 \right) \cdot (c_{ij} - \lambda_{ij}) \geq 0 \\
& \left( \bar{t} - t'_{ij} \right) \cdot (\lambda_{ij} - 0) \geq 0 \\
& \left( \bar{t} - t'_{ij} \right) \cdot (c_{ij} - \lambda_{ij}) \geq 0.
\end{aligned}
\]

Expanding (19) and substituting \( \mu_{ij} = t'_{ij} \cdot \lambda_{ij} \), we obtain the following RLT bound-factor product constraints.

\[
\begin{aligned}
& \mu_{ij} \geq 0 \\
& c_{ij} \cdot t'_{ij} - \mu_{ij} \geq 0 \\
& \bar{t} \cdot \lambda_{ij} - \mu_{ij} \geq 0 \\
& \bar{t} \cdot \lambda_{ij} - \mu_{ij} \geq 0.
\end{aligned}
\]

We then obtain an LP relaxation \( l\text{-OPT-TDLB}(T_1) \) as

\[
\begin{aligned}
& \text{minimize} & & (1/\lambda) \cdot \sum_{(i,j) \in E} t'_{ij} \\
& \text{subject to:} & & \\
& & & \text{constraints (11) \sim (16)} \\
& & & 0 \leq y_{ij} \leq 1, \text{ for } (i, j) \in E_c \\
& & & c_{ij} \cdot t'_{ij} - \mu_{ij} - \lambda_{ij} = 0, \text{ for } (i, j) \in E_c \\
& & & \text{RLT bound-factor constraints (20) for } (i, j) \in E_c.
\end{aligned}
\]

3.2. Reformulation and Linearization: OPT-TDLB(\( T_2 \))

The objective function \( T_2 \) consists of exponents. We define substitution variables \( t''_{ij} = [\lambda_{ij}/(c_{ij} - \lambda_{ij})]^{1/(2\cdot H)} \). Substituting \( t''_{ij} \) we obtain a linear objective function \( \sum t''_{ij} \), with additional nonlinear constraints: \( t''_{ij} \cdot (c_{ij} - \lambda_{ij}) \geq 0 \). Letting \( \nu_{ij} = \lambda_{ij} / c_{ij} \), we have \( t''_{ij} \cdot (\nu_{ij})^{\frac{H}{2H}} - \lambda_{ij} + \lambda_{ij} \geq 0 \). Finally, taking logarithms and defining substituting variables

\[
\zeta_{t''_{ij}} = \log(t''_{ij}), \quad \zeta_{\nu_{ij}} = \log(\nu_{ij}), \quad \zeta_{\lambda_{ij}} = \log(\lambda_{ij}),
\]

we obtain linear constraints \( \zeta_{t''_{ij}} - \frac{1}{2H} \zeta_{\nu_{ij}} \geq 0, \zeta_{\lambda_{ij}} = 0 \).

We still need to linearize the new nonlinear constraints (26), which are all in the form of \( y = \log(x) \). If \( x \) is bounded by \( 0 < x_l \leq x \leq x_u \), the logarithm relationship can be linearized using a polyhedral outer approximation comprised of a convex envelop in concert with several tangential supports [9], i.e.,

\[
\begin{aligned}
& y \geq \frac{\log(x_u) - \log(x_l)}{x_u - x} \cdot (x - x_u) + \log(x_u) \\
& y \leq x/\bar{x} + \log(\bar{x}) - 1,
\end{aligned}
\]

where \( x_k = x_l + \frac{k}{k_{\max} - 1} (x_u - x_l) \) for \( k = 0, \ldots, k_{\max} - 1 \). The convex envelope consists of a chord connecting two end points and \( k_{\max} \) supports each being tangent to the \( \log(x) \) curve at \( x_k \). Therefore, we generate \( k_{\max} + 1 \) new constraints for each logarithmic substitution variables in (26). Fig. 1 shows an example with the four-point tangential approximation.

Note that using a larger \( k_{\max} \) can reduce the area above the \( \log(x) \) curve. However, the reduction will not be significant since the area above the \( \log(x) \) curve is much smaller.
Figure 1: An example of polyhedral outer approximation for \( y = \log(x) \) where \( 0 < x_1 \leq x \leq x_u \) and \( k_{\text{max}} = 4 \). The two endpoints are marked by the stars (corresponding to \( x = x_1 \) and \( x = x_4 \)).

than the area below the \( \log(x) \) curve. However, using a larger \( k_{\text{max}} \) will produce more linear constraints in the relaxed problem, and cause the LP solver more time to solve the relaxed problem. Since the polyhedral outer approximation will be incorporated in the branch-and-bound framework, the infeasible area around the \( \log(x) \) curve will be iteratively reduced as the range \([x_1, x_k]\) is iteratively reduced. Our experience show that a small \( k_{\text{max}} \) of 3 or 4 would suffice.

We thus obtain an LP relaxation \( l\text{-OPT-TDLB}(T_2) \) as

\[
\begin{align*}
\text{minimize} & \quad \tau \cdot \sum_{(i,j) \in E'} v_{ij}' \\
\text{subject to:} & \quad \text{constraints (11) \sim (16)} \\
& \quad 0 \leq y_{ij} \leq 1, \text{ for } (i, j) \in \mathcal{E}_c \\
& \quad \zeta_{ij}' - \frac{1}{2 - 2H} \xi_{ij} + \frac{H}{1 - H} \xi_{ij} = 0, \text{ for } (i, j) \in \mathcal{E}_c \\
& \quad \nu_{ij} + \lambda_{ij} = c_{ij}, \text{ for } (i, j) \in \mathcal{E}_c \\
& \quad \text{polyhedral outer approximations (27) for } \\
& \quad \zeta_{ij}', \xi_{ij}, \nu_{ij} \text{ given in (26), for } (i, j) \in \mathcal{E}_c.
\end{align*}
\]

3.3. Branch-and-Bound Algorithm

In this section, we embedded the LP relaxations into the branch-and-bound framework to obtain solutions with bounded optimality gap. Branch-and-bound is an iterative optimization algorithm that is especially useful for solving discrete and combinatorial problems. It consists of two key components: (i) a strategy to split a problem into subproblems with smaller sizes, i.e., branching, and (ii) a fast way to obtain lower and upper bounds (LB, UB) for the subproblems, i.e., bounding. The resulting subproblems form a tree structure, while the set of leaf nodes is called the problem list \( \mathcal{P} \). During the solution process, a branch of the tree may be deleted (or, fathomed) from future search, if all its solutions are dominated by some other subproblems, therefore reducing the computational cost.

Our RLT-based Branch-and-Bound algorithm is given in Algorithm 1 and the flow chart is given in Fig. 2. Solving the LP relaxation \( l\text{-OPT-TDLB} \) with an LP solver, we can obtain a solution \( \hat{\delta} = (\hat{y}, \hat{f}) \), which is optimal to the LP relaxation. Note the relaxation is actually obtained by relaxing the constraints (i.e., augmenting the search region of the original problem). On one hand, the LP solution \( \hat{\delta} \) may lie outside the feasible region of the original problem (e.g., a fractional \( x_{ij} \), rather than binary) thus being infeasible to the original problem; on the other hand, if we substitute this possibly infeasible solution \( \hat{\delta} \) into the objective function, we obtain an objective value that is a lower bound (i.e., LB) of the original problem. Then we apply a local search algorithm to derive a feasible solution \( \delta \) in the neighborhood of \( \hat{\delta} \), which provides an UB for the original problem. The global LB and UB are updated as follows.

\[
\begin{align*}
\text{LB} &= \min \{ \text{LB}_h : \text{all problems } h \in \mathcal{P} \} \\
\text{UB} &= \min \{ \text{UB}_h : \text{all problems } h \in \mathcal{P} \}.
\end{align*}
\]

We adopt a simple local search algorithm as follows. First, we determine the network topology by fixing the \( y_{ij} \)'s to binaries. According to constraint (11), \( m \) links should be selected from the candidate set \( \mathcal{E}_c \) to form a topology. Hence, we choose the \( m \) largest \( \hat{y}_{ij} \)'s in \( \hat{y} \) and set them to 1; the rest smaller \( \hat{y}_{ij} \)'s are set to 0. When the topology is fixed, we then determine the optimal multipath routing by solving Problem \( l\text{-OPT-TDLB} \) again for the optimal \( f_{ij} \)'s. The resulting solution \( \delta = (y, f) \) is thus feasible and provides an upper bound.

For branching, we choose the subproblem \( h \) with the smallest \( \text{LB}_h \), which is indicative of the global optimal solutions. Subproblem \( h \) is then partitioned into two subproblems, \( h_1 \) and \( h_2 \), which replace the subproblem \( h \) in \( \mathcal{P} \). The corresponding \( \text{LB}_h, \text{LB}_{h_1}, \text{UB}_{h_1}, \) and \( \text{UB}_{h_2} \) are calculated, and the global LB and UB are updated as in (34). The iteration procedure terminates when there is a feasible solution satisfying \( \text{LB} \geq (1 - \epsilon) \cdot \text{UB} \), or when the problem list \( \mathcal{P} \) is empty.

For this minimization problem, the UB is from a feasible solution and the LB is from a possibly infeasible solution. If the global optimal solution is \( T^* \), then we have \( T^* \geq \text{LB} \geq (1 - \epsilon) \cdot \text{UB} \), which leading to

\[
\text{UB} \leq (1 + \epsilon + o(\epsilon)) T^*.
\]

That is, the feasible solution produced by the proposed scheme will be within the \( \epsilon \)-range of the global optimal.

4. Fast Heuristic Algorithm

4.1. Overview

In this section, we present a fast heuristic algorithm for Problem OPT-TDLB. In Section 3, we observe that joint optimization with binary variables \( y_{ij} \)'s and continuous variables \( f_{ij} \)'s requires long execution time. To speed up computation, we first determine the network topology.
Algorithm 1: Branch-and-Bound Algorithm for Problem OPT-TDLB

// Initialization
1 Initialize δ* = ∅ and UB = ∞ ;
2 Initialize program list P with the original Problem 1 ;
// Relaxation
3 Obtain l-OPT-TDLB for Problem 1 ;
4 Solve l-OPT-TDLB to obtain solution δ = (y, f) and t objective value as the lower bound LBh ;
5 Select problem h with the minimum LBh from P ;
6 Set LB = LBh and let δ be its relaxed solution ;
// Local Search
7 Obtain feasible solution δ from δ with the local search algorithm ;
8 Compute UBh from (y, f) ;
9 if (UBh < UB) then
10 Update δ* = δ and UB = UBh ;
11 if (LB ≥ (1−ε)·UB) then
12 Stop with (1−ε)-solution δ* ;
13 else
14 Discard all problems h’ from P satisfying LBh ≥ (1−ε)·UB ;
15 end
16 end
// Partition
17 Find ˆyij that is closest to 1 but not fixed yet ;
18 Split problem h into h1 and h2 with respect to ˆyij ;
// Bounding
19 Solve the RLT relaxations of the two subproblems and obtain their lower bounds LBh1 and LBh2 ;
20 Remove problem h from P ;
21 if (LBh1 < (1−ε)·UB) then
22 Insert problem h2 into P ;
23 end
24 if (LBh2 < (1−ε)·UB) then
25 Insert problem h1 into P ;
26 end
27 if (P = ∅) then
28 Stop with the current best solution, δ* ;
29 else
30 Go to 5 ;
31 end

to minimize average network load, and then solve the multipath routing problem for load balancing. To further improve the optimality, we iteratively perturb the topology and then compute new network flows f, until some termination criterion is met.

The pseudo-code of the algorithm is given in Algorithm 2. As shown in the flowchart in Fig. 3, it consists of three parts: (i) initial topology design, (ii) multipath routing for load balancing, and (iii) topology perturbation. The algorithm iteratively perturbs the current topology by deleting and inserting FSO links, and computes the network flow for the s-d pairs based on the new topology. The network flow is derived by solving the LP relaxations l-OPT-TDLB(L/T1/T2) with fixed ˆyij values, thus the computation is very fast. If the new topology and network flow produce a better objective value for the original problem, the new solution will replace the existing one. Thus the objective function will be progressively improved over iterations. The algorithm terminates when no further perturbation can be made or when a prescribed
maximum number of iterations is reached. The three key components of the algorithm are described in detail in the remainder of this section.

4.2. Initial Topology Design

As discussed, the proposed algorithm iteratively improves the topology and network flow starting from an initial topology. A properly designed initial topology could speed up the convergence and achieve better solutions. In the following, we show how to choose \( m \) links from the candidate set \( \mathcal{E}_c \) to form an initial topology. The initial topology is represented by an adjacency matrix \( \mathbf{Y} = [y_{ij}] \). \( \mathbf{Y} \) is a 0-1 \( n \times n \) matrix, where \( y_{ij} = 1 \) if and only if link \((i, j) \in \mathcal{E}_c \) is included in the topology.

4.2.1. Prerequisite Information

Assume the traffic matrix \( \mathbf{F} \) is known. Let \( \mathcal{N} \) denote node information, such as node ID and location, and \( C^2_n \) represent FSO channel information. Then we can derive the reliability of the FSO links as in (2), and the set of candidate links \( \mathcal{E}_c \). If \( \gamma_{ij} \geq \gamma_{th} \), we have \( x_{ij} = 1 \) and link \((i, j) \in \mathcal{E}_c \); otherwise \( x_{ij} = 0 \). The candidate links form a topology with adjacency matrix \( \mathbf{X} = [x_{ij}] \). The total number of candidate links is \( m_c = |\mathcal{E}_c| = \sum_{i=1}^{n} \sum_{j=1}^{n} x_{ij} / 2 \). The minimum degree of each node can be derived as in (18) using traffic matrix \( \mathbf{F} \) and link capacity \( c \).

4.2.2. Minimum Traffic Matrix \( \mathbf{F}' \)

For a given topology, the traffic matrix \( \mathbf{F} \) is usually supported with both single-hop and multihop flows. If nodes \( s \) and \( d \) are not one-hop neighbors, the flow \( f_{sd} \) will be forwarded multiple times through multihop paths. The network traffic load associated with \( f_{sd} \) is thus proportional to the hop-counts of the \( s-d \) paths. We aim to have small hop-counts for \( s-d \) pairs with high traffic demands when determining the initial topology.

We consider both single-hop and multihop traffic patterns by constructing a minimum traffic matrix, denoted as \( \mathbf{F}' \). Assume \( K \) disjoint paths \( \{P^{k}_{sd}\}_{k=1}^{K} \) between nodes \( s \) and \( d \). The \( s-d \) traffic subflow on path \( k^{P}_{sd} \) is denoted as \( f_{sd}^{k} \), and the length (or, hop-count) of \( f_{sd}^{k} \) is denoted as \( l_{sd}^{k} \), for \( 1 \leq i \leq K \). We have \( f_{sd} = \sum_{k=1}^{K} f_{sd}^{k} \). For the total traffic load associated with flow \( f_{sd} \), denoted as \( \lambda_{sd} \), we have

\[
\lambda_{sd} = \sum_{k=1}^{K} l_{sd}^{k} \cdot f_{sd}^{k} \geq \min \{ l_{sd}^{k} | k = 1, 2, \cdots, K \} \cdot f_{sd} \equiv f_{sd}'.
\]

We next introduce a technique to compute \( \min \{ l_{sd}^{k} \} \), which is based on the adjacency matrix \( \mathbf{X} \) defined earlier. We have the following from graph theory [16].

**Fact 1.** Let \( \mathbf{A} \) be the adjacency matrix of graph \( G \). The number of walks from vertex \( s \) to \( d \) in \( G \) with length \( l \) is \( [\mathbf{A}^l]_{sd} \).

From Fact 1, the hop-count of the shortest path between nodes \( s \) and \( d \) can be found by identifying the smallest \( l \), such that \( [\mathbf{A}^l]_{sd} > 0 \). That is,

\[
\min_k \{ l_{sd}^{k} \} = l, \text{ if } [\mathbf{A}^l]_{sd} > 0 \text{ and } [\mathbf{A}^h]_{sd} = 0, \text{ for all } 1 \leq h < l.
\]

Once the hop-counts of the shortest paths are obtained, we can derive the minimum traffic matrix \( \mathbf{F}' \), with elements

\[
f_{sd}' = \min_k \{ l_{sd}^{k} \} \cdot f_{sd}.
\]

4.2.3. Link Selection

When the minimum traffic matrix is obtained, we next construct the initial topology based on \( \mathbf{F}' \) and the adjacency matrix \( \mathbf{X} \) representing all the candidate links. To minimize the network-wide average load \( L \), we choose the links that are on the shortest paths for all \( s-d \) pairs. We examine the elements of \( \mathbf{F}' \) in nonincreasing order, starting from the largest one, and choose links from \( \mathcal{E}_c \) (or, elements in \( \mathbf{X} \)) to insert into \( \mathbf{Y} \) until \( m \) edges are selected. If two nodes are adjacent according to \( \mathbf{X} \), the direct link will be inserted. In the case of multihop flows, multiple paths will be selected for rich connectivity and load balancing.

4.3. Multipath Routing for Load Balancing

When topology \( \mathbf{Y} \) is fixed, the binary optimization variables \( y \) are all determined. The LP relaxations \( l\text{-OPT-TDLB}(L/T_1/T_2) \) now only have continuous subflow variables \( f \). We then solve Problem \( l\text{-OPT-TDLB}(L/T_1/T_2) \) again with all the \( y_{ij} \)'s determined, to obtain multipath routing of the \( s-d \) flows \( f \) for this given topology.

To obtain the LP relaxations \( l\text{-OPT-TDLB}(L/T_1/T_2) \), the following two types of constraints are relaxed: (i) the binary variables \( y_{ij} \) are allowed to take real values in \([0, 1]\), and (ii) the nonlinear objective functions \( T_1 \) and \( T_2 \) are linearized with additional RLT bound factor constraints and polyhedral out approximation constraints [9]. Except for these two, all the other constraints (11)-(16) are preserved during the procedure. When the \( y_{ij} \)'s are all fixed, the \( f \) solved from the LP relaxations are also feasible to the original problem \( OPT-TDLB(L/T_1/T_2) \). That is, the
solution \( \{y,f\} \) is feasible to both the LP relaxations and the original problem. Substituting it into the objective function of OPT-TDLB(\( L/T_1/T_2 \)) yields an upper bound.

Clearly the optimality of the feasible solution \( \{y,f\} \) depends on the topology from which the solution is computed. It is possible to obtain a better solution with a different topology. We next perturb the current topology \( Y \) to get a new topology \( \bar{Y} \) to further improve the current solution \( \{y,f\} \), as discussed in the next subsection.

4.4. Topology Perturbation

We adopt a branch exchange algorithm similar to that in [17] for topology perturbation. This algorithm deletes a link from the current topology \( Y \). It then chooses a remaining link from the candidate set \( E_c \) and inserts it into the reduced topology. When deleting and inserting links, the degree constraints (11) and (13) should always be satisfied. The performance of this algorithm depends on the decision rules for link deletion and insertion. We adopt the following strategies in our algorithm.

- Link deletion: for objective function \( L \), we delete the link with the minimum load \( \lambda_{ij} \). For objective functions \( T_1 \) and \( T_2 \), we delete the link with the minimum delay. That is, delete the least used link from the current topology.
- Link insertion: for objective function \( L \), let the sum load at node \( i \) be \( \lambda_i = \sum_{j=1}^{n} y_{ij} \cdot \lambda_{ij} \). We insert the link \((i,j)\) with
  \[
  \arg \max \left\{ x_{ij} \cdot (\lambda_i + \lambda_j) \mid i \neq j, i, j \in V, y_{ij} = 0 \right\}.
  \]
  \( i \neq j, i, j \in V, y_{ij} = 0 \).

For objective functions \( T_1 \) and \( T_2 \), let the sum delay at node \( i \) be \( t_i = \sum_{j=1}^{n} y_{ij} \cdot t_{ij} \). We insert the link \((i,j)\) with
  \[
  \arg \max \left\{ x_{ij} \cdot (t_i + t_j) \mid i \neq j, i, j \in V, y_{ij} = 0 \right\}.
  \]

If no improvement is achieved by inserting such a link \((i,j)\), the link that achieves the second largest value will be inserted in the next iteration, and so forth.

The algorithm iteratively perturbs the topology and computes the network flows with the resulting new topology. It terminates with solution \( \{y,f\} \) when the maximum number of iterations is achieved or when no further perturbation can be made.

4.5. Remarks

The heuristic algorithm can be executed independently to solve Problem OPT-TDLB for an underlying FSO network, as discussed above. In addition, the heuristic is complementary to the RLT-based branch-and-bound algorithm developed in Section 3. In practice, the RLT-based branch-and-bound algorithm can be executed at relatively large time intervals (e.g., when significant changes occur), to provide new topology \( y \) and network flows \( f \) with guaranteed optimality. On the other hand, the heuristic algorithm can be kept on running or executed at more frequent intervals. It keeps on perturbing the topology generated by the branch-and-bound algorithm and computing new network flows, thus making the FSO network dynamically reconfigurable and adaptive to small timescale changes such as bad weather or fluctuation in the traffic matrix \( F \).

5. Simulation Studies

5.1. Simulation Setting

In the simulations, \( n \) BS’s are randomly deployed in a rectangular region. We present simulation results for cases \( n = 5, n = 7 \), and \( n = 15 \). Assume the traffic matrix \( F \) is given, which is randomly generated for the \( s \)-\( d \) pairs [8]. Each \( s \)-\( d \) flow \( f_{sd} \) ranges from 0 to 40% of the FSO link capacity. The constraints of minimum node degrees are determined from \( F \) as in (18).

Link connectivity is determined by link reliability, which is derived using the FSO channel model given in Section 2.2. Then \( x_{ij} \)'s are found and the candidate link set \( E_c \) is found. We use reliability threshold \( \gamma_{th} = 99.99\% \) when determining the candidate set. For LRD traffic, the index of dispersion \( a \) is set to a half of the link capacity and Hurst parameter \( H \) is chosen to be 0.7.

The proposed algorithms are implemented in Matlab ver 7.4.0 for manipulating matrices and solving the LP relaxations. The codes are executed on a standard PC with a Core Duo 2.20 GHz processor and 2 GB memory.

5.2. RLT-based Branch-and-Bound Algorithm

5.2.1. Convergence and Optimality Gap

The RLT-based branch-and-bound algorithm can provide \((1-\epsilon)\)-optimal solutions. We first examine the convergence performance and the final optimality gap achieved by the proposed algorithm.

In Fig. 4, we plot the simulation results for the 5-node FSO network. We use the RLT-based branch-and-bound algorithm to solve the three variations of Problem OPT-TDLB, i.e., Problem OPT-TDLB(\( L \)) with SRD traffic, Problem OPT-TDLB(\( T_1 \)), and Problem OPT-TDLB(\( T_2 \)). The termination criteria are chosen to be \( \epsilon = 0.01, 0.01, 0.07 \) for the three problems, respectively. The simulations terminate when \( LB \geq (1-\epsilon) \cdot UB \), and the upper bounding solution is a feasible one. The Problem OPT-TDLB(\( L \)) results are shown in Fig. 4(a), where the proposed algorithm terminates after 3 iterations and achieves an optimality gap of 0.78%. The Problem OPT-TDLB(\( T_1 \)) results are shown in Fig. 4(b), where the algorithm terminates after 4 iterations and achieves a 0.62% optimal gap. In the case of Problem OPT-TDLB(\( T_2 \)), the algorithm terminates after 7 iterations (due to the high variation in LRD traffic and the need to handle logarithm functions) with a 3.33% optimality gap, as shown in Fig. 4(c).
In Fig 5, we show convergence and optimality gap results for the 7-node FSO network. The proposed algorithm takes more iterations to finish, due to increased network size. For Problem OPT-TDLB(L), the algorithm takes 12 iterations to achieve an optimal gap of 0.33%, as shown in Fig. 5(a). For Problem OPT-TDLB(T1), the algorithm takes 7 iterations to achieve an optimality gap of 0.58%, as shown in Fig. 5(b). For Problem OPT-TDLB(T2), the algorithm takes 7 iterations to achieve an optimality gap of 4.91%, as shown in Fig. 5(c).
5.2.2. Computational Cost

We next present the computational cost results in the form of execution time of the proposed algorithm, which largely depends on the efficiency of the underlying LP solver on handling large matrices. Since the algorithm is an iterative one, we focus on the execution time per iteration for clarity.

In Table 1, the average parameter building time (i.e., the time spent on obtaining the LP relaxations and assembling the input matrices to the LP solver), denoted by $T_{\text{build}}$, and the average execution time (i.e., the time it takes for the LP solver to solve the relaxed problem) for solving one sub-problem in the Problem List $\mathcal{P}$ are listed, denoted by $T_{\text{exec}}$. In the table, $Q$ and $Q_{eq}$ are matrix inputs to the LP solver, whose sizes largely depend on the number of constraints of the LP relaxation $l$-OPT-TDLB.

We find that the building and execution times are proportional to the parameter matrix sizes. To compute the LB for a subproblem, the corresponding LP relaxation, $l$-OPT-TDLB, should be solved to determine the topology design variables $y$ and the multipath routing variables $f$. The constraints for $f_{ij}^{rd}$'s have more impact on the matrix sizes, due to new linear constraints introduced during the reformulation and linearization procedure for the objective functions. Once the lower-bounding solution $\delta$ is found, it takes negligible time for the local search algorithm to find an upper-bounding solution $\delta$ in the neighborhood, since the topology is already given in $\delta$. For example, the average execution time is approximately 14 s to get the LB for a subproblem of $l$-OPT-TDLB($T_2$), while it only takes 0.3 s to obtain the UB for the subproblem.

From these simulation studies, we find that the branch-and-bound algorithm can produce feasible solutions that are highly competitive, as indicated by the very small optimality gaps when the algorithm terminates. Although the global optimum is unknown, the guarantee on the optimality gap ensures near-optimal solutions. The tolerance $\epsilon$ provides a convenient handle for the trade-off between computation time and optimality. These are useful features for the design and control of FSO networks.

5.3. Fast Heuristic Algorithm

In this section, we present simulation studies on the performance of the proposed heuristic algorithm. We first examine the optimality performance of the proposed heuristic algorithm. Since the parameters (such as traffic matrix and node location) are not provided in prior works [7, 8], it is non-trivial to truthfully reproduce their results. Our strategy is to compare the heuristic algorithm with the RLT-based branch-and-bound algorithm developed in Section 3. Specifically, the lower and upper bounds provided by the branch-and-bound algorithm are good indicators of the global optimal for the joint topology design and load balancing problem. The gap between the heuristic solution and the branch-and-bound algorithm solution provides an upper bound on the optimality gap achieved by the heuristic algorithm.

In Fig. 6, we plot the network-wide average traffic load $L$ and average delays $T_1/T_2$ for the 7-node network. In the figures, the markers correspond to the iterations for the algorithms. When the objective function is $T_1$, the RLT-based branch-and-bound algorithm achieves an optimality gap 0.33% in 7 iterations, while the heuristic achieves an optimality gap 1.12% in 16 iterations. When the objective function is $T_2$, the RLT-based branch-and-bound algorithm achieves an optimality gap 4.91% in 7 iterations, while the heuristic achieves an optimality gap 4.33% in 16 iterations.

The convergence and optimality of the heuristic solutions are shown in Fig. 7 for the 15-node network. We also plot the lower bound for comparison purpose. It takes 76, 75, and 55 iterations, respectively, for the heuristic algorithm to achieve optimality gaps of 9.16%, 9.15%, and 8.16%, for the three problems, respectively.

We next compare the computation times of the proposed algorithms. In Fig. 6, the $x$-axis on the top marks the execution time of the heuristic algorithm, while the $x$-axis at the bottom are for the execution times of the RLT-based branch-and-bound algorithm. We find that the heuristic algorithm is considerably more computationally efficient than the RLT-based branch-and-bound algorithm. For example, in Fig. 6(a), the heuristic algorithm achieves an optimality gap of 1.11% in about 16 s, while the branch-and-bound algorithm achieves an optimal gap of 0.33% in about 2,100 s. The same observation can be made in all the simulation studies.

Finally we present the execution time of each iteration for the heuristic algorithm in Table 2. It can be seen that the building time is in the range of a few seconds and the execution time is in the range from tens to a few hundreds of ms. The largest building time 27.538 s occurs for the 15-node network under LRD traffic, which is due to the extra constraints introduced by the polyhedral outer approximation given in (33).

6. Related Work

This paper is closely related to the work on network planning [18, 19, 20]. Network planning problems usually belong to the class of combinatorial optimization problems. Since such problems are NP-hard, metaheuristics, e.g., simulated annealing [18] or genetic algorithms [20] are used to provide sub-optimal solutions. The main limitations of these approaches are the lack of performance guarantees and the relatively high complexity.

This work is also related to FSO research, which has attracted considerable interest recently. See excellent surveys
in [3, 21] and references therein. Major FSO research has focused on the PHY, such as hardware architecture [3] and optical channel modeling [10, 13]. There have been several works on the design [5, 22] and configuration [4, 7, 21, 23] of FSO networks. As in network planning, MILPs are usually formulated and various heuristic algorithms are proposed to provide sub-optimal solutions. Since such MILP-based solutions are usually centralized ones, a bootstrapping procedure is required to build a connected network from the set of initially isolated FSO nodes [4, 24, 25].

FSO links and traditional RF links are complementary to each with respect to data rate, interference, robustness, and range. Several papers have investigated the hybrid RF/FSO networks for enhanced performance [6, 20, 23, 26]. In [6], we investigate the design and optimization of a two-tier wireless network: the lower tier is a wireless mesh network that is partitioned into clusters based on traffic demand, while the upper tier is an FSO network with a mesh topology for which the algebraic connectivity is maximized. In [20], genetic algorithms are used to improve the capacity performance with a minimum number of hybrid FSO/RF gateways. In [23], the authors propose to adaptively adjust both transmission power (of RF and FSO transmitters) and the optical beamwidth, to meet prescribed QoS requirements. In [26], the authors investigate radio signal transmission over terrestrial optical wireless channels under a WiMAX network setting, and provide an outage probability analysis.

The “pseudo-wired” FSO links are highly desirable for interference management and security. As contrast to wired links, the FSO links also allows great flexibility for adaptive to network dynamics since they are steerable. In [27], the authors exploit slow-fading FSO channels and propose an adaptive transmission algorithm that can adjust transmit power and modulation according to channel status information feedback. In [28], the authors propose a fiber-bundle approach for beam steering to enhance the tolerance of optical link misalignment. In [29], we investigated the problem of maximizing the FSO network-wide throughput under the constraints of power budget and number of FSO transceivers, where cooperative relays are incorporated.

### 7. Conclusion

In this paper, we studied problem of joint topology design and load balancing in FSO networks. For given traffic demand, the objective is to design the topology and multipath routing policies for end-to-end flows, such that network-wide average load or network-wide average delay

---

### Table 1: Average Building and Execution Time per Subproblem for the Branch-and-bound Algorithm

<table>
<thead>
<tr>
<th>Problem Type</th>
<th>Size of $Q$</th>
<th>Size of $Q_{eq}$</th>
<th>$E[T_{build}]$ (s)</th>
<th>$E[T_{exe}]$ (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPT-TDLB($L$), LB, 5 nodes</td>
<td>$425 \times 440$</td>
<td>$131 \times 440$</td>
<td>0.835</td>
<td>0.710</td>
</tr>
<tr>
<td>OPT-TDLB($L$), UB, 5 nodes</td>
<td>n/a</td>
<td>$112 \times 252$</td>
<td>0.018</td>
<td>0.036</td>
</tr>
<tr>
<td>l-OPT-TDLB($T_1$), LB, 5 nodes</td>
<td>$445 \times 480$</td>
<td>$151 \times 480$</td>
<td>1.075</td>
<td>0.219</td>
</tr>
<tr>
<td>l-OPT-TDLB($T_1$), UB, 5 nodes</td>
<td>$12 \times 276$</td>
<td>$124 \times 276$</td>
<td>0.024</td>
<td>0.042</td>
</tr>
<tr>
<td>l-OPT-TDLB($T_2$), LB, 5 nodes</td>
<td>$733 \times 540$</td>
<td>$159 \times 540$</td>
<td>4.518</td>
<td>0.242</td>
</tr>
<tr>
<td>l-OPT-TDLB($T_2$), UB, 5 nodes</td>
<td>$216 \times 312$</td>
<td>$136 \times 312$</td>
<td>0.140</td>
<td>0.091</td>
</tr>
<tr>
<td>l-OPT-TDLB($T_1$), LB, 7 nodes</td>
<td>$1,813 \times 1,848$</td>
<td>$358 \times 1,848$</td>
<td>85.067</td>
<td>7.995</td>
</tr>
<tr>
<td>l-OPT-TDLB($T_1$), UB, 7 nodes</td>
<td>n/a</td>
<td>$318 \times 1,032$</td>
<td>1.296</td>
<td>0.081</td>
</tr>
<tr>
<td>l-OPT-TDLB($T_1$), LB, 7 nodes</td>
<td>$1,855 \times 1,932$</td>
<td>$400 \times 1,932$</td>
<td>93.516</td>
<td>8.963</td>
</tr>
<tr>
<td>l-OPT-TDLB($T_1$), -UB, 7 nodes</td>
<td>$24 \times 1,080$</td>
<td>$342 \times 1,080$</td>
<td>1.613</td>
<td>0.089</td>
</tr>
<tr>
<td>l-OPT-TDLB($T_2$), LB, 7 nodes</td>
<td>$2,543 \times 2,058$</td>
<td>$436 \times 2,058$</td>
<td>184.305</td>
<td>14.110</td>
</tr>
<tr>
<td>l-OPT-TDLB($T_2$), UB, 7 nodes</td>
<td>$432 \times 1,152$</td>
<td>$366 \times 1,152$</td>
<td>4.823</td>
<td>0.281</td>
</tr>
</tbody>
</table>

### Table 2: Average Building and Execution Times per Iteration with the Heuristic Algorithm.

<table>
<thead>
<tr>
<th>Problem Type</th>
<th>Size of $Q$</th>
<th>Size of $Q_{eq}$</th>
<th>$E[T_{build}]$ (s)</th>
<th>$E[T_{exe}]$ (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>l-OPT-TDLB($L$), 7 nodes</td>
<td>n/a</td>
<td>$318 \times 1,032$</td>
<td>1.296</td>
<td>0.081</td>
</tr>
<tr>
<td>l-OPT-TDLB($T_1$), 7 nodes</td>
<td>$24 \times 1,080$</td>
<td>$342 \times 1,080$</td>
<td>1.613</td>
<td>0.089</td>
</tr>
<tr>
<td>l-OPT-TDLB($T_2$), 7 nodes</td>
<td>$432 \times 1,152$</td>
<td>$366 \times 1,152$</td>
<td>4.823</td>
<td>0.281</td>
</tr>
<tr>
<td>l-OPT-TDLB($L$), 15 nodes</td>
<td>n/a</td>
<td>$500 \times 1,550$</td>
<td>5.350</td>
<td>0.131</td>
</tr>
<tr>
<td>l-OPT-TDLB($T_1$), 15 nodes</td>
<td>$50 \times 1,650$</td>
<td>$550 \times 1,650$</td>
<td>6.452</td>
<td>0.175</td>
</tr>
<tr>
<td>l-OPT-TDLB($T_2$), 15 nodes</td>
<td>$900 \times 1,800$</td>
<td>$600 \times 1,800$</td>
<td>27.538</td>
<td>0.476</td>
</tr>
</tbody>
</table>
can be minimized. We developed an effective branch-and-bound algorithm based on RLT for the formulated MILP and MINLP problems, which can provide highly competitive solutions with guaranteed performance. To reduce computation overhead, we also developed a fast heuristic algorithm with suboptimal solutions. The proposed algorithms are complementary to each other, for adaptation to network dynamics at both large and small timescales.
Acknowledgment

This work is supported in part by the National Science Foundation under Grant CNS-1145446, and through the Broadband Wireless Access & Application Center (BWAC) at Auburn University.

References


In Keun Son received his B.E. degree in Information Engineering from Korea Military Academy, Seoul, South Korea in 1997, M.S. degree in Computer Science from KAIST (Korea Advanced Institute of Science and Technology), Daejeon in 2005, and Ph.D. degree in Electrical and Computer Engineering from Auburn University, Alabama, USA in 2010. His research interests include network topology design and performance optimization in Wireless Radio Networks and Wireless Optical Networks. He is currently serving as an information technology specialist in the military of South Korea. Since 2001, he has been working on developing military C2 (Command and Control) systems as an intelligence system developer, an interoperability coordinator, and a system integrator. He became a manager of Korea Army C2 System Development Program in DAPA (Defense Acquisition Program Administration) in 2012.

Shiwen Mao received Ph.D. in electrical and computer engineering from Polytechnic University, Brooklyn, NY, USA (now Polytechnic Institute of New York University) in 2004. He was a research staff member with IBM China Research Lab from 1997 to 1998. He was a Postdoctoral Research Fellow/Research Scientist in the Bradley
Department of Electrical and Computer Engineering at Virginia Polytechnic Institute and State University (Virginia Tech), Blacksburg, VA, USA from 2003 to 2006. Currently, he is the McWane Associate Professor in the Department of Electrical and Computer Engineering, Auburn University, Auburn, AL, USA.


Dr. Mao is a coauthor of *TCP/IP Essentials: A Lab-Based Approach* (Cambridge University Press, 2004). He is a co-recipient of The IEEE ICC 2013 Best Paper Award. He received the 2013 IEEE ComSoc MMTC Outstanding Leadership Award and was named the 2012 Exemplary Editor of IEEE Communications Surveys & Tutorials. He was awarded the McWane Endowed Professorship in the Samuel Ginn College of Engineering for the Department of Electrical and Computer Engineering, Auburn University in August 2012. He received the US National Science Foundation Faculty Early Career Development Award (CA-REER) in 2010. He is a co-recipient of The 2004 IEEE Communications Society Leonard G. Abraham Prize in the Field of Communications Systems and The Best Paper Runner-up Award at The Fifth International Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness (QShine) in 2008. He also received Auburn Alumni Council Research Awards for Excellence–Junior Award and two Auburn Author Awards in 2011. Dr. Mao holds one US patent.

Sajal K. Das is a University Distinguished Scholar Professor of Computer Science and Engineering and the Founding Director of the Center for Research in Wireless Mobility and Networking (CReWMaN) at the University of Texas at Arlington (UTA). During 2008-2011, he served the US National Science Foundation (NSF) as a Program Director in the Division of Computer Networks and Systems. His research interests include wireless and sensor networks, mobile and pervasive computing, cyber-physical systems and smart environments, smart grids and sustainability, security and privacy, biological and social networks, applied graph theory and game theory. He has published extensively in these areas, including 3 books, and holds five US patents. A recipient of the IEEE Computer Society Technical Achievement Award in 2009 for pioneering contributions to sensor networks, Dr. Das is the founding editor-in-chief of Elsevier’s Pervasive and Mobile Computing journal.