Time-domain Wavelength Interleaved Networking with Wavelength Reuse

Carl Nuzman and Indra Widjaja
Bell Laboratories, Lucent Technologies
Murray Hill, NJ 07974, USA

***Abstract*— Time-domain Wavelength Interleaved Networking (TWIN) is a cost-effective network architecture that can provide fine-grained, flexible connectivity using passive optics in internal nodes. Each wavelength is dedicated to carry traffic to a single node. We introduce TWIN with wavelength reuse (TWIN-WR), an extension of TWIN that allows the number of nodes to exceed the number of wavelengths. We formulate an associated design problem, decompose it into subproblems, and provide algorithms for solving the subproblems. We experimentally demonstrate that our algorithms come close to upper bounds over a wide range of parameters, and show that the resulting designs are robust in the face of traffic variability and uncertainty. Analysis and experiment show that high efficiency can be maintained as long as the number of nodes does not exceed the square of the number of wavelengths, in principle allowing a network with thousands of nodes using only one hundred wavelengths.**

I. Introduction

Time-domain Wavelength Interleaved Networking (TWIN) is a recently proposed architecture in which the network operates like a giant input-queued switch [1][2]. The architecture is motivated by the fact that optical-electronic-optical (OEO) conversion and processing at each node, as in a packet-switched network for example, is costly especially as transit traffic accounts for much (usually around 70%) of the total traffic [3]. At the other extreme, pure optical circuit-switched (OCS) networks are also not cost effective when end-to-end traffic demand between each pair of nodes in a network is much lower than the wavelength capacity. OCS utilization can be improved through traffic grooming, but this requires OEO conversion and additional processing such as TDM or packet switching. TWIN strives for cost effectiveness by performing *optical* grooming at intermediate nodes and only using OEO conversion at the edge of the network, where it must already be present for interfacing with external clients such as routers, switches, etc.

In a TWIN network of arbitrary topology, each source has a burst assembler which collects data units from various external clients, assembles them into bursts and buffers the bursts for transmission to different destinations. Each source also has a fast-tunable laser for transmitting bursts optically. When a source has a burst to send to a given destination, the source tunes its laser to the wavelength uniquely assigned to that destination for the duration of the burst. When a destination receives a burst, it simply disassembles it into individual native data units upon conversion to the electronic domain and then forwards them to appropriate clients. Each source can talk to different destinations by tuning its laser to different colors, hence allowing fractional wavelength services for each pair of nodes. The simplicity of TWIN lies in burst forwarding at each intermediate node, where a crossconnect *passively* routes optical signals (bursts) based purely on the color of the burst. This *self-routing* property (no lookup tables) makes the core of the network function like a switch fabric of an input-queue switch, except that there may be substantial propagation delays between sources and destinations. Further, if the routes for each destination form an optical multipoint-to-point tree, contention at an intermediate node also results in contention at the intended destination node. This makes the core to be non-blocking and simplifies TWIN scheduling [4]. An important property of TWIN with respect to network planning is that it can support any traffic matrix by simple rescheduling at the edge without the network core having to participate in resource management or other related functions.

The number of nodes N in TWIN is limited by the number of available wavelengths W. If each wavelength has a capacity of R bps, the capacity of TWIN is NR bps. Although the capacity is limited by WR, we argue that this is sufficient for any backbone network. To see this, we first note that recent experiment on a fiber optic link achieved a capacity of 6.4 Tbps ($W = 160$ wavelengths and $R = 42.7$ Gbps) [5]. Moreover, [6] estimates the fundamental limit on the information capacity of a fiber link with spectral efficiency of 3bps/Hz to be about 150 Tbps. This suggests that the capacity of TWIN can be expected to grow much further with continued improvement in transmission technology. We also note that from a study by Andrew Odylzko [7], it would conservatively take only 800 Gbps of capacity to transport all the traffic on *all* U.S. backbones at year-end 2002 (about the same time the experiment [5] was conducted).

Although there is a strong argument that the capacity of TWIN is sufficient for practical purposes, the number of nodes supported in a network may pose a limiting factor. In particular, there are situations where the number of nodes is greater than the number of available wavelengths; that is, $N > W$. In this paper, we focus on such networks which we call *TWIN with wavelength reuse* (*TWIN-WR*).

While a source in TWIN can send traffic directly (i.e., optically end-to-end) to any destination, a source in TWIN-WR may not be able to do so. We propose to add a simple gateway function at each node to act as a relay. The gateway acts like a router which examines the destination address of

each burst and if required, initiates a new wavelength for direct transmission to the next hop, and so on until the burst reaches its destination. In practice, the tranceiver in an ordinary TWIN node would normally be connected to client interfaces via a packet switch, in order to support hairpinning (direct forwarding of traffic between clients). The same packet switch could be used to provide the gateway functionality between receiver and transmitter in a TWIN-WR node; hence the additional hardware cost required for a gateway is negligible. The primary cost incurred by introducing multiple hops is an increase in the effective traffic load. This cost is quantified in Section V.

We formulate the problem of designing a TWIN-WR network as a mixed integer linear program. For efficient computation, we propose an optimization problem that is separated into three subproblems that are solved sequentially. The first subproblem assigns W colors to N nodes with the objective of maximizing the throughput of direct traffic, also called the single-hop throughput. After the colors have been assigned, the second subproblem deals with the design of a multipoint-to-point tree for each destination, taking into account physical-layer constraints. We rely on approximate algorithms which are efficient in the above subproblems. The output of the second subproblem is a virtual topology. Unlike traditional virtual topologies that have link capacity constraints, TWIN-WR virtual topology has the capacity constraint at the node. Finally, the third subproblem routes the flows on the virtual topology. In practice, the output of the third subproblem and possibly with the actual traffic flow would be fed to a scheduling algorithm, which can be centralized [4] or distributed [8], to coordinate the tunable lasers. The scheduling problem for TWIN-WR, however, is beyond the scope of this paper.

We evaluate the effectiveness of our design with different values of N and W, different physical topologies, different traffic matrix characteristics, and changing traffic patterns. Our main results indicate that:

1) The first two subproblems achieve close to the optimal single-hop throughput.
2) A large network with N nodes can be built with a relatively small number of wavelengths $W \approx \sqrt{N}$ with about 50% of gateway switching capacity devoted to transit traffic.
3) The virtual topology can be designed for a predicted traffic matrix without sacrificing robustness to traffic variability.

The rest of the paper is organized as follows. In Sec. II, we formulate the design of a TWIN-WR network as an optimization problem. In Sec. III, we consider the problem of assigning colors to destinations. In Sec. IV, we propose an algorithm that designs a tree for each destination, resulting in a virtual topology for the entire network. We present flow routing, analyze the scalability of the network as N increases and investigate the robustness as traffic patterns change in Sec. V. We then survey related work in Sec. VI. Finally, we conclude the paper in Sec. VII.

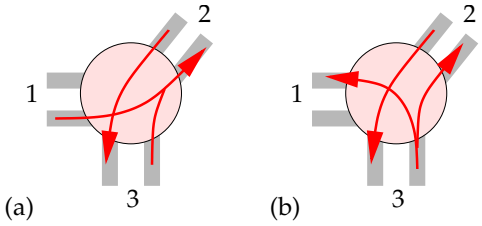


Fig. 1. 3-port merge-capable crossconnect in (a) a valid configuration, (b) an invalid configuration.

II. Model Formulation

A. Preliminaries

To understand the capabilities and constraints of TWIN crossconnects, consider Fig. 1, where a crossconnect contains a set of M input ports and M output ports (here $M = 3$) that terminate fibers. Typically one pair of ports connect to the local transmitter and receiver for the node, while the remaining ports connect to crossconnects in other nodes. The arrows indicate the flow of light on a particular wavelength channel. The crossconnect is wavelength-selective and merge-capable such that incoming wavelength w on input port i is always directed to a particular output port j according to a mapping $j = f(i, w)$. Any choice of output port can be made for each input port/wavelength pair (i, w), regardless of the choices made for other pairs; in particular, multiple input pairs may choose the same output, as shown in Fig. 1(a). For a given wavelength, the state of the crossconnect may be represented by an $M \times M$ array, where the ij element is non-zero if input port i connects to output port j. The array for the wavelength depicted in Fig. 1(a) would be

$$\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}. \quad (1)$$

In an ordinary crossconnect, the array is a permutation, containing a single 1 in each column and row. For a merge-capable crossconnect, there is a single non-zero entry in each row, but there can be multiple 1's in each column. Fig. 1(b) depicts a configuration which is not allowed, in which light entering the third input port on one wavelength is split between two output ports; the corresponding matrix would have two entries in the third row. Implementations of merge-capable crossconnects were discussed in [1].

The *physical topology* of a TWIN-WR network consists of a set of nodes connected by fibers. Each node contains a merge-capable optical crossconnect, as well as a transmitter/receiver pair (transceiver). An example network is illustrated in Fig. 2. Each transmitter is tunable to any of W wavelengths, and thus may be the source for W (time-shared) lightpaths. In this case, $W = 2$ wavelengths are depicted using solid and dashed lines, respectively. The crossconnects are configured such that (i) light emitted from each transmitter on each wavelength eventually finds its way to some receiver and (ii) each receiver receives light from only one wavelength. Because the crossconnect associated with a receiver acts as

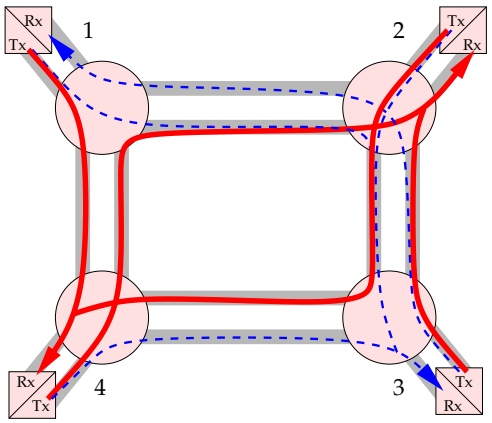


Fig. 2. Physical topology and wavelength trees with $N = 4$ and $W = 2$.

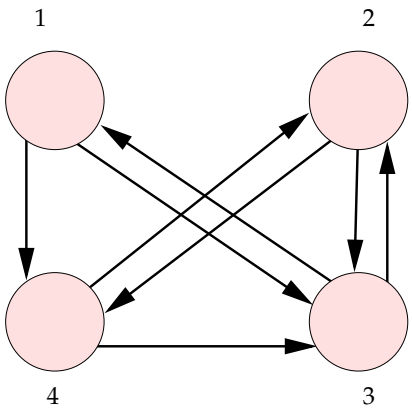


Fig. 3. Virtual topology derived from Fig. 2.

a wavelength filter, the receiver itself may be wavelength agnostic. In other words, each receiver is the root of a directed tree formed by merging all lightpaths of the same wavelength that lead to that receiver. The transmitters that form the leaves of the tree can communicate directly with the root on the specified wavelength. For example, node 1 in Fig. 2 can communicate directly with node 3 on the dashed wavelength and with node 4 on the solid wavelength. The transmissions from each transmitter are scheduled to avoid collisions at the receiver. The scheduling problem is an extension of the scheduling problem considered in [4]; for the purposes of this paper, we assume that the scheduling problem can be handled with reasonable efficiency.

The network described above defines a *virtual topology* between nodes that is visible to higher layers. The virtual topology is a directed graph on the nodes, where the graph contains an arc from node i to node j if the transmitter at node i can transmit information directly to the receiver at node j; that is, if node i is a leaf on the tree rooted at node j. Arcs need not be distinguished by wavelength in the virtual topology. In the virtual topology, each node has an out-degree of W, while the in-degree may vary. Note that in TWIN, the number of wavelengths is equal to the number of nodes, so that the out-degree and in-degree are $(W-1)$ and the virtual topology is a fully meshed graph. The virtual topology corresponding to the physical topology in Fig. 2 is depicted in Fig. 3. When node 1 wants to send traffic to node 2, it can first transmit to node 3 or node 4, which then relays the traffic to node 2.

For a given traffic matrix, the goal is to design a virtual topology compatible with the underlying physical layer, and flow routes compatible with the virtual topology that are capable of carrying the traffic. In reality, the virtual topology would be relatively static, while the traffic patterns may change in time. Thus, an important objective could be to design a virtual topology that would be well-adapted to changes in traffic patterns. Case studies in Sec. V indicate that our virtual topology design tends to be robust to large changes in traffic patterns.

B. Mixed Integer Linear Program Formulation

We formulate the overall design problem for a given traffic matrix as a mixed integer linear program. Our goal in setting out this formulation is not to provide a computational tool, but rather to provide a concrete problem definition.

We assume that the physical topology is given in the form of an directed graph $G(V, A)$, where V is the set of nodes and arcs A correspond to fibers connecting nodes. In this paper, we assume that the arcs always come in bi-directional pairs, although this is not necessary. Without loss of generality, it is convenient to assume that each node contains an optical crossconnect and a transceiver with a gateway for admitting local add/drop traffic and forwarding transit traffic. The set of wavelength channels available is denoted by $\mathcal{W}$. The nominal (directed) traffic load from source node $i \in V$ to destination node $j \in V$ is denoted by t_{ij} and let $T = [t_{ij}]$. The traffic is normalized according to the data rate of a single wavelength channel, so that $t_{ij} = 1.0$ consumes the capacity of one channel.

We also require a set $\mathcal{P}$ of potential lightpaths between nodes. Each element of $\mathcal{P}$ is a loop-free sequence of nodes $(v_1, v_2, \ldots, v_h)$ of length at least two, where each consecutive pair of nodes (v_i, v_{i+1}) must correspond to an arc $a \in A$. The set need not contain all possible paths; for example, it could be restricted to sequences of a given maximum length, or it could be restricted by engineering rules imposed by the physical layer system. We denote by $\mathcal{P}_j \subset \mathcal{P}$ the set of paths which end at node j.

The process of configuring the crossconnect is essentially equivalent to choosing a wavelength for the tree rooted in each destination node (receiver), and deciding which of the paths ending at that node will be included in the tree. For each $j \in V$ and $w \in \mathcal{W}$, define binary variables u_j^w which are set to one (zero otherwise) if the tree rooted at j is assigned to wavelength w. Since there is only one wavelength per tree, we have the constraint

$$\sum_{w \in \mathcal{W}} u_j^w = 1, \quad \forall j \in V \tag{2}$$

For each path $p \in \mathcal{P}$ and each wavelength $w \in \mathcal{W}$, we define binary variables $\gamma^w(p)$ which are set to one if the path is to

be included in the tree, using wavelength w. To ensure that all paths ending at node $j \in V$ use the wavelength specified by the values u_j^w, we have the constraint

$$\sum_{p \in \mathcal{P}_j} \gamma^w(p) \leq K u_j^w, \tag{3}$$

where K is a sufficiently large constant such as $|\mathcal{P}|$. To enforce crossconnect constraints, we need variables representing the crossconnect state in each node. For node j, let Γ_j^{in} and Γ_j^{out} represent the set of input and output ports, respectively. One input (output) port, denoted by $(+)$ $((-))$, is the add (drop) port connecting to the local transmitter (receiver), while the remaining input (output) ports correspond to arcs of the form (i,j) $((j,k))$. The crossconnect state is then denoted by binary variables $m_j^w(e,f)$ where $j \in V$ is a node, $w \in \mathcal{W}$, and where $e \in \Gamma_j^{\text{in}}$ and $f \in \Gamma_j^{\text{out}}$ range over the input and output ports, respectively. For fixed j and w, $m_j^w(\cdot,\cdot)$ is simply the binary matrix exemplified by (1) for the node depicted in Fig. 1. That is, $m_j^w(e,f) = 1$ if and only if light coming into node j on input port e with wavelength w is directed to output port f. There must be exactly one output port assigned to each input port and wavelength, hence the constraint

$$\sum_{f \in \Gamma_j^{\text{out}}} m_j^w(e,f) = 1, \quad \forall j \in V, w \in \mathcal{W}, e \in \Gamma_j^{\text{in}}. \tag{4}$$

One would like to include as many paths as possible in the trees; the constraints on $\gamma^w(p)$ stem from the fact that the paths that are included must be compatible with the routing state in each crossconnect. To express these constraints, denote by $\mathcal{P}_j(e,f)$ the set of paths which enter node $j \in V$ on port e and leave it on port f. Then to each variable $m_j^w(e,f)$ corresponds the crossconnect constraint

$$\sum_{p \in \mathcal{P}_j(e,f)} \gamma^w(p) \leq K m_j^w(e,f). \tag{5}$$

The interpretation of this equation is that the number of paths that take a particular route through the crossconnect must be zero if the crossconnect state does not support that route, and the number of paths is unlimited otherwise.

The above variables and constraints define the space of valid wavelength trees for a given physical topology. The resulting virtual topology is described by binary variables q_{ij}, where $q_{ij} = 1$ if and only if there is an arc from $i \in V$ to $j \in V$. The values of these variables are determined from $\gamma^w(p)$ as

$$q_{ij} = \sum_{p \in \mathcal{P}_j^i} \sum_{w \in \mathcal{W}} \gamma^w(p) \tag{6}$$

where $\mathcal{P}_j^i \subset \mathcal{P}_j$ is the set of paths which originate at source $i \in V$ and terminate at destination $j \in V$. The preceding constraints guarantee that $\gamma^w(p) = 1$ for at most one distinct path in $\mathcal{P}_j^i$.

The remaining variables follow a standard multicommodity flow program, except that the constraints are on receivers rather than links. For $s,d,i,j \in V$, let real variables $f_d^s(i,j)$ denote the amount of traffic from source s to destination d which flows across arc (i,j) in the virtual topology. The flow must satisfy the usual flow constraints

$$\sum_{i \in V} f_d^s(i,k) - \sum_{j \in \mathcal{V}} f_d^s(k,j) = \begin{cases} -t_{sd} & k = s \\ t_{sd} & k = d \\ 0 & k \in V \backslash \{s,d\} \end{cases}$$

for each $s,d,k \in V$. Moreover, the flow should be restricted to links in the virtual topology, by the constraint

$$\sum_{s,d \in V} f_d^s(i,j) \leq \bar{K} q_{ij}, \quad \forall i,j \in V,$$

where $\bar{K} \geq \sum_{s,d} t_{sd}$ is a large positive constant.

Because the traffic is normalized by the wavelength data rate, the total flow into each receiver should be limited to unity. We define the congestion C to be the maximum utilization of any destination in the network. For each destination $j \in V$, the congestion thus satisfies

$$\sum_{s,d,i \in V} f_d^s(i,j) \leq C$$

Finally, the overall objective is to minimize C, subject to all of the preceding constraints. If the minimum value of C is greater than one, then it is not possible to meet the given traffic matrix with the given network.

C. Decomposition

There are a number of ways to approach the design problem defined above. In this paper, we propose to break the problem into three separate subproblems, namely, color assignment, tree design and flow routing. The color assignment subproblem assigns a wavelength to the receiver in each node (destination), as represented by the variables u_j^w. The tree design subproblem chooses a feasible virtual topology, compatible with the physical network and the color assignment, effectively determining the variables $\gamma^w(p)$, $m_j^w(e,f)$, and q_{ij}. Since the congestion measure C cannot be used in the first two subproblems, we choose a natural proxy for the optimization objective, which is to maximize the single-hop throughput $\eta = \sum_{ij} q_{ij} t_{ij}$ or the amount of traffic associated with all the arcs in the virtual topology. Traffic that cannot be carried in a single hop will traverse multiple hops through the gateways. Finally, the flow routing subproblem minimizes the congestion C for the fixed virtual topology q_{ij} determined in the tree design subproblem. These three subproblems are described in the subsequent sections.

III. Color Assignment

The goal of color assignment is to determine the color (wavelength) w for each destination j, such that the single-hop throughput is maximized. We assume the non-trivial case where $W = |\mathcal{W}| < N = |V|$ so that multiple destinations need to be assigned the same color.

Note that the single-hop throughput is computed by summing the traffic matrix elements t_{ij} over all pairs in the virtual

topology. The first constraint which limits this sum is that each node can only send traffic directly to W neighbors; equivalently, only W elements may be selected from each row of the traffic matrix. A second constraint is that each of the W receivers listening to a given transmitter must be assigned different wavelengths. Thus we may think of each column of the traffic matrix having an assigned color, and in each row, one element of each color may be selected. Finally, there are additional constraints related to routing which are imposed by the particular physical network topology.

We consider a color assignment subproblem that assigns W distinct colors to the destinations in the network, with the objective of maximizing the single-hop traffic subject to only the first and second constraints above. For a given color assignment, the single-hop traffic is then maximized by always choosing the largest traffic element t_{ij} among all elements of the same row and having the same destination color. Below we describe an optimal color assignment using an integer linear program formulation and a simple and efficient heuristic (called color merging).

A. Optimal Color Assignment

Our formulation is a streamlined version of the overall design problem. Binary variables u_j^w as before encode whether or not the j-th receiver uses wavelength w. We replace the notion of a path p with a source destination pair (i, j), and define binary variables $\gamma^w(i, j)$ which encode whether or not pair (i, j) is included in the virtual topology using wavelength w. As before binary variables q_{ij} indicate whether or not pair (i, j) is included using any wavelength. The optimal color assignment problem is thus

$$\text{maximize} \sum_{i,j \in \mathcal{T}} t_{ij} q_{ij}$$

subject to

$$\begin{aligned} \sum_{w \in \mathcal{W}} u_j^w &= 1 \\ \sum_{i \in V} \gamma^w(i,j) &\leq |V| u_j^w \\ \sum_{j \in V} \gamma^w(i,j) &= 1 \\ \sum_{w \in \mathcal{W}} \gamma^w(i,j) &= q_{ij}. \end{aligned}$$

The four constraint equations are analogous to (2),(3),(4), and (6), respectively.

B. Color-Merging Algorithm

The main idea of color merging is to start with N unique colors and then iteratively merge two groups of nodes into one until there are W groups left. Here a group is defined as a set of nodes with the same color. When two groups are merged, single-hop throughput is reduced since each source can only choose to send its traffic to one member of each group. A simple algorithm is to greedily pick two groups that minimize throughput loss at each iteration. The color merging algorithm is described as follows.

```
Algorithm Color_Merging
Input: N, T, W
Output: {w : u_j^w = 1}, ∀j ∈ V
```

1) Assign each node with a unique color $w \in \bar{\mathcal{W}}$, where $\mathcal{W} \subset \bar{\mathcal{W}}$ and $|\bar{\mathcal{W}}| = N$, and let variable $n \leftarrow N$.
2) Merge two receiver groups of colors w and w' into the same group with color w that result in minimum loss of traffic, and update $n \leftarrow n - 1$.
3) Go to step 2 if $n > |\mathcal{W}|$.

Upon termination of the above algorithm, the receivers may be assigned a set of wavelengths that are different from $\mathcal{W}$ although they have the same cardinality. This can be easily adjusted by remapping the colors into ones that are part of $\mathcal{W}$.

Performing step 2 for the first time requires $\mathcal{O}\left(N^3\right)$ operations, since there are $N(N-1)$ pairs to consider, and $\mathcal{O}\left(N\right)$ operations are required to compute the loss of merging a given pair. If the pair-wise losses are stored, then only $\mathcal{O}\left(N^2\right)$ operations are required for each subsequent iteration, and the total time complexity is $\mathcal{O}\left(N^3\right)$.

Fig. 4(a) gives an example of how the color merging algorithm works for a 5-node network with three wavelengths. Each entry $(row, column) = (i, j)$ denotes the traffic load t_{ij}. The algorithm starts with five colors and ends with three colors after two iterations. An entry (i, j) that is not crossed out indicates that node i is allowed to transmit directly to node j. Fig. 4(b) shows the result of the optimal color assignment. Since the optimal color assignment is impractical for large N, we also consider an upper bound that is easily computed. Fig. 4(c) shows an upper bound result obtained by selecting the highest W values for each row in the traffic matrix. In this example, the single-hop throughputs are 1.267, 1.31 and 1.334 for color merging, optimal color assignment and upper bound, respectively. Obviously, the upper bound result in this case is not feasible.

C. Performance Comparison

To evaluate the effectiveness of the color-merging algorithm, we perform an experiment that involves different traffic characteristics, different number of wavelengths and different network sizes.

The traffic matrices used in this study were drawn from a random distribution defined by four parameters: the number of nodes n, the average amount of traffic μ, the variation v, and the maximum row or column sum p. For $i \neq j$, the nominal traffic level from node i to node j is an independent random variable of the form

$$t_{ij} = s * |v * Z + 1|$$

where Z is a zero mean, unit variance Gaussian random variable and where s is chosen so that the total expected traffic $(n-1)E[t_{ij}]$ is equal to μ.

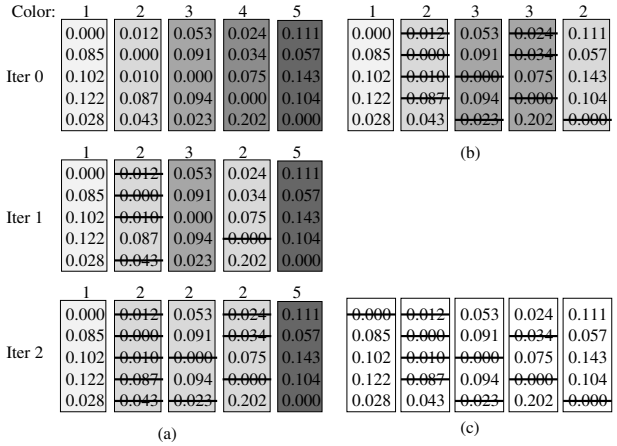


Fig. 4. Color assignment with: (a) color merging (b) optimal coloring and (c) upper bound.

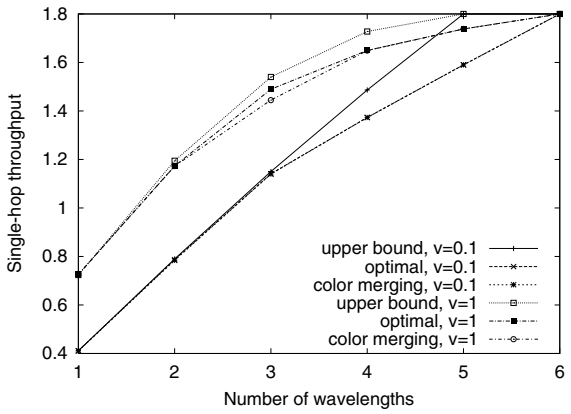


Fig. 5. Color merging achieves throughput very close to optimal while the upper bound is generally too optimistic.

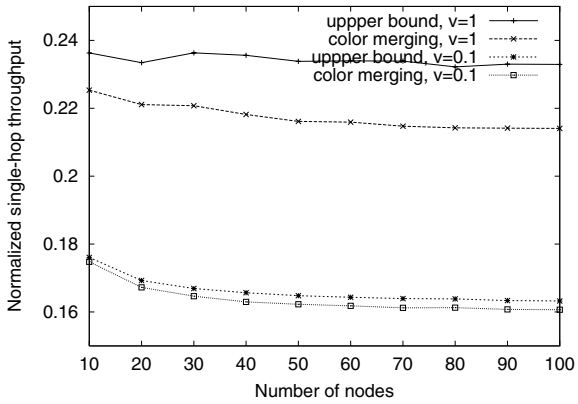


Fig. 6. The throughput gap between color merging and upper bound, which is not feasible in general, is within 8%.

When v is below about 1/2, the t_{ij} are approximately Gaussian, with coefficient of variation v. For larger values of v, the application of the absolute value affects the distribution and limits the coefficient of variation of t_{ij} to a maximum of about 0.8. If, for the nominal traffic matrix $\{t_{ij}\}$, any of the row or column sums exceed p, then an iterative algorithm is applied to enforce this constraint. Any rows whose sums exceed p are scaled down to be exactly equal to p, while the remaining rows are scaled up by a common factor in order to keep the total traffic constant. The same operation is performed on the columns, and the process is iterated until converging on a traffic matrix with the desired total traffic, with no row or column sum exceeding p.

Fig. 5 plots the single-hop throughputs with color merging, optimal color assignment and upper bound as the number of wavelengths W is varied for a relatively small network ($N = 6$). The plot shows two cases with fairly uniform traffic ($v = 0.1$) and highly non-uniform traffic ($v = 1$). We use $\mu = 0.3$ and $p = 1$ throughout. The results are obtained by averaging over 50 independently generated traffic matrices. Observe that the throughput of the color-merging algorithm is virtually identical to that of the optimal algorithm when the traffic is fairly uniform ($v = 0.1$). This is because merging of any two colors would result in approximately the same loss of throughput. When the traffic is highly non-uniform ($v = 1$), the largest discrepancy approximately occurs when $W \approx N/2$ while the throughputs of both would be identical when $W = 1$ and $W = N$. Observe also that the upper bound provides a fairly tight approximation for small values of W, but results in a very poor approximation when W is large. In fact, when $W = N-1$, the upper bound can always select exactly $N-1$ non-zero entries for each row in the traffic matrix (assuming that a node never transmits to itself) and thus incurs no loss of throughput. This upper bound result is, however, not realizable in general.

Fig. 6 plots the *normalized* single-hop throughput as N increases while keeping $W = N/2$ as this is where color merging yields the largest gap compared to optimal coloring. Here, we can only compare color merging with the upper bound since the optimal color assignment becomes computationally prohibitive. We recall from Fig. 5 (and other runs not shown) that the upper bound provides a looser approximation with $W \approx N/2$ when the traffic becomes more non-uniform. It is worth noting that even with this consideration, the gap between color merging and the upper bound in Fig. 6 is less than 8% even for highly non-uniform traffic ($v = 1$).

IV. Tree Design

In this section, we present the design of a multi-point to point tree for each destination, assuming that each node has already been colored. We use the single-hop throughput as the objective function. In terms of the variables defined in Sec. II, we seek to maximize $\sum t_{ij} q_{ij}$ for fixed u_j^w, subject to constraints (3), (4), (5), and (6).

We take a sequential, heuristic approach to the tree design problem. At any stage of the algorithm, the tree design is partially completed. In each iteration, a candidate node pair is selected, and then an attempt is made to find a lightpath for that pair which is compatible with the existing tree. If this attempt is successful, the new path is added to the tree and

the new arc is added to the virtual topology. The process is iterated until all nodes have an outdegree of W in the virtual topology, at which point the tree design is complete.

As long as the outdegree of some node i is less than W, it is always possible to add a lightpath. Suppose that node i is not yet the source of a lightpath of wavelength w; then it is possible to build a lightpath to one of the receivers assigned to this wavelength. To see this, choose any path from node i to one such node j. If this path does not conflict with the existing trees of wavelength w, then the lightpath from i to j may be added. Otherwise, define a new lightpath by merging onto the existing tree at the first point of conflict. This new path must end at some receiver j' using wavelength w. Our method for finding the route of a lightpath is to look for the shortest path in an auxiliary graph, described below.

A. Auxiliary Graph

The auxiliary graph enforces the crossconnect constraints, in the sense that any path of finite cost in the auxiliary graph corresponds to a lightpath compatible with the existing wavelength trees. Conversely, if a compatible lightpath exists for a given node pair, then one will be found by the shortest path algorithm. The concept of auxiliary graphs also has found other applications in Routing and Wavelength Assignment (RWA) problems [9]-[12].

Let a TWIN-WR network be represented by a directed graph $G(V, A)$, where V is the set of vertices representing the nodes and A is the set of directed edges representing fibers connecting nodes. Each edge $(i, j) \in A$ has a weight c_{ij}. We create an auxiliary graph $G'_w(V', A')$ for each wavelength channel $w \in \mathcal{W}$, where each vertex represents an input (or output) port of a TWIN node on wavelength channel w, or an add (or drop) port of a TWIN node. The add and drop ports are used to represent the transmitter (tunable laser) at the source node and receiver at the destination node, respectively. The transformation from G to G' can then be described as follows (we shall drop w since the transformation is independent of w). For each edge $(i, j) \in A$ where $i, j \in V$, we create a corresponding (external) edge $(v_i^{out}(j), v_j^{in}(i)) \in A'$ with weight c_{ij}, where $v_i^{out}(j) \in V'$ denotes the output port at node i that is connected to node j and $v_j^{in}(i) \in V'$ denotes the input port at node j that is connected to node i. For each node $k \in V$, we create an (internal) edge $(v_k^{in}(i), v_k^{out}(j)) \in A'$ with weight $c_k(i, j)$, for each node pair (i, j) that are connected to node k. The weight $c_k(i, j)$ is initially set to 0 to allow for any configuration. For each node k, we also create an edge $(v_k^{in}(+), v_k^{out}(j))$ to represent the link from the add port (transmitter) to $v_k^{out}(j)$ for each node j connected to node k, and an edge $(v_k^{in}(i), v_k^{out}(-))$ to represent the link connecting $v_k^{in}(i)$ to the drop port (receiver), for each node i connected to node k. The weights $c_k(i, -)$ and $c_k(+, j)$ are also initially set to 0. Fig. 7 shows an example of an original graph and the corresponding auxiliary graph. There are two wavelengths, indicated by solid and dashed lines respectively. Notice that after a dashed path from source 2 through nodes 4 and 3 to destination 5 has been established, $c_3(4, -)$ would be set to ∞. Thus, a direct dashed path from source 4 to destination 3 would be prohibited and the path 4, 2, 1, and 3 would incur a lower cost.

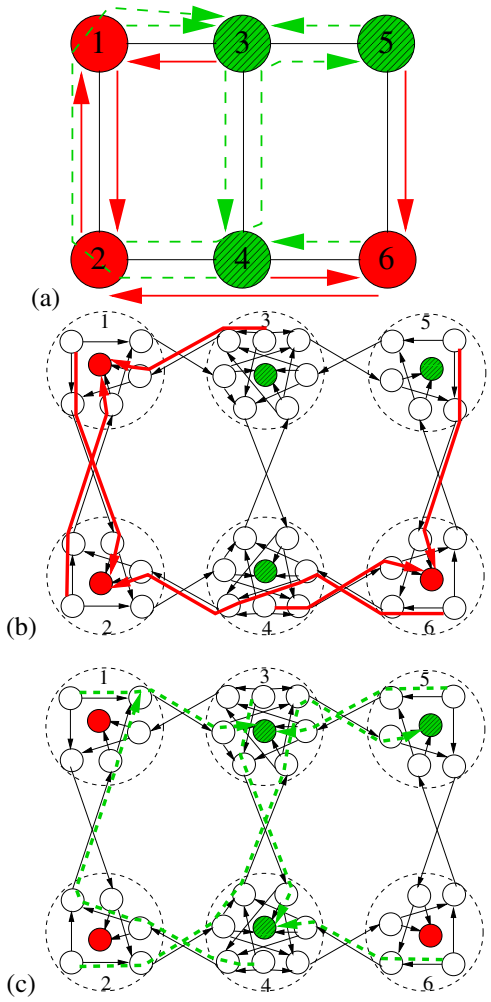


Fig. 7. Graph transformation: (a) original graph G; (b) auxiliary graph G' for solid wavelength; (c) auxiliary graph G' for dashed wavelength.

The auxiliary graph can be used to enforce certain constraints. For example, suppose a directed path of a given wavelength traverses nodes i, k and j. To prevent wavelength splitting at node k, the internal weights would have to be modified to $c_k(i, j') = \infty$ for each output port $j' \neq j$ and the drop port $j' = -$. Note that in a normal permutation cross-connect, $c_k(i', j)$, for each $j' \neq i$, would also be modified to ∞ to disallow merging. Besides enforcing constraints, the auxiliary graph can also be used to minimize resource usage in constructing a tree. A typical way is to set up each shortest path independently using static c_{ij} for each edge (i, j). However, this approach does not account for the fact that when a new path joins an existing tree, the path does not consume additional resources on the segment that is part of the tree. This fact can easily be accounted for by setting c_{ij} to 0 when a new path traverses link (i, j) has been established.

B. Algorithm

In tree design, various criteria could be used in selecting the next node pair to add in each iteration. One approach is to minimize throughput loss by choosing node pairs in order of decreasing traffic load t_{ij}. A randomized version would choose the next pair at random, with probability proportional to t_{ij}. By itself, either approach unfortunately will not ensure that the virtual topology is connected. To guarantee a connected virtual topology, we perform an initialization step which divides the graph into W connected subgraphs, based on destination wavelength. Suppose that there are $m_w > 1$ nodes with color w. By transmitting on wavelength w, each such node can connect to one of the remaining $m_w - 1$ nodes; arranging these connections in a cycle, all m_w nodes form a connected subgraph of the virtual topology. At the end of the tree design algorithm, each node in the virtual topology has W outgoing arcs, which connect the node to exactly one node of each color. Hence each node is connected to each of w connected subgraphs that cover the virtual topology.

The overall tree design algorithm is described below.

Algorithm Tree_Design
Input: $G(V, A)$, T, $\{w : u_j^w = 1\}$, $\forall j \in V$
Output: q_{ij}, $\forall i, j \in V$.

1. For each color w, set up a cycle of lightpaths between the nodes of this color, and update the auxiliary graph accordingly. The node pairs in the cycle are now considered to be routed, and no pairs have been rejected.
2. Among node pairs which have not been routed or rejected, choose the pair (i, j) with the largest traffic value t_{ij}.
3. Find the least-cost path from the transmitter at i to the receiver at j in the auxiliary graph. If the cost is finite, add (i, j) to the virtual topology, and update the auxiliary graph to reflect that (i, j) has been routed. Otherwise, mark the node pair as rejected.
4. Repeat from step 2 until all node pairs have been routed or rejected.

The time complexity of the tree design algorithm is dominated by the shortest-path computations in step 3. The auxiliary graph contains approximately $\hat{N} := 2(D + 1)N$ vertices, where N and D are the the number of nodes and average degree of the physical topology, respectively. Finding the shortest path between a pair of nodes with Dijkstra's algorithm requires $\mathcal{O}\left(\hat{N}^2\right)$ operations. Step 3 may be performed at most N^2 times, and hence the overall complexity is $\mathcal{O}\left(D^2N^4\right)$.

Remark: To form the cycle in step 1, it is not necessarily possible to connect the nodes in an arbitrary order. However, there is always a feasible order, which may be found using heuristics for the traveling salesman problem.

C. Experiments and Results

To generate the physical topologies, we adopt the Waxman model [13], where the probability that there is an undirected edge between i and j is given by

$$P(i, j) = \alpha e^{-d/\beta L}, \tag{7}$$

where α and β are model parameters in the range $(0, 1]$, L is the maximum distance between two nodes and d is the Euclidean distance between i and j. The locality parameter β controls the extent to which links are preferentially established between nearby neighbors: smaller values of β emphasize short links and discourage long ones. The α parameter can be used to globally increase or decrease the probability of all links. Zegura et al. [14] also consider other models based on the Waxman model and examine the similarities and differences between alternative models.

We modify the Waxman model so that the average node degree D can be explicitly specified and precisely controlled. To do this, we first compute Eq. (7) for each pair of nodes after N nodes have been randomly placed in a given rectangle. We then generate a new bi-directional edge iteratively until exactly $DN/2$ edges have been added. At each iteration, if there is not already an edge between i and j, a new edge between i and j is added with probability $P(i, j)/\sum_{i',j'} P(i', j')$, where the sum is over all such pairs. Note that an edge is picked with a probability that is independent of the parameter α. Finally, we do post processing to ensure that the graph is connected. The graph distribution is described by the parameters D and β.

Fig. 8 compares the single-hop throughputs after color assignment and after tree design, using the Waxman model with $\beta = 0.8$, $D = 2.5$ and $N = 64$. The plot is obtained by averaging the results over 50 trials, where a new traffic matrix with parameter $(\mu = 0.8, v = 0.5, p = 1)$ and a new physical network topology are independently generated for each trial. The throughput after tree design is generally lower than that after coloring (before tree design) because of the additional physical-layer constraints imposed in the tree design. However, notice that the gap is remarkably small. We have also experimented with ring, regular meshed and NSF topologies and found similar behaviors as in Fig. 8. Finally, we note that the tree design result generally improves as D increases, due to the relaxation of physical topology constraints.

V. Flow Routing

Once the tree design has been completed, the next subproblem is to route flows onto the virtual topology. Recall that in TWIN, the virtual topology is fully connected and so the flow routing is trivial. In TWIN-WR, the simplest approach is to route flows according to shortest paths on the virtual topology. Alternatively, routes which minimize the congestion measure C can be obtained by solving the multicommodity flow problem described in Sec. II. Other heuristics that balance loads among the receivers may also be considered.

There are two basic approaches where client data units (IP packets, SONET frames, etc) may be assembled and disassembled. In the first approach, client data units are assembled at the ingress node and disassembled at the egress node of the virtual

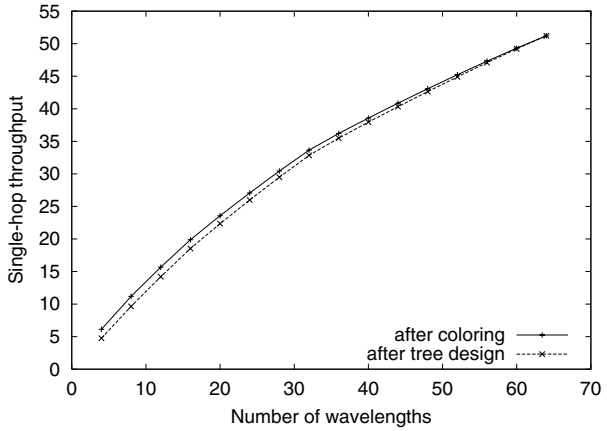


Fig. 8. Single-hop throughputs for a modified Waxman model with 64 nodes and average node degree of 2.5.

topology. In this case, each node has a 2x2 fixed-length burst switch that performs forwarding/dropping function. A timer is typically provided to ensure that the first data unit does not wait in an assembly buffer longer than a certain timeout. As a result, many bursts may be under filled with client data units when the end-to-end demand is very low. In the second approach, each node performs the assembly and disassembly functions and has a 2x2 variable-length packet switch to forward/drop client data units. Because of the aggregation at the link level, this approach may allow the assembler to function more efficiently (with fewer assembly timeouts). However, this approach requires assembly and disassembly to be performed at each hop. Under various load conditions and timeout constraints, the two approaches appear to present an interesting tradeoff study.

Recall that in TWIN, the traffic carrying capacity of the network is determined by the sum of all receiver capacities in the network. Moreover, the capacity of the network to carry traffic to any particular destination is completely independent of the traffic carried to other destinations. In other words, the transport resources (fibers, WDM transmission systems) are provisioned in such a way that transceiver capacity can be used with full efficiency and complete flexibility.

TWIN-WR provides an architecture for trading off transport resources (wavelengths) for transceiver capacity and traffic flexibility. Reducing the number of wavelengths used reduces the connectivity of the virtual topology, increasing the number of hops required to traverse the network, and thus increasing the fraction of receiver capacity that must be dedicated to forwarded data. Multi-hop routes introduce dependencies between receivers, meaning that the traffic that can be delivered to one destination depends on the capacity available at other destinations: the network is no longer as flexible in handling traffic patterns. On the other hand, the advantage of wavelength reuse is a reduction in the number of wavelengths required, or conversely, an increase in the number of nodes that can be supported with a given number of wavelengths.

A. Effective Node Capacity

In this section we discuss the effective node capacity that may be obtained in TWIN-WR for different numbers of wavelengths W and numbers of nodes N. By effective node capacity, we mean the inverse of the average number of hops. For example, if bursts travel over an average of three hops in the virtual topology, then on average, 2/3 of the traffic processed at a given node is transit traffic which must be forwarded to another node, while 1/3 of the traffic terminates at the destination. Thus the effective node capacity of this network is 1/3. In general, the effective node capacity may be computed for a particular traffic matrix, but in this section we will focus on a uniform traffic matrix. Note that the effective node capacity of TWIN is unity, regardless of the traffic matrix.

Although effective node capacity is a function of the TWIN-WR virtual topology and the traffic matrix, the physical topology has an indirect effect, constraining the possible virtual topologies that can be implemented. A richly connected physical topology supports a greater variety of virtual topologies than a sparsely connected one.

An upper bound on the effective node capacity may be obtained by considering perfect expansion. Recall that each node in a TWIN-WR virtual topology has an out-degree of W. Starting from an arbitrary node, one can reach at most W neighbors in one hop, no more than W^2 neighbors in exactly two hops, and in general no more than W^k distinct neighbors in exactly k hops. Thus the number of nodes that are more than k hops away from a given node is at least $(N-1-W-\ldots-W^k)^+$, where $(x)^+$ denotes the maximum of x and 0. If H is the shortest directed distance from a source node s to a randomly chosen destination $t \neq s$, then

$$E[H] = \sum_{k=0}^{\infty} P\{H > k\} \geq 1/(N-1) \sum_{k=0}^{\infty} \left(N - \sum_{j=0}^{k} W^j \right)^+ \tag{8}$$

The upper curve in Fig. 9 shows the resulting upper bound on effective node capacity $\hat{\rho} = 1/E[H]$ as a function of W, for a TWIN-WR network with $N = 128$. Note that the bound on effective node capacity initially increases very rapidly with W, achieving 50% capacity for $W \approx \sqrt{N}$.

For reasonably large W and N, a randomly chosen virtual topology can nearly achieve the above bound on effective node capacity. By a randomly chosen virtual topology, we mean the following: suppose that the W wavelengths are divided evenly between destinations, with approximately N/W destination assigned to each wavelength. For each source node and each wavelength, one of the N/W destination nodes assigned to that wavelength is chosen at random, and the corresponding edge is placed into the virtual topology.

Let T_i denote the number of distinct nodes that can be reached from a given source node s in exactly i hops. We have $T_1 = W$. Given T_i, one can compute T_{i+1} as follows. Each of T_i nodes selects one of N/W nodes at random. Such an experiment is equivalent to throwing $r = T_i$ balls at random

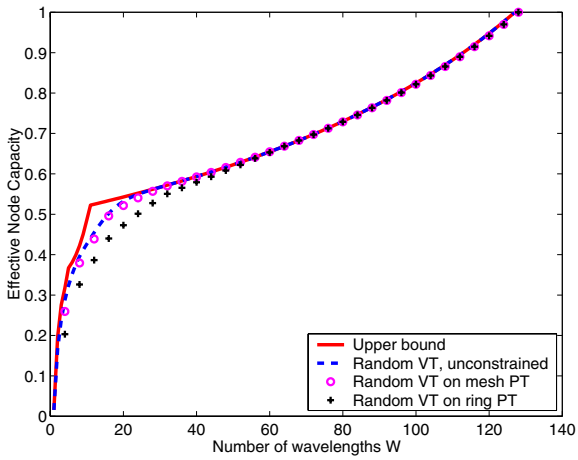


Fig. 9. Effective node capacity as a function of number of wavelengths used for a TWIN-WR network with 128 nodes. Shown are an upper bound, an approximation based on a random virtual topology without phyical constraints, and simulation results for mesh and ring physical topologies.

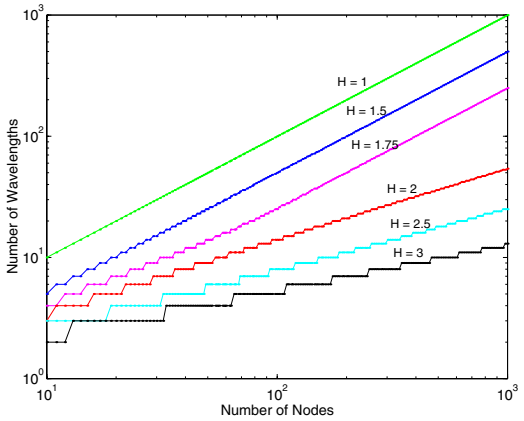


Fig. 10. Minimum number of wavelengths required to connect N nodes under TWIN-WR, using a random virtual topology, in order to keep the average number of hops below a given level.

into $M = N/W$ cells, in which case the expected number of distinct occupied cells is about $M\left(1 - e^{r/M}\right)$ [15]. Repeating this experiment for each of the W wavelengths, one obtains the approximation

$$T_{i+1} \approx N\left(1 - e^{-WT_i/N}\right).$$

Let A_i denote the number of nodes whose shortest paths from node s is greater than i hops. We have $A_0 = N - 1$, $A_1 = N - W - 1$. Note that T_{i+1} nodes can be reached from s in $i+1$ hops, while $N - T_{i+1}$ nodes cannot. Thus typically the number of nodes not reachable in less than $i+1$ steps is approximately

$$A_{i+1} = A_i\left(1 - T_{i+1}/N\right) = A_i e^{-WT_i/N}.$$

Similarly to the upperbound, the average number of hops may then be approximated as

$$E[H] = 1/N \sum_{k=0}^{\infty} A_k \tag{9}$$

The dashed curve in Fig. 9 shows the effective node capacity obtained from Eq. (9) as a function of W for a random virtual topology with 128 nodes. Note that the effective node capacity from Eq. (9) tracks the upper bound quite closely over the entire range, demonstrating that a random virtual topology of this type is nearly optimal.

In practice it will not be possible to route an arbitrary TWIN-WR virtual topology on a given physical topology. The sparser the physical topology, the less freedom there is in designing the virtual topology. To measure the effect of physical topology constraints, we performed experiments with "constrained random" virtual topologies. Given a physical topology, a "constrained random" virtual topology is generated as follows.

1) **Color Assignment:** A random color is assigned to each node.
2) **Tree Design:** The tree design algorithm of Sec. IV is used, except that source-destination pairs are chosen in a random order, rather than in order of decreasing traffic.

Fig. 9 shows the effective node capacity achieved in numerical experiments with constrained random virtual topologies based on mesh and ring physical topologies. The mesh topologies were randomly generated with our modified Waxman model with $\beta = 0.8$ and $D = 2.5$. On the scale of the graph, there is a barely perceptible decrease in effective capacity caused by the physical topology constraints. For mesh topologies with higher average degree, the observed capacity is even closer to a pure random virtual topology. In the case of a bidirectional ring, the effect of physical topology constraints is more pronounced, but even in this case the capacity rises very steeply for small W and is nearly optimal for say $W > 40$.

In summary, the results depicted in Fig. 9 demonstrate that random virtual topologies nearly minimize the average number of hops, and that this holds true even when the virtual topology must be implemented on a physical topology with modest connectivity.

The random topology results can be used to address the question of how many wavelengths are required to connect a given number of nodes with a given efficiency. Each curve in Fig. 10 corresponds to a given upperbound on the average number of hops H. For the given bound, the curve shows the minimum number of wavelengths required as a function of the number of nodes N, according to Eq. (9). On the log-log scale, the curves are nearly rectilinear in sections, meaning that the behavior can be approximately represented in the form $W = K(H)N^{\gamma(H)}$. Analysis of the upper bound on effective node capacity in the case of large W shows that the exponent is asymptotically $\gamma(H) = 1$ for $1 \le H < 2$ and $\gamma(H) = 2$ for $2 \le H < 3$. With an effective node capacity between $1/2$ and $1/3$, it is possible to connect ten-thousand nodes with a hundred of wavelengths in the TWIN-WR approach!

B. Robustness to traffic variability

One of the primary design features of the TWIN architectures is flexibility and robustness with respect to traffic distributions. In particular, a fully connected TWIN network can handle any traffic matrix such that the row and column sums are bounded by the transceiver capacity. In designing such a network, one only needs to estimate the row and columns sums and determine the number of transceivers required at each physical location. The flow of traffic on each wavelength layer may be carried out independently of the loads on other wavelengths. Robustness is useful because (1) it may be difficult to predict the true traffic matrix during the design stage, and (2) because traffic patterns may vary, so that there is no single true traffic matrix.

In TWIN-WR, some of the traffic traverses the network in multiple hops. As discussed in the previous section, this reduces the overall effective capacity of the network. The need to take multiple hops introduces dependencies between flows; the amount of traffic that can be carried between a given source destination pair now depends on how much is already being carried among other pairs. In such networks, performance can be improved by having a good estimate of expected traffic patterns, during the design stage. Nevertheless, simulation experiments demonstrate that the TWIN-WR architecture is still quite robust, in the sense that an effective network may be designed without detailed or accurate foreknowledge of traffic patterns.

Fig. 11 shows the results of an experiment to measure the robustness of the TWIN-WR design approach to traffic variability. The experiment was performed on a 64-node network, with the number of wavelengths ranging from 4 to 64. The physical topology is based on the modified Waxman with $\beta = 0.8$ and $D = 2.5$. To measure the congestion achieved *with* accurate traffic information, we generated a random nominal traffic matrix with parameters ($\mu = 0.3$, $v = 1$, $p = 1$), performed the design algorithms detailed in Secs. III-B and IV, routed flows according to shortest paths, and then computed the congestion (maximum receiver load). Each data point is obtained by averaging the results of several hundred independent trials. The results are plotted in the curve marked "good info" in the figure. To estimate the performance *without* accurate traffic information, we repeated the same design process, but in each trial we measured the congestion using a second *actual* traffic matrix, drawn independently from the same distribution as the nominal traffic matrix. As indicated by the curve labeled "bad info" in Fig. 11, the congestion increases by at most about 25% relative to the case of accurate knowledge of traffic information. As expected, when the number of wavelengths increases, the penalty of "bad info" disappears as the network gets closer to a fully connected TWIN architecture where knowledge of traffic information becomes irrelevant.

Fig. 11 also shows a third curve, labeled "no info". Here the nominal traffic matrix was drawn from a distribution with parameters ($\mu = 0.3$,$\nu = 0.01$,$p = 1$), i.e. the nominal traffic was uniform with a small proportion of random noise added. Applying the tree-design algorithm in this case produces a random virtual topology determined by the noise realization. Congestion was measured with respect to actual traffic matrices drawn from the same parameters as in the "good info" and "bad info" experiments. The performance with "no info" is essentially the same as the case with "bad info". The implication is that it is advantageous to use available nominal traffic information in the TWIN-WR design process, even if the accuracy of the information is suspect. To the extent that the information is accurate, congestion will decrease; this reduction is primarily due to an increase in the single-hop throughput.

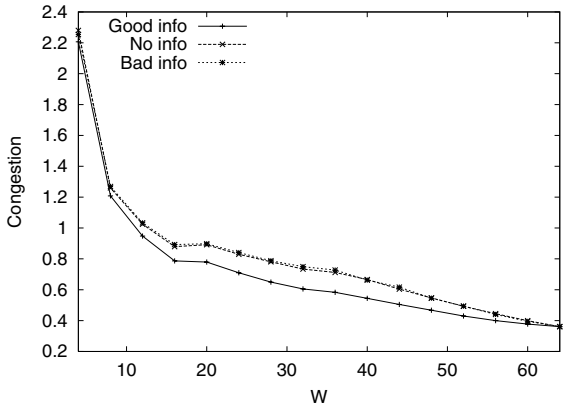


Fig. 11. The impact of traffic uncertainties on congestion when the virtual topology is designed: (i) using predicted traffic information that is accurate (good info), (ii) using predicted traffic information that is inaccurate (bad info) and (iii) without predicted traffic information (no info).

VI. Related Work

There are several related proposals that use optical network architectures capable of providing end-to-end flexible bandwidth (not just wavelength) services. The following reviews related areas.

A. Optical Packet Switching

Several optical packet switching architectures have been explored; e.g., [16]-[20]. Just like its electronic counterpart, optical packet switching needs to perform header lookup, optical fast switching and buffering. Optical packet buffering remains one of the biggest challenge to fully realize a system with comparable queueing capabilities found in electronic packet switching. Although optical delay lines are usually used in laboratories, they are unlikely to be practical for commercial products.

B. Optical Burst Switching

Optical burst switching is an alternative architecture that may or may not need optical buffering; e.g., [21]-[23]. Optical burst switching also needs some active processing modules at each node for contention resolution [24][25] and service differentiation and QoS support [26], and other functions.

C. TDM-WDM

The general idea of using tunable lasers for switching in optical networks has been documented (see [27][28] for surveys). Networks using passive core (no active switching) have only been considered with simple and regular physical topologies, including star networks based on star couplers or arrayed waveguide gratings (AWGs). EPON is standard example of a passive optical network with a tree topology [29], and HORNET is an experimental example with a ring topology [30]. Recent work has also proposed hybrid networks in which a star network based on an AWG and tunable transceivers is superimposed on a packet ring network [31]. Support for an arbitrary topology, as in TWIN, requires a passive network with wavelength-selective crossconnects capable of merging incoming signals of the same wavelength to the same outgoing fiber. Support of an arbitrary topology also makes scheduling fundamentally different from other regular topologies such as ring and tree. Finally, TDM can also be combined with WDM using active and fast optical switching such as [32]-[34].

VII. Conclusion

For networks that need end-to-end fractional wavelength services, TWIN promises to offer a cost-effective architecture due to its passive core with optical grooming capability. However, the number of nodes N supported by the network is limited by the number of available wavelengths W ($N \leq W$).

In this paper, we have proposed TWIN with wavelength reuse (TWIN-WR) where $N > W$. We formulated the design problem and presented a computationally efficient algorithm with three stages: color merging, tree design and flow routing. We have investigated the performance of the proposed algorithm over different topologies and traffic matrices. The results indicate that single-hop throughput is close to optimal. Our scaling study also shows that a very large network with tens of thousands of nodes can be built with only a hundred of wavelengths, which is well within the capability of today's technology. Finally, we have demonstrated the robustness of TWIN-WR virtual topology in the presence of variability of the traffic patterns.

References

[1] I. Widjaja, I. Saniee, R. Giles and D. Mitra, "Light-core and intelligent edge for a flexible, thin-layered, and cost-effective optical transport network," *IEEE Comm. Magazine*, pp. S30–S36, May 2003.

[2] I. Widjaja and I. Saniee, "Simplified Layering and Flexible Bandwidth with TWIN," in *Proc. SIGCOMM FDNA Workshop*, Portland, Aug. 2004.

[3] S. Sudipta, V. Kumar and D. Saha, "Switched optical backbone for cost-effective scalable core IP networks," *IEEE Comm. Magazine*, pp. 60-70, Jun. 2003.

[4] K. Ross, N. Bambos, K. Kumaran, I. Saniee, I. Widjaja, "Scheduling bursts in time-domain wavelength interleaved networks," *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 9, pp. 1441-1451, Nov. 2003.

[5] B. Zhu et al., "6.4-Tb/s (160x42.7 Gb/s) transmission with 0.8 bits/s/Hz spectral efficiency over 32x100 km of fiber using CSRZ-DPSK format," in *Proc. OFC 2003*, postdeadline paper PD19, Georgia, 2003.

[6] P.P. Mitra and J.B. Stark, "Nonlinear limits to the information capacity of optical fiber communications," *Nature*, vol. 411, pp. 1027-1030, Jun. 2001.

[7] A. Odlyzko, "Internet traffic growth: sources and implications," *Proceedings of SPIE 2003*, vol. 5247, 2003.

[8] A. Brzezinski et al., "Flow control and congestion management for distributed scheduling of burst transmissions in time-domain wavelength interleaved networks," in *Proc. OFC 2005*, Los Angeles, 2005.

[9] K.C. Lee and V. Li, "A wavelength-convertible optical network," *Journal of Lightwave Technology*, vol. 11, no. 5/6, pp. 962-970, May/Jun 1993.

[10] I. Chlamtac, A. Faragó and T. Zhang, "Lightpath (wavelength) routing in large WDM networks," *IEEE Journal on Selected Areas in Communications*, vol. 14, no. 5, pp. 909-913, Jun. 1996.

[11] A. Jukan and H.R. van As, "Service-specific resource allocation in WDM networks with quality constraints," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 10, pp. 2051-2061, Oct. 2000.

[12] K. Kumaran, C.J. Nuzman and I. Widjaja, "Wavelength assignment for partially transparent networks with reach constraints," *Journal of Optical Networking*, vol. 2, no, 8, pp. 285-302, Aug. 2003.

[13] B.M. Waxman, "Routing of multipoint connections," *IEEE Journal on Selected Areas in Communications*, vol. 6, no. 9, pp. 1617-1622, 1988.

[14] E.W. Zegura, K.L. Calvert and S. Bhattacharjee, "How to model an internetwork," in *Proc. INFOCOM*, Apr. 1996.

[15] W. Feller, "An Introduction to Probability Theory and Its Applications", vol. 1, Wiley, New York, 1968.

[16] C. Guillemot et al., "Transparent Optical Packet Switching: The European ACTS KEOPS Project Approach," *Journal of Lightwave Technology*, vol. 16, no. 12, pp. 2117-2134, Dec. 1998.

[17] A. Carena et al., "Opera: An optical packet experimental routing architecture with label swapping capability," *Journal of Lightwave Technology*, vol. 16, no. 12, pp. 2135-2145, Dec. 1998.

[18] D. K. Hunter et al., "WASPNET: A Wavelength Switched Packet Network," *IEEE Communication Magazine*, pp.120-129, Mar. 1999.

[19] S. Yao, S. Dixit and B. Mukherjee, "Advances in photonic packet switching: an overview," *IEEE Comm. Magazine*, pp. 84-94, Feb. 2000.

[20] T. S. El-Bawab and J. Shin, "Optical packet switching in core networks: between vision and reality," *IEEE Comm. Magazine*, pp. 60-65, Sep. 2002.

[21] J. Turner, "Terabit burst switching," *J. High Speed Networks*, vol. 8, no. 1, pp. 3-16, Jan. 1999.

[22] C. Qiao and M. Yoo, "Optical burst switching (OBS) - a new paradigm for an optical Internet," *J. High Speed Networks*, vol. 8, no. 1, pp. 69-84, Jan. 1999.

[23] L. Xu, H.G. Perros and G. Rouskas, "Techniques for optical packet switching and optical burst switching," *IEEE Comm. Magazine*, pp. 136-142, Jan.. 2001.

[24] V.M. Vokkarane and J.P. Jue, "Prioritized routing and burst segmentation for QoS in optical burst-switched networks," in *Proc. OFC 2002*, 2002.

[25] A. Zalesky et al., "Modeling and performance evaluation of optical burst switched networks with deflection routing and wavelength reservation," in *Proc. INFOCOM*, Apr. 2004.

[26] M. Yoo, C. Qiao and S. Dixit, "QoS performance of optical burst switching in IP-over-WDM networks," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 10, pp. 2062-2071, Oct. 2000.

[27] B. Mukherjee, "WDM-Based Local Lightwave Networks Part I: Single-Hop Systems," *IEEE Network*, pp. 12–26, May 1992.

[28] S. Alexander et al., "A Precompetitive Consortium on Wide-Band All-Optical Networks," *J. Lightwave Technologies*, vol. 2, no. 5/6, May/June 1993.

[29] IEEE, "Draft amendment to carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications," IEEE Draft P802.3ah/D3.1, Feb. 2004.

[30] K.V. Shrikhande et al., "HORNET: a packet-over-WDM multiple access metropolitan area ring network," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 10, pp. 2004-2016, Oct. 2000.

[31] M. Maier and M. Reisslein, "AWG-based metro WDM networking," *IEEE Optical Communications*, pp. S19-S26, Nov. 2004.

[32] N. Huang, G Liaw and C. Wang, "A novel all-optical transport network with time-shared wavelength channels," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 10, pp. 1863-1875, Oct. 2000.

[33] S. Subramaniam, E.J. Harder and H. Choi, "Scheduling multirate sessions in time division multiplexed wavelength-routing networks," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 10, pp. 2105-2110, Oct. 2000.

[34] R. Srinivasan and A. Somani, "A generalized framework for analyzing time-space switched optical networks," *IEEE Journal on Selected Areas in Communications*, vol. 20, no. 1, pp. 202-215, Jan. 2002.