Improved Access to Optical Bandwidth in Trees *

Vijay Kumar †
Eric J. Schwabe ‡

Department of ECE
Northwestern University
2145 Sheridan Road
Evanston, IL 60208

Abstract

We present improved bounds for efficient bandwidth allocation in a WDM optical network whose topology is that of a directed tree of fiber-optic links. The problem of bandwidth allocation is modeled as a coloring problem, where each path in a set of communication requests must be assigned a color (representing a wavelength) in such a way that no two paths using the same link in the same direction are assigned the same color. Letting L be the largest number of paths using any directed link, we show that for an arbitrary set of paths, $7L/4$ colors are sufficient to route all paths. This improves an upper bound of $15L/8$ due to Mihail, Kaklamanis and Rao [8]. In addition, we show that a family of problem instances given by Mihail, Kaklamanis and Rao [8] to establish a worst-case lower bound of $3L/2$ for the problem can in most cases be colored with only $5L/4$ colors (technically, $\lceil 5L/4 \rceil$). Finally, we show that in all cases $5L/4$ colors are in fact necessary for this family of instances, yielding a general lower bound of $5L/4$.

1 Introduction.

1.1 Background.

In this paper, we consider the problem of resource allocation in *optical* networks. In such networks, communication occurs through the transmission of a laser beam through an optical fiber link, using a chosen wavelength that remains the same over the entire path traversed by the beam. Multiple messages can be transmitted across the same channel simultaneously as long as they use distinct wavelengths — this technique is known as *wavelength division multiplexing* (WDM).

In practice, some minimum separation between the wavelengths used will have to be enforced to avoid interference, but we will assume that the wavelengths used simply come from some discrete set of permissible wavelengths that are known to be mutually non-interfering. Bandwidth being a costly resource, it is important to devise allocation schemes that can support a large amount of communication using a limited bandwidth. The problem of supporting a large set of communication requests at the same time while using the smallest possible amount of bandwidth is equivalent to the following coloring problem:

We are given a set S of communication requests (i.e., source-destination pairs), and a network N made up of fiber-optic links. We must choose a path in the network from each source to its corresponding destination, and assign a color to each path in such a way that no paths that traverse a common link are assigned the same color. We call this a *valid* coloring. This should be accomplished using as few colors (i.e., wavelengths) as possible.

For some simple networks, such as trees, the WDM routing problem is simpler, as there is always a unique path for each request from its source to its destination. In this case, the problem reduces to simply choosing a color for each path.

1.2 Previous Work.

Raghavan and Upfal [10] considered the problem of finding provably good routing algorithms for optical networks. They considered both general unstructured networks and specific networks such as trees, rings, and meshes. Earlier results by Aggarwal, Bar-Noy, Coppersmith, Ramaswami, Schieber and Sudan [1] focussed on

*This research was supported in part by the National Science Foundation under grant CCR–9309111.

†Author's email address: vijay@ece.nwu.edu.

‡Author's email address: schwabe@ece.nwu.edu.

the structure and permutation routing ability of optical networks.

Raghavan and Upfal [10] gave algorithms to route arbitrary sets of requests on undirected trees, rings, and trees of rings. They also gave randomized algorithms for routing on meshes and arbitrary bounded-degree networks, but with much looser probabilistic bounds. Erlebach and Jansen [2, 3] showed the problem of routing sets of requests with the minimum number of wavelengths to be NP-complete on trees, rings, and meshes and gave a better approximation algorithm for the problem on trees.

Mihail, Kaklamanis and Rao [8] were the first to consider *directed* networks, which more accurately reflect the actual asymmetric properties of optical fiber networks. The NP-completeness results of Erlebach and Jansen [2, 3] also apply to the directed case.

The algorithm of Mihail, Kaklamanis and Rao [8] for tree networks employs an inductive approach, moving through the tree vertex by vertex and modeling the inductive step as the coloring of a regular bipartite graph. Using that framework, it is relatively easy to route any set of requests using $2L$ wavelengths, where L is used to denote the largest *load* among all links, i.e., the largest number of routing paths that share a link *in the same direction*. Mihail, Kaklamanis and Rao [8] used a detailed coloring scheme to obtain an algorithm that improves this upper bound to $15L/8$. Their paper also considered routing on rings. (Kleinberg and Tardös [7] and Rabani [9] have considered this problem for directed meshes.)

Mihail, Kaklamanis and Rao [8] also proposed a family of problem instances with at most L paths using each directed link for which $3L/2$ wavelengths were claimed to be both necessary and sufficient. In particular, they demonstrated a request set with $L = 2$ that requires three wavelengths to route.

1.3 Our Results.

We improve the results of Mihail, Kaklamanis and Rao [8] on two fronts:

1. We give an algorithm for routing an arbitrary set of requests with at most L paths using each directed link that uses at most $7L/4$ wavelengths (this result was also proved independently by Kaklamanis and Persiano [6]);
2. The instance for $L = 2$ that they demonstrated to require three wavelengths to route shows that no algorithm can route every request set using fewer than $3L/2$ wavelengths. However, we show that the problem instance they gave does not establish this bound for values of L larger than two. In fact, for all problem instances in the family that they proposed, $5L/4$ (technically, $\lceil 5L/4 \rceil$) wavelengths are both necessary and sufficient to route all requests. This agrees with their bound when $L = 2$, but yields a valid lower bound for all values of L.

The algorithm that establishes the upper bound uses an inductive approach similar to that used by Mihail, Kaklamanis and Rao [8], but uses a modified inductive step with a tighter analysis to achieve the improved bound. The lower bound proof uses a relatively simple pigeonhole argument, along with techniques originally developed for channel assignment in cellular phone systems due to Jordan and Schwabe [5]. Our results bracket the actual number of wavelengths that are necessary and sufficient in general to be between $5L/4$ and $7L/4$, as opposed to the bracketing between $3L/2$ and $15L/8$ given by Mihail, Kaklamanis and Rao [8]. (For convenience, we omit the floors and/or ceilings that are technically present in these expressions.)

Recently, Jansen [4] gave an algorithm to route arbitrary request sets with maximum load L using $5L/3$ wavelengths, for the special case of binary trees. He has also demonstrated a message set with $L = 3$ that requires five wavelengths to route, suggesting that an improvement of the general lower bound to $5L/3$ may be possible.

2 An Algorithm for Path Coloring.

In this section, we present an algorithm that allocates wavelengths to a given set of communication requests. Let the network topology be that of a tree T in which each edge corresponds to a pair of oppositely directed fiber links. Suppose we are given a set of communication requests. Each request can be looked upon as a path in the tree. Suppose no more than L paths pass over any directed link. Our objective is to assign colors to the paths so that no two paths share the same color, while using as few colors as possible.

We establish the following result:

THEOREM 2.1. *Given a tree T and a set of paths in T such that no more than L paths pass over any directed link of T, it is possible to find in polynomial time a valid coloring of the given set of paths that uses no more than $7L/4$ colors.*

To establish this, we present such an algorithm. The algorithm that traverses the tree and colors paths as it encounters them. The algorithm visits the the vertices of the tree in DFS order. A step of this algorithm consists of visiting a vertex v and coloring all paths that touch (i.e., contain) v. Some of them will have already been colored — in fact, any path that touches a vertex with a smaller DFS number than v will already have been colored. Note that any path encountered for the first time while visiting v must be contained entirely in the subtree rooted at v.

2.1 General Framework.

Throughout the execution of the algorithm, the following three invariants will be maintained:

Invariant 1: No two paths sharing a link will be colored with the same color.

Invariant 2: No more than $7L/4$ colors will be used.

Invariant 3: The paths passing through a pair of corresponding oppositely directed links will be colored using no more than $3L/2$ colors.

We use induction on the number of vertices visited to show that all paths encountered can be colored while maintaining our two invariants. That is, given a valid partial coloring in which all the paths that touch vertices with DFS numbers smaller than v are colored in such a way that the three invariants above are satisfied, we will show how to extend the partial coloring to include the vertex v and the paths touching it.

The invariants are vacuously satisfied at the beginning of the algorithm. Therefore a correct polynomial-time coloring procedure that maintains the invariants will constitute a proof of theorem 2.1. We reduce the coloring step to the problem of edge-coloring a bipartite graph (as in Mihail et al. [8]), and present a solution to the latter problem. In the following, we describe the reduction in detail.

2.2 Modeling With Bipartite Graphs.

Assume that all paths that touch vertices with smaller DFS numbers than v have already been colored without violating the invariants. The inductive step involves extending this partial coloring to include all paths that touch the vertex v. Following [8], we reduce this to the problem of edge-coloring a bipartite graph. Let v be the vertex being visited. Let x_0 be the parent of v. In case v is the root, add a 'dummy' parent node linked only to v. Let $x_1, x_2, \cdots, x_k$ be the children of v. For each x_i, there are four nodes in the bipartite graph: y_i and v_{y_i} on the left side, and Y_i and V_{y_i} on the right side. A path p with one end at v that touches x_i is represented in the bipartite graph by an edge between v_{y_i} and Y_i if p is directed away from v, and by an edge between y_i and V_{y_i} otherwise. A path that touches x_i and x_j in that order is represented by an edge between y_i and Y_j in the bipartite graph.

More formally, the bipartite graph G_v is constructed as follows: The left and right vertex sets are $\bigcup_i \{y_i, v_{y_i}\}$ and $\bigcup_i \{Y_i, V_{y_i}\}$. Add edges to the bipartite graph as follows:

- For each path from some x_i to some x_j place an edge from y_i to Y_j in the bipartite graph.
- For each path from some x_i that terminates at v place an edge from y_i to V_{y_i}.
- For each path starting at v and directed into some x_i place an edge from v_{y_i} to Y_i.

For each path that has already been colored, color the corresponding edge in G_v with the same color. Figure 1 illustrates the construction of the bipartite graph.

Note that y_i represents the link from x_i to v in the sense that every path passing over this link is represented by an edge incident on y_i. Similarly, Y_i represents the link in the opposite direction.

For simplicity, we will make the following assumptions:

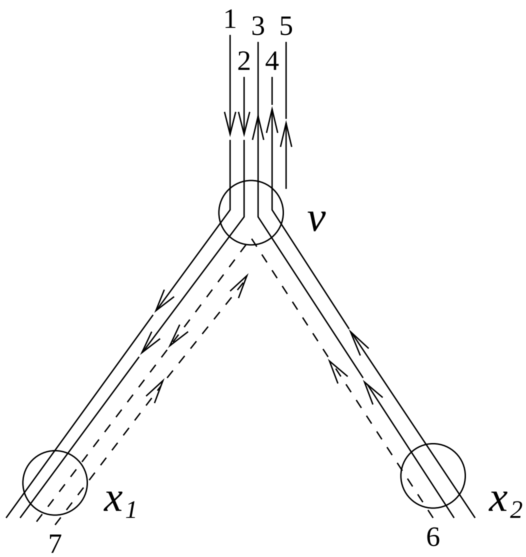
Assumption 1: There is a load of exactly L on each link.

Assumption 2: The two (oppositely directed) links between v and its parent together use exactly $3L/2$ colors.

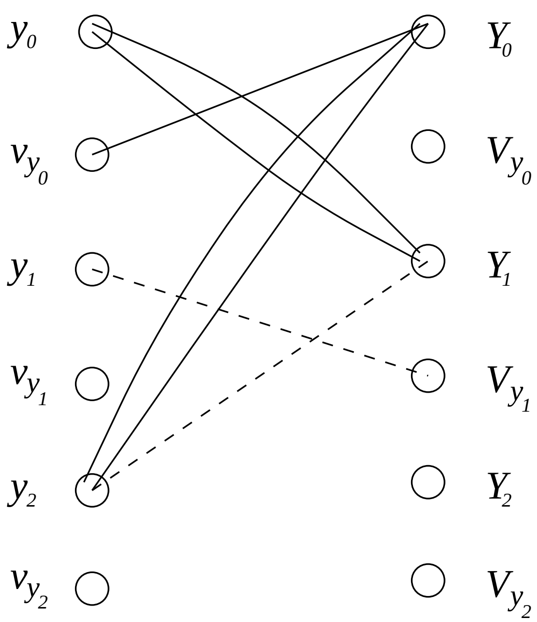
It is easy to see that these assumptions can be made without loss of generality, and that if they do not hold extra paths and/or colors can be added to a given instance to make them true.

The following lemma formalizes the correspondence between the inductive coloring step and the bipartite graph coloring problem.

LEMMA 2.1. *Any two paths that include v and can not be assigned the same color will be represented in the bipartite graph above by two edges that share a vertex; thus, a valid edge-coloring of the bipartite graph represents a valid coloring of the paths in the original graph.*



(a) Coloring step at vertex v: uncolored edges are shown by dotted lines.



(b) The corresponding bipartite graph.

Figure 1: Modeling the inductive step as a bipartite graph coloring problem.

Proof. The valid coloring of the paths is obtained from a coloring of this bipartite graph by giving each path the same color as the corresponding edge in the bipartite graph. ∎

As we noted above, a pair of oppositely directed links are represented by a pair of opposite vertices in the bipartite graph. Invariant 3 requires that any such pair of links should be colored with no more than $3L/2$ colors. That implies that the corresponding pair of vertices in the bipartite graph should together see no more than a total of $3L/2$ colors. We must color G_v without violating this requirement.

Next, we show how to color the bipartite graph G_v.

3 Coloring a Bipartite Graph.

Note that among the paths that touch v, the paths that contain x_0, the parent of v, are already colored, and no other paths are colored. Consequently, all the edges in G_v that touch y_0 or Y_0 are colored, and no other edges are colored. Also note that no edges run between opposite vertices, i.e., y_i and Y_i, or v_{y_i} and V_{y_i}.

Also, while no vertex has more than L incident edges, and y_0 and Y_0 have exactly L incident edges each, other vertices may have fewer than L incident edges. We will add extra edges to the graph to make it L-regular.

As y_0 and Y_0 have L colored edges each, and together they use $3L/2$ colors, they must share exactly $L/2$ colors. We will refer to the shared colors as *double* colors, and to the rest as *single* colors. The corresponding edges will be called double-color edges and single-color edges respectively.

Below, we describe how to color all the edges of G_v using $7L/4$ colors. The algorithm involves two distinct phases. In the first phase, we extract from G_v subgraphs of a particular kind, called *gadgets*, and color them in a way that maximizes the reuse of colors already present in G_v. When no more such subgraphs can be extracted, phase 2 is invoked to color the remaining graph, by splitting it into two parts: a subgraph which can be colored without using any new colors, and the rest.

As we saw in the previous section, a pair of oppositely directed links in the network is represented by a pair of opposite vertices in G_v, and the edges incident on that pair of vertices must be colored with no more than $3L/2$ colors in order to satisfy invariant 3. We will satisfy this requirement by ensuring that every pair of opposite vertices share at least $L/2$ colors. This will be referred to as the *color-sharing requirement.*

Our algorithm will involve breaking G_v into regular bipartite subgraphs which have the same vertex set but a smaller degree. We will ensure that each individual subgraph satisfies the color-sharing requirement. That is, each pair of opposite vertices will share one color in the case of a 2-regular subgraph, and two colors in the case of a 3-regular or 4-regular subgraph. Clearly, this is sufficient to meet the overall color-sharing requirement.

3.1 Gadgets.

A *gadget* is a subgraph of G_v which contains all the vertices of G_v and in which y_0 and Y_0 each have degree 3 while all other vertices have degree 2.

A gadget can be looked upon as a union of three paths in G_v. There are three edges going out of y_0, and through these, we can trace paths that lead to either y_0 or Y_0. Similarly for Y_0. There are three such paths in all, and either all three extend between y_0 and Y_0 or one extends between y_0 and Y_0 while the other two loop back to their starting points. Let us refer to these paths as *strands*. Each vertex is contained in exactly one strand.

Gadgets allow us to economize on the number of colors that we use, because some of the edges of the gadgets can be colored using colors already present in the gadget. Phase 1 involves repeated extraction and coloring of gadgets.

3.2 Phase 1.

In phase 1, our objective is to reduce the proportion of double-color edges in the graph by extracting suitable gadgets while economizing on the new colors used to color the gadgets.

In phase 1, we will repeatedly extract 3-regular subgraphs H, each consisting of a gadget H_1 and a matching H_2 that involves every vertex other than y_0 and Y_0. We ensure that each subgraph H extracted in this phase is such that of the six edges of H that are incident on y_0 or Y_0, two are colored with the same color a, while another two are colored with single colors b and c. The remaining two are colored with (possibly distinct) double colors.

Note that there are three *reusable* colors involved in H_1: a, b and c. That is to say, these colors can be used to color some edges of the gadget because we know that no edge of $G_v - H$ is colored with either of these colors. Using these three colors, we can color H_1 in such a way that at least one color is shared by every pair of opposite vertices of G_v. The details of the extraction and the coloring will appear in the full paper.

H_2 can now be colored using a *new* color — that is, a color that is not among the $3L/2$ colors initially present on the edges of G_v. Clearly, every pair of opposite vertices shares this new color.

We have colored H using no more than one new color such that every pair of opposite vertices shares at least two colors. Phase 1 of the algorithm consists of extracting and coloring such subgraphs till no more subgraphs that have the desired color property can be extracted.

Let us analyze the effect of phase 1.

LEMMA 3.1. *At least $L/6$ subgraphs are extracted in phase 1.*

Proof. Phase 1 will when end no more double-color edge-pairs are left. We had $L/2$ such pairs to begin with, and each 3-regular subgraph extracted contains one pair while it may *separate* another two pairs by including one edge from each. ∎

Our scheme of coloring these subgraphs implies that

OBSERVATION 3.1. *The number of new colors used in phase 1 is equal to the number of subgraphs extracted.*

So if the degree of each vertex is reduced by l as result of phase 1, then $l/3$ new colors have been used in phase 1.

3.3 Phase 2.

As phase 1 progresses, the ratio of double-color edges (separated or not) to single-color edges in the remaining graph diminishes. This is helpful, because single colors are reusable. Let $R(G)$ denote the ratio of double-color edges to single-color edges in a subgraph G of G_v. $R(G_v)$ is 1:1, and the ratio goes down with every extraction of a gadget: $R(H)$ is 2:1 for each gadget H extracted. This, together with lemma 3.1, has the following implication for G_v^r, the graph remaining at the end of phase 1:

LEMMA 3.2. *$R(G_v^r)$ is 1:2 or lower.* ∎

Phase 2 colors the remaining $(L-l)$-regular graph G_v^r. The coloring scheme depends on the value of $R(G_v^r)$.

LEMMA 3.3. *If $R(G_v^r)$ is 1:3 or lower, we can color G_v^r without using any new colors.*

Proof. Consider the case where the ratio is exactly 1:3. As single colors are reusable and as we have a good ratio of single colors double colors, we would like to extract subgraphs which contain more single than double colors and try to color them with the single colors they contain. To do this, we transform G_v^r into G_v' as follows: break y_0 into 3 vertices, y_{01}, y_{02} and y_{03}. Let $(L-l)/2$ single-color edges be incident on y_{03}, $(L-l)/4$ single-color edges on y_{02} and $(L-l)/4$ double-color edges on y_{01}. Split Y_0 into Y_{01}, Y_{02} and Y_{03} in exactly the same way. Split each of the other vertices into two, each part getting half the edges. Now add $(L-l)/4$ dummy edges between y_{01} and Y_{02}, and an equal number between y_{02} and Y_{01}. This is an $(L-l)/2$-regular bipartite graph. Split it into $(L-l)/2$ matchings, and from each matching M, throw away the dummy edges and merge the split vertices back.

M can contain up to two dummy edges. If M contained one dummy edge, the resulting graph is a cycle cover containing three single-color edges. Two of these three colors can be used to color this cycle cover.

If M contained two dummy edges, then the resulting subgraph is one edge short of a cycle cover: it visits each vertex twice except for y_0 and Y_0 which are visited once each. It can be 'unzipped' into a perfect matching and another matching. The former contains two colored edges which are colored with single colors. One of them can be used to color it. The second matching will be colored along with a gadget below. There are as many gadgets as there are such matchings, and a one-to-one assignment can be made.

If M contained no dummy edges, then the resulting subgraph is a gadget. Its three strands together contain four single and two double colors. Picking two of the single colors, we can color the gadget. The other two colors can be used to color the a matching from the previous case that may be assigned to this gadget.

In each case, the coloring can be done without violating the color-sharing requirement. The details will appear in the full paper.

By following this procedure for each matching, we can color G_v^r without using any new colors.

If the ratio is lower than 1:3, we can split G_v^r into two bipartite graphs: one in which the ratio is exactly 1:3, and the other in which there are no edges with double colors. Each can then be colored without using new colors.

■

Next, let us consider the case when $R(G_v^r)$ is exactly 1:2.

LEMMA 3.4. *If G_v^r is a k-regular bipartite graph such that $R(G_v^r)$ is 1:2, it can be colored using $k/6$ new colors.*

Proof. As before, we wish to break G_v^r into small-degree subgraphs which have a high ratio of single colors to double colors. To do this, transform G_v^r into G_v' as follows: break y_0 into 3 vertices, y_{01}, y_{02} and y_{03}. Let $k/3$ double-color edges be incident on y_{01}, and $k/3$ single-color edges on each of y_{02} and y_{03}. Split Y_0 into Y_{01}, Y_{02} and Y_{03} in exactly the same way. Split each of the other vertices into two, each part getting half the edges. Now add $k/6$ dummy edges between y_{01} and Y_{02}, and an equal number between y_{02} and Y_{01} as well as between y_{03} and Y_{03}. What we get is a $k/2$-regular bipartite graph. Split it into $k/2$ matchings, and from each matching M, throw away the dummy edges and merge the split vertices back.

What is the structure of the resulting subgraph M'? It depends on the number of dummies in M. Let us group the matchings into four sets depending on the number of dummies they contain: set S_i contains all the matchings with i dummies each. Let m_i be the cardinality of set S_i.

As there are $k/2$ dummies and $k/2$ matchings, there is one dummy per matching on the average. The matchings with no dummies have to be counterbalanced by matchings with more than one dummy. It is easily seen that:

$$m_0 = m_2 + 2m_3$$

In other words, we can *assign* two matchings from set S_0 to each matching from S_3 and one to each matching from S_2. Let us make these assignments.

First, let us consider the matchings with 3 dummies. For such a matching M, M' consists of two perfect matchings on $G_v^r - \{y_0, Y_0\}$. These matchings contain no colored edges. Along with each of the two S_0 matching assigned to M, we will color one of these matchings. This task postponed, we proceed to deal with the members of S_2.

If M contains two dummies, M' is the union of a perfect matching of G_v^r with a perfect matching of $G_v^r - \{y_0, Y_0\}$. The former contains at least one single color, and we can use that color to color all its uncolored edges. The task of coloring the latter is again left to the member of S_0 assigned to M.

Next, let us consider members of S_0 themselves. For each such M, M' has the structure of a gadget, containing two single and one separated color on each of y_0 and Y_0. Two of the single colors can be used to color the gadget, while the other two can be used to color the extra matching assigned to M in previous steps, as was done in the proof of lemma 3.3.

Lastly, let us consider members of S_1. For each such M, M' is a cycle cover of G_v^r. If the dummy edge in M is not of type 3, the cycle cover contains 3 single colors, and it is possible to use two of them to color M' without violating the 3-color requirement. However, if the dummy is of type 3, M' contains two single colors and two separated colors. If both the single colors come from one of the two matching that constitute the cycle cover, then there is no way of coloring M' without using a new color. In this case, we use one of the two single colors and a new color to color M'.

In each case, the coloring can be performed without violating the color-sharing requirement. Details will be presented in the full paper.

How many matchings in S_1 could contain a type 3 dummy? Clearly, their number is bound by the number of type 3 dummies, which is $k/6$.

We therefore have a way of coloring G_v^r using no more than $k/6$ new colors without violating the 3-color requirement. ■

We have already noted that $R(G_v^r)$ can not be higher than 1:2. What about the case when the ratio is between 1:2 and 1:3?

LEMMA 3.5. *If $R(G_v^r)$ lies between 1:2 and 1:3, we can partition G_v^r into two regular bipartite graphs G_1 and G_2 with the same vertex set as G_v^r, such that $R(G_1)$ is 1:2 and $R(G_2)$ is 1:3.*

Proof. Such a partitioning may be accomplished by splitting the graph into matchings and partitioning the set of matchings appropriately. (The details are straightforward, and will appear in the full paper.) ■

We have already seen how to color G_1 and G_2, in lemmas 3.3 and 3.4. Together with lemma 3.5, they take care of coloring G_v^r in all cases.

3.4 Overall Performance of the Algorithm.

Let us see what the overall color requirement for coloring G_v is. G_v contained $3L/2$ colors to begin with, and some new colors have been used by our algorithm.

THEOREM 3.1. *G_v can be colored using no more than $7L/4$ colors while preserving the invariants.*

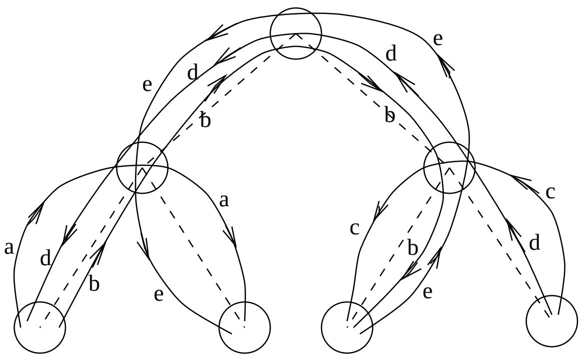


Figure 2: A set of paths where $L = 2$.

Proof. As we have obtained a valid coloring of G_v, it follows that invariant 1 is preserved (lemma 2.1). Invariant 3 is also preserved, since the coloring of G_v satisfies the color-sharing requirement. Invariant 2 requires that no more than $7L/4$ colors be used.

In the first phase, we use a new color every time we extract a gadget (observation 3.1). That translates to one new color for every four double-colored edges removed.

In phase two, we use at most $k/6$ new colors, where $k/3$ is the number of double-colored edges incident on each of y_0 and Y_0. So again we use no more than one new color for every four double-colored edges (lemmas 3.3, 3.4, 3.5). Let D be the number of double colors. Initially, G_v contained $2L - D$ colors. So at the end, it has been colored with $2L - D/2$ colors. Invariant 3 implies that D is at least $L/2$, and the result follows. ■

The coloring of G_v thus obtained can be used to extend the partial coloring of paths of T to include all paths that touch vertex v, in such a way that the three invariants are maintained.

Theorem 3.1 is a statement of the validity of our inductive step, and theorem 2.1 follows.

4 A Lower Bound.

Mihail, Kaklamanis and Rao [8] used a family of problem instances including the one in figure 2 to justify a lower bound of $3L/2$ colors for the problem or coloring a set of paths with maximum load L. In the instance pictured in figure 2, the value of L, the maximum load over a link, is 2. a, b, c, d and e are the five paths to be colored. It can be easily verified that no three paths out of the five can be colored with the same color. Three colors are, therefore, necessary. Three colors are sufficient too: a and c can share a color, as can d and b. It follows that for $L = 2$, no algorithm can route arbitrary request sets with fewer than $3L/2$ — that is, three — wavelengths.

However, this proposed lower bound of $3L/2$ does not hold for most problem instances in this family — in fact, it is only valid for $L = 2$. Consider the case where each of a, b, c, d and e represents not a single path but a set of $L/2$ paths; this yields a maximum load of L. Mihail, Kaklamanis and Rao [8] generalized from the preceding example and suggested that $3L/2$ colors would be required. However, as the following valid coloring shows, $5L/4$ colors will actually suffice for this example. As long as $L \geq 3$, this is fewer than $3L/2$ colors.

Assign $L/2$ colors to each of the five sets of paths, one color for each path in the set, according the following assignment of colors to sets:

$$
\begin{array}{ll}
a: & \{1, 2, \ldots, \frac{L}{2}\} \\
b: & \{\frac{L}{2}+1, \ldots, L\} \\
c: & \{L+1, \ldots, \frac{5L}{4}\} \cup \{1, \ldots, \frac{L}{4}\} \\
d: & \{\frac{L}{4}+1, \ldots, \frac{3L}{4}\} \\
e: & \{\frac{3L}{4}+1, \ldots, \frac{5L}{4}\}
\end{array}
$$

It is straightforward to verify that no two colors that use the same directed link are assigned the same color, so $5L/4$ colors are sufficient.

It turns out that $5L/4$ colors are also necessary in this instance. To see that, note that no color may be use to color more than two paths. This is so because a color may be used only once in each set, and it can not be used in more than two sets (since no three out of the five sets a, b, c, d and e may share a color). As there are a total of $5L/2$ paths to be colored and no color may be used more than twice, it follows that $5L/4$ colors are necessary. An alternate proof of the requirement of $5L/4$ colors comes from a more general technique developed by Jordan and Schwabe for lower bounds on channel assignment in cellular systems [5].

If we consider all possible values of L, the above example represents a class of instances for which $5L/4$ (to be precise, $\lceil 5L/4 \rceil$) colors are necessary and, incidentally, sufficient. This yields the following lower bound:

THEOREM 4.1. *For every value of L, there is a tree T and a set S of paths over T such that no directed link of T is used by more than L paths, and $5L/4$ colors are necessary to color S.* ■

References

[1] A. Aggarwal, A. Bar-Noy, D. Coppersmith, R. Ramaswami, B. Schieber, and M. Sudan,*Efficient Routing and Scheduling Algorithms for Optical Networks*, Proc. of the 5th ACM-SIAM Symp. on Discrete Algorithms, pp. 412-423, 1993.

[2] T. Erlebach and K. Jansen, *Call-Scheduling in Trees, Rings and Meshes*, Proc. of the 30th Hawaii International Conference on System Sciences, 1996, to appear.

[3] T. Erlebach and K. Jansen, *Scheduling of Virtual Connections in Fast Networks*, Proc. of the 4th Parallel Systems and Algorithms Workshop (Jülich, Germany), 1996, to appear.

[4] K. Jansen, *Approximation Results for Wavelength Routing in Directed Trees*, Preprint, 1996.

[5] S. Jordan and E. Schwabe, *Worst-Case Performance of Cellular Channel Assignment Policies*, ACM Journal on Wireless Networks, 1996, to appear.

[6] C. Kaklamanis and G. Persiano, *Efficient Wavelength Routing on Directed Fiber Trees*, Proc. of the 4th Annual European Symp. on Algorithms, 1996, to appear.

[7] J. Kleinberg and E. Tardös, *Approximations for the Disjoint Paths Problem in High-Diameter Planar Networks*, Proc. of the 27th ACM Symp. on Theory of Computing, pp. 26-35, 1995.

[8] M. Mihail, C. Kaklamanis, and S. Rao, *Efficient Access to Optical Bandwidth*, Proc. of the 36th IEEE Symp. on Foundations of Comp. Sci., pp. 548-557, 1995.

[9] Y. Rabani, *Path Coloring on the Mesh*, Proc. of the 37th IEEE Symp. on Foundations of Comp. Sci., 1996, to appear.

[10] P. Raghavan and E. Upfal, *Efficient Routing in All-Optical Networks*, Proc. of the 26th ACM Symp. on Theory of Computing, pp. 134-143, 1994.