Constrained Bipartite Edge Coloring with Applications to Wavelength Routing *

Christos Kaklamanis[1] **Pino Persiano**[2]
Thomas Erlebach[3] **Klaus Jansen**[4]

[1] Computer Technology Institute, University of Patras, Rio, Greece, kakl@cti.gr
[2] Dipartimento di Informatica ed Appl., Università di Salerno, I-84081 Baronissi, Italy, giuper@dia.unisa.it
[3] Institut für Informatik, TU München, D-80290 München, Germany, erlebach@informatik.tu-muenchen.de
[4] Fachbereich IV – Mathematik, Universität Trier, Postfach 3825, D-54286 Trier, Germany, jansen@dm3.uni-trier.de

Abstract. Motivated by the problem of efficient routing in all-optical networks, we study a constrained version of the bipartite edge coloring problem. We show that if the edges adjacent to a pair of opposite vertices of an L-regular bipartite graph are already colored with αL different colors, then the rest of the edges can be colored using at most $(1+\alpha/2)L$ colors. We also show that this bound is tight by constructing instances in which $(1+\alpha/2)L$ colors are indeed necessary. We also obtain tight bounds on the number of colors that each pair of opposite vertices can see.

Using the above results, we obtain a polynomial time greedy algorithm that assigns proper wavelengths to a set of requests of maximum load L per directed fiber link on a directed fiber tree using at most $5/3L$ wavelengths. This improves previous results of [9, 7, 6, 10].

We also obtain that no greedy algorithm can in general use less than $5/3L$ wavelengths for a set of requests of load L in a directed fiber tree, and thus that our algorithm is optimal in the class of greedy algorithms which includes the algorithms presented in [9, 7, 6, 10].

1 Introduction

In this paper, we study a constrained version of the well-known problem of coloring the edges of an L-regular bipartite graph. It is a classical result from graph theory (see e.g. [3]) that the edges of an L-regular bipartite graphs can be colored using exactly L colors so that edges that share an endpoint are assigned different colors. We call such edge colorings *legal* colorings. The problem does not have any other extra constraint: any given color can be used on any edge provided that no other adjacent edge is colored using that same color. Our constrained version of the bipartite edge coloring problem can be described in the following way.

* Partially supported by Progetto MURST 40%, Algoritmi, Modelli di Calcolo e Strutture Informative and by EU Esprit Project GEPPCOM.

We are given an L-regular bipartite graph $G = (\{v_1, \cdots, v_n\}, \{u_1, \cdots, u_n\}, E)$ along with a partial legal coloring of its edges that specifies a color for all edges incident to vertices v_1 and u_1. We denote the total number of constraining colors by αL, where $1 \leq \alpha \leq 2$. We want to color the remaining edges of the graph so as to minimize the total number of colors used and the number of colors used to color the edges touching a pair (u_i, v_i) of opposite vertices.

Our motivation lies in the field of WDM (wavelength division multiplexing) routing in all-optical networks. Optics is emerging as a key technology in state-of-the-art communication networks. A single optical wavelength supports rates of gigabits-per-second (which in turn support multiple channels of voice, data, and video [5] [8]). Multiple laser beams that are propagated over the same fiber on distinct optical wavelengths can increase this capacity much further; this is achieved through WDM (wavelength division multiplexing). We model the underlying fiber network as a directed graph. Communication requests are ordered transmitter-receiver pairs of nodes. WDM technology establishes connectivity by finding transmitter-receiver paths, and assigning a wavelength to each path, so that no two paths going through the same link use the same wavelength. Optical bandwidth is the number of available wavelengths. Bandwidth is a scarce resource: state-of-the-art technology allows for no more than 30-40 optical wavelengths in the laboratory, less than half as many in manufacturing, and there is no anticipation of dramatic progress in the near future [11]. It is thus important to minimize the number of wavelengths used to service a requested communication pattern. Variations of this problem have been studied by several authors [12, 1, 9, 7, 6, 10, 2].

In this paper, we concentrate on tree topologies which are relevant to wide-area networks. In particular we consider directed trees where each edge of the tree consists of two opposite directed fiberlinks. Directedness accurately reflects directed optical amplifiers placed on the fiber as well as asymmetries of the communication requests. Raghavan and Upfal [9] showed that routing requests of maximum load L per link of undirected trees can be satisfied using no more than $3L/2$ optical wavelengths and their arguments extend to give a $2L$ bound for the directed case. Mihail *et al.* [7] were the first to address the directed case. Their main result is a $15L/8$ bound for directed trees. They obtain this bound by reducing the wavelength assignment problem to the constrained bipartite edge coloring problem and obtain a solution specifically for the case $\alpha = 3/2$. This was improved in [6] (and independently in [10]) by solving optimally the constrained bipartite edge coloring problem for the value $\alpha = 3/2$ and yielding a bound of $7/4L$ for directed trees.

1.1 Summary of results

Our results can be summarized in the following theorems. We first present our results on the constrained bipartite edge coloring problem.

Theorem 1. *There exists a polynomial time algorithm that properly colors the uncolored edges of an L-regular bipartite graph constrained by αL colors using* *at most $(1+\alpha/2)L$ colors and so that each pair (v_i, u_i) of opposite vertices sees no more than $\max\{\alpha L, (1+\alpha/4)L\}$ different colors.*

The next lower bound states that the above result is in general tight.

Theorem 2. *For each $1 \leq \alpha \leq 2$ and for each $L > 0$ there exists an L-regular bipartite graph constrained by αL colors for which any legal coloring of the remaining edges requires at least $(1+\alpha/2)L$ total colors while there exists a pair of opposite vertices that sees at least $\max\{\alpha L, (1+\alpha/4)L\}$ different colors.*

Next we present our results for wavelength routing on directed trees. We express our results in terms of the maximum load L of a set of requests; i.e., the maximum number of paths between transmitter and receiver that share the same directed fiber link. The proposed algorithm is a greedy algorithm. A greedy algorithm is an algorithm that considers the vertices of the tree one at a time in a DFS manner and, while at vertex v, colors (i.e., assigns a wavelength to) all the requests that touch vertex v (i.e., start at, end at, or go through v) that are still uncolored. Once a request has been colored, a greedy algorithm never recolors it. Greedy algorithms do not require global control and are thus amenable of being implemented in a distributed setting without a "central authority" that has knowledge of the overall request pattern. All known algorithms for the problem of wavelength routing on directed trees are indeed greedy algorithms [7, 6, 10].

Theorem 3. *There exists a greedy polynomial time algorithm that assigns wavelengths to a set of requests of maximum load L on a directed tree using at most $5/3L$ wavelengths.*

Our next theorem shows a lower bound that implies that no greedy algorithm can in general beat the $5/3L$ barrier.

Theorem 4. *For each L, for each $\epsilon > 0$ and for each greedy algorithm G there exists a tree and a pattern of communication requests of maximum load L for which G uses at least $\left(\frac{5}{3} - \epsilon\right) L$ wavelengths.*

Therefore better bounds can only be obtained by non greedy algorithms. The only known general lower bound is $5/4L$ [10].

The rest of our paper is organized as follows.

In Section 2, we prove Theorem 1 by giving an algorithm that solves the constrained bipartite edge coloring problem. Next, in Section 3 we explain the reduction of the wavelength routing problem on directed trees to the constrained bipartite edge coloring problem. This reduction proves Theorem 3. Finally, in Section 4, we present our lower bounds.

2 The algorithm for the constrained bipartite edge coloring problem

In this section we present our algorithm for solving the constrained bipartite edge coloring problem.

The algorithm receives as input an L-regular bipartite graph $G = (\{W_0, \cdots, W_n\}, \{X_0, \cdots, X_n\}, E)$ where all the edges incident to W_0 and X_0 have been properly colored using αL different colors. We call the edges that are colored *color-forced* edges and a pair (W_i, X_i) of opposite vertices a *line*. We assume without loss of generality that no edge connects two opposite vertices. If a color appears on only one color-forced edge, then we call it a *single* color. If it appears on two color-forced edges, we call it a *double* color; note that one of these two color-forced edges has to be incident to W_0 and the other to X_0. We denote by D and S the number of double and single colors, respectively.

Step 1: Obtaining perfect matchings. We proceed by decomposing the bipartite graph into L perfect matchings which can always be done since it is L-regular. Each such matching includes exactly two color-forced edges: one incident to W_0 and one incident to X_0. A double color is called *separated* if its two color-forced edges appear in different matchings. On the other hand, if they appear in the same matching then the color is said to be *preserved*. We classify the matchings into four types: TT, PP, SS, ST, based on their corresponding color-forced edges. If the two color-forced edges of a matching are colored with separated colors, then the matching is of type TT. If the two color-forced edges are colored with the same preserved color, then the matching is of type PP. If the two color-forced edges are colored with two single colors, then the matching is of type SS. If the two color-forced edges are colored with a single color and with a separated color, then the matching is of type ST.

Step 2: Constructing chains and cycles of matchings. We partition the matchings into groups. Each such group is either a *chain* or a *cycle* of matchings. A chain of matchings is a sequence $M_0, M_1, \cdots, M_{l-1}$ of l matchings such that

1. M_0 and M_{l-1} are matchings of type ST;
2. $M_1, \cdots, M_{l-2}$ are all matchings of type TT;
3. for each $0 \leq i \leq l-2$, matchings M_i and M_{i+1} share exactly one double (separated) color. A chain consists of at least two matchings.

A cycle of matchings is a sequence $\langle M_0, M_1, \cdots, M_{l-1}\rangle$ of l TT matchings such that, for each $0 \leq i \leq l-1$, matchings M_i and $M_{i+1 \bmod l}$ share exactly one double (separated) color.

Step 3: making chains and cycles minimal. A sequence C of matchings (chain or cycle) is minimal if it does not contain any two parallel color-forced edges. A non-minimal sequence of matchings can be split into two shorter sequences in the following way. Consider the sequence $C = \langle M_0, \cdots, M_{l-1}\rangle$ of matchings and suppose that the edge colored c_i of M_i and the edge colored c_j of M_j are parallel. We exchange the two edges thus obtaining two new matchings M_i' and M_j' with color-forced edges colored c_j and c_{i+1} and c_i and c_{j+1} and the two new sequences of matchings $C_1 = \langle M_0, M_1, \cdots M_{i-1}, M_j', M_{j+1}, \cdots, M_{l-1}\rangle$ and $C_2 = \langle M_i', M_{i+1}, \cdots, M_{j-1}\rangle$. The sequence C_1 is of the same type (i.e., a cycle or a chain) as C while C_2 is always a cycle. We repeat this process of splitting one sequence into two new sequences until all sequences are minimal (i.e., they do not contain parallel edges).

Step 4: constructing triplets of matchings. Next we partition all the matchings into groups of three matchings that we call *triplets*. Each such triplet has six color-forced edges; of these, two are colored with single colors and the remaining four with double colors.

We obtain the triplets as follows. First, we consider all the chains of length 3 or greater. From each such chain $C = \langle M_0, M_1, \cdots M_{l-1} \rangle$ we obtain one triplet by stripping off C and grouping together the first two matchings M_0, M_1 and the last matching M_{l-1}. Triplets obtained in this way will consist of two ST matchings (that is M_0 and M_{l-1}) and one TT matching (that is M_1). The color-forced edges are colored with single colors s_0 and s_1 and double colors d_1, d_2, d_{l-1}, with d_1 being the common color of M_0 and M_1. Now we are left with cycles, "stripped chains," chains of length 2, and SS matchings. We consider the even length cycles and stripped chains first and construct triplets each consisting of two consecutive TT matching from the same cycle or stripped chain and one SS matching. We repeat the same process for odd length cycles and stripped chains. However, in this case for each cycle or stripped chain there will be exactly one "leftover" TT matching. We then construct triplets with one SS matching along with a pair of these TT leftover matchings. Finally, if at any time during the construction of the triplets we run out of SS matchings, we continue constructing triplets by grouping together each individual TT matching along with a pair of ST matchings that constitute a chain of length 2. If the total number of old colors is exactly $4/3L$, thus including exactly $2/3L$ single colors and $2/3L$ double colors all matchings can be grouped into such triplets. Instead, if we have less than $4/3L$ old colors, then we are left with some extra TT matchings for which there is no corresponding ST or SS matching. These extra TT matchings will be dealt with separately and we omit from this abstract further details. On the other hand, if the number of old colors exceeds $4/3L$, then we are left with extra SS or ST matchings for which no corresponding TT matching exists. Coloring these matchings is trivial since we can use no new color (we use the single colors to color the uncolored edges) and thus meet the two conditions presented below.

We will color the matchings maintaining the following two conditions which are sufficient to prove Theorem 1.

Condition 1. The number of new colors used is at most $D/2$. This condition will be enforced by using at most one new color per triplet.

Condition 2. Each line sees at most $\max\{(1+\alpha/4)L, \alpha L\}$ colors.

For values of $\alpha \geq 4/3$ this is enforced by making sure that if a line sees a new color it does not see one of the old colors. Consequently, the number of colors seen by a line does not exceed αL once all edges have been colored.

Lemma 5. *Condition 1 above implies that the total number of colors used is at most* $(1+\alpha/2)L$.

Proof. Since the number of edges adjacent to W_0 and X_0 is $2L$, we have $2D+S = 2L$ and, since αL colors are used to color these edges, we have that $D+S = \alpha L$.

From these two equalities we get directly that $D = (2-\alpha)L$. Therefore the total number of colors used is at most $D + S + D/2 \leq \alpha L + (2-\alpha)L/2 = (1+\alpha/2)L$.

Step 5: setting the active colors. We color each triplet individually using four of the old colors that appear on the color-forced edges of the triplet and, in some cases, a new color. The four old colors used are called the *active* colors for the triplet and they include the two single colors of the triplet. The remaining two active colors are chosen among the double colors of the triplet so that each double color is active for exactly one triplet.

We continue by determining what the active colors are going to be for each triplet. We have to be careful about consistency among triplets that share double colors; i.e., include TT matchings from the same cycle or chain.

First we fix the active colors of the triplets containing the leftover TT matchings. In order to properly color such a triplet (S, T_1, T_2) while maintaining the properties above we choose the active colors to be the two single colors of matching S along with the color of the color-forced edge touching W_0 in T_1 and the color of the color-forced edge touching X_0 in T_2.

This choice of active colors for such a triplet forces the choice of active colors for the triplets containing TT matchings coming from the same cycle or chain as T_1 and T_2 in the following obvious way. Let (S, T_3, T_4) be a triplet consisting of one SS matching and two consecutive TT matchings from the same cycle or stripped chain as T_1. Then the active colors of such a triplet are the colors of the color-forced edges touching W_0 in T_3 and T_4 along with the two colors of the color-forced edges of S_1. If, instead, T_3 and T_4 belong to the same cycle or stripped chain as T_2, then the active colors are going to be the old single colors appearing in S_1 along with the color of the color-forced edges of T_3 and T_4 that touch X_0.

Finally, we can determine the active colors of the triplets containing two TT matchings belonging to even length cycles or chains (i.e., those cycles or chains that did not give rise to leftover TT matchings) to be for each triplet the two old single colors of the triplet along with the color of the color-forced edges that touch X_0 or W_0, picked arbitrarily as long as we are consistent across each cycle or chain.

Step 6: coloring the triplets. As we mentioned above for each triplet we will use the active colors of the triplet and, sometimes, a new color. If we do use a new color for a triplet, we enforce the property that each line that sees the new color does not see one of the active colors of the triplet. This ensures that the total number of colors that a line will see across all triplets does not exceed the number of old colors and that the total number of new colors introduced for all triplets is at most half the number of old double colors.

There are four general types of triplets:

Type A These are triplets consisting of one SS matching and two leftover TT matchings. A special case of type A triplet occurs when one or both the leftover TT matchings is actually a PP matching that is the leftover matching of a cycle of length 1.

Type B These are triplets consisting of one SS matching and two consecutive TT matchings from the same cycle or chain. A special case of type B triplet occurs when the two TT matchings constitute a cycle of length 2.

Type C These are triplets consisting of the two ST matchings that constitute a chain of length 2 and one TT matching. A special case of type C triplet occurs when the TT matchings is actually a PP matching.

Type D These are triplets that were obtained by stripping off a chain the first two matchings (an ST and a TT matching) and the last matching (an ST matching). A special case of type D triplet occurs when the chain has length exactly 3.

Due to lack of space we next show the coloring algorithm only for triplets of type A. The complete coloring appears in the final version.

Step 6.A: coloring triplets of type A. Consider a triplet $R = (S, T_1, T_2)$ of type A, where $S = (s_1, s_2)$, $T_1 = (x, y)$ and $T_2 = (w, z)$. We note that s_1 and s_2 are single colors and that x, y, w, and z are double colors and let the active colors of R be s_1, s_2, x, and z. Here we concentrate on the case in which the four double colors are distinct separated colors. If $x = y$ or $w = z$, then the corresponding TT matching is actually a PP matching and the coloring is much simpler than what we are going to describe below. If $x = z$ or $y = w$, then R is actually a triplet of type B.

Suppose x, y, w, and z are distinct separated double colors. We consider matchings T_1 and T_2 together as one cycle cover of the bipartite graph. In what follows, for the sake of clarity we assume that the cycle cover of two matchings consists of one single cycle that spans the entire bipartite graph. We remark that all our colorings can be easily adapted if such a cycle cover consists of more than one cycle.

We first check if there exists an uncolored edge whose endpoints are incident to color-forced edges colored with all four active colors. Note that these may include the "fixed" color-forced edges colored with s_1, s_2, x, and z that belong to R as well as the two "free" color-forced edges colored with x and z that belong to other triplets. We denote by e_x and e_z the free color-forced edge colored with x and z, respectively and by e_{s_1} and e_{s_2} the color-forced edges colored with s_1 and s_2, respectively.

Suppose there is no edge restricted by all four active colors. We color the uncolored edges of the cycle cover by starting from one of the color-forced edges of the cycle colored with an active color (i.e., either x or z) and alternating between x and z. When we encounter a vertex v that is incident to a free color-forced edge e, we use color s_2 to color the edge, e', incident to v that would have been colored with the same color as e. Then we color the next edge x and continue alternating between z and x. This is possible unless e' is adjacent to e_{s_2} as well. Note that e_z cannot be incident to the same vertex as e_{s_2}, and, similarly, e_x cannot be incident to the same vertex as e_{s_1}.

Now if e' is restricted by both z and s_2, then we color with s_2 the other edge incident to v, color e' with x and continue alternating z and x (see Figure 1);

we finish by using s_1 to color the edges in the SS matching. This coloring is obviously proper and we do not need to argue about the number of colors seen by a line since we have used no new color.

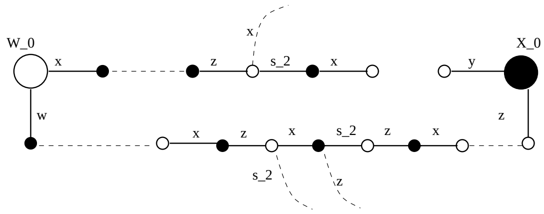

Fig. 1. The case in which an edge is restricted by both z and s_2.

On the other hand, if e_x and e_{s_2} are incident to the same vertex v then we color the conflicting adjacent edge $e = (v, u)$ with s_1 and continue alternating x and z starting with x. The uncolored edges of the SS matching are then colored using s_1 except for the edge $e_u = (u, u^*)$ incident to u. Edge e_u is colored s_2 unless u^* sees the edge colored s_2 used to fix the conflict with e_z in which case e_u is colored with z (see Figure 2).

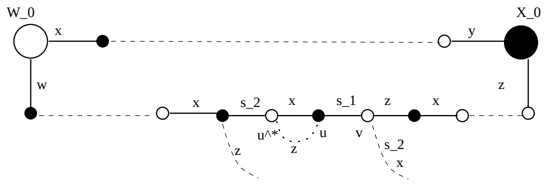

Fig. 2. The case in which an edge is restricted by both x and s_2.

The previous coloring is proper unless e is also adjacent to e_{s_1} in which case a more complex coloring is performed.

Finally we consider the situation where we have an edge (u, v) restricted by all four active colors. Note that such a restricted edge belongs to one of the TT matchings of the triplet, as edges of the SS matching cannot be restricted by s_1 or s_2. We color edge (u, v) with n, the rest of the uncolored edges of the cycle cover by alternating x and z, and the uncolored edges of the SS matching with n. This coloring is obviously proper. No line except the lines containing vertices u and v sees an edge colored with one of the two single colors. Moreover, since u and v cannot be a line as they are adjacent, the line containing u does not see color s_2 and the line containing v does not see color s_1. Therefore, if a line sees n then it does not see at least one of the active colors.

2.1 An alternative coloring approach for $\alpha = 4/3$

In this section, we briefly describe an alternative method for coloring edges of a bipartite graph G for the case $\alpha = 4/3$.

It is possible to show that the L perfect matchings obtained from G can be grouped entirely into triplets such that each triplet can be colored with at most one new color and with ≤ 4 colors per line, thus ensuring Conditions 1 and 2. Due to a result in [10], every triplet $t = (M_1, M_2, M_3)$ with two double colors and one single color incident to each of W_0 and X_0 can be colored in such a way, provided that at least one double color d appears twice in t (call such a triplet a *KS-triplet*) and that t can be partitioned into a gadget (a subgraph where W_0 and X_0 have degree 3 and all other vertices have degree 2) and a matching of all vertices except $\{W_0, X_0\}$. Only the two single colors and the double color d of t as well as one new color are used. Every KS-triplet can be partitioned into gadget and matching unless it contains a PP-matching. Therefore, we assume that G is decomposed into L perfect matchings such that the union of SS-, ST-, and TT-matchings does not contain further PP-matchings.

A PP-matching and a chain of length 2 as well as two PP-matchings and an SS-matching give triplets that can be colored without any new color and 4 colors per line. Chains of odd length and cycles of even length yield triplets of Type B, C, and D, which are KS-triplets. Two chains of even length, one of which has length > 2, yield KS-triplets by combining the first (last) two matchings of the longer chain with the first (last) matching of the shorter chain and producing triplets of Type B or C from the rest. If there is a chain of length 2, a cycle of odd length also yields triplets of Type B and C. Note that there is always a sufficient number of SS-matchings or chains of length 2 to produce KS-triplets, because we have $2/3L$ edges with single colors and $4/3L$ edges with double colors altogether.

After these reductions, we are left with at most one chain of even length > 2, at most one PP-matching, a number of cycles of odd length, and SS-matchings. Two cycles of odd length are handled by choosing an SS-matching and two TT-matchings, one from each cycle, such that the resulting triplet t does not have parallel color-forced edges. If t can be partitioned into gadget and matching, it is colored with reused old colors and one new color using techniques similar to [10], and triplets of Type B are produced from the remainder of the two cycles. Otherwise, the TT-matchings can be reassembled, turning the two given cycles into a single cycle of even length, which is handled as above. A chain of even length > 2 and a cycle of odd length are combined similarly.

For a PP-matching and a cycle of odd length, we choose an arbitrary SS-matching M_1 and a TT-matching M_2 from the cycle such that the cycle cover $M_1 \cup M_2$ does not contain parallel color-forced edges. This cycle cover can be colored with one new color and one of its single colors such that no line sees more than 3 colors. The PP-matching M_3 is colored using its preserved double color, thus ensuring that the coloring for $t = (M_1, M_2, M_3)$ meets the requirements. The remainder of the cycle is combined with SS-matchings into Type B triplets. A PP-matching and a chain of even length > 2 are handled similarly.

3 Reducing the routing problem to a constrained bipartite coloring problem

In this section we reduce the problem of assigning wavelengths to the constrained bipartite edge coloring problem. We do so by giving an algorithm that properly assigns wavelengths by using as a subroutine our algorithm for the constrained bipartite edge coloring problem of the previous section.

Our algorithm for assigning wavelengths is a greedy algorithm as the ones presented in [7, 6, 10]. The algorithm roots the tree at an arbitrary node and computes a depth-first numbering of the nodes of the tree. The algorithm proceeds in phases, one per each node v of the tree. The nodes are considered following their depth first numbering. The phase associated with node v assumes that a partial proper coloring of all paths that touch (i.e., start, end, or go through) nodes with numbers strictly smaller than v's has been computed and extends the partial coloring to one that assigns proper colors to all paths that touch v but have not been colored yet. We stress that the algorithm never recolors paths that have been colored in previous phases.

We now show the reduction of the path coloring problem of a phase associated with node v to an instance of the constrained bipartite edge coloring of a graph G_v. Without loss of generality, we assume to have full load L on each directed link and denote by c_0 v's parent and by $c_1, \cdots, c_k$ the children of v. We construct G_v in the following way. For each vertex c_i, G_v has four vertices W_i, X_i, Y_i, Z_i and the left and right partitions are $\{W_i, Z_i | i = 0, \cdots k\}$ and $\{X_i, Y_i | i = 0, \cdots k\}$. G_v has an edge from W_i to X_j, for each path of the tree directed out of c_i into c_j and an edge from W_i to Y_i, for each path from c_i to v. Finally, for each path from v to c_j, G_v has an edge from Z_i to X_i. See Figure 3. The above edges are called *real*. Notice that no real edge extends across opposite vertices Z_i and Y_i or W_i and X_i and only edges with an endpoint in W_0 or X_0 already have a color as they correspond to requests touching v's parent and have been assigned a color in a previous phase. Notice also that all vertices of type W_i and X_i have degree L whereas vertices of type Z_i and Y_i do not necessarily have degree L. We therefore add *fictitious edges* to the bipartite graph so that all vertices have degree L. Clearly, any proper coloring of the edges of G_v corresponds to a legal assignment of wavelengths to requests that go through vertex v and we compute such a coloring of the edges of G_v by running the algorithm of the previous section on G_v.

4 Lower bound

In this section we present our lower bounds for the wavelength routing problem by showing that any greedy algorithm for assigning paths to requests of load L on a tree cannot use less than $5/3L$ colors even if the tree is binary. The lower bound for the constrained bipartite edge coloring is obtained similarly.

We prove the lower bound inductively. We assume inductively that, for a vertex C there are $\alpha_n/2L$ requests along each link to its parent and that all of these requests are colored using different colors.

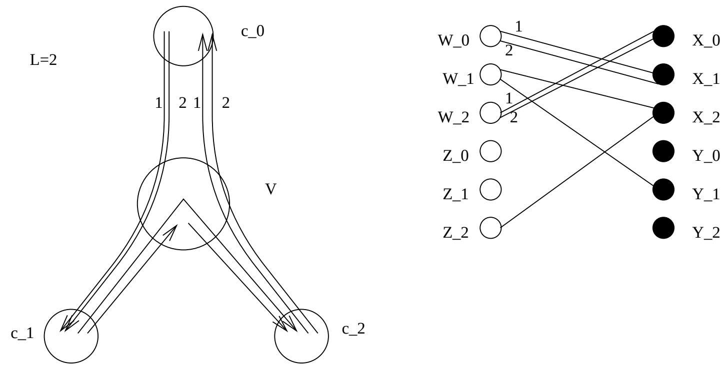

Fig. 3. Requests touching vertex v and the corresponding bipartite graph (only real edges are shown).

Then we assign requests between the two children A and B of C in such a way that $(1+\frac{\alpha_n}{2})L$ colors are used in total and the inductive hypothesis between one of A and B and one of its children is enforced for $\alpha_{n+1} = 1 + \frac{\alpha_n}{4}$. It is easy to see that $\alpha^* = \lim_{n\to\infty} \alpha_n = 4/3$, where $\alpha_1 = 1$ and $\alpha_n = 1 + \frac{\alpha_{n-1}}{4}$ for $n > 1$. Therefore, for any $\epsilon > 0$ and any greedy algorithm G, it is possible to construct a set of communication requests of maximum load L so that G uses at least $(5/3 - \epsilon)L$ colors.

The base of our induction for $\alpha_1 = 1$ is established in the following way. We start with L requests on each direction between the root R and one of its children R'. The greedy algorithm colors these request using at least L colors in each direction. We then choose two sets of $L/2$ request along each direction with each request colored with a different color, propagate them to one of the children of R' and stop at R' the remaining requests.

Let C be a vertex and A and B the two children of its left child. We denote by K_2 the set of colors used along the link (C, A), by K_3 the set of colors used along the link (B, C), by K_1 the set of colors used along the link (B, A) and by K_4 the set of colors used along the link (A, B) . We inductively assume that $K_2 \cap K_3 = \emptyset$ and $|K_2| = |K_3| = \alpha/2L$ whence thus $|K_2 \cup K_3| = \alpha L$. We fill the link (B, A) to capacity by assigning $k_1 = L(1-\alpha/2)$ requests. These requests need to be colored with new colors and thus the total number of colors used increases to $L(1 + \frac{\alpha}{2})$. Next we assign L requests to the link (A, B). The best that any greedy algorithm can do is to color these L requests colored using all the new colors employed for the link (B, A), plus half of the colors of K_3 and half of the colors of K_2. The edge (A, C) thus sees $|K_1 \cup K_2 \cup K_4| = |K_1| + |K_2| + |K_4| - |K_2 \cap K_4| - |K_1 \cap K_4| = (1 + \frac{\alpha}{4})L$. In order to complete the inductive step we have to enforce for A the same situation as in C for $(1 + \frac{\alpha}{4})$. This is achieved in the following way:

1. among the $|K_1| + |K_2| = L$ requests coming from C, we let only the following continue to the left child of A:
   - S_1: $\left(\frac{1}{2} - \frac{\alpha}{8}\right) L$ requests from K_1.

– S_2: $\frac{\alpha}{4}L$ requests from K_2.

for a total of $\frac{1}{2}\left(1+\frac{\alpha}{4}\right)$ colors.

2. and the $|K_4| = L$ requests coming up from A to C all originate from A except for the following ones which instead come from the right child of A
   – R_1: $\frac{\alpha}{4}L$ requests that were colored with colors used of K_2 and which were not considered in S_2 above;
   – R_2: $\frac{\alpha}{4}L$ requests that were colored with colors used of K_3;
   – R_3: $\left(\frac{1}{2}-\frac{3}{8}\alpha\right)L$ that were colored with colors of K_1 and which were not considered in S_1 above;

   for a total of $\frac{1}{2}\left(1+\frac{\alpha}{4}\right)$ colors.

Finally, observe that the requests going down to the left child of A and those coming up from the right child of A are colored with different colors (i.e. the sets of colors are disjoint). This completes the proof of Theorem 4.

References

1. A. Aggarwal, A. Bar-Noy, D. Coppersmith, R. Ramaswani, B. Shieber, and M. Sudan, *Efficient Routing and Scheduling Algorithms for Optical Networks*, in Proc. of SODA 93.
2. B. Beauquier, J.-C. Bermond, L. Gargano, P. Hell, S. Perennes, U. Vaccaro, *Graph Problems arising from Wavelength-Routing in All-Optical Networks*, in proc. of Workshop on Optics in Computer Science, 1997.
3. C. Berge, *Graphs*, North-Holland.
4. J.-C. Bermond, L. Gargano, S. Perennes, A. A. Rescigno, and U. Vaccaro, *Efficient Collective Communication in Optical Networks*, in Proc. of ICALP 96.
5. P. E. Green, *Fiber Optic Communication Networks*, Prentice Hall, 1992.
6. C. Kaklamanis and P. Persiano, *Efficient Wavelength Routing on Directed Fiber Trees*, in Proc. of Algorithms – ESA '96, Lecture Notes in Computer Science, 1136, pp. 460–470.
7. M. Mihail, C. Kaklamanis, and S. Rao, *Efficient Access to Optical Bandwidth*, in Proc. of FOCS 1995.
8. D. Minoli, *Telecommunications Technology Handbook*, Artech House, 1991.
9. P. Raghavan and E. Upfal, *Efficient Routing in All-Optical Networks*, in Proc. of STOC 1994.
10. V. Kumar and E. J. Schwabe. Improved access to optical bandwidth in trees. In *Proceedings of SODA '97*, 1997.
11. ONTC-ARPA, Brackett, Acampora, Sweitzer, Tangonan, Smith, Lennon, Wang, Hobbs, A Scalable Multiwavelength Multihop Optical Network: A Proposal for Research in All-Optical Networks, *IEEE J. of Lightwave Technology*, Vol 11 No 5/6, 1993, pp 736-753.
12. R. Pankaj, *Architectures for Linear Lightwave Networks*, Ph.D. Thesis, MIT, 1992.

This article was processed using the LaTeX macro package with LLNCS style