Wavelength Conversion in Optical Networks

Jon Kleinberg [*] Amit Kumar [†]

Abstract

In many models of optical routing, we are given a set of *communication paths* in a network, and we must assign a wavelength to each path so that paths sharing an edge receive different wavelengths. The goal is to assign as few wavelengths as possible, in order to make as efficient use as possible of the optical bandwidth.

Wilfong and Winkler considered the problem of placing *wavelength converters* in such a network: if a node of the network contains a converter, any path that passes through this node may change its wavelength. Having converters at some of the nodes can reduce the number of wavelengths required for routing, down to the following natural *congestion bound*: even with converters, we will always need at least as many wavelengths as the maximum number of paths sharing a single edge. Thus Winkler and Wilfong defined a set S of nodes in a network to be *sufficient* if, placing converters at the nodes in S, every set of paths can be routed with a number of wavelengths equal to its congestion bound. They showed that finding a sufficient set of minimum size is NP-complete.

In this paper, we provide a polynomial-time algorithm to find a sufficient set for an arbitrary directed network whose size is within a factor of 2 of minimum. For the special case of planar graphs with bi-directional edges, we obtain a polynomial-time approximation scheme. Our techniques establish a connection between the problem of finding a minimum sufficient set and an interesting simultaneous generalization of the Vertex Cover and Feedback Vertex Set problems in undirected graphs.

1 Introduction

Routing connections in an optical communication network involves a host of optimization problems that arise from the issue of *wavelength assignment*. In this paper, we focus on *wavelength division multiplexing* (WDM) networks [6, 10]. A basic type of problem that arises in such networks is the following: Given a set of *communication paths*, we wish to assign a *wavelength* to each path in such a way that if two paths share an optical link, then they must be assigned different wavelengths. We will call such a wavelength assignment *valid*.

[*]Department of Computer Science, Cornell University, Ithaca NY 14853. Email: kleinber@cs.cornell.edu.
Supported in part by an Alfred P. Sloan Research Fellowship and by NSF Faculty Early Career Development Award CCR-9701399.

[†]Department of Computer Science, Cornell University, Ithaca NY 14853. Email: amitk@cs.cornell.edu.

The problem of minimizing the number of wavelengths required under this constraint is equivalent to a graph-coloring problem on the *conflict graph* of the set of paths — the graph whose nodes are the given paths, with an edge between each pair of paths that share a link in the network. For a set $\mathcal{P}$ of paths in a network, we will use $\chi(\mathcal{P})$ to denote the minimum number of wavelengths required in a valid assignment. (It is more accurate to denote this by $\chi_G(\mathcal{P})$, where G is the underlying network, but we will omit the subscript when it is clear from context.) We note that there is a related problem, in which we are given only the pairs of nodes that wish to communicate; we must then choose a path joining each such pair, and a valid wavelength assignment for the set of paths, so that the number of wavelengths is minimized. Approximation algorithms have been developed for a number of different settings of this latter problem; see e.g. [1, 7, 8, 9, 11, 12].

A natural lower bound on the number of wavelengths required for a set $\mathcal{P}$ is paths is its *load*, or *congestion*, defined to be the maximum number of paths passing through any one link in the network. We will use $\nu(\mathcal{P})$ to denote this quantity. $\chi(\mathcal{P})$ is at least $\nu(\mathcal{P})$, since a different wavelength must be assigned to each path on the most heavily loaded link in the network, and $\chi(\mathcal{P})$ can in general be much larger than $\nu(\mathcal{P})$. For example, it is not difficult to construct a set $\mathcal{P}$ of n paths in a planar grid graph with the property that $\nu(\mathcal{P}) = 2$, but each pair of paths shares an edge, and hence $\chi(\mathcal{P}) = n$.

In many models of optical routing, one can place *wavelength converters* at certain of the nodes; if there is a converter at node v, then any path containing v can change its wavelength as it passes through v. In a network with converters, our notion of a *wavelength assignment* must become more general: it is now an assignment of a wavelength to each *edge* of each path, with the restriction that it must be constant on any sub-path that does not pass through a converter. We will use $\chi_{G,S}(\mathcal{P})$ to denote the minimum number of wavelengths required for a set $\mathcal{P}$ of paths, in a network G, with converters at a subset S of the nodes; as before, we will abbreviate this to $\chi(\mathcal{P})$ when G and S are clear from context. Clearly wavelength assignments in networks with converters can sometimes be more efficient (i.e. use fewer wavelengths) than optimal wavelength assignments for the same set of paths when no converters are available. For example, it is possible to prove that placing a converter at any node of a directed ring has the effect that $\chi(\mathcal{P}) = \nu(\mathcal{P})$ for any set $\mathcal{P}$ of paths.

Motivated by this notion, Wilfong and Winkler [13] proposed the following definition: A subset S of the nodes of a directed graph $G = (V, E)$ is *sufficient* if $\chi_{G,S}(\mathcal{P}) = \nu_G(\mathcal{P})$ for all sets of paths $\mathcal{P}$ — that is, if the number of wavelengths required is equal to the load, for all sets of paths, when converters are placed at the nodes of S.

A basic optimization problem in optical network design then becomes the following: Given a network G, represented by a directed graph $G = (V, E)$, find a sufficient set for G of minimum size. Wilfong and Winkler proved that this problem, Minimum Sufficient Set, is NP-complete; and it remains NP-complete even when G is *bi-directed* in the sense that (u, v) is an edge if and only if (v, u) is an edge. The existence of good approximation algorithms for the problem was left open.

Our Results. We provide a polynomial-time algorithm that produces a sufficient set of at most twice the minimum size, in an arbitrary directed network. This is done by exploiting a connection between Minimum Sufficient Set and an optimization problem that we call Generalized Feedback Set, a simultaneous generalization of the Vertex Cover and Feedback Vertex Set problems for undirected graphs. For Generalized Feedback Set, we develop a 2-approximation algorithm using a *primal-dual* method, based on an approach of Chudak, Goemans, Hochbaum, and Williamson [4] for the Feedback Vertex Set problem.

We begin by providing a simple 2-approximation algorithm for Minimum Sufficient Set in the special case of bi-directed graphs. Here we exploit a tight connection between this problem in bi-directed graphs and the Vertex Cover problem for undirected graphs. As a consequence of this connection, we also obtain a polynomial-time approximation scheme for Minimum Sufficient Set in bi-directed planar graphs, and we show that the problem is *fixed-parameter tractable* in the sense of Downey and Fellows [5]: there is an algorithm with running time $O(f(k) \cdot p(n))$, where p is a polynomial function. Finally, we give an approximation-preserving reduction from Vertex Cover to Minimum Sufficient Set in bi-directed graphs; since providing an approximation ratio better than 2 for Vertex Cover is a long-standing open problem, this indicates that improving on our performance guarantee for Minimum Sufficient Set will be difficult as well.

2 Preliminaries

A *network*, in this paper, is a directed graph $G = (V, E)$. For certain pairs of nodes u and v, G may contain edges (u, v) and (v, u); this will be called a *bi-directed pair* of edges. G will be called *bi-directed* if each edge belongs to a bi-directed pair. The *skeleton* of the network G, denoted $s(G)$, is the undirected graph obtained from G by ignoring the directions of the edges, and replacing each bi-directed pair of edges by a single undirected edge. For the sake of simplicity, we will assume that $s(G)$ is connected for all networks G that we deal with. We will also assume that G does not contain multiple copies of the same edge in the same direction.

Let $\mathcal{P}$ denote a set of directed paths in G. We will assume throughout that no path in $\mathcal{P}$ passes through any vertex more than once. Recall that the *conflict graph* of $\mathcal{P}$ is an undirected graph whose vertices are the set of paths in $\mathcal{P}$ and two paths P_i and P_j are joined by an edge if and only if they share an edge.

3 Bi-directed Graphs

As discussed in the introduction, we begin with the special case of a *bi-directed graph* $G = (V, E)$: one for which $(u, v) \in E$ if and only if $(v, u) \in E$. Let $G_s = (V, E')$ denote $s(G)$. By partial abuse of terminology, we will say a set is *sufficient* in G_s if and only if it is sufficient in G.

Wilfong and Winkler [13] developed some basic properties of converters in a bi-directed graph; we restate one of their results here, as it will be useful in what follows. We begin with some definitions from [13]. For a subset S of V, define $G_s(S) = (V(S), E'(S))$ as the graph in which $V(S)$ is the set of nodes in $V - S$, together with the set of pairs (s, e) for which $s \in S$ and $e \in E'$ is an edge incident on s. $E'(S)$ consists of (i) edges of G_s between vertices in $V - S$; (ii) edges of the form

$((s,e),v)$ whenever $(s,e) \in V(S)$ and $e=(s,v)$; and (iii) $((s,e),(t,e))$ where s and t are in S. Define a *spider* to be a tree with at most one vertex of degree greater than 2. Now the following theorem is proved in [13].

Theorem 3.1 *S is sufficient for G if and only if each component of $G(S)$ is a spider.*

We now give a 2-approximation for Minimum Sufficient Set in bi-directed graphs, and in the following sections extend the 2-approximation to general graphs.

Define a vertex v to be a *branching node* if its degree in $s(G)$ is greater than 2. Analogously, define a *straight node* to be a node whose degree in $s(G)$ is less than or equal to 2. (We will also refer to a node of degree 1 as a *leaf*.) We will assume that G_s contains at least one branching node, since otherwise G_s is either a path or a cycle, and Minimum Sufficient Set is easy to solve optimally. We say that a node of a path P is an *internal node* if it is not one of the two endpoints.

Lemma 3.2 *If S is a sufficient set for G_s, we can find another sufficient set S' for G_s such that $|S'| \leq |S|$ and S' does not contain any straight node.*

Proof. Suppose S contains a straight node v. Let v' be a branching node which is not in S such that the path between v and v' consists of straight nodes only - if there is no such v', then we can remove v from S and still have a sufficient set. Consider $S' = (S - \{v\}) \cup \{v'\}$. It can be easily seen that S' is a sufficient set also. ■

We will say that a sufficient set is *canonical* if it contains only branching nodes.

We construct a graph $H = (V_H, E_H)$ as follows. V_H consists of all branching nodes in G_s. For $u, v, \in V_H$, (u,v) is an edge in E_H if and only if there exists a path in G_s between u and v such that all internal nodes in the path are straight nodes. Note that H may have self-loops, which we retain as part of the graph.

Lemma 3.3 *Let S be a canonical sufficient set, and consider S as a subset of V_H. Then S is a vertex cover of H. Conversely, every vertex cover of H is also a sufficient set of G_s. (By definition, we require a vertex cover to include the unique endpoint of each self-loop.)*

Proof. Suppose S is a canonical sufficient set of G, but is not a vertex cover of H. Then there exists an edge (u,v) in H such that neither u nor v are in S. Now, in G_s, u and v are branching nodes such that there is a path P between them containing no other branching node. Since S cannot contain any straight node, none of the nodes in P connecting u and v are in S. When we form the graph $G_s(S)$, all the nodes in P — including u and v — will be in the same component. Moreover, the degrees of both u and v will be the same as that in G. Thus, if $u = v$, this would be a directed cycle in a component of $G_s(S)$; if $u \neq v$, this would be two branching nodes in a component of $G_s(S)$. In both cases, this is a contradiction. So, S must be a vertex cover of H.

Conversely, suppose C is a vertex cover for H, but not a sufficient set of G_s. Then there is a component Z of $G_s(C)$ that is not a spider. Note that any non-leaf node of Z corresponds to a node of the same degree in $V - S$. If Z contains a single branching node v, then Z must contain a cycle including v. But this implies that v is the endpoint of a self-loop of H; since we also know $v \notin C$, this contradicts the assumption that C

is a vertex cover. Otherwise, Z contains more than one branching node. Choose a pair $u, v \in Z$ of branching nodes whose distance in Z is minimal. Thus, there is a u-v path P in Z whose internal nodes all have degree 2. These nodes must also have degree two in G_s, and hence (u, v) is an edge of H. But this again contradicts the assumption that C is a vertex cover. ■

Given Lemma 3.3, it is easy to obtain a 2-approximation algorithm for MINIMUM SUFFICIENT SET. One first constructs the graph H from G_s, then computes a vertex cover C for H that is within a factor of 2 of minimum, and then returns C as sufficient set for G.

Two other consequences of Lemma 3.3 are the following. First, it is well-known that it requires time $O(f(k) \cdot p(n))$ to decide whether a graph has a vertex cover of size at most k (see e.g. [5]); thus we obtain a similar running time for MINIMUM SUFFICIENT SET in bi-directed graphs. Second, H is a contraction of G_s, so H is planar if G_s is. Since there exists a polynomial-time approximation scheme for VERTEX COVER in planar graphs [2, 3], Lemma 3.3 implies that there is also one for MINIMUM SUFFICIENT SET in bi-directed planar graphs.

The Minimum Sufficient Set Problem is as Hard as Vertex Cover

We now give an approximation-preserving reduction from VERTEX COVER to MINIMUM SUFFICIENT SET; this shows that improving on the approximation ratio of 2 for MINIMUM SUFFICIENT SET, even in bi-directed graphs, will be quite difficult.

Consider an instance of VERTEX COVER : $H = (V, E)$ is an undirected graph and k is an integer. We construct a new graph $H' = (V', E')$ as follows. Let $V_1' = V$, V_2' denote the set $\{x_1(e), x_2(e) | e \in E\}$, $V' = V_1' \cup V_2'$, and

$$E' = \{(u, x_1(e)), (u, x_2(e)), (x_1(e), v), (x_2(e), v) | e = (u, v) \in E\}$$

In other words, we split each edge into two edges and place new vertices on each of these edges. An example is shown in the figure below :

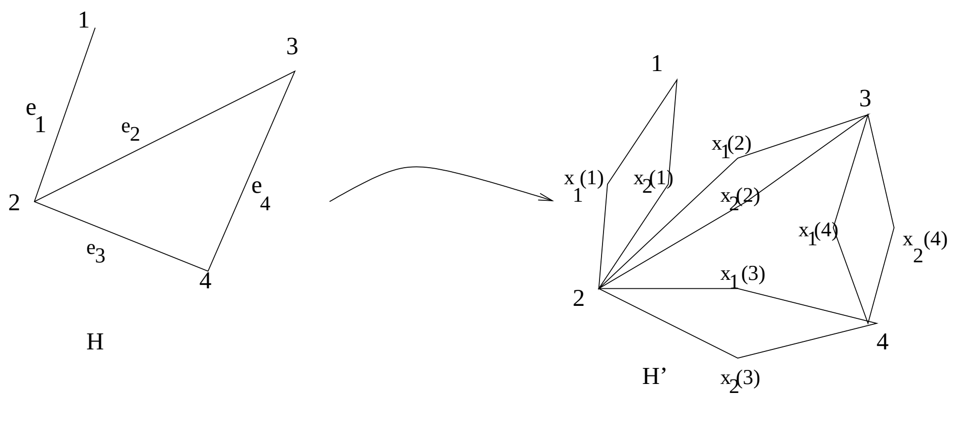

Figure 1: The construction for proving NP-completeness of SUFFICIENT SET

Theorem 3.4 *H has a vertex cover of size k if and only if H' has a sufficient set of size k.*

Proof. Suppose C is a vertex cover of H of size k. Then in each component of $H'(C)$, there is at most one node of $V_1' - C$; since V_2' consists only of straight nodes, it follows that each component of $H'(C)$ is a spider.

Conversely, suppose C' is a sufficient set for H'. By Lemma 3.2 we can assume that C' does not contain any straight nodes; i.e., C' is a subset of V_1'. If C' is not a vertex cover of H, then let $e = (u, v)$ be an edge in H such that C' doesn't contain u or v. Then, $u, x_1(e), v, x_2(e)$ is a cycle in H'. But this cycle will also be present in $H'(C)$, which contradicts our assumption that C' is a sufficient set. So, C must be a vertex cover of H. ■

4 General Graphs

We now consider the case of a general directed graph $G = (V, E)$. First note that on a directed cycle, one can place a set $\mathcal{P}$ of three paths so that $\nu(\mathcal{P}) = 2$, but each pair of paths meets at some edge, and hence $\chi(\mathcal{P}) = 3$. Thus, any sufficient set must contain at least one node from each directed cycle of G.

We now look at a different type of subgraph that also necessitates a converter. Consider a graph consisting of a bi-directional path with edges coming in and out at its two ends. For concreteness, let the path be P and its ends be u and v. Suppose that we have edges coming in and out at both ends, i.e., (u, u_1) and (u_2, u) are edges incident on u ($u_1 \neq u_2$). Similarly, $(v, v_1), (v_2, v)$ are incident on v ($v_1 \neq v_2$). Let us call such a graph an *$\mathcal{H}$-graph*. The bi-directional path P of an $\mathcal{H}$-graph will be called its *characteristic path*.

As shown in Figure 2, one can place 5 paths in an $\mathcal{H}$-graph whose conflict graph is a 5-cycle. Thus we have

Lemma 4.1 *Let G be a directed graph, S a sufficient set for G, and K an $\mathcal{H}$-graph in G with characteristic path P. Then $S \cap P \neq \phi$.*

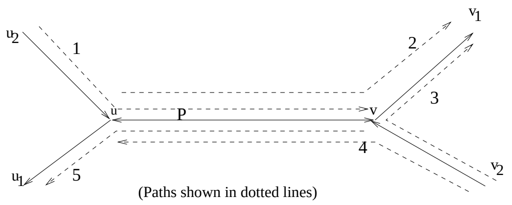

Figure 2: An $\mathcal{H}$-graph

If K and K' are $\mathcal{H}$-graphs in G, we say that K' *encloses* K if the characteristic path of K is a subset of the characteristic path of K'. Enclosure, defined in this way, imposes a partial order on the $\mathcal{H}$-graphs of G, and we say an $\mathcal{H}$-graph is *minimal* if it is minimal with respect to enclosure.

Lemma 4.2 *(i) If K is a minimal $\mathcal{H}$-graph in G, with characteristic path P, then there is no edge of $G - K$ incident on an internal node of P.*

(ii) The characteristic paths of two minimal $\mathcal{H}$-graphs are edge-disjoint.

Proof. The second statement follows directly from the first. To prove the first statement, suppose there were any edge e of $G - K$ incident on an internal node w of P. Then the sub-path between w and either end of P would form the characteristic path of an $\mathcal{H}$-graph in G, contradicting the minimality of K. ■

Lemma 4.3 *Suppose $s(G)$ is a tree. Then the empty set is sufficient for G if and only if G has no $\mathcal{H}$-graph.*

Proof. We know that if the empty set is sufficient, then G has no $\mathcal{H}$-graph.

We prove the converse by induction on the number of edges of G. An edge (u, v) will be called *internal* if neither u nor v is a leaf. First suppose G has an internal edge (u, v) for which $(v, u) \notin E$. Let $\mathcal{P}$ be a set of paths. If we delete the edge (u, v), $s(G)$ is partitioned into two components, R_1 and R_2. Construct two subgraphs of G: G_1, equal to the union of R_1 and the edge (u, v); and G_2, equal to the union of R_2 and (u, v). Since (u, v) is an internal edge, both these subgraphs have fewer edges than G. Restrict $\mathcal{P}$ to each subgraph as follows : if there is a path through (u, v), we only retain the edges of the path that lie in the respective subgraph. Paths not using (u, v) remain unchanged.

Now, by induction hypothesis, we can color the paths in G_1 and G_2 with at most $\nu(\mathcal{P})$ colors. We can now merge these two subgraphs by permuting the labels of colors used in G_1 so that paths using (u, v) receive the same color in both subgraphs.

Thus we may consider the case in which all internal edges come in bi-directional pairs. If G contains at most one branching node, then it is a subgraph of a spider. Hence we may further focus on the case in which G has at least two branching nodes; if we consider two branching nodes u, v at minimum distance in G, then the path P between them is bi-directional, consists only of straight nodes, and has the property that at one end, say u, all other incident edges come in the same direction. We now conclude by induction as follows. As shown in the figure below, create two copies of u: u' and u''. Replace P by two bi-directional paths - from u' to v and from v to u''. Moreover, all incoming edges at u are now incident on u' (see figure).

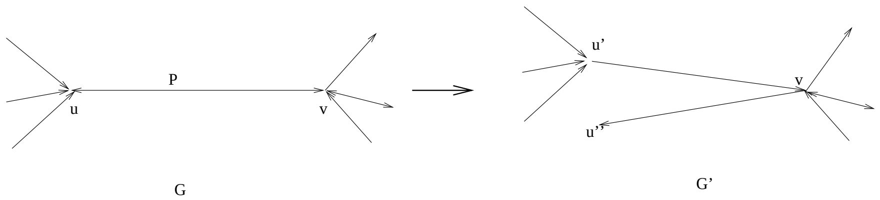

Figure 3: Replacing a bi-directional path by two directed paths

Let this new graph be G'. Note that G' can have paths (e.g., u' to v to u'') for which we may not have an analogue in G — but certainly, one can define a mapping from a path Q in G to a path Q' in G' in the usual manner — because no path Q in G will contain edges of P which go in both directions. Moreover, such a mapping clearly preserves the conflict graph and the load. So, if we can prove that empty set is sufficient for G', then empty set will be sufficient for G also. But G' has directed internal edges and so, by induction we have the result. ■

Again, let G be an arbitrary network. We say that a node of G is a *converging point* if it has at least two incident edges, so that each is oriented into v, or each is oriented out of v. First, we show how to eliminate converging points from G.

Suppose v is a converging point of G, with incident edges $(v, u_1), (v, u_2), \ldots, (v, u_d)$. (The case in which all edges are incoming is strictly analogous.) Define a new graph G_v as follows: G_v is same as G except that we split v into d parts, each joined to one of u_i's. More formally, $G_v = (V_v, E_v)$, where

$$V_v = (V - \{v\}) \cup \{v_{u_1}, v_{u_2}, \ldots, v_{u_d}\},$$

$$E_v = (E_v - \{(v, u_1), \ldots, (v, u_d)\}) \cup \{(v_{u_1}, u_1), \ldots, (v_{u_d}, u_d)\}.$$

We call this operation *splitting* of G at v.

Any path passing through v must have one of its end points on v; so we do not need to do wavelength conversion at v. Hence

Lemma 4.4 *it is a minimal No minimal sufficient set of G contains a converging point. Hence, if v is a converging point, a set S of vertices is a minimal sufficient set for G if and only if it is a minimal sufficient set for G_v.*

Next we show how to eliminate another type of structure in G.

Definition 4.5 *Let P be a bi-directional path with ends u and v. Let the other edges incident at u be called E_u and those incident at v be called E_v. Suppose all edges in E_u and E_v are directed into nodes u and v (respectively), or all are directed out of nodes u and v. Further, suppose there are no edges except those of P incident on any internal vertex of P. Then P is called a* bounded path *of G (see figure below).*

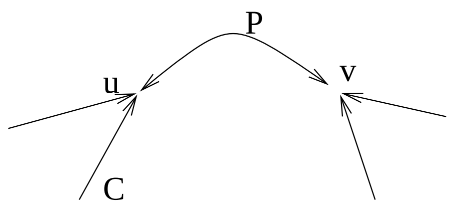

Figure 4: Example of a bounded path (P in the figure)

Lemma 4.6 *Let P be a bounded path in G, and S a minimal sufficient set. Then S does not contain any internal vertex of P.*

Proof. Let S be a set of converters in G, and S' the subset of those that do not lie internally on P. We show how to color any set of paths $\mathcal{P}$ with $\nu(\mathcal{P})$ colors, converting colors only at nodes of S'.

Let P have ends u and v; let P_0 denote the edges of P that are directed from u to v, and P_1 denote the edges of P that are directed from v to u. We consider the following three sets of paths.

- $\mathcal{P}'$, consisting of all paths that do not contain edges of P.
- $\mathcal{Q}_0$, consisting of all paths in $\mathcal{P}$ that use edges from P_0.
- $\mathcal{Q}_1$, consisting of all paths in $\mathcal{P}$ that use edges from P_1.

Using converters only at nodes in S', it is easy to verify that each of these path sets can be colored using a number of colors equal to their congestion. We can now extend this to a coloring of $\mathcal{P}$ using the same number of colors, by a re-labeling of colors: this is possible since $\mathcal{Q}_0 \cap \mathcal{Q}_1 = \phi$, and the only paths common to $\mathcal{P}'$ and $\mathcal{Q}_i$ $(i = 0, 1)$ all share an edge, and hence all receive different colors. ■

Thus we can apply the following transformation to a graph with a bounded path P, preserving the structure of the minimum sufficient set. Suppose P has ends u and v, and the edges at u and v are incoming edges - the other case is exactly similar. Then, we can break the cycle as follows : Split u and v (and every other vertex on P) into two vertices each - u', u'' and v', v''. Replace P by two paths : P' - a directed path from u' to v' and P'' - a directed path from v'' to u'' (see figure below) - let this new graph be called G'. Then it is clear that we can have 1-1 correspondence (in the usual way) between the paths of G and G'. In fact conflict graphs of two corresponding set of paths will also remain the same.

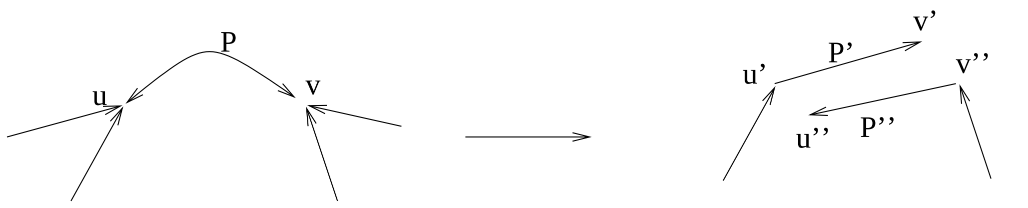

Figure 5: Breaking a bounded path

We will say that a graph G is *robust* if it contains no converging points and no bounded paths. By applying the above two transformations initially, we may assume for the remainder of the section that the input graph G is *robust.*

A *minimal cycle* in an undirected graph is a cycle such that no two vertices on it are joined by an edge other than those on the cycle itself. Clearly, a graph has a cycle if and only if it has minimal cycles. Let C be a minimal cycle in $s(G)$. Define a *meeting point* on C to be a vertex v of C such that the two edges of C incident on v are either both directed towards or directed away from v.

Lemma 4.7 *Let $G = (V, E)$ be robust, and C a minimal cycle of $s(G)$ so that if (u, v) is an edge of C, then $(v, u) \notin E$. Then any sufficient set must contain a vertex of C.*

Proof. Let v be a meeting point of C. Let the edges of C incident on v be incoming edges. Since, v is not a converging point, there is an outgoing edge incident on v. We call this edge the *non-converging* edge of v - we can define this for v analogously if the edges of C incident on v are outgoing edges. Since, C is a minimal cycle, no two meeting points have the same non-converging edge.

Observe that a cycle can have only an even number of meeting points. We prove the result by induction on the number of meeting points.

Induction Hypothesis If C has $2k$ meeting points, then we can find $2k+1$ paths, such that the conflict graph is an odd cycle (implying that we need at least 3 colors), even though the maximum load on an edge is 2. Moreover, we can find this set of paths such that exactly one path passes through a specified edge of C and for every non-converging edge, there are exactly two paths which share this edge.

For k = 1, we show an example below :

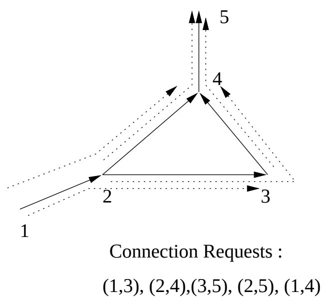

Figure 6: Example of paths whose conflict graph is an odd cycle

So, suppose that the hypothesis is true are all $k < r$. Now, we prove it for $k = r$. Let u and v be two consecutive meeting points (and suppose the specified edge is (v, u). Let the edges of the cycle incident on them be $(u_1, u), (v, u), (v, v_1)$ (see figure below). Let the non-converging edges of u and v be (u, u_2) and (v_2, v). We construct a new

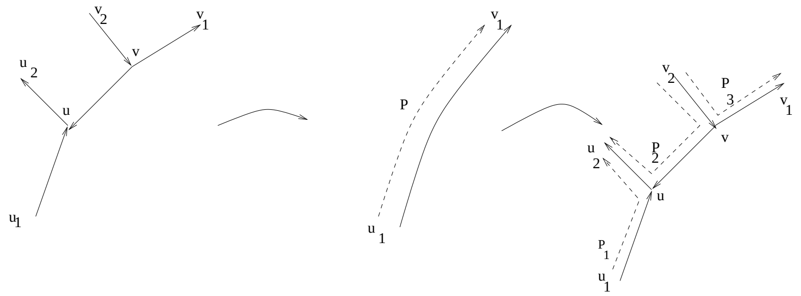

Figure 7: Induction case for the hypothesis

cycle C' by removing u and v (and the edges incident on them) and adding a directed edge from u_1 to v_1. Now, C' has only $2k-2$ meeting points - so by induction hypothesis we can find the specified set of paths such that there is exactly one path, call it P, through (u_1, v_1). Replace P by three paths P_1, P_2, P_3 as shown in the figure. It is clear that this routing satisfies the induction hypothesis. Thus, we are done. ■

Using this, we can prove the more general statement that any minimal cycle in $s(G)$ requires a converter.

Lemma 4.8 *Let $G = (V, E)$ be robust, and C a minimal cycle of $s(G)$. Then any sufficient set must contain a vertex of C.*

Proof. The idea is the following : we convert C into a new cycle C' such that all edges of C' are uni-directional edges. Moreover, C' will be such that it doesn't contain any converging points. Then, by Lemma 4.7 (and the induction hypothesis in the lemma) we can construct a set of paths $\mathcal{P}'$ such that their conflict graph is an odd cycle. Finally we show that $\mathcal{P}'$ can be mapped back to set of paths $\mathcal{P}$ in the original graph G without such that their conflict graph is also an odd cycle. This will prove the lemma.

Consider two vertices u and v in C such that the path P from u to v in C is bi-directional. Moreover, the other edges of C incident at u and v are uni-directional. In case C just consists of bi-directional edges, we have nothing to prove since C contains a directed cycle.

Two cases arise (see figure below) :

1. Let the other edges of C incident on u and v be (u_1, u) and (v, v_1) (the case when it is (u, u_1) and (v_1, v) is similar). Then replace P by a uni-directional path from u to v.
2. The other edges of C incident at u and v are (u_1, u) and (v_1, v) (the other case is similar) : Since P is not a bounded path one of the two cases must occur :
    (a) There exists a vertex w on path P (w can be u or v) such that there is an edge (w, w_1) going out of it. Then replace the bi-directional path from u to w and from v to w by uni-directional paths from u to w and v to w (see figure).
    (b) There exists a vertex w on path P, $w \neq u, v$ such that there is an edge (w_1, w) incident on w. Replace the bi-directional edge from u to w and from v to w by uni-directional edges from u to w and from v to w (as in Case 2(a)) and replace the edge (w_1, w) by the edge (w, w_1), i.e., we reverse the direction of this edge. Note that there may be more edges incident on P - but we don't care about them. So, we delete other edges incident on P.

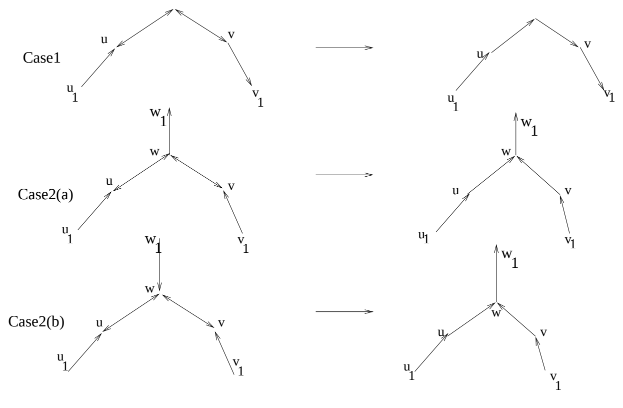

Figure 8: Constructing C' from C - various cases

We can apply this transformation to every such bi-directional path in C until we get a cycle C' which only has uni-directional links. Let the new graph obtained be G'. Note that there are no converging points in C' (because by applying any of the three transformations, no new converging points are introduced, i.e., the vertex w above is not a converging point in C').

So, by the induction hypothesis in Lemma 4.7, we can find a set of paths $\mathcal{P}'$ in C' such that their conflict graph is an odd cycle and there are exactly two paths sharing a non-converging edge. Now, we show how to map these paths back to C. Any path P' in $\mathcal{P}'$ which doesn't use the non-converging edges introduced in Case 2(b) (i.e., edges of the type (w, w_1) in Figure 8) can be mapped directly to G because they pass through edges which are already present in G.

Now, we consider the case for paths which pass through non-converging edges introduced in Step 2(b) (so, these edges are not present in G). Let (w, w_1) be an edge of C' introduced by Case 2(b) (so (w_1, w) is an edge in C). Let P_1' and P_2' be the two paths through (w, w_1) in $\mathcal{P}'$. They must look as shown in Figure 9. Modify P_1' to a path P_1 in $\mathcal{P}$ by removing the edge (w, w_1) from P_1' and adding the edge (w, v) to it. Similarly construct P_2 in G which is basically same as P_2' with the edge (w, w_1) replaced by (w, u). Now, add two more paths $P_3 = (w_1, w, v)$ and $P_4 = (w_1, w, u)$ to $\mathcal{P}$. Note that P_3 and P_4 use edges which were not present in G'. Also, the new edges used by P_1 and P_2 are also not in G'. So, it is easy to see that the conflict graph of $\mathcal{P}$ is same as that $\mathcal{P}'$ with the edge (P_1', P_2') replaced by the path (P_1, P_3, P_4, P_2). So, this conflict graph is also an odd cycle. In this way, we can modify all paths of $\mathcal{P}'$ which use edges introduced in Step 2(b) while maintaining the conflict graph structure.

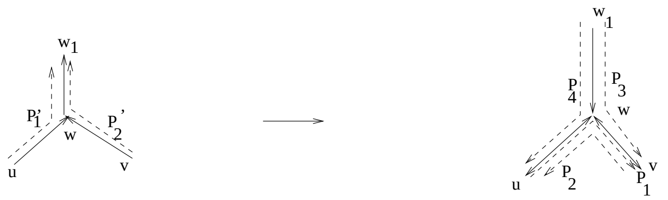

Figure 9: Mapping of paths in $\mathcal{P}'$ which pass through edges introduced by step 2(b) to paths in $\mathcal{P}$

Thus, the lemma is proved. ∎

A non-trivial example is given in the figure below :

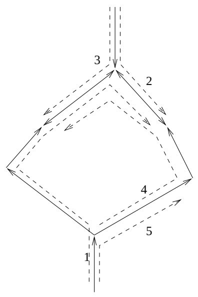

Figure 10: A set of paths needing a converter

Thus we have the following theorem.

Theorem 4.9 *Let G be robust. Then a set of nodes S in G is sufficient if and only if it intersects each minimal cycle of $s(G)$, and the characteristic path of each minimal $\mathcal{H}$-graph.*

Proof. Lemmas 4.1 and 4.8 imply that S must meet each minimal cycle and minimal $\mathcal{H}$-graph. Conversely, suppose that S has this property, and consider $G(S)$. Each component Z of $G(S)$ has the property that it contains no $\mathcal{H}$-graph, and $s(Z)$ is a tree. The result now follows from Lemma 4.3. ■

By Lemma 4.2, the characteristic paths of the minimal $\mathcal{H}$-graphs in G form a collection of edge-disjoint bi-directional paths $\{P_i\}$, each of whose internal vertices have degree 2 in $s(G)$. Given any sufficient set S for G, we can therefore transform it into a sufficient set S' of no greater size that meets each P_i at one of its ends.

Let G' denote the graph obtained from G by replacing each of the P_i by a single bi-directed edge e'_i between its endpoints, and deleting the internal nodes. As a corollary of Theorem 4.9 and the observation of the previous paragraph, our problem is equivalent to finding a set of minimum size that intersects each cycle in $s(G')$ and each of the edges e'_i.

We can phrase this as the following optimization problem.

> Generalized Feedback Set: Given an undirected graph $G = (V, E)$ and a set of edges $I \subseteq E$, find a set $S \subseteq V$ of minimum size such that S intersects all cycles of G and all edges in I.

By the results of this section, it follows that if we can obtain an α-approximation algorithm for the Generalized Feedback Set problem, then we can obtain an approximation algorithm with the same performance guarantee for Minimum Sufficient Set. In the next section, we shall obtain a 2-approximation for Generalized Feedback Set.

It is also not difficult to show that deciding whether an undirected graph has a generalized feedback set of size at most k can be done in time $O(f(k) \cdot p(n))$, where p is a polynomial function; thus a similar result holds for Minimum Sufficient Set in general directed graphs.

Graphs with no bi-directed pairs of edges. If there are no bi-directed pairs in G, i.e., all edges are uni-directional, then $I = \emptyset$ and so Generalized Feedback Set reduces to Feedback Vertex Set in undirected graphs. Moreover, the Sufficient Set problem remains NP-complete even for graphs containing no bi-directed pairs of edges, as we now show.

Proposition 4.10 *Suppose G is a directed graph with no bi-directed pairs of edges. Then deciding whether G has a sufficient set of size k or less is $NP-complete$.*

Proof. We reduce Feedback Vertex Set for undirected graphs to this problem.

So, suppose $G = (V, E), k$ is an instance of Feedback Vertex Set; i.e., do there exist at most k nodes in G which meet all cycles? We shall transform this into a directed graph $G' = (V', E')$ such that G' has a sufficient set of size k if and only if G has a feedback vertex set of size k. The idea is that we assign directions to each edge in E such that we must intercept all cycles of G'.

We prove this by the following Lemma:

Lemma 4.11 *We can orient edges of G in such way that no vertex (except the leaves) is a converging point (recall that a converging point is a vertex such that all edges incident on it are either incoming or outgoing).*

Proof. We prove this by induction on the number of edges of G. It is obvious if G has 1 edge.

So assume the lemma to be true for all graphs having fewer than m edges. Suppose G has m edges. If G has a leaf v, with incident edge (u, v), then we delete v from G. By induction we assign directions to all edges of $G - \{v\}$ so that there are no converging points; we then choose the direction of (u, v) so that u is not a converging point. If G does not have any leaves, let C be a cycle in G. Assign direction to edges in C such that C becomes a directed cycle. Now, remove the edges of C from G and solve recursively. Note that no vertex of C can be a converging point because C is a directed cycle, and the other vertices are not converging points due to the induction hypothesis. Thus, the lemma is proved. ■

So, by Lemma 4.7, a sufficient set must intersect all cycles in $s(G')$, which proves the theorem. ■

5 A Primal-Dual Algorithm

Recall the problem we introduced in the previous section :

> GENERALIZED FEEDBACK SET: Given an undirected graph $G = (V, E)$ and a set of edges $I \subseteq E$, find a set $S \subseteq V$ of minimum size such that S intersects all cycles of G and all edges in I.

We will call such a set S a *generalized feedback vertex set* (gfvs) for this problem instance.

This problem is generalization of both VERTEX COVER and FEEDBACK VERTEX SET, and we know 2-approximations for each of these. Our approximation algorithm in this section will be based on a primal-dual approach used in [4] for FEEDBACK VERTEX SET.

We make extensive use of the linear programming formulation for feedback vertex set described in [4], together with some of its analysis from that work. Because the overall approach works for the weighted version of the problem as well, we work at this greater level of generality.

For a subset $S \subseteq V$, define $G[S]$ as the subgraph induced by S. Let $G[S] = (S, E[S])$. Define $b(S) = |E[S]| - |S| + 1$. Let $d_S(v)$ denote the degree of vertex v in $G[S]$. $\delta(v)$ denotes the edges incident on v in G.

From the integer programming formulation for feedback vertex cover in [4], we obtain the following integer programming formulation for the generalized version :

$$\text{Min} \sum_{v \in V} w_v x_v$$

subject to

$$\begin{array}{ll} \sum_{v \in S}(d_S(v) - 1)x_v \geq b(S) & S \subseteq V, E[S] \neq \emptyset \\ x_u + x_v \geq 1 & (u, v) \in I \\ x_v \in \{0, 1\} & v \in V \end{array}$$

where I is the set of edges to be intersected.

The linear programming relaxation is :

$$(LP) \quad \begin{array}{lll} & \text{Min} \sum_{v \in V} w_v x_v & \\ \text{subject to} & & \\ & \sum_{v \in S}(d_s(v)-1)x_v \geq b(S) & S \subseteq V : E[S] \neq \emptyset \\ & x_u + x_v \geq 1 & (u,v) \in I \\ & x_v \geq 0 & v \in V \end{array}$$

The dual of the above linear program is :

$$(D) \quad \begin{array}{lll} & \text{Max} \sum_S b(S) y_s + \sum_{e \in I} z_e & \\ \text{subject to} & & \\ & \sum_{S: v \in S}(d_s(v)-1) y_s + \sum_{e \in \delta(v), e \in I} z_e \leq w_v & v \in V \\ & y_S, z_e \geq 0 & S \subset V : E[S] \neq \emptyset, e \in I \end{array}$$

We use a basic primal-dual approach to solve this.

$y \leftarrow 0$
$F \leftarrow \emptyset$
$l \leftarrow 0$
$V' \leftarrow V; E' \leftarrow E; I' \leftarrow I$
While F is not a gfvs for $G = (V, E)$
  $l \leftarrow l + 1$
  Recursively remove degree 1 vertices which are not in I' from V' and E'
  $X \leftarrow$ VIOLATION(V', E', I')
    (X can be either $S \subset V$ or $e \in I$ depending on which violated constraint is returned)
  Increase the dual variable corresponding to X (i.e., y_S or z_e)
   until $\exists v_l \in X : \sum_{T: v_l \in T}(d_T(v) - 1) y_T + \sum_{f: f \in \delta(v) \cap I} z_f = w_{v_l}$
  $F \leftarrow F \cup \{v_l\}$
  Remove v_l from V' and attached edges from E'
  If X was $e \in I$, then remove e from I'
For $j \leftarrow l$ downto 1
  if $F - \{v_j\}$ is an gfvs then $F \leftarrow F - \{v_j\}$
$F' \leftarrow F$
Output F'

The structure of this loop is essentially the same as in [4]. The significant difference is in the definition of the VIOLATION routine:

VIOLATION(V', E', I')
If $I' \neq \emptyset$
  return any element $e \in I'$
Else
   If (V', E') contains a semi-disjoint cycle C
   (a semidisjoint cycle contains at most 1 vertex of degree > 2)
    return the vertex set of C
   Else
    Return V'

Note that the VIOLATION routine runs in two stages. In Stage 1, I' is not empty, so it always returns a violated edge. Stage 2 begins when all edges of I have been intersected; during Stage 2, VIOLATION behaves as in [4].

Before we prove that this procedure is a 2-approximation, we mention the following lemma from [4].

Lemma 5.1 (Chudak et al.) *Let F denote a minimal feedback vertex set of a graph $G = (V, E)$. If every vertex in G is has degree at least 2 and G contains no semidisjoint cycles, then :*

$$\sum_{v \in F} (d(v) - 1) \leq 2b(V) - 1$$

Theorem 5.2 *The algorithm above constructs an gfvs F' such that*

$$\sum_{v \in F'} w_v \leq 2(\sum_S b(S) y_s + \sum_{e \in I} z_e)$$

where y_S and z_e are the feasible dual solution obtained by the algorithm.

Hence, the algorithm gives a 2-approximation for gfvs.

Proof. From Line 9 of the main algorithm, a vertex v is added to F (and hence in F') only if its dual constraint becomes tight. So,

$$\sum_{v \in F'} w_v = \sum_{v \in F'} [\sum_{S: v \in S} (d_S(v) - 1) y_S + \sum_{e \in \delta(v) \cap I} z_e] \tag{1}$$

$$= \sum_S [\sum_{v \in S \cap F'} (d_S(v) - 1) y_S] + \sum_{e \in I} z_e |e \cap F'| \tag{2}$$

Since, $|e \cap F'| \leq 2$, $\sum_{e \in I} z_e |e \cap F'| \leq 2 \sum_{e \in I} z_e$.

Now, observe that in Stage 1 of the algorithm, only z_e is incremented. So, y_S remains 0 in this stage. The vertices in F' can be divided into 2 parts :

- Those which were added to F in stage 1. These variables were used to satisfy constraints involving the edges in I. Suppose v was added in stage 1. So, the dual constraint involving v (see line 9 of main algorithm) became tight - but at this moment all y_S are 0. This means that all those y_S which were in the dual constraint involving v (i.e., all y_S such that $v \in S$) can not be increased now. So, for all variables in F' which were added in Stage 1, $y_S = 0$ if S contains any of these variables.
- Those added in Stage 2. Note that after Stage 1 finishes, in each iteration (V', E') contains only vertices of degree at least 2 (due to Line 7 of main algorithm) - this will be needed when we apply Lemma 5.1.

Consider an S such that $y_S > 0$. By the remarks above, $S \cap F'$ contains only vertices added in Stage 2. We claim that $S \cap F'$ is a minimal fvs for the graph (V', E') where (V', E') denotes that graph for which the VIOLATION routine returned S as the violated set.

Let F_c denote at the set F at this time (i.e., when S is the violated set). When S was chosen as the violated set F_c did not contain any element from $S \cap F'$. In line 13-15 of main algorithm, vertices are removed from F in the reverse order. So, if $F' - F_c$ is not a minimal fvs for (V', E') (note that E' at this time does not contain any element of I and so, minimal gfvs and fvs are same), then we would have removed more elements from F' which is not possible. So, $S \cap F'$ must be a minimal fvs for (V', E').

Now, if S is a semi-disjoint cycle, then each vertex has degree 2. Clearly, $|S \cap F'|$ is a singleton. So, $\sum_{v \in S \cap F'} (d_S(v) - 1) = 1$. Also, $2b(S) - 1 = 2 - 1 = 1$. Thus, the relation

$$\sum_{v \in S \cap F'} (d_S(v) - 1) \leq 2b(S) - 1 \tag{3}$$

holds in this case.

If S is not a semi-disjoint cycle, then by the way $VIOLATION$ is defined, it must be the case that (V', E') does not contain any semidisjoint cycles. Thus, by Lemma 5.1, Equation (3) holds in this case also. Therefore, if $y_S > 0$, then Equation (3) holds.

So, Equation (2) becomes :

$$\sum_{v \in F'} w_v \leq \sum_S (2b(S) - 1) y_S + 2 \sum_{e \in I} z_e \leq 2[\sum_S b(S) y_S + \sum_{e \in I} z_e]$$

Thus, we have the theorem. ∎

We note that it is also possible to develop a more *ad hoc* 2-approximation algorithm for Generalized Feedback Set, which uses the algorithm of Chudak et al. [4] in a more "black-box" fashion. We have chosen the current approach because it results in a "pure" primal-dual algorithm, and implies an integrality gap of 2 for a natural linear programming relaxation of the problem.

References

[1] Y. Aumann, Y. Rabani, "Improved bounds for all-optical routing," *Proc. 6th ACM-SIAM Symp. on Discrete Algorithms*, 1995, pp. 567–576.

[2] B. Baker, "Approximation algorithms for NP-complete problems on planar graphs," *Proc. 24th IEEE Symp. on Foundations of Computer Science*, 1983.

[3] R. Bar-Yehuda, S. Even, "On approximating a vertex cover for planar graphs," *Proc. 14th ACM Symp. on Theory of Computing*, 1982.

[4] F. Chudak, M. Goemans, D. Hochbaum, D. Williamson, "A primal-dual interpretation of two approximation algorithm for the feedback vertex set problem in undirected graphs," *Operations Research Letters*, to appear.

[5] R. Downey, M. Fellows, "Parametrized Computational Feasibility," in *Feasible Mathematics II*, P. Clote and J. Remmel, eds., Birkhauser, 1994.

[6] P.E. Green, *Fiber-Optic Communication Networks*, Prentice-Hall, 1993.

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

[8] V. Kumar, E. Schwabe, "Improved access to optical bandwidth," *Proc. 8th ACM-SIAM Symp. on Discrete Algorithms*, 1997.

[9] M. Mihail, C. Kaklamanis, S. Rao, "Efficient access to optical bandwidth," *Proc. 36th IEEE Symp. on Foundations of Computer Science*, 1995.

[10] C. Partridge, *Gigabit Networking*, Addison Wesley, 1993.

[11] Y. Rabani, "Path-coloring on the mesh," *Proc. 37th IEEE Symp. on Foundations of Computer Science*, 1996.

[12] P. Raghavan, E. Upfal, "Efficient all-optical routing," *Proc. 26th ACM Symp. on Theory of Computing*, 1994, pp. 134–143.

[13] G. Wilfong, P. Winkler, "Ring routing and wavelength translation," *Proc. 9th ACM-SIAM Symp. on Discrete Algorithms*, 1998.