Thomas Erlebach, Stamatis Stefanakos

Wavelength Conversion
in Networks of Bounded Treewidth

TIK-Report
Nr. 132, April 2002
Thomas Erlebach, Stamatis Stefanakos
Wavelength Conversion in Networks of Bounded Treewidth
April 2002
Version 1
TIK-Report Nr. 132

Computer Engineering and Networks Laboratory,
Swiss Federal Institute of Technology (ETH) Zurich

Institut für Technische Informatik und Kommunikationsnetze,
Eidgenössische Technische Hochschule Zürich

Gloriastrasse 35, ETH Zentrum, CH-8092 Zürich, Switzerland
Abstract

In all-optical networks where the technique of wavelength division multiplexing is employed, a connection between two nodes is established by first choosing a route connecting these two nodes and then assigning a wavelength to that route. Many connections can share the same physical link, provided that they use different wavelengths. One technology usually employed in order to make more efficient use of the available bandwidth is that of wavelength converters. A converter is placed in some node of the network and has the ability to alter the transmitting wavelength of any incoming signal. In this paper, we study the problem of placing as few converters as possible in a given network so that the number of necessary wavelengths for any routing is equal to the congestion of the network. This problem is known to be \textit{NP}-hard even for bidirected graphs. We give a linear time algorithm for directed graphs of bounded treewidth and an FPT algorithm for arbitrary directed graphs. For undirected graphs we show that the problem is solvable optimally in linear time.

1 Introduction

In all-optical networks where the technique of wavelength division multiplexing is employed, a connection between two nodes is established by first choosing a route connecting these two nodes and then assigning a wavelength to that route. Many connections can share the same physical link, provided that they use different wavelengths. Since the number of available wavelengths is a limited resource, many interesting algorithmic problems arise in the design and operation of such networks. One such problem, usually referred to as the \textit{wavelength assignment problem}, is to assign a wavelength to each route in a routing (i.e. a collection of routes) so that routes that share a common link get different wavelengths and the number of used wavelengths is minimized. A survey of known results on routing problems in all-optical networks can be found in [GV00].

One technology usually employed in order to make more efficient use of the available bandwidth in all-optical networks is that of \textit{wavelength converters} (see e.g. [KA96]). A wavelength converter (also referred to as wavelength translator) is a device that is placed in a node of the network and can change the transmitting wavelength of any incoming signal. Clearly, the use of converters can decrease the number of necessary wavelengths in an optical network. For example, any routing on a network that utilizes a converter at every node will not require more wavelengths than those already needed for establishing the connections on the most congested link. However, such improvident use of wavelength converters is to be avoided due to their high cost. As a consequence, we are interested in minimizing the number of required converters and placing them in suitable positions on the network such that the number of necessary wavelengths for any routing is minimum.

A communication network can be modeled by a graph $G = (V, E)$ where the vertices correspond to the nodes of the network and edges correspond to physical links between nodes. In what follows we will distinguish between undirected, directed and bidirected graphs depending on the type of physical links used in the actual communication network: if the physical links are bidirectional then we model it by undirected graphs, while if they are unidirectional we model it by directed or bidirected graphs. A directed graph on edge set $E$ is \textit{bidirected} if $(u, v) \in E$ implies $(v, u) \in E$.

Regarding wavelengths as colors, the wavelength assignment problem can be formulated in graph-theoretic terms as a path-coloring problem: when no converters are used it simply asks for
an assignment of colors to paths such that two paths that share a common edge get different colors and the number of colors is minimized. In the presence of converters, however, a wavelength assignment has to assign a color to each edge of a path:

**Wavelength Assignment.** Given a (possibly directed) graph \( G = (V, E) \), a set of paths \( \mathcal{P} \) on \( G \) and a set of vertices \( S \subseteq V \) containing wavelength converters, assign a color to each edge of each path such that

1. no edge gets the same color for two different paths that go over it and
2. for every node \( v \notin S \) every path containing edges \( \{u, v\}, \{v, w\} \) (in the directed case, edges \( (u, v), (v, w) \)) has the same color assigned to both edges.

The goal is to minimize the number of colors.

We call an assignment that satisfies the conditions in the definition above **valid** and denote the minimum number of different colors that are needed for a valid assignment by \( \chi(\mathcal{P}) \). The congestion of an edge \( e \) for a given routing \( \mathcal{P} \) is the number of paths in \( \mathcal{P} \) that go over \( e \) and is denoted by \( L(e) \). The congestion of the network \( L(\mathcal{P}) = \max_e L(e) \) is the maximum congestion over all edges of the network. It is easy to see that \( \chi(\mathcal{P}) \geq L(\mathcal{P}) \): one will always need at least as many colors as the maximum number of paths that share a single edge for a valid coloring.

Wavelength conversion was studied by Wilfong and Winkler who introduced the following problem:

**Sufficient Set.** Given a (possibly directed) graph \( G = (V, E) \), find a sufficient set \( S \) for \( G \), i.e.

a set \( S \subseteq V \) such that any routing \( \mathcal{P} \) on \( G \) can be colored with \( L(\mathcal{P}) \) colors if we place wavelength converters on the nodes of \( S \). The goal is to minimize the size of \( S \).

Wilfong and Winkler proved Sufficient Set to be \( \mathcal{NP} \)-hard even for bidirected graphs in [WW98]. Moreover they gave a characterization of sufficient sets for bidirected graphs. Let \( G = (V, E) \) be a bidirected graph; for a set \( S \subseteq V \) let \( G(S) \) be the bidirected graph obtained from \( G \) by “exploding” each node \( s \) of \( S \) into degree-of-\( s \)-many copies, each of which is made adjacent to one of the old neighbors of \( s \). A set \( S \subseteq V \) is sufficient for \( G \) if and only if every component of \( G(S) \) is a spider, i.e. a tree with at most one vertex of degree greater than two. Extending this work, Kleinberg and Kumar [KK01] gave a 2-approximation algorithm for directed graphs and a polynomial time approximation scheme for directed planar graphs using techniques based on the undirected Feedback Vertex Set problem. They also showed that any improvement on the approximation ratio for Sufficient Set on bidirected graphs would lead to a corresponding improvement for Vertex Cover. These well known \( \mathcal{NP} \)-hard problems are formally defined as follows:

**Feedback Vertex Set.** Given a (possibly directed) graph \( G = (V, E) \), find a set of vertices \( S \subseteq V \) such that \( G - S \) contains no (directed) cycles. The goal is to minimize the size of \( S \).

**Vertex Cover.** Given a graph \( G = (V, E) \), find a set of vertices \( S \subseteq V \), such that for every edge \( e \in E \) at least one of its endpoints is in \( S \). The goal is to minimize the size of \( S \).

In this work we investigate the complexity of Sufficient Set when restricted to directed graphs of bounded treewidth. The notion of treewidth was introduced by Robertson and Seymour [RS86] in their work on graph minors. Intuitively, the treewidth of a graph measures how much the graph resembles a tree. Many problems that are computationally intractable for arbitrary graphs...
have been found to be polynomially solvable in graphs of bounded treewidth. This is mainly due to the existence of polynomial time algorithms for computing tree decompositions of small width for such graphs. We give formal definitions in Section 2. We refer the reader to [Klo94] for more details on treewidth.

We also consider **Sufficient Set** from the view of parameterized complexity. Parameterized complexity tries to capture a part of the input or output of an \textbf{NP}-hard problem by a parameter, usually denoted by \( k \). The goal is to restrict the so-called combinatorial explosion that is typical for \textbf{NP}-hard problems to the parameter. This is achieved by a \textit{fixed parameter tractable} (FPT) algorithm, i.e. an algorithm that runs in time \( O(f(k) \cdot \text{poly}(n)) \), where \( f(k) \) is an arbitrary function of the parameter and \( \text{poly}(n) \) is a polynomial in the size of the input, denoted here by \( n \). We refer the reader to [DF99] for more details on the theory of parameterized complexity.

The results presented in this paper are as follows: for directed graphs of bounded treewidth we show that \textbf{Sufficient Set} can be solved optimally in linear time. We also observe that for arbitrary directed graphs there exists an FPT algorithm for the parameterized version of the problem, where the number of converters is taken as the parameter. Finally, we show that \textbf{Sufficient Set} is solvable optimally in linear time when restricted to undirected graphs.

### 1.1 Outline

In the following section we give some definitions and results on treewidth. In Section 3 we give the algorithm for graphs of bounded treewidth. Finally, in Section 4 we make some observations regarding the complexity of \textbf{Sufficient Set} on undirected graphs and its parameterized complexity on arbitrary directed graphs.

### 2 Preliminaries

We give some definitions and results on treewidth that we will need in Section 3. We refer to [Die00] for any standard graph terminology not defined here.
Let $G = (V, E)$ be a (possibly directed) graph. A tree decomposition $TD$ of $G$ is a pair $(T, \mathcal{X})$ where $T = (I, F)$ is a tree and $\mathcal{X} = \{X_i \mid i \in I\}$ is a family of subsets of $V$, one for each node of $T$, such that

- $\bigcup_{i \in I} X_i = V$
- for every edge $\{u, v\} \in E$ (in the directed case, for every edge $(u, v)$), there exists $i \in I$ with $u \in X_i$ and $v \in X_i$, and
- for all $i, j, k \in I$ if $j$ is on the path from $i$ to $k$ in $T$, then $X_i \cap X_k \subseteq X_j$.

The width of a tree decomposition $((I, F), \{X_i \mid i \in I\})$ is defined as $\max_{i \in I} |X_i| - 1$. The treewidth of $G$, denoted by $tw(G)$, is the minimum width over all possible tree decompositions of $G$. An example of a width 2 tree-decomposition of a graph is shown in Fig. 1.

We say that $H$ is a minor of $G$ and denote that by $H \prec G$ if $H$ is isomorphic to a contraction of some subgraph of $G$. The following is a well known result (see e.g. [Bod98]).

**Lemma 1** If $H \prec G$, then $tw(H) \leq tw(G)$.

For each fixed $k$ one can decide in linear time whether a given graph $G$ has treewidth at most $k$, and if so construct a tree decomposition of $G$ of width at most $k$ [Bod96].

Linear-time algorithms can be generated automatically for graphs of bounded treewidth for every problem that can be formulated in monadic second order logic. This general framework was developed by Courcelle [Cou90] and extended by Borie et al. [BPT92] and Arnborg et al. [ALS91].

Monadic second order logic is a language that can describe graph properties using constructions such as quantifications over vertices, edges, sets of vertices etc., membership tests, adjacency tests and logical operations along with extensions that allow e.g. to optimize over the size of a free set variable.

### 3 Networks of bounded treewidth

Throughout this section, unless stated otherwise, $G = (V, E)$ denotes a directed graph. We call a pair of directed edges $(u, v), (v, u)$ a bidirected pair. By $s(G)$ we denote the skeleton of $G$, i.e. the...
The following definitions are from [KK01]. We call a vertex \( v \) of \( G \) a \textbf{converging point} if it has at least two incident edges and all of them are either oriented out of \( v \) or all of them oriented into \( v \). Now consider a bidirected path \( p \) with endpoints \( u, v \). We call \( p \) a \textbf{bounded path} if all edges incident to \( u, v \), besides the edges of \( p \), are directed all out of \( u \) and \( v \) or all towards \( u \) and \( v \) and furthermore all internal vertices of \( p \) have degree 2 in \( s(G) \) (consider Fig. 4). A graph is said to be \textbf{robust} if it contains no converging points and no bounded paths.

It is easy to verify that for every directed cycle \( C \), a set of paths \( \mathcal{P} \) on \( C \) can be constructed with \( L(\mathcal{P}) = 2 \) and \( \chi(\mathcal{P}) = 3 \). Therefore any sufficient set will have to hit every directed cycle. Apart from that, there exists another configuration that has to be hit by any sufficient set. Such a configuration is depicted in Fig. 2. We will call this configuration, following the terminology of [KK01], an \( \mathcal{H} \)-graph. It consists of two vertices \( u, v \) joined by a bidirected path and directed edges \((u_1, u), (u, u_2)\) and \((v_1, v), (v, v_2)\). We refer to the bidirected path of an \( \mathcal{H} \)-graph as its \textbf{characteristic path}. We say that an \( \mathcal{H} \)-graph in \( G \) with characteristic path \( p \) is minimal if all internal vertices of \( p \) have degree two in \( s(G) \).

In summary, our approach is, similarly to the one followed in [KK01], as follows: we transform \( G \) to a robust graph \( G' \) with an \textit{equivalent} sufficient set. In \( G' \) any sufficient set has to hit not only the directed cycles and the \( \mathcal{H} \)-graphs of \( G' \), but all cycles in \( s(G') \). Thus, a sufficient set for \( G' \) can be found by solving a problem similar to \textsc{Feedback Vertex Set} in undirected graphs. The algorithm for graphs of bounded treewidth is obtained by showing that the series of transformations applied on \( G \) do not increase its treewidth and that the generalized feedback vertex set problem in undirected graphs can be expressed in monadic second order logic.

First we show how to transform \( G \) to a robust graph \( G' \).

**Removing converging points.** Let \( v \) be a converging point of degree \( d \) in \( G \) with all its incident edges directed out of \( v \) (the case where all edges are directed into \( v \) is similar). Let \( u_1, \ldots, u_d \) be the neighbors of \( v \). We remove \( v \) from \( G \) and add \( d \) new vertices \( v_1, \ldots, v_d \) along with the edges \((v_i, u_i)\) for \( 1 \leq i \leq d \) (consider Fig. 3).

**Removing bounded paths.** Let \( p \) be a bounded path in \( G \) of length \( l \) with endpoints \( u, v \) and all edges incident to \( u \) and \( v \) directed out of \( u \) and \( v \) respectively (the other case is similar). We remove all vertices of \( p \) except \( u \) and \( v \) and add two new vertices \( u', v' \). We then connect \( u' \) to \( u \) and \( v' \) to...
The following two lemmas are proven in [KK01]:

**Lemma 2** Let $S$ be a minimal sufficient set for $G$. $S$ does not contain any converging point of $G$.

**Lemma 3** Let $p$ be a bounded path in $G$, and let $S$ be a minimal sufficient set for $G$. $S$ does not contain any internal node of $p$.

**Corollary 4** A set of vertices $S$ is a minimal sufficient set for $G$ if and only if it is a minimal sufficient set for $G'$.

The following theorem, also from [KK01], indicates that Sufficient Set in robust directed graphs is closely related to Feedback Vertex Set in undirected graphs.

**Theorem 5** Let $G$ be robust. Then a set of nodes $S$ is sufficient for $G$ if and only if it intersects each induced cycle of $s(G)$ and the characteristic path of each minimal $H$-graph.

Consider the following problem:

**Generalized Feedback Vertex Set.** Given an undirected graph $G = (V, E)$ and a set of edges $I \subseteq E$, find a set of vertices $S \subseteq V$ such that $S$ intersects every cycle of $G$ and every edge in $I$. The goal is to minimize the size of $S$.

Now, transform $s(G')$ by contracting the characteristic path of every minimal $H$-graph $K$ to a single edge $e_K$. Let $H$ be the graph obtained this way and

$$I = \{e_K \mid K \text{ is a minimal } H\text{-graph in } G'\}$$

be the set of edges obtained from contracting the characteristic path of all minimal $H$-graphs. It is clear that a solution to Generalized Feedback Vertex Set for $H$ and $I$ is a solution to Sufficient Set for $G$.

**Lemma 6** $tw(H) \leq tw(s(G'))$

**Proof.** $H$ is obtained from $s(G')$ by a series of contractions and hence $H \prec s(G')$. The lemma follows from Lemma 1.

Next, we proceed to show that $tw(s(G')) \leq tw(s(G))$. We can assume that $G$ contains at least one edge since otherwise no path can be defined on $G$. We say that a vertex is pendant if its degree is equal to one. We will need the following lemma:
Lemma 7 Let $G = (V, E)$ be an undirected graph containing at least one edge and let $G_v = (V', E')$ be a graph obtained from $G$ by adding a single pendant vertex $v$. Then, $\tw(G_v) \leq \tw(G)$.

Proof. Let $TD = (T, X)$ be a tree decomposition of $G$ with $T = \{\{1, \ldots, n\}, F\}$ and $X = \{X_i \mid 1 \leq i \leq n\}$. Let $u$ be the unique neighbor of $v$ in $G_v$ and $j$ be such that $u \in X_j$. Now, consider the tree decomposition $TD' = (T', X')$ with

$$T' = \{(1, \ldots, n, n + 1), F \cup \{j, n + 1\}\}$$

and $X' = X \cup \{X_{n+1}\}$, where $X_{n+1} = \{u, v\}$. We claim that this is a valid tree decomposition for $G_v$. It is clear that $\cup_{1 \leq i \leq n+1} X_i = V \cup \{v\}$ and that for every edge $\{x, y\} \in E'$ there exists some $i, 1 \leq i \leq n + 1$ with $x \in X_i$ and $y \in X_i$. Now, for all $i, l, k \in \{1, \ldots, n\}$ if $l$ is on the path from $i$ to $k$ in $T'$ then $X_i \cap X_k \subseteq X_l$ since $TD$ is a proper tree decomposition for $G$. It remains to show that for all $i, l \in \{1, \ldots, n\}$, if $l$ is on the path from $i$ to $n + 1$, then $X_i \cap X_{n+1} \subseteq X_l$. Notice that if $l$ is on the path from $i$ to $n + 1$ then it is on the path from $i$ to $j$ and therefore $X_i \cap X_j \subseteq X_l$. Also $X_l \cap X_{n+1}$ is either empty or contains only $u$. In the latter case we have that $u \in X_i \cap X_j$ since $u \in X_j$ by the definition of $j$. Thus in both cases $X_i \cap X_{n+1} \subseteq X_l$ and we have shown that $TD'$ is a proper tree decomposition for $G_v$. It is easy to see that the width of $TD'$ is equal to the width of $TD$. \qed

Lemma 8 $\tw(s(G')) \leq \tw(s(G))$

Proof. Let $J$ be the graph obtained from $s(G)$ by removing all converging points and one edge from the characteristic path of every bounded path. Since $J$ is a subgraph of $s(G)$ we have that $J \prec s(G)$ and by Lemma 1: $\tw(J) \leq \tw(s(G))$. It is easy to see that we can obtain $s(G')$ from $J$ by a series of additions of pendant vertices. Thus by Lemma 7 we have $\tw(s(G')) \leq \tw(s(G))$. \qed

Theorem 9 There exists a linear time algorithm for Generalized Feedback Vertex Set in undirected graphs of bounded treewidth.

Proof. The problem can be formulated in the extended monadic second order logic developed in [BPT92] as follows:

$$\min_{S \subseteq V} |S| : (\forall e \in I : \exists x \in V : \text{Inc}(x, e) \land x \in S) \land (\forall F \subseteq E : \text{Cycle}(\text{IncV}(F), F) \rightarrow (\exists y \in V : y \in S \land y \in \text{IncV}(F))),$$

where $I$ is the set of edges specified in the given instance. $\text{Inc}(x, e)$ is true if and only if vertex $x$ is incident to edge $e$. The macros used in the expression given above ($\text{Cycle}(V, E)$ and $\text{IncV}(E)$) can also be expressed in this language (expressions for these and other macros are given in [BPT92]). $\text{Cycle}(V, E)$ simply checks whether some set of edges is a cycle and finally $\text{IncV}(E)$ gives the set of vertices incident to the set of edges $E$. \qed

It is easy to see that all transformations applied on $G$ can be done in linear time. Using Lemmas 6 and 8 and Theorem 9 we obtain:

Corollary 10 There exists a linear time algorithm for Sufficient Set in directed graphs of bounded treewidth.
4 Arbitrary networks

In this section we give exact algorithms for SUFFICIENT SET in arbitrary graphs. First we consider the undirected case. We call a vertex \( v \) a branching node, if its degree is greater than two.

If the graph does not contain a branching node then it is either a line or a single cycle. In the first case we do not need to place any converters since the intersection graph of a set of paths on a line is an interval graph (see e.g. [Gol80]). If it is a cycle then any node is a sufficient set. This is proven in [WW98] for bidirected cycles but their proof applies also to the undirected case. Thus, we can assume that the graph contains at least one branching node. In this case it is easy to see that every branching node necessitates a converter since we can use this node to construct a set of paths \( \mathcal{P} \) with \( \chi(\mathcal{P}) > L(\mathcal{P}) \) (consider Fig. 5). Moreover if we “explode” every branching node \( s \) into degree-of-\( s \)-many copies, and make each one of them adjacent to one of the old neighbors of \( s \), every component in the resulting graph is a line. Therefore any routing \( \mathcal{P} \) on the network can then be colored with \( L(\mathcal{P}) \) colors. All branching nodes can be found in linear time; if there are no branching nodes we can check again in linear time whether we are dealing with a line or a cycle. Thus, we obtain the following proposition:

**Proposition 11** There exists a linear time algorithm for SUFFICIENT SET in arbitrary undirected graphs.

In the directed case SUFFICIENT SET is \( \mathcal{NP} \)-hard. It follows directly from the work of Kleinberg and Kumar [KK01] though that SUFFICIENT SET is fixed parameter tractable. Regarding the size of the sufficient set as the parameter we obtain the following parameterized version of SUFFICIENT SET:

\( k \)-SUFFICIENT SET. Given a directed graph \( G = (V, E) \) and an integer \( k \geq 0 \) find a sufficient set \( S \) for \( G \) of size \( k \) or decide that no such sufficient set exists.

Since SUFFICIENT SET is shown in [KK01] to be equivalent to FEEDBACK VERTEX SET in undirected graphs, and since the latter problem is FPT ([DF95], [Bod93]), we obtain an FPT algorithm for \( k \)-SUFFICIENT SET; this was already observed in [KK98].

**Proposition 12** There exists an FPT algorithm for \( k \)-SUFFICIENT SET in arbitrary directed graphs.
References


