Connected Set Cover Problem and Its Applications*

Tian-Ping Shuai[1] and Xiao-Dong Hu[2]

[1] Department of Mathematics,
Beijing University of Post and Telecom., Beijing 100876, China
stpmath@sohu.com

[2] Institute of Applied Mathematics
Chinese Academy of Sciences
P. O. Box 2734, Beijing 100080, China
xdhu@amss.ac.cn

Abstract. We study an extension of the set cover problem, the connected set cover problem, the problem is to find a set cover of minimal size that satisfies some connectivity constraint. We first propose two algorithms that find optimal solutions for two cases, respectively, and then we propose one approximation algorithm for a special case that has the best possible performance ratio. At last we consider how to apply the obtained result to solve a wavelength assignment problem in all optical networks.

Keywords: Set cover, Approximation algorithm, Performance ratio, Wavelength assignment.

1 Introduction

Given a set system $(U, \mathcal{F})$, where U contains n elements and $\mathcal{F}$ is a family of m subsets of U such that every element of U belongs to at least one subset in $\mathcal{F}$, here each subset in $\mathcal{F}$ has a positive weight, a *set cover* C of U is a subfamily of $\mathcal{F}$ such that every element in U is in at least one of the subsets in C. The set cover problem is to find a set cover with the minimal total weight of subsets in the set cover. For this famous NP-hard problem, Johanson [4] proposed a simple greedy algorithm for the unweighted case (or equivalently all weights are the same) with approximation ratio upper bounded by $1 + \ln n$, and later Chvátal [2] generalized their algorithms to the weighted case and proved the same result.

The set cover problem has many applications in practice. For example, Ruan et al [7] studied how to route and allocate wavelengths to a broadcast connection so that the total wavelength conversions required is minimized. Under some conditions they formulate this problem as two closely related set cover problems, the minimum wavelength-covering problem and the minimum vertex-wavelength-covering problem. But some practical problems may have special

* This work was supported in part by the National Natural Science Foundation of China under Grant No. 70221001, 60373012 and 10531070.

configurations and the set cover model may not appropriate for them. In particular, the set cover model proposed by Ruan el al [7] is not applicable to the case of limited wavelength conversions. In this case, connectedness of a set cover appears to be an important requirement.

We therefore in this paper consider a natural extension of the set cover problem, the *connected set cover problem.* Besides the universal set U and a family $\mathcal{F}$ of U, we are also given a graph G with vertex-set $V(G) = \mathcal{F}$ and edge-set $E(G)$ consisting some edges between some pairs of subsets in $\mathcal{F}$. A set cover $C \subseteq \mathcal{F}$ of U is called connected if the induced subgraph $G(C)$ is connected, where $G(C)$ is a subgraph of G that consists of all edges whose two endpoints are both in C. The problem is to find a connected set cover with the minimal number of subsets. It is easy to see that the classic set cover problem is a special case of the connected set cover problem with a completed graph.

In this paper we will first show that the connected set cover problem is NP-hard even if at most one vertex of the given graph has degree greater than two, and it cannot be solved in polynomial-time. We then propose two polynomial-time algorithms for the case where every vertex in the graph has degree at most two. For the case where at most one vertex has degree greater than two, we propose an approximation algorithm with performance ratio at most $1 + \ln n$ that is the best possible. In the end we discuss an application of the connected set cover problem to the wavelength assignment of broadcast connections in the optical networks.

2 Complexity Study

In this section, we study the complexity of the connected set cover problem for some special graphs. We shall see that the difficulty in solving the connected set cover problem not only lies in the structure of $(U, \mathcal{F})$ system but also related to the property of give graph G. Graph G is called a *line graph* if two vertices in $V(G)$ have degree one and all others have degree two. Graph G is called a ring graph if it is connected and every vertex in $V(G)$ has degree two. Graph G is called a *spider graph* if G is a tree and only one vertex has degree greater than two, a spider graph is particularly called a *star graph* if one vertex has degree greater than one while all others have degree one.

Theorem 1. *The connected set cover problem on star graphs is NP-hard.*

Proof. Given an instance of the set cover problem of uniform weight, $(U, \mathcal{F})$, we construct an instance of the connected set cover problem, $(U', \mathcal{F}')$ and a graph G on $\mathcal{F}'$ as follows: the universal set $U' = U \cup \{u_0\}$, where $u_0 \notin U$, the family $\mathcal{F}' = \mathcal{F} \cup \{u_0\}$, and graph G has edge-set $E(G) = \{(u_0, f) \mid f \in \mathcal{F}\}$. Clearly, G is a star graph and every set cover of U' must include subset $\{u_0\}$ of U'. Thus the set cover problem has a set cover C if and only if the connected set cover problem has a set cover $C \cup \{u_0\}$.

Theorem 2. *The connected set cover problem on line or ring graphs can be solved in polynomial time.*

Proof. Every line graph G can be represented by a path $(f_1 f_2 \cdots f_{m-1} f_m)$, where each vertex f_i corresponds to a subset in $\mathcal{F}$, and f_1 and f_m have degree one (they are two ends of the path). Notice in this case that any connected set cover consists of subsets whose corresponding vertices make a subpath $(f_i f_{i+1} \cdots f_{j-1} f_j)$ for some i, j with $i \leq j$. Thus to find the minimum connected set cover we just need to check all $\binom{m}{2}$ possible solutions (some of them may not be set covers) and then choose the minimum one. This method requires time $O(m^3 n)$.

Similarly, for ring graphs we just need to check all $m(m-1)$ possible solutions and then choose the minimum one. This method also requires time $O(m^3 n)$.

3 Efficient Algorithms for Line and Ring Graphs

In the proof of Theorem 2 we have described a simple algorithm for the connected set cover problem for line and ring graphs, respectively, both have time-complexity of $O(m^3 n)$. In this section, we will propose more efficient algorithms for these two special cases.

We first study how to find the minimum connected set cover in line graphs in an efficient way. The basic idea is to delete as many vertices as possible until the remaining vertices cannot constitute a set cover. This can be carried out as follows: (1) Delete the vertices from the leftmost to the right one by one until the remaining vertices can not constitute a set cover, and then delete the vertices from the rightmost to left one by one until the remaining vertices can not constitute a set cover. (2) Do the same operations as in (1) but in the reverse order, that is, deleting first from the rightmost to left and then from the leftmost to right. (3) Delete the rightmost and then the leftmost vertices alternatively until the remaining vertices can not constitute a set cover. When the process is stopped, if the last vertex is deleted from the left (right) side then repeat delete the vertices from left (right) until the remaining vertices can not constitute a set cover. (4) Choose the best of these three solutions obtained.

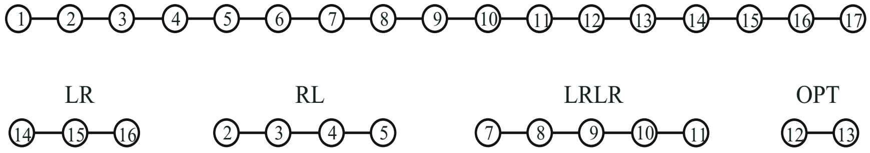

Fig. 1. An counterexample

Denote the above three operations by LR, RL, and LRLR. Unfortunately, the above described method could not find the optimal solution. Fig. 1 gives such an example with $U = \{i \,|\, i = 1, 2, \cdots, 9\}$ and $\mathcal{F} = \{f_j \,|\, j = 1, 2, \cdots, 17\}$, where f_i is defined as follows.

$f_1 = \{4,5,9\}$, $f_2 = \{3,9\}$, $f_3 = \{6,8\}$, $f_4 = \{3,4,7\}$,
$f_5 = \{1,2,5\}$, $f_6 = \{7,9\}$, $f_7 = \{6,7\}$, $f_8 = \{8\}$,
$f_9 = \{2,3\}$, $f_{10} = \{4,5\}$, $f_{11} = \{5,9\}$, $f_{12} = \{1,2,3,4,5\}$,
$f_{13} = \{5,6,7,8,9\}$, $f_{14} = \{1,2,3\}$, $f_{15} = \{4,5,6\}$, $f_{16} = \{7,8,9\}$,
$f_{17} = \{1,4,7\}$.

It can be verified that LR, RL,and LRLR produce three different solutions, but neither of them is optimal. Observe that the optimal solution $\{f_{12}, f_{13}\}$ is on the right half side of the line graph and does not contain the central vertex f_9. However, we shall see that if the optimal solution contain the central vertex, then the above method can be modified to find the optimal solution.

The above example and analysis suggest that we should first find such an optimal solution that contains the central vertex, and then find those two optimal solutions that belong to the left and right half sides of line graph, respectively. In the end we just choose the best solution among these three solutions.

To implement this method, we can modify operations LR and RL as follows: deleting vertices from left to right, or from right to left is stopped until the remaining vertices can not constitute a set cover or the central vertex is reached, and modify LRLR as follows:adding neighbor vertex (and its adjacent edge) of left endnodes of current path and deleting vertices from rightmost to left one by one until the remaining vertices can not constitute a set cover. We use RR represent the operation deleting vertices from rightmost to left is stopped until the remaining vertices can not constitute a set cover. The algorithm is described below as a recursive procedure, where initially $i = 1$ and $j = m$.

Algorithm A. Finding an Optimal Set Cover in Line Graphs

```
procedure LineCover($i, j$):
  if $j - i \leq 2$ then return $f_i$ or $f_j$ if one of them covers $U$
  else find a cover $F_1 = \{f_{l_1}, f_{l_1+1}, \cdots, f_{r_1}\}$ applying modified LR on path
         between $f_i$ and $f_j$;
       find a cover $F_2 = \{f_{l_2}, f_{l_2+1}, \cdots, f_{r_2}\}$ applying modified RL on path
         between $f_i$ and $f_j$;
       find a cover $F_3 = \{f_{l_3}, f_{l_3+1}, \cdots, f_{r_3}\}$ applying procedure
         Modified LRLR on path between $f_{l_1}$ and $f_{r_1}$ and path
         between $f_{l_2}$ and $f_{r_2}$;
       return the best among $\{F_1, F_2, F_3, \text{LineCover}(i, \frac{j+i}{2}), \text{LineCover}(\frac{j+i}{2}, j)\}$.
```

```
procedure Modified LRLR
Input: $F_1 = \{f_{l_1}, f_{l_1+1}, \cdots, f_{r_1}\}$ and $F_2 = \{f_{l_2}, f_{l_2+1}, \cdots, f_{r_2}\}$
  if $l_1 - l_2 \leq 1$ or $r_1 - r_2 \leq 1$ then return the best among $\{F_1, F_2\}$
  else for $j = 1, 2, \cdots, l_1 - l_2 - 1$ do
         find a cover $F_j = \{f_{l_1-j}, \cdots, f_{r'_j}\}$ by applying RR on path
         between $f_{l_1-j}$ and $f_{r'_{j-1}}$.
       return the best among $\{F_j | j = 1, 2, \cdots, l_1 - l_2 - 1\}$.
```

Theorem 3. *Given an instance of the connected set cover problem, $(U, \mathcal{F})$ and a line graph G on $\mathcal{F}$,* **Algorithm A** *finds an optimal solution to the problem in $O(nm^2)$ time.*

Proof. Notice that if the optimal solution does not include the central vertex in the line graph, then it must be in either the left half or the right half of the line graph. Thus to prove that the algorithm returns an optimal solution, it suffices to show that if there exists an optimal solution $F^* = \{f_l, f_{l+1}, \cdots, f_{r-1}, f_r\}$ that contains the central vertex, then it can be found by one of the three operations.

Let us denote the solutions obtained by applying modified operations LR, RL, and LRLR with $i = 1$ and $j = m$, by F_i for $i = 1, 2, 3$, respectively. By the rules of operations LR, RL, and LRLR, we have $l_2 \le l_3 \le l_1$ and $r_1 \ge r_3 \ge r_2$. See Fig. 2.

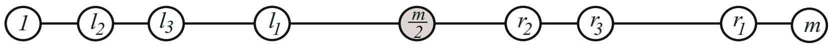

Fig. 2. For the proof of Theorem 3

It is easy to verify that when $l = l_1$, $F^* = F_1$, when $r = r_2$, $F^* = F_2$, and when $l < l_1$ and $r > r_2$, $F^* = F_3$. Thus the solution returned by the algorithm is optimal.

For the time-complexity of the algorithm, notice that in invoking the **procedure** LineCover(i, j) each of the two operations LR, and RL deletes at most $O(j - i)$ vertices and operation LRLR add and deletes total at most $O(j - i)$ vertices, and to check if the remaining vertices make a set cover needs time $O(mn)$. In addition, the **procedure** LineCover(i, j) is invoked at most $\cdot 2^k$ times for subpaths of length $m/2^k$, each time produces 3 solutions; In the total at most $3 \sum_{k=0}^{\log_2 m} 2^k = 3m$ solutions are produced, this requires time bounded by

$$3 \sum_{k=0}^{\log_2 m} 2^k (\frac{m}{2^k})^2 n \le 6m^2 n.$$

To find the best solution among $3m$ ones requires times $O(m)$. Thus the algorithm has the running time at most $O(m^2 n)$.

We now study how to find the minimum connected set cover in ring graphs in an efficient way. For the simplicity of the presentation, we just consider the case of even m. The basic idea is to make the given ring graph into a line graph by removing a vertex, say f_1, and then apply **Algorithm A** to the resulting line graph. Notice however that the minimum connected set cover may include vertex f_1 excluded from the line graph. Thus we need to apply **Algorithm A** to the line graph obtained by removing the vertex $f_{\frac{m}{2}+1}$, which is on the opposite side of vertex f_1. As a result, we find two connected set covers. See Fig. 3.

If the better of these two solutions has size less than $m/2$, then this must be the optimal solution. Otherwise it has size greater than $m/2$. In this case, the optimal solution must contain both vertices f_1 and $f_{\frac{m}{2}+1}$ and includes either all vertices in $\{f_1, f_2, \cdots, f_{\frac{m}{2}+1}\}$ or all vertices in $\{f_{\frac{m}{2}+1}, f_{\frac{m}{2}+2}, \cdots, f_m, f_1\}$. Therefore we

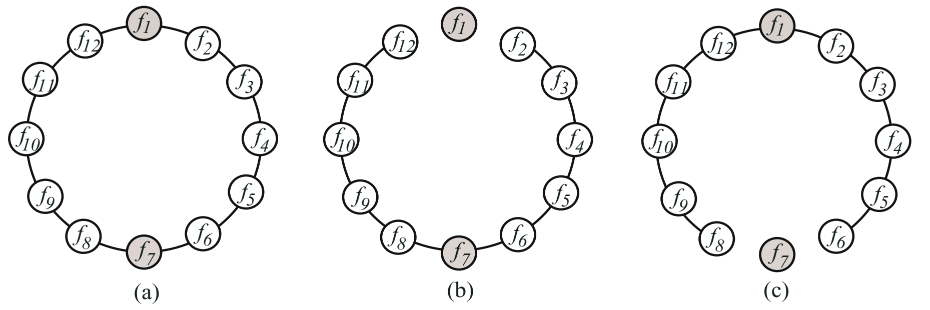

Fig. 3. (a) The original ring graph, (b) and (c) ling graphs obtained by removing two vertices oppositive to each other on the ring

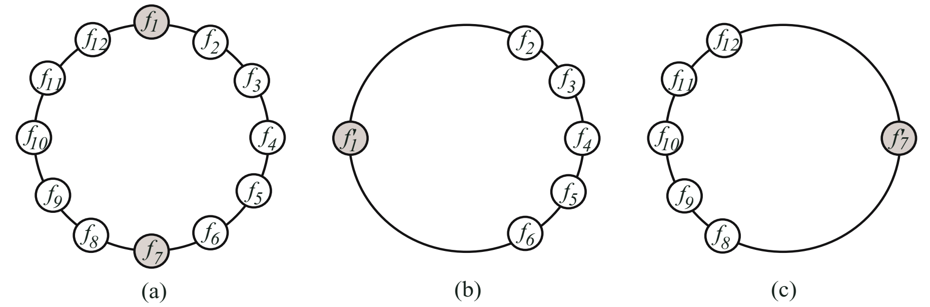

Fig. 4. (a) The original ring graph, (b) and (c) ring graphs obtained by merging half number of vertices

can shrink the ring graph of size m to two ring graphs of size $m/2$ by merging all vertices in $\{f_1, f_2, \cdots, f_{\frac{m}{2}+1}\}$ and $\{f_{\frac{m}{2}+1}, f_{\frac{m}{2}+2}, \cdots, f_m, f_1\}$ into one vertex f_1' and $f_{\frac{m}{2}+1}'$, respectively. See Fig. 4. In the end, we return the best of the four obtained solutions.

The problem is now reduced to two subproblems such that optimal solutions must contain the new vertex f_1' $(f_{\frac{m}{2}+1}')$ in two ring graphs of half size, where the universal set also becomes smaller since those vertices covered by $\{f_1, f_2, \cdots, f_{\frac{m}{2}+1}\}$ and $\{f_{\frac{m}{2}+1}, f_{\frac{m}{2}+2}, \cdots, f_m, f_1\}$ should be removed. This process is repeated until the optimal solution is found. The algorithm is again more formally described as a recursive procedure, where initially $i = 1$, $j = m$ and $U' = U$.

Theorem 4. *Given an instance of the connected set cover problem, $(U, \mathcal{F})$ and a ring graph G on $\mathcal{F}$,* **Algorithm B** *finds an optimal solution to the problem in $O(m^2 n)$ time.*

Proof. The correctness of the proof follows from two facts: (1) If the optimal solution F_{opt} has size less than $m/2$, then it must be either F_1 or F_2 since it must be included in either $\mathcal{F} \setminus \{f_1\}$ or $\mathcal{F} \setminus \{f_{\frac{m}{2}+1}\}$ (maybe in both of them). (2) If F_{opt}

has size greater than $m/2$ and it is not equal to F_1 or F_2, then it must be either $\{f_{\frac{m}{2}}, \cdots, f_1\} \cup$RingCover$(1, m/2, U_L)$ or $\{f_1, \cdots, f_{\frac{m}{2}}\} \cup$RingCover$(m/2, 1, U_R)$.
Algorithm B. Finding an Optimal Set Cover in Ring Graphs

```
procedure RingCover$(1, m, U)$:
  if $m - 1 \leq 2$ then return $f_1$ or $f_2$ or $\{f_1, f_2\}$ if one of them covers $U$.
  else find the optimal cover $F_1$ with $f_1$ removed using Algorithm A,
       find the optimal cover $F_2$ with $f_m$ removed using Algorithm A,
       find the optimal cover $F_3$ with $f_1$ included using
        Procedure RingCover$(i, j, U', f_i)$ on ring $(1, m)$.
          return the best of three covers, $F_1$, $F_2$, $F_3$.
```

```
procedure RingCover$(i, j, U', f_i)$:
  if $j - i \leq 2$ then return $f_i$ or $f_j$ if one of them covers $U$.
  else find the optimal cover $F_1$ with $f_{\frac{i+j}{2}}$ removed using Algorithm A,
       if the cover has size less than $\frac{j-i}{2}$ then return it
       else set $U_L$ be the set consisting of vertices in $U$ not covered by
             $\{f_{\frac{j+i}{2}}, \cdots, f_i\}$, $f_i = \emptyset$
            set $U_R$ be the set consisting of vertices in $U$ not covered by
             $\{f_i, \cdots, f_{\frac{j+i}{2}}\}$, $f_i = \emptyset$.
            return the best of three covers, $F_1$,
             $\{f_{\frac{j+i}{2}}, \cdots, f_i\} \cup$ RingCover$(i, \frac{j+i}{2} - 1, U_L, f_i)$, and
             $\{f_i, \cdots, f_{\frac{j+i}{2}}\} \cup$ RingCover$(\frac{j+i}{2} + 1, i, U_R, f_i)$.
```

For the time-complexity of the algorithm, notice that each time when we invoke the **procedure** RingCover(i, j, U'), we produce F_1 using **Algorithm A**, this requires time $O(n(j-i)^2)$ by Theorem 3. In addition, the **procedure** LineCover(i, j) is invoked at most $\cdot 2^k$ times for subrings of length $m/2^k$, each time produces 2 solutions; In the total at most $2\sum_{k=0}^{\log_2 m} 2^k = 2m$ solutions are produced, this requires time bounded by

$$2\sum_{k=0}^{\log_2 m} 2^k 6(\frac{m}{2^k})^2 n \leq 24m^2 n.$$

To find the best solution among $2m$ ones requires times $O(m)$. Thus the algorithm has the running time at most $O(m^2 n)$.

4 Approximation Algorithm for Spider Graphs

As in the previous section we have proved that the connected set cover problem is NP-hard even for star graphs, thus in this section we will propose an approximation algorithm for the problem in spider graphs. We will show that this algorithm has almost the best possible approximation ratio.

Suppose that G is a spider graph with central vertex $f_0 \in V(G)$ having degree $k > 2$. We now decompose the graph into line subgraphs $L_1, L_2, \cdots, L_k$, which have a common end f_0. Then there are two possible cases for an optimal set cover F^*, (1) it consists of only vertices in one of the line subgraphs, and (2) it includes at least two vertices belonging to different line graphs. For case (1) we can find the optimal solution by running **Algorithm A** k times (just choose the best among k solutions). For case (2) we can first transform the problem into the set cover problem, and then solve the problem approximately by the generalized greedy algorithm [2].

The transformation can be done as follows: For each subset $f \in \mathcal{F}$ with $f \neq f_0$, which corresponds to a vertex in graph G, we define a new subset f' that is the union of the subsets whose corresponding vertices in G are on the path $p(f, f_0)$ between f and f_0 and delete the elements contained in f_0, and define $f'_0 = f_0$. We then construct a new family of subsets $\mathcal{F}' = \{f' \mid f \in \mathcal{F}\}$. We also assign a weight $w(f')$ to f' which is equal to the number of edges in $p(f, f_0)$, $f \neq f_0$, and $w(f'_0) = 1$. The following lemma shows that the new set system $(U, \mathcal{F}')$ has the property that we need for our algorithm.

Lemma 1. *The set system $(U, \mathcal{F})$ with a graph G on $\mathcal{F}$ has a minimum connected set cover C of size $|C|$ that includes subset f_0 if and only if the set system $(U, \mathcal{F}')$ has a minimum weighted set cover C' with weight $w(C') = |C|$.*

Proof. "Only-if": For $i = 1, 2, \cdots, k$, let $\overline{f}_i \in C$ be the subset whose corresponding vertex in L_i has the longest path to f_0. Then C contains every subset whose corresponding vertex is on path $p(\overline{f}_i, f_0)$. Clearly, the size of $C \setminus \{f_0\}$ is equal to the sum of number of edges in path $p(\overline{f}_i, f_0)$ for $i = 1, 2, \cdots, k$. Hence $\overline{f}'_0$ and $\overline{f}'_i$ for $i = 1, 2, \cdots, k$ constitute a set cover of U that has weight $|C|$.

"If": Notice that C' does not contain two subsets f' and g' such that the corresponding vertices f and g are in the same line graph L_i for some i, otherwise either $f' \subset g'$ or $g' \subset f'$, thus one of them is redundant contradicting that C' is a minimum set cover. Let $f'_i \in C$ be the subset which includes $w(f'_i)$ vertices in $L_i \setminus \{f_0\}$, $i = 1, 2, \cdots, k$. Hence the union of the subsets whose corresponding vertices are within $w(f'_i)$ distance from f_0 on line graph L_i, for $i = 1, 2, \cdots, k$, makes a connected set cover C of U with size $|C| = \sum_i w(f'_i)$.

Algorithm C. Finding Connected Set Covers in Spider Graphs

```
Decompose the spider graph into line graphs $L_i$'s.
Find the optimal set cover $F_i$ for each $i$ using **Algorithm A**.
Construct a new set system $(U, \mathcal{F}')$.
Find a set cover $F'_0$ of $U$ using the generalized greedy algorithm.
Produce the corresponding set cover $F_0$ of $(U, \mathcal{F}')$ from $F'_0$.
**Return** the best set cover among $\{F_i \mid i = 0, 1, 2 \cdots\}$.
```

Theorem 5. *Given an instance of the connected set cover problem,* $(U, \mathcal{F})$ *and a spider graph* G *on* $\mathcal{F}$*,* **Algorithm C** *returns a solution in time of* $O(m^2 n)$ *whose size is at most* $\log n$ *times that of the optimal solution.*

Proof. Let F_{opt} be an optimal solution, and F_C be a solution returned by **Algorithm C**. If a line graph L_i contains F_{opt} for some i, then $F_C = F_{opt}$. If not (there are two sets in F_{opt} such that one is in line graph L_i while the other in line graph L_j), $w(F_C) \le w(F'_0) \le \log n w(F_{opt})$, the last inequality comes from Chvátal's result [2].

In fact, we will show that **Algorithm C** is the best possible approximation algorithm for the connected set cover problem in spider graphs. To prove this we need the following lemma due to Feige [3].

Lemma 2. *For any* $0 < \rho < 1$*, there is no approximation algorithms with performance ratio* $\rho \ln n$ *for the set cover problem unless* $NP \subset D_{TIME}(n^{poly \log n})$.

Theorem 6. *For any* $0 < \rho < 1$*, there is no approximation algorithms with performance ratio* $\rho \ln n$ *for the connected set cover problem in spider graphs unless* $NP \subset D_{TIME}(n^{poly \log n})$.

Proof. Suppose, by contradiction argument, that there exists an algorithm $A_{\rho'}$ with approximation performance ratio $\rho' < 1$ for the connected set cover problem in spider graphs. We now design an algorithm A_ρ for the set cover problem using algorithm $A_{\rho'}$ as a subroutine.

Given an instance I of the set cover problem, a set system $(U, \mathcal{F})$, construct an instance I' of the connected set cover problem, a set system $(U', \mathcal{F}')$ and a graph G on $\mathcal{F}'$ as follows: Let $U = \{u_i \,|\, i = 1, 2, \cdots, n\}$, $\mathcal{F}= \{f_i | i = 1, 2, \cdots, m\}$, and take $k = \lceil \rho'/(1-\rho') \rceil$. w.l.o.g suppose that $k < m$. Set $U' = U \cup \{u_0\} \cup W$ and $\mathcal{F}' = \mathcal{F} \cup \{f_{ij} \,|\, i = 1, 2, \cdots, m, \; j = 1, 2, \cdots, k\} \cup \{u_0\}$, and spider graph G has central vertex u_0 and m paths $< u_0 f_{i1} f_{i2} \cdots f_{ik} f_i >$ are attached to u_0 for $i = 1, 2, \cdots, m$, where $W = \{i_1, i_2, \cdots, i_m\}$, $\bigcup_{j=1}^{k} f_{ij} = W, \quad i = 1, \cdots, m$, and for every pair $(i, j) \ne (i', j')$, $f_{ij} \ne f_{i'j'}$. It is easy to see that instance I has a set cover C with $|C| > 1$ if and only if instance I' has a set cover C' with $|C| > 2$ and $C' = C \cup \{f_{ij} \,|\, f_i \in C\} \cup \{u_0\}$. Let C_{opt} and C'_{opt} be optimal solutions to instances I and I', respectively, then $|C'_{opt}| = k|C_{opt}| + 1$ if C_{opt} is not a singleton. We now apply algorithm $A_{\rho'}$ to instance I' and obtain a set cover C' satisfying $|C'| \le \rho'(\ln n)|C'_{opt}|$. Thus we have

$$k|C| + 1 \le \rho'(\ln n)(k|C_{opt}| + 1) = k\rho'(\ln n)|C_{opt}| + \rho' \ln n,$$

from which we deduce

$$|C| < \rho'(\ln n)|C_{opt}| + \frac{\rho'}{k} \ln n \le \rho'(\ln n)(1 + \frac{1}{k})|C_{opt}|.$$

This contradicts Lemma 2 since $\rho'(1 + 1/k) < 1$.

5 Application

In this section we shall study the wavelength assignment problem of broadcast connections in optical networks, which is a special case of the connected set cover problem. An optical network can be modelled as a connected graph $G(V, E, w)$, where V is the vertex-set of graph G representing the set of routing nodes in the network, E is the edge-set of graph G corresponding to optical fiber links between nodes in the network, and $w(e)$ represents the wavelengths available on edge $e \in E$. Fig. 5(a) shows an example of such a network of 10 vertices. Observe that 5 wavelengths $\{w_1, w_2, \cdots, w_5\}$ are used in the network, but only two wavelengths w_2 and w_3 are available on edge between v_4 and v_8.

In multi-hop optical networks where wavelength converters are equipped at routing nodes, a broadcast connection between communication nodes consists of one or more light-trees. A wavelength conversion is required at the joint of two light-trees if they use different wavelengths. In an all optical network, the optical signal is allowed in the optical domain throughout the conversion process, however shifting wavelength channels from one to another makes routing/switching complicated. Thus an incoming wavelength at a routing node is allowed to convert to a subset of available wavelengths [6]; In particularly, it is allowed to be shifted only to neighboring wavelengths. For example, w_3 can only be shifted to w_2 or w_4. Thus it is desirable to minimize the number of wavelength used to reduce the conversion delay and workload of routing nodes.

Our problem here is how to construct a spanning tree T of given graph $G(V, E, w)$ such that the number of wavelengths used is minimized. Fig. 5(b) shows an optimal wavelength assignment for the example given in Fig. 5(a), where four wavelengths are needed and wavelength conversions are required at vertices v_4 and v_8.

Let us see how to formulate this problem as a connected set cover problem. Let U be the vertex-set V, and f_i be the set of vertices in V that are incident to some edges e with $w_i \in w(e)$, that is the set of vertices covered by wavelength w_i. For the example of Fig. 5(a), $f_1 = \{v_2, v_3, v_4, v_5\}$. First we assume that the vertices in f_i induce a connected subgraph of $G(V, E, w)$. The network of Fig. 5(a) satisfies this assumption. Now define a graph G_w on the set system

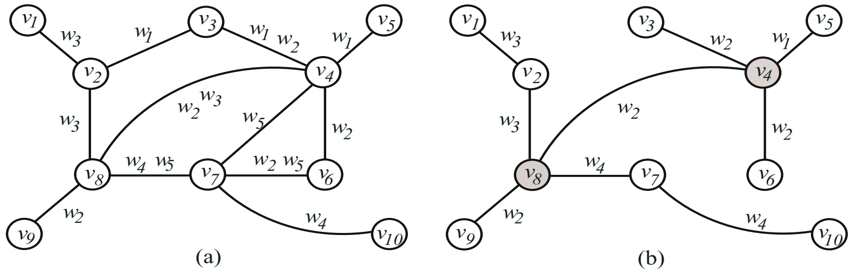

Fig. 5. (a) An optical networks, and (b) an optimal wavelength assignment

$(U, \mathcal{F} = \{f_i\})$ such that there is an edge between f_i and f_{i+1} if and only if $f_i \cap f_{i+1} \neq \emptyset$. Clearly, G_w is a line graph of union of two or more line graphs.

Thus we can use **Algorithm A** to find the minimal number of wavelengths to cover all vertices in V. As a result, we obtain a subgraph of $G(V, E, w)$ with only selected wavelengths on its edges. Notice that the subgraph may not be a tree, so we need to remove some edges. Moreover, there may exist some edge e with $w(e)$ including more than one wavelengths selected, so we must determine which one to use. These two tasks can be carried out as follows: remove edges and the wavelength w_i such that the vertices in the resulting f_i still constitute a connected subgraph of $G(V, E, w)$. Fig. 6 shows the obtained subgraph of example Fig. 5(a). After removing two edges (v_2, v_3), (v_6, v_7), and wavelengths w_2 and w_3 on (v_3, v_4) and (v_4, v_8), respectively, we get the optimal solution as shown in Fig. 5(b).

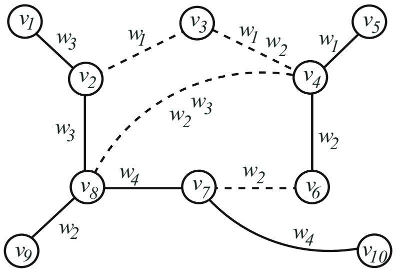

Fig. 6. Establishing a broadcast connection from the obtained subgraph

In the end let us consider the case where the vertices in f_i do not induce a connected subgraph of $G(V, E, w)$ for some i. In this case we can modify the original graph $G(V, E, w)$ as follows: Suppose that the vertices in f_i form k disjoint connected components $C_1, C_2, \cdots, C_k$ for some $k > 1$. We can replace one wavelength w_i by k different dummy wavelengths $w_{i1}, w_{i2}, \cdots, w_{ik}$ in such a way that wavelength w_{ij} is available on all edges in C_j. These k new wavelengths are not introduced physically in the network, they are just wavelength w_i and used only for the simplicity of discussion. After such a modification, we are able to use the above described method.

6 Conclusions

We have studied the connected set cover problem and also discussed its application to the wavelength assignment problem of broadcast connections in all optical networks.

Another possible application comes from the biological conservation [1,5]. The problem concerned is how to establish a series of protected areas or reserves in order to conserve species or habitat types. The objective is to select the minimal number of sites (from some candidate sites) to represent all species in the area. Clearly, this can be modeled as a set cover problem. The obtained solution, however, often produces a highly fragmented network since the solution generally neglect the spatial location of sites. This restricts the possibility of dispersal between sites, which for many species may be essential for long-term persistence. When incorporating considerations of reserve connectivity and the cost, we will get a weighted version of the connected set cover problem, which is more difficulty to solve.

References

1. A. Briers, Incorporating connectivity into reserve selection procedures, *Biological Conservation*, 14(2000) 342-355.
2. V. Chvátal, A greedy heuristic for the set-covering problem, *Mathematics of Operations Research*, 4(1979), 233-235.
3. U. Feige, A threshold of $\ln n$ for aproximating set cover, *Proc. 28th ACM Symposium on Theory of Computing*, (1996), 314-318.
4. D. S. Johnson, Approximation algorithms for combinatorial problems, *Journal of Computer and System Sciences*, 9(1974), 256-278.
5. H. Possingham, I. Ball and S. Andelman, Mathematical methods for representative reserve networks, In: S. Ferson and M.Burgman, eds., *Quantitative Methods for Conservation Biology*, Springer-Verlag, New York, (2000), 291-306.
6. R. Ramaswami and G. Sasaki, Multiwavelength optical networks with limited wavelength conversion, *IEEE/ACM Transactions on Networking*, 6(6)(1998), 744-754.
7. L. Ruan, D.-Z. Du, X.-D. Hu, X.-H. Jia, D.-Y. Li, and Z. Sun, Converter placement supporting broadcast in WDM networks, *IEEE Transactions on Computers*, 50 (7) (2001), 750-758.