Approximations for minimum and min-max vehicle routing problems

Esther M. Arkin[*] Refael Hassin[†] Asaf Levin[‡]

Abstract: We consider a variety of vehicle routing problems. The input to a problem consists of a graph $G = (N, E)$ and edge lengths $l(e)$ $e \in E$. Customers located at the vertices have to be visited by a set of vehicles. Two important parameters are k the number of vehicles, and λ the longest distance traveled by a vehicle. We consider two types of problems: (1) Given a bound λ on the length of each path, find a minimum sized collection of paths that cover all the vertices of the graph, or all the edges from a given subset of edges of the input graph. We also consider a variation where it is desired to cover N by a minimum number of stars of length bounded by λ. (2) Given a number k find a collection of k paths that cover either the vertex set of the graph or a given subset of edges. The goal here is to minimize λ, the maximum travel distance. For all these problems we provide constant ratio approximation algorithms and prove their NP-hardness.

Keywords: Approximation algorithms, Edge-routing, Vehicle routing problem, Min-max problems.

[*]Department of Applied Mathematics and Statistics, SUNY Stony Brook, Stony Brook, NY 11794-3600, estie@ams.sunysb.edu. Partially supported by NSF (CCR-0098172).

[†]Department of Statistics and Operations Research, Tel-Aviv University, Tel-Aviv 69978, Israel, hassin@post.tau.ac.il

[‡]Department of Statistics and Operations Research, Tel-Aviv University, Tel-Aviv 69978, Israel, levinas@post.tau.ac.il

1 Introduction

We consider a variety of vehicle routing problems. The input to a problem consists of a graph $G = (N, E)$ and edge lengths $l(e)$ $e \in E$. In some of the problems we will assume that the edge lengths satisfy the triangle inequality. Customers located at the vertices have to be visited by a set of vehicles. Two important parameters are k the number of vehicles, and λ the longest distance travelled by a vehicle.

We consider two types of problems: (1) Given a bound λ on the length of each walk (path), find a minimum sized collection of walks (paths) that cover all the vertices of the graph, or all the edges from a given subset of edges of the input graph. We also consider a variation where it is desired to cover N by a minimum number of stars of length bounded by λ. (2) Given a number k find a collection of k walks or paths that cover either the vertex set of the graph or a given subset of edges. The goal here is to minimize λ, the maximum travel distance.

All graphs considered in this paper are undirected. A *path* in this paper is a sequence $P = (v_0, e_1, \ldots, e_k, v_k)$ such that $v_i \in V$, $i = 0, \ldots, k$, $e_i \in E$, $i = 1, \ldots, k$ and $e_i = (v_{i-1}, v_i)$ are distinct (vertices may repeat). A *walk* is a path possibly with repeated edges. When we consider the length of a walk, multiple edges are charged multiply. A *star* is a subset of E having a common vertex (i.e., a tree with at most one vertex whose degree is greater than one). The length of a subgraph H with edge set E' is $l(H) = \sum_{e \in E'} l_e$. Throughout this paper ϵ is an arbitrary positive number.

The problems considered in this paper are:

MINIMUM RURAL POSTMEN COVER
Input: A complete graph $G = (N, E)$, a length function $l : E \to I\!R_+$, a subset $E' \subseteq E$, $\lambda > 0$, such that $l(e) \leq \lambda$, $\forall e \in E'$.
Output: Walks $P_1, \ldots, P_k$, $l(P_i) \leq \lambda$, $i = 1, \ldots, k$, such that $E' \subset \cup P_i$.
Measure: k.
Our approximation ratio: 4.

MINIMUM PATH COVER
Input: A complete graph $G = (N, E)$, a metric $l : E \to I\!R_+$, $\lambda > 0$.
Output: Paths $P_1, \ldots, P_k$, $l(P_i) \leq \lambda$, $i = 1, \ldots, k$, such that $N \subseteq \cup P_i$.
Measure: k.
Our approximation ratio: 3.

MINIMUM POSTMEN COVER
Input: A graph $G = (N, E)$, a function $l : E \to I\!R_+$ (not required to be a metric), $\lambda > 0$, such that $l(e) \leq \lambda$, $\forall e \in E$.
Output: Walks $P_1, \ldots, P_k$, $l(P_i) \leq \lambda$ $i = 1, \ldots, k$, such that $E \subseteq \cup P_i$.
Measure: k.
Our approximation ratio: 3.

MINIMUM STAR COVER
Input: A complete graph $G = (N, E)$, a metric $l : E \to I\!R_+$, $\lambda > 0$.
Output: Stars $S_1, \ldots, S_k$, $l(S_i) \leq \lambda$, $i = 1, \ldots, k$, such that $N \subseteq \cup S_i$.
Measure: k.
Our approximation ratio: $(2(3 + \frac{2}{k})(1 + \epsilon) + 1)$.

MINIMUM TREE COVER
Input: A complete graph $G = (N, E)$, a metric $l : E \to I\!R_+$, $\lambda > 0$.
Output: Trees $T_1 = (V_{T_1}, E_{T_1}), \ldots, T_k = (V_{T_k}, E_{T_k})$, $l(T_i) \leq \lambda$, $i = 1, \ldots, k$, such that $N \subseteq \cup V_{T_i}$.
Measure: k.
Our approximation ratio: 3.

MIN-MAX RURAL POSTMEN COVER
Input: A complete graph $G = (N, E)$, a length function $l : E \to I\!R_+$, a subset $E' \subseteq E$, $k > 0$.
Output: Paths $P_1, \ldots, P_k$, such that $E' \subseteq \cup P_i$.

Measure: $\max_i l(P_i)$.
Our approximation ratio: 7.

MIN-MAX PATH COVER
Input: A complete graph $G = (N, E)$, a metric $l : E \rightarrow I\!R_+$, $k > 0$.
Output: Paths $P_1, \dots, P_k$, such that $N \subseteq \cup P_i$.
Measure: $\max_i l(P_i)$.
Our approximation ratio: 4.

MIN-MAX POSTMEN COVER
Input: A graph $G = (N, E)$, a length function $l : E \rightarrow I\!R_+$ (not required to be a metric), $k > 0$
Output: Walks $P_1, \dots, P_k$, such that $E \subseteq \cup P_i$.
Measure: $\max_i l(P_i)$.
Our approximation ratio: 3.

MIN-MAX STAR COVER
Input: A complete graph $G = (N, E)$, a metric $l : E \rightarrow I\!R_+$, and an integer $k \geq 1$.
Output: A set of stars $S_1, \dots, S_k$, such that $N \subseteq \cup S_i$.
Measure: $\max_i l(S_i)$.
Our approximation ratio: Bicriteria $(3 + \epsilon, 3 + \epsilon)$.

MIN-MAX TREE COVER
Input: A complete graph $G = (N, E)$, a metric $l : E \rightarrow I\!R_+$, $k > 0$.
Output: Trees $T_1, \dots, T_k$, such that $N \subseteq \cup T_i$.
Measure: $\max_i l(T_i)$.
Our approximation ratio: 4.

We will also use the following problems:
THE CHINESE POSTMAN PATH PROBLEM
Input: A graph $G = (N, E)$, a function $l : E \rightarrow I\!R_+$
Output: A walk P such that $E \subset P$.
measure: $l(P)$.
The problem is polynomially solvable [7].

THE RURAL POSTMAN PATH PROBLEM
Input: A complete graph $G = (N, E)$, a length function $l : E \rightarrow I\!R_+$, $E' \subseteq E$.
Output: A walk P such that $E' \subset P$.
measure: $l(P)$.
The problem is NP-hard [14] and has a $\frac{3}{2}$-approximation algorithm [10].

Previous work. Several approximation algorithms have been designed for variations of the problems with the assumption that each path should start and end at a designated *depot* vertex.

Li, Simchi-Levi and Desrochers [15] gave an approximation algorithm with bound $1 + \frac{\beta}{\beta - 2}$ where $\beta = \frac{\lambda}{d_m}$, and d_m is the maximum distance between the depot and a customer, for the MIN PATH COVER with a depot. In this instance the paths should start at the depot but may end at any point. Notice that when d_m approaches λ the approximation ratio approaches infinity.

Applegate, Cook, Dash and Rohe [2] solved an instance of the problem from the Whizzkids'96 competition. The problem they solved is an instance of the problem considered in [15], with an additional secondary goal which is minimizing the average latency of all the vertices. They solved to optimality the instance using a branch and bound method.

Fredrickson, Hecht and Kim [9] considered the depot version of MIN-MAX PATH COVER and gave a $\left(\frac{5}{2} - \frac{1}{k}\right)$-approximation algorithm.

Averbakh and Berman [4] considered a variant of MIN-MAX PATH COVER where the paths have to be cycles (i.e., closed paths) and l is a tree metric. They also considered the MIN-MAX POSTMEN COVER problem where the walks in the solution have to be closed walks, E' is the edge set of a tree and l is the metric induced by this tree. For all these variants they gave $2 - \frac{2}{k+1}$-approximation algorithms.

Bellmore and Hong [5] considered the following MIN-SUM PATH COVER variation: given a graph $G = (N, E)$ and m potential salesmen at a depot, a potential salesman i has an activation cost C_i. An edge $e \in E$ has length l_e and the objective is to cover all the vertices of the graph by cycles ("tours") such that each vertex except for the depot is covered by exactly one tour and such that the sum of the tour lengths and the activation costs is minimized. They provided a transformation of this problem to the TRAVELLING SALESMAN PROBLEM (TSP). We note that if the triangle inequality is satisfied then the problem is the TSP (any reasonable solution activates only one salesman).

Even, Garg, Konemann, Ravi and Sinha [8] considered the min-max star cover where it is named UNROOTED k-STARS COVER. They presented a bicriteria $(4, 4)$-approximation algorithm. I.e., a polynomial-time algorithm that outputs a solution which covers N with no more than $4k$ stars, and the cost of the solution is no more than four times the cost of an optimal solution which uses no more than k stars. Their method is based on LP-rounding. We show how to use our algorithm for the MINIMUM STAR COVER problem to obtain a bicriteria $(3 + \epsilon, 3 + \epsilon)$-approximation algorithm for this problem. They showed that MIN-MAX TREE COVER problem is NP-hard, and presented a 4-approximation algorithm for it that runs in weakly-polynomial time and a strongly-polynomial time $(4 + \epsilon)$-approximation algorithm for this problem.

Guttmann-Beck and Hassin [12] considered the variant of MIN-MAX TREE COVER problem where the number of vertices in each tree of the cover is equal to $\frac{n}{k}$. They showed that without the triangle inequality unless $P = NP$ the problem cannot be approximated within a constant factor. Assuming that the length function is a metric, they presented an $O(k)$-approximation.

Notations. When considering the problems: MINIMUM RURAL POSTMEN COVER, MINIMUM PATH COVER, MINIMUM POSTMEN COVER, and MINIMUM STAR COVER we will denote by k^* the value of an optimal solution. When considering the problems: MIN-MAX RURAL POSTMEN COVER, MIN-MAX PATH COVER, and MIN-MAX POSTMEN COVER, we will denote by λ^* the value of an optimal solution.

For a path P, we denote by $N(P)$ the vertex multi-set along P (and if a vertex appears multiple times along P we keep its multiplicity), and by $E(P)$ its edge set. For $E' \subseteq E(P)$ a suffix or prefix edge set of P, we denote by $P \setminus E'$ the path obtained from P by deleting the edges in E' and removing isolated vertices. Similarly, for $N' \subseteq N(P)$ that is either a suffix or prefix vertex set of P we denote by $P \setminus N'$ be the path obtained from P by removing the vertices in N' and their incident edges.

2 Minimum cover problems

In this section we consider the first type of problems, in which we are given a bound λ and we want to find a minimum sized collection of subgraphs that "cover" the graph. Our method guesses the optimal solution value k^*, then we solve (or approximate) the problem of covering the required set of edges or vertices with k^* subgraphs such that the total length is minimized. We use the fact that the optimal solution for our original problem, induces a feasible solution to the new problem with cost at most λk^*. Then, we partition each subgraph into smaller pieces such that each piece has length at most λ. The partitioning process guarantees that each piece has length at least $\alpha\lambda$, and therefore our partitioning will result at most $k^* + \frac{k^*}{\alpha}$ subgraphs.

2.1 Minimum rural postmen cover

We use the following auxiliary problem:
k RURAL POSTMEN PROBLEM
Input: A complete graph $G = (N, E)$, $E' \subseteq E$, a length function $l : E \rightarrow \mathbb{R}_+$, $k > 0$.
Output: Walks $P_1, \ldots, P_k$ such that $E' \subset \cup P_i$.
Measure: $\sum_i l(P_i)$.

The problem is NP-hard even for $k = 1$, as the version where the output is a closed walk is NP-hard, and a standard transformation shows the hardness of the (open) walk problem. We note that w.l.o.g. for any vertex $v \in N$ there is an edge in E' that is incident in v (otherwise, v can be removed without affecting the optimal solution, due to the triangle inequality which we assumed).

Lemma 1 *There exists a 2-approximation algorithm for the k RURAL POSTMEN PROBLEM.*

Proof: Find a minimum cost subgraph G'' of G that includes all the edges of E' and has at most k connected components. Double this subgraph. The resulting multigraph has even vertex degrees, and can be covered by k closed walks. For every such walk, delete one of the edges not in E' to produce an open walk. This is a 2-approximation algorithm due to the following: The optimal solution to the problem has at most k connected components and includes all the edges of E'. Therefore, its length is at least the cost G''. The length of the approximate solution is at most twice the length of G''. It remains to note that G'' can be computed in polynomial time by starting with (V, E') and if (V, E') has more than k connected components, then add edges in increasing order of length if they connect distinct connected components in the current solution, until the resulting graph has exactly k connected components. ■

Remark 2 We cannot use a Christofides-like approximation for the k-RURAL POSTMEN PROBLEM, because a minimum cost perfect matching of the odd vertices in G'' may cost more than half the optimal solution cost. In particular, we cannot argue that the connected components of G'' are exactly the connected component of the optimal solution.

Theorem 3 *Algorithm Rural_Postmen_Min_k (see Figure 1) is a 4-approximation algorithm.*

Proof: To prove feasibility we notice that if there is a feasible solution then every edge in E' has length at most λ. Therefore, the **while** loop terminates. Since the k-rural postmen algorithm returns a feasible solution, Algorithm Rural_Postmen_Min_k also returns a feasible solution.

The algorithm is clearly polynomial-time, and therefore it remains to show the approximation ratio of the algorithm. Let OPT be an optimal solution. We show that $|SOL_{k^*}| \leq 4k^*$.

Since OPT is a feasible solution to the k^* rural postmen problem, the length of the optimal k^* rural postmen is at most $l(OPT) \leq \lambda k^*$. Therefore, the length of the solution returned by the k^* rural postmen approximation is at most $2\lambda k^*$. Moreover, since OPT covers the edges of E', $l(E') \leq k^* \lambda$. Therefore, the sum of $l(E')$ and of the total length of the k^* rural postmen solution is at most $3\lambda k^*$.

Consider a pair of consecutive paths that we add to SOL_{k^*} during the **while** loop. Denote by x the reduction of the length of P_i caused by the first of them. This reduction contains the length of the edge (v^i_{j-1}, v^i_j) if and only if this edge is not in E'. Denote by y the total length of edges from E' in the second path. By the "**if** $(v^i_{j-1}, v^i_j) \notin E'$ " line, we are guaranteed that $x + y \geq \lambda$. This is so because $x < \lambda$ only if $(v^i_{j-1}, v^i_j) \in E'$, and then $x + l(v^i_{j-1}, v^i_j) > \lambda$. In the next iteration of the **while** loop the edge $(v^i_{j-1}, v^i_j) \in E'$ will be in the new path and therefore, $y > \lambda - x$. Note that while partitioning a path, we can use the original path to take care the last sub-path (that the argument above does not take care of it). Therefore, throughout the iterations we add to the initial set of k^* paths at most $3k^*$ new paths. Together, the solution returned by Algorithm Rural_Postmen_Min_k has at most $4k^*$ paths. ■

2.2 Minimum path cover

In this subsection we consider the MINIMUM PATH COVER PROBLEM. This problem is a special case of the MINIMUM RURAL POSTMEN COVER PROBLEM where we want to cover all the vertices of G. We can use Algorithm Minimum_Rural_Postmen_Min_k to obtain a 4-approximation. We present a better approximation for this special case.

```
$Rural\_Postmen\_Min\_k$
  **input**
  $A$ complete graph $G = (N, E)$.
  $A$ length function $l : N \times N \to \mathbb{R}_+$.
  $A$ subset $E' \subseteq E$.
  $A$ real number $\lambda > 0$ such that $l(e) \leq \lambda \ \forall e \in E'$.
  **returns**
  $A$ set of walks, each of length at most $\lambda$ covering $E'$.
  **begin**
  **for** $k = 1, 2, \ldots, |E'|$
     $P_1, P_2, \ldots, P_k :=$ a 2-approximation $k$ rural postmen solution.
     $SOL_k := \emptyset$.
     **for** $i = 1, 2, \ldots, k$
        **while** $l(P_i) > \lambda$
             $(v_1^i, v_2^i, \ldots, v_{|P_i|}^i) := P_i$.
             $j :=$ the index of the first vertex on $P_i$ such that the length
             of the $v_1^i - v_j^i$ sub-walk of $P_i$ is greater than $\lambda$.
             $P_i := P_i \setminus \{(v_1^i, v_2^i), \ldots, (v_{j-2}^i, v_{j-1}^i)\}$.
             $SOL_k := SOL_k \cup \{v_1^i, (v_1^i, v_2^i), \ldots, (v_{j-2}^i, v_{j-1}^i), v_{j-1}^i\}$.
             **if** $(v_{j-1}^i, v_j^i) \notin E'$
               **then**
                    $P_i := P_i \setminus \{(v_{j-1}^i, v_j^i)\}$.
             **end if**
        **end while**
        $SOL_k := SOL_k \cup \{P_i\}$.
     **end for**
  **end for**
  $\bar{k} := argmin|SOL_k|$.
  **return** $SOL_{\bar{k}}$.
  **end** $Rural\_Postmen\_Min\_k$
```

Figure 1: Algorithm *Rural_Postmen_Min_k*

This problem is NP-hard even for $k = 1$, as can be shown by a straightforward reduction from TSP. In fact deciding whether one path is sufficient or at least two paths are needed is NP complete, and thus the best approximation factor we can hope for (unless P=NP) is 2.

Theorem 4 *Algorithm Path_Min_k (see Figure 2) is a polynomial time 3-approximation algorithm.*

Proof: The algorithm runs in polynomial time since the forest F can be computed in polynomial time (simply stop the greedy minimum spanning tree algorithm after it chooses the first $n - k$ edges), and the other steps are clearly polynomial.

The feasibility of the returned solution is trivial, and therefore, it remains to prove the approximation ratio.

The algorithm tries all the possibilities of k and chooses the best one. Consider the iteration in which $k = k^*$.

The optimal solution is composed of k paths each of length at most λ. Therefore, $l(OPT) \leq \lambda k$. This optimal solution has $n - k$ edges, and since F is a minimum cost forest with $n - k$ edges, $l(F) \leq \lambda k$, implying that $\sum_{i=1}^{k} l(TOUR_i) \leq 2\lambda k$.

For every i the following is satisfied:

$$N(i, k) \leq \left\lceil \frac{l(TOUR_i)}{\lambda} \right\rceil \leq \frac{l(TOUR_i)}{\lambda} + 1.$$

```
$Path\_Min\_k$
  input
  $A$ complete graph $G = (N, E)$.
  $A$ metric $l : N \times N \to I\!R_+$.
  $A$ real number $\lambda > 0$.
  returns
  A set of paths, each of length at most $\lambda$, covering $N$.
  begin
  for $k = 1, 2, \ldots, n$
      $F :=$ a minimum weight forest on $G$ with $n - k$ edges.
      $C_1, C_2, \ldots, C_k :=$ the connected components of $F$.
      $SOL_k := \emptyset$.
      for $i = 1, 2, \ldots, k$
          $N(i, k) := 1$.
          $TOUR_i :=$ a tour on the vertices of $C_i$ obtained by doubling $F$ and shortcutting.
          $P_i :=$ a path resulting from $TOUR_i$ by deleting an edge.
          while $l(P_i) > \lambda$
              $(v_1^i, v_2^i, \ldots, v_{|P_i|}^i) := P_i$.
              $j :=$ the index of the first vertex on $P_i$ such that the length of the
              $v_1^i - v_{j+1}^i$ subpath of $P_i$ is greater than $\lambda$.
              $P_i := P_i \setminus \{(v_1^i, v_2^i), (v_2^i, v_3^i), \ldots, (v_{j-1}^i, v_j^i), (v_j^i, v_{j+1}^i)\}$.
              $SOL_k := SOL_k \cup \{v_1^i, (v_1^i, v_2^i), v_2^i, (v_2^i, v_3^i), \ldots, (v_{j-1}^i, v_j^i), v_j^i\}$.
              $N(i, k) := N(i, k) + 1$.
          end while
          $SOL_k := SOL_k \cup P_i$.
      end for
  end for
  $\hat{k} := argmin_k \sum_{i=1}^{k} N(i, k)$.
  return $SOL_{\hat{k}}$.
  end $Path\_Min\_k$
```

Figure 2: Algorithm *Path_Min_k*

Therefore,

$$|SOL_k| = \sum_{i=1}^{k} N(i,k) \leq \frac{1}{\lambda} \sum_{i=1}^{k} l(TOUR_i) + k \leq \frac{2\lambda k}{\lambda} + k = 3k,$$

and the algorithm returns a solution with at most $3k^*$ paths. ■

Remark 5 If we require each vehicle to return to its starting point, i.e., we are searching for a set of tours instead of paths that cover all the vertices, we can obtain a 6-approximation as follows: consider the set of paths returned by Algorithm Path_Min_k, and partition each path into two parts, each of length at most $\frac{\lambda}{2}$. Close the obtained paths to form tours, each of length at most λ. This is a 6-approximation since the size of optimal covering by tours is clearly at least the size of optimal covering by paths.

2.3 Minimum postmen cover

This problem is a special case of the MINIMUM RURAL POSTMEN COVER PROBLEM where we want to cover a subset E of the edge set of a complete graph, and the metric is defined as shortest path length according to a distance function on E. We can use Algorithm Minimum_Rural_Postmen_Min_k to obtain a 4-approximation. We present a better approximation for this special case.

Lemma 6 MINIMUM POSTMEN COVER *is NP-hard.*

Proof: The EDGE PARTITION INTO 3-EDGE PATHS PROBLEM is defined as follows: given a graph $G' = (N, E_{G'})$, is there a partition of $E_{G'}$ into $E_1, E_2, \ldots, E_k$ that each corresponds to a 3-edge path. This problem is NP-complete (Holyer [13]). Given an instance of the EDGE PARTITION INTO 3-EDGE PATHS PROBLEM, we define l as

$$l(u,v) = \begin{cases} \frac{\lambda}{3} & \text{if } (u,v) \in E_{G'} \\ \lambda & \text{otherwise} \end{cases} .$$

Clearly, $E_{G'}$ can be partitioned into 3-edge paths if and only if G can be covered by $\frac{|E_{G'}|}{3} + \binom{N}{2} - |E_{G'}| = \binom{N}{2} - \frac{2|E_{G'}|}{3}$ paths of length at most λ. ■

We use the following auxiliary problem:
k POSTMEN PROBLEM
Input: A graph $G = (N, E)$, $l : E \rightarrow I\!\!R_+$, $k > 0$.
Output: Walks $P_1, \ldots, P_k$ such that $E \subset \cup P_i$.
Measure: $\sum_i l(P_i)$.

Lemma 7 *The k POSTMEN PROBLEM can be solved in polynomial time.*

Proof: The edges of a connected graph can be traversed exactly once using k walks if and only if the number of odd degree vertices is at most $2k$. Therefore, the edges that we need to add to G in order to get a feasible solution to the k-POSTMEN PROBLEM correspond to a minimum cost matching (in the metric closure of G) over the odd degree vertices that contains exactly $\max\{0, \frac{n_o - 2k}{2}\}$ edges (where n_o is the number of odd degree vertices). Such a minimum cost matching can be found in polynomial time. ■

Theorem 8 *Algorithm Postmen_Min_k (see Figure 3) is a 3-approximation algorithm.*

Proof: To prove feasibility we note that if there is a feasible solution then every edge in G has length at most λ. Therefore, every walk we add to the solution has length at most λ. Therefore, Algorithm Postmen_Min_k also returns a feasible solution.

The algorithm clearly runs in polynomial time. It remains to analyze its approximation ratio. Let OPT be an optimal solution. We show that in SOL_{k^*} there are at most $3k^*$ paths.

Since OPT is a feasible solution to the k^* postmen problem, the length of the k^* postmen solution is at most $l(OPT) \leq \lambda k^*$. Consider the sum of walk lengths over the walks with length greater than λ. In the beginning (before the inner **for** loop) this sum is at most λk^*. In each iteration of the **while** loop we add two new walks and we reduce this sum by at least λ. Therefore, throughout all iterations we add at most $2k^*$ paths. With the k^* initial paths the resulting solution has at most $3k^*$ paths. ■

2.4 Minimum star cover

Theorem 9 MINIMUM STAR COVER *is NP-hard.*

Proof: The VERTEX PARTITION INTO 2-EDGE PATHS PROBLEM is defined as follows: given a graph $G = (N, E_G)$, does there exist a partition of N into $N_1, N_2, \ldots, N_k$ each containing 3 vertices such that the induced subgraph of G over N_i contains a path with 2 edges. This problem is NP-complete (see [11] page 76).

Given an instance to the VERTEX PARTITION INTO 2-EDGE PATHS we define a metric

$$l(u,v) = \begin{cases} \frac{\lambda}{2} & \text{if } (u,v) \in E_G \\ \lambda & \text{otherwise} \end{cases} .$$

Clearly, G can be partitioned into paths of length 2 if and only if there is a cover with $\frac{n}{3}$ stars each of length λ. ■

```
Postmen_Min_k
  input
  A graph $G = (N, E)$.
  A length function $l : E \rightarrow \mathbb{R}_+$.
  A real number $\lambda > 0$.
  returns
  A set of walks, each of length at most $\lambda$, covering $E$.
  begin
  for $k = 1, 2, \ldots, |E|$
     $P_1, P_2, \ldots, P_k :=$ an optimal $k$ postmen solution.
     $SOL_k := \emptyset$.
     for $i = 1, 2, \ldots, k$
        while $l(P_i) > \lambda$
             $(v^i_1, v^i_2, \ldots, v^i_{|P_i|}) := P_i$.
             $j :=$ the index of the first vertex on $P_i$ such that the length
             of the $v^i_1 - v^i_j$ sub-walk of $P_i$ is at least $\lambda$.
             $P_i := P_i \setminus \{(v^i_1, v^i_2) \ldots, (v^i_{j-1}, v^i_j)\}$.
             $SOL_k := SOL_k \cup \{v^i_1, (v^i_1, v^i_2), \ldots, (v^i_{j-2}, v^i_{j-1}), v^i_{j-1}\}$.
             $SOL_k := SOL_k \cup \{v^i_{j-1}, (v^i_{j-1}, v^i_j), v^i_j\}$.
        end while
        $SOL_k := SOL_k \cup \{P_i\}$.
     end for
  end for
  $\bar{k} := argmin_k |SOL_k|$.
  return $SOL_{\bar{k}}$.
  end Postmen_Min_k
```

Figure 3: Algorithm *Postmen_Min_k*

We use the following auxiliary problem:

THE MINIMUM METRIC k-MEDIAN PROBLEM
Input: A complete graph $G = (N, E)$, a metric $l : N \times N \rightarrow \mathbb{R}_+$, $k > 0$.
Output: $U \subseteq N$, $|U| \leq k$.
Measure: $\sum_{v \in N} \min_{u \in U} l(v, u)$.
The problem is NP-hard. Charikar and Guha [6] provided a 4-approximation algorithm for this problem. Arya, Garg, Khandekar Pandit, Meyerson and Munagata [3] gave a $(3 + \frac{2}{k})(1 + \epsilon)$-approximation algorithm.

Theorem 10 *Algorithm Star_Min_k (see Figure 4) is a $(2\alpha + 1)$-approximation algorithm, where α is the approximation ratio for the minimum metric k-median problem.*

Proof: Because we sort the nodes according to their distance to their serving facility, $\sum_{j=s+1}^{|S|} l(v_j, r_i) \geq (|S| - s) l(v_s, r_i)$. Therefore, by the triangle inequality, any star that is created in the **while** loop has length which is at most $\sum_{j=s}^{|S|} l(v_j, r_i) + (|S| - s) l(v_s, r_i) \leq 2 \sum_{j=s+1}^{|S|} l(v_j, r_i) \leq \lambda$, and therefore the algorithm returns a feasible solution in polynomial time.

We prove the performance ratio. We show that $|SOL_{k^*}| \leq (2\alpha + 1)k^*$. Denote by OPT an optimal solution.

Since OPT is a feasible solution to the k^*-median problem, the length of the optimal k^*-median is at most $l(OPT)$. Therefore, the solution returned by the k^*-median approximation has length at most $\alpha\lambda k^*$.

Consider the sum of lengths of only the stars with length greater than λ. This sum is initially (before the inner **for** loop) at most $\alpha k^* \lambda$. Every star we add during the **while** loop reduces this sum by at least $\frac{\lambda}{2}$. Therefore, throughout the algorithm we add at most $2\alpha k^*$ stars. Therefore, the total number of stars is at most $(2\alpha + 1)k^*$. ■

```
*Star_Min_k*
  **input**
  A complete graph $G = (N, E)$.
  A metric $l : N \times N \to I\!R_+$.
  A real number $\lambda > 0$.
  **returns**
  A set of stars, each of length at most $\lambda$, covering $N$.
  **begin**
  **for** $k = 1, 2, \ldots, n$
      $r_1, r_2, \ldots, r_k :=$ a solution returned by an $\alpha$-approximation algorithm
      for the $k$-median problem on $G$.
      $SOL_k := \emptyset$.
      **for** $i = 1, 2, \ldots, k$
          $S := \{v_1, v_2, \ldots, v_{|S|}\}$ the set of vertices that are served by $r_i$
          in the $k$-median solution, ordered so that
          $l(v_1, r_i) \le l(v_2, r_i) \le \cdots \le l(v_{|S|}, r_i)$.
          **while** $\sum_{v \in S} l(v, r_i) > \lambda$
              $s :=$ the maximum index such that $\sum_{j=s}^{|S|} l(v_j, r_i) \ge \frac{\lambda}{2}$.
              $SOL_k := SOL_k \cup \{$a star rooted at $v_s$, with a leaf set
              $\{v_{s+1}, v_{s+2}, \ldots, v_{|S|}\}\}$.
              $S := S \setminus \{v_s, v_{s+1}, \ldots, v_{|S|}\}$.
          **end while**
          $SOL_k := SOL_k \cup \{$a star rooted at $r_i$ with a leaf set $S\}$.
      **end for**
  **end for**
  $\bar{k} := argmin|SOL_k|$.
  **return** $SOL_{\bar{k}}$.
  **end** *Star_Min_k*
```

Figure 4: Algorithm *Star_Min_k*

Remark 11 The bound in Theorem 10 is the best possible for Algorithm *Star_Min_k*. We now present a bad example with ratio $2\alpha + 1$. Assume that the approximation algorithm for the k-median returned a solution composed of $k - 1$ singletons and one large star whose total length is $\alpha k \lambda$ and each edge has length $\frac{\lambda}{2} + \epsilon$ (the non-star edges have length greater than λ). Our solution adds $2\alpha k - 2$ new singletons, and returns a solution with $(2\alpha + 1)k - 2$. Therefore, the approximation ratio is $2\alpha + 1$.

Remark 12 Note that Algorithm *Star_Min_k* returns disjoint stars. This is not required by the problem, however, we did not find an improved approximation algorithm for the case where we allowed non-disjoint stars. To emphasize a difficulty arising in the partitioning where we allow non-disjoint stars, note that in the example of the previous remark the partitioning of the large star will result a set of $(2\alpha + 1)k - 1$ stars composed of a single edge (instead of the singletons).

2.5 Minimum tree cover

The minimum tree cover problem is NP-hard by the same proof as in Theorem 9. Moreover, the same algorithm given for the minimum cover with paths problem yields a factor 3-approximation for this problem as well. To see that the approximation ratio still holds note that if the optimal solution has k trees each of length at most λ, then $l(OPT) \le \lambda k$, and the forest F that is the minimum cost forest that has $n - k$ edges, satisfies $l(F) \le \lambda k$. The rest of the proof is the same as the proof of Theorem 4.

We conclude this by the following theorem:

Theorem 13 *The minimum tree cover problem is NP-hard, and Algorithm Path_Min_k is a 3-approximation algorithm for it.*

3 Min-max problems

In this section we consider the second type of problems, in which we are given a number k and we want to find a set of k subgraphs that "cover" the graph. In this type of algorithms we allow the algorithms to return a set of less than k subgraphs, assuming that the remaining subgraphs are empty subgraph.

3.1 Min-max rural postmen cover

The Min-max rural postmen cover problem is NP-hard, as for $k = 1$ it is the rural postman problem.

In this subsection we give a simple algorithm that results in an approximation ratio of 7. Denote by $l_1 < l_2 < \cdots < l_m$ the different lengths of the edges in G. We add an edge $m+1$ with cost M such that $\lambda^* < M$. We assume that $k < |E'|$ otherwise the problem is trivial.

Algorithm Min-Max_k_Rural_Postmen (see Figure 5) receives a guess λ of the optimal solution value λ^*. Given this guess, it either constructs a solution with cost at most 7λ or it returns that the guessed value is too small, i.e., $\lambda^* > \lambda$. Note that λ^* is in the interval $[0, n \cdot l_m]$, and therefore we can apply a binary search to obtain a closed value λ such that the algorithm returns a solution with cost at most 7λ, and for $\lambda - \epsilon$ it returns that $\lambda^* > \lambda - \epsilon$. This will be a $(7+\epsilon)$-approximation algorithm.

Lemma 14 *Algorithm Min-Max_k_Rural_Postmen returns a correct answer.*

Proof: We claim that if $\sum_i K_i > k$, then $\lambda^* > \lambda$. To see this, assume the opposite, that $\lambda^* \le \lambda$. This implies that in an optimal solution, no path connects two vertices that are in different connected components of the subgraph induced by the edges of length at most λ, and so we can solve the problem separately in each connected component. Consider a connected component C_i with K_i^* walks of the optimal solution. Each walk has length at most λ^*, and the walks can be connected into a connected subgraph S_i of C_i using $K_i^* - 1$ edges each of length at most λ, resulting in a subgraph of length at most $2K_i^*\lambda$. Thus $l(S_i) \le 2K_i^*\lambda$, and the cost of an optimal rural postman walk over C_i is at most $2l(S_i) \le 4K_i^*\lambda$. Since the approximation ratio of the rural postman algorithm is $\frac{3}{2}$, $l(P_i) \le 6K_i^*\lambda$. Thus, $K_i = \lceil \frac{l(P_i)}{6\lambda} \rceil \le K_i^*$, implying that $\sum_i K_i \le \sum_i K_i^* \le k$, which is a contradiction! Therefore, if the algorithm returns that $\lambda^* > \lambda$, this is a correct answer.

Assume that for a value λ, $\sum_i K_i \le k$. The length of each edge in a returned walk is at most λ. The length of a walk $v_1^i, (v_1^i, v_2^i), \ldots, (v_{p-1}^i, v_p^i), v_p^i$ is composed of the length of the last edge (v_{p-1}^i, v_p^i) (which is at most λ) and the length of the rest of the walk (which is at most 6λ). Therefore, the maximum length of a walk in the resulting solution is at most 7λ. Therefore, in this case the algorithm also returns a correct answer. ■

Remark 15 The binary search to find the best solution, can be replaced by a strongly polynomial time procedure that removes the extra error of ϵ. First, we separately conduct binary search in each interval $[l_j, l_{j+1})$, for $j = 1, \ldots, m$. For such an interval the partition into connected components is the same for all values of λ in the interval, and therefore the approximated rural postman walk in each connected component is the same for all values of λ in the interval. We note that $\sum_i K_i$ is changed only for values of λ that corresponds to the length of a sub-walk of the approximated rural postman walk between a pair of nodes. Therefore, the number of breaking points is at most $\binom{n}{2}$ and we can compute them in strongly polynomial time.

We conclude the following theorem:

Theorem 16 *There is a 7-approximation algorithm for the* Min-max rural postmen cover problem.

```
Min-Max_k_Rural_Postmen
  input
  A complete graph $G = (N, E)$.
  A length function $l : N \times N \to I\!R_+$.
  A subset $E' \subseteq E$.
  An integer $k$.
  A guess $\lambda$ of $\lambda^*$
  returns
  A set of $k$ walks covering $E'$ each of length at most $7\lambda$, or that $\lambda^* > \lambda$.
  begin
  $\{C_i\}_{i=1}^{r_j}$ := the connected components of the subgraph induced by
  the edges of length at most $\lambda$.
  for $i = 1, 2, \dots, r_j$
      $P_i$ := an approximate solution to the rural postman path problem
      for the induced subgraph of $G$ over $C_i$ with $E' \cap (C_i \times C_i)$.
      $K_i := \lceil \frac{l(P_i)}{6\lambda} \rceil$.
  end for
  if $\sum_i K_i > k$
    then
         return "$\lambda^* > \lambda$."
  end if
  $SOL := \emptyset$.
  for $i = 1, 2, \dots, k$
      while $l(P_i) > 6\lambda$
           $(v_1^i, v_2^i, \dots, v_{|P_i|}^i) := P_i$.
           $p$ := the index of the first vertex on $P_i$ such that the length of
           the $v_1^i - v_p^i$ sub-walk of $P_i$ is at least $6\lambda$.
           $P_i := P_i \setminus \{(v_1^i, v_2^i), \dots, (v_{p-1}^i, v_p^i)\}$.
           $SOL := SOL \cup \{v_1^i, (v_1^i, v_2^i), \dots, (v_{p-1}^i, v_p^i), v_p^i\}$.
      end while
      $SOL := SOL \cup P_i$.
  end for
  return $SOL$
  end Min-Max_k_Rural_Postmen
```

Figure 5: Algorithm *Min-Max_k_Rural_Postmen*

3.2 Min-max path cover

The MIN-MAX PATH COVER PROBLEM. This problem is NP-hard, as for $k = 1$ it is the TSP problem. We can use Algorithm Min-Max_k_Rural_Postmen to obtain a 7-approximation algorithm. However, in the following we present a 4-approximation algorithm. Denote by $l_1 < l_2 < \cdots < l_m$ the different lengths of the edges in graph G. We assume that $k < n$ otherwise the problem is trivial. This implies that $\lambda^* \geq l_1$.

Algorithm Min-Max_k_Paths (see Figure 6) is given a guess λ of λ^*. It returns either that $\lambda^* > \lambda$ or it returns a cover by k paths each of length at most 4λ. As in the previous subsection, for a given $\epsilon > 0$, we can apply a binary search on the interval $[0, n \cdot l_m]$ to find a value λ such that the algorithm returns a solution with cost at most 4λ, and for $\lambda - \epsilon$ it returns that $\lambda^* > \lambda - \epsilon$. This will provide a polynomial-time 4-approximation algorithm. As in the previous sub-section, we can replace the binary search by a strongly-polynomial time procedure by computing a superset of the breaking-points.

Lemma 17 *Algorithm Min-Max_k_Paths returns a correct answer.*

Proof: We claim that if for a given value of λ, $\sum_i K_i > k$, then $\lambda^* > \lambda$. To see this, assume the

```
Min-Max_k_Paths
  input
  A complete graph $G = (N, E)$.
  A metric $l : N \times N \to I\!R_+$.
  An integer $k$.
  An integer guess $\lambda$ of $\lambda^*$
  returns
  A set of $k$ paths covering $N$ each of length at most $4\lambda$ or returns $\lambda^* > \lambda$.
  begin
  $\{C_i\}_{i=1}^{r_j}$ := the connected components of the subgraph induced by
  the edges of length at most $\lambda$.
  for every $i = 1, 2, \ldots, r_j$
            $MST_i$ := a min weight spanning tree on $C_i$.
            $P_i$ := a path on $C_i$ obtained by doubling $MST_i$, shortcutting,
            and removing one of the edges.
            $K_i := \lceil \frac{l(P_i^j)}{4\lambda} \rceil$.
            end for
  if $\sum_i K_i^j \geq k$,
    then
         return $\lambda^* > \lambda$
  end if
  $SOL := \emptyset$.
  for $i = 1, 2, \ldots, r_j$
      while $l(P_i) > 4\lambda$
            $(v_1^i, v_2^i, \ldots, v_{|P_i|}^i) := P_i$.
            $p$ := the index of the first vertex on $P_i$ such that the length of
            the $v_1^i - v_p^i$ sub-path of $P_i$ is at least $4\lambda$.
            $P_i := P_i \setminus \{(v_1^i, v_2^i), \ldots, (v_{p-1}^i, v_p^i)\}$.
            $SOL := SOL \cup \{v_1^i, (v_1^i, v_2^i), \ldots, (v_{p-2}^i, v_{p-1}^i), v_{p-1}^i\}$.
      end while
      $SOL := SOL \cup P_i$.
  end for
  return $SOL$.
  end Min-Max_k_Paths
```

Figure 6: Algorithm_*Min-Max_k_Paths*

opposite, that $\lambda^* \leq \lambda$. This implies that in an optimal solution, no path connects two vertices that are in different connected components of the subgraph induced by the edges of length at most λ, and so we can solve the problem separately in each connected component. Consider a connected component C_i of the optimal solution, and suppose that it has K_i^* paths. Each path has length at most λ^*, and the paths can be connected into a spanning tree of C_i using $K_i^* - 1$ edges each of length at most λ, resulting in a tree of length at most $(2K_i^* - 1)\lambda$. Thus $l(MST_i) \leq 2K_i^*\lambda$, and by the metric assumption $l(P_i) \leq 4K_i^*\lambda$. Thus, $K_i = \left\lceil \frac{l(P_i)}{4\lambda} \right\rceil \leq K_i^*$, implying that $\sum_i K_i \leq \sum_i K_i^* \leq k$, which is a contradiction! Therefore, $\lambda^* > \lambda$ and the algorithm returns a correct answer.

Assume that for a value λ we obtain that $\sum_i K_i \leq k$, then the algorithm returns k paths each of length at most 4λ, and therefore it returns a correct answer. ■

Therefore, we establish the following theorem:

Theorem 18 *There is a 4-approximation algorithm for the* MIN-MAX PATH COVER PROBLEM.

3.3 Min-max tree cover

In this subsection we note that using the algorithm for the min-max path cover problem, we get a 4-approximation algorithm for the min-max tree cover problem as well. To see this note that the proof of Lemma 17 is also correct for this case after the following modification: we consider a connected component C_i of the optimal solution, and suppose that it has K_i^* trees. Each tree has length at most λ^*, and the trees can be connected into a spanning tree of C_i using $K_i^* - 1$ edges each of length at most λ, resulting in a tree of length at most $(2K_i^* - 1)\lambda$. Then, the rest of the proof goes without any further change. We conclude this in the following theorem:

Theorem 19 *There is a 4-approximation algorithm for the* MIN-MAX TREE COVER PROBLEM.

3.4 Min-max postmen cover

```
Min-Max_k_Postmen
  input
  A graph $G = (N, E)$.
  A length function $l : E \to I\!R_+$.
  An integer number $k > 0$.
  returns
  A set of $k$ walks covering $E$.
  begin
  $P :=$ an optimal solution for the Chinese postman walk instance on $G$.
  $A := \frac{l(P)}{k}$.
  $SOL := \emptyset$.
  while $l(P) > A$.
      $(v_1, v_2, \ldots, v_n) := P$.
      $j :=$ the index of the first vertex on $P$ such that the length of the
      $v_1 - v_j$ sub-walk of $P$ is at least $A$.
      $P := P \setminus \{(v_1, v_2) \ldots, (v_{j-1}, v_j)\}$.
      $SOL := SOL \cup \{v_1, (v_1, v_2), \ldots, (v_{j-1}, v_j), v_j\}$.
  end while
  $SOL := SOL \cup \{P\}$.
  return $SOL$.
  end Min-Max_k_Postmen
```

Figure 7: Algorithm *Min-Max_k_Postmen*

Theorem 20 *Algorithm Min-Max_k_Postmen (see Figure 7) is a 3-approximation algorithm.*

Proof: Denote by OPT an optimal solution to the problem. OPT uses k walks and covers E. Therefore, $l(E) \leq k\lambda^*$.

To construct P from G, we double some of the edges in E, therefore, $l(P) \leq 2l(E)$, and $A \leq 2\lambda^*$. Consider an arbitrary walk in SOL. It is composed of the last edge of it and the rest of the walk. The last edge belongs to E and therefore, its length is at most λ^*. The rest of the walk has length at most A. Therefore, the total length of the walk is at most $3\lambda^*$. ■

3.5 Min-max star cover

Let $\epsilon > 0$. In order to obtain a $(3+\epsilon, 3+\epsilon)$-approximation algorithm for the min-max star cover problem, we suggest to apply Algorithm Star_Min_k with $\lambda = 3\lambda^*$ (where λ^* is a $(1+\epsilon)$-approximation of the optimal cost of the min-max star cover problem, the value of λ^* is obtained using a binary search). Then, using the analysis of Algorithm Star_Min_k, we get that the number of stars is at most $(\frac{2}{3}\alpha + 1)k^*$ (this is so because each new star that we create in the **while** loop reduces the total length of the remaining stars by at least $\frac{\lambda}{2} = \frac{3\lambda^*}{2}$, and the result is obtained using the fact that the solution to the k^*-median has total length of at most $\alpha\lambda^* k^*$). Using a $3+\epsilon$-approximation for the k-median problem [3], we obtain the following result.

Theorem 21 *There is a bicriteria $(3+\epsilon, 3+\epsilon)$-approximation algorithm for the min-max star cover problem.*

References

[1] R.K. Ahuja, T.L. Magnanti, and J.B. Orlin, *Network Flows: Theory, Algorithms, and Applications*, Prentice Hall, 1993.

[2] D. Applegate, W. Cook, S. Dash and A. Rohe, "Solution of a min-max vehicle routing problem," manuscript 2001.

[3] V. Arya, N. Garg, R. Khandekar, V. Pandit, A. Meyerson and K. Munagata, "Local search heuristics for k-median and facility location problems," *Proc. STOC 2001*, 21-29.

[4] I. Averbakh and O. Berman, "$(p-1)/(p+1)$-approximate algorithms for p-traveling salesmen problems on a tree with min-max objective," *Discrete Applied Mathematics* **75** (1997), 201-216.

[5] M. Bellmore and S. Hong, "Transformation of multisalesmen problem to the standard traveling salesman problem," *J. of the ACM*, **21**, (1974), 500-504.

[6] M. Charikar and S. Guha, "Improved complexity algorithms for the facility location and the k-median problems," *Proc. FOCS 1999*, 378-388.

[7] J. Edmonds, "The Chinese postman problem," *Operations Research* **13**, (1965), B73-B77.

[8] G. Even, N. Garg, J. Konemann, R. Ravi and A. Sinha, "Covering graphs using trees and stars," In: Proc. APPROX 2003, 24-35, 2003.

[9] G. N. Fredrickson, M. S. Hecht and C. E. Kim, "Approximation algorithms for some routing problems," *SIAM J. on Computing* **7**, (1978) 178-193.

[10] G. N. Frederickson, "Approximation algorithms for some postman problems," *J. of the ACM* **26**, (1979), 538-554.

[11] M.S. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W.H. Freeman, New York, 1979.

[12] N. Guttmann-Beck and R. Hassin, "Approximation algorithms for min-max tree partition," *Journal of Algorithms* **24**, 266-286, 1997.

[13] I. Holyer, "The NP-completeness of some edge-partition problems," *SIAM J. on Computing* **10**, (1981), 713-717.

[14] J. K. Lenstra and A. H. G. Rinnooy-Kan, "On general routing problems," *Networks*, **6**, (1976), 273-280.

[15] C-L Li, D. Simchi-Levi and M. Desrochers "On the distance constrained vehicle routing problem," *Operations Research* **40** No. 4 (1992), 790-799.