Optimization Letters
https://doi.org/10.1007/s11590-024-02108-x

ORIGINAL PAPER

On penalized reload cost path, walk, tour and maximum flow: hardness and approximation

Donatella Granata[1,2]

Received: 6 February 2023 / Accepted: 4 March 2024

Abstract
A meticulous description of a real network with respect to its heterogeneous physical infrastructure and properties is necessary for network design assessment. Quantifying the costs of making these structures work together effectively, and taking into account any hidden charges they may incur, can lead to improve the quality of service and reduce mandatory maintenance requirements, and mitigate the cost associated with finding a valid solution. For these reasons, we devote our attention to a novel approach to produce a more complete representation of the overall costs on the reload cost network. This approach considers both the cost of reloading due to linking structures and their internal charges, which we refer to as the *penalized reload cost*. We investigate the complexity and approximability of finding an optimal path, walk, tour, and maximum flow problems under *penalized reload cost*. All these problems turn out to be NP-complete. We prove that, unless P=NP, even if the reload cost matrix is symmetric and satisfies the triangle inequality, the problem of finding a path, tour, and a maximum flow with a minimum *penalized reload cost* cannot be approximated within any constant $\alpha < 2$, and finding a walk is not approximable within any factor $\beta \leq 3$.

Keywords Reload cost · Approximability · NP-completeness · Penalized reload cost · Network design

1 Introduction

Edge labeled graphs can be utilized to model heterogeneous physical infrastructure and properties in various network design problems [1]. Labels can represent different technologies in telecommunication networks or different carriers in

✉ Donatella Granata
donatella.granata@cnr.it

[1] Istituto per le Applicazioni del Calcolo "Mauro Picone", National Research Council, via dei Taurini 19, 00185 Rome, Italy

[2] Dipartimento di Scienze Matematiche, Fisiche e Informatiche, Plesso di Matematica, University of Parma, Parco Area delle Scienze 53/A, 43124 Parma, Italy

Published online: 03 April 2024

transportation contexts. The concept of *reload cost*, introduced by Wirth and Steffan [2], occurs whenever it is necessary to traversing two consecutive arcs that differ in label. The *reload cost* of a s-t path on an edge label graph, for example, is the sum of the arising reload costs incurred while traversing the path, from the source s to the destination t. This calculation excludes any factor related to what is happening in the remaining substructures, such as any of the connected components. In the presence of huge and heterogeneous networks, dominated by a set of properties, and substructures, where each can represented by a label, the reload costs can become a useful instrument to evaluate the maintenance costs required to put a real network to work. This evaluation goes beyond just assessing that only cost is produced by finding specific structures but also includes hidden costs coming from the remain subparts. For example, in transportation network applications, one may be interested in finding the path with the lowest *reload cost* when traveling from a source storage to a destination storage. The goal is to pay as little as possible for tool and transportation prices while also minimizing the local freight costs associated with delivering goods from destination storages to all consumers based on their demands. This overall cost of this network is referred to as the *penalized reload cost*. To the best of our knowledge, this is a novel specialization of the concept of *reload cost*, and we believe that it can have clear practical relevance in many areas, such as transportation, telecommunication networks, distribution of goods, and energy distribution networks. In this paper, we investigate the computational complexity of problems in finding the optimum path, walk (which allows the repetition of nodes), and maximum flow (of value f) between the source s and destination t, and tour from a specific node s, minimizing the *penalized reload cost* over the graph $\mathcal{G}$. The remainder of this paper is organized as follows. In Sect. 2, we present a brief literature review of the canonical *reload cost* problems. In Sect. 3, we provide formal definitions for path, flow, walk and tour problems with the minimum *penalized reload cost*. Additionally, we will demonstrate the clear difference from the standard *reload cost* model using an example. Sects. 4.1 and 4.3, present the computational complexity of finding an optimum path, maximum flow, and tour under the *penalized reload cost* model. We prove that finding a path/flow/tour is NP-complete and not approximable within any factor $\alpha < 2$ even if they are restricted to having symmetric reload cost matrix that satisfies the triangle inequality. In Sect. 4.2, we deal with the walk problem and prove that it is NP-complete, and not approximable within any factor $\beta \leq 3$, even when the reload cost matrix is symmetric and satisfies the triangle inequality.

2 Literature review

In recent years, only a few key works have been presented in the literature, although the concept of reload cost can find application in many utilization areas and usefulness in modeling complex cost structures in the telecommunications and transportation industry and energy distribution problems. Reload cost applied to spanning tree

problem has been presented in [2–5], to paths, tours and flows in [6], to cycle cover in [7, 8], to paths, trails and walks in [9], and to the minimum diameter spanning tree in (*Diameter-Tree*) [2]. Particular attention has been given to the concept of diameter, because it is an upper limit on the reload cost between any two nodes. *Diameter-Tree* [2] problem is not approximable even if it is restricted to graphs of maximum node degree 5 but it can be solved exactly when the maximum node degree is reduced to 3. Furthermore, when the triangle inequality holds the problem is not approximable within any factor $\alpha < 1/6\ ln\,|V|$ on graphs of $|V|$ vertices. Instead, the minimum diameter spanning tree [3] cannot be approximated within any constant $\alpha < 2$, which was proposed by [2] as an open question. Besides, if the reload costs satisfy the triangle inequality then the problem is not approximable within any constant $\alpha < 5/3$. Some heuristics methods have been presented to solve the minimum reload spanning tree such as: an Ant Colony Optimization approach, a greedy and a random search techniques [10] and tree-nontree edge swap neighborhood [11]. The complexity of problems involving paths, trails (which allow a vertex to be revisited), and walks (which allow vertices and edges to be revisited) with symmetric and asymmetric costs between a fixed couple *s*,*t* of source and destination was studied in [9]. An extensive discussion of the complexities of several problems, dealing with finding paths, tours and flows, can be found in [6]. Some of these problems, such as the shortest paths and the minimum cost flows are solvable in polynomial time. However, others such as the minimum shortest path tree and minimum unsplittable multicommodity flows have been proven to be NP-Hard. They demonstrate that even on graphs where maximum degree is 4, finding a path tree rooted from s that minimizes the maximum among the transportation costs of its paths is not approximable within any factor $\alpha < 2$. Additionally, in undirected graphs with reload costs satisfying the triangle inequality, it is not possible to approximate the problem within any factor $\beta < 5/3$.

3 Definitions and preliminaries

Let $\mathcal{G} = (N, A, L, c, u, w, s, t, f)$ be a digraph with a node set N, an arc set A, a label set L, f is the value of the maximum flow. We identify two specific nodes: *s*,*t*, which represent the source and the destination, respectively. Further, three arc functions are defined: the labeling function l, a weight function w, and a capacity function u, all of them assign non-negative values to each arc $(i, j) \in A$. Moreover, a reload cost function c is defined on all couple (l_1, l_2), that assigns zero whenever the labels are equivalent (i.e., $l_1 = l_2$) and a value greater than zero otherwise. Additionally, we can say that the triangle inequality holds when given a triple of edge (e_1, e_2, e_3), where all of them are incident on the same node then $c(l(e_1), l(e_3)) \leq c(l(e_1), l(e_2)) + c(l(e_2), l(e_3))$. Without loss generality, we suppose to not have parallel edges, whenever it happens we can transform the parallel edges

into a chain of them, and we deal with directed graphs. Then, the reload cost arises whenever two consecutive arcs differ in label. We distinguish three types of costs:

- *reload cost*, that is the expense coming from finding a specific substructure (classical version of reload cost found in the literature);
- *reload cost component*, which is the cost incurred by every connected component obtained after the deletion of the found substructure;
- *penalized reload cost*, which is the aggregate of all preceding costs.

In the reload cost graph, the cost is only generated by the consecutive arc couples; a pair of arcs $[(h, i), (i, j)]$ is considered consecutive if they share an endpoint node i. Given a path/walk/tour/flow $x = \{x_{ij}\}$, then, the *penalized reload cost* $r_p(x)$ is obtained by adding the reload cost $r(x)$ and reload cost component $r_C(\Omega(x))$, that is $r_p(x) = r(x) + r_C(\Omega(x))$, where the reload cost is given by

$$r(x) = \sum_{[(h,i),(i,j)]\in x} c(l(h,i), l(i,j))$$

and reload cost component of every connected component $y \in \Omega(x)$ can be depicted as follows:

$$r_C(y) = \sum_{[(h,i),(i,j)]\in y\,:\,h<j} c(l(h,i), l(i,j))$$

Let $\mathcal{G}^x = (N^x, A^x)$ denotes the subgraph of $\mathcal{G}$ induced by x, which contains only the arcs and nodes included into x. Here, $\Omega(x)$ stands for all connected components in $(\mathcal{G} \setminus \mathcal{G}^x)$ obtained by removal of the nodes and arcs in $\mathcal{G}^x$ from $\mathcal{G}$, that is $(\mathcal{G}\backslash\mathcal{G}^x) = (N\backslash N^x, A\backslash A^x)$. Then, we indicate with $r_C(\Omega(x))$ the total reload cost component found that is obtained as:

$$r_C(\Omega(x)) = \sum_{y\in\Omega(x)} r_C(y)$$

Furthermore, we denote the overall transportation cost of x as:

$$C(x) = r_p(x) + w(x),$$

being *penalized reload cost* $r_p(x) = r(x) + r_C(\Omega(x))$, and and total weight $w(x) = \sum_{(i,j)\in x} w(i,j)$. Notice that, if $|L| = 1$, all the introduced problems are equivalent to finding a path, walk, maximum flow and tour in a monochromatic graph $\mathcal{G}$, all of them polynomial resolvable, then for this reason, we will assume to deal with graphs where $|L| \geq 2$. With this notation in place, we are ready to formally define a path, maximum flow, walk and tour looking for minimizing the *penalized reload cost*.

3.1 Problems statement

In this paper, the problems we investigate are defined by taking into account unitary weights on arcs and dealing only with the total transportation cost caused by *penalized reload cost*. To assist with understanding, we will illustrate the solution application outcomes for the presented problems, taking into account the example graph $G = (N, A, L, c, u, s, t, f = 2)$ shown in Fig. 1. This network has eleven nodes:

$$N = \{s, 1, \ldots, 9, t\},$$

and eighteen arcs:

$$A = \{(s,1)(s,9), (1,2), (1,3), (2,4), (3,4), (4,5), (4,6), (4,9), (5,9), \\ (6,7), (6,8), (6,9), (7,8), (8,9), (9,s), (9,4), (9,t)\},$$

and ten labels

$$L = \{l_1, \ldots, l_9, l_t\}.$$

Without loss of generality, we can consider the cost function c assigning unitary value to any pair of different labels (l_i, l_j). The capacity function u allocate one unit capacity to the arcs $\{(s,9), (9,s), (4,9), (9,4)\}$ and all others get a value of two. A label l_j is assigned to each arc (i, j) with head j and for each node $j \in N$, apart from the arcs $\{(9,s), (9,4)\}$, which have tail label l_9. The arc notation *[capacity, label]*, shown in Fig. 1, explicitly shows the capacity and the label that goes with each arc.

Definition 3.1 *Penalized Reload Cost Path (PRC-P)*

Given a labeled, reload cost digraph $\mathcal{G} = (N, A, L, c, s, t)$, the Penalized Reload Cost Path (PRC-P) problem is looking for a path P from s to t such that the amount of overall *penalized reload cost* is minimized.

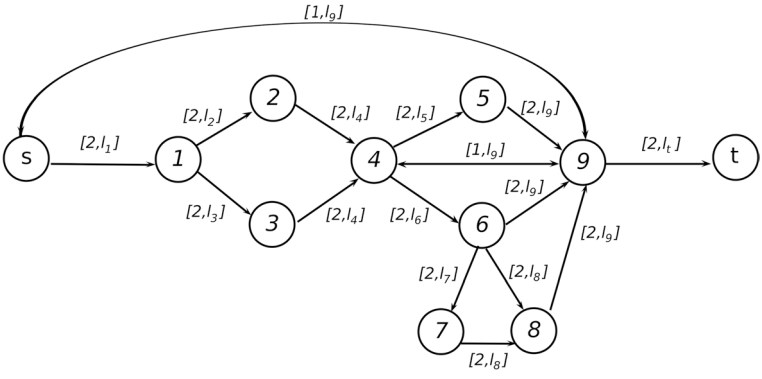


Fig. 1 Example of a reload cost graph G. The notation [capacity,label] denotes capacity and label assigned for each arc.

The optimal solution for PRC-P is the path $p = \{(s,1),(1,2),(2,4),(4,6),(6,9),(9,t)\}$, with a *penalized reload cost* $r_p(x)$ of 5. This is derived from the sum of the reload cost for the path, i.e. $r(p) = 5$, and the zero contribution of reload cost component $r_C(\emptyset) = 0$. Despite the fact that the remaining connected components after p removal are $\{5\},\{7,8\}$, they are not capable of generating any reload cost component with fewer than two consecutive arcs. Instead, if we consider the two arcs path $p' = \{(s,9),(9,t)\}$, we have *penalized reload cost* equals to 10, that is obtained by $r_p(p') = r(p') + r_C(\{1,2,3,4,5,6,7,8\}) = 1 + 9 = 10$. In particular, the reload cost component $r_C(\Omega(p') = \{1,2,3,4,5,6,7,8\}))$ is generated by the nine consecutive arc couples: [(1, 2), (2, 4)], [(1, 3), (3, 4)], [(2, 4), (4, 5)], [(2, 4), (4, 6)], [(3, 4), (4, 5)], [(3, 4), (4, 6)], [(4, 6), (6, 7)], [(4, 6), (6, 8)], [(6, 7), (7, 8)].

Definition 3.2 *Penalized Reload Cost Walk (PRC-W)*
Given a labeled, reload cost digraph $\mathcal{G} = (N, A, L, c, s, t)$, the Penalized Reload Cost Walk (PRC-W) problem is looking for a walk W, which allows node repetitions, from s to t such that *penalized reload cost* is minimized.

The PRC-W application solution $w = \{(s,9),(9,4),(4,6),(6,9),(9,t)\}$, produces *penalized reload cost* equals to 3, that is $r_p(w) = r(w) + r_C(\emptyset) = 3 + 0$. Where reload cost $r(w) = 3$ is produced by consecutive arcs couples [(9, 4), (4, 6)], [(4, 6), (6, 9)], [(6, 9), (9, t)] that differ in label, but there are not consecutive arcs into connected components $\{1,2,3\},\{7,8\}$, that can produce any reload cost component cost.

Definition 3.3 *Penalized Reload Cost Tour (PRC-T)*
Given a labeled, reload cost digraph $\mathcal{G} = (N, A, L, c, s)$, the Penalized Reload Cost Tour (PRC-T) problem is looking for a tour T, from a specified node s such that the *penalized reload cost* is minimized.

The PRC-T example solutions $\tau = \{(s,1),(1,2),(2,4),(4,9),(9,s)\}$, $\tau' = \{(s,1),(1,3),(3,4),(4,9),(9,s)\}$ and $\tau'' = \{(s,1),(1,2),(2,4),(4,6),(6,9),(9,s)\}$ are optimal, having the same *penalized reload cost* value, $r_p(\tau) = r_p(\tau') = r(\tau) + r_C(\{6,7,8\}) = 3 + 1 = 4$ and $r_p(\tau'') = r(\tau') + r_C(\emptyset) = 4 + 0 = 4$.

Definition 3.4 *Penalized Reload Cost Maximum Flow (PRC-MF)*
Given a capacited, labeled, reload cost digraph $\mathcal{G} = (N, A, L, c, u, s, t, f)$ of maximum flow value f, the *Penalized Reload Cost Maximum Flow (PRC-MF)* problem is looking for a feasible optimum maximum flow x^* from s to t such that the amount of *penalized reload cost* is minimized and the subgraph $\mathcal{G}^x$ contains no cycles.

Instead, as regards the PRC-MF application solution $x = \{(s,1),(1,2),(2,4),(4,6),(6,9),(9,t)\}$ with maximum flow f=2, has *penalized reload cost* $r_p(x) = r(x) + r_C(\emptyset) = 5 + 0 = 5$.

4 Hardness and approximation results

In the following subsections, we will provide definitions and findings regarding the complexity and approximability of each specific problem discussed in this paper. Especially, Sect. 4.1 addresses Penalized Reload Cost Path (PRC-P) and Penalized Reload Cost Maximum Flow (PRC-MF) problems, Sect. 4.2 deals with the Penalized Reload Cost Walk (PRC-W) and the last Sect. 4.3 with Penalized Reload Cost Tour (PRC-T) problem.

4.1 PRC-P and PRC-MF problems

In this subsection, we prove that PRC-P and PRC-MF are NP-complete and we provide inapproximability results. Before proceeding with the complexity proofs we begin to introduce the decision version of PRC-P as follows:

Definition 4.1 *k-Penalized Reload Cost Path (kPRC-P)*

INSTANCE: A labeled, reload cost digraph $\mathcal{G} = (N, A, L, c, s, t)$ and a non-negative integer k.
QUESTION: Is there a path P from s to t such that the *penalized reload cost* is less than or equal to k?

Next, we prove that kPRC-P is NP-complete, in doing so, we prove the NP-completeness by reduction from the Hamiltonian Path problem, well known to be NP-complete [12].

Definition 4.2 *Hamiltonian Path*

INSTANCE: Undirected network $\mathcal{G} = (V, E, v_s, v_t)$.
QUESTION: Is there an $v_s - v_t$ path which visits every vertex in $\mathcal{G}$ exactly once?

Since a non-deterministic algorithm requires polynomial time to determine whether a path P from s to t produces *penalized reload cost* at most k, so kPRC-P is in NP.

Theorem 4.1 *The kPRC-P problem is NP-complete.*

Given an instance $\mathcal{G} = (V, E, v_s, v_t)$ of HP, we seek to determine if there is a path P_{HP} from v_s to v_t in $\mathcal{G}$, with $|V(P_{\text{HP}})| = k$, which visits every vertex $v \in V$ exactly once. This is equivalent to determining if there exists a path $P_{\text{PRC-P}}$ for the kPRC-P problem of *penalized reload cost* equal to k. We formalize this observation in the following proof of Theorem 4.1.

Proof We construct a network $\mathcal{G}\simeq = (N, A, L, s, t)$ for an arbitrary instance $\mathcal{G} = (V, E, v_s, v_t)$ of the HP problem, where the size of the label set is $|L| = k + 1$ with $k = |V|$. To construct $\mathcal{G}\simeq$ we perform the following steps

1. Create two dummy vertices s and t.
2. Create two directed arcs (s,v_s) and (v_t,t), with labels l_{vt},l_{vs} respectively.
3. For each edge $e = (i,j) \in E$ create two directed arcs (i, j) and (j, i) with labels l_j and l_i, respectively.
4. Create a clique of size $k + 2$ for each vertex $v \in V$, having as nodes $\{v, v_1, \ldots, v_{k+1}\}$ such that $v < v_1 < v_2 < \ldots < v_{k+1}$ and all the created arcs are labeled with label l_v except for the arc (v, v_1) labeled as $l_{v'}$ (instead of the arc (v_1, v), labeled with l_v).
5. We assign $c(l_i, l_j) = 1$ for any couple of labels $(l_i, l_j) \in L$ with $l_i \neq l_j$, value zero otherwise.

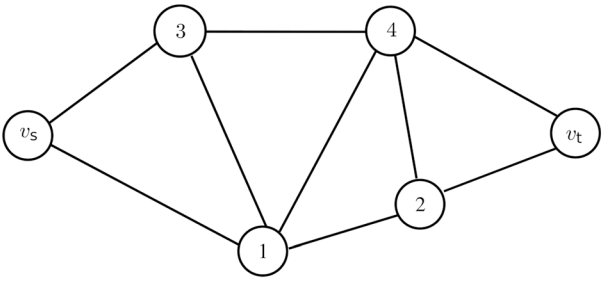
Figure 2 shows an example of the construction that can be performed in polynomial time.

$\Rightarrow$" Let $P_{\text{HP}} = \{v_s, \ldots, v_t\}$ be a Hamiltonian Path between v_s and v_t. We can construct the path $P_{\text{PRC-P}} = \{s\} \cup P_{\text{HP}} \cup \{t\}$ from s to t by selecting only the edges that belong to P_{HP} and arcs (s, v_s) and (v_t, t). The deletion of $P_{\text{PRC-P}}$ removes all the vertices $v \in V$ which decreases the size of each clique constructed on each vertex by one unit, thus producing $|\Omega(P_{\text{PRC-P}})| = |V|$ connected components, with cardinality equal to $k + 1$ and zero cost, so $r_C(\Omega(P_{\text{PRC-P}})) = 0$. This is because the path P deletes all the arcs having different label from cliques. The *penalized reload cost* is restricted to the path reload cost, that is $r_p(P_{\text{PRC-P}}) = r(P_{\text{PRC-P}}) = k$.

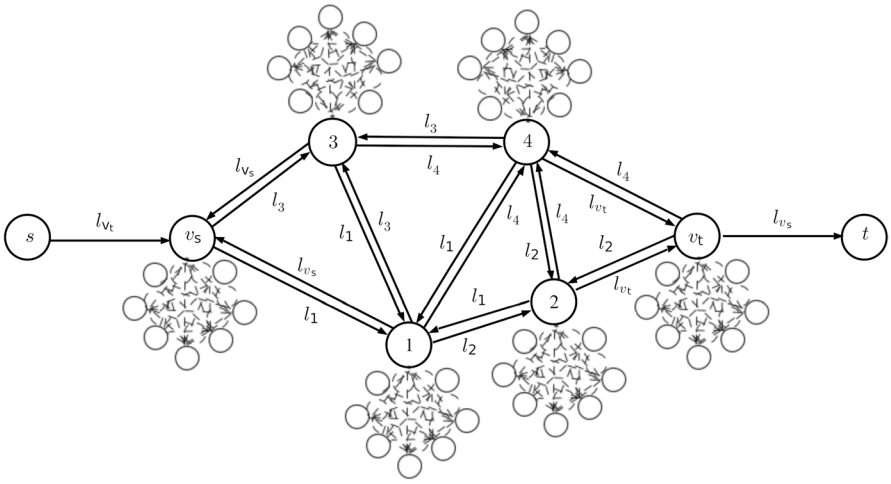
"$\Leftarrow$" If the kPRC-P problem is a yes instance for $\mathcal{G}'$, then there is a kPRC-P path $P_{\text{PRC-P}} = \{s, v_s, \ldots, v_t, t\}$ from s to t of $r_p(P_{\text{PRC-P}}) = k$, then this means that the instance covers all the V nodes because otherwise each node v not included in the solution should increment r_p of k units. As result of all the k arcs couples $[(v, v_1)(v_1, v_z)]$ $z = 2, \ldots, k + 1$ which differ in label, belonging to the clique built on v. By construction, the path from v_s to v_t is a Hamiltonian Path. □

Therefore, the same construction of the Proof of Theorem 4.1, presented above can be used for the decision version of PRC-MF, that is the k-Penalized Reload Cost Maximum Flow problem, that is looking for a maximum flow between s and t in a digraph with *penalized reload cost* less than or equal to k. This proof is obtained in an equivalent way by assigning unitary capacity to every arc and sending a unit of flow in the graph. This leads us to the following corollary:

Corollary 4.1.1 *Given a capacited labeled, reload cost digraph $\mathcal{G} = (N, A, L, c, w, s, t, f)$ of maximum flow value f, the Penalized Reload Cost Maximum Flow (PRC-MF) problem, that is looking for a flow x from s to t such that minimizes the amount of penalized reload cost in an acyclic G^x, is NP-complete.*



a) Original graph example $\mathcal{G}$.



b) *Graph $\mathcal{G}'$ obtained from graph $\mathcal{G}$ for the kPRC-P complexity proof.*

Fig. 2 Example construction of proof Theorem 4.1

So, from Theorem 4.1, we can claim that deciding between YES/NO is hard, we would like to show that even deciding, between those instances that are (almost) satisfiable and those that are far from being like, is also hard. In other words, there exists a gap between YES and NO instances. This gap implies the hardness of the approximation of the optimization version.

Theorem 4.2 *The PRC-P problem cannot be approximated with a factor $\alpha < 2$ unless P=NP.*

We use the same reduction presented in Proof of Thereom 4.1, to derive the non-approximability bound. We consider $opt(\mathcal{G}')$ to be the feasible optimal solution of the PRC-P problem found on the graph $\mathcal{G}'$. Now we show that this reduction introduces a gap such that:

I. if HP is a yes-instance: $opt(\mathcal{G}') \leq K$
II. if HP is a no-instance: $opt(\mathcal{G}') \geq 2K$

To prove (I.), we observe that HP is a Hamiltonian path, from its definition all the vertices are covered once by the path, then $opt(\mathcal{G}') \leq K$ by construction. To prove (II.), we suppose to have a solution with $opt(\mathcal{G}') \leq 2K - 1$, this means there exists a PRC-P solution P from s to t that meets all the nodes, because every node not included in the path increases the reload cost of a factor equal to K. This path is forced to be exactly a Hamiltonian path HP, a contradiction holds for the property (I.) that affirms that $opt(\mathcal{G}') \leq K$, but then no optimal solution can exist with $opt(\mathcal{G}') < 2K$. Properties (I.) and (II.) together imply a gap-introducing reduction. It follows that PRC-P is not approximable within $\frac{2K - \epsilon'}{K}$ for any $\epsilon' > 0$. Then we can conclude that unless $P = NP$ for any given $\epsilon > 0$, PRC-P is not approximable within a ratio $2 - \epsilon$, since $2 - \epsilon \leq \frac{2K - \epsilon'}{K}$.

Considering the proof above we note that the proof holds even if the reload matrix is symmetric and triangle inequalities are satisfied. Thus, this leads us to the following corollaries.

Corollary 4.2.1 *The PRC-P problem is not approximable within any factor* $\alpha < 2$, *even if the reload matrix is symmetric and satisfies the triangle inequality.*

Following the same line of reasoning as done for Corollary 4.2.1, we obtain the following result

Corollary 4.2.2 *The PRC-MF problem is not approximable within any factor* $\alpha < 2$, *even if the reload matrix is symmetric and satisfies the triangle inequality.*

4.2 PRC-W problem

Before turning our attention to the walking problem we state its decision version kPRC-W as follows:

Definition 4.3 k-Penalized Reload Cost Walk (kPRC-W)
INSTANCE: A labeled, reload cost digraph $\mathcal{G} = (N, A, L, c, s, t)$ and a non-negative integer k.
QUESTION: Is there a walk W from s to t such that the *penalized reload cost* is less than or equal to k?

We propose a reduction from the 3SAT problem:

Definition 4.4 3SAT

INPUT: Collection $C = \{c_1, c_2, c_3, \ldots, c_k\}$ of clauses on a finite set of variables $U = \{u_1, u_2, u_3, \ldots, u_m\}$ such that $|c_i| = 3$ for $1 \leq i \leq k$.

OUTPUT: Is there an assignment for U that satisfies all the clauses in C?

3*SAT* is known to be NP-Complete in the sense strong. We again refer to Garey and Johnson [12]. We construct a graph $\mathcal{G} = (V, E, L, c, s, t)$ for an arbitrary instance ϕ of the 3SAT problem, such that graph $\mathcal{G}$ contains a *k*PRC-W walk from s to t with *penalized reload cost* less than or equal to $k = 3m - 1$, if and only if there is a truth assignment t to the variables which satisfies all the clauses of formula ϕ. This graph $\mathcal{G}$ is constructed as follows:

1. Create two dummy nodes s and t.
2. For each clause $c_i \in \phi$ create a node c_i in the graph $\mathcal{G}$.
3. For each variable $u_i \in U$ create a nodes u_i, $\overline{u}_i$ in the graph $\mathcal{G}$, with $u_1 \leq \overline{u}_1 < u_2 \leq \overline{u}_2 < \ldots < u_m \leq \overline{u}_m$.
4. For each variable $u_i \in U$ $\forall i = 2, \ldots, m$ create a node p_i in the graph $\mathcal{G}$.
5. For every node p_i $\forall i = 2, \ldots, m$ create directed arcs (u_{i-1}, p_i) and $(\overline{u}_{i-1}, p_i)$ with label l_{p_i}.
6. For every node p_i $\forall i = 2, \ldots, m$ create directed arcs (p_i, u_i) and $(p_i, \overline{u}_i)$ with label l'_{p_i}.
7. Create directed arcs (s, u_1) , $(s, \overline{u}_1)$ and label them as l_s.
8. Create directed arcs (u_m, t) , $(\overline{u}_m, t)$ and label both with l_t.
9. For each clause c_j and literal $u_i \in c_j$ create a couple of arcs (c_j, u_i) and (u_i, c_j) in the graph $\mathcal{G}$ with label l_{u_i}.
10. For each clause c_j and literal $\overline{u}_i \in c_j$ create a couple of arcs $(c_j, \overline{u}_i)$ and $(\overline{u}_i, c_j)$ in the graph $\mathcal{G}$ with label $l_{\overline{u}_i}$.
11. Assign unitary cost for each label couple (l_i, l_j) except for the couples (l_{u_i}, l_{u_j}), $(l_{\overline{u}_i}, l_{u_j})$, $(l_{u_i}, l_{\overline{u}_j})$, and $(l_{\overline{u}_i}, l_{\overline{u}_j})$ where the considered reload cost is equal to $3m$.

An example is given in Fig. 3. This construction of $\mathcal{G}$ can be accomplished in polynomial time. We need to show that the proposed construction leads to a one-to-one correspondence between the solutions of 3*SAT* instances and *k*RCP-W problem solutions.

Since it takes polynomial time to determine whether a walk from s to t produces *penalized reload cost* at most k, so *k*PRC-W is in NP.

Theorem 4.3 *The PRC-W problem is NP-complete.*

We formalize this observation in the following proof of Theorem 4.3.

Proof Given a formula ϕ, the 3-SAT seeks to determine if there is a truth assignment t to the variables which satisfies all the clauses of the formula ϕ, this is equivalent to determining if there exists a feasible solution W for *k*PRC-W problem with *penalized reload cost* at most equal to $k = 3m - 1$.

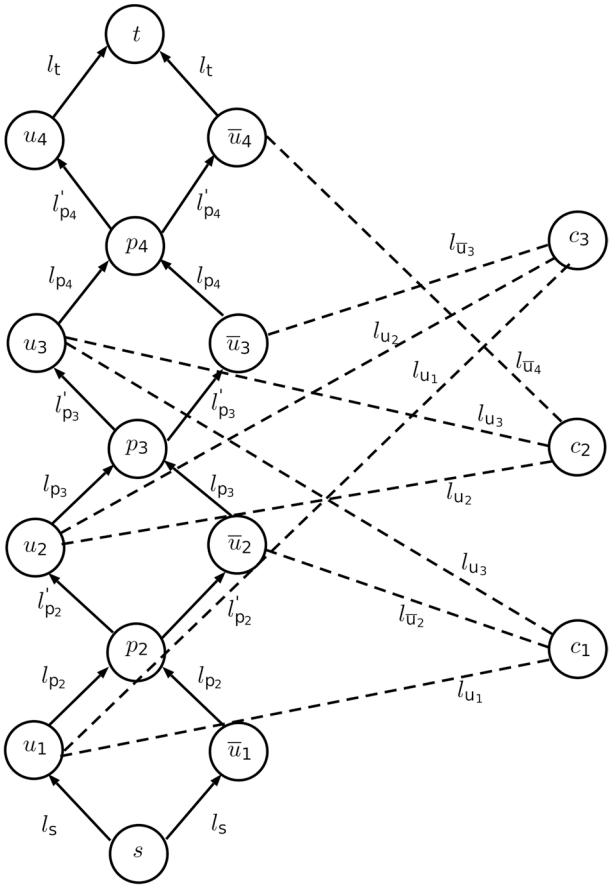


Fig. 3 Graph $\mathcal{G}$ obtained from the formula $\phi = (u_1 \vee \overline{u}_2 \vee u_3) \wedge (u_2 \vee u_3 \vee \overline{u}_4) \wedge (u_1 \vee u_2 \vee \overline{u}_3)$

Without loss of generality, we can suppose to have a clause with no more than one occurrence for each literal."⇒" Let $t = \{u_1 = 1; u_2 = 1; u_3 = 0; u_4 = 0\}$ be a truth assignment to the variables which satisfies the formula $\phi = (u_1 \vee \overline{u}_2 \vee u_3) \wedge (u_2 \vee u_3 \vee \overline{u}_4) \wedge (u_1 \vee u_2 \vee \overline{u}_3)$, at least one variable u_i per clause is assigned the true value, we can construct from graph $\mathcal{G}$ (please refer to Fig. 3) the following walk:

$$W = \{(s,u_1),(u_1,c_1),(c_1,u_1),(u_1,c_3),(c_3,u_1),(u_1,p_2),(p_2,u_2),(u_2,c_2),(c_2,u_2),$$
$$(u_2,c_3),(c_3,u_2),(u_2,p_3),(p_3,\overline{u}_3),(\overline{u}_3,c_3),(c_3,\overline{u}_3),(\overline{u}_3,p_4),(p_4,\overline{u}_4),$$
$$(\overline{u}_4,c_2),(c_2,\overline{u}_4),(\overline{u}_4,t)\}$$

from s to t by selecting only the arcs that are incoming into or coming out nodes correspondent to the variables with true value in the truth assignment, whenever there is an arc going to or from a c_i node, we select it. We can see that W deletion

produces $\Omega(W) = \emptyset$, so *penalized reload cost* is restricted solely to the cost contribution of the walk $r(W) \leq 3m - 1 = k$.

"⇐" If there is a feasible k-Penalized Reload Cost Walk

$$W = \{(s, u_1), (u_1, c_1), (c_1, u_1), (u_1, c_3), (c_3, u_1), (u_1, p_2), (p_2, u_2), (u_2, c_2), (c_2, u_2), (u_2, c_3), (c_3, u_2), (u_2, p_3), (p_3, \overline{u}_3), (\overline{u}_3, c_3), (c_3, \overline{u}_3), (\overline{u}_3, p_4), (p_4, \overline{u}_4), (\overline{u}_4, c_2)(c_2, \overline{u}_4), (\overline{u}_4, t)\}$$

from s to t, with *penalized reload cost* $k = 3m - 1$, we can assign the value 1 to every u_i that is in the walk, this is a feasible truth assignment to the variables which satisfy every clause of ϕ formula. We can suppose for absurd that there is a feasible walk W from s to t, for which does not exist a truth assignment for the formula ϕ. If this is true, means the existence of at least one clause c_i with all false variables, but this is absurd because we should have a connected component composed of the clause c_i and all nodes u_j, representing the false variables in graph $\mathcal{G}$. Then, $r_C(\Omega(W)) \geq 3m$ and the *penalized reload cost* is forced to be greater than k, but this a contradiction because we have supposed W to be an optimal solution. □

Theorem 4.4 *The PRC-W problem is not approximable within any factor* $\alpha \leq 3$.

To prove the non-approximability of PRC-W, we refer to the reduction used in the Proof of Theorem 4.3 presented above. We denote with $opt(\mathcal{G}')$ the PRC-W optimal solution and with t the truth assignment that satisfies all the clauses and assume $K = 3m - 1$. Now we show that this reduction introduces a gap such that:

I. if t is a yes-instance: $opt(\mathcal{G}') \leq K$
II. if t is a no-instance: $opt(\mathcal{G}') \geq 3K + 3$

To prove I.), if t is a feasible truth assignment to the variables which satisfies every clause of ϕ formula than then $opt(\mathcal{G}') \leq K$. So, each clause c_i has at least a literal with true value in the assignment, and then there exists at least an arc covering the node c_i. To prove II.), we suppose that $opt(\mathcal{G}') \leq 3K + 2$ then this means that there exists, for construction, truth assignment to the variables which satisfies every clause, because every clause not satisfied by truth assignment produces a node c_i not included in the walk increasing the reload cost of a factor at least equal to $3K + 3$. But this means that t is a feasible truth assignment. A immediate contradiction to I.) holds, since $opt(\mathcal{G}') \leq K$. Properties I.) and II.) together imply a gap-introducing reduction. Therefore, PRC-W is not approximable within $\frac{3K + 3 - \epsilon'}{K}$ for any $\epsilon' > 0$. Then we can conclude that unless $P = NP$ for any given $\epsilon \geq 0$, W-RCP is not approximable within a ratio $3 - \epsilon$, since $3 - \epsilon \leq \frac{3K + 3 - \epsilon'}{K}$.

4.3 PRC-T problem

We now turn our attention to the problem of finding the tour.

The decision version of PRC-T is stated as follows:

Definition 4.5 *k*-Penalized Reload Cost Tour (kPRC-T)

INSTANCE: A directed capacited and labeled and weighted reload cost graph $\mathcal{G} = (N, A, L, c, s)$ and a non-negative integer k.
QUESTION: Is there a tour T such that the *penalized reload cost* is less than or equal to k?

Next, we prove that *k*PRC-T is NP-complete, in doing so, we prove the NP-completeness by reduction from the Hamiltonian Path problem, well known to be NP-complete [12].

Since a non-deterministic algorithm requires polynomial time to determine whether a tour $T_{\text{PRC-T}}$ produces reload cost at most k, so *k*PRC-T is in NP.

Theorem 4.5 *The kPRC-T problem is NP-complete.*

We construct a network $\mathcal{G}\simeq = (N, A, L, c, s)$ for an arbitrary instance $\mathcal{G} = (V, E, v_s, v_t)$ of the HP problem, where the size of the label set is $|L| = k + 1$ with $k = |V|$, such that the network $\mathcal{G}\simeq$ contains a tour $T_{\text{PRC-T}}$ with the minimum reload cost at most k, if and only if there is a Hamiltonian Path from v_s to v_t. To construct $\mathcal{G}\simeq$ we proceed as follows.

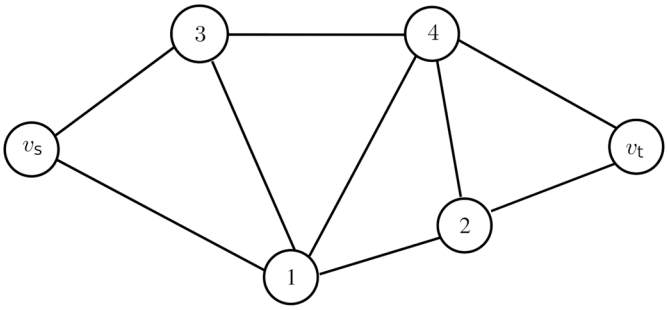
1. Create a vertex s
2. Create the arc (s, v_s) to link v_s to the dummy source s and (v_t, s) to link v_t to s with labels l_{v_s}.

The construction is completed by the execution of steps 3-5 of the proof Theorem 4.1. An example of this polynomial time construction is shown in Fig. 4.

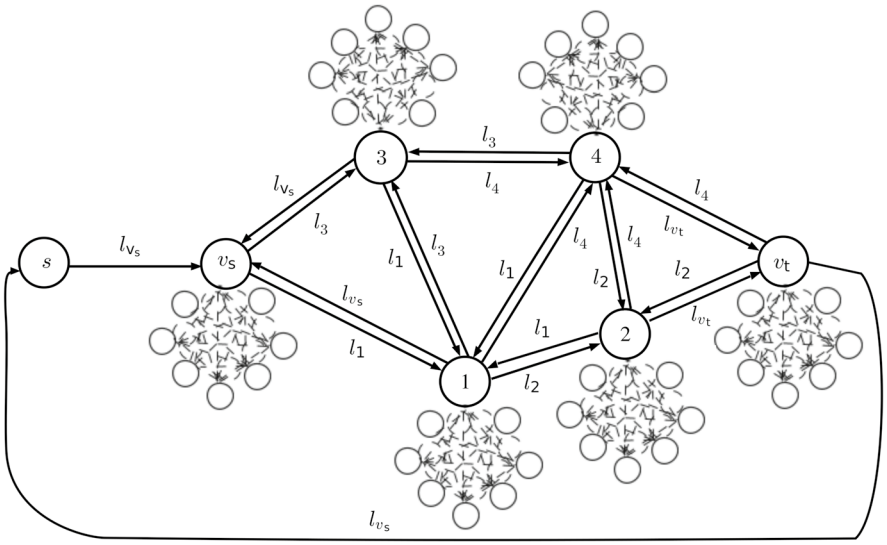
Given an instance $\mathcal{G} = (V, E, v_s, v_t)$, we try to determine whether there is an HP path P_{HP} of $\mathcal{G}$, with $|V(T_{\text{HP}})| = k$, that visits every vertex v exactly once. This is equivalent to determining if there exists a $T_{\text{PRC-T}}$ tour for the *k*PRC-T problem whose deletion creates $|V|$ connected components, all of size k with reload cost component zero and reload cost up to k. We formalize this observation in the following Proof of Theorem 4.5.

Proof "⇒" Let $P_{\text{HP}} = \{v_s, \ldots, v_t\}$ be a Hamiltonian Path between v_s and v_t. We can construct the tour $T_{\text{PRC-T}} = \{s\} \cup P_{\text{HP}} \cup \{s\}$, from s to s, by selecting only the edges that belong to P_{HP} and arcs (s, v_s) and (v_t, s). The deletion of $T_{\text{PRC-T}}$ removes all vertices $v \in V$, not only decrementing the size of each clique, constructed on every vertex, but also removing all the arcs differing in label. Then the *penalized reload cost* $r_p(T_{\text{PRC-T}}) = r(T_{\text{PRC-T}}) + r_C(\emptyset) = k$.

"⇐" If the *k*PRC-T problem is a yes instance for $\mathcal{G}'$, then there is a PRC-T tour $T'_{\text{PRC-T}} = \{s\} \cup P_{\text{HP}} \cup \{s\}$ with *penalized reload cost* $r_p(T_{\text{PRC-T}}) = k$, then this means that the instance is covering all the V nodes, because otherwise each node not included in the solution should increase the *penalized reload cost* of k units (due to



a) Original graph example G.



b) *Graph $\mathcal{G}'$ obtained from graph $\mathcal{G}$ during the construction phase for the kPRC-T proof.*

Fig. 4 Example construction of proof Theorem 4.5

the clique labels). Therefore, by construction the path from v_s to v_s is a Hamiltonian path, meeting all the nodes once. □

Note that using the same procedure used to prove that finding PRC-P is not approximable, (i.e., Theorem 4.2) it is possible to derive the following non-approximability bound.

Corollary 4.5.1 *The PRC-T problem is not approximable within any factor $\beta < 2$.*

Funding Open access funding provided by Consiglio Nazionale Delle Ricerche (CNR) within the CRUI-CARE Agreement.

References

1. Granata, D., Cerulli, R., Scutellà, M.G.., Raiconi, A.: Maximum flow problems and an NP-complete variant on edge-labeled graphs. In: Panos, M.P., Du D.-Z., Graham, R.L. (Eds), Handbook of Combinatorial Optimization, pp. 1913–1948. Springer New York, NY (2013)
2. Wirth, H.C., Steffan, J.: Reload cost problems: minimum diameter spanning tree. Discrete Appl. Math. **113**(1), 73–85 (2001)
3. Galbiati, G.: The complexity of a minimum reload cost diameter problem. Discrete Appl. Math. **156**(18), 3494–3497 (2008)
4. Gamvros, I.: Satellite Network Design, Optimization, and Management. PhD thesis, University of Maryland (2006)
5. Gamvros, I., Gouveia, L., Raghavan, S.: Reload cost trees and network design. Networks **59**(4), 365–379 (2012)
6. Amaldi, E., Galbiati, G., Maffioli, F.: On minimum reload cost paths, tours, and flows. Networks **57**(3), 254–260 (2011)
7. Galbiati, G., Gualandi, S., Maffioli, F.: On minimum reload cost cycle cover. Discrete Appl. Math., **164**, Part 1 (0), 112–120 (2014). Combinatorial Optimization
8. Büyükçolak, Y., Gözüpek, D., Özkan, S.: Minimum reload cost cycle cover in complete graphs. Networks **74**(3) (2019). ISSN 1097-0037
9. Gourvès, L., Lyra, A., Martinhon, C., Monnot, J.: The minimum reload s-t path, trail and walk problems. Discrete Appl. Math. **158**(13), 1404–1417 (2010)
10. Khalil, A., Singh, A.: A swarm intelligence approach to the minimum reload cost spanning tree problem. In: 2010 1st International Conference on Parallel Distributed and Grid Computing (PDGC), 260–265 (2010)
11. Raghavan, S., Sahin, M.: Efficient edge-swapping heuristics for the reload cost spanning tree problem. Networks **65**(4), 380–394 (2015)
12. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W.H. Freeman and Company, San Francisco, CA (1979)