Parameterized Complexity of $k$-Chinese Postman Problem

Gregory Gutin, Gabriele Muciaccia  
Royal Holloway, University of London  
Egham, Surrey TW20 0EX, UK  
gutin@cs.rhul.ac.uk|G.Muciaccia@cs.rhul.ac.uk

Anders Yeo  
Singapore University of Technology and Design  
20 Dover Drive, Singapore 138682, and  
Department of Mathematics, University of Johannesburg  
Auckland Park, 2006 South Africa  
andersyeo@gmail.com

May 11, 2014

Abstract

We consider the following problem called the $k$-Chinese Postman Problem ($k$-CPP): given a connected edge-weighted graph $G$ and integers $p$ and $k$, decide whether there are at least $k$ closed walks such that every edge of $G$ is contained in at least one of them and the total weight of the edges in the walks is at most $p$? The problem $k$-CPP is NP-complete, and van Bevern et al. (to appear) and Sorge (2013) asked whether the $k$-CPP is fixed-parameter tractable when parameterized by $k$. We prove that the $k$-CPP is indeed fixed-parameter tractable. In fact, we prove a stronger result: the problem admits a kernel with $O(k^2 \log k)$ vertices. We prove that the directed version of $k$-CPP is NP-complete and ask whether the directed version is fixed-parameter tractable when parameterized by $k$.

1 Introduction

Let $G = (V,E)$ be a connected graph, where each edge is assigned a non-negative weight (a weighted graph). Herein $n = |V|$ and $m = |E|$. A closed walk is a non-empty multiset $T = \{ e_1, \ldots, e_r \}$ of edges such that there exists a permutation $\sigma$ of $\{ 1, \ldots, r \}$ satisfying the following: $e_{\sigma(i)}$ and $e_{\sigma(i+1)}$ share an end-vertex for every $1 \leq i \leq r$ (where $\sigma(r + 1) = \sigma(1)$). The Chinese Postman Problem is one of the most studied and useful problems in combinatorial optimization.
**Chinese Postman Problem (CPP)**

*Input:* A connected weighted graph $G = (V, E)$ and an integer $p$.

*Question:* Is there a closed walk on $G$ such that every edge of $G$ is contained in it and the total weight of the edges in the walk is at most $p$?

In this paper, we will study the following generalisation of CPP.

**k-Chinese Postman Problem (k-CPP)**

*Input:* A connected weighted graph $G = (V, E)$ and integers $p$ and $k$.

*Parameter:* $k$

*Question:* Is there a set of $k$ closed walks such that every edge of $G$ is contained in at least one of them and the total weight of the edges in the walks is at most $p$?

If a vertex $v$ of $G$ is part of input and we require that each of the $k$ walks contains $v$ then this modification of $k$-CPP is polynomial-time solvable [18] [13]. However, the original $k$-CPP is NP-complete; this result was proved by Thomassen [17]. The following reduction from the 3-Cycle Partitioning Problem is easier than Thomassen’s reduction. In the 3-Cycle Partitioning Problem, given a graph $G$, we are to decide whether the edges of $G$ can be partitioned into 3-cycles. The problem is known to be NP-complete [10]. Let $k = m/3$ and let the weight of each edge of $G$ be 1. Observe that the solution of $(m/3)$-CPP is of weight $m$ if and only if the edges of $G$ can be partitioned into 3-cycles. This reduces the 3-Cycle Partitioning Problem into the $k$-CPP.

Note that the above reduction works because undirected graphs do not contain 2-cycles and any traversal of an edge twice is forbidden as an optimal solution must traverse each edge only once. Directed graphs may contain directed 2-cycles and so the above reduction cannot be used for digraphs unless we restrict ourselves to oriented graphs, i.e., digraphs without directed 2-cycles. However, we could not find, in the literature, a proof that the Directed 3-Cycle Partitioning Problem is NP-complete for oriented graphs, and so we used a different proof, in Section 3, to show that the Directed $k$-CPP, where $G$ is a directed graph, is NP-hard.

While a large number of parameterized algorithmic and complexity results have been obtained for graph, hypergraph and constraint satisfaction

---

[1] For background and terminology on parameterized complexity we refer the reader to the monographs [7], [9], [12].
problems, not much research has been carried out for combinatorial optimisation problem, apart from studying local search for such problems. Perhaps, the main reason is that the standard parameterizations developed for graphs, hypergraphs and constraint satisfaction (such as the value of an optimal solution or a structural parameter) are of little interest for many combinatorial optimisation problems. Recently, Niedermeier’s group identified several practically useful parameters for the CPP and its generalizations, obtained a number of results and posed some open problems, see, e.g., [8, 15, 16]. This research was summarized in [4] and overviewed in [14].

van Bevern et al. [4] (see Problem 3) and Sorge [14] suggested to study the $k$-CPP as parameterized problem with parameter $k$ and asked whether this parameterized problem is fixed-parameter tractable, i.e., can be solved by an algorithm of running time $O(f(k)n^{O(1)})$, where $f$ is a function of $k$ only. In Section 2 we prove that the $k$-CPP is indeed fixed-parameter tractable. In fact, we prove a stronger result: the problem admits a proper kernel with $O(k^2 \log k)$ vertices. This means that, in polynomial time, we can either solve an instance $(G, k)$ of the $k$-CPP or obtain another instance $(G', k')$ of $k$-CPP such that $(G, k)$ is a Yes-instance if and only if $(G', k')$ is a Yes-instance, $G'$ has $O(k^2 \log k)$ vertices and $k' \leq k$. In fact, in our case, $k' = k$.

In Section 3 we prove that the Directed $k$-CPP is NP-hard. It is natural to ask whether the Directed $k$-CPP parameterized by $k$ is fixed-parameter tractable. Our approach to prove that the $k$-CPP is fixed-parameter solvable does not seem to solve the Directed $k$-CPP and the complexity of Directed $k$-CPP remains an open problem.

## 2 Kernel for $k$-CPP

In this section, $G = (V, E)$ is a connected weighted graph. For a solution $T = \{T_1, \ldots, T_k\}$ to the $k$-CPP on $G$ ($k \geq 1$), let $G_T = (V, E_T)$, where $E_T$ is a multiset containing all edges of $E$, each as many times as it is traversed by $T_1 \cup \cdots \cup T_k$. Note that given $k$ closed walks which cover all the edges of a graph, their union is a closed walk covering all the edges and, therefore, it is a solution for the CPP. Hence, the following proposition holds:

**Proposition 1.** The weight of an optimal solution for the $k$-CPP on $G$ is not smaller than the weight of an optimal solution for the CPP on $G$.

**Lemma 1.** Let $T$ be an optimal solution for the CPP on $G$. If $G_T$ contains at least $k$ edge-disjoint cycles, then an optimal solution for the $k$-CPP on $G$ has the same weight as $T$. Furthermore if $k$ edge-disjoint cycles in $G_T$ are given, then an optimal solution for the $k$-CPP can be found in polynomial time.

\(^2\)The notion of a proper kernel was introduced in [1].
Proof. Let \( \mathcal{C} \) be any collection of \( k \) edge-disjoint cycles in \( G_T \). Delete all edges of \( \mathcal{C} \) from \( G_T \) and observe that every vertex in the remaining multigraph \( G' \) is of even degree. Find an optimal CPP solution for every component of \( G' \) and append each such solution \( F \) to a cycle in \( \mathcal{C} \) which has a common vertex with \( F \). As a result, we obtain a collection \( Q \) of \( k \) closed walks for the \( k \)-CPP on \( G \) of the same weight as \( T \). So \( Q \) is optimal by Proposition 1.

Let \( \mathcal{C} = \mathcal{C}_1 \cup \mathcal{C}_2 \cup \mathcal{C}_3 \), where \( \mathcal{C}_1 \) is the set of vertices of degree 1, \( \mathcal{C}_2 \) is the set of vertices of degree 2 and \( \mathcal{C}_3 \) is the set of vertices of degree at least 3. Below we will show that, in polynomial time, we can either solve \( k \)-CPP or bound \( |V| \) from above, by bounding \( |\mathcal{C}_1|, |\mathcal{C}_3| \) and \( |\mathcal{C}_2| \) separately.

Let \( u \) be a vertex with exactly two neighbors \( v \) and \( w \). The operation of bypassing \( u \) means deleting edges \( uv \) and \( uw \) and adding an edge \( vw \) whose weight is the sum of the weights of \( uv \) and \( uw \). Note that the operation of bypassing may create parallel edges.

We will need the following lemma. There, as in the rest of the section, unless stated otherwise, the logarithms are of base 2. The order of a graph is the number of its vertices.

**Lemma 2.** [5] There exists a constant \( c \) such that every graph \( H \) with minimum degree at least 3 and of order at least \( ck \log k \) contain \( k \) edge-disjoint cycles. Such \( k \) cycles can be found in polynomial time.

The fact that the \( k \) cycles in Lemma 2 can be found in polynomial time is not mentioned in [5], but it is easy to deduce it from the greedy algorithm given in the proof of the lemma in [5]. The greedy algorithm repeatedly chooses a shortest cycle (\( k \) times) and deletes it from \( H \). Since a shortest cycle can be found in polynomial time [11], the algorithm is polynomial.

Note that the result of Lemma 2 also holds for multigraphs as if there are parallel edges then there is a cycle of length two.

In the proof of the next lemma, we will use the following well-known fact: there is an optimal solution \( T \) of the CPP on \( G \) which uses at most two copies of every edge of \( G \) and such a solution can be found in polynomial time, see, e.g., [6].

**Lemma 3.** If \( |\mathcal{C}_1| \geq k \) or \( |\mathcal{C}_3| \geq c \log k + k \), where \( c \) is given in Lemma 2, then an optimal solution for the \( k \)-CPP on \( G \) is of the same weight as an optimal solution for the CPP on \( G \). Moreover, an optimal solution for the \( k \)-CPP on \( G \) can be obtained in polynomial time.

**Proof.** Let \( \mathcal{C}_1 = \{v_1, \ldots, v_r\} \) with \( r \geq k \) and let \( w_i \) be the neighbor of \( v_i \). Now, find in polynomial time an optimal solution \( T \) for the CPP on \( G \). In \( G_T \) every vertex is of even degree, hence \( G_T \) contains two copies of the edge \( v_iw_i \) for
every $1 \leq i \leq r$, thus giving at least $k$ edge-disjoint 2-cycles. We conclude by Lemma 1.

Now, assume $|V_1| \leq k$ and $|V_{\geq 3}| \geq ck \log k + k$. Remove all vertices of degree 1 and bypass all vertices of degree 2. The resulting multigraph contains at least $ck \log k$ vertices and has minimum degree at least three, hence by Lemma 2 it contains at least $k$ edge-disjoint cycles, which can be found in polynomial time. Therefore, in polynomial time we are able to find at least $k$ edge-disjoint cycles in $G$ and we conclude using Lemma 1 (note that for every optimal solution $T$ for the CPP on $G$, $G_T$ is a supergraph of $G$).

By Lemma 3, we may assume that $|V_1 \cup V_{\geq 3}| = O(k \log k)$, and so it remains to bound $|V_2|$. In order to do this we will use a reduction rule, but before giving it we will show the following lemma.

**Lemma 4.** There exists an optimal solution, $T$, for the $k$-CPP on $G$, such that no edge in $G$, except possibly one edge of minimum weight, is used more than twice in $T$.

**Proof.** Let $xy$ be an edge of minimum weight in $G$. Let $T$ be an optimal solution for the $k$-CPP on $G$, such that $xy$ is used as many times as possible. Assume for the sake of contradiction that $uv$ is used at least three times in $T$ and $uv$ is distinct from $xy$. Observe that $G_T$ is eulerian and contains $k$ edge-disjoint cycles.

Let $T'$ be obtained from $T$ by removing two copies of $uv$ and adding two copies of $xy$. If $C_1$ and $C_2$ are two edge-disjoint cycles using different copies of $uv$ in $G_T$, then we note that $uvu$ is a cycle in $G_T$ and that there exists a cycle containing only edges from $C_1$ and $C_2$, distinct from $uv$. This implies that deleting two copies of $uv$ from $T$ only decreases the maximum number of edge-disjoint cycles in $G_T$ by at most one. Adding two copies of $xy$ increases the maximum number of edge-disjoint cycles by at least one. Therefore as $G_T$ contains $k$ edge-disjoint cycles, so does $G_{T'}$. This implies that $G_{T'}$ contains a solution to the $k$-CPP of weight at most that of $T$, which is the desired contradiction.

We are now ready to give our reduction rule.

**Reduction Rule 1.** Let $P = v_0v_1 \ldots v_rv_{r+1}$ be a path in $G$ such that $v_i$ is a vertex of degree 2 for $1 \leq i \leq r$. If $r > k$, bypass a vertex $v_i$ such that $2 \leq i \leq r - 1$. Choose $v_i$ in such a way that bypassing it does not change the minimum weight of an edge in $G$.

This rule is safe because in $P$ a solution to the $k$-CPP either duplicates all the edges or it duplicates none (with the exception of a minimum-weight edge, that may be duplicated more than once), and this is true in both $G$ and the
graph $G'$ obtained after an application of the rule. Therefore duplicating a contracted edge corresponds to duplicating all the edges that where contracted. In addition, duplicating the path in $G'$ creates at least $k$ edge-disjoint cycles, as it was for $G$.

From now on, we will assume that $G$ is reduced under Reduction Rule[1] Define a multigraph $H$, such that $V(H) = V_1 \cup V_{\geq 3}$ and add $r$ edges between two vertices, $u$ and $v$, in $H$ if and only if there are exactly $r$ distinct paths from $u$ to $v$ in $G$ where all internal vertices have degree two in $G$ (an edge $uv \in E$ is such a path as it has no internal vertices).

Recall that under our assumptions, $|V(H)| = O(k \log k)$. Moreover, if there are vertices $u, v$ in $H$ such that there are at least $2k$ parallel edges between them, then $G$ contains at least $k$ edge-disjoint cycles (which can be found in polynomial time), and we may apply Lemma[1] Therefore we may assume that this is not the case and this ensures that $|E(H)| = O(k^2 \log k)$. Finally, since $G$ is reduced under Reduction Rule[1] we have $|E(G)| = O(k^3 \log k)$.

However, we can show a better bound on $|E(H)|$, which leads to an improved (polynomial) kernel.

**Lemma 5.** Let $k \geq 2$ and let $c_1$ be any constant. There exists a constant $c_2$ such that every multigraph $H$ with at most $c_1 k \log k$ vertices and at least $c_2 k \log k$ edges contain $k$ edge-disjoint cycles. Such cycles can be found in polynomial time.

**Proof.** Let $c_2 = 2c_1 + 4 + (2 \log c_1 + 2)$. This implies that the following holds (as $k \geq 2$ and therefore $k \log k \geq 1$):

$$c_2 \geq 2c_1 + 4 + \frac{2 \log c_1 + 2}{\log k} = \frac{2c_1 k \log k + k(4 \log k + 2 \log c_1 + 2)}{k \log k}.$$  

(1)

Alon et al. [2] showed that a graph with average degree $d$ and $n$ vertices has a cycle of length at most $2(\log_{d-1} n) + 2$. Note that this result also holds for multigraphs as parallel edges form cycles of length two.

Consider the following greedy algorithm used in [5]: repeatedly choose a shortest cycle and delete its edges from $H$. We will show that by the assumptions of this theorem and the value of $c_2$, it is possible to run this algorithm until $k$ edge-disjoint cycles have been removed and at each step the average degree $\frac{2|E(H)|}{|V(H)|}$ is at least 4. To prove this claim, let us run the algorithm until either it removed $k$ cycles or the average degree dropped below 4 and suppose that the algorithm stopped after removing $0 \leq r < k$ cycles.

Note that the number of edges removed from $H$ by the algorithm is at most the following:

$$r(2 \log_3 n + 2) \leq r(2 \log(c_1 k^2) + 2) = r(4 \log k + 2 \log c_1 + 2).$$  

(2)
Then by (1), (2) and \( r < k \), we have that when the algorithm stops the graph still contains at least \( 2c_1 k \log k \) edges, which implies that the average degree is still at least 4. This is a contradiction, which completes the proof. \( \square \)

Using Lemma 5 we may assume that \(|E(H)| \leq c_2 k \log k\) for some constant \( c_2 \). Since \( G \) is reduced under Reduction Rule 1 we have \(|V_2| = O(k^2 \log k)\), which implies the following:

**Theorem 1.** The \( k \)-CPP admits a kernel with \( O(k^2 \log k) \) vertices.

### 3 NP-completeness of Directed \( k \)-CPP

Our proof below is based on two facts:

- Deciding whether a digraph has at least \( c \) arc-disjoint directed cycles, is NP-complete, see Theorem 13.3.2 in [3];
- The weight of an optimal solution to the \( k \)-CPP on a eulerian digraph \( H \) equals the weight of \( H \) if and only if \( H \) has at least \( k \) arc-disjoint directed cycles (it follows from the fact that each closed walk has a cycle).

In the following part, \( d_D^+(u) \) and \( d_D^-(u) \) denote the outdegree and the indegree, respectively, of a vertex \( u \).

**Theorem 2.** Directed \( k \)-CPP is NP-complete.

*Proof.* Let \( D \) be a directed graph. Now define \( D' \), as follows. Let \( D' \) contain \( D \) as an induced subgraph and add an extra vertex \( x \). Now for every vertex \( u \in V(D) \) with \( d_D^+(u) > d_D^-(u) \) add \((d_D^+(u) - d_D^-(u)) \) paths of length two from \( x \) to \( u \) (the central vertices on the paths are new vertices). For every vertex \( u \in V(D) \) with \( d_D^-(u) > d_D^+(u) \) add \((d_D^-(u) - d_D^+(u)) \) paths of length two from \( u \) to \( x \). Observe that \( D' \) is eulerian.

We will now show that \( D \) contains \( r \) arc-disjoint directed cycles if and only if \( D' \) contains \( r + d_{D'}^+(x) \) arc-disjoint directed cycles. Let \( \mathcal{C} \) be a set of \( r \) arc-disjoint directed cycles in \( D \) and let \( D^* = D' - A(\mathcal{C}) \) be the digraph obtained from \( D' \) by deleting all arcs from cycles in \( \mathcal{C} \). As \( D^* \) is balanced (for every vertex the indegree is equal to the outdegree) and \( d_{D'}^+(x) = d_{D'}^+(x) \) we note that there exists at least \( d_{D'}^+(x) \) arc-disjoint cycles in \( D^* \) (just repeatedly remove any cycle containing \( x \), which can be done \( d_{D'}^+(x) \) times as \( D^* \) will remain balanced). Therefore there are at least \( r + d_{D'}^+(x) \) arc-disjoint cycles in \( D' \).

Conversely, assume that \( D' \) contains \( r + d_{D'}^+(x) \) arc-disjoint cycles. At most \( d_{D'}^+(x) \) of these contain \( x \) and thus the remaining \( r \) cycles must all lie in \( D \).
So to decide whether $D$ has $r$ arc-disjoint cycles is equivalent to deciding whether $D'$ contains $r + d_{D'}^+(x)$ arc-disjoint cycles, which is the same as deciding whether $(r + d_{D'}^+(x))$-CPP on $D'$ has a solution without the need to duplicate any arc (i.e. the weight of the solution is the same as the sum of all arc-weights in $D'$).

\[\square\]

Acknowledgement We’d like to thank the referee for several useful suggestions for improvement of the paper.

References

[1] F. Abu-Khzam and H. Fernau, Kernels: Annotated, Proper and Induced. In IPEC’2006, Lect. Notes Comput. Sci. 4169:264–275, 2006.

[2] N. Alon, S. Hoory, and N. Linial, The Moore Bound for Irregular Graphs. Graphs and Combinatorics 18(1):53–57, 2002.

[3] J. Bang-Jensen and G. Gutin, Digraphs: Theory, Algorithms and Applications, 2nd Ed., Springer, 2009.

[4] R. van Bevern, R. Niedermeier, M. Sorge, and M. Weller, Complexity of Arc Rooting Problems. Chapter 2 in A. Corberán and G. Laporte (eds.), Arc Routing: Problems, Methods and Applications, SIAM, Phil., in press.

[5] H. L. Bodlaender, S. Thomassé, and A. Yeo, Kernel Bounds for Disjoint Cycles and Disjoint Paths. Theor. Comput. Sci. 412(35):4570–4578, 2011.

[6] W.J. Cook, W.H. Cunningham, W.R. Pulleyblank, and A. Schrijver, Combinatorial Optimization, Wiley, 1997.

[7] R. G. Downey and M. R. Fellows. Parameterized Complexity, Springer, 1999.

[8] F. Dorn, H. Moser, R. Niedermeier, and M. Weller. Efficient algorithms for Eulerian extension. SIAM J. Discrete Math. 27(1):75–94, 2013.

[9] J. Flum and M. Grohe. Parameterized Complexity Theory, Springer, 2006.

[10] I. Holyer, The NP-Completeness of Some Edge-Partition Problems. SIAM J. Comput. 10(4):713–717, 1981.

[11] A. Itai and M. Rodeh, Finding a minimum circuit in a graph. SIAM J. Comput. 7: 413–423, 1978.
[12] R. Niedermeier. Invitation to Fixed-Parameter Algorithms. Oxford University Press, 2006.

[13] W.L. Pearn, Solvable cases of the k-person Chinese postman problem. Oper. Res. Lett. 16(4):241–244, 1994.

[14] M. Sorge, Some Algorithmic Challenges in Arc Routing, talk at NII Shonan Seminar no. 18, May 2013.

[15] M. Sorge, R. van Bevern, R. Niedermeier and M. Weller, From Few Components to an Eulerian Graph by Adding Arcs, in WG’2011, Lect. Notes Comput. Sci. 6986:307–319, 2011.

[16] M. Sorge, R. van Bevern, R. Niedermeier and M. Weller, A new view on Rural Postman based on Eulerian Extension and Matching, J. Discrete Alg., 16:12–33, 2012.

[17] C. Thomassen, On the complexity of finding a minimum cycle cover of a graph. SIAM J. Comput. 26(3):675–677, 1997.

[18] L. Zhang, Polynomial Algorithms for the k-Chinese Postman Problem, in Information Processing ’92, vol. 1:430–435, 1992.