The Ferry Cover Problem

Michael Lampis and Valia Mitsou

School of Electrical & Computer Engineering,
National Technical University of Athens, Greece
mlampis@cs.ntua.gr, valia@corelab.ntua.gr

Abstract. In the classical wolf-goat-cabbage puzzle, a ferry boat man must ferry three items across a river using a boat that has room for only one, without leaving two incompatible items on the same bank alone. In this paper we define and study a family of optimization problems called FERRY problems, which may be viewed as generalizations of this familiar puzzle.

In all FERRY problems we are given a set of items and a graph with edges connecting items that must not be left together unattended. We present the FERRY COVER problem (FC), where the objective is to determine the minimum required boat size and demonstrate a close connection with VERTEX COVER which leads to hardness and approximation results. We also completely solve the problem on trees. Then we focus on a variation of the same problem with the added constraint that only 1 round-trip is allowed (FC_1). We present a reduction from MAX-NAE-{3}-SAT which shows that this problem is *NP*-hard and *APX*-hard. We also provide an approximation algorithm for trees with a factor asymptotically equal to $\frac{4}{3}$. Finally, we generalize the above problem to define FC_m, where at most m round-trips are allowed, and MFT_k, which is the problem of minimizing the number of round-trips when the boat capacity is k. We present some preliminary lemmata for both, which provide bounds on the value of the optimal solution, and relate them to FC.

Keywords: approximation algorithms, graph algorithms, vertex cover, transportation problems, wolf-goat-cabbage puzzle.

1 Introduction

The first time algorithmic transportation problems appeared in western literature is probably in the form of Alcuin's four "River Crossing Problems" in the book *Propositiones ad acuendos iuvenes.* Alcuin of York, who lived in the 8th century A.D. was one of the leading scholars of his time and a royal advisor in Charlemagne's court. One of Alcuin's problems was the following:

A man has to take a wolf, a goat and a bunch of cabbages across a river, but the only boat he can find has only enough room for him and one item. How can he safely transport everything to the other side, without the wolf eating the goat or the goat eating the cabbages?

This amusing problem is a very good example of a constraint satisfaction problem in operations research, and, quite surprisingly for a problem whose solution

P. Crescenzi, G. Prencipe, and G. Pucci (Eds.): FUN 2007, LNCS 4475, pp. 227–239, 2007.

is trivial, it demonstrates many of the difficulties which are usually met when trying to solve much larger and more complicated transportation problems ([2]).

In this paper we study generalizations of Alcuin's problem which we call Ferry problems. In these problems, which belong to a wide family of transportation problems, the goal is to ferry a set of items across a river, while making sure that items that remain unattended on the same bank are safe from each other. The relations between items are described by an incompatibility graph, and the objective varies from minimizing the size of the boat needed to minimizing the number of trips.

There are many reasons which make the study of Ferry problems interesting and worthwhile. First, as they derive from a classical puzzle, they are amusing and entertaining, while at the same time having algorithmic depth. This makes them very valuable as a teaching tool because puzzles are very attractive to students. Several other applications of these concepts are possible. For example in cryptography, the items may represent parts of a key and the incompatibilities may indicate parts that could be combined by an adversary to gain some information. A player wishes to transfer a key to someone else, without allowing him to gain any information before the whole transaction is complete.

The rest of this paper is structured as follows: basic definitions and preliminary notions are given in Section 2. In Section 3 we study the Ferry Cover problem without constraints on the number of trips and present hardness and approximation results, as well as results for several graph topologies. Section 4 consists of an analysis of the Trip-Constrained Ferry Cover problem with the maximum number of trips being three, i.e. only one round-trip allowed. We present a reduction from MAX-NAE-{3}-SAT which leads to hardness results for this variation. In Section 5 we analyze the general Trip-Constrained Ferry Cover and Min Ferry Trips problems presenting several lemmata that provide bounds on the value of the optimal solution and relate them to FC. Finally, conclusions and directions to further work are given in Section 6.

2 Definitions – Preliminaries

The rules of the Ferry games can be roughly described as follows: we are given a set of n items, some of which are incompatible with each other. These incompatibilities are described by a graph with vertices representing items, and edges connecting incompatible items. We need to take all n items across a river using a boat of fixed capacity k without at any point leaving two incompatible items on the same bank when the boat is not there. We seek to minimize the boat size in conjunction with the number of required trips to transfer all items.

Let us now formally define the Ferry problems we will focus on. To do this we need to define the concept of a *legal configuration*. Given an incompatibility graph $G(V,E)$, a *legal configuration* is a triple (V_L, V_R, b), $V_L \cup V_R = V, V_L \cap V_R = \emptyset, b \in \{L, R\}$ s.t. if $b = L$ then V_R induces an independent set on G else V_L induces an independent set on G. Informally, this means that when the boat is on one bank all items on the opposite bank must be compatible.

Given a boat capacity k a *legal left-to-right trip* is a pair of legal configurations $((V_{L_1}, V_{R_1}, L), (V_{L_2}, V_{R_2}, R))$ s.t. $V_{L_2} \subseteq V_{L_1}$ and $|V_{L_1}| - |V_{L_2}| \leq k$. Similarly a *right-to-left trip* is a pair of legal configurations $((V_{L_1}, V_{R_1}, R), (V_{L_2}, V_{R_2}, L))$ s.t. $V_{R_2} \subseteq V_{R_1}$ and $|V_{R_1}| - |V_{R_2}| \leq k$. A *ferry plan* is a sequence of legal configurations starting with $(V, \emptyset, L)$ and ending with $(\emptyset, V, R)$ s.t. successive configurations constitute left-to-right or right-to-left trips. We will informally refer to a succession of a left-to-right and a right-to-left trip as a round-trip.

Definition 1. *The* FERRY COVER (FC) *problem is, given an incompatibility graph G, compute the minimum required boat size k s.t. there is a ferry plan for G.*

We will denote by $\mathrm{OPT}_{\mathrm{FC}}(G)$ the optimal solution to the FERRY COVER problem for a graph G.

We can also define the following interesting variation of FC.

Definition 2. *The* TRIP-CONSTRAINED FERRY COVER *problem is, given a graph G and an integer trip constraint m compute the minimum boat size k s.t. there is a ferry plan for G consisting of at most $2m + 2$ configurations, i.e. at most $2m + 1$ trips, or equivalently m round-trips plus the final trip.*

We will denote by $\mathrm{OPT}_{\mathrm{FC}_m}(G)$ the optimal solution of TRIP-CONSTRAINED FERRY COVER for a graph G given a constraint on trips m.

The problem of minimizing the number of trips when the boat capacity is fixed can be defined as follows:

Definition 3. *The* MIN FERRY TRIPS *problem is, given a graph G and a boat size k determine the number of round-trips of the shortest possible ferry-plan for G with capacity k.*

We will denote by $\mathrm{OPT}_{\mathrm{MFT}_k}(G)$ the optimal solution of MIN FERRY TRIPS for a graph G given a boat capacity k. It should be noted that for some values of k there is no valid ferry-plan. In these cases we define $\mathrm{OPT}_{\mathrm{MFT}_k} = \infty$.

For the sake of completeness let us also give the definition of the well-studied *NP*-hard VERTEX COVER and MAX-NAE-{3}-SAT problems ([3]).

Definition 4. *The* VERTEX COVER *problem is, given a graph $G(V, E)$ find a minimum cardinality subset V' of V s.t. all edges in E have at least one endpoint in V' (such subsets are called vertex covers of G).*

We denote by $\mathrm{OPT}_{\mathrm{VC}}(G)$ the cardinality of a minimum vertex cover of G.

Definition 5. *The* MAX-NAE-{3}-SAT *problem is, given a CNF formula where each clause contains exactly* 3 *literals, find the maximum number of clauses that can be satisfied simultaneously by any truth assignment. In the context of* MAX-NAE-{3}-SAT*, we say that a clause is satisfied when it contains two literals with different values.*

Finally, let us give the definition of the H-COLORING problem, which will be useful in the study of FC_1.

Definition 6. *For a fixed graph $H(V_H, E_H)$ possibly with loops but without multiple edges, the H-*Coloring *problem is the following: given a graph $G(V_G, E_G)$, find a homomorphism θ from G to H, i.e. a map $\theta : V_G \to V_H$ with the property that $(u, v) \in E_G \Rightarrow (\theta(u), \theta(v)) \in E_H$.*

The above problem was defined in [4]. Informally, we will refer to the vertices of H as colors.

3 The Ferry Cover Problem

In this section we present several results for the Ferry Cover problem which indicate that it is very closely connected to Vertex Cover. We will show that Ferry Cover is *NP*-hard and that it has a constant factor approximation.

Lemma 1. *For any graph G,* $\mathrm{OPT_{VC}}(G) \leq \mathrm{OPT_{FC}}(G) \leq \mathrm{OPT_{VC}}(G) + 1$.

Proof. For the first inequality note that if we have boat capacity k and $\mathrm{OPT_{VC}}(G) > k$, then no trip is possible because any selection of k vertices to be transported on the initial trip fails to leave an independent set on the left bank.

For the second inequality, if we have boat capacity $\mathrm{OPT_{VC}} + 1$ then we can use the following ferry plan: load the boat with an optimal vertex cover and keep it on the boat for all the trips. Use the extra space to ferry the remaining independent set vertex by vertex to the other bank. Unload the vertex cover together with the last vertex of the independent set. □

Theorem 1. *There are constants $\epsilon_F, n_0 > 0$ s.t. there is no $(1+\epsilon_F)$-approximation algorithm for* Ferry Cover *with instance size greater than n_0 vertices unless P=NP.*

Proof. It is known that there is a constant $\epsilon_S > 0$ such that there is no $(1-\epsilon_S)$-approximation for MAX-3SAT unless P=NP([1]) and that there is a gap preserving reduction from MAX-3SAT to Vertex Cover. We will show that there is also a gap-preserving reduction from MAX-3SAT to Ferry Cover.

The gap-preserving reduction to Vertex Cover in [3] and [6] implies that there is a constant $\epsilon_V > 0$ s.t. for any 3CNF formula ϕ with m clauses we produce a graph $G(V, E)$ s.t.

$$\mathrm{OPT_{MAX-3SAT}}(\phi) = m \Rightarrow \mathrm{OPT_{VC}}(G) \leq \frac{2}{3}|V|$$

$$\mathrm{OPT_{MAX-3SAT}}(\phi) < (1 - \epsilon_S)m \Rightarrow \mathrm{OPT_{VC}}(G) > (1 + \epsilon_V)\frac{2}{3}|V|$$

In the first case it follows from Lemma 1 that

$$\mathrm{OPT_{VC}}(G) \leq \frac{2}{3}|V| \Rightarrow \mathrm{OPT_{FC}}(G) \leq \frac{2}{3}|V| + 1.$$

In the second case,

$$\mathrm{OPT_{VC}}(G) > (1 + \epsilon_V)\frac{2}{3}|V| \Rightarrow \mathrm{OPT_{FC}}(G) > (1 + \epsilon_V - \frac{1 + \epsilon_V}{\frac{2}{3}|V| + 1})(\frac{2}{3}|V| + 1).$$

For $|V| > \frac{3}{2}\frac{1}{\epsilon_v}$ there is a constant $\epsilon_F > 0$ s.t. $\epsilon_V - \frac{1+\epsilon_V}{\frac{2}{3}|V|+1} > \epsilon_F$. Setting $n_0 = \lceil \frac{3}{2}\frac{1}{\epsilon_v} \rceil$ completes the proof. □

Corollary 1. Ferry Cover *is NP-hard*

Proof. It follows from Theorem 1 that an algorithm which exactly solves large enough instances of Ferry Cover in polynomial time, and therefore achieves an approximation ratio better than $(1 + \epsilon_F)$, implies that *P=NP*. □

It should be noted that the constant ϵ_F in Theorem 1 is much smaller than ϵ_V. However, this is a consequence of using the smallest possible value for n_0. Using larger values would lead to a proof of hardness of approximation results asymptotically equivalent to those we know for Vertex Cover. This is hardly surprising, since Lemma 1 indicates that the two problems have almost equal optimum values. Lemma 1 also leads to the following approximation result for Ferry Cover.

Theorem 2. *A ρ-approximation algorithm for* Vertex Cover *implies a* $(\rho + \frac{1}{\mathrm{OPT_{FC}}})$*-approximation algorithm for* Ferry Cover.

Proof. Consider the following algorithm: use the ρ-approximation algorithm for Vertex Cover to obtain a vertex cover of cardinality $\mathrm{SOL_{VC}}$, then set boat capacity equal to $\mathrm{SOL_{FC}} = \mathrm{SOL_{VC}} + 1$. This provides a feasible solution since loading the boat with the approximate vertex cover leaves enough room to transport the remaining independent set one by one as in Lemma 1. Observe that $\mathrm{SOL_{FC}} = \mathrm{SOL_{VC}} + 1 \leq \rho\mathrm{OPT_{VC}} + 1 \leq \rho\mathrm{OPT_{FC}} + 1$ (the first inequality from the approximation guarantee and the second from Lemma 1). □

We now present some examples for specific graph topologies.

Example 1. If G is a clique, i.e. $G = K_n$, then $\mathrm{OPT_{FC}}(G) = \mathrm{OPT_{VC}}(G) = n-1$.

Example 2. If G is a ring, i.e. $G = C_n$ then $\mathrm{OPT_{FC}}(G) = \mathrm{OPT_{VC}}(G) = \lceil \frac{n}{2} \rceil$.

Example 3. Consider a graph $G(V, E)$, $|V| \geq n + 3$ s.t. G contains a clique K_n and the remaining vertices form an independent set. In addition every vertex outside the clique is connected with every vertex of the clique. For example see Figure 1.

We will show that $\mathrm{OPT_{FC}}(G) = \mathrm{OPT_{VC}}(G) + 1$. Assume that $\mathrm{OPT_{FC}}(G) = \mathrm{OPT_{VC}}(G)$. The optimal vertex cover of G is the set of vertices of K_n. A ferry plan for G should begin by transferring the clique to the opposite bank and then leaving a vertex there. On return the only choice is to load a vertex from the independent set, because leaving any number of vertices from the clique is impossible. On arrival to the destination bank we are forced to unload the vertex from the independent set and reload the vertex from the clique. We are now at a deadlock, because none of the vertices on the boat can be unloaded on the left bank.

The graph G described in this example is a generalization of the star, where the central vertex is replaced by a clique. The star is the simplest topology where $\mathrm{OPT_{FC}}(G) = \mathrm{OPT_{VC}}(G) + 1$.

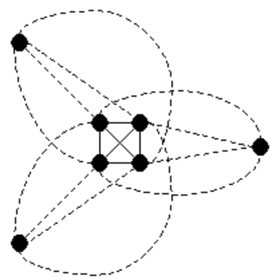

Fig. 1. An example of the graph described in Example 3

The following theorem, together with the observation of Example 3 about stars, completely solves the FERRY COVER problem on trees.

Theorem 3. *If G is a tree and* $\mathrm{OPT_{VC}}(G) \geq 2 \Rightarrow \mathrm{OPT_{FC}}(G) = \mathrm{OPT_{VC}}(G)$.

Proof. Let v_1, v_2 be two vertices of an optimal vertex cover of G. Then v_1 and v_2 have at most one common neighbor, because if they had two then G would contain a cycle. We denote by u the common neighbor of v_1 and v_2, if such a vertex exists.

Then a ferry plan for G is the following: load the vertex cover in the boat and unload v_1 in the opposite bank. Then transfer all the neighbors of v_2 vertex by vertex, leaving vertex u last to be ferried. When u is the only remaining neighbor of v_2 on the left bank, unload v_2 and load u on the boat. On arrival to the destination bank unload u and load v_1. The remaining vertices of the independent set are now transported one by one to the destination bank and finally v_2 is loaded on the boat on the last trip and transported across together with the rest of the vertex cover. □

Remark 1. If $\mathrm{OPT_{VC}}(G)$ for a tree G is 1 (i.e. the tree is a star) then $\mathrm{OPT_{FC}}(G) = 2$ unless the star has no more than 2 leaves, in which case $\mathrm{OPT_{FC}}(G) = 1$.

Corollary 2. *The* FERRY COVER *problem can be solved in polynomial time in trees.*

Proof. The VERTEX COVER problem can be solved in polynomial time in trees. Theorem 3 and Remark 1 imply that determining $\mathrm{OPT_{VC}}$ is equivalent to determining $\mathrm{OPT_{FC}}$.

4 The Trip-Constrained Ferry Cover Problem with Trip Constraint 1

An interesting variation of FC is the TRIP-CONSTRAINED FERRY COVER problem where there is a limit on the number of trips allowed. In this section we study

TRIP-CONSTRAINED FERRY COVER in the case of a very tight trip constraint, i.e. when only one round-trip is allowed (recall that we denote this variation by FC_1). We will show that even in this case the problem is *NP*-hard, by obtaining a reduction from MAX-NAE-{3}-SAT. Our reduction is gap-preserving, and therefore we will also show that FC_1 is *APX*-hard.

We will use the *H*-COLORING problem to obtain an equivalent definition for FC_1.

Lemma 2. *A ferry plan of a graph G for* FC_1 *is equivalent to an F_1-coloring of graph G, where F_1 is the graph of Figure 2.*

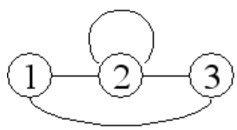

Fig. 2. Graph F_1 of Lemma 2

Proof. Given a ferry plan we can define the following homomorphism θ from G to F_1:

- $\theta(u) = 1$, for all vertices u of G remaining on the boat only during the first trip,
- $\theta(u) = 2$, for all vertices u of G remaining on the boat throughout the execution of the plan,
- $\theta(u) = 3$, for all vertices u of G remaining on the boat only during the final trip.

Given an F_1-coloring we can devise a ferry plan from the above in the obvious way. □

Corollary 3. *For any graph $G(V, E)$* $\mathrm{OPT}_{\mathrm{FC}_1}(G) = \min\{|V_2|+\max\{|V_1|, |V_3|\}\}$, *where the minimum is taken among all proper F_1-colorings of G and V_1, V_2, V_3 are the subsets of V that have taken the colors 1, 2 and 3 respectively.*

Proof. From Lemma 2 we obtain a ferry plan for FC_1: load the subsets V_1 and V_2 in the first trip and unload the subset V_1 in the opposite bank while keeping V_2 on the boat. Then return to the first bank and load V_3 together with V_2 and transport them to the destination bank.

This implies that the boat should have room for V_2 together with the larger of the sets V_1 and V_3. □

From now on we will refer to the value $|V_2| + \max\{|V_1|, |V_3|\}$ as the *cost* of an F_1-coloring. Thus, FC_1 can be reformulated as the problem of finding the minimum cost over all possible F_1-colorings. This reformulation leads to the following theorem:

Theorem 4. FC_1 *is NP-hard. Furthermore, there is a constant* $\epsilon_F > 0$ *s.t. there is no polynomial-time* $(1+\epsilon_F)$*-approximation algorithm for* FC_1*, unless P=NP.*

Proof. We present a gap-preserving reduction from MAX-NAE-{3}-SAT. Our first step in the reduction is, given a formula ϕ with m clauses, to construct a formula ϕ' with $2m$ clauses by adding to ϕ for every clause $(l_1 \vee l_2 \vee l_3)$ the clause $(\overline{l_1} \vee \overline{l_2} \vee \overline{l_3})$. Observe that if a formula contains the clause $(l_1 \vee l_2 \vee l_3)$, we can add the clause $(\overline{l_1} \vee \overline{l_2} \vee \overline{l_3})$ without affecting the formula's satisfiability, since a truth assignment satisfies the first clause (in the NAESAT sense) iff it satisfies both. Note that this also has no effect on the ratio of satisfied over unsatisfied clauses for any truth assignment. In addition, for any i, literals l_i and $\overline{l_i}$ appear in ϕ' the same number of times. Note that, since this is the version of NAESAT where every clause has exactly three literals, the sum of the numbers of appearances of all variables in ϕ' is equal to $6m$.

Next, we construct a graph G from ϕ'. Every variable x_i must appear an even number of times in ϕ', half of them as x_i and half as $\neg x_i$. Let $2f_i$ denote the total number of appearances of the variable x_i. Then, for every variable x_i we construct a complete bipartite graph K_{f_i,f_i}. One half of the bipartite graph represents the appearances of the literal x_i and the other half the appearances of the literal $\neg x_i$.

For every clause $(l_1 \vee l_2 \vee l_3)$, we construct a triangle. We connect each vertex of the triangle to a vertex of the bipartite graph that corresponds to its literal, and has not already been connected to a triangle vertex. This is possible, since the vertices in the bipartite graphs that correspond to a literal l_i are as many as the appearances of the literal l_i in ϕ', and therefore as many as the vertices of triangles that correspond to l_i. This completes the construction, and we now have a graph where every vertex of a triangle has degree 3 and every vertex of a K_{f_i,f_i} has degree $f_i + 1$.

Suppose that our original MAX-NAE-{3}-SAT formula ϕ had m clauses, and we are given a truth assignment which satisfies t of them. Let us produce an F_1-coloring of G with cost $8m - t$. The given truth assignment satisfies $2t$ of the $2m$ clauses of ϕ'. Assign colors 1 and 3 to the vertices of the bipartite graphs, depending on the truth value assigned to the corresponding literal (1 for false and 3 for true). Every triangle corresponding to a satisfied clause can be colored using all three colors, by assigning 1 to a true literal, 3 to a false literal and 2 to the remaining literal. Triangles corresponding to clauses with all literals true are colored with two vertices receiving 2 and one receiving 1. Similarly, triangles corresponding to clauses with all literals false are colored with two vertices receiving 2 and one receiving 3. Note that, due to the construction of ϕ', the number of clauses with all literals true, is the same as the number of clauses with all literals false. Therefore, $|V_1| = |V_3| = \sum_i f_i + 2t + \frac{2m-2t}{2} = 4m+t$, while $|V_2| = 2t + 2(2m - 2t) = 4m - 2t$ making the total cost of our coloring equal to $8m - t$.

Conversely, suppose we are given an F_1-coloring of G with cost at most $8m-t$, we will produce a truth assignment that satisfies at least $2t$ clauses of ϕ' and therefore at least t clauses of ϕ. We will first show that this can be done when the color 2 is not used for the vertices of the bipartite graphs, and then show that any coloring which does not meet this requirement can be transformed to one of at most equal cost that does.

If color 2 is not used in the bipartite graphs, then the cost for these vertices is $\sum_i f_i = 3m$. Therefore, the cost for the $2m$ triangles is at most $5m - t$. No triangle can have cost less than 2, therefore there are at most $m - t$ triangles with cost 3, or equivalently at least $m + t$ triangles of cost 2. Suppose that no triangle uses color 2 three times (if not, pick one of its vertices arbitrarily and color it with 1 or 3, without increasing the total cost). Also, wlog suppose that $|V_3| \geq |V_1|$ (if not, colors 1 and 3 can be swapped without altering the cost).

Now, triangles can be divided in the following categories:

1. Triangles that use color 2 once. These triangles also use colors 1 and 3 once and their cost is 2.
2. Triangles that use color 2 twice and color 1 once. These triangles have a cost of 2.
3. Triangles that use color 2 twice and color 3 once. The cost of these triangles is 3.

Suppose that the first category has k triangles (these correspond to clauses that will be satisfied by the produced truth assignment). Now, $|V_3| \leq \sum_i f_i + m - t + k$, but $|V_3| \geq |V_1| \geq \sum_i f_i + m + t$, thus, $m + t \leq m - t + k \Rightarrow k \geq 2t$. Produce a truth assignment according to the coloring of the bipartite graphs ($1 \rightarrow$ false and $3 \rightarrow$ true). The assignment described above satisfies at least k clauses.

If color 2 is used in the bipartite graphs, we distinguish between two separate cases: first, suppose that the same side of a bipartite graph does not contain both colors 1 and 3. In other words, one side is colored with 1 and 2, and the other with 2 and 3. On the first side, pick a vertex with color 2. If its only neighbor from a triangle has received colors 2 or 3, change its color to 1. If its neighbor has received color 1 exchange their colors. Repeat, until no vertices on that side have color 2 and proceed similarly for the other side, thus eliminating color 2 from the bipartite graphs without increasing the total cost.

Finally, suppose that the same side of a bipartite graph contains both colors 1 and 3 (let A denote the set of vertices of this side). Then, the other side (the set of its vertices is denoted by B) must contain only color 2. We will reduce this case to the previous one. Let A_1 be the subset of A consisting of vertices colored with 1 and A_3 the subset of vertices colored with 3 ($|A_1| + |A_3| \leq |A|$). Let B_1 be the subset of B consisting of vertices connected with triangle vertices colored with 2 or 3, and let B_3 be the subset of B consisting of vertices connected with triangle vertices colored with 2 or 1 ($|B_1| + |B_3| \geq |B|$). Since $|A| = |B|$ then $|A_1| \leq |B_1|$ or $|A_3| \leq |B_3|$. If $|A_1| \leq |B_1|$ then assign color 2 to all vertices of A_1 and color 1 to all vertices of B_1 (this does not increase the total cost), thus eliminating color 1 from side A. If $|A_3| \leq |B_3|$ similarly assign color 2 to the vertices of A_3 and color 3 to the vertices of B_3.

The above reduction shows that given a MAX-NAE-{3}-SAT formula ϕ with m clauses we can construct a graph G s.t.

$$\mathrm{OPT}_{\mathrm{MAX-NAE-\{3\}-SAT}}(\phi) = m \Rightarrow \mathrm{OPT}_{\mathrm{FC}_1}(G) = 7m$$
$$\mathrm{OPT}_{\mathrm{MAX-NAE-\{3\}-SAT}}(\phi) < (1-\epsilon)m \Rightarrow \mathrm{OPT}_{\mathrm{FC}_1}(G) > (1+\epsilon_F)7m$$

where $\epsilon_F = \frac{\epsilon}{7}$. In other words we have constructed a gap-preserving reduction from MAX-NAE-{3}-SAT to FC_1, by making use of the reformulation with H-colorings. Well-known hardness results for MAX-NAE-{3}-SAT (see for example [5]) complete the proof of this theorem. □

Theorem 5. *There is a $\frac{3}{2}$-approximation algorithm for* FC_1 *on trees.*

Proof. First observe that $\mathrm{OPT}_{\mathrm{FC}_1}(G) \geq \frac{n}{2}$, since the boat only arrives to the destination bank twice, and therefore it must be able to carry at least half of the vertices of G. Next, it can be shown that $\mathrm{OPT}_{\mathrm{VC}}(G) \leq \frac{n}{2}$, since on trees there is always an independent set of size at least $\frac{n}{2}$. This can be trivially shown, since trees are bipartite graphs.

A ferry plan for a tree is the following: compute an optimal vertex cover (its size is at most $\frac{n}{2}$) and place all its vertices on the boat. Fill the boat with enough of the remaining vertices so that it contains $\lceil\frac{n}{2}\rceil$ vertices. Move to the other side, compute an optimal vertex cover of the graph induced on the original graph by the vertices on the boat (its size is at most $\frac{\lceil\frac{n}{2}\rceil}{2}$) and keep only those vertices on the boat. Return to transfer the remaining vertices to the destination bank. Clearly, a boat capacity of at most $\frac{\lceil\frac{n}{2}\rceil}{2} + \lfloor\frac{n}{2}\rfloor \leq \frac{3n}{4}$ is sufficient to execute this plan, and this is at most $\frac{3}{2}$ times the optimal. □

The ideas of the previous theorem can be further refined to produce the following result:

Theorem 6. *There is an approximation algorithm for* FC_1 *on trees with approximation guarantee asymptotically equal to $\frac{4}{3}$.*

Proof. Suppose now that instead of transporting $\frac{n}{2}$ vertices on the first trip we wish to transport $\frac{n}{k}$ vertices for some $k > 1$. Upon arrival to the destination bank we unload at least half of them and return with at most $\frac{n}{2k}$ vertices. Now we need to take all the remaining vertices to the other side.

This plan requires a boat capacity of $\max\{\frac{n}{k}, \frac{n}{2k} + n - \frac{n}{k}\}$. It is not hard to see that this is minimized for $k = \frac{3}{2}$. Thus, by taking two thirds of the vertices on the initial trip we devise a ferry plan that requires a capacity of $\frac{2n}{3}$ vertices. Clearly, this is at most $\frac{4}{3}$ times the optimal.

Unfortunately, the preceding analysis requires that n is a multiple of 3. If this is not the case we would be required to take $\lceil\frac{2n}{3}\rceil \leq \frac{2n}{3} + 1$ vertices. This results to an approximation ratio bounded by $\frac{4}{3} + \frac{2}{n}$ which tends to $\frac{4}{3}$ as n tends to infinity. □

5 The Trip-Constrained Ferry Cover and Min Ferry Trips Problems

In this section we study Trip-Constrained Ferry Cover for general values of the trip constraint and present several lemmata which provide bounds on the optimal solution and relate FC_m to FC. We extend the reasoning behind those lemmata to prove a set of similar results for Min Ferry Trips.

First note that a very loose constraint on the number of trips makes the problem equivalent to the Ferry Cover problem.

Lemma 3. *For any graph $G(V,E)$, $|V| = n$, $\mathrm{OPT}_{\mathrm{FC}_{2^n-1}}(G) = \mathrm{OPT}_{\mathrm{FC}}(G)$.*

Proof. Any solution to $\mathrm{FC}_{2^n-1}(G)$ allows a ferry plan with at most 2^{n+1} configurations. There are at most 2^n partitions of the vertices of G into two sets, therefore there are at most 2^{n+1} possible legal configurations. No optimal ferry plan repeats the same configuration twice, since the configurations found between two successive appearances of the same configuration in a ferry plan can be omitted to produce a shorter plan. Therefore, any optimal ferry plan for the unconstrained version has at most 2^{n+1} configurations and can be realized within the limits of the trip constraint. $\square$

Loosening the trip constraint can only improve the value of the optimal solution.

Lemma 4. *For any graph G and any integer $i \geq 0$, $\mathrm{OPT}_{\mathrm{FC}_i}(G) \geq \mathrm{OPT}_{\mathrm{FC}_{i+1}}(G)$.*

Proof. Observe that a ferry plan with trip constraint i can also be executed with trip constraint $i + 1$. $\square$

A different lower bound is given by the following Lemma.

Lemma 5. *For any graph $G(V,E)$, $\mathrm{OPT}_{\mathrm{FC}_m}(G) \geq \frac{|V|}{m+1}$*

Proof. Observe that a trip constraint of m implies that for any ferry plan the boat will arrive at the destination bank at most $m + 1$ times. Therefore, at least one of them it must carry at least $\frac{|V|}{m+1}$ vertices. $\square$

Corollary 4. *There is an $(m + 1)$-approximation algorithm for FC_m.*

Proof. A boat of capacity $|V|$ can trivially solve the problem. From Lemma 5 it follows that this solution is at most $m + 1$ times the optimal. $\square$

Setting the trip constraint greater or equal to the number of vertices makes the constrained version of the problem similar to the unconstrained version.

Lemma 6. *For any graph $G(V,E)$, with $|V| = n$, $\mathrm{OPT}_{\mathrm{VC}}(G) \leq \mathrm{OPT}_{\mathrm{FC}_n}(G) \leq \mathrm{OPT}_{\mathrm{VC}}(G) + 1$.*

Proof. For the first inequality, a boat capacity smaller than the minimum vertex cover allows no trips. For the second inequality it suffices to observe that the ferry plan of Lemma 1 can be realized within the trip constraint. $\square$

Corollary 5. *Determining* $\mathrm{OPT}_{\mathrm{FC}_m}$ *is NP-hard for all* $m \geq n$*. Furthermore, there are constants* $\epsilon_F, n_0 > 0$ *s.t. there is no* $(1 + \epsilon_F)$*-approximation algorithm for* $\mathrm{OPT}_{\mathrm{FC}_m}$ *with instance size greater than* n_0 *vertices unless P=NP.*

Proof. By using Lemmata 6 and 4 we can show that $\mathrm{OPT}_{\mathrm{VC}}(G) \leq \mathrm{OPT}_{\mathrm{FC}_m} \leq \mathrm{OPT}_{\mathrm{VC}}(G) + 1$. The rest of the proof is similar to that of Theorem 1. □

It is unknown whether there are graphs where $\mathrm{OPT}_{\mathrm{FC}_n}(G) > \mathrm{OPT}_{\mathrm{FC}}(G)$. We conjecture that there is a threshold $f(n)$ s.t. for any graph G, $\mathrm{OPT}_{\mathrm{FC}_{f(n)}}(G) = \mathrm{OPT}_{\mathrm{FC}}(G)$ and that $f(n)$ is much closer to n than $2^n - 1$ which was proven in Lemma 3.

Following similar reasoning as in Lemmata 3 - 6 we reach the following results for MFT_k:

Lemma 7. *For any graph* $G(V, E)$*,* $|V| = n$

1. *If* $k = \mathrm{OPT}_{\mathrm{VC}}(G)$ *then* $\mathrm{OPT}_{\mathrm{MFT}_k}(G) \leq 2^n - 1$ *or* $\mathrm{OPT}_{\mathrm{MFT}_k}(G) = \infty$
2. *For any integer* k*,* $\mathrm{OPT}_{\mathrm{MFT}_k}(G) \geq \mathrm{OPT}_{\mathrm{MFT}_{k+1}}(G)$
3. $\mathrm{OPT}_{\mathrm{MFT}_k} \geq \frac{n}{k} - 1$
4. *If* $k \geq \mathrm{OPT}_{\mathrm{VC}}(G) + 1$ *then* $\mathrm{OPT}_{\mathrm{MFT}_k}(G) \leq n$

Proof. Similar to proofs of Lemmata 3,4,5,6 respectively.

MFT_k is at least as hard as FC since determining the optimal number of trips involves deciding whether a ferry-plan is possible with the given boat capacity, which is exactly the decision version of FC. However, it would be interesting to investigate whether MFT_k remains *NP*-hard even for $k \geq \mathrm{OPT}_{\mathrm{VC}} + 1$, in which case there is always a valid ferry plan. We conjecture that the problem remains *NP*-hard in that case.

6 Conclusions and Further Work

In this paper we have investigated the algorithmic complexity of several variations of Ferry problems. For the unconstrained Ferry Cover problem we have presented results that show it is very closely related to Vertex Cover, which is a consequence of the fact that the optimal values for the two problems are almost equal.

For FC_1 we have presented hardness results, but the question of how the problem can be efficiently approximated is open. It would be interesting to see an approximation algorithm which achieves a ratio better than 2, which can be achieved trivially by setting boat size n.

For the Trip-Constrained Ferry Cover and Min Ferry Trips problems, we have presented several lemmata that point out their relation to FC. We believe that these variations are more interesting because they appear to be less related to Vertex Cover than FC. It remains an open problem to determine at which value of the trip constraint FC_m becomes equivalent to FC (however, an upper bound on this value is $2^n - 1$, as shown in Lemma 3). This is a particularly interesting question since so far it remains open whether FC is in *NPO*, or there exist graphs where every optimal ferry plan is of exponential length. However, we believe it is highly unlikely that FC is not in *NPO*. Finally, it would be interesting to investigate whether there are values of m for which FC_m can be solved in polynomial time, but we believe that hardness results similar to those for $m = 1$ and $m \geq n$ hold for all values of m.

Acknowledgements. This paper is based on work done at the Computation and Reasoning Lab at NTUA. The authors would like to thank Aris Pagourtzis and Georgia Kaouri for the original inspiration of the algorithms exercise from which this paper began, Stathis Zachos for useful discussion and the anonymous referees for their comments which helped to clarify the presentation of some of our results.

References

1. Arora, S., Lund, C., Motwani, R., Sudan, M., Szegedy, M.: Proof verification and intractability of approximation problems. In: Proc. 33rd IEEE annual Symposium on Foundations of Computer Science (FOCS), pp. 13–22 (1992)
2. Borndörfer, R., Grötschel, M., Löbel, A.: Alcuin's transportation problems and integer programming (1995)
3. Garey, M.R., Johnson, D.S.: Computers and Intractability: A guide to the theory of NP-completeness. W. H. Freeman and Co., New York, NY (1979)
4. Hell, P., Nešetřil, J.: On the complexity of h-coloring. J. Comb. Theory Ser. B. 48(1), 92–110 (1990)
5. Petrank, E.: The hardness of approximation: Gap location. In: Computational Complexity, vol. 4, Springer, Heidelberg (1994)
6. Vazirani, V.V.: Approximation Algorithms. Springer, Heidelberg (2001)