The Carrier Pigeon Internet Protocol: An Algorithmic (and Lighthearted) Perspective

Matthias Bentert
TU Berlin
Berlin, Germany
matthias@bentert.de

Shay Kutten
Technion
Haifa, Israel
kutten@technion.ac.il

Darya Melnyk
TU Berlin
Berlin, Germany
melnyk@tu-berlin.de

Tijana Milentijević
TU Berlin
Berlin, Germany
tijana.milentijevic@tu-berlin.de

Stefan Schmid
TU Berlin
Berlin, Germany
stefan.schmid@tu-berlin.de

Abstract

The theoretical model behind the pigeon post as a link layer in a communication network was introduced by Shannon (under the guise of studying One-Time Pads for cryptography). That is, to send a one-hop message to v, a node u needs a mail pigeon bred and raised at v. When sending a message using a pigeon to v, node u loses the pigeon. To send another message to v, node u needs another pigeon of v. It has been demonstrated that the communication bandwidth achievable with pigeon post can exceed that of networks using other media. This has already motivated the introduction of Internet standards that allow the use of pigeons as Internet link-layer media.

In this paper, we begin to fill in the missing piece: designing algorithms for breeding and scheduling pigeons to meet a given communication demand efficiently, minimizing the number of pigeons required. We consider singlehop, 2-hop, and multihop pigeon use. While the singlehop variant admits a simple characterization, both the 2-hop and the multihop variants are NP-hard. For the latter variants, we present a polynomial-time algorithm based on demand aggregation that achieves a 2-approximation for the number of pigeons used. We believe that this pigeon-based perspective offers both amusing and instructive insights into network design and hopefully, into ornithology.

CCS Concepts

• **Networks** → **Traffic engineering algorithms**; • **Theory of computation** → *Design and analysis of algorithms.*

Keywords

Internet protocols, routing, non-terrestrial networks, carrier pigeon service

1 Introduction

The AI revolution places heavy demands on communication bandwidth, which is often the bottleneck in AI tasks such as training models; see, e.g., [24, 32]. In the quest to increase bandwidth, researchers turn to the time-tested communication method of the pigeon post, but adapt it for modern communication. See IP over Avian Carriers (IPoAC) [10, 47] and the Carrier Pigeon Internet Protocol (CPIP) [22]. A motivation for such link layer standards can be the demonstration that the bandwidth of communication using pigeons could be much higher than that of competing media[1], since a pigeon can carry a chip containing a substantial amount of information by using the PEI protocol (Pigeon Enabled Internet) in the TCP framework (Transmission by Carrier Pigeons) [7]. This was demonstrated again in [5, 44] and in [35, 43]. Pigeons have also been used in other layers of the Internet, see, e.g. [42, 48].

Environmental aspects also support the use of pigeon post. Fiber optics, the current leading alternative, requires substantial amounts of silicon, commonly in the form of sand. This comes at a time when sand in various parts of the world is disappearing, a development that worries environmentalists, economists, and governments [9, 28, 36]. The UN is also worried [33]. So, next time you are sitting on the beach in the Riviera, or Cancun, or Thailand, or the Maldives, remember that the sand is in danger. If you want to keep having such vacations, supporting Pigeon Post as opposed to fiber optics is the way to go.

In the long term, pigeon post also has the potential to help increase agricultural yields. Using more pigeons generates more guano [7]. This guano is a good fertilizer [30, 39], which suggests that using pigeon post may help grow more food. This, in turn, may allow one to free up land for ecological restoration, thus restoring forests and wetlands, and increasing biodiversity. [2]

The theoretical basis for Pigeon Post was laid in Shannon's seminal work [38]. Shannon, however, did not mention pigeons by name. Instead, he referred to his version of Vernam's encryption protocol [46]. This is known today as a "one-time pad" and appears in many popular spy-related books in the fiction literature, e.g. [19, 41].[3] The spy has some text (or a bit string) given to her by her country's secret service. Let us call this text a "pigeon". Using this pigeon, she encrypts a message to generate a "ciphertext" to be read by the secret service (often, the encryption is just an XOR operation between each bit of the message and the corresponding bit of the pigeon). The service reads the message by applying the reverse operation (often just XORing the ciphertext with the corresponding bit of the same pigeon). Shannon analyzed the version in

[1] Provided that one does not communicate at night and that one does not mind the shit [7].

[2] On a serious note, whatever one may think of pigeon post, the environmental crisis [14] and the hunger crisis [26] are prevalent; let us hope that this paper will make at least a small contribution to increasing awareness.

[3] A reader who reads these two books as a result of our paper has already gained a lot from the paper. The authors do not have any financial interest in the books or the publishers.

Model	Approximation	Number of pigeons	Runtime
SINGLEHOP -	optimal Algorithm 1	$\Theta(n^2)$ Theorem 2	$O(\|S\| + \|D\|)$ Theorem 2
2-HOP -	2-approx. Algorithm 2	$O(\|S\| + \|D\|)$ Theorem 3	$O(n^2)$ Theorem 3
2-HOP - ILP	optimal	optimal	exponential Theorem 5
MULTIHOP -	2-approx. Algorithm 2	$O(\|S\| + \|D\|)$ Theorem 3	$O(n^2)$ Theorem 3
MULTIHOP - ILP	optimal	optimal	exponential Theorem 7

Table 1: Summary of results.

which both sides then discard the pigeon. Hence, if the spy initially has x pigeons to send messages to the secret service (each such pigeon could not be sent to anyone but that secret service), then after the message is sent, the spy has only $x - 1$ pigeons remaining.

As is sometimes regrettably the case, the ancient Egyptians, Persians, Greeks, Romans, etc., did not wait for Shannon's theoretical foundations but instead proceeded with the practice of using pigeons to send messages.[4] See [49].

Our Contributions. We view this paper as an "Introduction to Pigeon Post". We first formally define how network communication can be implemented using mail pigeons (also known as carrier or homing pigeon). We thereby assume that the mail pigeons can be bred in a demand-aware manner, i.e., the breeders are aware of the future communication demands. Optimizing these breeding locations is critical as later in their lives, mail pigeons can only fly in one direction: *home* (their birthplace). The objective is to find a pigeon traffic pattern that uses as few pigeons as possible (model details will follow).

We first analyze a scenario where messages have to be sent *directly* to their receivers, by a single pigeon (singlehop), and present a simple optimal algorithm. We then generalize the model and allow forwarding of messages using two or more homes; these homes can serve as intermediate nodes where messages are relayed to other pigeons (multihop). We show that the general problem is NP-hard, even if the number of homes for forwarding is unlimited.

We further present a 2-approximation algorithm for a scenario where we are allowed to use one intermediate node, that is, two pigeons (2-hop). We also present elegant Integer Linear Programs for the NP-hard problems we discuss. Our results are summarized in Table 1.

Further Related Work. The design of demand-aware structures such as codes (e.g., Huffman codes [25]), data structures (e.g., biased binary search trees [8, 12, 40]), and networks (e.g., splay nets [3, 37]) is an evergreen topic in algorithm theory. Recently, demand-aware networks have received particular interest in the context of reconfigurable datacenter networks, whose topology can be optimized to match the traffic workload [4, 16, 18], see, for example, Google's Jupiter datacenter [34]. Demand-aware networks are attractive as communication traffic is known to exhibit substantial structure [2], which can be exploited for optimization.

[4]Modern physics does not rule out the possibility of time travel, so it may be the case that those early adopters did rely on Shannon's work after all [29].

Many optimization problems in communication rely on demand matrices, including traffic engineering [17], and the early works date back to the beginnings of the Internet [27]. There is also interesting research on estimating the amount of communication to be sent from each node i to each node j [31]. Matrices for a directed graph (such as the graph we use here) are addressed, e.g., in [21]. For even older versions, see e.g., [23] (the famous transportation problem) and [6].

The problem is also connected to multi-commodity flow problems [20] as well as graph layout problems [13, 15]: the design of a demand-aware graph of degree 2 corresponds to the minimum linear arrangement problem [3]. These problems are NP-hard in many scenarios, also on directed networks [15] (like our demand graph). Vehicle routing problem [45] relates to our problem as well. However, different variants of this problem, including the multi-depot vehicle routing [11], to the best of our knowledge, do not consider the relaying of goods between vehicles, which is essential for our problem.

More remotely, our storage model is inspired by the Pigeon-Hole Principle (see, e.g., [1]): that is, if there are more pigeons than holes in a pigeon house, then some holes will simply need to accommodate multiple pigeons.

2 Model

In our model, communication demands may arise between different locations. We represent these locations by a set of nodes V with $|V| = n$ and the communication demand as a directed and unweighted *demand graph* $G_D = (V, E_D)$. There is a directed edge $(i, j) \in V \times V$ with $i \neq j$ in G_D, if there is a communication demand from node i to node j. We assume that the demand graph is stored in an adjacency matrix M_D, which we refer to as the demand matrix. Note that M_D is not necessarily symmetric and satisfies $M_D(i, i) = 0$ for all $i \in V$. Nodes with outgoing demand are referred to as sources $s \in S$, and nodes with incoming demand as destinations $d \in D$; note that a node may play both roles.

The communication infrastructure is induced by pigeons and can be viewed as a dynamic directed multigraph, called *infrastructure graph*. Let P denote a set of pigeons (at a given time). Each pigeon $p \in P$ is associated with two nodes: a home node $h(p) \in V$, where the pigeon was bred and (only) to which it can fly, and a remote node $r(p) \in V$, where the pigeon is initially placed. As we will see, our problems concern defining home and remote nodes so that a minimal number of pigeons can serve a given communication demand. Upon release from its remote location, pigeon p flies directly from $r(p)$ to $h(p)$ without intermediate stops. A pigeon thus induces a directed edge $(r(p), h(p))$ in the infrastructure graph. When p flies, the edge is deleted. Multiple pigeons may induce parallel edges between the same pair of nodes, each used at a different time, to ensure that transportation demand is satisfied.

The communication demand does not need to be carried directly from its source to the destination by a single pigeon. Instead, demand may be routed along directed paths in the infrastructure graph. A unit of demand originating at node i and destined for node j may traverse a sequence of intermediate nodes. At intermediate nodes, demand can be gathered from arriving pigeons and split to the corresponding departing pigeons. Similarly, demand from

multiple arriving pigeons can be batched together and sent using a single pigeon.

We distinguish between singlehop, 2-hop, and multihop uses of pigeons. In the Singlehop - problem variant, each unit of demand must be transported directly from its source to its destination by a single pigeon, without using intermediate nodes. Consequently, no aggregation or forwarding of demand is allowed, and a pigeon flying from node i to node j can only carry demand destined for j that originates at i. In the 2-hop - problem, the demand from node i to node j may either be transported directly or routed via a single intermediate node l, resulting in a path $i \to l \to j$ in the infrastructure graph.

The Multihop - problem is a generalization of the 2-hop - , in which communication demand may be forwarded through multiple intermediate nodes via multiple pigeons sequentially.

In this paper, we assume that pigeons can carry arbitrary amounts of information. When a pigeon p arrives at its home node $h(p)$, all the demand it carries is delivered to the home node. If $h(p)$ is not the final destination of the demand, the information can be stored at that node and forwarded at a later time by pigeon p', which has $h(p)$ as its remote node, i.e., $h(p) = r(p')$. In that case, $h(p)$ is an intermediate node for the demand. Demand may wait arbitrarily long at nodes before being forwarded further. Nodes have unlimited storage capacity and may aggregate incoming demand without restriction. This part of the model actually follows from the Pigeon-Hole Principle: that is, if there are more pigeons than holes at pigeon house $h(p)$, then some holes simply will need to accommodate multiple pigeons. Figure 1 illustrates a demand graph and a corresponding infrastructure graph induced by pigeons in a Multihop - problem.

The demand is delivered successfully if there exists an assignment of all demands to pigeon flights over time such that, for every demand pair $(i, j) \in G_D$, the demand is routed from i to j along a directed path in the (dynamic) infrastructure graph. In general, for the three problems Singlehop - , 2-hop - and Multihop - , the pigeon post operates in two conceptual phases. In the first, offline planning phase, the complete demand matrix M_D is known, and our objective is to breed pigeons (i.e., define their home nodes) and place them in their remote nodes according to our network design decisions. In the second phase, the execution phase, pigeons fly to their home nodes according to the required schedule (i.e., wait for the predecessor pigeon in multihop routing with intermediate nodes). The demand is routed along the established infrastructure graph following a predefined schedule.

Our objective is to construct an infrastructure graph that routes all demands from the demand matrix while minimizing the total number of pigeons. Besides the minimization problems, we also consider decision versions of these problems (e.g., when studying hardness), where we ask whether a given demand can be satisfied with at most k pigeons.

3 Theoretical Results

In this section, we present theoretical results for the three problem variants: Singlehop - , 2-hop - , and Multihop - . For all three problems, we present a simple lower bound on the number of pigeons needed:

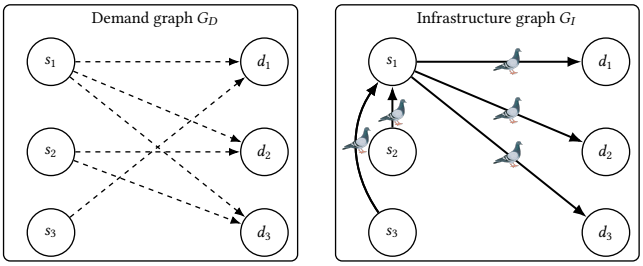

Figure 1: On the left side a demand graph G_D is depicted with 3 source and 3 destination nodes. The right side shows the corresponding infrastructure graph G_I induced by pigeons, illustrating an optimal placement and routing solution that satisfies all the demands while minimizing the total number of pigeons used. In this example, a pigeon p flying from s_3 to s_1 has s_1 as its birthplace, i.e. $h(p) = s_1$. The pigeon is brought to a remote node s_3, i.e. $r(p) = s_3$, and flies home directly to s_1.

Theorem 1 (Lower bound on the number of pigeons). *The minimal amount of pigeons needed to transfer the entire demand is $|P| \geq \max(|S|, |D|)$, where P denotes the set of pigeons, S and D a set of demand sources and demand destinations, respectively.*

Proof. Observe that for each edge (i, j) in the demand graph, there must be at least one pigeon starting in the remote node i. Similarly, there must be at least one pigeon arriving in its home j. The lower bound is achieved by summing over all remote nodes and home nodes, respectively, and choosing the maximum value. □

3.1 Singlehop - Solution

We begin by considering the singlehop pigeons (Singlehop -) problem, in which each unit of demand must be transported directly from its source to its destination by pigeons. In contrast to the multihop setting, intermediate nodes are not permitted, and demand cannot be aggregated or forwarded through other nodes. Consequently, each pigeon can only carry demand originating at its remote node and destined for its home node.

Algorithm 1 shows a trivial solution in the Singlehop - setting, in which each source node sends a pigeon directly to the corresponding destination. In particular, the problem reduces to selecting a pigeon for each edge in the demand graph. Observe that the optimal solution can be computed efficiently, as it only needs to consider all sources and destinations. In the worst case, however, all n^2 demand edges are present in the graph. These results are summarized in the following theorem:

Theorem 2. *Algorithm 1 computes the optimal amount of pigeons in $O(|S| + |D|)$ time. This algorithm uses $\Theta(n^2)$ pigeons in the worst case.*

Algorithm 1 SINGLEHOP - Algorithm

$P \leftarrow \emptyset$
Offline Planning Phase
for all directed edges $(i, j) \in V \times V$ with $i \neq j$ **do**
  if $(i, j) \in G_D$ **then**
    breed pigeon p at node j
    $r(p) \leftarrow i, h(p) \leftarrow j$ ▷ set remote and home node
    ship pigeon p to node i
    $P \leftarrow P \cup \{p\}$
  end if
end for
Flight Scheduling Phase
let all pigeons fly simultaneously to their respective homes

3.2 2-HOP - Solution

We now move beyond direct communication and study the use of intermediate nodes. Allowing demand to be forwarded through other nodes significantly increases the capabilities of the pigeon post by enabling mail aggregation before it is delivered to its final destinations. We first focus on the 2-HOP - problem, in which each demand is routed using at most two pigeon flights. The 2-HOP - problem can be viewed as a restricted form of the general MULTIHOP - problem, in which all mail delivery paths are required to have a length of at most two.

Algorithm 2 presents the 2-HOP - coordinator algorithm. The coordinator algorithm is based on gathering all the demand at one node, called the coordinator. Then, the coordinator spreads the demand and sends pigeons to the corresponding destinations. Note that we consider each weakly connected component separately in the algorithm, and pick the node with the largest degree in the connected component as the coordinator. Any demand in a connected component is then carried from one source to the coordinator (with one pigeon) and then from the coordinator to the destination (using a single pigeon for each destination).

THEOREM 3. *Let c denote the number of weakly connected components in G_D, and $C_1, \ldots, C_c$ be the corresponding weakly connected components. We use $\Delta(C_i)$ to denote the maximum degree of the weakly connected component C_i. Algorithm 2 computes a solution with up to $|S|+|D|-\sum_{i\in[c]} \Delta(C_i)$ pigeons. It is thus a $(2-\sum_{i\in[c]} \Delta(C_i)/n)$-approximation of the optimal 2-HOP - solution. The algorithm runs in $O(n^2)$ time.*

PROOF. In each weakly connected component, Algorithm 2 selects a coordinator node and routes the whole demand in two phases: first, all sources send their entire demand to the coordinator, and second, the coordinator sends the accumulated demand to the corresponding destinations. In each weakly connected component, the algorithm correctly serves the demand.

This solution requires at most one pigeon from each source in S to the respective coordinator and at most one pigeon from the coordinator to each destination in D, resulting in a cost of at most $|S| + |D|$ pigeons. Since the coordinator is chosen to be the highest degree node in the weakly connected component, i.e., it is a source and/or a destination, the solution saves at least as many pigeons as $\Delta(C_i)$ for each weakly connected component $C_i, i \in [c]$. Let ALG denote the total cost of the algorithm. Then, $ALG \leq |S| + |D| - \sum_{i\in[c]} \Delta(C_i)$.

Algorithm 2 2-HOP - Coordinator Algorithm

$P \leftarrow \emptyset$
Execute the following algorithm for each connected component of G_D
coordinator $K \leftarrow$ highest degree node of the current connected component
Offline Planning Phase
for all nodes $i \in V$ with $i \neq K$ **do**
  if $\sum_{j\in V} M_D[i, j] > 0$ **then**
    breed pigeon p
    $r(p) \leftarrow i, h(p) \leftarrow K$ ▷ gather all outgoing demand of i at coordinator
    ship pigeon p to node i
    $P \leftarrow P \cup \{p\}$
  end if
end for
for all nodes $j \in V$ with $j \neq c$ **do**
  if $\sum_{i\in V} M_D[i, j] > 0$ **then**
    breed pigeon p
    $r(p) \leftarrow K, h(p) \leftarrow j$ ▷ send all demand destined for j from coordinator
    ship pigeon p to node K
    $P \leftarrow P \cup \{p\}$
  end if
end for
Flight Scheduling Phase
let pigeons with a home in K fly to K
let pigeons with a home not in K fly from K to their homes

In Theorem 1, we derived a lower bound of $\max(|S|, |D|)$ on the number of pigeons for an optimal solution OPT. The approximation ratio of Algorithm 2 can be computed as

$$ALG \leq |S| + |D| - \sum_{i\in[c]} \Delta(C_i) \leq 2 \cdot \max(|S|, |D|) - \sum_{i\in[c]} \Delta(C_i)$$
$$\leq (2 - \sum_{i\in[c]} \Delta(C_i)/n) \cdot OPT.$$

Hence, the algorithm computes a $(2-\sum_{i\in[c]} \Delta(C_i)/n)$-approximation of the optimal solution.

The computed approximation factor for Algorithm 2 is tight. Consider for example a cyclic demand instance on n nodes, where each node has exactly one outgoing demand and exactly one incoming demand. There is only one connected component in the demand graph, and every node has degree 2. In this case, the optimal solution requires n pigeons, while the Algorithm 2 uses $2 \cdot n - 2$ pigeons, resulting in an approximation factor $2 - 2/n$.

The runtime of the algorithm is dominated by the computation of the weakly connected components which can be done in $O(n^2)$ time. □

In the following, we show that it is hard to compute an optimal solution for the 2-HOP - problem.

Theorem 4. *The decision variant of 2-hop -🕊 is NP-hard.*

Proof. We reduce from 3SAT. An instance of 3SAT consists of a Boolean formula φ in conjunctive normal form, where each clause contains exactly three literals.

Given an instance φ of 3SAT, we construct an instance (G_D, k) of 2-hop -🕊 as follows. The demand graph $G_D = (V, E)$ consists of one node for each clause $C_1, \ldots, C_m$; one node for each positive literal $x_1, \ldots, x_n$ and negative literal $\overline{x}_1, \ldots, \overline{x}_n$; one special node ⭐ and additional $(6n + 12)(2n + 3m)$ nodes, which will be explained later. We set $k = 12n^2 + 18nm + 27n + 39m$.

We place the demand edges as follows: from each clause C_i to each literal x_j or $\overline{x}_j$ that appears in that clause; from each clause C_i to the special node ⭐; from each literal to ⭐; from x_j to $\overline{x}_j$ and from $\overline{x}_j$ to x_j for each $j \in [n]$. Demand edges from each clause C_i to each literal x_j or $\overline{x}_j$ that appears in that clause and between literals are called *forced edges*. In total, there are $2n + 3m$ forced edges. For each forced edge $e = (a, b)$, we add $(6n + 12)$ additional nodes $u^e_{i,j}$ with $i \in [2n + 4]$ and $j \in [3]$. We also add demand edges $(u^e_{i,1}, u^e_{i,2})$, $(u^e_{i,1}, u^e_{i,3})$, $(u^e_{i,2}, u^e_{i,3})$, $(u^e_{i,2}, a)$, $(u^e_{i,3}, a)$, and $(u^e_{i,3}, b)$ for each forced edge $e = (a, b)$ and each $i \in [2n + 4]$. For each forced edge e and each $i \in [2n+4]$, we call the set $\{u^e_{i,1}, u^e_{i,2}, u^e_{i,e}\}$ an *arm* of the forced edge gadget for e. An example of the construction is shown in Figure 2 and the forced edge gadget is illustrated in Figure 3. Since the construction can clearly be computed in polynomial time, it remains to show correctness.

($\Rightarrow$) Assume that φ is a yes-instance, that is, there exists a truth assignment to the variables of φ such that all clauses are satisfied. Now, we build a solution for the constructed instance (G_D, k) of 2-hop -🕊. We place the pigeons in the following way: three pigeons in each clause C_i with home node literal x_j that appears in that clause (total $3m$ pigeons); one pigeon in literal x_j with home in $\overline{x}_j$ and one pigeon in $\overline{x}_j$ with home in x_j for all $j \in [n]$ (total $2n$ pigeons); one pigeon in x_j or $\overline{x}_j$ with home node ⭐ based on the truth assignment in φ for all $j \in [n]$ (total n pigeons). Additionally, for each forced edge $e = (a, b)$ and each $i \in [2n+4]$, we add a pigeon in $u^e_{i,1}$ with home node $u^e_{i,2}$; one pigeon in $u^e_{i,2}$ with home node $u^e_{i,3}$; one pigeon in $u^e_{i,3}$ with home node a (total $6n + 12$ pigeons per forced edge). This construction is shown in Figure 3. Note that the total number of pigeons placed is $3m+2n+n+(2n+3m)(6n+12) = 12n^2 + 18mn + 27n + 39m = k$.

We next discuss how to schedule the pigeons. First, we let the pigeons fly from $u^e_{i,1}$ to $u^e_{i,2}$, then $u^e_{i,2}$ to $u^e_{i,3}$, and finally $u^e_{i,3}$ to a for each forced edge $e = (a, b)$ and each $i \in [2n + 4]$. Next, pigeons fly from each clause C_i to the literals appearing in that clause. Then, pigeons between the literals x_j and $\overline{x}_j$ for all $j \in [n]$. Finally, for every variable x_j, a pigeon is released from x_j to ⭐ if x_j is assigned true and a pigeon is released from $\overline{x}_j$ to ⭐ if it is assigned false.

Now, we show that each demand is satisfied either directly or over two hops using an intermediate node in the constructed infrastructure graph. In particular, whenever a pigeon is placed on an edge (u, v), the corresponding demand (u, v) is satisfied by this direct flight. The only demands that are not satisfied in this way are $(u^e_{i,1}, u^e_{i,3})$, $(u^e_{i,2}, a)$, $(u^e_{i,3}, b)$ for each forced edge $e = (a, b)$ and each $i \in [2n+4]$, $(C_i, ⭐)$ for each clause C_i, and for each variable x_j either $(x_j, ⭐)$ (if x_j is set to false) or $(\overline{x}_j, ⭐)$ (if x_j is set to true).

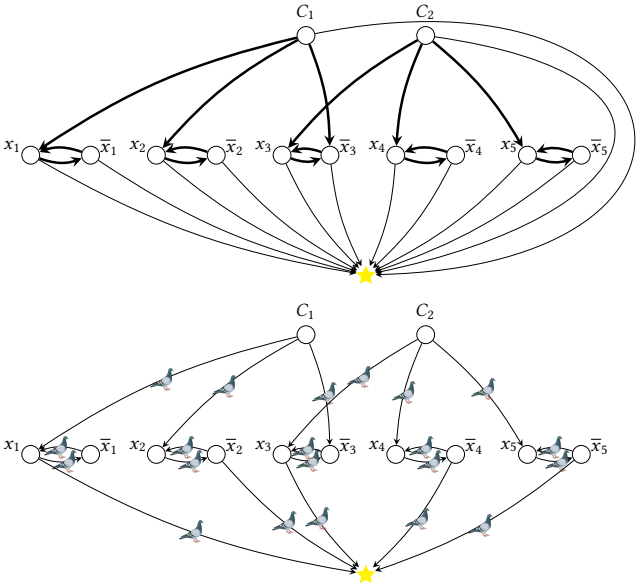

Figure 2: The constructed demand graph for $\varphi = (x_1 \vee \overline{x}_3 \vee x_2) \wedge (x_3 \vee x_4 \vee x_5)$ without forced edge gadgets is illustrated above. Demand edges are added from C_1 and C_2 to their three literal nodes and to the star, from x_j to $\overline{x}_j$ and from $\overline{x}_j$ to x_j for all $j \in [n]$, and from each literal (x_j or $\overline{x}_j$) to the star. Forced edges are depicted with thick lines. Below is the corresponding infrastructure graph for the assignment $(x_1, x_2, x_3, x_4, x_5) =$ (TRUE, FALSE, TRUE, FALSE, FALSE).

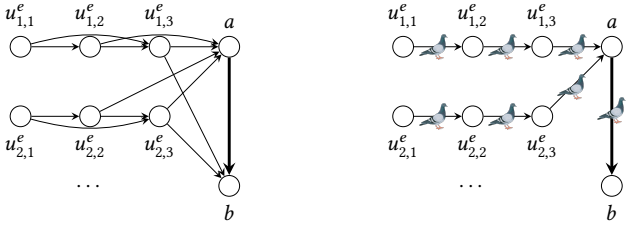

Figure 3: Two arms of a forced edge gadget in the demand graph are illustrated on the left. Edge $e = (a, b)$ is forced and depicted with a thick line. On the right, a corresponding infrastructure graph that satisfies the forced edge gadget from the demand graph with the forced edge and 3 pigeons per arm is shown.

The demands within each forced edge gadget are satisfied over 2-hop paths with pigeons flying $u^e_{i,1} \to u^e_{i,2} \to u^e_{i,3}$; $u^e_{i,2} \to u^e_{i,3} \to a$; and $u^e_{i,3} \to a \to b$, respectively. The demand from each literal x_j assigned false (or $\overline{x}_j$ where x_j is set to true) to ⭐ is satisfied over a path $x_j \to \overline{x}_j \to$ ⭐ (or $\overline{x}_j \to x_j \to$ ⭐). Finally, the demand $(C_i, ⭐)$ is satisfied with a 2-hop flight $C_i \to x_j \to$ ⭐ where x_j is a variable appearing in C_i that is assigned true (or $\overline{x}_j$ and x_j is set to

false). Note that this variable must exist as otherwise φ would not be satisfied, which contradicts our assumption.

($\Leftarrow$) Now, assume that $(G_D, (11n + 15m))$ is a yes-instance of 2-HOP - 🕊. We will show that each forced edge gadget can be assigned a score of $6n + 13$ in any solution where each pigeon gives a total score of at most 1. Hence, a total score of at most $k - (2n + 3m)(6n + 13) = n$ can be unassigned and no forced edge gadget can be assigned more than $7n + 13$ as otherwise the solution contains more than k pigeons. To define the score of a gadget, we distinguish between public and private nodes. A node $u^e_{i,j}$ for some forced edge e and any i, j is called a private node and all remaining nodes ($C_i, x_j, \overline{x}_j$, ⭐) are called public. First, for each private node v, note that at least one pigeon has to leave from v since v is a source for some demand. We arbitrarily assign one of the pigeons leaving v (a score of 1) to v and the score of any private node $v = u^e_{i,j}$ is also assigned the forced edge gadget for e. Afterwards, all unassigned pigeons are assigned as follows: If the pigeon flies from u to v where both u and v are public nodes, then no node is assigned a score, but if (u, v) is a forced edge, then a score of one is added to the forced edge gadget for (u, v). If exactly one of the nodes u and v is a public node and the other is a private node, then a score of one is assigned to the private node. Finally, if both are public nodes, then a score of $\frac{1}{2}$ is added to each of the two nodes. Note that each pigeons gives a total score of at most one to all nodes and also a total score of at most one to each forced edge gadget. Moreover, since each private node is assigned a score of at least one, each forced edge gadget is assigned at least a score of $6n + 12$.

Let $e = (a, b)$ be a forced edge, we show that if no pigeon flies from a to b in the solution, then each arm of the forced edge gadget is assigned a total score of at least 3.5. Since the number of arms is $2n + 4 \geq 2$, each forced edge gadget is thus assigned a score of at least $6n+13$. Moreover, if no pigeon flies from a to b, then the forced edge gadget for e is assigned a score of at least $3.5(2n+4) = 7n+14$. As argued above, this means that the number of pigeons in the solution is larger than k, a contradiction. So now assume towards a contradiction that any arm $\{u^e_{i,1}, u^e_{i,2}, u^e_{i,3}\}$ is assigned a total score of less than 3.5 and no pigeon flies from a to b. Since each pigeon assigns a score of $\frac{1}{2}$ or 1 to a node and each node is assigned a score of at least one, the three nodes in the arm are assigned a score of exactly three and each node in the arm is assigned a score of exactly one. Note that this implies that each node has exactly one pigeon in the solution that leaves the respective node. No pigeon flies from $u^e_{i,3}$ to a as no pigeon flies from a to b in the solution and thus the demand $(u^e_{i,3}, b)$ cannot be satisfied by a 2-hop path. For the same reason, no pigeon flies from $u^e_{i,2}$ to $u^e_{i,3}$ as the demand $(u^e_{i,2}, a)$ could not be satisfied and no pigeon flies from $u^e_{i,1}$ to $u^e_{i,2}$ as the demand $(u^e_{i,1}, u^e_{i,3})$ could not be satisfied. Let c be the node such that a pigeon flies from $u^e_{i,1}$ to c in the solution. We make a case distinction whether $c = u^e_{i,3}$ or not. If $c \neq u^e_{i,3}$, then note that one pigeon has to fly from c to $u^e_{i,2}$ and one pigeon has to fly from c to $u^e_{i,3}$. At most one of these pigeons is fully assigned to c and the other adds a score of at least $\frac{1}{2}$ to the three nodes in the arm, raising the total score to at least 3.5. So assume that $c = u^e_{i,3}$. To satisfy the demand $(u^e_{i,1}, u^e_{i,2})$, one pigeon has to fly from $u^e_{i,3}$ to $u^e_{i,2}$. Then, a pigeon has to fly from $u^e_{i,2}$ to both a and b, contradicting that exactly one pigeon leaves each of the three nodes in the arm. This shows that if no pigeon flies from a to b, then the total score of each arm is at least 3.5 and the number of pigeons in the solution is larger than k. Finally, note that since we may assume that a pigeon flies from a to b for each forced edge $e = (a, b)$, it holds that the score assigned to the forced edge gadget for e is at least $6n + 13$ and each pigeon that flies from a public node to a private node in the gadget increases the score by one.

To conclude the proof, consider the set of all pigeons in a solution that do not fly from a to b for some forced edge (a, b) and that do not start in a private node. These are all but at least $(2n + 3m)(6n + 13) = k - n$. Hence, these are at most n pigeons. Assume towards a contradiction that for some variable x_j, none of these pigeons starts at x_j or $\overline{x}_j$. Then, the only pigeons leaving x_j and $\overline{x}_j$ are the pigeons flying the forced edges $(x_j, \overline{x}_j)$ and $(\overline{x}_j, x_j)$. Thus, the demands $(x_j$, ⭐) and ($\overline{x_j}$, ⭐) are not satisfied, a contradiction. Thus for each variable x_j, exactly one of the n pigeons flies from x_j or from $\overline{x}_j$. We define a truth assignment by setting x_j = TRUE if x_j is the node that the additional pigeon flies from and to FALSE otherwise.

It remains to show that the constructed assignment satisfies φ. To this end, consider any clause C_i and the demand (C_i, ⭐). As no pigeons are left to fly from C_i to ⭐ directly and the only pigeons flying from C_i are to the three nodes representing literals that appear in C_i (the three forced edges incident to C_i), it must hold that a pigeon flies from one of these three nodes to ⭐. These has to be one of the n pigeons that do not fly forced edges and do not start in a private node. Hence, at least one of the three nodes is assigned an additional pigeon and by construction, the assignment of that variable satisfies C_i. Since all variables are satisfied in this way be the assignment, φ is satisfied and the original instance of 3SAT is a yes-instance. This concludes the proof.

□

In the following, we will present an ILP formulation of the 2-HOP - 🕊 problem. To simplify the formulation, we make use of the following lemma:

LEMMA 1. *Any optimal solution for both* MULTIHOP - 🕊 *and* 2-HOP - 🕊 *requires at most* $2(n - 1)$ *pigeons.*

PROOF. We show this statement by presenting a solution that uses $2(n - 1)$ pigeons for any demand graph. We build an infrastructure graph as a directed cycle, where the nodes along the cycle are following some arbitrary order. Assume WLOG that the nodes are ordered as $v_1, v_2, \ldots, v_n$ along the cycle. We place pigeons at the nodes as follows: two pigeons at each node $v_i, i \in [n - 2]$ with a home at v_{i+1}, one pigeon at v_{n-1} with a home at v_n, and one pigeon at v_n with a home at v_1. We release the pigeons sequentially, one at a time, starting with node v_1. This pigeon carries the demand from v_1 to all other nodes in the graph. Once the pigeon arrives at v_2, the remaining demand from v_1 is forwarded with the next pigeon, together with the demand from v_2 to all other nodes. After making one cycle through all nodes, node v_1 will receive a pigeon with demands from nodes $v_2, \ldots, v_n$ with destinations in $v_1, \ldots, v_{n-1}$. Observe that at this point, the demand of node v_1 has been delivered to all nodes. However, the demand of node v_n has only been delivered to node v_1 so far. Therefore, we need to continue forwarding the remaining demand through the cycle, up to

$$\sum_{u,v \in V} x^i_{u,v} \leq 1 \qquad \forall i \in [2n-2] \tag{1}$$

$$\sum_{i \in [2n-2]} (x^i_{u,v} + \sum_{w \in V} y^i_{u,w,v}) \geq 1 \qquad \forall (u,v) \in E \tag{2}$$

$$y^i_{u,w,v} \leq x^i_{u,w} \qquad \forall (u,v) \in E,\ i \in [2n-2] \tag{3}$$

$$y^i_{u,w,v} \leq \sum_{\substack{j \in [2n-2] \\ j > i}} x^j_{w,v} \qquad \forall (u,v) \in E,\ i \in [2n-2] \tag{4}$$

Figure 4: The constraints of the ILP for 2-HOP - .

node v_{n-1}. This construction always forwards all the demand with $2n-2$ pigeons. □

The following theorem establishes the properties of an ILP we present for the 2-HOP - problem.

THEOREM 5. *There is an ILP with $O(n^4)$ variables and $O(n^3)$ constraints for 2-HOP - , where all variables are binary.*

PROOF. Let $(G = (V,E), k)$ be an instance of 2-HOP - . We construct an ILP with a binary variable $x^i_{u,v}$ for each pair $u, v \in V$ and each $i \in [2n-2]$ and a binary variable $y^i_{u,w,v}$ for each edge $(u,v) \in E$, each node $w \in V$, and each $i \in [2n-2]$. The variable $x^i_{u,v}$ is set to true if the i^{th} pigeon flies from u to v. Note that by Lemma 1, we may assume that at most $2n-2$ pigeons are required. The goal is to minimize the number of pigeons, that is, $\sum_{u,v \in V} \sum_{i \in [2n-2]} x^i_{u,v}$, and a solution with k pigeons will exist if and only if the goal value is at most k. The constraints are listed in Figure 4.

Since the number of variables is clearly in $O(n^4)$ and the numbers of constraints is in $O(n^3)$, it remains to show that the ILP is correct. The variable $y^i_{u,w,v}$ will be set to true if and only if the demand (u,v) is routed via node w and the i^{th} pigeon transports the message from u to w.

To show correctness, first assume that there is a solution with k pigeons. We may assume without loss of generality that no two pigeons fly at the same time, and hence we can order all pigeons in the order they fly. Then, we set $x^i_{u,v} = 1$ if and only if the i^{th} pigeon flies from u to v. For each demand (u,v), if any pigeon flies from u to v directly, then we set all variables $y^i_{u,w,v} = 0$. Otherwise, there is at least one node w such that a pigeon i flies from u to w and a later pigeon j that flies from w to u. We arbitrarily chose one such pair w, i and set $y^i_{u,w,v}$ to true. Note that requirements 1, 2, and 3 are now all satisfied by construction. Moreover, since we assume that for each chosen pair (w,i) a later pigeon j flies from w to v, also constraint 4 is satisfied.

In the other direction, assume that there is a solution to the ILP where the goal value is at most k. We will let a pigeon fly from a node u to node v in time step i if and only if $x^i_{u,v} = 1$. Note that by construction at most k pigeons fly (each at some time step between 1 and $2n-2$). It now only remains to show that each demand is satisfied by the constructed solution. So consider any demand (u,v). If some pigeon flies from u to v directly, then the demand is satisfied. Otherwise, $x^i_{u,v} = 0$ for all $i \in [2n-2]$. By constraint 2, at least one variable $y^i_{u,w,v}$ is set to true. By constraints 3 and 4, $x^i_{u,w}$ and $x^j_{w,v}$ are set to 1 for at least one node w and one value $j \in [2n-2]$ with $j > i$. Thus, the demand (u,v) is satisfied by the pigeon flying from u to w at time i and the pigeon flying from w to v at time j. Since the demand was chosen arbitrarily, all demands are satisfied in this way, and the ILP is therefore correct. □

3.3 MULTIHOP - Solution

Observe that Algorithm 2 is also a $(2 - \sum_{i \in [c]} \Delta(C_i))$-approximation for the MULTIHOP - problem. We next show that also the MULTIHOP - problem is NP-hard, and provide an ILP for this problem.

The proof of NP-hardness is based on the following three lemmas:

LEMMA 2. *If the demand graph $G = (V,E)$ is weakly disconnected, then each weakly connected component of G can be solved independently for MULTIHOP - and the minimum number of pigeons required for G is the sum of the minimum numbers of pigeons required for each connected component of G.*

PROOF. Consider a graph $G = (V,E)$ with two weakly connected components, denoted C_1 and C_2. Assume by means of contradiction that there is an optimal solution where a pigeon p flies from C_1 to C_2. Since C_1 and C_2 are weakly disconnected, the destination nodes in C_1 and C_2 are disjoint. Consider first the case where p has not carried any demand, then the solution was not optimal because it wasted a pigeon. Thus, pigeon p must have carried some demand. WLOG, assume that this demand was from sources in C_1, and thus has destinations in C_1. But then, there must exist at least one other pigeon that carries the demand back from C_2 to C_1. In this case, however, we could have saved at least one pigeon by avoiding a detour over the nodes in C_2. This is a contradiction to the fact that we had an optimal solution. □

LEMMA 3. *If the demand graph $G = (V,E)$ is weakly connected, then there is always an optimal solution for MULTIHOP - in which only one pigeon flies at a time and the pigeon starting at time i starts at the same node that the pigeon flying at time $i-1$ ended their flight for all $i > 1$.*

PROOF. Let OPT be an optimal solution. We assume without loss of generality that the pigeon flights in OPT are strictly ordered. This is possible, as any two flights that happen at the same time cannot conflict each other and thus can be ordered arbitrarily. For simplicity, let $e_i = (a_i, b_i)$ denote the edge in the infrastructure graph that corresponds to the i-th pigeon flight in OPT and let OPT_i denote the partial solution only containing the first i flights.

For each $0 \leq i \leq |OPT|$, we build a sequence σ_i of pigeon flights satisfying the following two conditions. First, all pigeon flights in a weakly connected component of the infrastructure graph corresponding to σ_i occur consecutive and the starting node of each pigeon is the destination of the previous pigeon unless it is the first flight in a weakly connected component. Second, for each pair (u,v)

of nodes (not necessarily terminals), if a pigeon route from u to v exists in OPT_i, then it also exists in σ_i, that is, each node has at least as much information after pigeons flew according to σ_i as for OPT_i. Note that if we succeed with this construction for all i, then the final sequence $\sigma_{|OPT|}$ is an optimal solution satisfying the requirements of the lemma.

Let σ_0 be the empty sequence. Note that σ_0 fulfills our requirements. Now assume that σ_{i-1} fulfills our two requirements. We will show how to construct σ_i. Consider the i-th pigeon flight $e_i = (a_i, b_i)$. We consider two cases: either both a_i and b_i belong to the same weakly connected component of the infrastructure graph corresponding to σ_{i-1} or not. If both belong to the same weakly connected component, then consider the subsequence of σ_{i-1} of all flights in this component and let c be the destination of the last edge in this sequence. Now insert the flight (c, b_i) to the end of the subsequence (and shift all flights that appear later in σ_{i-1} one time slot back). Note that the length of the sequence increased by exactly one. Moreover, the first requirement is met since e_i only contains vertices from one weakly connected component and the new flight starts at the last node of the previous subsequence. For the second requirement, note that all nodes who have a pigeon tour to a_i in σ_{i-1} also have a pigeon route to c. Thus, the pigeon flight (c, b_i) ensures that node b_i learns at least as much information in σ_i as it does in OPT_i.

Now consider the case where $e_i = (a_i, b_i)$ connects two weakly connected components of the infrastructure graph corresponding to σ_{i-1}. We build σ_i as follows. First, let π_a and π_b be the subsequences of σ_{i-1} of all edges in the weakly connected components containing a_i and b_i, respectively. Let c be the destination of the last edge in π_a and let d be the first starting point of an edge in π_b. Now, we remove both π_a and π_b from σ_{i-1} and instead add the sequence $\pi_a \circ ((c, d)) \circ \pi_b$ at the end. Note that σ_i is longer than σ_{i-1} by exactly one and thus has size $i = |OPT_i|$. The first requirement is met by construction and it remains to show that the second requirement is also met. To this end, note that π_a and π_b are disjoint and hence no node in either weakly connected component has received any message from a node in the other component. Adding e_i in OPT hence only gives b_i information about all nodes that a_i knows about. Note that c has knowledge about all nodes appearing in π_a in σ_{i-1} and therefore also in τ_a. Moreover, there is a pigeon flight from d to b_i in τ_b and hence b_i has full knowledge about all nodes in the component of a_i in σ_i. Thus, the second requirement is met and this concludes the proof. □

Using these lemmas, we can now show the hardness of the Multihop-🕊 problem. Note that this result is incomparable to Theorem 4 and neither result immediately implies the other.

Theorem 6. *The decision variant of Multihop-🕊 is NP-hard.*

Proof. We reduce from Vertex Cover. To this end, let $(G = (V, E), k)$ be an instance of Vertex Cover and assume without loss of generality, that G is connected and contains at least one edge. Note that this implies that each node in V is incident to at least one edge in E. We will construct an equivalent instance $(G' = (V, E'), k')$ of Multihop-🕊 on the same node set V as follows. For each edge $\{u, v\} \in E$, we add the two edges (u, v) and (v, u) to E'.

Figure 5: An example instance of Vertex Cover on the left and the corresponding equivalent instance of Multihop-🕊 on the right.

To conclude the construction, we set $k' = n + k - 1$, where $n = |V|$. See Figure 5 for an example of the above construction.

Since the construction can clearly be computed in polynomial time, it only remains to prove that the constructed instance of Multihop-🕊 is a yes-instance if and only if the original instance of Vertex Cover is a yes-instance. For this, first assume that the original instance of Vertex Cover is a yes-instance and let K be a vertex cover of size at most k in G. Let $s \in K$ be an arbitrary node and let $\pi = (u_1, u_2, \ldots, u_{|K|})$ be an arbitrary ordering of the nodes in K where s is the last node and let $\tau = (v_1, v_2, \ldots, v_{n-|K|})$ be an arbitrary ordering of the nodes in $V \setminus K$.

We now construct a solution for the constructed instance of Multihop-🕊. First, we place one pigeon in each node v_i whose home is node v_{i+1} for all $i < n - |K|$. We also place one pigeon in $s = u_{|K|}$ whose home is v_1 and one pigeon in $v_{n-|K|}$ whose home is u_1. Finally, we place two pigeons in each node u_i with $1 \leq i < |K|$ whose home is u_{i+1}. Note that we placed exactly $n + |K| - 1 \leq n + k - 1 = k'$ pigeons. Next, we iteratively let one pigeon fly at a time as follows. We start with a pigeon flying from u_1 to u_2. Then a pigeon flying from u_2 to u_3 and so on. Once a pigeon arrives at $u_{|K|} = s$, the next pigeon flies from s to v_1. Afterwards, a pigeon flies from v_1 to v_2, then from v_2 to v_3 and so on until a pigeon arrives at $v_{n-|K|}$. Now, the pigeon starting in $v_{n-|K|}$ flies to u_1. From now on, the second pigeon in each node u_i iteratively flies to u_{i+1} for each $i < |K|$.

We will next show that each demand is satisfied by the above construction. To this end, we show that for each demand (u, v), there is a subsequence in $\pi \circ \tau \circ \pi$ which starts in u and ends in v, where $\circ$ is the concatenation operation. By construction, any subsequence corresponds to a sequence of pigeon flights that start and end at the respective nodes and are performed in the order given by the sequence. Thus, the information starting in node u is moved via pigeons iteratively and finally reaches v.

Note that each demand (u, v) implies that there is an edge $\{u, v\}$ in E and hence at least one of the two nodes is contained in K. If u is contained in K, then the sought-after subsequence starts in the first occurrence of π and ends in either τ or the second occurrence of π depending on whether v is contained in $V \setminus K$ or in K. If only v is contained in K, then the subsequence starts in τ and ends in the second occurrence of π. Since each node in K is contained in π and each node in $V \setminus K$ is contained in τ by definition, the subsequences exist and we have successfully constructed a solution.

In the other direction, assume that the constructed instance of Multihop-🕊 is a yes-instance. By Lemma 3, we may assume that a solution describes a walk through G' as G' is connected since G is connected. Let K be the set that includes all nodes in which at

$$\sum_{v \in V} x_v^i \leq 1 \qquad \forall i \in [2n] \quad (5)$$

$$\sum_{i<j \in [2n]} y_{u,v}^{i,j} = 1 \qquad \forall (u,v) \in E \quad (6)$$

$$2y_{u,v}^{i,j} \leq x_u^i + x_v^j \qquad \forall (u,v) \in E \text{ and } i < j \in [2n] \quad (7)$$

Figure 6: The constraints of the ILP for Multihop -🕊.

least two pigeons are initially placed in a solution as well as the the destination of the last pigeon flight. We will show that K is a vertex cover of size at most k. First, note that since each node in V is incident to at least one edge in E, it also holds for each node u that there is a demand of the form $(u, v) \in E'$. Hence, at least one pigeon needs to start in each node as otherwise this demand cannot be satisfied. Since the number of pigeons is at most $k' = n + k - 1$, it holds that $|K| \leq k - 1 + 1$ (the last $+1$ comes from the last destination added to K). Assume towards a contradiction that K is not a vertex cover. Then, there is an edge (u, v) such that neither u nor v is contained in K. Then, both of u and v appear once in the walk described by the assumed solution, as each node appearing twice either ends the walk or has two outgoing edges, which corresponds to two pigeons starting there initially.

Since both u and v appear once, one of the two occurs earlier than the other. Let us assume without loss of generality that u appears before v. Then, the constructed demand (v, u) (recall that $\{u, v\} \in E$ and hence $(u, v) \in E'$ and $(v, u) \in E'$) cannot be satisfied as whenever a pigeon leaves from v, no pigeon arrives at u to deliver the message, a contradiction to the assumption that we started with a solution. Thus, K is a vertex cover of size at most k and the original instance of Vertex Cover is a yes-instance. This concludes the proof. □

In the following theorem, we present an ILP formulation of the Multihop -🕊 problem.

Theorem 7. *There is an ILP with $O(n^4)$ variables and constraints for Multihop -🕊, where all variables are binary.*

Proof. Let $(G = (V, E), k)$ be an instance of Multihop -🕊. By Lemma 2, we may assume that G is connected as otherwise, we can solve each connected component independently. We construct an ILP with a binary variable x_v^i for each $v \in V$ and each $i \in [2n]$ and a binary variable $y_{u,v}^{i,j}$ for each edge $(u, v) \in A$ and each pair $i, j \in [2n]$ with $i < j$. The goal is to minimize $\sum_{v \in V} \sum_{i \in [2n]} x_v^i$ and a solution with k pigeons will exist if and only if there is a solution to the ILP where the goal value is at most $k + 1$. The constraints are listed in Figure 6.

Since the number of variables and constraints is clearly both in $O(n^4)$, it remains to show that the ILP is correct. To this end, we use Lemma 3 and Lemma 1. We may assume that a solution is described by a walk of length at most $2n-1$. We simply represent this walk by its at most $2n$ nodes. We set $x_v^i = 1$ if and only if v appears in position i in this sequence. Note that Constraint 5 ensures that at most one node appears in each position, and the goal function describes the total number of nodes appearing in the sequence. This number minus one is then the required number of pigeons as we claimed.

So it only remains to show that each demand is satisfied by a solution to the ILP. To this end, we use variable $y_{u,v}^{i,j}$ to denote that demand (u, v) is picked up at time step i and delivered in time step j. Constraint 6 ensures that each demand is satisfied in this way, and Constraint 7 ensures that the solution walk is at node u at time i and at node v at time j. Note that whenever $y_{u,v}^{i,j}$ is set to 1, then in order to satisfy Constraint 7, both x_u^i and x_v^j have to be set to 1 as well. This concludes the proof. □

4 Future Work

We view this paper as an "Introduction to Pigeon Post Theory" because it opens the door to many research avenues. First, throughout the paper, we focused on unbounded capacity pigeons, and the algorithmic and hardness results were derived under this assumption. An immediate question is how these results change when pigeons have bounded capacities, that is, when each pigeon can carry only a limited amount of information. For the special case of unit capacities, our hardness results for the 2-hop and multi-hop models seem to remain similar. Whether tight approximation results persist for bounded capacities, or whether new phenomena arise, remains an interesting open problem. Second, while the present work focuses on minimizing the number of pigeons, other cost measures are equally natural. For instance, we could consider every pigeon hop as a time step and try to reduce the time needed to deliver all messages. Additionally, one may consider the cost of transporting pigeons to their remote nodes, possibly allowing batching of deliveries to multiple nodes. Studying such cost-aware variants would further connect the model to classical network design and facility-location problems. Another promising direction concerns the dynamic demand. In this work, demands are assumed to be known in advance. Instead it would be interesting to study settings in which demands arrive over time, either according to a known process, stochastically, or adversarially. This naturally leads to online and competitive variants of the problem. Finally, it would be interesting to investigate distributed algorithms in the pigeon model. Since sending a message consumes a pigeon and effectively removes a directed edge from the infrastructure, one may ask how to perform distributed computation while minimizing pigeon usage, or how to compute without disconnecting the network.

Acknowledgments

This work was supported by the German Research Foundation (DFG), SPP 2378 (ReNO-2), 2025-2029.

References

[1] Miklós Ajtai. 1994. The complexity of the pigeonhole principle. *Combinatorica* 14, 4 (1994), 417–433.
[2] Chen Avin, Manya Ghobadi, Chen Griner, and Stefan Schmid. 2020. On the complexity of traffic traces and implications. *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 4, 1 (2020), 1–29.
[3] Chen Avin, Kaushik Mondal, and Stefan Schmid. 2020. Demand-aware network designs of bounded degree. *Distributed Computing* 33, 3 (2020), 311–325.
[4] Chen Avin and Stefan Schmid. 2025. Revolutionizing Datacenter Networks via Reconfigurable Topologies. *Commun. ACM* 68, 6 (2025), 44–53.
[5] BBC. 2011. SA pigeon 'faster than broadband'. https://web.archive.org/web/20110414225332/http://news.bbc.co.uk/2/hi/africa/8248056.stm Accessed: 24/12/2025.

[6] Martin J. Beckmann, C. B. McGuire, and Christopher B. Winsten. 1956. *Studies in the Economics of Transportation.* Yale University Press.

[7] Yossi Ben-Bassat. 2025. A New Israeli test confirms: PEI (Pigeon Enabled Internet) is FASTER than ADSL. https://web.archive.org/web/20080713090722/http://www.notes.co.il/benbasat/5240.asp Accessed: 24/12/2025.

[8] Samuel W Bent, Daniel D Sleator, and Robert E Tarjan. 1985. Biased search trees. *SIAM J. Comput.* 14, 3 (1985), 545–568.

[9] Zhi Cao and Eric Masanet. 2022. Material efficiency to tackle the sand crisis. *Nature Sustainability* 5, 5 (2022), 370–371.

[10] B. Carpenter and R. Hinden. [n. d.]. *Adaptation of RFC 1149 for IPv6.* RFC 6249. RFC Editor. doi:10.17487/RFC2549

[11] Benoit Crevier, Jean-François Cordeau, and Gilbert Laporte. 2007. The multi-depot vehicle routing problem with inter-depot routes. *European journal of operational research* 176, 2 (2007), 756–773.

[12] Erik D Demaine, Dion Harmon, John Iacono, and Mihai P a ˘ traşcu. 2007. Dynamic optimality—almost. *SIAM J. Comput.* 37, 1 (2007), 240–251.

[13] Josep Díaz, Jordi Petit, and Maria Serna. 2002. A survey of graph layout problems. *ACM Computing Surveys (CSUR)* 34, 3 (2002), 313–356.

[14] United Nations Environment Programme, Edgar E. Gutiérrez-Espeleta, Nyovani Madise, Ying Wang, Robert Watson, Tamiru A. Abiye, Ana Paula Aguiar, Peter Alexander, Barbara Amon, Apoorva Arya, Ghassem Asrar, Lindsay Beevers, Medani Bhandari, Meena Bohara, Gillian Bowser, David Broadstock, Monday Businge, Donovan Campbell, Kateřina Černý Pixová, Lynette Cheah Leila Dagher, Vassilis Daioglou, Jonathan Davies, Mark Elder, Parfait M. Eloundou-Enyegue, et al. 2025. Global Environment Outlook 7: A future we choose – Why investing in Earth now can lead to a trillion-dollar benefit for all. (2025).

[15] Shimon Even, Alon Itai, and Adi Shamir. 1975. On the complexity of time table and multi-commodity flow problems. In *16th annual symposium on foundations of computer science (FOCS)*. IEEE, 184–193.

[16] Nathan Farrington, George Porter, Sivasankar Radhakrishnan, Hamid Hajabdolali Bazzaz, Vikram Subramanya, Yeshaiahu Fainman, George Papen, and Amin Vahdat. 2010. Helios: a hybrid electrical/optical switch architecture for modular data centers. In *Proceedings of the ACM SIGCOMM 2010 Conference*. 339–350.

[17] Bernard Fortz and Mikkel Thorup. 2000. Internet traffic engineering by optimizing OSPF weights. In *Proceedings IEEE INFOCOM 2000. conference on computer communications. Nineteenth annual joint conference of the IEEE computer and communications societies (Cat. No. 00CH37064)*, Vol. 2. IEEE, 519–528.

[18] Monia Ghobadi, Ratul Mahajan, Amar Phanishayee, Nikhil Devanur, Janardhan Kulkarni, Gireeja Ranade, Pierre-Alexandre Blanche, Houman Rastegarfar, Madeleine Glick, and Daniel Kilper. 2016. Projector: Agile reconfigurable data center interconnect. In *Proceedings of the 2016 ACM SIGCOMM Conference*. 216–229.

[19] Graham Greene. 1958. *Our Man in Havana.* Heinemann.

[20] Lacy M Greening, Santanu S Dey, and Alan L Erera. 2025. Strengthening Dual Bounds for Multicommodity Capacitated Network Design with Unsplittable Flow Constraints. *arXiv preprint arXiv:2512.25018* (2025).

[21] Mohammad Taghi Hajiaghayi, Jeong Han Kim, Tom Leighton, and Harald Räcke. 2005. Oblivious routing in directed graphs with random demands. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing (STOC '05)*. ACM, 193–201. doi:10.1145/1060590.1060619

[22] The highly unofficial CPIP WG. 2001. *The highly unofficial CPIP WG.* Technical Report. University of Bergen. https://web.archive.org/web/20140215072548/http://www.blug.linux.no/rfc1149/ Retrieved 24/12/2025.

[23] Frank L. Hitchcock. 1941. The Distribution of a Product from Several Sources to Numerous Localities. *Journal of Mathematics and Physics* 20, 1–4 (1941), 224–230.

[24] Yanping Huang, Youlong Cheng, Ankur Bapna, Orhan Firat, Dehao Chen, Mia Chen, HyoukJoong Lee, Jiquan Ngiam, Quoc V Le, Yonghui Wu, et al. 2019. Gpipe: Efficient training of giant neural networks using pipeline parallelism. *Advances in neural information processing systems* 32 (2019).

[25] David A Huffman. 2007. A method for the construction of minimum-redundancy codes. *Proceedings of the IRE* 40, 9 (2007), 1098–1101.

[26] UNICEF IFAD et al. 2017. *The state of food security and nutrition in the world 2017.* FAO;.

[27] Leonard Kleinrock. 1976. *Queueing Systems, Volume II: Computer Applications.* John Wiley & Sons.

[28] Vanessa Lamb. 2023. Constructing the global sand crisis: Four reasons to interrogate crisis and scarcity in narrating extraction. *The Extractive Industries and Society* 15 (2023), 101282.

[29] Jean-Pierre Luminet. 2021. Closed Timelike Curves, Singularities and Causality: A Survey from Gödel to Chronological Protection. *Universe* 7, 1 (Jan. 2021), 12. doi:10.3390/universe7010012

[30] Carolina Elisabet Masin, Alejandra Duran, Cristina Susana Zalazar, and Maria Emilia Fernandez. 2024. Composting-vermicomposting of pigeon dropping waste: A contribution to the reduction of urban contamination. (2024).

[31] Alberto Medina, Nina Taft, Kavé Salamatian, Supratik Bhattacharyya, and Christophe Diot. 2002. Traffic matrix estimation: existing techniques and new directions. In *Proceedings of the ACM SIGCOMM 2002 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM '02)*. ACM, 161–174. doi:10.1145/633025.633041

[32] Deepak Narayanan, Mohammad Shoeybi, Jared Casper, Patrick LeGresley, Mostofa Patwary, Vijay Korthikanti, Dmitri Vainbrand, Prethvi Kashinkunti, Julie Bernauer, Bryan Catanzaro, et al. 2021. Efficient large-scale language model training on gpu clusters using megatron-lm. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*. 1–15.

[33] Pascal Peduzzi, Josefine Reimer Lynggaard, and Stephanie Chuah. 2022. Sand and sustainability: 10 strategic recommendations to avert a crisis. (2022).

[34] Leon Poutievski, Omid Mashayekhi, Joon Ong, Arjun Singh, Mukarram Tariq, Rui Wang, Jianan Zhang, Virginia Beauregard, Patrick Conner, Steve Gribble, et al. 2022. Jupiter evolving: transforming google's datacenter network via optical circuit switches and software-defined networking. In *Proceedings of the ACM SIGCOMM 2022 Conference*. 66–85.

[35] Hungry Beast Real Human Stories. 2010. Pigeons vs. Australian Internet. url: https://www.youtube.com/watch?v=ci2bFFGM8T8. Accessed: 24/12/2025.

[36] Grégory Salle. 2022. On the 'global sand crisis': From capital accumulation to ecological planning. (2022).

[37] Stefan Schmid, Chen Avin, Christian Scheideler, Michael Borokhovich, Bernhard Haeupler, and Zvi Lotker. 2015. Splaynet: Towards locally self-adjusting networks. *IEEE/ACM Transactions on Networking* 24, 3 (2015), 1421–1433.

[38] Claude E Shannon. 1949. Communication theory of secrecy systems. *The Bell system technical journal* 28, 4 (1949), 656–715.

[39] Sharanpreet Singh, Jaswinder Singh, Amandeep Kaur, Jagroop Kaur, Adarsh Pal Vig, and Sartaj Ahmad Bhat. 2019. Nutrient recovery from pigeon dropping by using exotic earthworm Eisenia fetida. *Sustainable Chemistry and Pharmacy* 12 (2019), 100126.

[40] Daniel Dominic Sleator and Robert Endre Tarjan. 1985. Self-adjusting binary search trees. *Journal of the ACM (JACM)* 32, 3 (1985), 652–686.

[41] Neal Stephenson. 1999. *Criptonomicon.* Avon.

[42] Google team. 2002. Google PigeonRank. url: http://www.google.com/technology/pigeonrank.html. Accessed: 24/12/2025.

[43] BBC News Technology. 2016. Pigeons vs. Australian Internet. url: Pigeon flies past broadband in data speed race. Accessed: 24/12/2025.

[44] Niren Tolsi. 2009. Winston the homing pigeon draws tweets of support. url: https://mg.co.za/article/2009-09-10-winston-the-homing-pigeon-draws-tweets-of-support/. Accessed: 24/12/2025.

[45] Paolo Toth and Daniele Vigo. 2002. *The vehicle routing problem.* SIAM.

[46] Gilbert S Vernam. 1926. Cipher printing telegraph systems: For secret wire and radio telegraphic communications. *Journal of the AIEE* 45, 2 (1926), 109–115.

[47] D. Waitzman. 1990. *A Standard for the Transmission of IP Datagrams on Avian Carriers.* RFC 1149. RFC Editor. doi:10.17487/RFC1149

[48] Wikipedia contributors. 2014. Google Pigeon Protocol. https://en.wikipedia.org/wiki/Google_Pigeon#:~:text=Google%20Pigeon%20is%20the%20code,local%20listings%20in%20a%20search. [Online; accessed 24-December-2025].

[49] Wikipedia contributors. 2025. Pigeon post. https://en.wikipedia.org/wiki/Pigeon_post?utm_source=chatgpt.com [Online; accessed 24-December-2025].