The Cardinality-Constrained Paths Problem: Multicast Data Routing in Heterogeneous Communication Networks

Alvaro Velasquez¹, Piotr Wojciechowski², K. Subramani³, Steven L. Drager⁴, Sumit Kumar Jha⁵

Department of Computer Science, University of Central Florida, Orlando, FL¹,⁵
LCSEE, West Virginia University, Morgantown, WV²,³
Information Directorate, Air Force Research Laboratory, Rome, NY⁴
{velasquez, jha}@cs.ucf.edu, pwojciec@mix.wvu.edu, ksmani@csee.wvu.edu, steven.drager@us.af.mil

Abstract—In this paper, we present two new problems and a theoretical framework that can be used to route information in heterogeneous communication networks. These problems are the cardinality-constrained and interval-constrained paths problems and they consist of finding paths in a network such that cardinality constraints on the number of nodes belonging to different sets of labels are satisfied. We propose a novel algorithm for finding said paths and demonstrate the effectiveness of our approach on networks of various sizes.

I. INTRODUCTION

One big challenge in securing today’s communication networks lies in their dynamic nature, with nodes constantly connecting to and disconnecting from other nodes. This is the case with mobile devices and increasingly so with vehicle-to-infrastructure (V2I) and vehicle-to-vehicle (V2V) network[5] communication, whose safety is a significant research focus of the United States Department of Transportation [7]. In these vehicular ad-hoc networks (VANETs), road-side units (RSUs) can send information to vehicles’ on-board units (OBUs), which can then route information to other OBUs as a multi-hop network [1]. The topology of vehicle-to-vehicle and vehicle-to-infrastructure networks changes quickly due to the high speeds of moving vehicles, which causes nodes to connect and disconnect from the VANET frequently [8]. The size of these VANETs can also be quite large in areas of heavy traffic and dense populations [9]. These challenges necessitate efficient routing algorithms.

In this paper, we solve the problem of finding paths in a heterogeneous network where there are upper-bound constraints on the number of nodes of each type that can be traversed. We call this the cardinality-constrained paths problem. We also explore the case where there are lower and upper bounds on the cardinality constraints. We call this the interval-constrained paths problem. Such constraints arise naturally in delay-tolerant networks (DTNs) where there is intermittent connectivity and interference or faults in the data are likely [10]. This is the case in VANETs and many ad-hoc networks. For dealing with these precarious domains, carry-and-forward routing schemes are often used [3]. These protocols store the message in intermediate nodes which then forward the message when a connection is available. The intermediate nodes can also act as verification nodes which ensure the integrity of the message being sent. Thus, placing a lower bound on the number of intermediate vertices is sensible for trustworthy communication.

We make the following contributions:

• The problems of finding cardinality-constrained paths (CCP) and interval-constrained paths (ICP) are introduced. These problems entail finding a path between two nodes in a network such that constraints on the number of nodes belonging to each label are satisfied. The CCP problem deals with upper bound constraints while the ICP problem pertains to upper and lower bounds.
• We establish the complexity of the CCP and ICP problems. more specifically, we show that they belong to the NP-Hard complexity class.
• For the CCP problem, we propose an efficient solution whose runtime is polynomial in the number of
nodes in the network when the number of labels is fixed. For the ICP problem, a similar polynomial-time algorithm is proposed when the network of interest is a directed acyclic graph (DAG).

II. Problem Definition and Complexity
In order to define the cardinality-constrained path problem, we need to formalize the notion of labeled graphs. In untrusted communication networks, this may simply consist of a graph \( G = (V, E) \), where \( V \) is the set of vertices and \( E \) is the graph’s adjacency matrix, such that every vertex \( v \in V \) belongs to either some set of trusted or authenticated nodes \( L_T \) or a set of untrusted nodes \( L_U \). The general case can be formalized as follows. A labeled graph \( G = (V, E, \{L_1, L_2, \ldots, L_p\}) \), \( |V| = n \), is a graph whose nodes belong to one of \( p \) disjoint labels \( L_1, L_2, \ldots, L_p \subseteq V \), such that \( L_1 \cup L_2 \cup \cdots \cup L_p = V \). A path \( \pi^{s\rightarrow t} = \{s, v_1, v_2, \ldots, v_k, t\} \subseteq V \) from source node \( s \in V \) to destination node \( t \in V \) is an ordered set defined by the vertices in the path such that \( (\pi^{s\rightarrow t}_i, \pi^{s\rightarrow t}_{i+1}) \in E \). We are interested in the number of nodes in the path that belong to each label. Let \( L_i = \pi^{s\rightarrow t} \cap L_i \) denote the set of nodes belonging to the path that have label \( i \).

In practice, these labels can be determined in many ways depending on the domain of the network. For network security purposes, authenticated nodes and compromised nodes could be two labels, with all other hosts belonging to a third label. In the vehicular domain, on-board units (OBUs) and road-side units (RSUs) have their respective labels, which could be augmented by adding labels to nodes that are vulnerable to malfunctions via structural damage or malicious hacking. There can also be labels denoting the performance of a node in terms of processing power or its reliability in terms of inherent fault-tolerance. In wireless sensor networks (WSNs) where resources are scarce [6], labels could be used to categorize different levels energy consumption or storage capacity.

**Cardinality-Constrained Paths Problem**

**Definition 1:** (Cardinality-Constrained Path) Given integers \( k_1, k_2, \ldots, k_p \), a labeled graph \( G = (V, E, \{L_1, \ldots, L_p\}) \), and source and destination nodes \( s, t \in V \), the cardinality-constrained path problem consists of finding a path \( \pi^{s\rightarrow t} = \{s, v_1, v_2, \ldots, v_k, t\} \) such that \( |L_i^1| \leq k_1, |L_i^2| \leq k_2, \ldots, |L_i^p| \leq k_p \).

We show, by reduction from 3SAT, that the cardinality-constrained path problem is **NP-Hard**. Consider an instance \( \Phi \) of 3SAT with \( n \) variables and \( m \) clauses. For each variable \( x_i \) in \( \Phi \), let \( k_{2i} \) be the number of clauses in which the literal \( x_i \) appears. Similarly, let \( k_{2i+1} \) be the number of clauses in which the literal \( \neg x_i \) appears. Associated with the variable \( x_i \), we create gadget \( VG_i \) shown in Figure 1.

![Fig. 1. Gadget VG_i for variable x_i. This gadget is a labeled graph G = (V, E, {L_1, L_2, L_{i+1}})](image)

In this gadget, the number of vertices along the top path is \( k_{2i} \). Each of these vertices is assigned the label \( L_{2i} \). Likewise, the number of vertices along the bottom path is \( k_{2i+1} \). Each of these vertices is assigned the label \( L_{2i+1} \).

For each clause \( \phi_j \) in \( \Phi \), we create the gadget \( CG_j \) shown in Figure 2 such that the labels of the three intermediate vertices depend on the literals in the clause. If \( x_i \) is in \( \phi_j \), then an intermediate vertex is assigned label \( L_{2i} \). Conversely, if \( \neg x_i \) is in \( \phi_j \), then an intermediate vertex is assigned label \( L_{2i+1} \).

![Fig. 2. Gadget for clause (x_i, ¬x_j, x_k). This gadget is a labeled graph G = (V, E, {L_1, L_2, L_{j+1}, L_k})](image)

We construct \( G \) by combining the gadgets as shown in Figure 3. Note that in \( G \), there are \((m + n + 1)\) vertices labeled \( L_1 \). Also note that all these vertices must appear on any path from \( s \) to \( t \). Thus, we set \( k_1 = (m + n + 1) \) to ensure that this limit is always satisfied. Now we need to show that there is a path of the desired type in \( G \) if and only if \( \Phi \) is satisfiable.

**Theorem 1:** Cardinality-Constrained Path is **NP-Hard**

**Proof:** Using the construction above, we will show that \( \Phi \) has a satisfying assignment if and only if there is a path \( \pi^{s\rightarrow t} \) in \( G \) such that \( |L_i^1| \leq k_1, \ldots, |L_i^p| \leq k_p \).

\( (\Rightarrow \) Suppose \( \Phi \) is satisfiable. For each variable \( x_i \) assigned a value of **true** in the satisfying assignment, the path will traverse the bottom path of gadget \( VG_i \). This uses \( k_{2i+1} \) vertices of label \( L_{2i+1} \). Similarly, for each variable \( x_i \) assigned a value of **false**, the path will traverse the top path of \( VG_i \). This uses \( k_{2i} \) vertices of label \( L_{2i} \).
For each clause $\phi_j$, the path will traverse an intermediate vertex of $CG_j$ corresponding to a true literal. Let us focus on the variable $x_i$. If $x_i$ is assigned a value of true, then the following conditions hold:

- We traverse $k_{2i+1}$ vertices of label $L_{2i+1}$ in $VG_i$.
- We do not traverse any vertices of label $L_{2i+1}$ in the clause gadgets since $\neg x_i$ has a value of false.
- We traverse at most $k_{2i}$ vertices of label $L_{2i}$ since the literal $x_i$ appears in only $k_{2i}$ clauses.

Thus, the satisfying instance of $\Phi$ yields a path $\pi^{s \rightarrow t}$ with $|L_1^s| \leq k_1, \ldots, |L_n^s| \leq k_n, |L_{2n+1}^t| \leq k_{2n+1}$. Assume $G$ has a path $\pi^{s \rightarrow t}$ with $|L_1^s| \leq k_1, \ldots, |L_n^s| \leq k_n$. For each variable $x_i$ in $\Phi$, assign a value of true if $\pi^{s \rightarrow t}$ traverses the bottom of gadget $VG_i$. If $\pi^{s \rightarrow t}$ traverses the top of gadget $VG_i$, then assign a value of false to $x_i$.

For each clause $\phi_j$ in $\Phi$, the following hold:

- If $\pi^{s \rightarrow t}$ traverses the intermediate vertex of $CG_j$ corresponding to the literal $x_i$, then $\pi^{s \rightarrow t}$ cannot traverse the top path of gadget $VG_i$. Otherwise, $\pi^{s \rightarrow t}$ would contain $k_{2i} + 1$ vertices of label $L_{2i}$, which violates the constraint $|L_{2i}| \leq k_{2i}$. Thus, $x_i$ must be assigned a value of true and clause $\phi_j$ is satisfied.
- If $\pi^{s \rightarrow t}$ traverses the intermediate vertex of $CG_j$ corresponding to the literal $\neg x_i$, then $\pi^{s \rightarrow t}$ cannot traverse the bottom path of gadget $VG_i$. Otherwise, $\pi^{s \rightarrow t}$ would contain $k_{2i} + 1$ vertices of label $L_{2i+1}$, which violates the constraint $|L_{2i+1}| \leq k_{2i+1}$. Thus, $x_i$ must be assigned a value of false and clause $\phi_j$ is satisfied.

It follows that a path $\pi^{s \rightarrow t}$ with $|L_1^s| \leq k_1, \ldots, |L_n^s| \leq k_n$ yields a satisfying assignment to $\Phi$. □

**Interval-Constrained Paths Problem**

**Definition 2:** (Interval-Constrained Path) Given integer pairs $(k_1, k_1'), (k_2, k_2'), \ldots, (k_p, k_p')$, a labeled graph $G = \langle V, E, \{L_1, \ldots, L_p\} \rangle$, and source and destination nodes $s, t \in V$, the interval-constrained path problem consists of finding a path $\pi^{s \rightarrow t} = \{s, v_1, v_2, \ldots, v_k, t\}$ such that $k_1 \leq |L_1^s| \leq k_1', k_2 \leq |L_2^s| \leq k_2', \ldots, k_p \leq |L_p^s| \leq k_p'$.

We demonstrate that this problem is NP-Hard in the simplest case where there is only a single label, i.e. $p = 1$. Naturally, the problem is at least as hard for $p > 1$.

**Theorem 2:** Interval-Constrained Path is NP-Hard

**Proof:** We will prove this result through a reduction from the Hamiltonian path problem. Let $G = \langle V, E \rangle$ be a directed graph with $n$ vertices and $m$ edges, and let $s$ and $t$ be vertices in $V$. We construct $G' = \langle V', E', \{L_1\} \rangle$ as shown in Figure 4. Assign the label $L_1$ to all vertices in $G'$, and set $k_1 = n$ and $k_1' = \lceil \frac{3n}{2} \rceil$. Then there is a Hamiltonian path in $G$ if and only if there is a path $\pi^{s \rightarrow t}$ in $G'$ such that $k_1 \leq |L_1^s| \leq k_1'$.

---

**III. METHODOLOGY**

In this section, we define the dynamic programming procedures used to solve the cardinality-constrained paths (CCP) and the interval-constrained paths (ICP) problems. In terms of complexity, the main difference between the two problems is that CCP is fixed-parameter tractable when the number of labels is fixed. Meanwhile, we have shown that ICP remains NP-Hard even with this fixed parameter. Consequently, we propose a polynomial-time algorithm for solving CCP in the general case for a fixed label size. The extension proposed for ICP is also solvable in polynomial time, but is only applicable to directed acyclic graphs.
Cardinality-Constrained Paths

We have devised an algorithm that is exponential only in the number of labels $p$ and polynomial in the number of vertices $|V|$ in the graph. Thus, our algorithm is tractable for a fixed number of labels. To the best of our knowledge, this is the first such algorithm to tackle this problem. Our proposed dynamic programming algorithm uses equations (1), (2), and (3) to populate the dynamic programming matrix, which is then used to infer the solution path through a standard backtracking search.

Let $u_{j}^{m}(t_{1},t_{2},...,t_{p-1})$ denote the minimal number of $p$-labeled nodes in a path from the source node $v_{1}$ to $v_{j}$ such that the number of $k$-labeled nodes is less than or equal to $t_{k}$, $1 \leq k \leq p-1$, and the path consists of at most $m$ edges. Then we can define a dynamic programming recurrence as shown in equations (1), (2), and (3), where $I_{k}(v_{i})$ is 1 if $v_{i} \in L_{k}$ and 0 otherwise. The initialization equation (1) states that the number of $p$-labeled nodes from the source node $v_{1}$ to itself using 0 edges is 1 if $v_{1} \in L_{p}$, 0 if $v_{1} \in L_{i}$, $i \neq p$ and its corresponding constraint parameter $t_{i}$ is greater than or equal to 1, and $\infty$ otherwise. Similarly, the number of $p$-labeled nodes from $v_{1}$ to any other vertex $v_{j}$ using 0 edges is initialized to $\infty$ (2) since no such path can exist. The recurrence (3) minimizes the cost (number of $p$-labeled nodes) of the path by verifying whether there is a path of smaller cost to $v_{j}$ and iterating through all incoming edges $(v_{i},v_{j}) \in E$ to determine whether the minimum-cost path to $v_{j}$ will yield a minimum-cost path to $v_{j}$.

**Theorem 3 (Optimal Substructure):** Let $\pi_{v_{1} \rightarrow v_{n}} = \pi_{v_{1} \rightarrow v_{j}} \cup \pi_{v_{j} \rightarrow v_{n}}$ be the path constructed from (1), (2), (3). Then $\pi_{v_{1} \rightarrow v_{j}}$ is the path from $v_{1}$ to $v_{j}$ with the minimum number of $p$-labeled nodes such that constraints $|L_{1}^{\pi_{1}}| \leq k_{1},...,|L_{p-1}^{\pi_{p-1}}| \leq k_{p-1}$ are satisfied.

**Proof:** Suppose a path $\sigma_{v_{1} \rightarrow v_{j}}$ from $v_{1}$ to $v_{j}$ exists such that $|L_{p}^{\sigma}| < u_{j}^{m-1}(t_{1},...,t_{p-1})$ and $|L_{1}^{\pi_{1}}| \leq t_{1},...,|L_{p-1}^{\pi_{p-1}}| \leq t_{p-1}$. Then there exists some path $\sigma' \subset \sigma_{v_{1} \rightarrow v_{j}}$, $\sigma' \cap \pi_{v_{1} \rightarrow v_{j}} = \emptyset$ not contained in $\pi_{v_{1} \rightarrow v_{j}}$. Let $v_{a}$ denote the last vertex in path $\sigma'$ and let $v' \in \sigma_{v_{1} \rightarrow v_{j}} \cap \pi_{v_{1} \rightarrow v_{j}}$ be the vertex connecting $\sigma'$ to $\pi_{v_{1} \rightarrow v_{j}}$ (i.e. $(v_{a},v') \in E$). $\pi_{v_{1} \rightarrow v'} \subseteq \pi_{v_{1} \rightarrow v_{j}}$ is the path from $v_{1}$ to $v'$ contained in $\pi_{v_{1} \rightarrow v_{j}}$. Similarly, let $\sigma_{v_{1} \rightarrow v_{j}} \setminus \sigma_{v_{1} \rightarrow v_{j}}' \subset \sigma_{v_{1} \rightarrow v_{j}}$ represent sub-paths of $\sigma_{v_{1} \rightarrow v_{j}}$. Two cases arise:

- $|\sigma_{v_{1} \rightarrow v_{j}} \cup \sigma_{v_{1} \rightarrow v_{j}}'| < |\pi_{v_{1} \rightarrow v_{j}}|$: When $m = |\sigma_{v_{1} \rightarrow v_{j}} \cup \sigma_{v_{1} \rightarrow v_{j}}'|$, the first parameter in the outer minimization of (3) will set $u_{j}^{m}(t_{1},...,t_{p-1}) \leq |L_{p}^{\sigma}|$. This result will then propagate to larger $m$.
- $|\sigma_{v_{1} \rightarrow v_{j}} \cup \sigma_{v_{1} \rightarrow v_{j}}'| \geq |\pi_{v_{1} \rightarrow v_{j}}|$: Since $(v_{a},v') \in E$, the inner minimization in (3) will check the value $u_{j}^{m}(t_{1},...,t_{p-1})$ when $m = |\sigma_{v_{1} \rightarrow v_{j}} \cup \sigma_{v_{1} \rightarrow v_{j}}'|$ and set $u_{j}^{m}(t_{1},...,t_{p-1}) \leq |L_{p}^{\sigma}|$.

It follows that no such path $\sigma_{v_{1} \rightarrow v_{j}}$ with $|L_{p}^{\sigma}| < u_{j}^{m-1}(t_{1},...,t_{p-1})$ exists. □

It follows from Theorem 3 that our algorithm returns the path from $v_{1}$ to $v_{n}$ with the minimum number of $p$-labeled nodes such that the constraints $|L_{1}^{k_{1}}| \leq k_{1},...,|L_{p}^{k_{p}}| \leq k_{p}$ are met. The proof of optimal substructure in Theorem 4 is similar and is therefore omitted.

Note that (3) iterates through paths of length $m$. The maximum length of a path in $G = (V,E,\{L_{1},...,L_{p}\})$ is $|V| - 1$ and there can be no path of length $k_{1} + k_{2} + \cdots + k_{p}$, that satisfies $|L_{1}^{k_{1}}| \leq k_{1},...,|L_{p}^{k_{p}}| \leq k_{p}$ since one of the constraints would be violated. Thus, $m = 0,1,\ldots,(\sum_{i=1}^{p} k_{i}) - 1$, leading to $\sum_{i=1}^{p} k_{i} \leq |V| - 1$ iterations in (3). In each iteration, (3) must also process every vertex and check the incoming edges for said vertex, yielding $|V|d_{avg}$ iterations, where $d_{avg}$ is the average degree of the vertices in $G$. Furthermore, for every constraint $k_{i} (i \neq p)$, we must iterate from 0 to $k_{i}$, yielding $\Pi_{i=1}^{p-1}(k_{i} + 1)$ iterations. This results in a runtime complexity of $O\left(|V|d_{avg} \left(\sum_{i=1}^{p} k_{i}\right) \left(\Pi_{i=1}^{p-1} k_{i}\right)\right)$.

Interval-Constrained Paths

Given a labeled graph $G = (V,E,\{L_{1},...,L_{p}\})$ and constraint pairs $(k_{1},k'_{1}),\ldots,(k_{p},k'_{p})$, we make the assumption that there is some $k_{i} = 0$. This assumption allows us to minimize over the set $L_{i}$ using a similar procedure to the one used to solve the cardinality-constrained paths problem. Without loss of generality, we assume that $k_{p} = 0$. Thus, the problem that we solve is a special class of the interval-constrained paths problem.

In order to satisfy the interval constraints introduced by the ICP problem, we add a simple modification to the pre-
vious approach. The only change is to equation (1), which is now replaced by (4). This change to the initialization equation causes the constraint parameters \( t_1, \ldots, t_{p-1} \) to be met exactly as opposed to as upper bounds. Let \( \mathcal{A}_i^n(t_1, t_2, \ldots, t_{p-1}) \) denote the minimum number of \( p \)-labeled nodes in a path from the source node \( v_1 \) to \( v_j \) such that the number of \( k \)-labeled nodes is exactly \( t_k, 1 \leq k \leq p - 1 \), and the path consists of at most \( n \) edges. The values of \( \mathcal{A}_i^n(t_1, t_2, \ldots, t_{p-1}) \) can be defined by equations (2), (3), (4). The addition of (4) causes the entries in the resulting dynamic programming matrix to match the constraint parameters \( t_1, \ldots, t_{p-1} \) with exactitude. It is easy to see that the runtime of the algorithm given by (2), (3), (4) is \( \mathcal{O}(|V|d_{\text{avg}}(\sum_{i=1}^{p} k_i)(\| \Pi_{i=1}^{p-1} k_i \|)) \). That is, the runtime is the same as that of (1), (2), (3) used to solve the constrained-paths problem.

\[
\mathcal{A}_i^n(t_1, \ldots, t_{p-1}) = \begin{cases} 
1 & (\mathcal{I}_p(v_1) = 1) \land \bigwedge_{i=1}^{p-1} (t_i = 0) \\
0 & \bigvee_{i=1}^{p-1} (t_i = 1 \land \mathcal{I}_i(v_1) = 1 \land \bigwedge_{j \neq i} t_j = 0) \\
\infty & \text{otherwise} 
\end{cases}
\]  

(4)

The solution to the interval-constrained paths (ICP) problem given by (2), (3), (4) yields a simple path when the network of interest is a directed acyclic graph. Otherwise, the solution may yield a path with a cycle as a solution. This is to be expected due to the complexity of the problem even when the number of labels is fixed. Therefore, this problem is believed to be intractable to solve in the general case.

**Theorem 4 (Optimal Substructure):** Let \( \pi^{v_1 \rightarrow v_n} = \pi^{v_1 \rightarrow v_j} \cup \pi^{v_j \rightarrow v_n} \) be the path constructed from (2), (3), (4). Then \( \pi^{v_1 \rightarrow v_n} \) is the path from \( v_1 \) to \( v_j \) with the minimum number of \( p \)-labeled nodes such that constraints \( |L_1^\pi| = k_1, \ldots, |L_{p-1}^\pi| = k_{p-1} \) are satisfied.

**IV. Experimental Results**

The average length of a path in random networks with varying degrees of connectivity has been studied in [4]. This length follows the equation \( \lambda(n, k) = (ln(n) - \gamma)/ln(k) + 1/2 \), where \( n \) is the number of vertices, \( \gamma \) is the Euler-Mascheroni constant, and \( k \) is the average degree of nodes in the graph. For large networks with one million nodes, this yields an average path length of 10 for low-connectivity networks (\( k = 4 \)) and 5 for the case of high-connectivity (\( k = 20 \)). We use \( \lambda(n, 4) \) and \( \lambda(n, 20) \) to define the constraints used in our runtime results (Fig. 5). These small path lengths have been evidenced in real complex networks. In an analysis of the Facebook social network with 721 million user nodes, it was found that the average path length between two nodes was just 4.74

![Algorithm Execution Time](image)

Fig. 5. Running time, in milliseconds, of our algorithm on randomized networks of various sizes. We look at networks whose average degree and path length are \( k = 4 \) and \( \lambda(n, 4) \), respectively, and \( k = 20, \lambda(n, 20) \), where \( n \) is the number of nodes in the network.

[2]. This leads us to believe that cardinality-constrained paths in \( G = (V,E,\{L_1, \ldots, L_p\}) \) have the property \( |\pi^{v_1 \rightarrow v_n}| \leq \sum_{i=1}^{p} k_i << |V| \) in the general case.

**REFERENCES**


