COMPRESSO: Latency-aware Transmission of Compressed IoT Measurement Data over SDN

Wendi Feng *Member, IEEE*, Xintan Dou, Amirhosein Taherkordi *Member, IEEE*, Bo Cheng *Member, IEEE*, and Wei Zhang *Member, IEEE*

***Abstract*—Measurement data obtained from "things" in the Internet of Things (IoT) faces challenges in efficient transmission due to the low-bandwidth data transmission link. We observe that measurement data are fixed in size and format, and low-entropy in the time domain, indicating that compression can be benefited. Rather than employing a single compression algorithm as advocated in existing literature, we argue that optimal transmission can be achieved by jointly considering compression overheads and network status, where software-defined networking (SDN) is employed to enforce network statistics and packet forwarding. This paper presents a new paradigm that achieves optimal transmission of SDN-empowered compressed measurement data. We formulate the problem as an optimization problem and prove its non-polynomial hardness time complexity. Due to this complexity, we introduce COMPRESSO, a heuristic algorithm that efficiently solves the problem. We conduct rigorous simulations, and the results demonstrate the efficiency of the new paradigm and COMPRESSO, *i.e.*, attaining comparable performance to the optimal solution with 50% time usage reduction.**

***Index Terms*—Internet of Things, measurement data, transmission of compressed data , quality of service.**

I. INTRODUCTION

EVERY morning, You go to work and park your car at a curbside, and then, as part of the smart city, the traffic control officer drives a car (or simply an autopilot car) equipped with 360° cameras to identify parking violations automatically. These daily routines are all made possible by the Internet of Things (IoT), a network of physical devices that can collect and exchange data [1]. IoT devices collects various types of *measurement data* (with equipped sensors), *e.g.*, temperature, motion, and location. The data is then sent to a central hub for analysis, and the hub can in turn send *control messages* to the devices, manipulating their behaviors.

For achieving the desired quality of services (QoS), it is important to ensure the efficient transmission of measurement data and control messages [2]. However, IoT transmission networks have limited bandwidth (in the order of megabits or even kilobits per second) due to the low power requirements [3], [4]. Although control messages are typically lean, measurement data can be frequently collected and voluminous. Hence, transmitting measurement data can lead to potential network congestion [5] and subsequent degradation of services.

Existing transmission optimization techniques can be classified into two categories: *reducing the data size* [6], [7] and *improving the network performance* [8]. We argue that transmission of measurement data is suitable to be optimized by data size reduction through an application-agnostic approach: *compression*. This assertion is based on our observation that measurement data are fixed in size and format, having low entropy in the time domain, hence suitable for data compression (see Section II). Furthermore, network performance can be improved by selecting the "most suitable" path to avoid network issues. Thanks to the programmability provided by software-defined networking (SDN), monitoring network conditions and enforcing packet transmissions (*i.e.*, choosing paths for packets) can be achieved flexibly and much finer-grained [9]. Both data compression for transmission and path selection are well-known concepts in the computer network community [7], [10]. However, *previous work has focused on leveraging a single compression algorithm without considering the impact of computational overhead introduced by compression, network status, and differences of various compression algorithms to achieve optimal measurement data transmission*.

This paper proposes an SDN-empowered solution for efficient transmission of compressed measurement data in IoT transmission networks. We present the design of the *SDN-empowered Transmission Of Compressed mEasurement Data SYStem* (STOCED-SYS) that judiciously considers the joint optimization of end-host computational and network resources (detailed in Section III). We formulate the *Optimal SDN-empowered Transmission Of Compressed mEasurement Data* (O-STOCED) problem, which aims to minimize the average latency by selecting compression algorithms from various compression "ratios – overhead" combinations and network paths across the overlayed IoT transmission SDN. Our mathematical analysis reveals that the O-STOCED problem is non-polynomial hard (NP-hard). Hence, we propose an efficient heuristic algorithm called COMPRESSO. Simulation results demonstrate that the proposed system can achieve up to 1000× average latency reduction. Besides, the COMPRESSO algorithm achieves comparable performance to the optimal solution while significantly reducing the execution time (over 50%) on real-world topologies of various sizes.

To summarize, our contribution is three-fold, as follows.

This work is supported in part by National Natural Science Foundation of China under grant 62402049, National Key R&D Program of China under grant 2022YFC3320903, R&D Program of Beijing Municipal Education Commission under grant KM202311232005, and Open Foundation of State key Laboratory of Networking and Switching Technology (Beijing University of Posts and Telecommunications) under grant SKLNST-2023-1-01.

Wendi Feng is with College of Computer Science, Beijing Information Science and Technology University (BISTU) and State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications (BUPT).

Xintan Dou and Wei Zhang are with College of Computer Science, BISTU.

Amirhosein Taherkordi is with Dept. of Informatics, University of Oslo.

Bo Cheng is with State Key Laboratory of Networking and Switching Technology, BUPT.

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2025.3543479

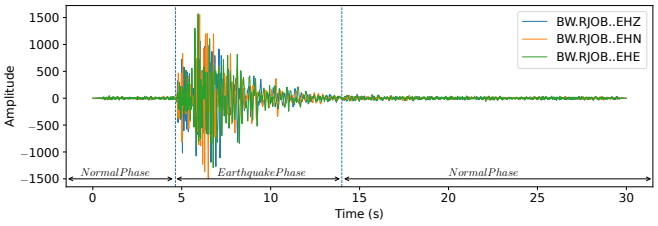


Fig. 1: Seismic data example (data from [15]).

- We identify the inadequacy of existing research that overlooks the comprehensive consideration of compression algorithms, compression overhead, and network conditions for optimal measurement data transmission.
- We formally define the problem of optimal transmission of SDN-empowered compressed measurement data, demonstrate its NP-hardness complexity, and introduce an efficient heuristic algorithm as a solution.
- Rigorous simulations are conducted to evaluate the effectiveness of the COMPRESSO algorithm.

The remainder of the paper is organized as follows. Section II introduces background knowledge and motivates the O-STOCED problem using examples. Section III describes the system setting. Section IV formulates the O-STOCED problem. In Section V, we prove the time complexity and present the COMPRESSO algorithm. To show the effectiveness of COMPRESSO, Section VI evaluates the COMPRESSO algorithm. Next, we survey related work in VII. Finally, Section VIII concludes the paper.

II. BACKGROUND AND MOTIVATION

A plethora of work has been proposed to reduce data transmission latency. However, we argue our scenario differs from existing ones. In this section, we motivate the problem by jointly considering the character of data and system platforms.

A. Stability of Measurement Data Size and Collection Rate

Measurement data is collected by sensors and devices, which convert analog signals into digital data at a fixed rate and using specific sampling algorithms. Data elements within a particular data type share a common format, and multiple data types can be nested within a monolithic data block. The structure of these data blocks remains consistent, resulting in a fixed data size [11]. Moreover, data collectors typically retrieve measurement data at a specific rate, leading to a fixed data collection rate. For example, computer systems collect and report system performance metrics every second (according to configuration). **These observations suggest that traditional queuing theory-based approaches [12]–[14] are too complex and unnecessary in this scenario.**

B. Stability and Compressibility of Measurement Data Values

Measurement data is generally stable but can experience significant changes under certain circumstances. As demonstrated in a seismic data example obtained from ObsPy [15] in Figure 1, seismic data collected from seismic data collectors shows steady or limited fluctuations in the absence of earthquakes. **This stability indicates measurement data has low entropy in the time domain, making it suitable for compression and enabling efficient data transmission over limited IoT transmission network bandwidth.**

C. Unpredictability and Lossless Requirements of Measurement Data

Identifying data patterns and predicting succeeding data points is often challenging or impossible for measurement data [16]. Sudden changes, such as earthquake spikes, occur without forecastable signs and are rare compared to data under common conditions [16]. Statistical approaches, including lossy compression, may regard these spikes as noise and omit them, leading to incorrect measurements and potentially missing critical alerts. Therefore, **most machine learning (ML)-based approaches and lossy compression algorithms cannot effectively represent measurement data.**

D. Merits of SDN and Multi-path Technologies

SDN has emerged as a key enabler in networking, providing programmability and fine-grained data transmission management over the past decade. With SDN, network entities can obtain a global view of the network, allowing them to calculate available capacity and congestion for each network path. Consequently, multi-path transport protocols (*e.g.*, multi-path TCP [17], multi-path QUIC [18], and multi-path RDMA [19], [20]) are widely adopted. This allows communication entities to select the most suitable path for data transmission based on network information provided by the SDN control plane. This capability reduces average data transmission latency, improves network link utilization, and helps avoid network congestion.

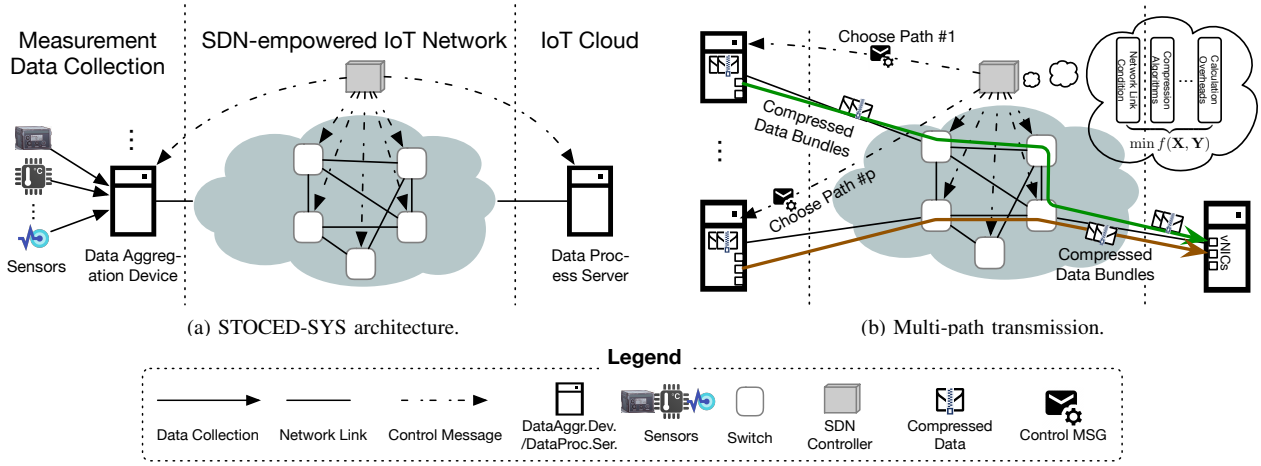
III. DESIGN OF STOCED-SYS

Motivated by the need for efficient measurement data transmission, we present the STOCED-SYS. This section begins with an overview of STOCED-SYS and then introduces the problem of *optimal transmission of SDN-empowered compressed measurement data* (O-STOCED).

A. System Overview

From a bird's eyes view, STOCED-SYS leverages SDN for monitoring network status and guiding end-host data transmission across multiple paths. As illustrated in Figure 2a, the STOCED-SYS consists of three main parts, *viz.*, **i)** *measurement data collection*, **ii)** *SDN-empowered IoT network*, and **iii)** *IoT cloud*. In part **i)**, dedicated *sensors* collect measurement data, converting analog signals to digital data. Collected *atomic measurement data* (*i.e.*, one single data) is directed to a *data aggregation device*. Next, a number of atomic measurement data is *batched* as a *data bundle* on the data aggregation device, which is then compressed by an compression algorithm at a particular compression ratio to be efficiently transmitted. Then, the aggregation device chooses a suitable path to transmit data bundles across the SDN-empowered IoT network (*i.e.*, part **ii)**) to the *data process server* that is reside in the IoT cloud (*i.e.*, part **iii)**). Multiple aggregation servers and processing servers can exist, but we only consider the *many-to-one* scenario in this paper, where measurement data is collected from multiple sources and sent to one processing server (see Figure 2b). Transmission between data aggregation

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2025.3543479



(a) STOCED-SYS architecture.

(b) Multi-path transmission.

Fig. 2: STOCED-SYS Overview.

devices and the data process server employs the multipath QUIC protocol [18], a multipath, reliable, low-latency transport protocol, to tackle the packet loss issue. The SDN-empowered IoT network is built as an *overlay* SDN on the underlying IoT network infrastructure. This enables us to apply traffic engineering policies for efficient flow steering across the network. The underlying network can either be a physical infrastructure, similar to Google's B4 [21] or Microsoft's SWAN [10] networks, or an existing overlay network built on top of a cloud platform, in which case we can leverage the network infrastructure provided by the cloud. To build an overlay SDN, we can leverage the Segment Routing [22] protocol, which can efficiently employ *labels* to enforce data being transmitted through selected paths. The *controller* of the SDN-empowered IoT network monitors the network status, calculates, and tells the best paths for transmitting data bundles for data aggregation devices. *Control messages* between the controller, network nodes, data aggregation devices, and data process servers are sent across a reserved virtual dedicated control network akin to SWAN [10]. On each data aggregation device and data process server, we create virtual network interface cards (vNICs), with which we pair each vNIC with an IP address corresponding to one path.

B. Trade-off between Compression Algorithms and Ratios

The information theory establishes a theoretical upper bound for the compression ratio [23], but achieving it contributes to high computational complexity [24]. For example, literature [24] reports that the Brotli algorithm achieves 98.3 MB/s compression throughput when the compression ratio is 3.381, but the performance drops significantly to 0.5 MB/s when the ratio is 4.347. Moreover, data block sizes also play a key role in terms of decompression performance, and a larger data block size contributes to a higher compression ratio but lower encoding/decoding performance [24]. Hence, the trade-off between compression overhead and benefits (*i.e.*, the reduction of transmitted data) for optimal data transmission requires a judicious consideration.

C. Performance Metrics in Network Transmission Systems

Modern networks employ techniques like *batching* to improve bandwidth and average latency [25]. Batching consolidates multiple packets and sends them in bulk, effectively reducing the latency required for each packet. However, batch size has an optimal setting, *e.g.*, the recommended batch size number (*i.e.*, bundle size) for DPDK is 32 or 64 [26]. Furthermore, the SDN-empowered IoT network may have multiple paths with varying *bandwidths* and *latencies*. Paths with higher bandwidth can transmit more data at a short period of time, but if leveraging the larger bandwidth and sending a large bulk of data simultaneously, preceding measurement data have to wait a long time before sending, which diminishes the timeliness of the data. Also, if the latency is low, batching may not be needed. Hence, selecting the appropriate compression algorithm, ratio, data batch size, and path is crucial to achieving optimal transmission performance.

D. Problem Statement

Inspired by the aforementioned considerations, we propose to leverage the data compression mechanism, batching, SDN, and multi-path transmission for efficient measurement data transmission. Important factors include the size of the bundles on each path, the compression algorithm and ratio used for each bundle, capacity and latency of each path. The challenge is: *with this information, how do we judiciously select one particular compression algorithm and ratio, the size of measurement data bundles, and a proper path to achieve the optimal SDN-empowered transmission of compressed measurement data?* We call the problem the *optimal transmission of SDN-empowered compressed measurement data* (O-STOCED) problem. Consequently, a systematic decision by jointly considering the data characteristics, compression algorithm features, and network conditions should be required. We abstract the problem out as a mathematical problem in Section IV and solve it in Section V.

TABLE I: Notation definitions.

Notation	Definition
V	SDN-empowered IoT network node set. $V=\{v_1, v_2, \dots, v_N\}$.
E	SDN-empowered IoT network edge set. $E=\{e_1, e_2, \dots, e_M\}$.
P^i	Path set of the i^{th} bundle (flow). $P^i=\{p_1^i, p_2^i, \dots\}$.
A	Set of compression algorithms. $A=\{a_1, a_2, \dots, a_H\}$.
$\mathbf{X}$	Mapping of bundle and compression algorithms, where $x_{ij}=1$ indicates the i^{th} bundle and compression algorithm a_j is selected. $\mathbf{X}=\{(i,j)\in[1,N-1]\times[1,H] \mid x_{ij}=1\}$.
α	Number of paths for each bundle.
b	Number of atomic measurement data in a data bundle.
$\mathbf{Y}$	Mapping of bundle and path selection, where $y_{il}=1$ indicates that path p_l is selected for the i^{th} bundle. $\mathbf{Y}=\{(i,l)\in[1,N-1]\times[1,\alpha(N-1)] \mid y_{il}=1\}$.
$r(\cdot)$	Compression ratio function.
$o(\cdot)$	Overhead function of compression and decompression.
$c(\cdot)$	Link and path capacity function.
w	Window size for (de)compressing data.
clk	CPU clock speed of communications entities in $\mathcal{E}_s$ and $\mathcal{E}_r$.
γ	Network link capacity.
ι^{link}	Network link tranmission latency.
ι^{node}	Network node processing latency.
λ	Size of each atomic measurement data.
δ	Time interval of two consecutive atomic measurement data.

IV. Formulation

This section mathematically formulates the O-STOCED problem. We first provide a formal description of the problem, present the metrics and constraints considered, and finally formulate the problem as an *integer programming* problem.

A. System Description

A STOCED-SYS comprises multiple identical *communication entities* $\{\mathcal{E}_s, \mathcal{E}_r\}$ and an SDN-empowered IoT network $G=(V,E)$. Here, $\mathcal{E}_s$ and $\mathcal{E}_r$ represent the sets of sender and receiver entities, respectively. We assume identical clock speeds for all devices, denoted by clk. $V=\{v_1, v_2, \dots, v_N\}$ represents the node set of the (SDN-empowered IoT) network, and $E=\{e_1, e_2, \dots, e_M\}$ is the set of network links. The capacity of each link is γ. Each network node hosts a data aggregation/process device/server, while $\mathcal{E}_r$ contains only one data process server due to the many-to-one model. Let $P^i=\{p_1^i, p_2^i, \dots\}$ be the set of paths for transmitting data from node v_i to the destination, where v_i is not connected to the process server. The capacity function $c(\cdot)$ of path p_l is defined as $c(p_l)=\min(\{c(e_k) \mid e_k\in p_l\})$. The transmission latency of each network hop and the processing latency of each network node are represented by constants ι^{link} and ι^{node}, respectively. Let λ be the size of an atomic measurement data, and δ be the time interval between consecutive atomic measurement data. $A=\{a_1, a_2, \dots, a_H\}$ represents the set of compression algorithms[1]. The compression ratio function $r(\cdot)$ returns the compression ratio of an algorithm. Compression algorithms operate on data within *windows* of size w. $o(\cdot)$ represent the overhead function of compression and decompression (*i.e.*, the # of CPU cycles). Measurement data bundles from different entities of $\mathcal{E}_s$ are injected into the network simultaneously. A network has $N-1$ data aggregation devices (*i.e.*, one node is the recevier), and hence, $N-1$ bundles are transmitted. Each flow between an entity in $\mathcal{E}_s$ and the process server may have multiple paths, but we only consider at most α paths in our formulation. Let $\mathbf{X}=\{x_{ij}\}$ be the matrix of bundle and compression algorithm selections. $x_{ij}=1$ indicates that the i^{th} bundle is compressed using compression algorithm a_j. Let b be the bundle size. Let $\mathbf{Y}=\{y_{il}\}$ be the matrix of bundle and path selections. $y_{il}=1$ indicates that the i^{th} bundle is transmitted along path p_l. All notations used are defined in TABLE I.

[1]For brevity, we consider an algorithm that compresses data at varying ratios as distinct algorithms.

B. Performance Metrics

1) Latency of Atomic Data Preparation

Atomic measurement data are collected at a constant time interval δ. Since atomic data is bundled to send, predecessor atomic data have to wait until all b atomic data is collected that can be transmitted. Hence, the atomic data preparation latency can be written as

$$l_i^{ATM}=(b-1)\,\delta. \tag{1}$$

2) Latency of Compressing Data Bundles

In the STOCED-SYS, atomic measurement data is first bundled into a measurement data bundle, and then, the sender communication entity compresses the bundle with the selected compression algorithm and ratio. The compressed bundle is then transmitted over the link. On the receiver side, the bundle is decompressed. Hence, the data processing latency can be written as

$$l_i^C=\frac{\frac{b\lambda}{w}\sum_{j=1}^{H}(o(a_j)x_{ij})}{clk}, \tag{2}$$

where $\sum_{j=1}^{H}(o(a_j)x_{ij})$ represents the summation of compression and decompression overheads for each compression window under the selection of $\mathbf{X}$. $\frac{b\lambda}{w}$ is the number of compression windows.

3) Latency of Transmitting Data Bundles

Each data bundle is transmitted over the SDN-empowered IoT network along a path selected from $\mathbf{Y}$. Therefore, the transmission time for the bundled data across the network is the size of the bundled data over the path's throughput plus the path latency. Thus, the compressed bundle transmission latency is given by

$$l_i^{TX}=\frac{b\lambda\sum_{j=1}^{H}(r(a_j)x_{ij})}{\sum_{l=1}^{\alpha(N-1)}(c(p_l)y_{il})}+\left(\sum_{e_k\in p_l}\iota^{link}+\sum_{v_k\in p_l}\iota^{node}\right). \tag{3}$$

It should be noted that latency resulting from the packet loss is well studied in [27], which is not our focus. Hence, we merge it into ι^{link}. To avoid the issue of variable $\mathbf{Y}$ presenting in the divisor, which is unsupported by most optimization solvers [28], [29], we reformulate Equation (3) by the following procedures. First, we matrixify the expression. Let $R=\{r_j \mid r_j=r(a_j), \forall j\in[1,H]\}$ be the compression algorithm compression ratio vector and $C^P=$

$\left\{c_l^P \mid c_l^P = c(p_l), \forall l \in [1, \alpha(N-1)]\right\}$ be the path capacity vector. Then, let $B = \mathbf{X}R^\intercal$ and $D = \mathbf{Y}C^{P\intercal}$. With these definitions, $\sum_{j=1}^{H}(r(a_j)x_{ij})$ can be written as B_i, and $\sum_{l=1}^{\alpha(N-1)}(c(p_l)y_{il})$ can be written as D_i. Thus,

$$\sum_{i=1}^{N} \frac{\sum_{j=1}^{H}(r(a_j)x_{ij})}{\sum_{l=1}^{\alpha(N-1)}(c(p_l)y_{il})}$$

can be represented as $(\mathbf{X}R^\intercal)(\mathbf{Y}\frac{1}{C^{P\intercal}})^\intercal$. Next, we expand this matrix multiplication form as

$$\begin{aligned} l_i^{TX} = b\lambda \sum_{j=1}^{H}(r(a_j)x_{ij}) \sum_{l=1}^{\alpha(N-1)} \left(\frac{1}{c(p_l)} y_{li}\right) \\ + \sum_{e_k \in p_l} \iota^{link} + \sum_{v_k \in p_l} \iota^{node}. \end{aligned} \tag{4}$$

C. Constraints

1) Data Bundle Compression Constraint

Each bundle should select exactly one compression algorithm at a time. This constraint can be formally written as

$$\sum_{j=1}^{H} x_{ij} = 1. \tag{5}$$

2) Path Selection Constraint

Similarly, each bundle should select one suitable path from the SDN-empowered IoT network for transmission. This can be mathematically expressed as

$$\sum_{l=1}^{\alpha(N-1)} y_{il} = 1, \tag{6}$$

where $(N-1)\alpha$ represents the total number of path candidates because the SDN controller calculates α paths for each bundle.

D. Objective Function

The problem of optimal transmission of SDN-empowered compressed measurement data aims at finding the optimal compression-based measurement data transmission strategy. Therefore, the objective for the problem is to reduce all aforementioned latencies under the STOCED-SYS (*i.e.*, finding the minimum latency). Consequently, our objective function is mathematically written as

$$obj = \sum_{i=1}^{N-1} \left(l_i^{ATM} + l_i^{C} + l_i^{TX}\right). \tag{7}$$

E. Problem Formulation

The goal of the problem of optimal transmission of SDN-empowered compressed measurement data is to minimize the overall latency of transmitting measurement data from $N-1$ data aggregation devices simultaneously across an SDN-empowered IoT network over multiple paths. This is achieved by judiciously selecting compression algorithms and data bundles from $\mathbf{X}$ and selecting the path for each bundle from $\mathbf{Y}$. Consequently, we formulate the problem of optimal transmission of SDN-empowered compressed measurement data (Problem P) as follows:

$$\begin{aligned} &\min_{\mathbf{X},\mathbf{Y}} \ obj \\ &\text{s.t. } (5)(6) \\ &\quad x_{ij} \in \{0,1\}, \forall i \in [1, N-1], \forall j \in [1, H], \\ &\quad y_{il} \in \{0,1\}, \forall i \in [1, N-1], \forall l \in [1, \alpha(N-1)], \end{aligned} \tag{P}$$

where w, clk, α, λ, and δ are constants. $\{x_{ij}\}$, $\{y_{il}\}$ are designed binary variables. Since the variables are integers, the problem is an *integer programming* problem.

V. Solution

We first prove that the problem optimal transmission of SDN-empowered compressed measurement data is NP-hard. Then, we present an efficient heuristic algorithm to solve it.

A. Complexity Analysis of the Problem

Proposition 1. *For a special case of Problem P with the following three conditions: (1) network link transmission latency and network node processing latency are "far less" than the bundled data transmission latency, (2) $c(\cdot)$ is a constant function, and (3) the bundle size is fixed at $b = 1$. The O-STOCED problem is NP-hard.*

Proof. We prove NP-hard complexity by showing that the special case of Problem P under the listed conditions in Proposition 1 is equivalent to the well-known NP-hard problem, the *Generalized Assignment Problem* (GAP). GAP aims to minimize the overall costs of assigning n *tasks* to m *workers*, with each task assigned to exactly one worker subject to the workers' capacity limitations. GAP can be mathematically formulated as follows:

$$\begin{aligned} \min_{X} \quad & \sum_{j=1}^{m}\sum_{i=1}^{n} c_{ij}x_{ij} \\ \text{s.t.} \quad & \sum_{i=1}^{n} a_{ij}x_{ij} \le C_j, \forall j \in [1, m], \\ & \sum_{j=1}^{m} x_{ij} = 1, \forall i \in [1, n], \\ & x_{ij} \in \{0,1\}, \forall i \in [1, n], \forall j \in [1, m], \end{aligned} \tag{8}$$

where c_{ij} denotes the cost of assigning task i to worker j, and a_{ij} indicates the capability required for executing task i on worker j. C_j is the available capability of worker j. x_{ij} is a binary variable indicating if task i is assigned to worker j (1 for yes, 0 for no). It has been proven that the GAP problem is NP-hard [30].

For the special case of Equation (P) under conditions (1)-(2), the equivalence of Equation (P) and the GAP problem can be established. Given condition (1), $\sum_{e_k \in p_l} \iota^{link} + \sum_{v_k \in p_l} \iota^{node}$ can be omitted in Equation (3). Given condition (2), $c(p_l)$ can be replaced by a constant ψ outside of $\sum$. Hence, we can rewrite Equation (3) as

$$l_i^{TX} = \frac{b\lambda \sum_{j=1}^{H} (r(a_i)x_{ij})}{\psi \sum_{l=1}^{\alpha(N-1)} y_{il}}. \tag{9}$$

Due to the path selection constraint defined in Equation (6), Equation (9) can be written as

$$l_i^{TX} = \frac{b\lambda \sum_{j=1}^{H} (r(a_i)x_{ij})}{\psi}. \tag{10}$$

Given condition (3), the objective function becomes

$$obj = \frac{\lambda}{w \times clk} \sum_{i=1}^{N-1} \sum_{j=1}^{H} (o(a_j)x_{ij}) + \frac{\lambda}{\psi} \sum_{i=1}^{N-1} \sum_{j=1}^{H} (r(a_j)x_{ij}). \tag{11}$$

Let $obj_1 = \frac{\lambda}{w \times clk} \sum_{i=1}^{N-1} \sum_{j=1}^{H} (o(a_j)x_{ij})$ and $obj_2 = \frac{\lambda}{\psi} \sum_{i=1}^{N-1} \sum_{j=1}^{H} (r(a_j)x_{ij})$. Thereby, Problem P can be reformulated as

$$\begin{aligned} \min_{\mathbf{X}} \quad & (obj_1 + obj_2) \\ \text{s.t.} \quad & (5)(6), \\ & x_{ij} \in \{0,1\}, \forall i \in [1, N-1], \forall j \in [1, H]. \end{aligned} \tag{P'}$$

Finding the solution of $\{x_{ij}\}$ to achieve the minimum objective value defined by Equation (11) becomes equivalent to achieving $\min_{\mathbf{X}} obj_1 + \min_{\mathbf{X}} obj_2$. We divide the special case into two problems, *viz.*, Problem P1 and Problem P2, as follows.

$$\begin{aligned} \min_{\mathbf{X}} \quad & obj_1 \\ \text{s.t.} \quad & (5)(6). \end{aligned} \tag{P1}$$

$$\begin{aligned} \min_{\mathbf{X}} \quad & obj_2 \\ \text{s.t.} \quad & (5)(6). \end{aligned} \tag{P2}$$

For each problem in Problem P1 and Problem P2, the goal is to minimize the latency (*i.e.*, latency of compressing and transmitting data bundles) introduced by compression and data transmission of each bundle. In each problem, the mapping between the i^{th} bundle and compression algorithm a_j can be considered as task j and worker m. Under this construction, we can prove that the solution of minimum overall latency of transmission of compressed measurement data exists if and only if the optimal solution of the GAP problem exists. This construction can be conducted in polynomial time. Since GAP is NP-hard, Problem P1 or Problem P2 are also NP-hard. Hence, combining Problem P1 and Problem P2, Problem P' is NP-hard. Problem P' is a special case of Problem P. □

Therefore, we can conclude:

Theorem 1. *The problem optimal transmission of SDN-empowered compressed measurement data is NP-hard.*

Algorithm 1: The COMPRESSO algorithm.

Input: $G = (V, E)$: The topology;
Input: A: compression algorithm set;
Input: γ: link capacity; $\iota^{link|node}$: latency of link/node;
Input: λ: atomic data size; δ: time interval between atomic data; α: number of paths at most;
Output: $\mathbf{X}$: bundle and compression algorithm selection mapping;
Output: $\mathbf{Y}$: bundle and path selection mapping.

1 $A' \leftarrow \left\{ a'_j | a'_j = \frac{r(a_j)\lambda}{\gamma} - \frac{o(a_j)}{clk} \right\}$;
2 `Sort(`A`, by value of` a'`, DESC);`
3 Retrieve all paths P and flows F from G;
4 $obj \leftarrow \infty$; `Init(`$\mathbf{X}, \mathbf{Y}$`);`
5 **for** $f_i \in f$ **do**
6 $j \leftarrow$ `Id(`a'_1`);`
7 $x_{ij} \leftarrow 1$;
8 Retrieve the paths set P^i of flow f_i from P;
9 `Sort(`P^i`, by path capacity, ASC);`
`/* Only consider the` α `most shortest paths. */`
10 $P^i \leftarrow P^i(0 : \alpha - 1)$;
11 **for** $p_l \in P^i$ **do**
`/* Index starts from 1. */`
12 $l \leftarrow$ `Id(`$P^i[1]$`);`
`/* Initialize variable` $\mathbf{Y}$`. */`
13 **if** $\sum_{l=1}^{\alpha(N-1)} y_{il} = 0$ **then**
14 $y_{il} \leftarrow 1$;
`/* Iteratively find the minimum objective solution. */`
15 **for** $a_j \in A$ **do**
16 $x_{ij}^{temp} \leftarrow 1$; $y_{il}^{temp} \leftarrow 1$; $obj^{temp} \leftarrow obj(\mathbf{X}^{temp}, \mathbf{Y}^{temp})$;
17 **if** $obj > obj^{temp}$ **then**
18 $obj \leftarrow obj^{temp}$;
19 $\mathbf{X} \leftarrow \mathbf{X}^{temp}$; $\mathbf{Y} \leftarrow \mathbf{Y}^{temp}$;
20 **return** $\mathbf{X}, \mathbf{Y}$;

B. The COMPRESSO Algorithm

Solving the problem of optimal transmission of SDN-empowered compressed measurement data is straightforward for small problem instances (*i.e.*, a small number of compression algorithms and ratios, and a small network topology) using an integer programming solver. However, integer programming solvers struggle to find optimal or even feasible solutions for large problem instances with a large number of compression algorithms and ratios due to the NP-hard complexity. To address this, we propose a heuristic solution called COMPRESSO (detailed in Algorithm 1) that balances compression performance and transmission overhead.

Algorithm 1 consists of two main phases: *preparation* and *iterative selection* phases. Lines 1-4 cover the preparation phase, where compression algorithms are reordered by the *influence* of compression (defined by $\frac{r(a)\lambda}{\gamma}$) and overhead

(defined by $\frac{o(a)}{clk}$). Calculating A' takes $O(H)$ time complexity, and the time complexity of the sorting procedure is $O(H \log H)$. Lines 5-19 describe the iterative selection phase. It tests if the algorithm can reduce the overall objective value for each flow (*i.e.*, data bundle). Since the algorithms are ordered by compression and overhead, we can easily select a compression algorithm that is "cost-effective" (*i.e.*, having a good compression ratio without introducing much overhead defined by A'). Whenever the current iterated solution has a smaller objective value, the objective value and the current solution $\mathbf{X}, \mathbf{Y}$ are updated. Calculating the objective value has a time complexity of $O(N)$. Thus, the total time complexity of the iterative selection phase is $O(N^2H)$, and the time complexity of Algorithm 1 is $O(N^2H) + O(H)$, which is polynomial time complexity.

VI. Simulation

In this section, we first introduce the simulation setup and compare algorithms, and then present the simulation result analysis. By evaluating the performance of the compared algorithms, we answer two questions: **i)** Does COMPRESSO reduce average transmission latency? **ii)** Does the computation of the COMPRESSO algorithm require a large amount of time?

A. Simulation Setup

1) Testbed Setup

We employ topologies from TopologyZoo [31] in our simulation. We run our simulation under all topologies but only show results of three topologies (*viz.*, Abilene [32], Chinanet [33], and Ion [34]) due to space limitations. Abilene has 11 network nodes and 14 links, Chinanet has 42 nodes and 66 links, while Ion has 128 nodes and 150 links. These topologies represent small (*i.e.*, Abilene), medium (*i.e.*, Chinanet), and large (*i.e.*, Ion) topologies in TopologyZoo. It is noted that the Ion topology is the largest topology we can run on our testbed. Our simulation is conducted using Python, and we employ `python-igraph`, a popular Python graph library, to read topologies from the `gml` file provided by TopologyZoo. Our simulation runs on a server equipped with an Intel Xeon 6420 @2.8 GHz CPU (32-Core) socket and 64 GB DRAM. In the simulation, each node attaches to a data aggregation/process device/server, in which we choose one node as the destination (that attaches to the data process server), and the remainder nodes are sources (that attach to data aggregation devices). Each data aggregation server can choose one path in the network to transmit (bundled) data to the destination based on the calculation of $\mathbf{X}, \mathbf{Y}$. We set $b = 4$ (apart from the bundle size simulations in Figure 3), set $\gamma \in [1 \text{ Kbps}, 10 \text{ Mbps}]$ randomly (apart from the path capacity simulations in Figure 5), set $\lambda = 100$ Kb (apart from the atomic data size tests in Figure 7), and set the time interval as 1 second (apart from the time interval tests in Figure 7). Moreover, we set $\alpha = 5$, $w = 5000$ b, $\iota_{link} = 1$ ms, $\iota_{node} = 0.5$ ms. We argue these parameters are practical settings for IoT networks [35]. Based on the observation in Section III-B, the compression ratios are generated randomly in $[1, 200]$, and the corresponding compression overhead is set as $2 \times 10^6 r(\cdot)$.

2) Compared Algorithms

1) Shortest Path (Shortest): this algorithm selects the path with the minimum number of hops (*i.e.*, the shortest path) to transmit each individual atomic measurement data that is neither bundled nor compressed before transmission.
2) Shortest Path with Bundle (Shortest Bundle): this algorithm bundles atomic measurement data into a single data bundle before transmission. The data bundle is then transmitted over the shortest path without data compression.
3) Shortest Path with Highest Compression Ratio (Shortest HiComp): this algorithm bundles atomic measurement data and compresses the bundle using the compression algorithm (or compression level) that achieves the highest compression ratio. The compressed data bundle is then transmitted over the shortest path.
4) Optimal: this algorithm represents the optimal solution to the O-STOCED problem. We utilize a popular optimization solver, Gurobi [36], to calculate the optimal solution.
5) COMPRESSO: this algorithm is detailed in Algorithm 1.

B. Can COMPRESSO Reduces Average Latency?

In this subsection, we answer this question by evaluating the average latency of different parameter settings. We first present the definition of *average latency* to avoid ambiguous understanding. We then show the simulation results under different topologies from TopologyZoo of various sizes. In conclusion, **COMPRESSO outperforms all other compared algorithms (except Optimal) in all settings.**

1) Average Latency

We define the average latency as the average time to transmit b consecutive atomic measurement data. This can be mathematically expressed as

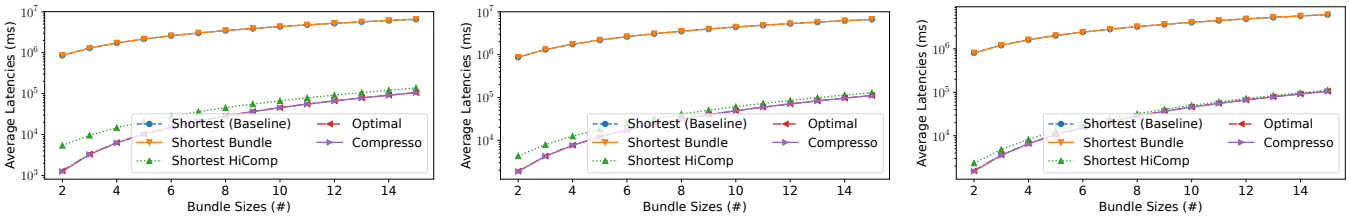
$$l = \frac{l_{\text{data process}} + l_{\textit{data transmission}} + l_{\textit{network path}} + l_{\textit{interval}}}{b}, \quad (12)$$

where $l_{\text{data process}}$ is the time taken to prepare all b atomic measurement data for transmission, $l_{\textit{data transmission}}$ is the time taken to transmit all b atomic measurement data, $l_{\textit{network path}}$ is the latency of the network path between the sender and receiver, and $l_{\textit{interval}}$ is the accumulation of all intervals between b atomic measurement data. The average latency metric is a crucial indicator of the QoS of a network [37], measuring the responsiveness of the network in transmitting data packets.

2) Average Latency of Different Bundle Sizes

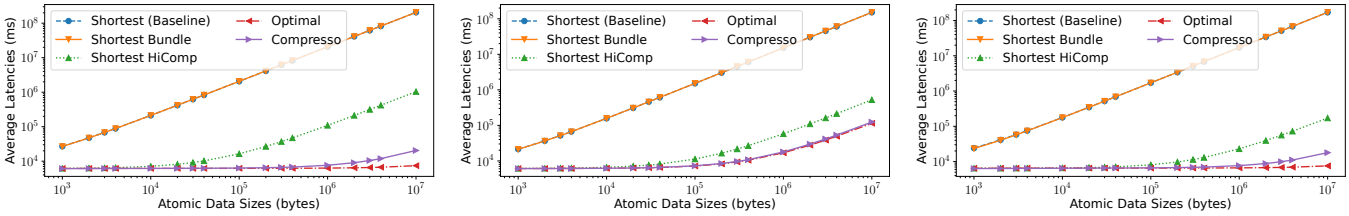
We vary the bundle size from 2 to 15 to evaluate the performance of the compared algorithms. As shown in Figure 3, **Optimal and COMPRESSO consistently outperform the other algorithms.** While the average latency slightly increases with larger bundle sizes (Shortest HiComp, Optimal, and COMPRESSO), the benefit of bundling diminish as bundle size increases. This suggests that the time interval becomes more influential with larger bundle sizes. Notably, Optimal and COMPRESSO overlap in all bundle sizes, indicating that COMPRESSO achieves comparable performance to Optimal.

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2025.3543479



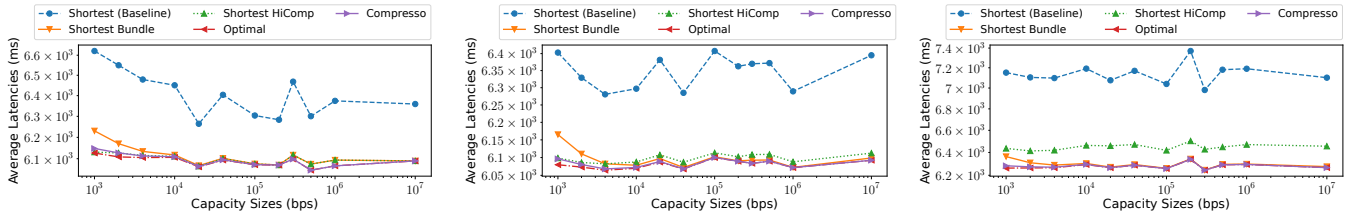
(a) Abilene topology. The lower the better. (b) Chinanet topology. The lower the better. (c) Ion topology. The lower the better.

Fig. 3: Average latencies of various bundle sizes under different topologies.



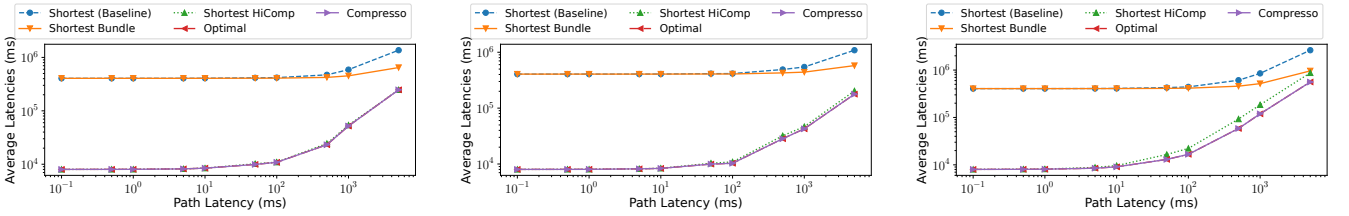
(a) Abilene topology. The lower the better. (b) Chinanet topology. The lower the better. (c) Ion topology. The lower the better.

Fig. 4: Average latencies of various atomic data sizes under different topologies.



(a) Abilene topology. The lower the better. (b) Chinanet topology. The lower the better. (c) Ion topology. The lower the better.

Fig. 5: Average latencies of various path capacities under different topologies.



(a) Abilene topology. The lower the better. (b) Chinanet topology. The lower the better. (c) Ion topology. The lower the better.

Fig. 6: Average latencies of various path latencies under different topologies.

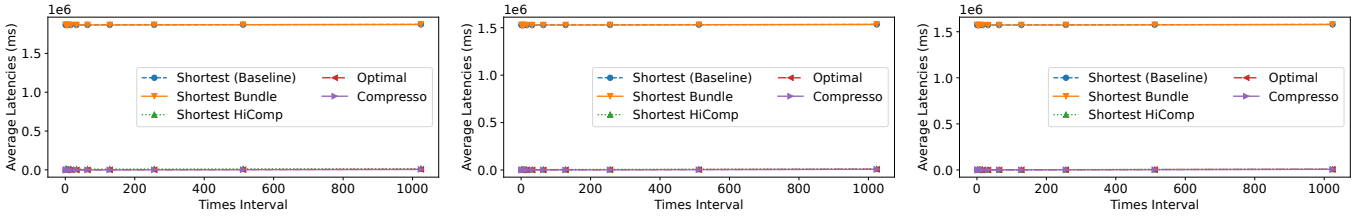
3) Average Latency of Different Atomic Data Sizes

Atomic data size can affect data transmission and compression times. We vary λ from 1 Kb to 10 Mb. As demonstrated in Figure 4, Shortest and Shortest Bundle show minimal change as atomic data size increases. Optimal and COMPRESSO exhibit similar performance for atomic data sizes below 400 Kb. However, for larger data sizes, the performance difference becomes significant in the Abilene and Ion topologies. This is attributed to the increased impact of data transfer times with larger atomic data sizes. On the other hand, COMPRESSO shows similar performance across all atomic data sizes in the Chinanet topology due to its "star-like" structure [38], which limits path diversity. Notably, despite the larger size of the Ion topology, the average latency is not significantly higher, indicating that path latency is less critical in the O-STOCED problem.

4) Average Latency of Different Path Capacities

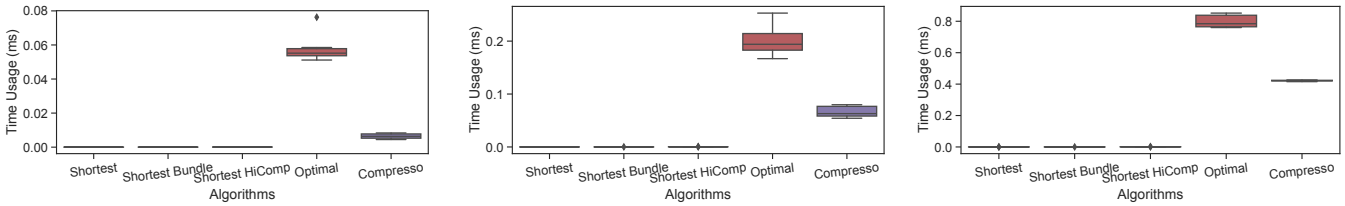
We evaluate the impact of path capacities by varying link capacity sizes from 1 Kb to 10 Mb. As shown in Figure 5, Shortest introduces up to 16.7% higher average latency than Optimal, while Shortest HiComp is approximately 4% higher.

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2025.3543479



(a) Abilene topology. The lower the better. (b) Chinanet topology. The lower the better. (c) Ion topology. The lower the better.

Fig. 7: Average latencies of various time intervals between atomic measurement data under different topologies.



(a) Abilene topology. The lower the better. (b) Chinanet topology. The lower the better. (c) Ion topology. The lower the better.

Fig. 8: Comparison of execution times of different algorithms under the topologies.

Shortest Bundle and COMPRESSO achieve similar performance to Optimal, particularly as capacity size increases. This is because path latency is relatively small (in the magnitude of 1/1000) compared to data transmission time usage (cf. Figure 6 for the impact of path latencies). Furthermore, Shortest HiComp generates higher latency than Shortest Bundle due to its overhead.

5) Average Latency of Different Path Latencies

We investigate the impact of path latencies by varying both link latency and node processing latency from 0.1 to 5000. As depicted in Figure 6, Optimal and COMPRESSO generally achieve over 100× improvement compared to Shortest and Shortest Bundle. In smaller topologies (*i.e.*, Abilene and Chinanet), Shortest HiComp and COMPRESSO initially perform comparably to Optimal for low path latencies. However, as latencies increase, the performance of Shortest HiComp and Optimal diverges. This is because latency has a more pronounced impact as it increases, leading to an "early" divergence in the Ion topology due to its longer paths (cf. Figure 6c).

6) Average Latency of Different Collection Rates

Finally, we explore the influence of time intervals[2] between consecutive atomic measurement data. The time interval introduces the same latency to all algorithms according to Equation (1). As expected, we observe a (asymptotical) linear increase in average latency as the time interval increases in Figure 7. However, the change is relatively small[3] due to the bandwidth being the primary bottleneck, resulting in similar performance for Shortest and Shortest Bundle.

[2]We use "time interval" for short if not explicitly point out otherwise.

[3]The y-axis is plotted in the logarithm scale, which diminishes the differences.

C. Does COMPRESSO Itself Introduce Significant Overhead?

Figure 8 presents the execution time required for each algorithm. Shortest, Shortest Bundle, and Shortest HiComp exhibit negligible execution times. This is because they solely utilize pre-calculated data (*e.g.*, shortest paths, compression ratios, and overhead) without additional computations. In comparison to Optimal, **COMPRESSO demonstrates a significant reduction in execution time, achieving over 50% improvement, while maintaining comparable performance levels.**

VII. PRIOR ARTS

Data compression techniques have been extensively utilized in data transmission over computer networks [39], [40]. However, most approaches typically employ a single specific compression algorithm. For instance, gzip [41] is commonly used in hypertext transfer protocol (HTTP) [42]. However, these methods do not address the optimization of compression ratios concerning computational complexity and data transmission performance. This section briefs existing arts under two categories: *lossless* and *lossy* solutions.

Lossless solutions. Lossless compression algorithms preserve data integrity by utilizing advanced encoding techniques. For example, the string "111111" can be compressed as "1" ×6 and readily decompressed to the original string. In such systems, data is initially compressed into an encoded format, and the compressed data is transmitted over the network. Encoding-based approaches have been in existence for approximately 80 years [23]. Even today, researchers continue to explore optimizations for data compression techniques [43]–[46]. For example, Brotli [43] is a modern dictionary-based compression algorithm developed by Google that prioritizes resource optimization while achieving high compression ratios. Similarly, x3 [46] incorporates the insertion of phrases into dictionaries. Chipm [44] and ELF [45] are designed for database systems but can also be employed as compression

algorithms for data transmission. Nevertheless, none of these approaches consider the selection of optimal compression algorithms and ratios for latency-sensitive communication.

In contrast to lossless compression algorithms, **lossy solutions** achieve higher compression ratios by compromising data fidelity [47]–[49]. Additionally, machine learning-based (ML-based) approaches constitute another category of lossy solutions [50]–[52]. Instead of directly identifying duplicate data, ML-based solutions convert it into an ML model, which is then transmitted across the network. ML-based approaches are effective for compressing multimedia data that exhibits tolerance to minor data inconsistencies [52] (*e.g.*, inconsistencies can be treated as noise points). Volumetric video transmission is a practical example of using ML models to enhance video quality under constrained bandwidth [53], [54]. However, the errors introduced by ML models can become excessive, resulting in the retrieved data unusable, particularly in measurement data compression where spikes are challenging to predict. Given the potential loss of critical information in lossy compression algorithms (cf. Section II), they are not suitable for measurement data transmission.

VIII. Conclusion

In this paper, we identified the stability of IoT measurement data, which can benefit from compression during transmission. Based on this insight, we proposed an SDN-empowered system for transmitting compressed measurement data, referred to as STOCED-SYS, which integrates SDN and compression to optimize measurement data transmission. We formulated the optimal transmission of SDN-empowered compressed (O-STOCED) problem as an optimization problem and proved its NP-hardness. To efficiently address this problem, we presented an efficient heuristic algorithm called Compresso. Our simulations demonstrate that Compresso achieves performance comparable to the optimal solution, reducing computational time by over 50%, with a latency reduction on the order of 0.5 ms.

References

[1] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A Survey," *Comput. Netw.*, vol. 54, no. 15, pp. 2787–2805, 2010.

[2] P. Schulz, M. Matthe, H. Klessig *et al.*, "Latency Critical IoT Applications in 5G: Perspective on the Design of Radio Interface and Network Architecture," *IEEE Commun. Mag.*, vol. 55, no. 2, pp. 70–78, 2017.

[3] Wikipedia. (2024) Narrowband IoT. Accessed: 2025-01-07. [Online]. Available: https://en.wikipedia.org/wiki/Narrowband_IoT

[4] Verizon. (2024) Verizon ThingSpace Marketplace. Accessed: 2025-01-07. [Online]. Available: https://thingspace.verizon.com/marketplace.html

[5] K. Muteba, K. Djouani, and T. Olwal, "5G NB-IoT: Design, Considerations, Solutions and Challenges," *Procedia Comput. Sci.*, vol. 198, pp. 86–93, 2022, EUSPN'21 / ICTH'21.

[6] P. Zetterberg and B. Ottersten, "The Spectrum Efficiency of a Base Station Antenna Array System for Spatially Selective Transmission," *IEEE TVT*, vol. 44, no. 3, pp. 651–660, 1995.

[7] C. Liu, C. Guo, Y. Yang *et al.*, "Adaptable Semantic Compression and Resource Allocation for Task-Oriented Communications," *IEEE TCCN*, 2023.

[8] F. Mehmeti, A. Papa, and W. Kellerer, "Maximizing Network Throughput Using SD-RAN," in *IEEE CCNC'23*, 2023, pp. 561–566.

[9] A. Liatifis, P. Sarigiannidis, V. Argyriou *et al.*, "Advancing SDN from Openflow to P4: A Survey," *ACM Comput. Surv.*, vol. 55, no. 9, pp. 1–37, 2023.

[10] C.-Y. Hong, S. Kandula, R. Mahajan *et al.*, "Achieving High Utilization with Software-Driven WAN," in *ACM SIGCOMM'13*, 2013, pp. 15–26.

[11] A. Gavrilovski, H. Jimenez, D. N. Mavris *et al.*, "Challenges and Opportunities in Flight Data Mining: A Review of the State of the Art," in *AIAA Infotech Aerospace'16*, 2016, p. 0923.

[12] J. Prados-Garzon, P. Ameigeiras, J. J. Ramos-Munoz *et al.*, "Performance Modeling of Softwarized Network Services Based on Queuing Theory with Experimental Validation," *IEEE TMC*, vol. 20, no. 4, pp. 1558–1573, 2019.

[13] S. H. Kamali, M. Hedayati, A. S. Izadi *et al.*, "The Monitoring of the Network Traffic Based on Queuing Theory and Simulation in Heterogeneous Network Environment," in *IEEE ICCTD'09*, vol. 1, 2009, pp. 322–326.

[14] K. M. Chandy, U. Herzog, and L. Woo, "Parametric Analysis of Queuing Networks," *IBM J. Res. Dev.*, vol. 19, no. 1, pp. 36–42, 1975.

[15] ObsPy.org. (2023) ObsPy Waveform File Examples. Accessed: 2025-01-07. [Online]. Available: https://examples.obspy.org

[16] R. J. Geller, D. D. Jackson, Y. Y. Kagan *et al.*, "Earthquakes Cannot be Predicted," *Sci.*, vol. 275, no. 5306, pp. 1616–1616, 1997.

[17] Q. Peng, A. Walid, J. Hwang *et al.*, "Multipath TCP: Analysis, Design, and Implementation," *IEEE/ACM ToN*, vol. 24, no. 1, pp. 596–609, 2014.

[18] Q. De Coninck and O. Bonaventure, "Multipath QUIC: Design and Evaluation," in *ACM CoNEXT'17*, 2017, pp. 160–166.

[19] Y. Lu, G. Chen, B. Li *et al.*, "Multi-Path Transport for RDMA in Datacenters," in *USENIX NSDI'18*, 2018, pp. 357–371.

[20] F. Tian, W. Feng, Y. Zhang *et al.*, "A Novel Software-Based Multi-path RDMA Solutionfor Data Center Networks," *arXiv preprint arXiv:2009.00243*, 2020.

[21] S. Jain, A. Kumar, S. Mandal *et al.*, "B4: Experience with a Globally-Deployed Software Defined WAN," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 43, no. 4, pp. 3–14, 2013.

[22] P. L. Ventre, S. Salsano, M. Polverini *et al.*, "Segment Routing: A Comprehensive Survey of Research Activities, Standardization Efforts, and Implementation Results," *IEEE Commun. Surv. Tutor*, vol. 23, no. 1, pp. 182–221, 2020.

[23] C. E. Shannon, "A Mathematical Theory of Communication," *BSTJ*, vol. 27, no. 3, pp. 379–423, 1948.

[24] J. Alakuijala, E. Kliuchnikov, Z. Szabadka *et al.*, "Comparison of Brotli, Deflate, Zopfli, LZMA, LZHAM and Bzip2 Compression Algorithms," *Google Inc*, pp. 1–6, 2015.

[25] W. Feng, C. Liu, and J. Chen, "BatchSketch: A "Network-Server" Aligned Solution for Efficient Mobile Edge Network Sketching," in *ACM MobiCOM'22*, 2022, pp. 811–813.

[26] M. Miao, W. Cheng, F. Ren *et al.*, "Smart Batching: A Load-Sensitive Self-Tuning Packet I/O Using Dynamic Batch Sizing," in *HPCC/SmartCity/DSS*, 2016, pp. 726–733.

[27] A. Mishra, S. Lim, and B. Leong, "Understanding Speciation in QUIC Congestion Control," in *ACM IMC'22*, 2022, pp. 560–566.

[28] IBM. (2023) Gurobi 11.0 Delivers Global Nonlinear Solving, Speed Enhancements, Dynamic Distributed Tuning, and Enterprise Features. Accessed: 2025-01-07. [Online]. Available: https://www.ibm.com/products/ilog-cplex-optimization-studio/cplex-optimizer

[29] L. Gurobi Optimization. (2023) IBM ILOG CPLEX Optimizer. Accessed: 2025-01-07. [Online]. Available: https://www.gurobi.com/news/gurobi-11-delivers-global-nonlinear-solving/

[30] M. R. Garey and D. S. Johnson, *Computers and Intractability*. freeman San Francisco, 1979, vol. 174.

[31] S. Knight, H. X. Nguyen, N. Falkner *et al.*, "The Internet Topology Zoo," *JSAC*, vol. 29, no. 9, pp. 1765–1775, October 2011.

[32] Topology Zoo. (2024) Abilene Network Topology. Accessed: 2025-01-07. [Online]. Available: http://www.topology-zoo.org/files/Abilene.gml

[33] Topology Zoo. (2024) Chinanet Network Topology. Accessed: 2025-01-07. [Online]. Available: http://www.topology-zoo.org/files/Chinanet.gml

[34] Topology Zoo. (2024) Ion Network Topology. Accessed: 2025-01-07. [Online]. Available: http://www.topology-zoo.org/files/Ion.gml

[35] C. Gündoğan, P. Kietzmann, M. Lenders *et al.*, "NDN, CoAP, and MQTT: A Comparative Measurement Study in the IoT," in *ACM ICN'18*, 2018, pp. 159–171.

[36] Gurobi. (2024) Gurobi Optimizer. Accessed: 2025-01-07. [Online]. Available: Gurobi:http://www.gurobi.com

[37] Y. Chen, Y. Wang, M. Liu *et al.*, "Network Slicing Enabled Resource Management for Service-Oriented Ultra-Reliable and Low-Latency Vehicular Networks," *IEEE TVT*, vol. 69, no. 7, pp. 7847–7862, 2020.

[38] L. Goratti, T. Baykas, T. Rasheed *et al.*, "NACRP: A Connectivity Protocol for Star Topology Wireless Sensor Networks," *IEEE Wirel. Commun. Lett.*, vol. 5, no. 2, pp. 120–123, 2015.

[39] D. A. Lelewer and D. S. Hirschberg, "Data Compression," *ACM CSUR*, vol. 19, no. 3, pp. 261–296, 1987.

[40] N. Hu, "Network Aware Data Transmission with Compression," *(SOCS-4)*, p. 33, 2001.

[41] P. Deutsch, *GZIP File Format Specification Version 4.3*, Std. RFC 1952, 1996.
[42] J. Reschke, *RFC 7694: Hypertext Transfer Protocol (HTTP) Client-Initiated Content-Encoding*, Std. RFC 7694, 2015.
[43] J. Alakuijala, A. Farruggia, P. Ferragina *et al.*, "Brotli: A General-Purpose Data Compressor," *ACM TOIS*, vol. 37, no. 1, pp. 1–30, 2018.
[44] P. Liakos, K. Papakonstantinopoulou, and Y. Kotidis, "Chimp: Efficient Lossless Floating Point Compression for Time Series Databases," *VLDB'22*, vol. 15, no. 11, pp. 3058–3070, 2022.
[45] R. Li, Z. Li, Y. Wu *et al.*, "Elf: Erasing-based lossless floating-point compression," *VLDB'23*, vol. 16, no. 7, pp. 1763–1776, 2023.
[46] D. Barina, "Experimental Lossless Data Compressor," *Microprocess. Microsyst.*, vol. 98, p. 104803, 2023.
[47] L. Theis, T. Salimans, M. D. Hoffman *et al.*, "Lossy Compression with Gaussian Diffusion," *arXiv preprint arXiv:2206.08889*, 2022.
[48] J. Sun, T. Yan, H. Sun *et al.*, "Lossy Compression of Communication Traces Using Recurrent Neural Networks," *IEEE TPDS*, vol. 33, no. 11, pp. 3106–3116, 2022.
[49] A. Elzanaty, A. Giorgetti, and M. Chiani, "Lossy Compression of Noisy Sparse Sources Based on Syndrome Encoding," *IEEE ToC*, vol. 67, no. 10, pp. 7073–7087, 2019.
[50] M. I. Patel, S. Suthar, and J. Thakar, "Survey on Image Compression Using Machine Learning and Deep Learning," in *IEEE ICCS'19*, 2019, pp. 1103–1105.
[51] X. Yu, Y. Peng, F. Li *et al.*, "Two-Level Data Compression Using Machine Learning in Time Series Database," in *IEEE ICDE'20*, 2020, pp. 1333–1344.
[52] D. Sculley and C. E. Brodley, "Compression and Machine Learning: A New Perspective on Feature Space Vectors," in *IEEE DCC'06*, 2006, pp. 332–341.
[53] S. Wang, S. Yang, H. Su *et al.*, "Robust Saliency-Driven Quality Adaptation for Mobile 360-Degree Video Streaming," *IEEE TMC*, 2023.
[54] Y. Liu, B. Han, F. Qian *et al.*, "Vues: Practical Mobile Volumetric Video Streaming Through multiview transcoding," in *ACM MobiCOM'22*, 2022, pp. 514–527.