Title: Optimized flow assignment for applications with strict reliability and latency constraints using path diversity

Author(s): Zahraa Sweidan, Rania Islambouli, Sanaa Sharafeddine

Journal: Journal of Computational Science

DOI/Link: https://doi.org/10.1016/j.jocs.2020.101163

How to cite this post-print from LAUR:


Year 2020

This Open Access post-print is licensed under a Creative Commons Attribution-Non Commercial-No Derivatives (CC-BY-NC-ND 4.0)

This paper is posted at LAU Repository

For more information, please contact: archives@lau.edu.lb
Abstract—The unprecedented increase in the number of smart connected devices invoked a plethora of diverse applications with different performance requirements stipulating various network management strategies. Ultra-reliable low-latency communication (URLLC), one of the promised 5G dimensions, is expected to enable mission-critical applications while adhering to the heterogeneity of their quality metrics. At its core, URLLC rests on the notion of providing stringent reliability and latency requirements, in which guaranteed network availability becomes a necessity. Network Slicing (NS) is one of the key paradigms that can offer performance guarantees through customized network management of software defined networking (SDN). However, unlocking URLLC with network slicing based mechanisms requires careful demultiplexing of the network into various slices and proper assignment of traffic flows generated by ultra-reliable low latency applications over those slices. Within each slice, multiple disjoint paths may be selected to ensure the reliability requirement of the assigned application while meeting its latency constraint. Hence, we study, in this paper, the joint problem of forming end-to-end network slices, mapping URLLC applications to corresponding slices and assigning their traffic flows over multiple disjoint paths; then formulate it as a mixed integer program. Due to its complexity, we decompose the problem into two subproblems; end-to-end disjoint paths and traffic flow assignment for ultra-reliable low latency applications. Simulation results are presented for various scenarios to demonstrate the performance and scalability of the proposed decomposition approach as compared to the general formulation.

Index Terms—URLLC, Industry 4.0, network slicing, multi-path diversity, traffic flow assignment

I. INTRODUCTION

Recent technological breakthrough are unlocking attractive applications in the fourth industrial revolution, or Industry 4.0. The value of the global Industry 4.0 is estimated to reach 155.30 billion USD in 2024 [1]. Industry 4.0 is expected to support high flexibility in manufacturing and customized production leading to premium service quality, increased productivity and eventually significant revenues. Furthermore, it allows a perfect alignment between humans and machines through smart decision making in real-time [2], [3]. Other Industry 4.0 applications include factory automation [4], drone-based delivery [5], telemedicine monitoring [5], and intelligent transportation [6] among others.

While Industry 4.0 applications promise integrating main pillars such as Internet of Things (IoT) [7], integrated systems, big data, cloud computing [8], augmented reality, autonomous systems, cyber security [9], simulation [10], and additive manufacturing concepts systems to the current industrial environment [2], it is coupled with different challenges. First, supporting several applications with heterogeneous quality of service (QoS) requirements demands a flexible network that satisfies the particular needs of each type of application. For example, mission-critical applications require very stringent latency and reliability requirements such as alarm systems with end-to-end latency of 0.5 ms and reliability in the order of $1 - 10^{-9}$ [11]. Other applications may require high data rates such as virtual reality exceeding 100 Mbps with an average latency of 10 ms [12]. Second, Industry 4.0 applications encapsulate a lot of industrial devices with different capabilities and communication technologies. Another challenge is to meet the stringent reliability and latency of those mission-critical applications. For example, customized medicine manufacturing machines is based on initiating several patient information requests from the cloud with very tight reliability ($10^{-4}$) and latency (10ms) communication specifications to adjust the number of chemicals in the produced drug [13]. If the system experiences a reliability or latency drop down, the system往往 comes to halt, incurring a lot of losses and hindering the productivity of the system. A major network challenge is to manage the network to accommodate this complexity of heterogeneity among those devices [2]. Thus, devising innovative networking solutions is essential to support the diverse needs of those applications.

5G comes along with different dimensions to support this wide range of applications with variable reliability, latency and bandwidth demands. In particular, 5G offer performance improvements along three use-cases including [14]: 1) Enhanced Mobile Broadband (eMBB) for providing high data rates in a wide coverage area; 2) Massive Machine Type Communications (mMTC) for supporting a huge number of
devices in a small area taking into account sporadic behavior; 3) Ultra Reliable Low Latency Communications (URLLC) for providing high reliability and low latency for mission critical communications. Accordingly, supporting ultra-reliable low latency communications (URLLC) to enable mission-critical applications is an important dimension. Moreover, URLLC must account to the stringent network demands and ensure seamless user-experience [15]. However, achieving URLLC at the network level is one of the major challenges in 5G networks due to i) the high amount of interactions among devices, backhaul, and fronthaul links and the cloud and ii) increasing uncertainty resulting from random changes in the network topology [16]. Various techniques have been proposed to explore the tradeoff relation between reliability and latency for URLLC applications including spatial diversity [17], [18], interface diversity [19], [20], and network slicing [13] among others. Yet the coupling of high reliability and low latency requirements in URLLC use cases introduce a set of challenges in terms of system design.

While some solutions with single monolithic architecture fail to support various industrial scenarios simultaneously, developing new solutions revolves around achieving flexible networking techniques that adhere to the variable latency and reliability demands requested by various applications. Network slicing aims at providing a service-oriented architecture while satisfying the diverse requirements of those applications [21]. With the advent of software-defined networks (SDN) and network function virtualization (NFV) [10], network slicing allows to reserve dedicated end-to-end logical network, or slice, for each service with isolated virtual resources on a shared physical infrastructure. Further, the network controller assigns slices for each type of service that adheres to the specific service demands in terms of reliability, latency, bandwidth, and others [4], [22]. A network slice can be granted for an application on demand and revoked when users are not benefiting from it. This leads to better utilization of the network resources and higher flexibility to meet service demands. The bandwidth requirements of each network slice takes into consideration the quality of service demand of each application type and marginal bandwidth may be allocated to accommodate a margin of traffic deviation within the same network slice. Techniques used for network planning in [23], [24] may be applied to dimension the different network slices allocated for the given application types.

In a previous work in [25], we investigate the trade-off relation between reliability and latency for one application over multiple logical paths. We extend on our previous work to consider a more general and practical network model with multiple applications and identify the specific paths in the network as part of a dedicated slice that satisfy the applications’ stringent performance requirements. We aim at providing an end-to-end network slicing mechanism for URLLC applications in the context of a smart factory and explore the assignment of a dedicated slice for each application while ensuring reliability and latency requirements. The problem intends to: i) maximize the number of admitted URLLC applications running over a given network ii) dedicate network paths as part of the network slice assigned to the traffic flows generated by each application.

We hence address the problem of determining end-to-end network slices to support ultra-reliable low latency applications (NS-URLLC) over a given network infrastructure. We formulate the NS-URLLC problem as a mixed integer program (NS-URLLC-MIP) with the objective of maximizing admitted application requests. We consider replicating application requests on several disjoint paths to fulfil the reliability requirement [26]. This, however, comes at the cost of increasing the delay experienced by the admitted applications due to the increased traffic at the network nodes.

The paper is structured as follows. Section II surveys related literature and highlights the main contributions of this work as compared to existing research. Section III describes the system model and the major components. Section IV formulates the problem and Section V decomposes the problem into two subproblems to handle the high complexity of the general problem formulation. Simulation results are presented and analyzed in Section VI. Finally, conclusions are drawn in Section VII.

II. RELATED LITERATURE

Achieving ultra-reliable low-latency communication (URLLC) represents one of the major challenges facing 5G networks which is being greatly examined in the literature, of which we are going to survey the most related work. In [16], the authors explain different enablers for URLLC in wireless communication. However, the authors show how the intersection between the latency and reliability enablers requires careful trade-offs and incur huge risks in terms of resource consumption. In [27], the authors investigate the effect of multi-path solutions in improving the quality of the delivered video in an ad-hoc network while exploring two different schemes: video redundancy coding and redundant frames. Similarly, in [28], the authors discuss several performance gains obtained when utilizing multiple communication paths to transmit data between two end systems such as load balancing, boosted performance and reliability. In [29], the authors propose enhancements on Media Access Control (MAC) layer to enable reliable heterogeneous vehicular services while considering device-to-device communication. In [17], the authors suggest spatial diversity at the physical layer technologies to enable URLLC. The proposed approach includes waveform multiplexing, multiple-access scheme, channel code design, synchronization, and full-duplex transmission for higher spectral efficiency. In [30], the author proposes an application-specific radio resource slicing strategy for 5G networks with haptic communications. The author assumes a 5G LTE-A RAN network with conventional multiple access schemes where a slicing controller allocates special slices for users with haptic communication. A low-complexity heuristic algorithm is introduced to find an optimal slicing strategy that minimizes the power and resource block allocation for different users.
in both uplink and downlink multiple access schemes. However, one application type is only explored in this work. The authors in [31] present a dynamic resource reservation mechanism for bandwidth allocation based on deep reinforcement learning. The system model considers applications, considered as virtual networks with known reliability and latency requirements, which are connected to different base stations (BSs). The controller reserves minimum resource requirement ratios for the slices at each BS in the initial step. Then, the radio resources are assigned to slices according to their weights based on deep reinforcement learning. The controller adjusts the overall unused resources to allow other services to utilize the unused resources. Also, intra-slice physical resource allocation is formulated with two options: 1) maximizing the rate, and 2) minimizing delay based on the QoS requirements. While the work formulated in [31] considers one type of resource to allocate for the different given slices, the authors in [32] propose more than one type of resource to enhance the network slicing mechanism for different types of slices such as mini-slots to URLLC users and physical resource blocks to eMBB users leading to higher utilization of the spectral efficiency. The authors discuss the feasibility of allocating physical resource blocks to eMBB users and analyze resource enhancements. Unlike the work done before, the authors in [33] utilize the edge to provide URLLC guarantees. The authors propose an edge offloading mechanism based on extreme value theory for resource-limited devices with URLLC applications. The formulated policy for local task execution, task offloading, and resource allocation is based on Lyapunov stochastic optimization. Yet the given scheme can only provide an approximation for the optimal solution. The authors in [34] demonstrate the feasibility of URLLC for machine type communication services in a factory automation use case with a latency of sub-milliseconds and failure rate of $10^{-9}$. To meet the stringent reliability and latency requirements, they assume a physical layer with reduced transmission time intervals, shorter OFDM symbol duration, convolutional codes for user data and high diversity levels. Only radio access delay in an OFDM based system is considered. The authors in [35], [36] propose two deterministic models to provide an end-to-end QoS routing with real-time guarantees in a software-defined networking (SDN) framework. The first network model assigns a rate and buffer budget for each queue over the network and calculates a worst-case delay. The second model prevents the prior assignment of a specific rate and buffer budget through defining a maximum delay for each queue and thus leading to better resource utilization. Both proposed models can be deployed in a centralized control plane thus facilitating quick provisioning of the new requested flows. The authors in [37] then extend to present a quality of service (QoS) framework for arbitrary hybrid wired/wireless networks which provide the delay bound and the target reliability of each application provided. In addition, a reliability-based scheduler for Wireless Sensor Network (WSN) is proposed to achieve the targeted reliability while accounting dynamic interference. In [13], the authors consider different industrial applications with different reliability and latency requirements communicating along with a factory with a hierarchical network structure. The authors present three transmission schemes for one deterministic and several stochastic applications transmitting simultaneously to calculate the probability of reliable transmission. The proposed network slicing mechanism aims to provide dedicated slices through associating an egress queue for each slice in each hop in the network. The authors use deterministic network calculus to trace the delay performance of the network slicing scheme. The results show that overwriting the sent telegrams of a deterministic application by a stochastic application minimizes end-to-end reliability failure. However, the results depict that the mechanism fails when the number of applications increases for applications with low priority. The authors in [7] propose an adaptive data rate control mechanism for IoT services in low power wide area networks to enhance the availability of the network.

A. Novelty of our work

In light of the surveyed literature, the vast majority of the existing work considers enabling URLLC at the access network. In terms of problem definition, the key novelty of our work is the focus on enabling URLLC applications over a factory core network. We also consider replicating traffic flows generated by applications over selected paths to meet their reliability and latency thresholds.

![Fig. 1: Resource allocation for two industrial applications with strict reliability and latency over different network slices in the available network.](image)

III. SYSTEM MODEL

In this section, we define our system model that comprises two parts: 1) the network model which illustrates our proposed scenario and assumptions 2) the network latency and reliability models used throughout the paper.
TABLE I: Different URLLC Application Types [13]

<table>
<thead>
<tr>
<th>Application</th>
<th>Reliability</th>
<th>Latency</th>
<th>Request Size</th>
<th>Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Augmented Reality (AR)</td>
<td>$1 - 10^{-2}$</td>
<td>20 ms</td>
<td>40 Kb</td>
<td>100</td>
</tr>
<tr>
<td>Alarm Signals (AS)</td>
<td>$1 - 10^{-6}$</td>
<td>1 ms</td>
<td>10 Kb</td>
<td>10</td>
</tr>
<tr>
<td>Cloud Requests (CR)</td>
<td>$1 - 10^{-4}$</td>
<td>10 ms</td>
<td>128 B</td>
<td>25</td>
</tr>
<tr>
<td>Motion Control (MC)</td>
<td>$1 - 10^{-4}$</td>
<td>2 ms</td>
<td>128 B</td>
<td>40</td>
</tr>
</tbody>
</table>

A. Network model

Our system model resembles the scenario depicted in Fig. 1. We envision a smart factory that includes different URLLC applications. We assume multiple applications running whereby tight reliability and latency requirements must be ensured. Each application is provisioned a network slice that adheres to its reliability and latency requirements. Table I presents four mission-critical example types of applications of a smart factory, their requirements in terms of reliability and latency, and their specifications in terms of request size, and arrival rate. Such applications are supported via dedicated end-to-end resource provisioning to maintain a seamless and accurate data transmission. Therefore, various slices serving different reliability and latency thresholds are devised over the factory network. Formally, we consider the factory network as a graph $G(V, L)$, where $V$ is the set of backbone network equipment (e.g., routers, switches,...) interconnected with a set of communication links $L$. We consider one ingress node $S$ responsible for aggregating incoming application requests and providing the slicing mechanism that admits a seamless and accurate data transmission. Therefore, various slices serving different reliability and latency thresholds are devised over the factory network. Formally, we consider the factory network as a graph $G(V, L)$, where $V$ is the set of backbone network equipment (e.g., routers, switches,...) interconnected with a set of communication links $L$. We consider one ingress node $S$ responsible for aggregating incoming application requests and providing the slicing mechanism that admits the maximum number of application requests over the network. Once provisioned, each slice will reserve enough number of paths to guarantee the given reliability and latency thresholds for every application. Let $A$ be the set of all ultra-reliable low latency applications. The ingress node will select, for each application $a \in A$, a number of disjoint paths $p \in P$ that ensures meeting its reliability and latency demands and thus physical paths are utilized by traffic generated from various applications; for example, customer cloud requests and motion control may allocate common paths along the network to satisfy a reliability threshold of $1 - 10^{-4}$. Hence, selecting a number of disjoint paths for an application $a$ depends on meeting minimum reliability $\theta_a$ and a maximum tolerated delay $T_a$. The aggregate data generated by each application $a$ is assumed to follow a Poisson process with an arrival rate $\lambda_a$(requests/sec).

B. Network Latency and Reliability Models

- Experienced Latency: We consider the overall delay $D_p$ experienced by an admitted application along a certain path $p$ composed of multiple links $ij$ and evaluate it as:

$$D_p = \sum_{ij \in p} D_{ij}$$

(1)

where $D_{ij}$ is the delay incurred by link $ij \in p$, joining nodes $i$ and $j$. Besides, we assume that application data at a given node $i \in G$ is queued in a first-come-first-served manner. Let $\mu_{ij}$ represent the service rate (packets/sec) which corresponds to the time needed to serve a packet with an average size $B_{av}$, and is expressed as follows:

$$\mu_{ij} = \frac{C_{ij}}{B_{av}}$$

(2)

where $C_{ij}$ corresponds to link $L_{ij}$ transmission speed. Having applications generate data according to a Poisson process, the aggregation of multiple Poisson arrivals at one network node also follows a Poisson process whose rate is the summation of the individual rates. Further, we assume that the service time at the network nodes is exponentially distributed; thus, each node is then modeled as an M/M/1 queuing system. According to Little’s Theorem, the delay experienced along a link $ij$ by an application is expressed as follows:

$$D_{ij} = \frac{1}{\mu_{ij} - \lambda_{ij}}$$

(3)

- Network Reliability: In this work, we express the reliability of a given path $R_p$ in terms of the physical link characteristics [13]. We denote $r_{ij}$ as the reliability of a given link $ij$. The experienced reliability at a given path $p$ is then computed as follows:

$$R_p = \prod_{ij \in p} r_{ij}$$

(4)

Furthermore, the total reliability $R$ experienced by an application jointly depends on the reliability of all selected paths as follows:

$$R = 1 - \prod_{p \in P} (1 - R_p)$$

(5)

As per (5), the replication of an application data over multiple paths increases the achieved reliability. The tradeoff between the achieved latency and reliability is demonstrated through the following example.

Latency/Reliability Tradeoff Example: Consider two different URLLC application flows are generated by one device as shown in the figure 2. We consider two applications $A_1$ and $A_2$ which corresponds to alarm signals and motion control applications respectively. The arrival rate $\lambda_a$ for each generated application is given in Table I. Also, as given in the table, alarm signals has to be sent with a maximum delay of 1 ms and reliability of $1 - 10^{-6}$. For motion control, delay should be no greater than 2 ms while reliability should be at least $1 - 10^{-4}$. To accommodate for the given requirements, we assume three different disjoint paths: $p_1$, $p_2$, and $p_3$ with reliability $R_1$, $R_2$, and $R_3$ respectively. For the sake of simplicity we assume equal capacities for all paths (capacity = 1000 Mbps). If we consider sending the two given application flows over the most reliable path, $p_2$, the reliability requirement for both applications will be fulfilled. However, this will incur an extra delay which can’t be tolerated by the alarm application requests. The service rate will decrease for the given applications as the average packet
Fig. 2: Tradeoff between reliability and latency for two different applications on multiple paths. Two applications \( A_1 \) and \( A_2 \) generating different sized-packets toward different destinations \( D_1 \) and \( D_2 \) respectively whereby data replication achieves the target reliability size is the average between both applications and evaluates to 5512 bits. This leads to an experienced delay of:

\[
\frac{1}{1 \times 10^3} = 7 \times 10^{-3} \text{sec} = 0.7 \text{ms} \not\leq 1 \text{ms}. \tag{6}
\]

As both applications had their latency target violated, traffic flow assignment of the applications is modified whereby each application flow utilizes a separate path. The latter scenario satisfies the requirements of alarm signals while motion control is rejected. However, considering a different assignment where motion control flows over two paths achieves its required reliability and thus lead to admitting both applications.

In light of the above example, one can note that maximizing the total number of admitted application flows is challenging due to the tradeoff between the performance requirements of low latency and ultra high reliability.

C. Problem Description

While supporting ultra-reliable low latency communication for mission-critical applications may require extending the network infrastructure or modifications at hardware level, we explore utilizing the existing infrastructure to jointly adhere to the latency and reliability demands. We study, based on the proposed system model, the joint problem of forming end-to-end network slices, mapping URLLC applications to corresponding slices and assigning their traffic flows over multiple disjoint paths. The main objective of this problem is to admit the maximum number of application flows that can run over the assigned network slices with guaranteed quality metrics.

IV. END-TO-END NETWORK SLICING FOR URLLC APPLICATIONS - A MIXED INTEGER PROGRAM (NS-URLLC-MIP)

A. Problem Definition

**Definition 1:** Given \( G(V, L) \), a source ingress node \( S \), a destination egress node \( D \) and a set \( A \) of URLLC applications of defined reliability and latency thresholds, select the paths between \( S \) and \( D \) in \( G(V, L) \) that maximizes the number of admitted applications.

B. Problem Formulation

Table II shows the system model variables and parameters used in the problem formulation. We define a variable \( x_a \) to determine whether an application \( a \in A \) is admitted to the network.

\[
x_a = \begin{cases} 
1 & \text{if application } a \text{ is admitted} \\
0 & \text{otherwise}
\end{cases}
\]

In this work, our objective is to maximize the number of admitted application flows while adhering to the different delay and reliability requirements of the given URLLC applications:

\[
\text{maximize } \sum_{a \in A} w_a x_a \tag{7}
\]

where \( w_a \) refers to the priority level assigned to application \( a \). Let \( l^p_{ij} \) be a binary decision variable that determines whether a path \( p \in P \) is routed through a link \( ij \in L \).

\[
l^p_{ij} = \begin{cases} 
1 & \text{if path } p \text{ is routed through link } ij \\
0 & \text{otherwise}
\end{cases}
\]

We also define a binary variable that determines whether the traffic flow of application \( a \) is transferred along path \( p \).

\[
z^p_a = \begin{cases} 
1 & \text{if traffic flow of application } a \text{ is assigned path } p \\
0 & \text{otherwise}
\end{cases}
\]

The problem of maximizing the total number of URLLC applications and determining their respective end-to-end network slices is subject to several constraints. Constraints (8) and (9) express the conditions of flow conservation. Constraint (8) maintains the flows at the source node \( s \). We need to ensure that the number of outgoing flows at the source node \( s \) is equal to 1 for each path \( p \in P \). Constraint (9) ensures that incoming application flows to a node \( i \in V \) should be equal

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( P )</td>
<td>Set of paths available from source ( s ) to destination ( d )</td>
</tr>
<tr>
<td>( L )</td>
<td>Total number of links</td>
</tr>
<tr>
<td>( A )</td>
<td>Total number of application requests</td>
</tr>
<tr>
<td>( c_{ij} )</td>
<td>Capacity of link ( ij )</td>
</tr>
<tr>
<td>( B_{kv} )</td>
<td>Average packet size</td>
</tr>
<tr>
<td>( \lambda_a )</td>
<td>Arrival rate of application request ( a )</td>
</tr>
<tr>
<td>( w_a )</td>
<td>Weight reflecting the priority of an application ( a )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variables</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x_a )</td>
<td>Binary variable that indicates whether a request ( a ) is admitted</td>
</tr>
<tr>
<td>( b_{ij} )</td>
<td>Binary variable that indicates whether request ( a ) utilizes path ( p )</td>
</tr>
<tr>
<td>( y_p )</td>
<td>Binary variable that indicates whether a path ( p ) is activated</td>
</tr>
<tr>
<td>( z^p_a )</td>
<td>Binary variable that indicates whether a request ( a ) uses path ( p )</td>
</tr>
<tr>
<td>( l^p_{ij} )</td>
<td>Binary variable that determines whether a path ( p ) is routed through link ( ij )</td>
</tr>
</tbody>
</table>

| TABLE II: System model parameters and variables |
to the number of outgoing application flows over this path $p \in P$:

$$\sum_{j:(i,j)\in L} p_{ij}^p = 1 \quad \forall p \in P, \forall i = s \in V \quad \forall ij \in L$$

(8)

$$\sum_{j:(i,j)\in L} p_{ij}^p - \sum_{j:(i,j)\in L} p_{ij}^0 = 0 \quad \forall p \in P, i(\neq s, d) \in V \quad \forall ij \in L$$

(9)

Constraint (10) indicates that possible paths must not share any common link.

$$\sum_{p\in P} p_{ij}^p \leq 1 \quad \forall ij \in L$$

(10)

Constraint (11) ensures that an application $a$ uses path $p$ if $p$ associates with at least one outgoing link $ij$ from source $s$.

$$z_a^p \leq \sum_{p\in P} p_{ij}^p \quad \forall a \in A \quad \forall p \in P \quad \forall i = s \in V$$

(11)

Constraints (12) and (13) ensures that an application request $a$ is selected only if it is sent along at least one path $p$.

$$z_a^p \leq x_a \quad \forall a \in A \quad \forall p \in P$$

(12)

$$x_a \leq \sum_{p\in P} z_a^p \quad \forall a \in A \quad \forall p \in P$$

(13)

- Delay Constraints: As mentioned earlier, given the incoming application flows from different devices, we model each node in the network as an M/M/1 queuing system with aggregate request arrival rate of $\sum_{p\in P} \sum_{a\in A} \lambda_a z_a^p p_{ij}^p$ and service rate of $C_{ij}$ where $C_{ij}$ is the link capacity in bits per second and $B_{av}$ is the average data size in bits. Therefore, we constrain the total delay in terms of the queuing and transmission delay as follows:

$$\sum_{ij\in L} z_a^p p_{ij}^p \left( \frac{C_{ij}}{B_{av}} - \sum_{p\in P} \sum_{a\in A} \lambda_a z_a^p p_{ij}^p \right) \leq T_a \quad \forall a \in A \quad \forall ij \in L$$

(14)

Further, the queue of each node should remain stable. Then, the average service rate of a certain node should be greater than the aggregate average arrival rates of all application flows routed through this node as given in (15).

$$\frac{C_{ij}}{B_{av}} - \sum_{p\in P} \sum_{a\in A} \lambda_a z_a^p p_{ij}^p > 0 \quad \forall ij \in L$$

(15)

- Reliability constraint: we need to ensure satisfying the reliability requirement of each application to allow it to be admitted to the network. As a result, the achieved reliability by utilizing multiple paths for each application $a$ should meet the minimum required reliability as follows:

$$1 - \prod_{p\in P} (1 - z_a^p R_p) \geq \theta_a \quad \forall a \in A$$

(16)

$$R_p$$ corresponds to the reliability at a given path $p \in P$ expressed in the following constraint:

$$R_p = \prod_{ij\in L} (1 - f_{ij}^p p_{ij}) \quad \forall ij \in L \quad \forall p \in P$$

(17)

where $f_{ij}$ corresponds to the failure probability of a link $ij$ and it is evaluated as $1 - r_{ij}$. Constraints (14), (15), (16) and (17) are linearized to obtain a linear optimization model. Linearization of those constraints is omitted for brevity.

V. NETWORK SLICING FOR URLLC APPLICATIONS - A DECOMPOSITION APPROACH (NS-URLLC-DA)

After addressing end-to-end network slicing over a given network for URLLC applications, we examine splitting it into subproblems in the aim of reducing its complexity. To do so, we consider determining the end-to-end disjoint paths (E2E-DP) as one subproblem whose output is input to the second subproblem of assigning traffic flows of ultra-reliable low latency applications over the determined end-to-end paths (TFA-URLLC). We depict in Fig.3 a flowchart that describes the process of NS-URLLC decomposition approach to illustrate how the problem is divided into two subproblems. In fact, NS-URLLC-DA divides the problem into two independent subproblems through which disjoint paths in E2E-DP act as an input to the second subproblem responsible for assigning the traffic flows for all applications.

[width=0.35]paper/images/flowchart.png

Fig. 3: Flowchart of NS-URLLC-DA.

A. E2E-DP Problem Definition and Formulation

Definition 2: Given a network $G(V, L)$, a source ingress node $S$ and a destination egress node $D$, determine all possible link disjoint paths between $S$ and $D$.

We aim at finding a set of $P$ disjoint paths from the source ingress node to the destination egress node. We consider the maximum number of possible disjoint paths equals to the number of outgoing links from the ingress node. We consider the constraints (8), (9) and (10) in NS-URLLC-MIP.

$$\sum_{j:(i,j)\in L} p_{ij}^p = 1 \quad \forall p \in P, i = s \in V \quad \forall ij \in L$$

(18)

$$\sum_{j:(i,j)\in L} p_{ij}^p - \sum_{j:(i,j)\in L} p_{ij}^0 = 0 \quad \forall p \in P, i(\neq s, d) \in V \quad \forall ij \in L$$

(19)

$$\sum_{p\in P} p_{ij}^p \leq 1 \quad \forall ij \in L$$

(20)

B. TFA-URLLC Problem Definition and Formulation

Definition 3: Given a network $G(V, L)$, a set of $A$ of URLLC applications and a set of paths $P$ from source ingress node $S$ to destination egress node $D$, determine the maximum number of applications that can be admitted to the network and assign paths for each admitted application.

We define the decision variable $y_p \in \{0, 1\}$ to determine whether path $p \in P$ is selected to transfer application requests as follows:

$$y_p = \begin{cases} 1 & \text{if path } p \text{ is selected} \\ 0 & \text{otherwise} \end{cases}$$
To simplify some constraints, we declare a binary variable $b_{ij}^a$ to indicate whether an application $a$ uses a link $ij \in L$.

$$b_{ij}^a = \begin{cases} 1 & \text{if application } a \text{ flows over link } ij \\ 0 & \text{otherwise} \end{cases}$$

We should also ensure that a path $p \in P$ is selected only if an application $a \in A$ is admitted to this path.

$$z_p^a \leq y_p \quad \forall a \in A \quad \forall p \in P \quad (21)$$

One constraint should ensure that an application $a \in A$ is admitted only if this application is sent over a certain path $p \in P$

$$z_p^a \leq x_a \quad \forall a \in A \quad \forall p \in P \quad (22)$$

Furthermore, an application $a$ is selected if it is admitted to at least one path $p \in P$.

$$x_a \leq \sum_{p \in P} z_p^a \quad \forall a \in A \quad (23)$$

Moreover, constraint (24) ensures that a path $p \in P$ is selected if at least one application $a \in A$ selects this path.

$$y_p \leq \sum_{a \in A} z_p^a \quad \forall a \in A \quad \forall p \in P \quad (24)$$

Another constraint is to guarantee that if an application $a$ is traversing a path $p \in P$ and link $ij \in L$ is part of $p$, then this application should be arriving at link $ij$.

$$z_p^a \leq b_{ij}^a \quad \forall a \in A \quad \forall p \in P \quad (25)$$

- Delay constraints: To guarantee the delay requirements $T_a$ of each application $a$, we consider the delay experienced by the application flow at each hop in the network. As mentioned previously, we constrain the total delay in terms of the queuing and transmission delay. Given that each IoT application is modeled as M/M/1 queue with an average arrival rate of $\sum_{a \in A} \lambda_a b_{ij}^a$ and service rate of $\frac{C_{ij}}{B_{av}}$, the system delay is given as follows:

$$\sum_{ij \in L} \left( \frac{1}{B_{av}} - \sum_{a \in A} \lambda_a b_{ij}^a \right) \leq T_a \quad \forall a \in A \quad (26)$$

Further, to avoid congestion at each IoT application, the service rate of each queue should remain stable. Then, the average service rate of an outgoing link $\frac{C_{ij}}{B_{av}}$ of a certain node should be greater than the aggregate arrival rates of all application $\sum_{a \in A} \lambda_a b_{ij}^a$ routed through that link as follows:

$$\frac{C_{ij}}{B_{av}} - \sum_{a \in A} \lambda_a b_{ij}^a > 0 \quad \forall ij \in L \quad (27)$$

- Reliability constraint: Constraint (26) ensures that the achieved reliability for each application $a$ while routed over the selected paths should be greater than or equal to the required application reliability $x_a$. The reliability of the whole path $p \in P$ is expressed as the product of the reliability of all the links $\prod_{ij \in p} r_{ij}$ in this path.

$$1 - \prod_{p \in P} \left( 1 - \sum_{ij \in p} r_{ij} \right) \geq x_a \quad \forall a \in A \quad \forall p \in P \quad \forall ij \in L \quad (28)$$

Constraints (26) and (28) are linearized to obtain a linear optimization model. Linearization of these two constraints is omitted for brevity.

VI. SIMULATION RESULTS AND PERFORMANCE ANALYSIS

In this section, we compare the performance of NS-URLLC-MIP and NS-URLLC-DA through extensive numerical evaluation under different system parameters. We present simulation results that show the performance of our solution in terms of network resources that allows meeting the delay and reliability requirements for all admitted applications. The formulated mixed integer optimization problems are solved using Matlab's branch and bound optimization toolbox and tested under different system parameters.

To evaluate both approaches, a factory network with $N = 8$ nodes interconnected with $L = 11$ links is considered. Each link $ij$ in the network has a reliability $r_{ij}$ that is randomly generated between $[0.9 \ 0.99]$ in addition to a capacity between $[1000 \ Mbps \ - 2000 \ Mbps]$. Further, a random number of applications for each type are generated by different devices with given latency and reliability thresholds. The arrival rates of each application type are drawn from Table I. We start by evaluating the optimality and scalability of the proposed decomposition approach NS-URLLC-DA with respect to the general formulation of NS-URLLC. We then analyze the reliability and latency trade-off for the given set of applications. Finally, we consider varying the reliability and latency threshold for one type of application and analyse the effect of these two metrics on the admission rate over a given network.

A. NS-URLLC-MIP vs. NS-URLLC-DA

We evaluate the performance of the decomposition approach as compared to the optimal solution in terms of execution time and optimality. We increase the number of application flows generated by devices. Obviously, increasing the number of applications leads to increasing the aggregate arrival rates at the ingress node. Figure 4 depicts the number of application flows admitted to the network. Since the decomposition approach identified two independent subproblems, it is expected that NS-URLLC-DA achieves the optimal solution. This is verified in Figure 4 that shows that NS-URLLC and NS-URLLC-DA achieve identical results. The E2E-DP only selects the disjoint paths giving an optimal number of available paths. The solution of this first subproblem provides an input for the second subproblem that solves the main problem of flow assignment for the URLLC applications.

Figure 5 presents the execution time (sec) of both approaches. As the number of application flows increase over the network, the execution time of NS-URLLC-MIP increases at a significantly higher rate than NS-URLLC-DA. Hence, the latter is verified to be a more scalable approach.
Fig. 4: Percentage of admitted requests when varying the number of applications

Fig. 5: Execution time (sec) when increasing the number of applications

B. NS-URLLC-DA Performance Analysis

Being a more efficient solution approach, NS-URLLC-DA is further evaluated against various network parameters. We consider between 6 and 40 application randomly selected from the four given application types in Table I, while varying the network size between 8 and 30 nodes interconnected with links ranging between [11 - 60]. Obviously, increasing the network size may increase the chance of having more available paths in the network. We first use the end-to-end disjoint paths (E2E-DP) subproblem to determine the possible paths from source to destination. Then, we run the second subproblem (TFA-URLLC) over the determined paths to assign the traffic flows of the URLLC applications. Figure 6 shows the number of admitted application requests for each of the four types as the number of available disjoint paths increases. The number of admitted applications increases when the network size increases; and this is due to the higher number of available disjoint paths in a larger network. For example, the number of admitted alarm signal requests increases from 60% to 78% when the network size increases from 8 nodes and 11 links to 30 nodes and 60 links. Furthermore, results show that applications with stricter reliability and latency requirements such as alarm signals are less admitted to the network than other applications with relatively looser requirements such as augmented reality. This shows the effectiveness of increasing the number of disjoint paths in admitting more applications through fulfilling their latency and reliability requirements.

Fig. 6: Number of admitted applications with varying network sizes

Furthermore, we examine the number of selected paths for all admitted applications types. Figure 7 presents the number of paths selected for the four application types with different network sizes. The number of utilized paths increases with the increase in the network size to accommodate for a higher number of applications.

Fig. 7: Number of selected paths for the given applications
Further, we then consider the effect of varying the delay threshold on the admission rate of applications to a network with 8 nodes. We consider $A = 100$ applications of AR type with arrival rate and size as per Table I. We vary the delay requirement of the AR applications between 1 and 20 ms and set the reliability target to 0.99. The results in Figure 8 show that the percentage of admitted application flows increases as the delay limit is relaxed.

![Fig. 8: Admitted AR applications with varying delay requirement](image)

C. Summary of Results

In this subsection, we summarize the major findings derived out of the results. In fact, results show that, applications with stricter reliability requirements are assigned more paths in the network to ensure the desired reliability constraint. Stricter latency constraints require larger amounts of bandwidth to be allocated to ensure a timely delivery of packets. This being said, large amounts of resources in terms of bandwidth as well as network paths are required to support ultra reliable and low latency applications. Our proposed decomposition approach generates an efficient solution of flow assignment for URLLC applications with given arrival request rate while maximizing the total number of admitted applications.

![Fig. 9: Admitted AR applications with varying reliability requirement](image)

Furthermore, we consider the admission rate behavior as the reliability threshold gets stricter. We vary the AR reliability limit in the range $[1 - 10^{-1}, 1 - 10^{-6}]$ and plot the resulting admission rate in Fig. 9. A steep decrease in the number of admitted flows as the reliability requirement increases is depicted.

![Fig. 9: Admitted AR applications with varying reliability requirement](image)

VII. Conclusion

In this paper, we studied the problem of traffic flow assignment of applications with ultra-reliable and low latency requirements to maximize the total number of admitted applications. We formulated the problem as a mixed integer program then decomposed it into two subproblems to offer a more scalable version. The decomposed approach demonstrated significant improvement in scalability in comparison with the general problem formulation. Simulation results are presented to analyze the tradeoff behavior among various system parameters including number of applications, network paths, latency, and reliability. Based on our findings, we identified interesting extensions to this work including dynamic flow assignment that considers variable arrival requests of applications with strict reliability and latency requirements. An intelligent edge controller can be employed to learn the behaviour of the arriving application requests and maximize the number of admitted ones while meeting their reliability and latency constraints.

REFERENCES


