On limiting Delay and Jitter characteristics at application-layer of Multi-connected Systems

Megha Sahu[1], Sri Pramodh Rachuri[1], Ahtisham Ali Ansari[1], Deepaknath Tandur[2] and Arzad A. Kherani[1]

[1]Department of Electrical Engineering and Computer Science, Indian Institute of Technology Bhilai, India
[2]ABB Corporate Research, Bangalore, India

***Abstract*—This paper addresses the challenge of limiting the delay and jitter characteristics in a communication system under a multi-connectivity setup. We consider an end-to-end system where the traffic source is connected to the network via multiple wireless interfaces and the application is sensitive to the perceived delay and jitter. We consider both the delay and jitter characteristics for each individual link with a varying degree of channel condition and scheduling requirements. Several challenges are identified for the application responsible for splitting and aggregating the traffic. In this system there is no cellular network coordination for the interface association to different available cellular base stations. The paper provides simulation and empirical results along with analytical models for understanding the implications of jitter and delay parameters in multi-connectivity setup. The work proposes a method to select the received packets such that both delay and jitter are limited in order to improve the overall system performance.**

***Index terms*—** Multi-connectivity, delay, Jitter

I. Introduction

Multi-connectivity refers to the end-user utilizing multiple communication links in order to improve the communication performance. These systems are the key solutions for reliable transmission and provides enhanced data rates by using multiple links from source to destination. There are several commercial solutions available in the market where the control of traffic into the various available links is achieved by using a Software-Defined Wide Area Network (SD-WAN) approach [1], [2], where virtualization techniques are used in order to expand or shrink the network bandwidth dynamically. The main focus of SD-WAN is to provide efficient services with lower costs. The success of such solutions leads one to wonder about using multiple connectivity with cellular links. Multi-connected systems can also make efficient use of different wireless technologies by connecting to 3G, 4G, WiFi,etc., thereby supporting multi-Radio Access Technology (multi-RAT) [3]. This fulfills the need for more capacity and improves reliability as well. These systems can be the driver for fulfilling the requirements of 5G mobile network as well. [4] includes one such work for 5G from the network's energy efficiency perspective.

The emergence of SD-WAN on fixed (enterprise) operators has had significant effects on the market for MPLS VPNs [2]. SD-WAN enables businesses to bond several standard Internet Service Provider (ISP) broadband lines as well as small-capacity Multi-Protocol Label Switching (MPLS) links. The main advantage being the low overall cost when compared to a higher cost involved in an MPLS-like setup. In SD-WAN, the management of multiple interfaces can be performed at different OSI model layers, see [5]. This ensures the traffic encapsulation can be implemented from Medium Access Control (MAC) to the application layer [5]. However, most of the multi-access SD-WAN solutions available in the market [2] are proprietary in nature. The solutions utilize some sort of policy management by a centralized controller. These solutions mainly use the multi-access capability to increase the throughput performance of the system. However, one can also look at the problem of improving end-to-end delay and jitter performance for applications that require such guarantees. Many real time multimedia services like voice and video are delay and jitter sensitive and demand for good control in these two parameters to have the best quality output. Also, many applications installed in power generation plants, electrical substations, and control centres requires immediate actions in case of any emergency or abnormal situations. Such applications may generate very small traffic volume, but require a stricter delay or jitter performance requirements. Thus depending upon the Quality of Service (QoS) requirement few applications are listed in Table I.

A cellular based User-Equipment (UE) (3G/4G) device is usually served by a single network link, but in multi-connectivity scenario, a UE may be simultaneously connected to more than one link at a time. The links are independent in their characteristic with different network infrastructure resources provided to it at any instant of time. Efficient utilization of these resources and links with appropriate amount of traffic is thus necessary. The focus in multi-connectivity is to operate and choose these

Application	Delay (ms)	Jitter (ms)	Packet loss
Voice/Video call [6]	150	30	10^{-2}
Teleprotection [7]	10	0.2	10^{-9}
Substation Automation [8] [9]	50	5	10^{-9}
Factory Automation [8] [9]	100	10	10^{-9}
Process Automation [8] [9]	1000	300	10^{-5}

TABLE I: Applications with performance specifications limits.

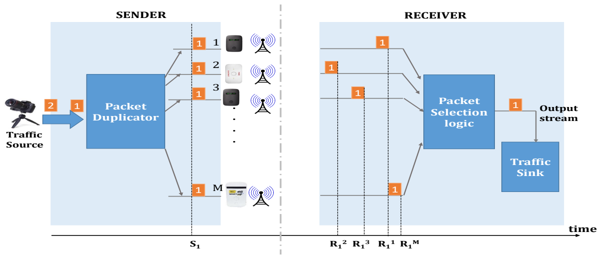


Fig. 1: A Multi-connectivity Setup.

links in a way which will lead to achieve the desired performance characteristics. Multiple links also provide the added benefit of redundancy during link fail-overs. At the receiver, the diversity in channel conditions is utilized to remove the redundancy, thereby achieving higher reliability. Such adjustments and selection of links at the UE leads to achieve appropriate levels of delay, jitter, and packet-loss depending on the application or traffic type.

In this paper we explore the viability of cellular link based multi-connected systems to limit the delay and jitter characteristics. The challenge is in managing the varying channel and network conditions at individual link level. The selection of links should then be such that we can meet the application QoS priorities. Monitoring and understanding these variations are important as the instability in channel conditions impacts the delay and jitter performance. We first analyze both delay and jitter profile in these conditions and then the solution is proposed for limiting the delay and jitter in multi-connectivity setup. The paper is structured as follows. Section II gives a brief idea about delay and jitter under multi-connectivity and share the simulation results to highlight the challenges using greedy approach. Section III covers the end-to-end delay and jitter analysis under the effect of uplink scheduling and Hybrid-Automatic Repeat Request (HARQ) mechanism. In Section IV, we propose an algorithm to improve the jitter performance in multi-connectivity and compare with the greedy approach using simulation and experimental results. In the last Section V we conclude our work.

II. Delay and Jitter under Multi-connectivity

Figure 1 represents a multi-connectivity setup, where a sender system is connected to multiple cellular modems of same or different service providers (like Reliance Jio, Airtel, Idea, etc). The focus is on the uplink transmission from sender to base station (eNB) in Long Term Evolution (LTE) network[10]. At the sender end, the traffic generated from the source is distributed across the available interfaces. Let M be the total number of interfaces connected to the sender system. The traffic source can be a stored media file/camera (live streaming) or a free running packet generator. Here, we duplicate each packet generated by the traffic source and transmit them across all the available interfaces (M) concurrently. This leads to multiple instances of the same packet being fed into the packet selection block at the receiver. We assume no packets are lost over any interface during transmission. The receiver is fixed with static public IP. The packet selection block at the receiver consists of the logic using which an interface is selected for a packet to be played out in the output stream.

In this section we study the $delay$ and $jitter$ performance metrics of the stream coming out of the packet selector block of Figure 1; this stream will be referred to as the *output stream*. This is done under a *greedy approach* (defined in Section II-A) of packet selection logic used in the block. The objective of this study is to understand whether we can simultaneously reduce the delay and jitter of the output stream by intelligently selecting packets arriving over different interfaces. We first define the delay and jitter of the output stream.

Let S_i be the timestamp associated with the i^{th} packet generated by the traffic source, and R_i^m be the receive timestamp of i^{th} generated packet transmitted over the m^{th} interface (see Figure 1).

Then, the *per-packet* end-to-end delay and jitter are defined as, $D_i^m = R_i^m - S_i$ and $J_i^m = |(R_i^m - R_{i-1}^m) - (S_i - S_{i-1})|$. The (moving) average delay and jitter associated with the k^{th} received packet on the m^{th} interface are given by,

$$\tilde{D}_k^m = \frac{1}{16} \sum_{i=k-15}^{k} (R_i^m - S_i), \tag{1}$$

$$\tilde{J}_k^m = \frac{1}{16} \sum_{i=k-15}^{k} |(R_i^m - R_{i-1}^m) - (S_i - S_{i-1})|. \tag{2}$$

Here, the average delay and jitter are calculated using a moving average with a subset of 16 packets [11].

Let $m^*(i)$ be the interface selected for the i^{th} source packet and let $m = 0$ be a special virtual interface for the overall playout of the stream. Then, $R_i^0 = R_i^{m^*(i)}$ and the average delay and jitter for the output stream are given by,

$$\tilde{D}_k^0 = \frac{1}{16} \sum_{i=k-15}^{k} (R_i^{m^*(i)} - S_i), \tag{3}$$

$$\tilde{J}_k^0 = \frac{1}{16} \sum_{i=k-15}^{k} |(R_i^{m^*(i)} - R_{i-1}^{m^*(i-1)}) - (S_i - S_{i-1})|. \tag{4}$$

A. The Greedy Algorithm

In this Greedy Algorithm (GA), the packet selection block simply plays out the first packet received amongst all the interfaces so that $R_i^{m^*(i)} = min(R_i^1, R_i^2, .., R_i^M)$. The average delay and jitter for the output stream are obtained by replacing $R_i^{m*(i)}$ to $min(R_i^1, R_i^2, .., R_i^M)$ in Equation 3 and 4. Then, we observe, $\tilde{D_k^0} \leq \tilde{D_k^m}\ \forall\ k \geq 1,\ m \in [1, M]$. This clearly indicates the average delay for the output stream can be decreased by using the GA, but jitter in this regard needs to be analyzed further since a good control on average jitter is not evident (from Equation 4).

B. Sample-path based study and observations

We now consider a sample-path approach to see what happens to average jitter when the receive timestamp of only one packet on one interface is changed while keeping all the other packet's sender and receive timestamp the same. Clearly, to study the impact of such change on jitter, we need to consider the sender and receive timestamps of the packets sent immediately before and immediately after this packet.

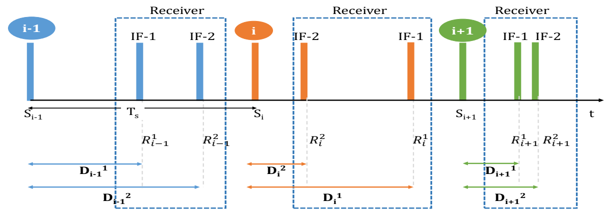


Fig. 2: Change in receive timestamp of an IP packet.

Let us consider a Constant Bit Rate (CBR) traffic with packet size of $P(Bytes)$ generated at every $T_s(ms)$ time interval, producing Rate(R) = $P \times 8/T_s\ (bps)$. This implies $(S_i - S_{i-1}) = T_s\ \forall\ i$. Let $M = 2$ with interface names, IF-1 and IF-2. We consider the three consecutive IP packets generated and their receive timestamps across IF-1 and IF-2 are shown in Figure 2. Here, all the packets are received first on IF-1 except for i^{th} packet (received first on IF-2). Then the contribution of per-packet jitter due to this (i^{th}) and the next received packet in the average jitter using GA is $\frac{1}{2} \cdot (|D_i^2 - D_{i-1}^1| + |D_{i+1}^1 - D_i^2|)$. If we see by making a sample-path change and assuming that all the packets are received first on IF-1 or IF-2, then $E[J^0] = E[J^m]\ for\ m = 1\ or\ 2$.

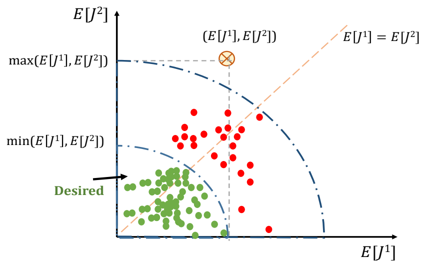


Fig. 3: Representing possible points of $E[J^0]$ ($M = 2$).

We implement the Monte-Carlo simulation by considering the changes shown in Figure 2 and for 1000 samples get the $E[J^0]$ using GA. Let us assume R_i^m are uniformly distributed over the interval $[0, 10]$. We observe, around 700 samples have $E[J^0] \leq min(E[J^1], E[J^2])$, but for 300 samples the performance is bad with $min(E[J^1], E[J^2]) \leq E[J^0] \leq max(E[J^1], E[J^2])$. An illustration is shown in Figure 3. This shows about 30% possibility of adverse effect on the average jitter due to the change in receive timestamps of an IP packet in the output stream (using GA). This motivates us to search for a better algorithm that improves average jitter performance even in such changes.

III. End-to-End Delay and Jitter Analysis

In this section we provide end-to-end delay and jitter analysis in LTE system. In LTE, end-to-end delay and jitter are determined by the uplink scheduling of IP packets and successful transmission, controlled by LTE's HARQ mechanism. Further, we study the behaviour of delay and jitter under the effect of HARQ and significant observations are made towards the end of the Section for the requirements to be considered in multi-connectivity setup.

Let us consider the CBR traffic with one IP packet per source burst generated at every T_s (large) time interval such that every duplicate packet sent into a cellular modem sees LTE Packet Data Convergence Protocol (PDCP) [12] queue empty. The generated packet is assumed to be small enough to be served using a single HARQ process. Let X_i^m and Y_i^m be the time required for scheduling i^{th} IP packet, and its successful transmission on m^{th} interface, respectively. Then the total end-to-end delay of i^{th} packet due to scheduling and its successful transmission in the interface m is, $D_i^m = X_i^m + Y_i^m$.

Under the HARQ transmissions, let $\pi_{k,m}$ be the probability of success in k^{th} HARQ attempt on m^{th} interface. We assume $\pi_{1,m} + (1 - \pi_{1,m}) \cdot \pi_{2,m} = 1$, i.e., a HARQ process is over in at most two HARQ attempts. Generally in LTE, the maximum number of retransmissions is limited to four [10]. The delay and jitter analysis provided in [13], considers the success probabilities to be fixed, but in our work, we account for variation in the channel and assume $\pi_{1,m}$ to be a random variable that is uniformly distributed over an interval $[P_{(m)min}, P_{(m)max}]$.

In LTE FDD, the average HARQ Round Trip Time (RTT) is $8ms$ [10], leading to an additional $8ms$ in the total delay for each HARQ retransmission. Hence, Y_i^m takes values of $0ms$ (successful in 1^{st} HARQ attempt) or $8ms$ (successful in 2^{nd} HARQ attempt) with probabilities, $\pi_{1,m}$, and $(1-\pi_{1,m})$, respectively. Then the expected value of Y_i^m, accounting for randomness in $\pi_{1,m}$, is,

$$\sum_{i=1}^{k} Y_i^m E[Y_i^m|\pi_{1,m}] = 8\cdot(1-\pi_{1,m}), \tag{5}$$

$$E[Y_i^m] = 8\cdot\left[1-\frac{P_{(m)max}}{2}\right]. \tag{6}$$

Here, we have assumed $P_{(m)min}=0$. Considering the other component, X_i^m, of the delay over m^{th} interface, to have an average value of $\frac{1}{\lambda_m}$ (assuming X_i^m exponentially distributed with λ_m), the average end-to-end delay is given by,

$$E[D_i^m] = \frac{1}{\lambda_m} + 8\left[1-\frac{P_{(m)max)}}{2}\right]. \tag{7}$$

Similarly, the average jitter due to HARQ is given by,

$$E[|Y_i^m - Y_{i-1}^m| \mid \pi_{1,m}] = 2\cdot 8\cdot\pi_{1,m}\cdot(1-\pi_{1,m}) \tag{8}$$

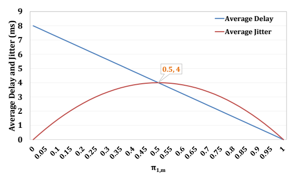
$$E[|Y_i^m - Y_{i-1}^m|] = 2\cdot 8\cdot P_{(m)max}\left[\frac{1}{2}-\frac{P_{(m)max}}{3}\right] \tag{9}$$

The average end-to-end jitter obtained due to both scheduling and HARQ is given by,

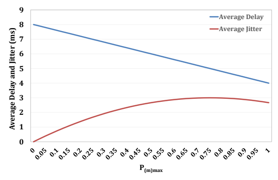
$$E[J_i^m] = \frac{1}{\lambda_m} + 2\cdot 8\left[\frac{P_{(m)max}}{2}-\frac{P_{(m)max}^2}{3}\right]. \tag{10}$$

Equation 7 and 10, shows the average end-to-end delay and jitter depends on parameters λ_m and $P_{(m)max}$ of X_i^m and $\pi_{1,m}$.

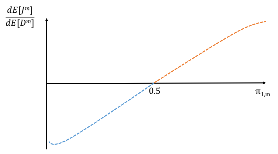
Further, the delay and jitter analysis is continued by considering the effect of HARQ only and therefore we assume $E[D^m]=E[Y_i^m]$ and $E[J^m]=E[|Y_i^m - Y_{i-1}^m|]$. Now, we see the average delay and jitter performance under the effect of HARQ mechanism.
Figure 4a and 4b gives average delay and jitter profile for an interface independent of all the other interfaces connected. In Figure 4a, delay and jitter are plotted against $\pi_{1,m}$ using Equation 5 and 8, and in Figure 4b against $P_{(m)max}$ using Equation 6 and 9. In Figure 4a, we observe for bad channel conditions (lower values of $\pi_{1,m}$), the delay decreases, and jitter increases till



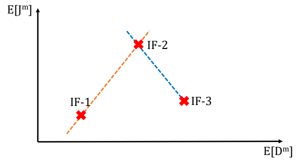
(a) Variation with $\pi_{1,m}$



(b) Variation with $P_{(m)max}$



(c) $\frac{dE[J^m]}{dE[D^m]}$ vs $\pi_{1,m}$



(d) $(E[D^m], E[J^m])$ for $M=3$

Fig. 4: Average Delay and Jitter under the effect of HARQ for an interface.

$\pi_{1,m} = 0.5$, but after this point, both start decreasing. This implies, in better channel conditions (higher values of $\pi_{1,m}$), average delay and jitter performance are good, which generally happens in LTE where the operators set $\pi_{1,m}$ values to at least 0.9 [14]. This is also shown in Figure 4c in terms of slope, $\frac{dE[J^m]}{dE[D^m]}$, where the slope is negative for $\pi_{1,m}\le 0.5$, and positive for $\pi_{1,m} > 0.5$. The average delay and jitter values are equal, and plots intersect at points $\pi_{1,m}=0.5, 1$. Figure 4d shows the multiple $(E[D^m], E[J^m])$ points for the individual interfaces in 2-D space (for $M=3$) depending on the channel conditions. The two regions shown in red and blue dotted lines can be related to Figure 4a and 4c such that $\pi_{1,m}$ values for the interfaces are $\pi_{1,3}\le\pi_{1,2}\le\pi_{1,1}$. An optimum $(E[D^0], E[J^0])$ is needed for the overall multi-connected system, and a better solution to get that optimum point is required in multi-connectivity.

Let us evaluate the $E[D^0]$ and $E[J^0]$ for the final output stream using GA (II-A) for $M=2$. Here, delay is $0ms$ or $8ms$ with probabilities $P_r(8)=(1-\pi_{1,1})(1-\pi_{1,2})$ and $P_r(0)=1-(1-\pi_{1,1})(1-\pi_{1,2})$ respectively. The average delay due to HARQ for the output stream is given by,

$$E[D^0|\pi_{1,1},\pi_{1,2}] = 8\cdot(1-\pi_{1,1})\cdot(1-\pi_{1,2}), \tag{11}$$

$$E[D^0] = 2\cdot\left[P_{(1)max}-2\right]\left[P_{(2)max}-2\right]. \tag{12}$$

Similarly, the average jitter with randomness in $\pi_{1,1}$ and $\pi_{1,2}$ is given by,

$$\begin{aligned} E[J^0|\pi_{1,1},\pi_{1,2}] = 2\cdot 8\cdot(1-\pi_{1,1})(1-\pi_{1,2}) \\ (1-(1-\pi_{1,1})(1-\pi_{1,2})). \end{aligned} \tag{13}$$

Figure 5 provides a 3-D plot for the average jitter against the instantaneous values of $\pi_{1,1}$ and $\pi_{1,2}$. The

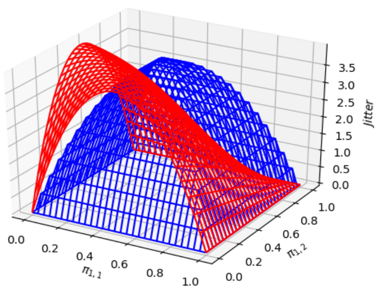


Fig. 5: 3-D plots for $E[J^0]$ (red) and min($E[J^1]$,$E[J^2]$) (blue) against $\pi_{1,1}$ and $\pi_{1,2}$.

plots are $E[J^0]$ (Equation13) and $min(E[J^1], E[J^2])$. We observe that if both the interfaces have good channel conditions then $E[J^0] < min(E[J^1], E[J^2])$. When any one of them is bad, then $E[J^0] = min(E[J^1], E[J^2])$, and in case of both performing bad (rare in practice), the jitter performance degrades. This implies in GA the overall jitter performance is *improved* when at least one of the interface has a good channel condition.

It is understood that the above model may not be the statistical equivalent of the real-world processes; however, it gives a good insight into the dependence of delay and jitter on the various phenomenon involved in the diverse interfaces in the multi-connectivity setup. For example, $\pi_{1,m}$ provides an abstraction of the channel conditions on the various interfaces, while the initial delay X_i^m provides an abstraction for the uplink scheduling delay, which depends on the overall load on the base station.

IV. A Dynamic Windowing Algorithm

In this Section, we provide an intuitive packet selection logic to play out the final stream, demonstrating the possibility of using the instantaneous performance of multiple links to improve the overall jitter performance. The interface $m^*(i)$ for the i^{th} packet is selected in such a way that the per-packet jitter for the i^{th} packet in the output stream is as small as possible. This is achieved by selecting the received packet from the interface $m^*(i) = argmin \mid (R_i^m - R_{i-1}^{m^*(i-1)}) - (S_i - S_{i-1}) \mid$.

Working of this Dynamic Window Algorithm (DWA) is shown in Figure 6 where the arrival instants of the multiple copies of i^{th} packet received from multiple interfaces ($M = 4$) are shown. The marked packet is selected to play out as it is nearest to the delay of the previous packet (D_{i-1}). This is how the algorithm tries to reduce the per packet jitter values. One can observe that, due to causality requirements, one may not be able to select the best packet always, i.e., the decision on whether a packet needs to be sent to the output stream should happen soon after receiving the packet.

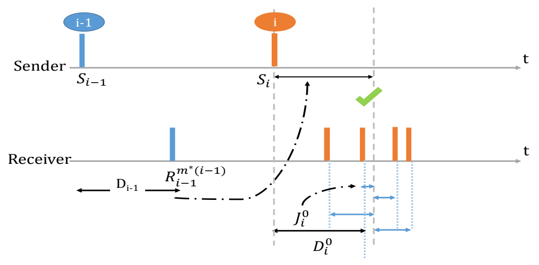
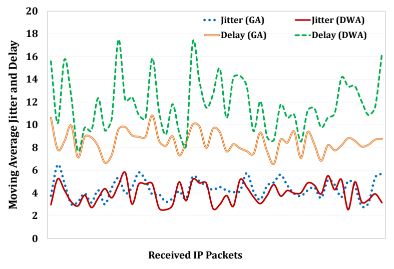


Fig. 6: Packet Selection in DWA ($M = 4$).

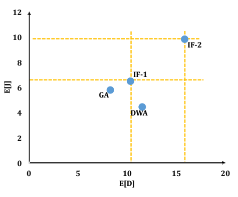
A. Simulation Results

A python based simulation is implemented for both GA and DWA. Let $M = 2$ and R_i^m on interface m to be uniformly distributed over the interval $[0, \theta_m]$ with $\theta_1 = 20$ and $\theta_2 = 30$. We implement for 1000 samples.

The moving average delay and jitter for the output stream are obtained for the subset of 16 samples and represented in Figure 7a. Figure 7b also represents $(E[D^m], E[J^m])$ points for the two received streams $(m = 1, 2)$ and the output stream $(m = 0)$ for the two algorithms. Here, $E[J^0]$ is 5.91 and 4.49 for GA and DWA. $E[D^0]$ is 8.52 and 11.50 for GA and DWA. It is observed that the average jitter using DWA is



(a) Moving Average Delay and Jitter plots.



(b) $E[J]$ vs $E[D]$.

Fig. 7: Delay and Jitter for output steam using GA and DWA ($M = 2$).

improved as compared to the jitter of GA. Jitter is reduced by $\approx 1.5ms$ as shown in Figure 7b. However, this improvement in jitter comes at the expense of an increased delay ($3ms$). The Delay-Jitter tradeoff is not new and is a key aspect in designing a playout buffer in a single interface system. The difference here is that we are talking about the delay and jitter of the stream *into* the playout buffer (the output stream of the Packet Selector block is fed into a playout buffer for further application-specific processing).

B. Experimental Results

We have implemented the algorithms in our real-world experimental setup described in [15]. At higher level the reference can be drawn from a multi-connectivity setup shown in Figure 1. The experiments are performed with $M = 2$ and a CBR traffic with $P = 500 Bytes$, $T_s = 40ms$.

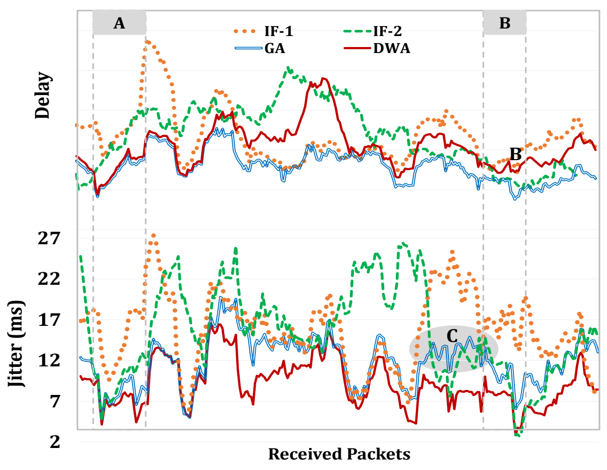


Fig. 8: Moving Average Delay and Jitter for individual interfaces ($M = 2$) and output stream using GA and DWA.

Figure 8 shows the moving average delay and jitter for $m = 1, 2$, and the output stream (m=0) using GA and DWA. They are measured in the unit of microsecond, but exact delay values are not mentioned in the vertical axis because of lack in time synchronization between the two end machines in our set-up. However, since the synchronization error is the same for the two setup, we can compare the delay results for the two algorithms qualitatively. From Figure 8 following observations are made :

1) Initially, delay in IF-1 is more than the delay in IF-2 and then becomes less than IF-2, possibly a change in channel condition or load on the serving eNB led to reduced uplink allocations.
2) If both the interfaces have similar channel conditions, implies the delay experienced by them (IF-1 and IF-2) almost same ($\tilde{D}_k^1 \approx \tilde{D}_k^2$) then the jitter in the output stream decreases (see region A and B).
3) The jitter for GA performs bad at few points; like in region C, it becomes more than $\tilde{J}_k^2$. But jitter for DWA is always below or equal to the $min(\tilde{J}_k^1, \tilde{J}_k^2)$ $\forall\ k \geq 1$.
4) The delay in case of GA is always below or equal to $min(\tilde{D}_k^1, \tilde{D}_k^2)$ $\forall\ k \geq 1$, but for DWA, the delay is significantly high.

We observe, by using GA, the performance of delay improves, but jitter performance is bad. Similarly, the jitter in the case of DWA is better than the GA, but the delay becomes very high. This is similar behavior as deduced from the simulation of our model in Figure 7.

V. Conclusion

We have presented a simple analysis and modelling for delay and jitter under the multi-connectivity set-up. We have validated our points raised in multi-connectivity set-up and highlighted challenges on moving from single to multiple network links by simple simulations. We observe delay and jitter cannot be improved together at a time, and there always exists a trade-off. But the solution is possible where both metrics can be limited and we can achieve the minimum possible delay and jitter for the overall multi-connectivity setup. We have studied two such natural algorithms as a solution to define delay and jitter in case of a multi connected system. The comparative study of these results have been presented in the paper.

References

[1] J. Gintert and W. Dynamics, "How Software Defined Wide Area Networking (SD-WAN) provides reliable voice and video services over the internet," *IEEE Softwarization*, 03 2018.
[2] [Online]. Available: https://www.networkworld.com/article/3284367/10-hot-sd-wan-startupsto- watch.html
[3] "Technical specification group radio access network; Evolved Universal Terrestrial Radio Access (E-UTRA) and NR; multi-connectivity; stage 2 (release 15)." *3GPP TS 37.340 version 15.3.0 Release 15*, 2017.
[4] V. Poirot, M. Ericson, M. Nordberg, and K. Andersson, "Energy efficient multi-connectivity algorithms for ultra-dense 5G networks," *Wireless Networks*, 06 2019.
[5] P. S. Schmidt, R. Merz, and A. Feldmann, "A first look at multi-access connectivity for mobile networking," *Proceedings of ACM Workshop on Capacity Sharing*, pp. 9–14, 2012.
[6] C. Lewis and S. Pickavance, "Implementing Quality of Service over Cisco MPLS VPNs," *CISCO Press*, 2007.
[7] R. Bachli, M. Hausler, and M. Kranich, "Teleprotection solutions with guaranteed performance using packet switched wide area communication networks," *CPRE*, pp. 1–6, 04 2017.
[8] M. Gidlund, T. Lennvall, and J. Akerberg, "Will 5G become yet another wireless technology for industrial automation?" *IEEE ICIT*, pp. 1319–1324, 03 2017.
[9] T. Lennvall, M. Gidlund, and J. Akerberg, "Challenges when bringing IoT into industrial automation," *IEEE AFRICON*, pp. 905–910, 09 2017.
[10] N. Abu-Ali, A. E. M. Taha, M. Salah, and H. Hassanein, "Uplink Scheduling in LTE and LTE-Advanced: Tutorial, Survey and Evaluation Framework," *IEEE Communications Surveys Tutorials*, pp. 1239–1265, 03 2014.
[11] I. Marsic, *Computer Networks Performance and Quality of Service*, Rutgers University, NewJersey, 2013.
[12] "LTE; Evolved Universal Terrestrial Radio Access (E-UTRA); Packet Data Convergence Protocol (PDCP) specification," *3GPP TS 36.323 version 14.3.0 Release 14*.
[13] A. Duc, P. Ciblat, and C. Le Martret, "Delay and jitter closed-form expressions for Cross-Layer Hybrid ARQ Schemes," *VTC, 1988, IEEE 38th*, pp. 1 – 5, 10 2009.
[14] M. Woltering, D. Wubben, A. Dekorsy, U. Doetsch, and V. Braun, "Link level performance assessment of reliability-based HARQ schemes in LTE," *IEEE VTC2014-Spring*, vol. 2015, 05 2014.
[15] S. Damle, M. Sahu, and A. A. Kherani, "An uplink multi-access system for high definition video transmission," *COMSNETS*, pp. 532–534, 2019.