# Cross-layer congestion control and quality of services in mobile networks

Zhenzhe Zhong

INSTITUT POLYTECHNIQUE DE PARIS

NNT : 2020IPPAT022

Thèse de doctorat

# Cross-layer congestion control and quality of service in mobile networks

Thèse de doctorat de l'Institut Polytechnique de Paris
préparée à Telecom Paris

École doctorale n°626 École Doctorale de l'Institut Polytechnique de Paris (IP Paris)
Spécialité de doctorat : Informatique, Données et Intelligence Artificielle

Thèse présentée et soutenue à Palaiseau, le 15/07/2020, par

**ZHENZHE ZHONG**

Composition du Jury :

| | |
|---|---|
| Isabel Amigo<br>Assistant Professor,IMT Atlantique | Président |
| M.Lyes Khoukhi<br>Professeur,University of Technology of Troyes | Rapporteur |
| M.Pascal Lorenz<br>Professeur, University of Haute Alsace | Rapporteur |
| Dominique Gaiti<br>Professeur, University of Technology of Troyes | Examinateur |
| Stéphane Tuffin<br>Gestionnaire de projet, Orange Labs | Examinateur |
| Isabelle Hamchaoui<br>ingénieur de recherche senior, Orange Labs | Directeur de thèse(CIFRE) |
| Ahmed Serhrouchni<br>Professeur, Telecom ParisTech | Directeur de thèse |
| Rida Khatoun<br>Associate Professor, Telecom ParisTech | Co-directeur de thèse |

# Table of contents

# List of figures

# List of tables

# List of Algorithms

# List of Abbreviations

$BW_{Btlnck}$ Bottleneck Bandwidth

$RTT_{min}$ Minimum Round Trip Time

3GPP 3rd Generation Partnership Project

ACK Acknowledgement

ADD Asymmetric Deep DRAIN

ADD asymmetric Deep Drain

AM Acknowledgement mode

AM Client Driven Bandwidth Estimation

AMC Active Queue Management

AMC Adaptive Modulation and Coding

APP APPlication(Layer)

BBR Bottleneck Bandwidth and RTT

BIF Bytes In Flight

BLWR Block Error Rate

BN Bottleneck

BS Base Station

## List of Abbreviations

BW Bandwidth

BWE Bandwidth Estimation

CCA Congestion Control Algorithm

CDBE Client Driven Bandwidth Estimation

CDBEv2 Client Driven Bandwidth Estimation version 2

CDF Cumulative distribution function

CQI Channel Quality Index

CQIC CQI Congestion Control

CWND Congestion WiNDow

DCI Downlink Control Information

DD Deep DRAIN

DD Deep Drain

DL (Mobile) Downlink

DS Download Stream/Flow

DTX Discontinuous Transmission

DUPACK Duplicated ACK

E2E End-to-End

ECN Explicit Congestion Notification

EDGE Enhanced Data for GSM Evolution

EMM EPS Mobility Management

EPC Evolved packet core

EPS Evolved packet system

ESM EPS Session Management

ETSI European Telecommunications Standards Institution

EUTRAN Evolved Universal Terrestrial Radio Access Network

FIFO First in, First Out

FQ Fair Queue

FQM Fair Quit Method

FRecv Fast Recovery

FReTx Fast Retransmission

GC Growth Compensation

GC growth compensation

GPRS General Packet Radio Service

GSM Global system for Mobile

GTP GPRS Tunneling Protocol

HARQ Hybrid Automatic Repeat ReQuest

HSDPA High Speed Downlink Packet Access

HSPA High Speed Packet Access

IETF Internet Engineering Task Force

IP Internet Protocol

ISG Industry Specification Group

ITCP Indirect-TCP

ITU International Telecommunication Union

KB Kilo-Bytes

Kb Kilo-bits

KBps Kilo-Bytes per second

Kbps Kilo-bits per second

L1 Layer 1(Physical layer)

**List of Abbreviations**

L2 Layer 2(MAC layer)

LEBAT Low Extra Delay Background Transport

LTE Long-Term Evolution

M-TCP Tcp for Mobile cellular networks

MAC Medium Access Control

MBps Mega-Bytes per second

Mbps Mega-Bits per second

MBR Maximum Bit Rate

MCS Modulation and Coding scheme

METP Mobile End Transport Protocol

MIMO Multiple Input Multiple Output

MME Mobile Management Entity

MSR Mobility Support Router

MTCP Mobile TCP

NACK Negative-ACK

NAS Non-Access Stratum

NIN Non-IP-Networking

NS3 Network Simulator 3

OFDM Orthogonal Frequency Division Multiple

OFDMA Orthogonal Frequency Division Multiple Access

PDCP Client Driven Bandwidth Estimation

PDN Packet Data Network

PEP Performance Enhancement Proxy

PF Proportional Fair

PGW Packet Data Network Gateway

PHY PHysical layer

piStream Physical layer informed adaptive video streaming over lte

Pkt Packet

PLR Packet Loss Ratio

QCI QoS Class Identifier

QoE Quality of Experience

QoS Quality of Service

QUIC Quick UDP Internet Connections

RAN Radio Access Network

RB Resource Block

RBG Resource Block Group

RFC Request For Comment

RLC Radio Link Control

RRC Radio Resource Control

RRM Radio Resource Management

RTO Re-transmission TimeOut

RTT Round Trip Time

SACK Selective ACK

SCTP Stream Control Transmission Protocol

SGW Serving Gateway

SINR Signal to Interference plus Noise Ratio

ssthresh Slow Start Threshold

ST Sequence Trace

TBS Transport Block Size

TCP Transport Control Protocol

TM Transparent Mode

TTI Time Transmission Interval

UCI Uplink Control Information

UDP User Datagram Protocol

UE User Device

UL (Mobile) Uplink

UM Un-Acknowledgement Mode

UMTS Universal Mobile Telecommunications System

US Upload Stream/Flow

WCDMA Wideband Code Division Multiple Access

WebRTC Web Real-Time COmmunication

# List of Publication

## Conference Papers:

- Zhenzhe Zhong, Isabelle Hamchaoui, Rida Khatoun: **Perils of using cqic in lte network and a quick fix with delayed ack**. *–15th IEEE Annual Consumer Communications & Networking Conference (CCNC)*, January 2018, Las Vegas, USA
- Zhenzhe Zhong, Isabelle Hamchaoui, Rida Khatoun, Ahmed Serhrouchni: **Performance evaluation of cqic and tcp bbr in mobile network**. *–21st Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN)*, February 2018, Paris, France
- Zhenzhe Zhong, Isabelle Hamchaoui, Alexandre Ferrieux, Rida Khatoun, Ahmed Serhrouchni: **CDBE: A cooperative way to improve end-to-end congestion control in mobile network**. *–2018 14th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, October 2018, Limassol, Cyprus

## Journal Paper In preparation:

**CDBEv2: toward better end-to-end performance in mobile network**. *–To be submitted to a journal*

# Chapter 1

# Introduction

This thesis aimed to address the congestion control issue in mobile radio access networks with a cross-layer method, and finally proposed a ubiquitous solution for a mobile network.

## 1.1 Context and Objective

The mobile network consists of radio access network with cells to serve user devices, packet core network for traffic delivery and gateways which connect to other parts of the whole internet.

From the second generation (2G)[1] of commercial mobile networks to the widely deployed Long Term Evolution (LTE, also known as 4G)[2] nowadays, the capacity of a mobile radio access network experienced explosive multiplication thanks to the progress of advanced radio techniques. The peak PHY[3] layer capacity has increased from tens of Kilo-bits per second (Kbps) level up to hundreds of Mega-Bits per second (Mbps) level in each Transmission Time Interval (TTI). Furthermore, thanks to the dynamic Modulation and Coding Scheme (MCS), Multiple-Input Multiple-Output (MIMO)[4]. More advanced wireless technologies are applied to fifth-generation(5G) of the Mobile network: massive MIMO antennas, wide spectrum bandwidths, multi-band carrier aggregation, etc.. Plus the corresponded media access(MAC) layer and the Radio Link Control layer on the Radio access network (RAN), the mobile network nowadays can provide the theoretical bandwidth ranging from Kbps up to Gbps in a short period. Such agility allows LTE, 5G and the following generations of mobile networks to support a wide variety of mobile networking applications in daily life.

Apart from Cellular part, the backbone infrastructures are the special type of fixed network with layered design. User data flows from the outside source (remote server or CDN server) to packet delivery network (PDN) gateways (PGW), then finally sent to the user equipment (UE) through RAN. The under-utilisation of RAN is a long-standing problem3.1. The upper

layers which take care of end-to-end traffic transportation in a mobile network should have the adaptability to utilise the ever-mounting capacity better. Our focus is to address the issue from the perspective of congestion control design in transport-layer.

In this thesis, we are going to:

1. understand the reason behind this underutilisation in a mobile cellular network,
2. explore the existing solutions and test their performance in the network
3. design algorithms, protocols or architectures of congestion control to improve the end-to-end performance and analysis the pros and cons of the proposed algorithm

There are three types of design logic in a congestion control algorithm on the transport layer. They are loss-based congestion control, delay-based congestion control and latest bandwidth-delay based congestion control. Concerning the bandwidth utilisation, loss-based method trades the utilisation with loss, as the extreme form of delay in a buffered network. Delay-based methods trade utilisation with different amount of delay. The bandwidth-delay based CCA, as succeeder delay-based methods, balances the bandwidth utilisation and delay by frequent probing and draining operation on the time domain.

Apart from the Transport layer oriented CCAs, there are also several cross-layer attempts to invoke the lower layer information to improve the bandwidth utilisation. In our review, we focus only on the end-to-end cross-layer solutions on a wireless network.

In the current mobile network, the conventional TCP/IP loss-based algorithm is still the main-stream. A new Industry Specification Group Non-IP-Networking (ISG NIN)[5] of European Telecommunications Standards Institute (ETSI) also working toward looking for a replacement of conventional TCP/IP in the 5G core network since we are tired of the bufferbloat phenomenon caused by the loss-based CCA.

The congestion control architecture we are going to design is on an end-to-end basis, which takes the nature of a mobile cellular network into consideration. The assumption is that all the incoming traffic will flow through the PGW, and the PGWs can guarantee the CCA in the whole mobile backbone. Hence we can design a transport layer end-to-end congestion control algorithm to guarantee the fair share of capacity in both mobile backbone and RAN.

These conventional and lately proposed CCAs are the origin of this thesis topic: To find out whether the cross-layer design like CQIC is suitable for the full mobile network. After the validation and the evaluation of CQIC, one the one hand, we need to decide whether to follow the explicit CQIC manner or introduce an implicit bandwidth estimation. On the other hand, the cooperated congestion control architecture, which was not well introduced on the CQIC server, should also be proposed, so that the server has the capability of probe more, relief the congestion and equally share the bottleneck bandwidth.

Hence this PhD thesis aims to study and propose innovative Congestion control mechanism or Architecture to improve network utilisation and customer experience on mobile networks.

The method we are going to follow and the objectives of the thesis are:

- Review the existing CCA algorithms and the features of mobile network and analyse their feature in mobile network
- Review the cross-layer proposals for mobile networks and perform a brief evaluation of these mechanisms,
- Implement and review the latest BBR in NS3[6] simulator as the baseline for the comparison
- Propose and design an innovative CCA architecture for 3G and 4G networks,
- Derive a model and evaluate the performance of these architectures and develop required functions, and
- Implementation and evaluation of the proposed CCA

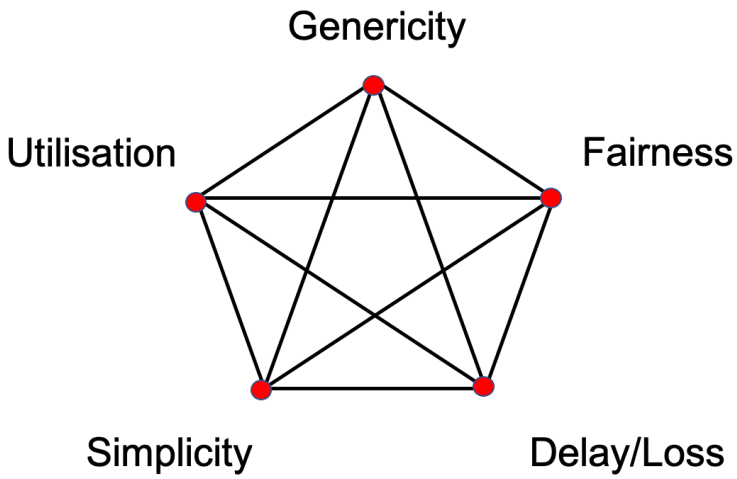



Fig. 1.1 Goals and tradeoffs for an ubiquitous CCA design

Overall, the design of the CCA in this thesis use the goal and tradeoff shows in Fig.2.1 as the principle. So based on these facts and assumption, we have these design trade-offs to guild our research.

The challenge for utilisation mainly comes from the real-time variation of RAN capacity.

The challenge for the delay and loss is mainly caused by buffer size management. Note that loss is the extreme expression of delay.

The simplicity is also our concern since the mobile device has limited processing power and limited battery life.

Fairness is the essential goal of the equilibrium of distributed CCA would like to achieve.

Last but not least, Genericity is the special requirement for a CCA running in a hybrid network like mobile network since the bottleneck can be anywhere such as RAN or the wired backhaul.

Simply speaking, the goal is to design a simple (to save computation power) CCA to improve the radio resource utilisation with the cost of minimum delay and loss in the whole mobile network. Step by step this thesis should achieve the goals on the tradeoffs in the following chapters.

## 1.2 Contributions

Based on the objectives, we decide to use the NS3 simulator to validate the existing CQIC algorithm and transplant the algorithm from 3G versions to 4G versions, to compare the performance of CQIC and traditional TCP CCAs. Furthermore, the most recent TCP BBR is implemented in NS3 simulator. Based on these advance work, the pros and cons of using CQIC and BBR are identified. Besides, based on the discovery of the earlier-mentioned work, an efficient cooperative CCA architecture is proposed. Though the proposed algorithm is not explicitly invoking information from the lower layer(MAC/PHY), but it achieves similar performance. More specifically, the performance of the proposed algorithm achieves better downstream (DS) delay (DSDL) compared to BBR and CQIC and has slightly worse goodput and DS throughput compared to CQIC. Furthermore, an improvement on the state machine is proposed with the configuration and the details analysed by proposed models. The improvement allows the state machine to achieve BW fair share in the fixed network and to have lower average RTT than BBR on the mobile network.

The contributions of this thesis are as follows:

- Identify the main feature of the conventional congestion control in both wired and mobile network. The features are implemented and validated by NS3 simulator.
- Identify the main feature of the existing CQIC algorithm and find a solution to release the congestion caused by original CQIC. An NS3 version of CQIC, named DCIC, is implemented.
- Introduce a state transition mechanism on server side for DCIC to complete the congestion control logic.

- ☐ Introduce the client slide bandwidth estimation concept into the congestion control design.
- ☐ for simplicity CDBE:Designed a prototype of Client-side Driven Bandwidth feedback loop is built and tested.
- ☐ Toward ubiquitous congestion control:1.Improve the state transition method in server-side to achieve BW fair share in the hybrid network. 2. Simplify the BW estimation for prototype.3. Introduced a parameter analysis method to achieve a low-delay/loss equilibrium for the startup of newcomer flows.

The future direction of this research can be the validation of TCP friendliness and further deploy CDBEv2 in a real network to further debug and enhance the capability of a faster fair BW share merge. The possibility of using RTT instead of DSDL remains to be validated since, in our assumption, the congestion only happens in the downstream, and the asymmetric Uplink and Downlink capacity is harmful to RTT based decision making. Furthermore, investigate the performance during the handover period, and high-speed mobility to the validation of the proposed CCA is also of our interest.

## 1.3 Thesis outline

The rest of the thesis is organised as follow: In Chapter 3, background on LTE mobile network and CCAs in end-to-end data transport are presented. A brief overview of the baseline CCAs, CQIC and BBR, is also described. The design principle for trade-off, necessary tools for analysis, network models and the general philosophy of CCAs are demonstrated in Chapter 4. The detailed comparison of between CQIC and baseline algorithms is shown in Chapter 5. The CQIC-s, which is CQIC, with state transition manner on server-side, and the BBR are also compared in this chapter. A novel CCA architecture is proposed in Chapter 6, and the performance is tested against the baselines. The improvement of this prototype architecture, CDBEv2, is proposed tested and validated in Chapter 7. The conclusion of this thesis is conducted in Chapter 8. Future works and the perspectives are also presented in this chapter.

# Chapter 2

# Introduction en Français

Cette thèse visait à aborder le problème du contrôle de la congestion dans les réseaux d'accès radio mobile avec une méthode cross-layer, et a finalement proposé une solution omniprésente pour un réseau mobile.

## 2.1 Contexte et objectif

Le réseau mobile se compose d'un réseau d'accès radio avec des cellules pour desservir les appareils des utilisateurs, d'un réseau central de paquets pour la distribution du trafic et de passerelles qui se connectent à d'autres parties de l'ensemble de l'Internet.

De la deuxième génération (2G) [1] de réseaux mobiles commerciaux à l'évolution à long terme largement déployée (LTE, également connue sous le nom de 4G) [2] de nos jours, la capacité d'un réseau d'accès mobile a connu une multiplication explosive grâce à les progrès des techniques radio avancées. La capacité maximale de la couche PHY [3] est passée de dizaines de kilo-bits par seconde (Kbps) à des centaines de méga-bits par seconde (Mbps) dans chaque intervalle de temps de transmission (TTI). De plus, grâce au schéma de modulation et de codage dynamique (MCS), Multiple-Input Multiple-Output (MIMO) [4]. Des technologies sans fil plus avancées sont appliquées à la cinquième génération (5G) du réseau mobile: antennes MIMO massives, bandes passantes à large spectre, agrégation de porteuses multi-bandes, etc. En plus de la couche d'accès multimédia (MAC) correspondante et de la couche de contrôle de liaison radio sur le réseau d'accès radio (RAN), le réseau mobile peut aujourd'hui fournir la bande passante théorique allant de Kbps à Gbps sur une courte période. Une telle agilité permet au LTE, à la 5G et aux générations suivantes de réseaux mobiles de prendre en charge une grande variété d'applications de réseau mobile dans la vie quotidienne. Outre la partie cellulaire, les infrastructures dorsales sont le type spécial de réseau fixe avec une conception en couches. Les données utilisateur circulent de la source externe (serveur distant ou serveur CDN) vers

les passerelles du réseau de distribution de paquets (PDN) (PGW), puis finalement envoyées à l'équipement utilisateur (UE) via le RAN. La sous-utilisation de RAN est un problème de longue date 3.1. Les couches supérieures qui prennent en charge le transport du trafic de bout en bout dans un réseau mobile devraient avoir une capacité d'adaptation pour mieux utiliser la capacité toujours croissante. Notre objectif est d'aborder le problème du point de vue de la conception du contrôle de la congestion dans la couche transport.

Dans cette thèse, nous allons:

1. comprendre la raison de cette sous-utilisation dans un réseau mobile cellulaire,

2. explorer les solutions existantes et tester leurs performances sur le réseau

3. concevoir des algorithmes, des protocoles ou des architectures de contrôle de la congestion pour améliorer les performances de bout en bout et analyser les avantages et les inconvénients de l'algorithme proposé

Il existe trois types de logique de conception dans un algorithme de contrôle de congestion sur la couche transport. il s'agit du contrôle de la congestion basé sur les pertes, du contrôle de la congestion basé sur le retard et du dernier contrôle de la congestion basé sur le délai de bande passante. En ce qui concerne l'utilisation de la bande passante, la méthode basée sur la perte échange l'utilisation avec la perte, en tant que forme extrême de retard dans un réseau tamponné. Les méthodes basées sur les retards négocient l'utilisation avec des délais différents. Le CCA basé sur le délai de bande passante, en tant que méthodes basées sur le délai de succés, équilibre l'utilisation et le retard de la bande passante par des opérations de sondage et de drainage fréquentes sur le domaine temporel.

Outre les CCA orientés couche de transport, il existe également plusieurs tentatives entre couches pour invoquer les informations de la couche inférieure afin d'améliorer l'utilisation de la bande passante. Dans notre examen, nous nous concentrons uniquement sur les solutions multicouches de bout en bout sur un réseau sans fil.

Dans le réseau mobile actuel, l'algorithme conventionnel basé sur la perte TCP / IP est toujours le flux principal. Un nouveau groupe de spécification de l'industrie Non-IP-Networking (ISG NIN) cite NIN de l'Institut européen des normes de télécommunications (ETSI) travaille également à la recherche d'un remplacement du TCP / IP conventionnel dans le réseau central 5G, car nous sommes fatigués du phénomène de bufferbloat causé par le CCA basé sur les pertes.

L'architecture de contrôle de congestion que nous allons concevoir est de bout en bout, qui prend en considération la nature d'un réseau cellulaire mobile. L'hypothèse est que tout le trafic entrant passera par le PGW, et les PGW peuvent garantir le CCA dans l'ensemble du

réseau fédérateur mobile. Par conséquent, nous pouvons concevoir un algorithme de contrôle de la congestion de bout en bout de la couche de transport pour garantir la juste part de capacité dans le backbone mobile et RAN.

Ces CCA conventionnels et récemment proposés sont à l'origine de ce sujet de thèse: Pour savoir si la conception multicouche comme CQIC est adaptée à l'ensemble du réseau mobile. Après la validation et l'évaluation de CQIC, d'une part, nous devons décider de suivre la manière explicite de CQIC ou d'introduire une estimation implicite de la bande passante. D'autre part, l'architecture de contrôle de congestion coopéré, qui n'a pas été bien introduite sur le serveur CQIC, devrait également être proposée, afin que le serveur ait la capacité de sonder davantage, de soulager la congestion et de partager également la bande passante de goulot d'étranglement.

Cette thèse vise donc à étudier et proposer un mécanisme ou une architecture innovants de contrôle de la congestion pour améliorer l'utilisation du réseau et l'expérience client sur les réseaux mobiles.

La méthode que nous allons suivre et les objectifs de la thèse sont:

- Passez en revue les algorithmes CCA existants et les fonctionnalités du réseau mobile et analysez leur fonctionnalité dans le réseau mobile
- Revoir les propositions multicouches pour les réseaux mobiles et effectuer une brève évaluation de ces mécanismes,
- Mettre en œuvre et examiner le dernier BBR dans le simulateur NS3  cite ns3 comme base de comparaison
- Proposer et concevoir une architecture CCA innovante pour les réseaux 3G et 4G,
- Dériver un modèle et évaluer les performances de ces architectures et développer les fonctions requises, et
- Mise en œuvre et évaluation du CCA proposé

Dans l'ensemble, la conception du CCA dans cette thèse utilise comme principe les objectifs et les compromis illustrés dans la figure  ref fig: goalNtradeoff. Donc, sur la base de ces faits et hypothèses, nous avons ces compromis de conception pour guider nos recherches.

Le défi de l'utilisation vient principalement de la variation en temps réel de la capacité RAN.

Le défi pour le retard et la perte est principalement causé par la gestion de la taille de la mémoire tampon. Notez que la perte est l'expression extrême du retard.

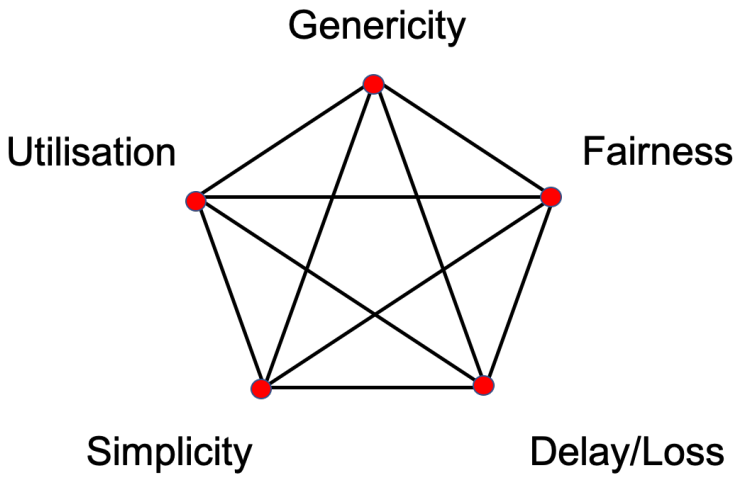



Fig. 2.1 Objectifs et compromis pour une conception CCA omniprésente

La simplicité est également notre préoccupation car l'appareil mobile a une puissance de traitement limitée et une autonomie de batterie limitée.

L'équité est l'objectif essentiel de l'équilibre de l'ACC distribuée que l'on souhaite atteindre.

Enfin, la généricité est l'exigence particulière pour un CCA fonctionnant dans un réseau hybride comme un réseau mobile puisque le goulot d'étranglement peut être n'importe où, tel que RAN ou le backhaul filaire.

En termes simples, l'objectif est de concevoir un CCA simple (pour économiser la puissance de calcul) pour améliorer l'utilisation des ressources radio avec un coût de retard et de perte minimum dans l'ensemble du réseau mobile. Étape par étape, cette thèse devrait atteindre les objectifs sur les compromis dans les chapitres suivants.

## 2.2 Contributions

Sur la base des objectifs, nous décidons d'utiliser le simulateur NS3 pour valider l'algorithme CQIC existant et transplanter l'algorithme des versions 3G vers les versions 4G, afin de comparer les performances des CCA CQIC et TCP traditionnels. De plus, le TCP BBR le plus récent est implémenté dans le simulateur NS3. Sur la base de ces travaux avancés, les avantages et les inconvénients de l'utilisation du CQIC et du BBR sont identifiés. En outre, sur la base de la découverte des travaux mentionnés précédemment, une architecture CCA coopérative efficace est proposée. Bien que l'algorithme proposé n'invoque pas explicitement les informations de la couche inférieure (MAC / PHY), il atteint des performances similaires. Plus précisément, les performances de l'algorithme proposé permettent d'obtenir un meilleur

retard en aval (DS) (DSDL) par rapport au BBR et au CQIC et ont un rendement et un débit DS légèrement inférieurs par rapport au CQIC. De plus, une amélioration de la machine d'état est proposée avec la configuration et les détails analysés par les modèles proposés. L'amélioration permet à la machine d'état d'atteindre une part équitable du BW dans le réseau fixe et d'avoir un RTT moyen inférieur à celui du BBR sur le réseau mobile.

Les contributions de cette thèse sont les suivantes:

- ☐ Identifiez la caractéristique principale du contrôle de congestion conventionnel dans les réseaux filaires et mobiles. Les fonctionnalités sont implémentées et validées par le simulateur NS3.

- ☐ Identifiez la caractéristique principale de l'algorithme CQIC existant et trouvez une solution pour libérer la congestion causée par le CQIC d'origine. Une version NS3 de CQIC, nommée DCIC, est implémentée.

- ☐ Introduce a state transition mechanism on server side for DCIC to complete the congestion control logic.

- ☐ Introduisez le concept d'estimation de la bande passante de la diapositive client dans la conception du contrôle de congestion. Ce changement

- ☐ pour plus de simplicité CDBE: Conçu un prototype de boucle de rétroaction de bande passante pilotée côté client est construit et testé.

- ☐ Vers un contrôle de congestion omniprésent: 1. Améliorez la méthode de transition d'état côté serveur pour obtenir une part équitable de BW dans le réseau hybride. 2. Simplifiez l'estimation de BW pour prototype.3. Introduction d'une méthode d'analyse des paramètres pour atteindre un équilibre à faible retard / perte pour le démarrage des flux de nouveaux arrivants.

L'orientation future de cette recherche peut être la validation de la convivialité TCP et le déploiement ultérieur de CDBEv2 dans un réseau réel pour déboguer davantage et améliorer la capacité d'une fusion de partages BW plus rapide et équitable. La possibilité d'utiliser RTT au lieu de DSDL reste à valider car, dans notre hypothèse, la congestion ne se produit qu'en aval, et la capacité asymétrique de liaison montante et descendante nuit à la prise de décision basée sur le RTT. En outre, enquêter sur les performances pendant la période de transfert, et la mobilité à grande vitesse pour la validation du CCA proposé est également de notre intérêt.

## 2.3 Aperçu de la thèse

Le reste de la thèse est organisé comme suit: Dans le chapitre 3, des informations générales sur le réseau mobile LTE et les CCA dans le transport de données de bout en bout sont présentées. Un bref aperçu des CCA, CQIC et BBR de base est également décrit. Le principe de conception pour le compromis, les outils nécessaires pour l'analyse, les modèles de réseau et la philosophie générale des CCA sont présentés au chapitre 4. La comparaison détaillée entre les algorithmes CQIC et de base est présentée dans le chapitre 5. Le CQIC-s, qui est CQIC, avec une manière de transition d'état côté serveur, et le BBR sont également comparés dans ce chapitre. Une nouvelle architecture CCA est proposée dans le chapitre 6, et les performances sont testées par rapport aux lignes de base. L'amélioration de cette architecture prototype, CDBEv2, est proposée testée et validée au chapitre 7. La conclusion de cette thèse est conduite au chapitre 8. Les travaux futurs et les perspectives sont également présentés dans ce chapitre.

# Chapter 3

# Overview of Mobile network architecture and congestion control algorithms

## 3.1 Introduction

In most of the time, when the public is talking about a mobile network, it is highly likely that they are talking about a mobile cellular network. Generally speaking, a cellular, or a cell, is the last hop connecting the mobile user devices. These user devices include Internet of Thing (IoT)devices like smart meters, Mobile phones or laptops with low or zero mobility. RAN can also serve the high mobility scenarios like trackside communication for onboard Wi-Fi, or vehicular use cases. However, the mobile network is far beyond the cellular part. Various types of cells, including Pico Micro Macrocells, these cells are part of the Radio Access Network, aka RAN, and a RAN must cooperate with mobile Backhaul network and the gateway to forming a complete functioning mobile network. Mobile networks managed by different operators are then further connected to other types of networks to form the internet we see today.

In this chapter, firstly, we review the Architecture of User Plane in the LTE network will be introduced. Understand this layered architecture is helpful for us to implement CQIC from 3G to 4G network. Secondly, different types of congestion control algorithms are categorised and reviewed. This information helps us to extract the basic logic of congestion control and the basic trade-off a traffic equilibrium a buffered is facing. Based on the knowledge of this chapter, we can further discuss the tools and the analysis we are going to discuss in Chapter 4

## 3.2 LTE mobile network

The architecture of the LTE core network is the ground true network we are making our abstract and assumptions on. Its architecture is shown in Fig.3.1. The architecture of 5G RAN and Non standalone 5G has the exact same backbone as LTE. As we can see that the network is composed of Evolved Universal Terrestrial Radio Access Network (EUTRAN) and the EPC.

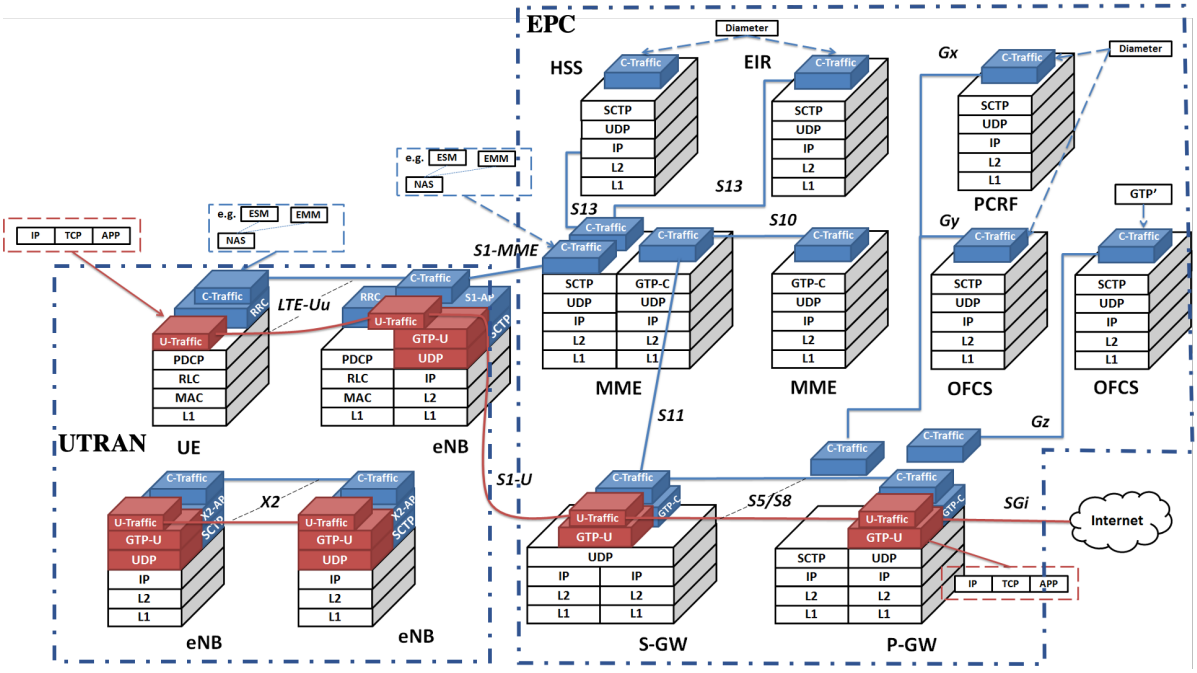



Fig. 3.1 LTE interface stacks

The stack used among the whole LTE network is shown in Fig. 3.1. It is a heterogeneous network with a wired backhaul network with user and control plane and a Radio access network. LTE treats the Upper layer TCP/IP packets as the 'Application' layer data. Firstly we simply revisit the data bearer structure in mobile backbone network.

### 3.2.1 LTE backhaul

The connection between a mobile network and the outside internet is Service and Packet delivery network gateway (SPGW). It is the entrance of mobile network and manages the traffic in the mobile backhaul(wired backbone network). It encapsulates the TCP/IP packet into the 'Bearer'.

The BLUE protocol stack in Fig. 3.1 is used by Control plane, where the signalling of mobile network operators flows through including authentication, billing, policy control, etc.. The RED protocol stack is specifically used by the user plane. The data from a remote server

will be delivered to eNB by this family of the protocol. The communication between eNB and the EPC entities, in both control plane and user plane, are mostly delivered in a wired manner. For a different type of packet, the bearer uses a different combination of the protocols to accomplish the packet delivery. For the user plane, the bearer uses GTP-U, which generally running over UDP, to carry the traffic from UEs. Since GTP protocol has no retransmission or feedback mechanism, the end to end data transfer from UE to the SPGW is unreliable. The design of the backhaul fits the original purpose of the TCP well since congestion can also exist in the LTE core and backhaul network. It seems reasonable that the congestion control mechanism in TCP server can also manage the congestion in the LTE backhaul. However, there may be up to tens of hops of routing/switching between EUTRAN and EPC entities. The delay in the backhaul network may also be considerable. The upper bound of the non-really time U-plane data is up to 300ms [7]. The signalling in the LTE system, on the other hand, uses SCTP protocol which is a reliable transport layer protocol. To guarantee the QoS in an LTE core/backhaul network, QoS Class Identifier(QCI) has tagged to each, bearer. Generally, the control signal exchange has higher priority, and when the congestion arrives at the specific nodes in the LTE network, the buffered data with lower priority will be discarded first. However, in the existing network, QCI and the dedicated option is generally disabled. Basically, all the traffic, except VoLTE service, are running on the default bearer. Since the control traffic is negligible in the LTE xhaul(backhaul/fronthaul), the data packets are delivered in a similar best effort manner on the internet.

Such design of using TCP/IP to bear traffic can be dangerous since the loss-based congestion control will cause the bufferbloat mentioned above effect. The ETSI is considering changing such architecture[5].

In this research, the simplified model, as in Fig.3.4 is assumed. In the following sections and chapters, our discussion on the mobile network is based on this illustration. Our assumption is the CCA we designed is working in the PGW where all the traffic from outside internet (remote server or local CDN) is treated as the saturated application layer input. Hence with the filtering of SPGWs, a network with only pure novel CCA in the mobile backbone is formed where the network operator has full control of the performance of traffic.

### 3.2.2 LTE Radio Access Network

A mobile Radio access network is a network utilise the wireless channel with centralised scheduling manner. In such system, radio link spectrum resource, spatial reuse feature, code (for code division multiple access) and time slots (for time division multiple access) [8] are the managed by the infrastructures running mobile network standards.

The reason why mobile Radio Access Network (RAN) must be so complex is: radio link without appropriate PHY technique and MAC design is not as reliable as a wired channel. It can be lossy, and the BER is high[9, 10]. The appropriate Radio link technique is critical for the wireless channel to have robust performance. With developing an understanding of radio PHY characteristics, two mainstream wireless technique exists:

1. Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA)[11] based distributed resource random access. The typical PHY/MAC branch is the 802.11 family Wi-Fi technique [12]. The traffic management in such type of network can be fuzzy[13–15]

2. Mobile/Cellular network with a centralised base station scheduling mechanism[16]. The traffic management has more dynamics compared to distributed traffic competition manner in Wi-Fi [17–19].

Though Wi-Fi and mobile cellular network have a different manner of sharing or scheduling the resource in a time-varying wireless channel, they do have some common fundamental elements which affect congestion states in the network:

- More and more advanced Radio Link technique to improve the robustness and wireless resource utilisation

- Larger buffer size compared to the decades ago, thanks to the lower cost of hardware

The mobile cellular network(mobile network, in short) is our research focus. To have a well-performing system, not only the RAN and simply larger buffer are taken into consideration but also the various hardware and software are carefully combined together. We are going to introduce the fundamentals of mobile network system from the perspective of the Radio Link technique and buffer in the following subsection.

**Radio Link Capacity**

The development of mobile network on radio link use the following line of thought to makes a tradeoff between the lossy nature and the

1. Increase the unit time resource utilisation with more and more advanced PHY layer with more and more advanced Modulation and coding schemes (MCS) and massive Multiple-Input Multiple-Output(MIMO) technique

2. Reduce the interference by using beamforming thanks to the multi-antenna[20] and by using the dynamic MCS of central scheduling manner of the base station to select the most efficient MCS which guarantees robustness.

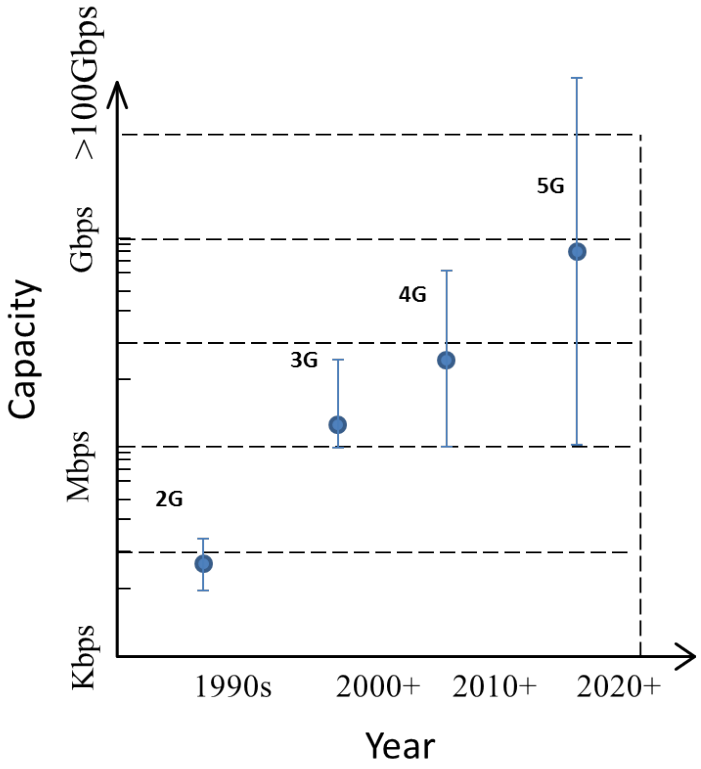



Fig. 3.2 Capacity variation in different generations of Cellular network

3. Reduce the loss by trading the detected packet loss with processing time or retransmission by applying error correction coding and ARQ/HARQ mechanism.

All these features are made possible thanks to the advancing processing and storage power, research inputs and engineering efforts. These features are implemented in different layers of RAN. It will be introduced latter.

As mentioned in Chapter 1, in the GSM and EDGE[1] of 2G network architecture, the PHY capacity varies in tens of KBps level. RTT from the Remote Server(RS) to the User Equipment(UE) is usually above the one second and some times even more than 10-second [21, 22]. With the development of 3G/4G mobile network, the PHY capacity grows significantly(up to tens of MBps in 3G and hundreds of MBps in 4G), while the per-user buffer size is extended to MB level(1+MB/4+MB in HSPA/HSPA+, 5MB+ in LTE[23]). The variation of the radio link capacity is shown in Fig3.2.

The other typical difference among 2G, 3G, 4G [22–29] networks is listed in the table below: Note that the capacity in Table 3.1 is the PHY capacity.

| | RTT | Transition Time interval | Buffer(Per user) | Typical Capacity(DL) |
|---|---|---|---|---|
| 2G[1, 30] | >1s | >138ms | ≈1MB | 9.05-21.4Kbps(GPRS), 9.2-59.6Kbps(EDGE) |
| 3G[28, 31, 32] | ≈100-200ms | 10-80ms(UMTS), 2ms(HSDPA) | ≈4MB | 0.9-14.4Mbps (WCDMA) |
| 4G[32–34] | ≈70ms | 1ms | ≈5MB | 0.9-345.6Mbps(LTE) |

Table 3.1 Difference among 2G 3G and 4G mobile radio link(typical values)

The layered throughput, which can be measured from Media Access Control(MAC), Radio Link Control(RLC), all the way through Packet Data Convergence Protocol (PDCP), presented to the transport layer is much lower than this peak PHY value due to the presence of lossy wireless nature, the resulting HARQ and other error correction technique. This is to be discussed later in this chapter. To catch the trend of the advanced mobile design, and the buffer feature of backbone network we review the full LTE mobile network architecture in the following sections.

**Buffer size and bufferbloat**

Due to the variation of Radio link capacity, a node in the network is treated as a bottleneck once its capacity cannot digest the incoming traffic. Bufferbloat is a phenomenon caused by the network device with a deep buffer while the BW is not high enough. Namely the length of queue inflated to a level where the packets in it is experiencing a high delay before loss.

Naturally, an appropriate amount of buffer is necessary to reduce the loss in the network. This makes a network node to become a queuing system. In such a system, pursue the queue length can be a harmful strategy in the perspective of the round trip time for time-sensitive application or a Data-Acknowledgement loop. According to the type of network and the traffic expectation, Queuing theory[35] can suggest an optimal queue length. However, the total buffer size is relatively large[36] and a typical buffer depth in LTE network is between 2MB to 6MB[26]. The mainstream AIMD loss-based congestion control is greedy enough to always inflate the queue in the network. Until a loss timeout timer expired, a loss-based congestion control mechanism will keep on increasing its CWND in Additive increase or even more aggressive manner.

The interesting fact is that in the cross-layer tracing study[30], the error losses are not detected due to the reliable link layer protocol exists in GPRS radio link protocol stack. Though typical buffer size is small(measured as up to 50KB/30KB in DL/UL per user[30]), the low capacity tends to drain the buffer slowly. Hence the bufferbloat problem exists in the GPRS network as a result of buffering nature of the mobile network[30]. These data show that the bufferbloat will exist as long as the buffer size and the capacity of radio link do not match.

Last but not least, with the increased peak radio capacity and buffer size, some rethinks should also be made on the judgement of radio link is the bottleneck in the network. For example, the router in the mobile backhaul/Internet/EPC may introduce higher delay than the radio access network if it locates in a busy crossway. Hence the design of CCA in a mobile network for edge data transmission should take the radio link and wired bottleneck into consideration. In the following section, we will look into the full architecture of LTE mobile network.

**Radio Access Network Protocol stack**

In this thesis, we firstly focus on the radio access network(RAN) bottleneck. Hence the starting point is to utilise the information invoked from PHY, MAC or RLC layer. The layered technical detail is reviewed to find out the information we need to implement such information.

The wireless link was described as lossy and poisoned the TCP connection quality in the way of high Packet Error rate[10]. Decades latter, with the development of the Coding techniques, the dynamic scheduling schemes, e.g. dynamic MCS, and cross-layer ARQ the link failure rate exposed to the upper layer is lower than $10^{-6}$[7] in LTE network.

An abstract of the architecture of the user plane of LTE network is shown in the Fig.3.3.

On the radio link side, the LTE RAN consists of PDCP, RLC, MAC and wireless PHY layers. PDCP, RLC and MAC layers have their duty on QoS:

- PDCP layer compresses the IP data and manages their order with PDCP sequence number(SN). It ensures the intra-LTE handover is seamless, and the packets are delivered in

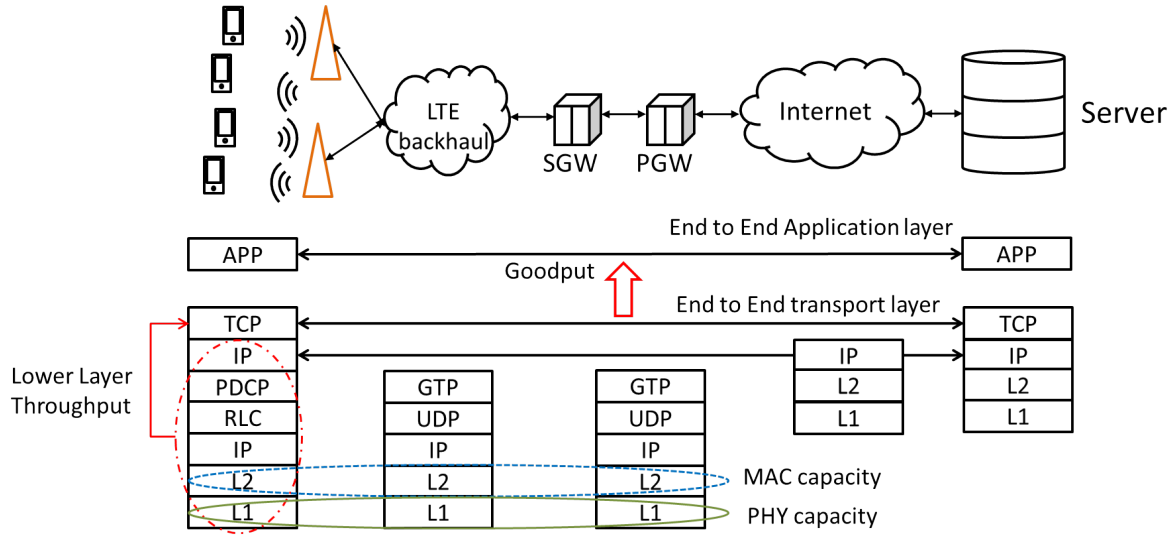



Fig. 3.3 Brief LTE protocol stack

order up to the higher layer. When the handover of a user between cells happens, the data in the specific UE buffer is transmitted from one eNB to another. A temporary increase of RTT may be experienced at handover event. However, the reliable delivery at handover and PDCP status report may not always implement. As a result, the retransmission and the lossless handover may not always be the case in the LTE network. LTE private buffer space for each mobile user equipment (UEs). Hence to keep the number minimum number of the in-flight packet can help avoid catastrophic loss during handover and further achieve better link utilisation.

- RLC layer consists of three modes: Transparent Mode (TM), Unacknowledged Mode (UM), and Acknowledged Mode (AM). AM mode provides most functions including resegmentation, reordering, duplication discard, etc., while TM mode buffers the upper layer data a bit. Since there can be an error in MAC layer, the erroneous data packet can lead to MAC layer retransmission and further cause disorder delivery for upper layer while the incorrect ACK/NACK can cause retransmission from RLC layer.

- MAC layer will request data from the RLC layer buffer when the resource allocation scheduling negotiation is finished. The eNB manages both uplink and downlink scheduling. UE will report the Channel status to eNB in a control channel, and the scheduling algorithm in eNB MAC layer will decide according to several constraints including the reported channel quality, the number of users, and the size of data. The scheduler further determines the scheduling for the next time slot, and distribute the results back to the UE by Downlink/Uplink Control Information (DCI/UCI).

In this work, the characteristic of LTE-RAN above are studied in the purpose of transplanting, for the first stage, the 3G-QUIC based CQIC algorithm into LTE UE. Further, the study is used to design the first prototype of CDBE and CDBEv2 in Chapter 6 and 7.

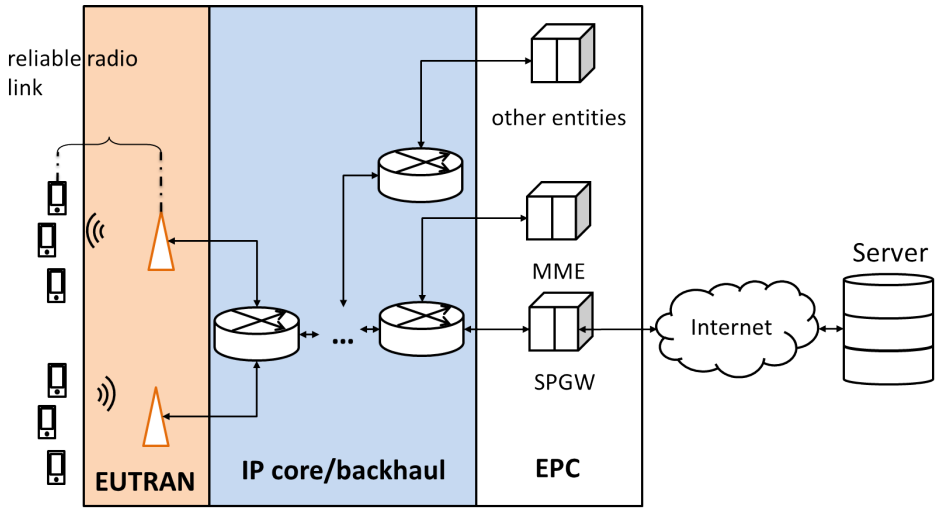



Fig. 3.4 Simplified LTE Network architecture with RAN, backhaul and gateway

## 3.3 End-to-End Congestion control methods for Quality of Service

Work in [37] classified TCP improvement on the last-hop wireless network into four categories:

1. split connection
2. end-to-end connection
3. explicit connection
4. localization of wireless loss

With the development of the mobile network technique, some function of subcategories mentioned in the article is entirely fulfilled by the existing mobile network infrastructures, e.g. ARQ/HARQ in eNB These techniques address the wireless loss at the cost of time domain efficiency or processing powers. Hence the delay in radio uplink or downlink can vary in a wide range. In our research a new category scheme is applied to the development of the mobile infrastructure.

Based on the different network position where the solutions should be applied, existing designs for the QoS improvement on the mobile network can be classified into two categories: end-to-end(e2e) and the middlebox. The definition of the middlebox here is generalised. It can stand for any devices in the network between the two endpoints of the service including eNB, S/PGW and multifunction router. The location of middleboxes may affect the tradeoff between the cost of deployment and effectiveness.

Most TCP solutions can further be subcategorised by with or without the support of cross-layer features. The other cross-layer protocol needs to implement both server and a massive amount of clients, even if the improvement is remarkable, their usage may be limited. Again the deployment problem is through severe but out of the discussion of this report. Other solutions may include the suggestion of upgrade the RLC status[38] , A brief comparison among different categories will be concluded, and we are going through the detail of them in the following subsections.

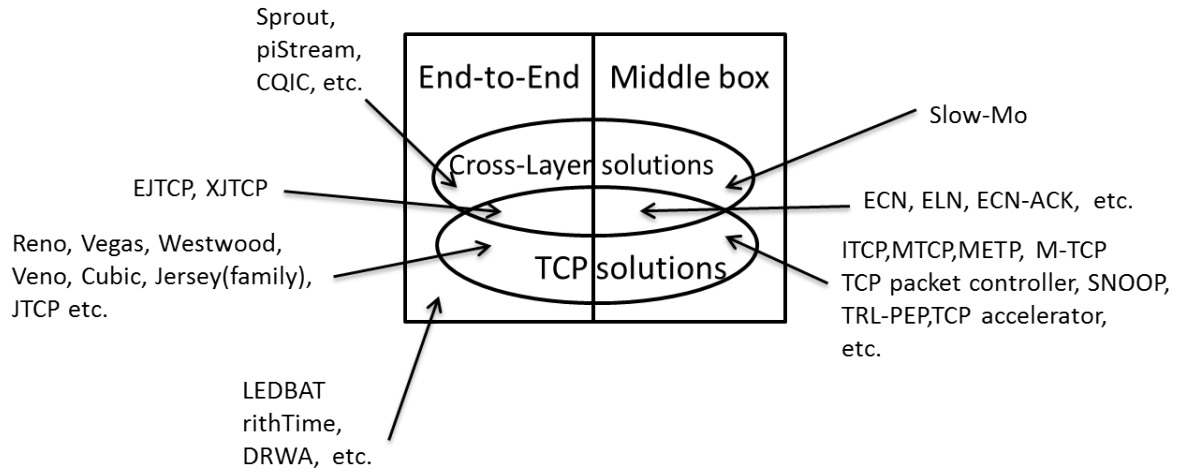



Fig. 3.5 Categories of congestion control solutions

### 3.3.1 Non-Cross-Layer protocols

#### End-to-End

The main end-to-end congestion control algorithms review and analyzed in this thesis are listed below:

1. loss-based CCA. There are plenty of variations in this category of CCA. Loss-based CCA can take over the full depth of the buffer in the network. In a network with shallow buffer, such design can get frequent loss and low bandwidth utilisation. In a network

with a deep buffer, such architecture can result in a queue heap up of BIF and the when an RTO happens a large number of packets will be retransmitted. Hence the bandwidth utilisation is also low from the perspective of goodput.

2. There is another category of CCA, named delay-based CCA. This kind of CCA uses the delay as an indicator of congestion and further alter the CWND to control the traffic. The low-delay and fairly decent goodput can be achieved when the RTT status is relatively stable. Note that the delay-based CCA can lose its share of buffer and pipeline occupation.

3. The latest TCP version combines the bottleneck bandwidth and round trip delay (BBR)[39] proposed by Google. This algorithm possesses a new state machine to utilise the bandwidth estimation(BE) and RTT. This CCA is widely deployed in Google's network. Its behaviours in the mobile network have not been tested yet. Hence it is interesting to try the algorithm in this thesis.

4. In one of the Google proposal, called "CQIC"[40], the customer terminal can try to predict the radio allocation and transmit this information to the TCP source. TCP source rate adaptation is then possible, similarly to the previous proposal.

Thought not pointed out explicitly, the most important features of a congestion control algorithms are:

1. probe for more bandwidth and,

2. balance share of bottleneck.

To achieve the goal mentioned above, a congestion control algorithm must not only rely on the lower layer capability report but implement an engine function on the transport layer as a guarantee of end-to-end service. Major Non-Cross-Layer E2E solutions fall in the TCP domain, and the way to achieve the goal is to use a feedback signalling control loop[41]. The abstract and the modelled analysis is shown in the next Chapter 4. Now we discuss the typical engineering features of the feedback loop.

The two mainstream transport layer protocols are User Datagram Protocol (UDP)[42] and Transport Control Protocol (TCP)[43] and the latter is used in mobile network transport layer for congestion control. Different from the unreliable transmission provided by UDP, the connection-oriented TCP uses feedback packets named Acknowledgements (ACK) to provide reliable end-to-end data delivery. Moreover, these ACKs are opportunistically used to detect network congestion, through congestion control algorithms (CCA); These algorithms offer different throughput-RTT tradeoff and are to be used in different scenarios, including but not limited to the type of application and the feature of the network.

For reliable transmission, a positive acknowledgement (ACK) is employed. The receiver of the data packet will send ACK if the packet is received successfully. Furthermore, the buffer state of the receiver will also be embedded in the ACK as Flow Control (FC) indicator. With the help of this feedback signalling and the information within, the server can regulate the transmission rate using Sliding Window (*SW*) mechanism with the FC information, named receiver window (RWND) size on the server. The other constraint of the SW mechanism is the CCAs.

Generally, a transport layer manages the end-to-end traffic in a IP network nowadays[44]. A transport layer also exists in a mobile PDN for both control plane and user plane since the data exchange in a mobile network also use the layered TCP/IP packet delivery architecture.

The congestion window (CWND) is their cornerstone, as it controls the unacknowledged data volume in transit (Bytes In Fly, BIF) between the TCP source and destination.

In the 4G network, the typical Round Trip Time(RTT) is much lower than that in 3G[32], and it can be expected that delay in a mobile network shall further be reduced to 10ms level[34]. However, the wireless resource scheduling, mobility and buffering instability in the network make the RTT tricky evidence of packet loss. Hence the accurate RTT based TCP vegas[45] will not an appropriate solution for RTT varying wireless network since it is per ACK based CWND modification manner and the reaction to BW variation can be too slow.

The conventional Congestion control algorithm(CCA) in consists of Slow Start(SS), Congestion Avoidance(CA), Fast Retransmission(FReTx), Fast Recovery(FRecv) and Timeout Retransmission. SS increases the transmission window(a.k.a. congestion window, CWND) size exponentially per ACK before the CWND reaches Slow Start Threshold(*ssthresh*). Once the CWND is equivalent or larger than *ssthresh*, TCP server stops SS and is engaged to Congestion Avoidance. In CA, once the packet loss is confirmed (generally Retransmission Timeout, RTO), TCP server will set *ssthresh* as half the current CWND size and retreat to SS. If the third duplicated ACK reaches before RTO. However, the FRTx starts before the RTO. This is the features of most conservative old version TCP Tahoe implemented as suggested by [46]. TCP Reno [47][48] triggers FRecv, instead of FRTx, which does not force TCP Server to go back to SS by just half the CWND. The legacy TCP considers network congestion results in all the losses.

By replacing the conventional CCA growth logic in CA state with the CUBIC function, the TCP CUBIC[49] is one of the most widely deployed TCP variants since it is by default built-in Linux kernels from version 2.6.19.

All these kind of the aforementioned Loss-Based CCAs have a slow start state. A slow start is to discover the current bandwidth by filling the buffers in the network. Once the congestion loss point is discovered, the *ssthresh* is the operating baseline defined in equation as twice

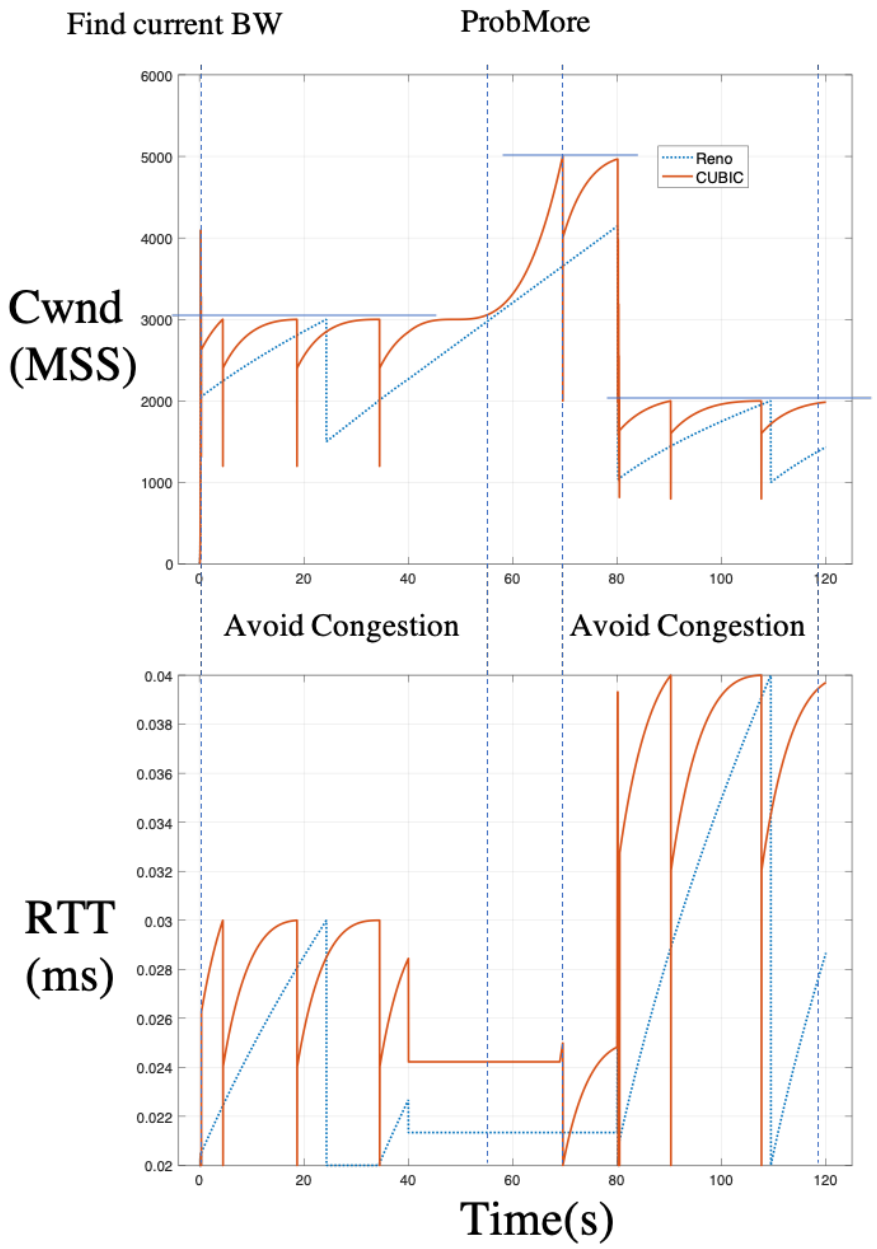



Fig. 3.6 Illustrate the basic concept behind loss-based congestion control

the Bandwidth Delay Product(BDP). To avoid frequent loss, the loss-based method 'carefully' operate in the buffer-filling operation area, which is between the full pipeline and the full buffer queue lengths. Once the bottleneck bandwidth($BW_{Btlnck}$) is increased for any reason, the Reno style congestion avoidance manner may take a very long time to catch the newly available bandwidth. Hence BIC[50] introduces the faster recovery and probe. The basic idea is to be more cautiously near the recorded losing point and be more aggressive when far away from

the losing point whether it is in actual probe stage or congestion avoidance. To avoid the RTT sensitivity caused by the ACK-driven binary search method, TCP Cubic [49] is introduced to use the following cubic function to smooth the procedure:

$$\begin{aligned} W(t) &= C(t-K)^3 + W_{max} \\ K &= \sqrt[3]{\frac{W_{max}\beta}{C}} \end{aligned} \tag{3.1}$$

where provides the configurable parameters $C$ and $\beta$. $C$ controls the speed of converge, and $\beta$ controls the friendliness to TCP.

To compare the performance of CUBIC and Reno, we simulate a fixed network in NS3 simulator, where Maximum BIF of the network varies from 3000pkts to 5000pkts at t=50s and decreases to 2000pkts at t=80s. The result is simply shown in Fig.3.6. It shows that CUBIC has faster converge when a loss happens and has more aggressive bandwidth probe manner. Concerning the Throughput, CUBIC has a better performance. However, higher throughput is not necessarily resulting in higher Goodput. The RTT performance of CUBIC is also worse than that of Reno.

To be more polite and mitigate the potential extra delay, the delay-based CCA like Tcp vegas[45] is used to maintain the buffer on a lower level:

$$\begin{aligned} expected^{t+1} &= \frac{CWND^t}{RTT_{min}} \\ actual^{t+1} &= \frac{CWND^t}{RTT_{measured}} \\ Diff &= expected^{t+1} - actual^{t+1} \\ CWND_{diff} &= Diff * RTT_{min} \end{aligned} \tag{3.2}$$

By calculating the difference between expected capacity and actual capacity of the round, if the upper-bound($\alpha$) or a lower-bound($\beta$) is exceeded, the algorithm will be increased or decreased respectively:

**Algorithm 3.1:** Vegas congestion avoidance logic

```
if CWND_diff <= α then
    // There is still room, send more
    cwnd = cwnd + 1;
if CWND_diff > β then
    // congestion happens, take a backoff
    cwnd = cwnd − 1;
```

Delay-based congestion control algorithms can maintain the queue depth to a relatively low level. However, the space for this kind of algorithm will be compressed by the loss based algorithm since the nature delay-based CCA is to reduce the CWND when RTT increase is made by the loss-based congestion control algorithms[51, 52]. Similarly, the rerouting[53] can also cause a similar effect.

TCP-Jersey[54], TCP New Jersey[55] and TCP NJ+[56], estimates the available bandwidth using more accurate algorithms according to the ACK arriving time. The *ssthresh* is also adapted according to the estimation. The Congestion warning mechanism using the ECN bits is also implemented in the Jersey family of TCP and conveys the simple image of the bottlenecked queue to the sender. The JTCP[57] and uses the jitter ratio to predict the reason for packet loss and adapt to the congestion control strategy. They employed the one-way delay jitter in RTP[58] and Jitter Ratio in [59] to compare with the threshold to distinguish the network congestion or wireless link loss. However, these widely used/experimented variances of TCP cannot fully utilise the available wireless resources nowadays.

Based on the hypothesis that the wireless link is lossy, and the BER is high[9, 10], TCP Westwood is proposed[60]. It monitors the returning ACKs and estimates the available bandwidth accordingly.

TCP Veno[61], similar to Westwood, estimate the state of the connection while AIMD scheme of Reno is also applied. The SS triggered by RTO will have adaptive *ssthresh* according to the congestion status. It keeps on measuring the minimum RTT and update the bandwidth estimation:

$$BWE = P_{acked}/RTT \tag{3.3}$$

Once a triple duplicated ACK event or a retransmission timeout happens, the *ssthresh* and the CWND are reset following the Algorithm 3.2. Such a manner not only reveal the expected operation starting point but also allows the sender to stick to it. The tradeoff is to

**Algorithm 3.2:** DupAck in Westwood

```
1 ssthresh = (BWE * RTT_min) / PktSize ;
2 if cwnd > ssthresh then
3     cwnd = ssthresh
```

However, as described earlier, the lossy nature of wireless link in the mobile network has been managed by the HARQ/ARQ mechanisms in mobile MAC and RLC layers [2].

The utilisation of the available bandwidth is less than 50% for TCP connections[26] and Westwood, Veno, and Cubic does not significantly outperform Reno[23]. This is predictable since these variance estimates the Congestion and Bandwidth on per ACK level. Such estimation

**Algorithm 3.3:** RTO in Westwood

```
1 ssthres = BWE*RTT_min / PktSize ;
2 cwnd = 1 ;
3 if ssthresh < 2 then
4     ssthresh = 2
```

will always be inaccurate due to the fast-changing nature of the Mobile network as adaptive MCS and resource allocation is applied, even though the UE is static.

LEDBAT [62] is one of the Non-TCP and non-cross-layer e2e solutions. It has faster congestion reaction than TCP thanks to the one-way delay estimation. It also uses a less aggressive manner to utilise the background resource to improve the overall bandwidth utilisation.

These are the popular conventional CCAs which are widely tested or deployed in the existing networks. Not only tested in the fixed network but also in a mobile network. The design logic of loss-based CCA treats the buffer as an extra resource which should also be shared by the co-existing flows.

However, from my point of view, the buffer is a redundancy and flexibility for the flows to temporally invoke while allocating the bandwidth, rather than a must-use resource to fill with greedy manner.

### Middle box design

There are several important studies on split-connection protocol designs. The more accurate name should be TCP proxy technique. The middlebox receives the packets from the TCP sender and reacts according to their consciousness of the UE status. This procedure may introduce buffering, processing of the data packet and forward of the ACK packet. The end-to-end TCP connection manner may not be maintained. Indirect TCP(I-TCP) in[63, 64] is the initiator of split TCP concept with complete design and implementation. There was no 3G/4G network back to 1995, and the designer employs Mobility Support Router (MSR) to separate the connection between the TCP servers and clients as Fig.3.7 shows. The original end to end semantic is split into a regular TCP connection between MSR and RS and one modified wireless TCP connection between UE and the MSR. The protocol takes care mobile handover by re-establishing socket for the same pair of endpoints on newly connected MSR and transfer the corresponding buffered data in old MSR to the new MSR. The handover latency is high(1430 micro seconds for buffers depth of 32Kbtes) since the technique is limited to the computer constraint back then. Similar designs are MTCP[65] and METP [66] with different wireless link Optimisation as in Fig.3.8. MTCP optimise the wireless last hop on socket level

| | Improvement in CCA | problem to address |
|---|---|---|
| Westwood | Reset the *ssthresh* and congestion window size according to the BW estimation. BW is estimated by monitoring the returning rate of ACKs. | varying bandwidth in heterogeneous network |
| Veno | Judge the reason for Packet Loss by RTT. React If the RTO is caused by congestion use legacy TCP. Otherwise, the Congestion window is reduced in an aggressive manner. | High loss rate in wireless link |
| Cubic | Replace the traditional TCP congestion control SS with the CUBIC function to harvest the high capacity faster. | utilise the high-speed link with high latency |
| Jersey(family) | Similar to Westwood, but the estimator algorithm is different | varying bandwidth in the heterogeneous network. Extend the ECN support in TCP |
| JTCP | Use jitter ratio to distinguish the wireless packet loss from congestion packet loss | High loss rate in wireless link |
| LEBAT(Non-TCP) | Estimate the queuing status using the difference of one-way delay. | fully utilise the bandwidth as background traffic |

Table 3.2 Summary of non cross-layer E2E solutions

over TCP while METP, on the other hand, uses Link-layer ACK on the previous wireless hop to recover the losses.

Different from ITCP, METP and MTCP, M-TCP[67] keeps the underlying end-to-end semantic. The end-user ACK rather than an ACK from the relay is sent back to the un-modified TCP server. This allows the UE, RS and M-TCP supervisory host (SH) to synchronise but operational flexibility remains on SH. The handover latency is also taken care of by the SH. Since M-TCP uses one SH to monitor several cells, it proxies the connection between the RS and the UE, but it locates one or more hops away from the base station. Hence when handover happens, the port and address mapping of wireless connection will not frequently be changed. TCP packet controller[68] is implemented on the BS, it inspects all the TCP flows to manipulate the DATA and ACK packet. The DUPACK and Packet TRL-PEP[69] is a more recent work combining the existing 3G technique with flow control and loss recovery. It implements the

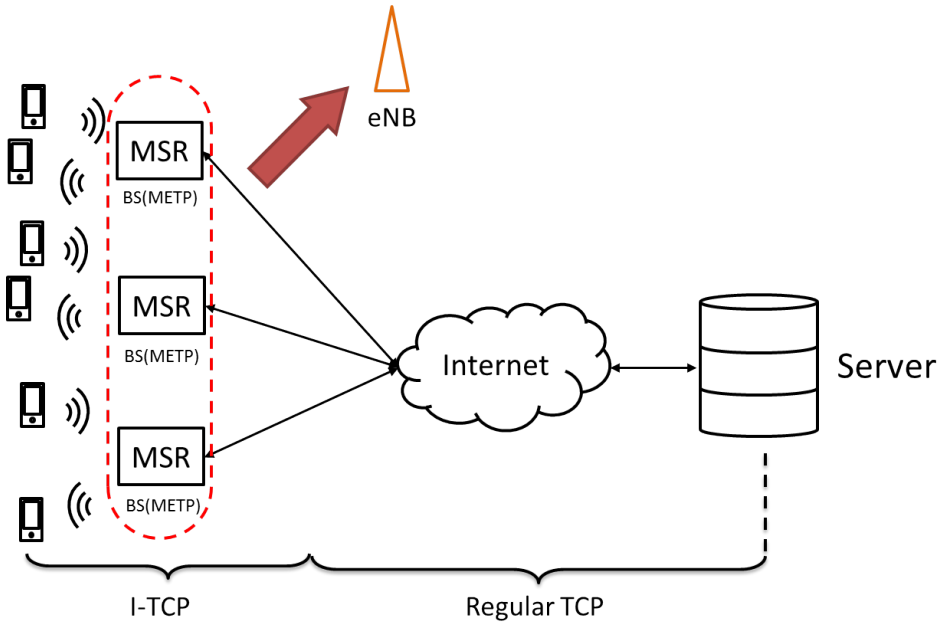



Fig. 3.7 Illustration of ITCP

performance enhancement proxy (PEP) before the packet is sent to the 3G packet delivery network. PEP, as a middlebox, should be located as close to the wireless TCP receiver as possible. Once UE receives TCP requirement, TRL-PEP will request the data from RS till its buffer is filled. Then the TRL-PEP will send the data to the wireless user in predefined typical value according to the study, and the amount is defined as the highest achievable rate. A latest TCP accelerator is proposed and implemented as reported in [23]. The accelerator is designed to be network transparent. It is built below the link-layer and modifies the TCP header to split the connection. The RWND from the UE is replaced by the value related to accelerator buffer status to maximise the send rate of RS while the accelerator will transmit beyond the UE RWND limit as opportunistic transmission and take advantage of ample per user buffer space in the mobile network. The transmission rate control algorithm in the wireless part of the connection is redesigned to exclude rate limit from legacy the congestion control. It will start to transmit at the pre-configured rate, which is the maximum link capacity measured by the network operator and then adapt the speed to prevent it from becoming unbounded. In case of the packet loss, the accelerator will retransmit the lost packet directly thanks to SACK and do not half/cut the transmission rate or go back to the TCP SS status. The transmission rate is only regulated and modified by the controller algorithm proposed in [23]. Last but not least, round-robin transmission achieves the fairness of bandwidth sharing. The accelerator can reach more than 90% bandwidth utilisation in both 3G and 4G network.

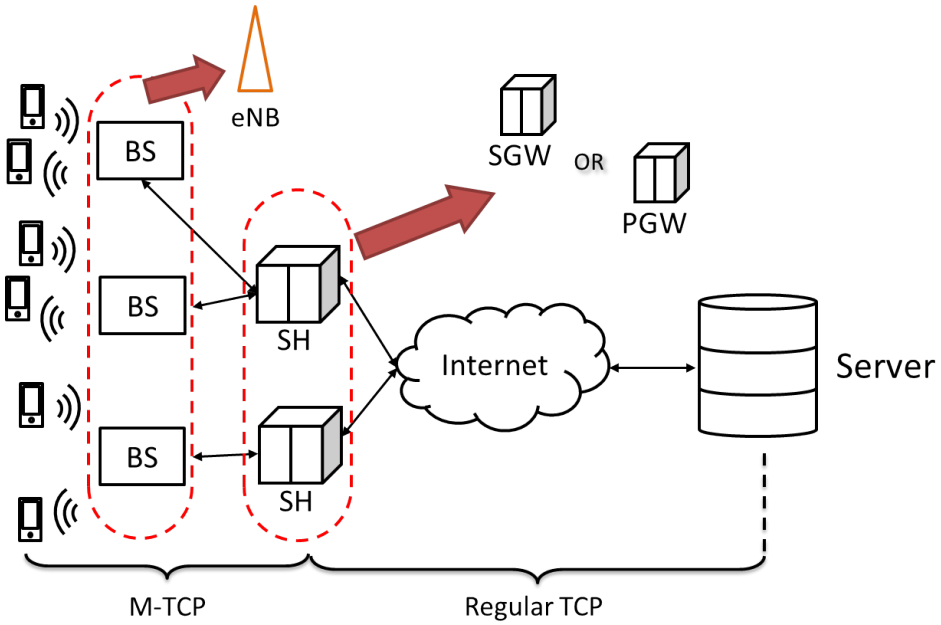



Fig. 3.8 Illustration of MTCP

The architecture and the corresponding LTE adaptation is shown in the Fig.3.9.The role of the relay host is basically replaced by the 3G base station or LTE eNB. When we put M-TCP into the LTE architecture today, the proposed service host(SH)can be seen implemented as one of the functions in SGW or PGW.

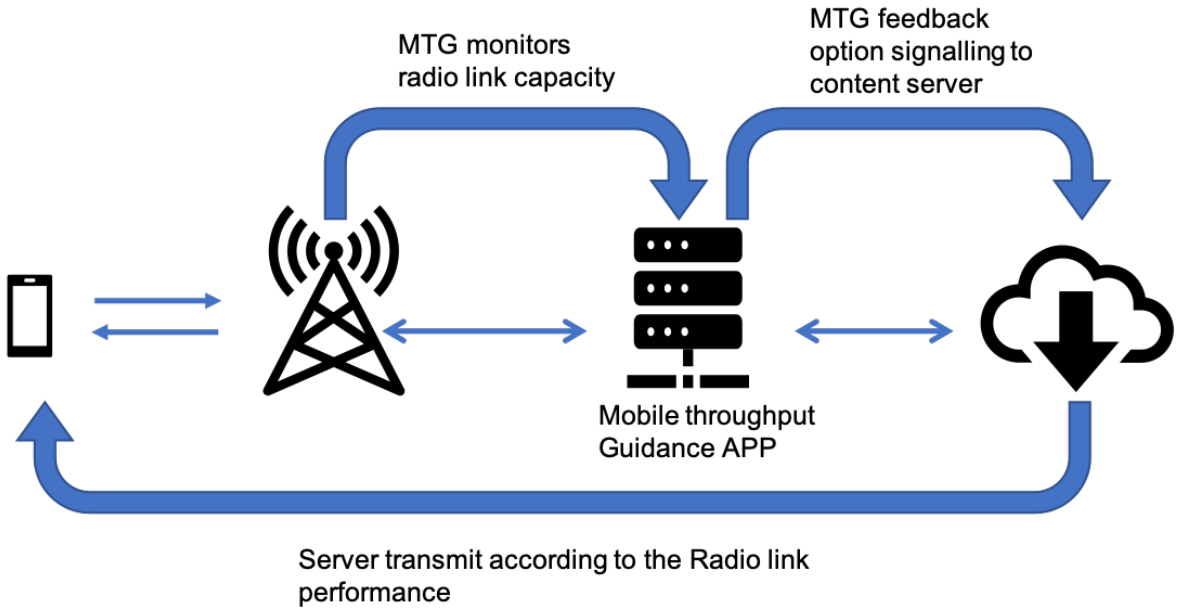



Fig. 3.9 Mobile throughput Guidance

| | Location of Middle box | Preservation of End-to-End Semantics | Compatibility in LTE network and beyond |
|---|---|---|---|
| ITCP | Base Station | No | Need extra layer on BS |
| MTCP | Base Station | No | Need extra layer on BS |
| METP | Base Station | No | Need extra layer on BS |
| M-TCP | One-hop before Base Station | Yes | Yes, as middle box in mobile network |
| TCP packet controller | Base Station | No | Need extra layer on BS |
| SNOOP | Base Station | No | Need extra layer on BS |
| TRL-PEP | One-hop before Mobile network | No | Yes |
| TCP accelerator | In mobile network/backhaul | No | Yes |
| Mobile TP Guidance | In mobile network/backhaul | No | Need edge computing capability |

Table 3.3 Summary of non-cross-Layer middle box solutions

Last but not least, the latest mobile throughput guidance [70, 71] is proposed to monitor the Radio link capacity in the network and feedback the available BW in RAN from eNB to the TCP server for the server to adapt its performance. This method falls in the category of mobile edge computing method and use extra computing power on the edge of the mobile network and send a signalling option field feedback to the TCP server as shown in Fig 3.9. This provides a possibility for us to use new BW option field or the design of BW feedback in our later design in Chapter 5, 6 and 7.

All the methods mentioned above are summarised in table.3.3.

### 3.3.2 Cross-layer protocols

#### End-to-End

The cross-layer design includes implicit and explicit bandwidth estimation(BWE) method. The first two methods we are going to introduce are categorised to include explicit BWE methods(CQIC and piStream) and the last one, Sprout, has an implicit BWE method on the server-side. **CQIC** CQI control (CQIC)[40] receiver feedbacks the estimated bandwidth, which

is calculated by the CQI and Discontinuous Transmission (DTX), for next Time Window to the RS. The server uses the estimated bandwidth to control the packet transmission interval to control the data rate further. The illustration of CQI clients is shown in Fig. 3.10. Since the

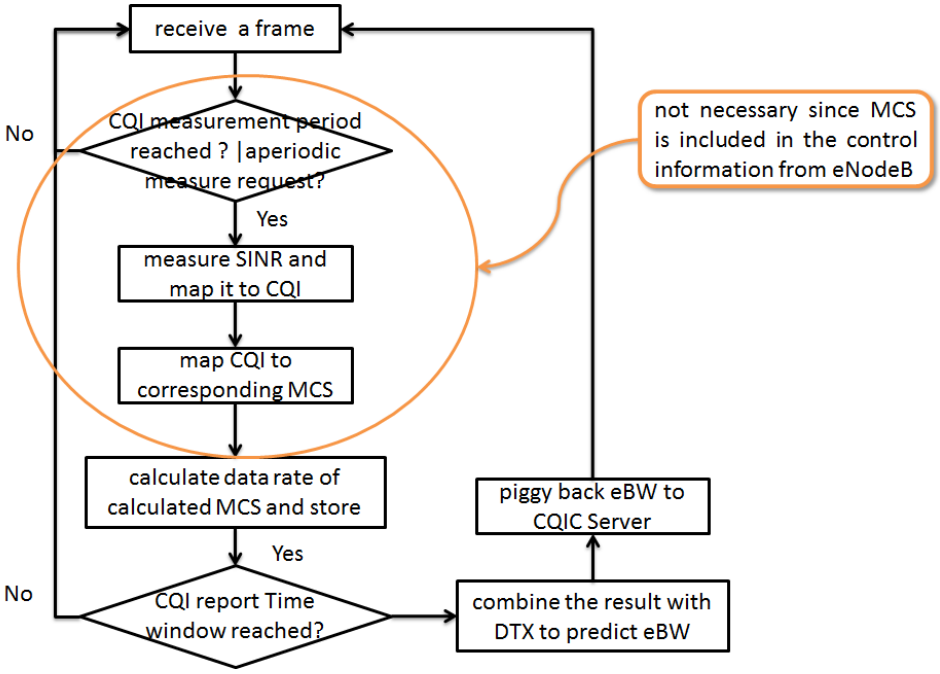



Fig. 3.10 CQIC Flowchart

control frame indicates the MCS of the next receiving frame, from BS to the mobile user, the CQI to rate mapping is not necessary. It is simpler for UE to extract MCS from the Downlink Control Information (DCI) rather than guessing the overall strategically decision of an eNodeB only by the CQI it submits. Accordingly, the possible improvement for CQI presents the Fig.3.11

Furthermore, the time-window-based throughput estimation accounts on the assumption that large-scale fading dominates the status of the channel, which is not always the case. Firstly, the mobility of the UE and the shadowing of the objectives may introduce more small-scale fading. Secondly, the cellular status is kept on changing, handover and the dis/re-connection of the user happened frequently. The eNB will dynamically allocate the resource according to the Overall UE measurement in a proportional fair scheduling manner. Thirdly, the deep individual buffer can introduce an extra delay and retransmission mechanism in eNB.

The previous link bandwidth oriented transport layer protocol design cannot take care of the wired network congestion. CQIC does not employ the TCP Congestion window design which probes the available bandwidth every RTT and adapt the congestion window in AIMD manner. On the contrast, the CWND is defined as the twice the product of minimum RTT and available

bandwidth: $CWND = BW * 2RTT_min$, which is based on the hypothesis of the wireless link is the bottleneck link.

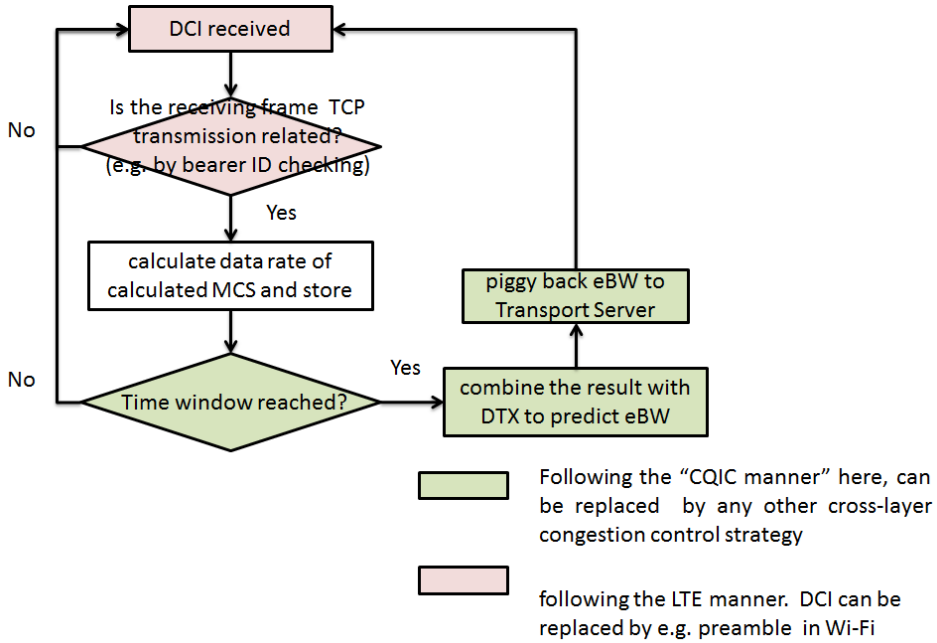



Fig. 3.11 Improved CQIC Flowchart

**piStream**

Different from CQIC, piStream [38] uses a PHY layer Energy-based radio monitor to supervise the available source in the wireless cellular and map the estimation into the available bandwidth. It is a video stream transport protocol cooperates with DASH[72] manner server. According to the theoretical work[73], the accurate prediction of future bandwidth can improve DASH performance significantly. Instead of predicting the exact value of future bandwidth, the Long Range Dependence (LRD) of LTE downlink Traffic is constantly estimated. Pareto probability function is used to calculate the data rate level change probability is shown below:

$$\mathbb{P}\{T_s > t\} = \begin{cases} (\frac{\alpha}{t})^{\beta} & \text{if } t \geq \alpha \\ 1 & Otherwise \end{cases} \tag{3.4}$$

$$p_s(t) = \mathbb{P}(T_s > t + \Delta t | T_s > t) = (\frac{t}{t + \Delta t})^{\beta} \tag{3.5}$$

where the $\Delta t$ is the time difference between two ACK packets and where $\alpha$ and $\beta$ are referred to as the scale and shape parameter for the Pareto model. Intuitively, the longer a UE stayed in one level, the higher possibility for it to change the state. The parameter of the probability is linearly regressively analysed by the data rate in a time window. The flowchart of the piStream

client function is shown in the Fig.3.12 The implementation of USRP offers the radio resource

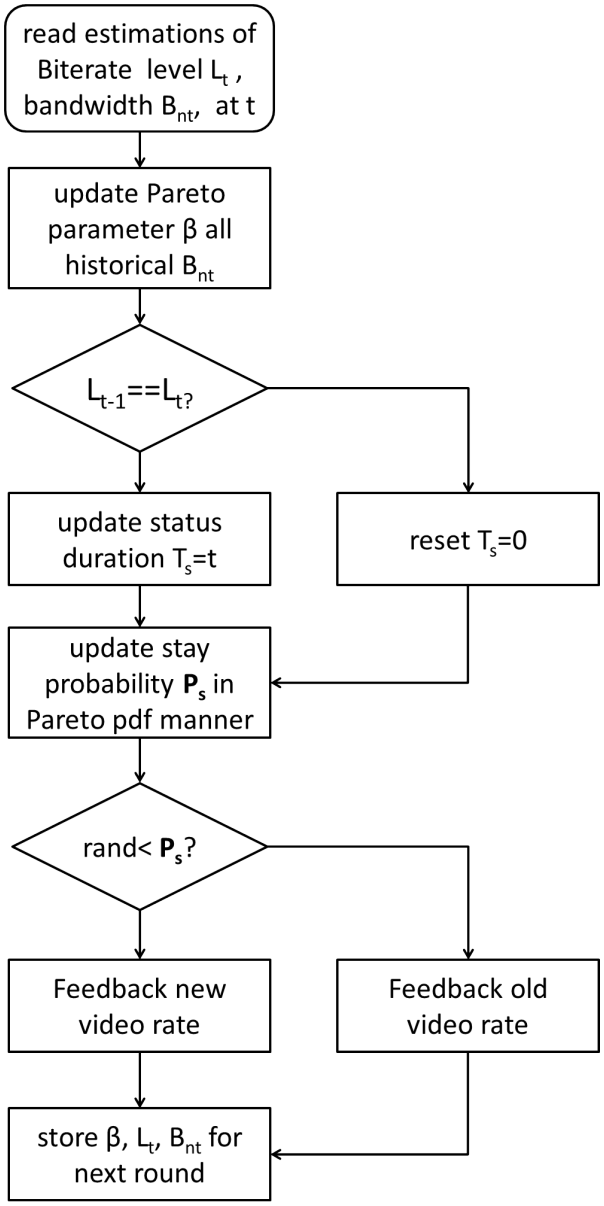



Fig. 3.12 piStream flowchart

monitoring statistics over the 200ms time window, which is $1/10$ of the DASH segment length. In this case, the piStream estimate and feedback the channel estimation in real-time. Note that its experiment also validates the linear relationship between PRB utilisation and available bandwidth. Based on all these reasoning and validation, piStream gives a reasonable better performance than other existing video streaming protocol benchmarks. The shortage of piStream is similar to CQIC. Even if the LRD property has been well-established [74, 75], of the bandwidth estimation of the current stage is not sufficient.

### Sprout

Sprout[76] is another end to end transport layer protocol which focusing on achieving low delay and high throughput for the mobile network. The protocol makes a short-term forecast of the bottleneck link rate using probabilistic inference which employs packet arrival rate. The assumption behind this protocol is to models the link and estimates the link behaviour with reserved uncertainty. The model is then used to forecast the bits which may be transmitted from the queue shortly. Sprout is designed based on the following assumptions:

1. Cellular last is the bottleneck link in the End-to-End connection route.

2. The individual queue for each UE is implemented in the eNB

3. All the competing flows are wrapped in the Sprout

4. The prediction of safe range of data is made by the sender:a cautious estimate, at the 5th percentile, of how many bytes will arrive at its receiver during the next eight ticks or 160ms.

| | Assumption and concept | TCP family | other Layers needed |
|---|---|---|---|
| CQIC | Server estimate the end-to-end BW by piggyback cross-Layer information. Server estimate the BW by proposed algorithm | No | MAC Layer |
| piStream | server infer the available BW by piggybacking cross-Layer information. Server update the prediction model according to | No | PHY Layer, MAC Layer |
| Sprout | Receiver infer the digestible packet according to Brownie Motion model and piggyback the prediction. | No | MAC Layer |
| EJTCP | Server infer the Packet loss by monitoring the jitters of delays. The parameter is updated according to MAC layer packet loss information | Yes | Wi-MAX MAC Layer |
| XJTCP | Server infer the Packet loss by monitoring the jitters of delays. The parameter is updated according to MAC layer packet loss information | Yes | Wi-Fi MAC Layer |

Table 3.4 Summary of novel e2e solutions

The experiments shows that SPROUT can provide more throughput and the less RTT. Such model takes the shared queue into the consideration and its accuracy of forecasts depends on whether the application is providing offered load sufficient to saturate the link.

Different from those above new cross-layer protocol, EJTCP [77] and XJTCP [78], as the enhancements of JTCP, keep the TCP congestion control feature. They used MCS information of WiMax/Wi-Fi MAC information to be piggybacked in the optional field of TCP header, and the sender will change the performance accordingly.

From the table 3.4 The novel e2e solutions tried to use the information as explicit or implicit feedback to the server to calculate or predict the potential data rate for the radio link. They achieved better performance compared to the default of TCP/IP. The problem is they are designed to deal with a network with a different type of PHY/MAC design or to work with a fixed network. In the mobile cellular network, we need a different architecture.

#### Middle box supported cross-layer solution

The network devices between the two end users can manipulate the header of the packet to inform the endpoints about the upcoming congestion or the status of the network. Explicit Congestion Notification (ECN) [79] is a cross-layer solution that allows the sender to be informed of the network status. It requires the coordination of-of both IP and TCP layer. When a pair of ECN server and client is communicating while one ECN-capable router detects the possible congestion with Random Early Detection (RED), the router will set the two ECN bits in Differentiated Services (DS)field in IP header to 11 and forward the packet to the receiver. The receiver learnt the congestion by decoding the IP header of the data packet. In the ACK of the data, the receiver returns congestion information by setting the ECN-Echo (ECE) bits in the corresponding ACK to 1. The data sender will get half its transmission rate once it received the ACK with ECE bit=1, and return 1 in Congestion Window Reduce(CWR) bit to indicate the congestion information is successfully obtained. This solution requires all the router and both end-users in the network to support the ECN mechanism to work. Otherwise, the performance may be affected.

Explicit Loss Notification (ELN)[10] is an End-to-End cross-layer solution in TCP manner. The last hop host informs the receiving transport layer that the packet from server reaches the wireless link transmitter, but the packet loss due to the harsh wireless environment. The receiver thus reports the wireless loss in ELN bits of ACK. The sender transport layer will decide to invoke CCA according to the ELN bits in the ACKs. As long as wireless link loss causes the loss, the CWND is unchanged. Otherwise, the congestion algorithm is invoked. Such design based on the hypothesis that wireless link is lossy and the design of signalling informing receiving transport layer is tricky. One more practical improvement of ELN design is

Explicit lousy state Notification[80]. When the wireless link is in a bad status, the Base station will send an EBSN to the RS. The server will reset the timer according to the current estimation of RTT. This requires the support of the base station, and whichever layer the notification is located (TCP/IP or above)such design requires the extensive modification of the base station.

The SNOOP[81] and its follower use the cross-layer information available in the base station to manipulate the TCP packets. The proxy and delay operation of Data and ACK packet gives the throughput improvement on the legacy TCP performance.

Slo-Mo[17] is one of the latest innovative solutions to allow the middlebox to estimate the buffer status in the BS and vary the packet delivery speed from the middlebox to the BS despite the flow rate from the server to the UEs. This requires the smart queue strategy in the middlebox. Since the middle-box needs to get access to all the traffic in the mobile network, the proposed location of the middle-box is LTE PGW.

The comparison of the mentioned middlebox solution is summarized in the table below:

| | Device Modified | Dedicated to TCP | Layers needed |
|---|---|---|---|
| ECN | Routers,Server, UE | Yes | Network Layer, Transport Layer |
| ELN | Server, BS, UE | Yes | MAC Layer, Transport Layer |
| ELN-ACK | Server, BS, UE | Yes | MAC Layer, Transport Layer |
| SNOOP | BS, UE | Yes | MAC Layer, Transport Layer |
| Slow-Mo | PGW | No | Network Layer |

Table 3.5 Cross-Layer middle box solutions

### 3.3.3 Other solutions

Apart from the protocol designs, some other solutions are indicating the changes should be made in endpoints, Base stations, routers and other nodes in the network.

The scheduler in the eNB controls the wireless resource management among its UEs and the most widely used scheduling algorithm is the Proportional Fair (PF)[82]. As [83] suggested, the performance of the PF scheduler varies with the constraints such as the number of serving users. Hence the scheduler parameters should be adapted dynamically. Thought PF scheduler might outperform other scheduling algorithms on some specific use case, e.g. DASH streaming[84], [85] reports the discovery that the one-way delay (OWD) of Internet packets transmitted across an LTE or HSPA link depends on the absolute sending time within a 10ms scheduling interval and recommends to align sending times of applications with time slots accordingly as a way to reduce the overall delay.

The RWND size may also be one of the possible limiting the throughput of the TCP. RWND is defined by the UE vendor for some reason according to [32], and it proposed Dynamic receive window adjustment (DRWA)to adopt the RWND dynamically for both better throughput and lower delay.

## 3.4 Discussion and conclusion

The lossy wireless link is hidden, from the transport layer, thanks to the lower layer protocol in a mobile network. This manner is very likely to still be true this concealing manner from 2G to 5G[86] and beyond. Especially in the 5G network, the range of capacity can vary from Kbps up to Gbps level while the buffer size is about 15MB the bufferbloat variation will be more severe as [87] indicates.

The hidden loss is a tradeoff at the cost of time. Hence the RTT in an LTE mobile network is unstable, and different time window size can also reflect different BW which links to the instability of radio link. The Dynamic MCS, scheduling algorithm and the individual bearer buffer, in the lower layer, reduce the loss probability and increase the time domain instability of BW observed by the higher layer. On the other hand, ACK/NACK, ARQ, reordering, and Error correction used in the LTE protocol stack further mitigates the potential loss.

In such design logic, stead of loss, varying throughput and higher e2e RTT variation are presented to the transport layer in a mobile network. To maintain the better overall bandwidth utilisation, a CCA must not only track the bandwidth in a hybrid network but also maintain the small queue size at a certain level.

To clarify our assumption, illustration and tradeoff, the abstract network model is demonstrated Fig.3.13. The e2e reaction delay is at least $OWD = t_1 + t_2 + t_3 + t_4 + t_5 + t_{allprocessing}$ later than the . The use of the middlebox can reduce the $OWD$ by geographically shorten the distance of the sender and receiver, which may improve the accuracy of the bandwidth estimation. However, the cost of the solution and the difficulty of implementation may vary according to the place of the middlebox/module/edge-computing, as concluded in the table below. Recalling the tradeoff metrics of tradeoff among the objectives shown in the introduction

| | Middle Box | | | | |
|---|---|---|---|---|---|
| Metrics | Base Station | Mobile Backhaul | Near PGW | Internet | End-to-End |
| Hardware Cost | Highest | High | Lowest | Higher | Low |
| Real-Time | Highest | Higher | High | Low | Lowest |
| Deployment | Difficult | Easy | Easiest | Difficult | Difficult |

Table 3.6 Merit of the solutions

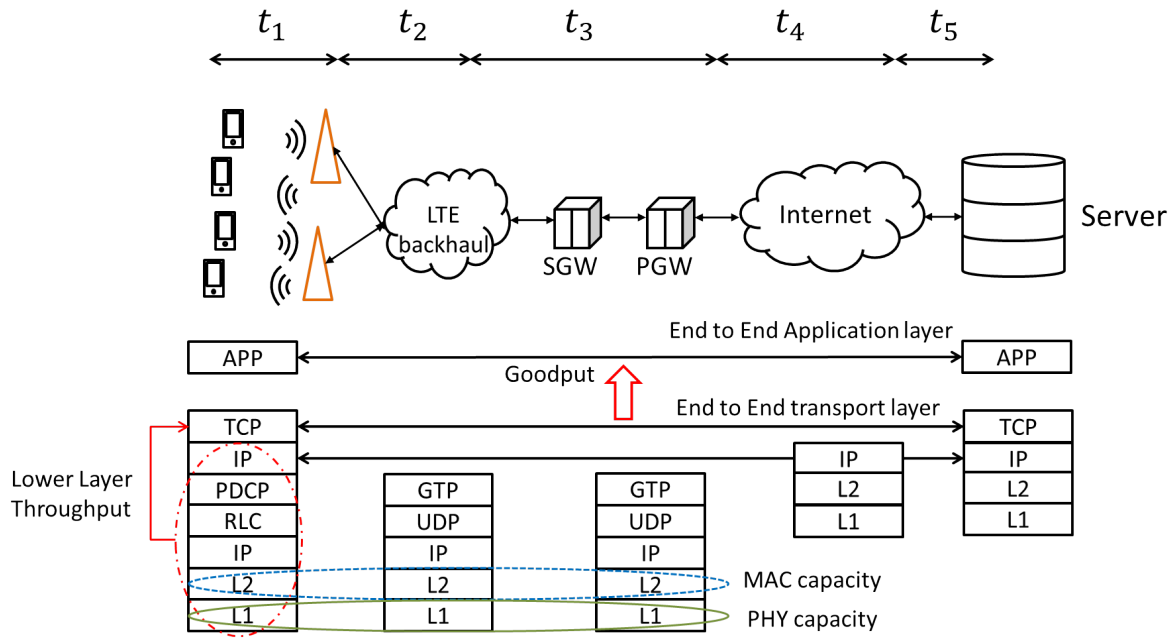



Fig. 3.13 Time consumption in different section of a Mobile edge network

chapter in Fig. 3.14 . The design must take care of the delay/loss feature in a mobile network while the bandwidth utilisation should also be high enough. Since the lower layer design is complicated enough, the transport layer CCA should be as simple as possible to reduce the processing power. In this chapter, we discussed the utilisation, delay/loss genericity from a layered perspective of view. In the following chapter, we are going to discuss further discuss the delay, fairness and genericity from a network perspective. Based on the hypothesis that

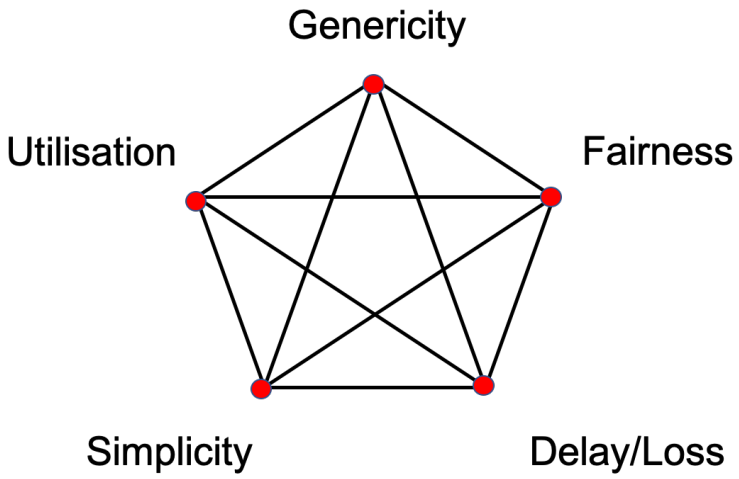



Fig. 3.14 Goals and tradeoffs for an ubiquitous CCA design

bottleneck located on the wireless link, some end-to-end solutions are proposed in a cross-layer or non-cross-layer manner e.g.CQIC piStream, etc.. However, the performance of pure wireless link oriented CCA will decreases, if the wireless link is no longer the bottleneck.

The concern in designing our congestion control strategy in a mobile network are:

- The server should be able to know the capacity of the bottleneck so that the reaction can be as real-time as possible.
- To allow the server-side to get the real-time bottleneck bandwidth, the client should be able to
    1. trace the adapting lower layer throughput at a reasonable level if the radio link the last hop is the bottleneck.
    2. be able to report the remote bottleneck bandwidth if the radio link is not the constrain in a network
- The server should also be the ability to distinguish the congestion and random loss even if the advanced hardware and protocol in the internet and mobile network can prevent most of the random loss.
- Distinguish the buffer in network device from the bandwidth-delay product and treat the buffer size in the bottleneck as the redundancy and flexibility for operations. As a result, to avoid the bufferbloat in the Base station.
- Ideally, the location of traffic control server for data transmission for a mobile network is the gateways which manage the data traffic. That is to say, the PGWs or SPGWs in a mobile network. Based on the hypothesis that bottleneck located on the wireless link, some end-to-end solutions are proposed in a cross-layer or non-cross-layer manner e.g.CQIC piStream, etc..

According to the summaries above, we focus on CQIC and TCP BBR as the starting point of the research. The former is the most recent well-deployed TCP CCA version while the latter originates the idea of this thesis. Based on CQIC, we made our first proposal, DCIC, which was also known as TCP-CQIC-LTE. Throughout the rest of the thesis, represents the DCIC/TCP-CQIC-LTE proposal we made.

The general idea and tools used for CCA design are introduced in the next chapter. Detail of the two algorithms is briefly reviewed in chapter 5. Their pros, cons and the performance is also validated in the latter chapters. The designs to look beyond the two CCAs for mobile traffic management are proposed in the last two chapters.

# Chapter 4

# Models and Tools used in Congestion Control Algorithms

## 4.1 Introduction

The categories of the congestion control strategy, architecture and algorithms are reviewed in the last chapter. In this chapter, the general design principle of CCAs will be analysed and discussed. According to the discussion in last chapter, the generalised principle and technical details for CCA designs are discussed in this chapter:

1. The design logic of a congestion control: take TCP as an example.
2. The tool for CCA analysis: TCP Transmit/Receive Sequence Trace figures
3. The reason why multiple received ACK is necessary to confirm a phenomenon in the network: Take DUPACK as an example.
4. The configurations for Probe more period: Take BBR STARTUP parameters as an example
5. Analyses on Pacing traffic and bursty traffic: which one is better.

## 4.2 Design logics in a Congestion Control Algorithm: a TCP example

The conventional design of loss-based TCP contains congestion window(CWND), slow start threshold($ssthresh$). Typical CCA is achieved by probing the network status via the ACKs

received. The server will maintain a congestion window (CWND) and infer the network congestion by the ACK timeout or the number of the successive ACKs (a.k.a Duplicated ACK) with the same sequence number (SN). With the help of the SW mechanism, TCP is typically composed of three status: Slow Start (SS), Congestion Avoidance (CA) and Fast Recovery (FRecv). The data transfer starts with SS status after the connection establishment. The CWND will be increased linearly whenever a new ACK arrives with a new SN and will fall back to the minimum when a timeout occurs. Once CWND reaches SS threshold, the server will transit to CA status. CA status acts in additive-increase, multiplicative decrease (AIMD) manner. Whenever an ACK with new SN is successfully received, the CWND increases linearly. When Third Duplicated ACK (DupACK) is received, an FR is triggered to mitigate the possible non-congestion caused packet loss, and the CWND is halved from the previous value. If the DACK continues, the server will conclude that there is congestion in the network and set the CWND to the minimum value and enter the slow start again.

In the mechanism described above, the whole network is abstractly modelled as a long pipeline plus a buffer to bear the queue as in Fig.4.1. The buffering capacity $2 * N$ represents

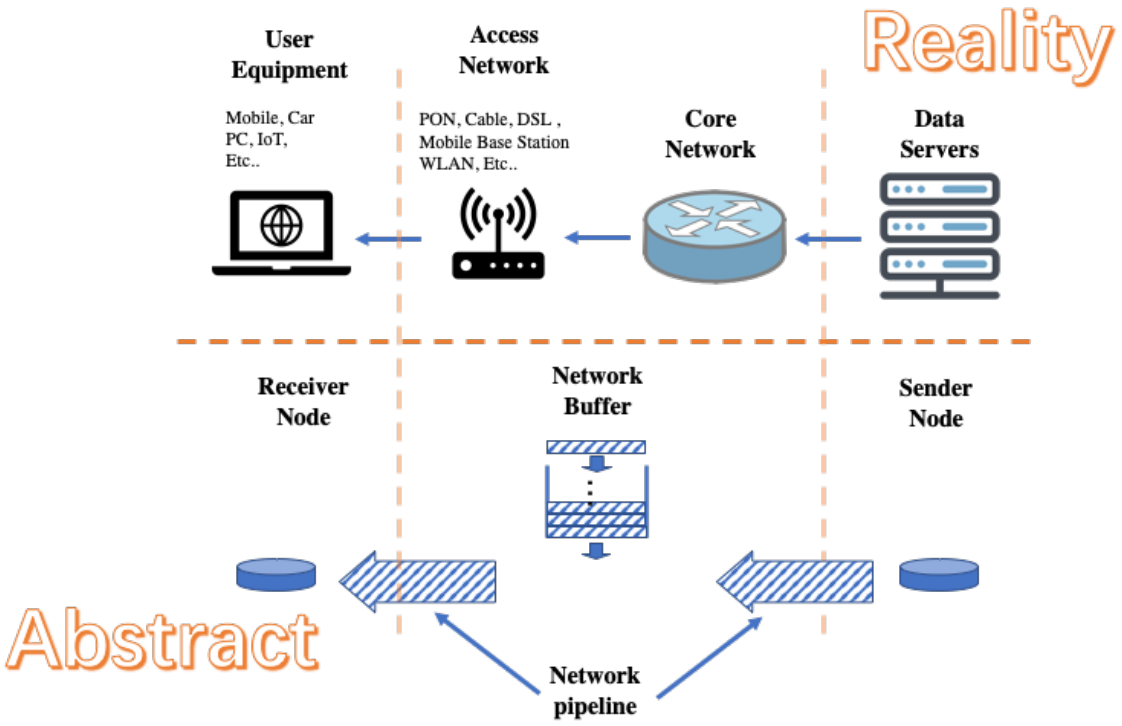



Fig. 4.1 Abstract of the network assumption in the thesis

the total queue availability in all the nodes in the network and the network itself. From sender

A to receiver B, the tipping point of loss is defined by the equation below:

$$2 * N = RTT * deliveryRate \tag{4.1}$$

which means the network is fully occupied by the data sent by A and the ACK replied by B. The assumption ACK size is equivalent to the data size. Once a loss detected by the predefined method e.g. duplicated ACK or timeout, the maximum available capacity $C$ in the network is detected. *ssthresh* is then set as $C/2$, and congestion avoidance stage is initiated. Until now, current network characteristic is known by the sender. Considering the network reality, the perfect C may never be reached. As a result $ssthresh < C/2$, but *ssthresh* would not be too far from $C/2$. Also, due to the complicated network condition, congestion control algorithm should be careful when probing in this 'dangerous' area. Hence the reserved congestion avoidance manner is used under this condition. This is the basic principle of how loss-based congestion

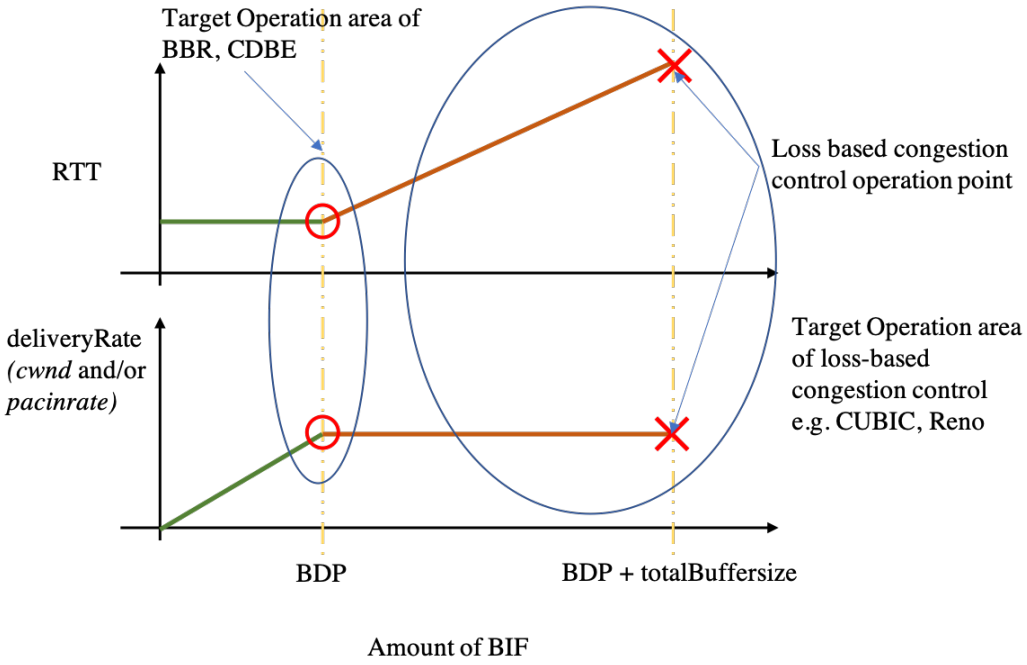



Fig. 4.2 Compare the target operating point and area of loss-based Congestion control and bandwidth/round trip delay based control algorithm

control algorithm probe bandwidth and their engine is basically the loss of a packet. The typical buffer depth in a mobile network is 3MB to 6MB, and the BW to digest such amount of buffer is from 1Mbps up to 500Mbps. Hence the theoretical maximum delay can be calculated:

$$\Delta Delay_{max} = \frac{Q_{max}}{BW_{min}} = 48s \tag{4.2}$$

which is far above the majority of default RTO limit used in TCP nowadays. As long as there is an AIMD CCA, there will be an extra delay which is the cost to reveal the point of operation, and their operation are shown in Fig.4.2: Loss based CCA reacts to the loss which occurs when the buffer is fully filed if no random loss policy is deployed, and the BW-Delay based CCA will operate around the point between an almost filled pipeline state and a slightly built up queue.

#### The logics behind the Congestion control

According to the analysis in previous chapter and last subsection, the duty of a congestion control algorithm are concluded. It includes the following procedure TCP is trying to do the following procedure:

Step 1. To find bandwidth in current time period

Step 2. To avoid congestion in the bottleneck

Step 3. To probe more bandwidth, and go back to Step 2.

This is not only true in BBR or the following designed CQIC-s or CDBE(v2) but also the general design principle in CUBIC/Reno or any other AIMD/Loss-based CCA.

## 4.3 TCP Transmit/Receive Sequence Trace figures

A example of TCP Sequence Trace (ST)is illustrated in Fig.4.3. The red line is the ACK with specific Sequence receiving moment on the receiver side. The solid green line is transmitted packet with specific transmission moment. The green dotted-dashed line is the Ideal sending sequence which will not cause extra queue-built in the BN buffer. Assuming the network has enough buffer space. The sender starts to send at $t_0$. Note that the STARTUP stage ignored. The horizontal difference between the received ACK line and the actual sending sequence line is the actual round trip propagation experienced by each packet. The vertical difference between the two lines is the Bytes in Flight (BIF). Ideally, it equals to the Bandwidth Delay Product (BDP). Before $t_1$, the sender is sending at an ideal rate which means there is no queue building up in the network, RTT is equivalent to the minimal Round Trip propagation ($RT_{prop}$), and the minimal BIF is observed. At $t_1$ sender rise its sending rate for some reason(e.g. congestion control probing). The queue in the BN builds up due to the rate mismatching between the sender and the bottleneck. As a consequence, RTT witnessed in by Timestamp (TSval-TSecr) in bottleneck also increases. Between $t_2$ and $t_3$, the sender slows itself down a bit, and the sending rate is equivalent to the $BW_{Btlnck}$. Hence the RTT stops growing as well as the queue

length stays the same. After $t_3$, the sender sends at an even slower rate, which is lower than the $BW_{Btlnck}$. Essentially, this move will release the congestion in the bottleneck. Come with the emptying queue, the shorter RTT will be observed. At $t_4$, the slope of ideal sending sequence

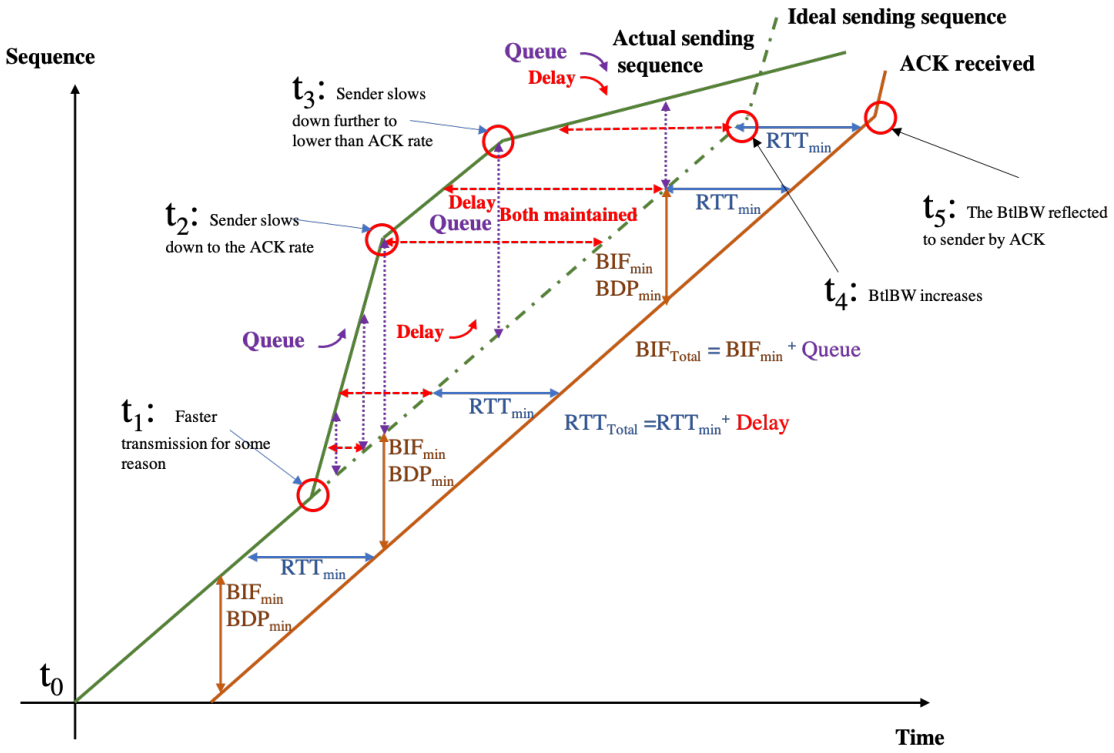



Fig. 4.3 Reading the TCP Trace figure

line increases as a result of an increasing available $BW_{Btlnck}$. This variation is reflected in the sender at $t_5$. We conclude the feature of the TCP trace figure below:

1. The horizontal difference between actual /ideal sending and ACK lines is the actual /minimal RTT.

2. The vertical difference between actual/ideal sending and ACK lines is the actual/minimal Bytes in Flight.

3. The slop of the lines reveal the rates:

   (a) The slope of the actual sending sequence line is the sending rate of sender.

   (b) The slope of ideal sending sequence, in our assumption, the line represents $BW_{Btlnck}$.

   (c) The slope of ACK reception line reflects network capacity 'a while ago'.

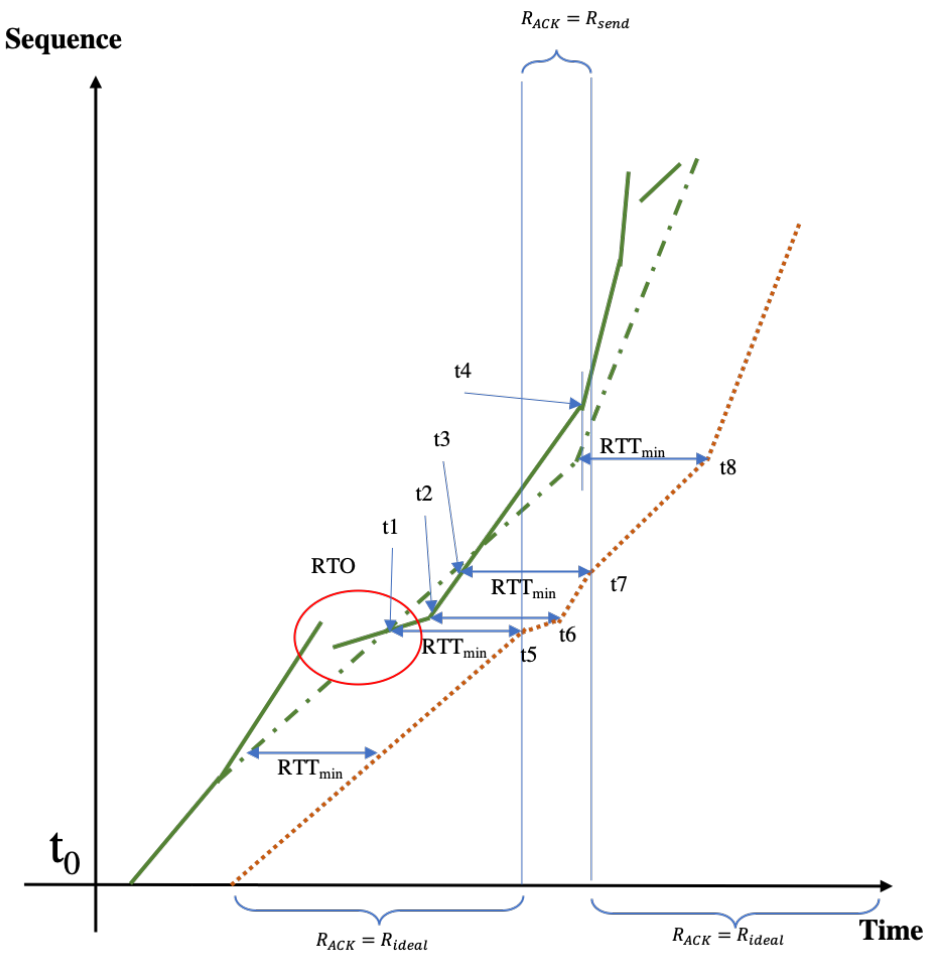



Fig. 4.4 Rough working pattern of Loss-based congestion control.

Further extend the usage of Fig.3.6 in chapter 3, we use it to illustrate the loss-based congestion control in Fig.4.4. The green dotted-dashed line is the 'ideal' sequence of data transmission if the sender sends the data at the 100% $BW_{Btlnck}$ while no queue piles up network buffer. The solid green line is the sender behaviour on sending the data with loss-based congestion control. The red dotted line is the received ACK sequence alone the timeline. The scale is altered for a better visibility With the help of this trace figure, we now analyse the behaviour of a data sender and the available measurement from ACKs. After the SS state and the first RTO, the *ssthresh* is recorded on the sender. The sender keeps on decreasing its CWND, which means the sending rate is decreasing in fast recovery. At $t_1$, the sending rate declined to the ideal rate, and one $RTT_{min}$ later, the decrease of the sending rate start to be visible on the received ACK sequence. The fast recovery ends at $t_2$. The sender will increase the and sending rate according to the *CA* stage algorithm. For the same reason, 1 $RTT_{min}$ later, the sending rate variation at $t_2$

can be seen from the measurement of ACK. Actually, any the acts taken on the sender side can be seen from the ACK arriving pattern at least $RTT_{min}$ later. Ideally, the sending rate, which is slower than $BW_{Btlnck}$, between $t_1$ and $t_3$, is reflected on slope of ACK sequence between $t_5$ and $t_7$. At $t_3$, the send rate exceeds the $BW_{Btlnck}$ once again, and the buffer started to build and the vertical gap between data sequence and ACK sequence grows. As a consequence, the horizontal distance between the two lines, which indicates that the actual $RTT$ are also growing. After $t_3$ the ACK rate reflects the $BW_{Btlnck}$. As long as the data sequence is above the ideal sequence line, the time-averaged reflected $R_{ACK}$ is always equivalent to the $BW_{Btlnck}$. Once the $BW_{Btlnck}$ increases at $t_4$, the sender will try to grab the newly available resource and the queue growth till loss pattern will repeat.

In summary, by analysing the Sequence Trace Figures, we can design, debug and validate our own CCA architecture, features and protocols according to the ST variation on data Tx and ACK Rx for a single flow of data.

## 4.4 The STARTUP procedure in bottleneck bandwidth and round trip delay-based congestion control

As the latest congestion control, BBR is evolving rapidly in the latest years [88]. In this thesis, the BBR is one of the main benchmark used. As a successor of RTT based congestion control, BBR inherits the concept of using BDP: product of BW and RTT as the last equation in equation3.2. The original BBR characterises the network with two main parameters: $BW_{Btlnck}$ and Round Trip propagation (RTProp). Compared to the conventional congestion control, the optimisation goal is set: maximise the throughput and lowest delay. Furthermore, the parameters should meet the following two conditions:

- 1. the packet arriving speed equivalent to $BW_{Btlnck}$
- 2. Bytes in flight is equivalent to the target bandwidth-delay product (BDP)

The target BDP in the BBR context is the product of $BW_{Btlnck}$ and the minimum RTProp. As long as the conditions above are fulfilled, the bottleneck is 100% utilised while the bufferbloat is prevented. However, the share of $BW_{Btlnck}$ allocated to each traffic flows varies through time. As a consequence, the minimum RTProp is also varying. Hence the sender will keep on measuring the two parameters on every received ACK at moment $t$. In [89] BBR models minimum RTProp at moment $t_i$ as:

$$RTProp_i = RTT_i - \eta_i \tag{4.3}$$

where $\eta_i$ is the noise introduced by several potential factors: delayed ACK, ACK aggregation etc.. Hence, as the majority of other measurement methods, the RTprop is approximated as minimum RTT in a windowed-time manner.

$$\hat{RT}_{Prop} = RTT_{min} \tag{4.4}$$

As for $BW_{Btlnck}$ at moment $t$, it is defined as the time-windowed maximum of average deliver rate. The delivery rate is defined as below:

$$\begin{aligned} deliveryRate_{t_i} &= \frac{\Delta P_{delivered}}{\Delta t_i} \\ \hat{BW_{Btlnck}}(t) &= max(deliveryRate_t) \end{aligned} \tag{4.5}$$

Where $P_{Delivered}$ is known precisely, while $\Delta t$ is noise sensitive due to the dynamics of the network. The $deliveryRate$ should lower than $BW_{Btlnck}$ so that the traffic is not inflating the network. Plus another basic assumption is that the delivery rate on receiver side is not higher than the sending rate, hence the time elapse for bandwidth estimation is defined as follow:

$$\begin{aligned} &\Delta t = max(\Delta t_{acked}, \Delta t_{sent}), \\ &\Delta t_{acked} = t^i_{acked} - t^{i-1}_{acked}, \\ &\Delta t_{sent} = t^i_{sent} - t^{i-1}_{sent} \end{aligned} \tag{4.6}$$

By using this method, the potential error introduced by delayed ACK is eliminated. Both approximations above updates in time windowed manner, and pacing rate of transmission is given by the estimated $BW_{Btlnck}$:

$$pacingRate(t) = G_{Startup} * \hat{BW_{Btlnck}}(t-1) \tag{4.7}$$

Similar to Cubic, the startup stage of BBR tried to converge to a smooth curve with the target of double pacing rate per RTT:

$$pacingRate = f(t) = 2^t \tag{4.8}$$

To derive the Gain in Startup stage of BBR, the bandwidth estimation and the integrate the pacing rate above over RTT:

$$\hat{BW_{Btlnck}}(t) = \int_{t-2}^{t-1} f(t)dt \tag{4.9}$$

combines equation 4.7,4.8 and 4.9, we can derive $G_{startup} = 4ln2 \approx 2.77$. It differs from the original proposal $2/ln2$. The original $G_{Startup}$ can be derived as follow:

$$\begin{aligned} G_{Startup} * f(t-2) &= \Delta P_{send}(t-1), \\ \Delta P_{send}(t-1) &= \int_{t-1}^{t} f(t)dt = \frac{2^t}{2ln2} \end{aligned} \tag{4.10}$$

Substitute equation 4.8 into 4.10, we get $G_{Startup} * 2^{t-2} = \frac{2^t}{2ln2}$, and $G_{Startup}$ is calculated as $2/ln2 \approx 2.89$. Both value make sense while currently 2.77 is currently used in BBR development google group [1]. The original design aimed to delivered packets smoothly and converged to curve 4.8 as quick as possible. However, this intention is very likely to suffer from traffic variation in the network.

The second problem of BBR is the convergence speed. The main convergence effort is designed to be made by the Gain and drain cycle. However, the fixed Cycle length of 8 $RTT_{min}$ made the convergence speed to below, and the ProbRTT stage is synchronised due to the coexistence of multiple flows are feeding the round trip extra amount of data.

## 4.5 DupAck procedure for the lower bound to confirm a trend

Despite many variants, BBR series ignores the duplicated ACK Recovery stage and only follows the general BBR state transition logic. The original Dupack logic is illustrated in Figure 4.5. The reason why we talk about this topic is that in our following design in Chapter 7, an RTT based decision threshold is employed. A reasonable method to use decide the number of repeated RTT to confirm the trend and to eliminate the jitter in RTT for an RTT varying network is necessary. In Fig.4.5, Assuming the Repeated data Sequence number is N and all the transmissions before N are successfully received and note that the ACK sequence($Seq_{ACK}$) of N-1 is N. For the following three ACKs if we have the all the permutation shown in Fig.4.5, we can see that: If we received the second DUPACK, there is a probability that there is no loss at all. If there is no loss, the correct reception of ACK for N ($Seq_{ACK} = N+1$) is equivalent to the probability of $Seq_{ACK} \neq N+1$. If there is a loss, we will receive at least 3DUPACKs.

Hence if we choose to retransmit directly with two-DUPACK threshold, it may affect the transmission of the next valid data packet and result in the significant degradation of goodput if the reordering happens regularly.

[1] https://groups.google.com/forum/#!forum/bbr-dev

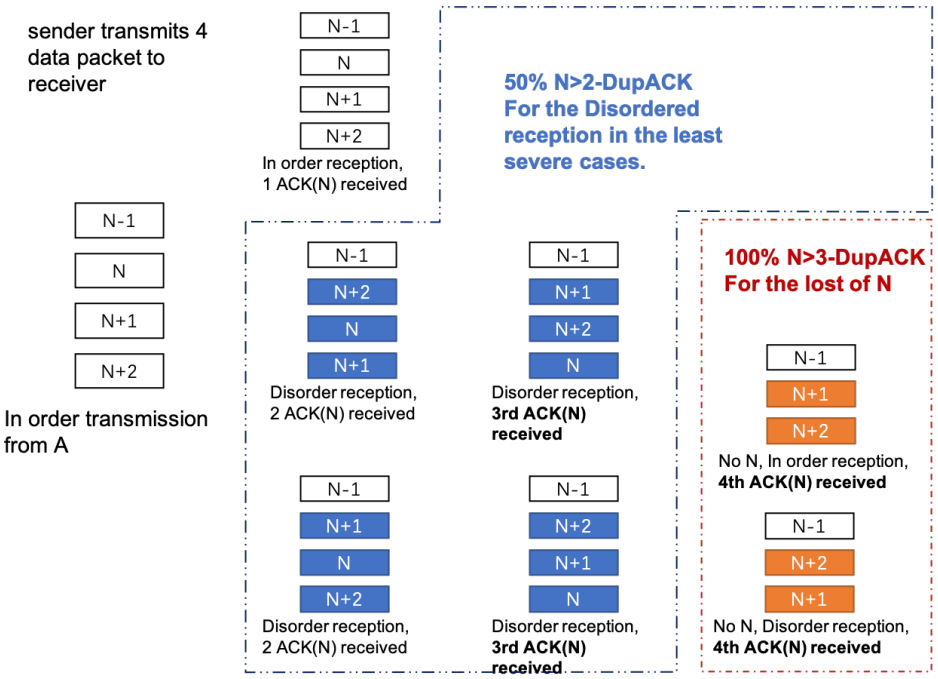



Fig. 4.5 Simplest case for 3 Dupack

Inherited from this concept, we will set the threshold for per-packet based RTT related threshold to a value larger than 3. Furthermore, to mitigate the retransmission in the RLC layer, the RTT gradient of a valid sample should also be positive. This feature will be further described in Chapter 7.

## 4.6 BURSTY traffic and PACING traffic

As mentioned before, the conventional AIMD model has low bandwidth efficiency. The HTTP protocol nowadays tends to create multiple TCP flows at the same time. More computational resources have to be spent on the management of more and more connections and on the synchronisation of data. For those reasons, we believe it is reasonable to improve single connection bandwidth utilisation. For the single flow to better utilise the bandwidth, it is necessary to use pacing feature. In this subsection, we model and validate the merit of pacing feature for a single flow connection. We define the cost of the network on a time scale. $C_{idle}$ is the idle time of a network device in a unit time. $C_{Queue}$ is the time of processing the queued data in unit time.

$$C_{idle} = UnitTime - \frac{\lambda}{BW} \tag{4.11}$$

where, $UnitTime$ is a constant we defined as 1, and $\lambda$ is the amount of data arriving in a $UnitTime$. $BW$ is the bandwidth of the network. The assumption is that the network has enough buffer and loss is not the The conventional TCP AIMD CCAs uses ACK driven transmission manner. In more and more CCAs, pacing has been introduced to shape the traffic. Since in reality, the network is not pacing the ACK of data sequences perfectly as Van Jacobson expected [90]. The difference is caused by several factors:

- Multiple TCP flows coexist, and the disappearance and the arrival of flow are unpredictable.
- The upstream route of the downstream can be heterogeneous.
- The loss of ACK
- some router has ACK aggregation function, etc..

Hence the bursty traffic pattern of packets is expected in the network[91]. In such traffic, the time cost of the queued data $C_{Queue}$ is defined as follow:

$$C_{QueueBursty} = \frac{\lambda}{BW} \tag{4.12}$$

and the total time cost of bursty traffic of AIMD is a constant:

$$C_{TotalBursty} = C_{QueueBursty} + C_{idle} \tag{4.13}$$

In a more practical scenario, The arriving pattern of multiple flows in a router, like in Fig. 4.1 in Chapter 4,tend to be in a Poisson manner and independent[92].

The Queuing state of Poisson traffic is explained as follow: The arrival and the departure of data packets are in an equilibrium when the queue length is at dynamically stabilised at a certain amount.

The packet-arrival-interval of a flow subjects to an exponential distribution. The rate parameter is $\lambda$ ( this indicates the same $\lambda$ as in bursty traffic). Similarly, the leaving rate of a node at equilibrium is an exponential distribution with the rate parameter of $BW$. The state transition from $t_0$ to $t_n$ follows the birth-death process of a M/M/1 Queue model[35] as shown in Fig. 4.6:

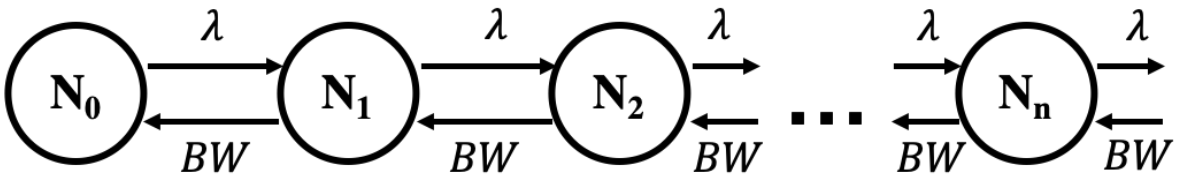



Fig. 4.6 Birth-death process of a M/M/1 Queue model

With the probability of each transition defined as $q_n, n = 0, 1, ...$, the state transition follows the equations below:

$$\begin{aligned} \lambda p_0 &= BW p_1 \\ \lambda p_0 + BW p_2 &= \lambda p_1 + BW p_1 \\ \lambda p_1 + BW p_2 3 &= \lambda p_2 + BW p_2 \\ &... \\ \lambda p_{n-1} + BW p_{n+1} &= \lambda p_n + BW p_n \\ &... \end{aligned} \tag{4.14}$$

Induct the equations above, we have the probability of state $N_n$ is :

$$p_n = (\frac{\lambda}{BW})^n p_0, n = 0, 1, 2, 3... \tag{4.15}$$

The sum of $p_n$ is 1, hence the $p_0$ and $p_n$ can be calculated as follows:

$$\begin{aligned} &\sum_{n=0}^{\infty} p_n = \frac{1}{1 - C_n} p_0 = 1 \\ \Rightarrow \quad &\begin{cases} p_0 = 1 - \rho \\ p_n = (1-\rho)\rho^n \end{cases} \end{aligned} \tag{4.16}$$

*where*,

$$\rho = \frac{\lambda}{BW}$$

the expectation of the queue length at equilibrium can further be calculated as:

$$\begin{aligned} Q = \sum_{n=0}^{\infty} n p_n &= \sum_{n=0}^{\infty} n(1-\rho)\rho^n \\ &= \frac{\rho}{(1-\rho)^2} - \frac{\rho^2}{(1-\rho)^2} \\ &= \frac{\rho}{(1-\rho)} \\ &= \frac{\lambda}{BW - \lambda} \end{aligned} \tag{4.17}$$

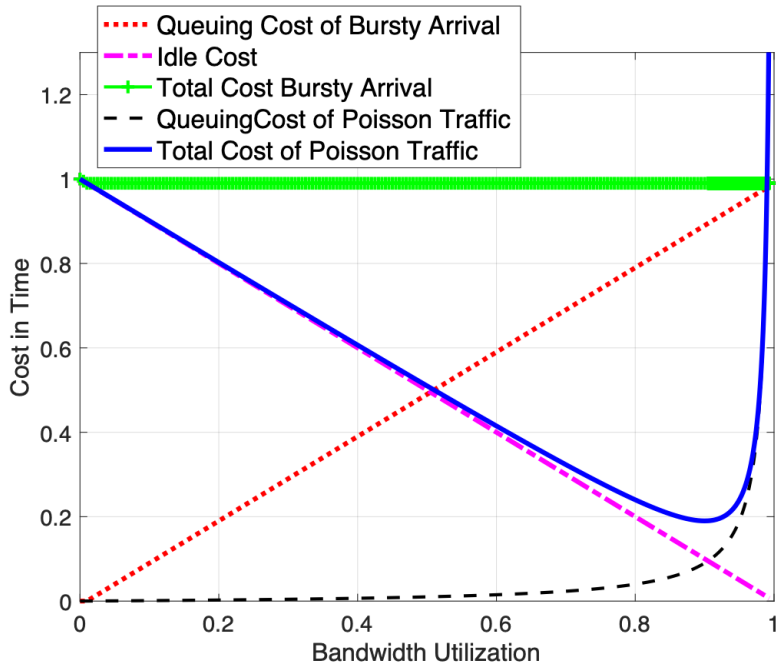



Fig. 4.7 Bandwidth utilisation against the cost in time for poisson type of traffic and bursty traffic

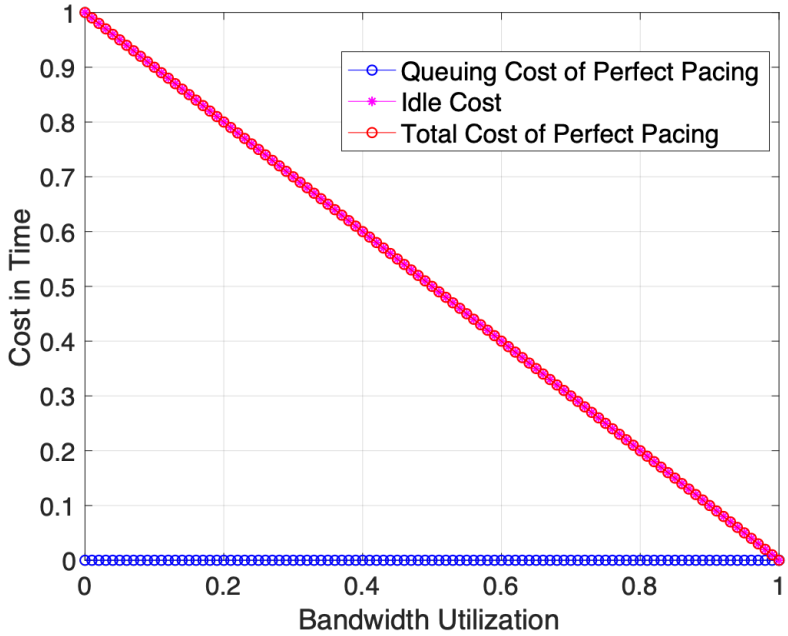



Fig. 4.8 Cost of idle, queuing and sum of the two when the pacing data rate is equivalent to bottleneck bandwidth

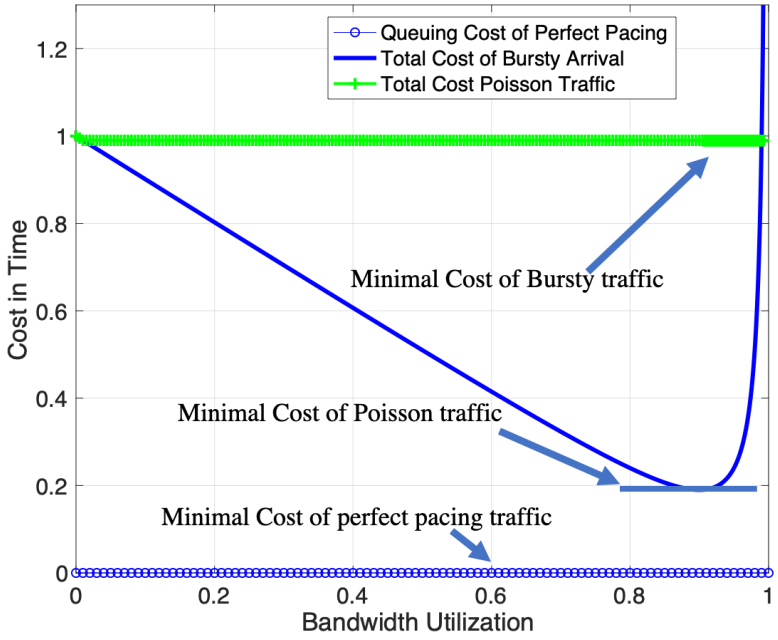



Fig. 4.9 The effect of pacing in a network.

The cost of draining the queue or the time each new packets spend in the queue is:

$$C_{QueuePoisson} = \frac{\lambda}{BW(BW-\lambda)}$$
$$C_{totalPoisson} = C_{QueuePoisson} + C_{idle} \tag{4.18}$$

The $C_{idle}, C_{QueuePoisson}, C_{QueuePoisson}, C_{QueuePoisson}$, and $C_{totalPoisson}$ are plotted in Fig.4.7. The BW used in the figure is 100 packet per unit time, which is equivalent to about 500mbps. From the Fig.4.9 we can tell that the Poisson type traffic has an optimal operation point at which the ratio of Cost and Bandwidth utilisation is minimum. However, when the rate of transmission exceeds this point, the time expense of the network increases sharply and finally even higher than the bursty traffic situation. To better utilise the network resource, a pacing solution must be used to match the $BW_{Btlnck}$ since the Queuing cost in such case is 0, and the total cost of the network totally depends on the idle cost $C_{idle}$. As a result, the perfect pacing version of the traffic gives the best cost-performance ratio among the three types of traffic as shown in Fig.4.8. This is also the goal we will try to achieve in our final CCA design.

## 4.7 Conclusion and discussion

In this chapter, we revisited the typical features we concern, e.g. Slow-Start configuration, DupACK, Pros and Cons of pacing and bursty traffic, in conventional congestion control algorithms and the philosophy behind congestion control algorithm design. An abstract of the network is also demonstrated for further analysis and the design of the CCA. The design and validation of CQIC, CQIC-S, CDBE and CDBEv2 are all based on the knowledge from this chapter.

Besides, combining the review in chapter 3, we find that all the congestion controls follow the same standard operation methodology: To probe an initial $BW_{Btlnck}$, to maintain the stability of the network, loss recovery and to prob for more available bandwidth. From a higher perspective, the design of Congestion Control Algorithm should achieve a 'One for all and also all for one' state:

- One for all: CCA on every single machine is a blind distributed system which has no clear and direct access to the outside information. The best it can do is to get as many allocated resources as possible and cause as little trouble to the network as possible.
- All for one: The whole network should achieve an equilibrium, the one goal, where all the candidates equally share the resource.

This may sounds like an ethical argument for the human being to think. However, when the problem becomes the algorithm designed for an equilibrium, the issue for us to address is caused by the choice we made. A humble participant of this game is [62]. It is so humble that the idea is to take idle resource in the network to bear the background traffic of a host and do not use the loss as a sign to pursue high bandwidth utilisation. While in the middle, Vegas[45] tried to react politely to share the network. However, the goodwill may lose the competition since other loss-based flows keep on building up the queue and force the RTT based congestion control to retreat.

Until now, our assumption, question and the tradeoffs are pretty clear:

- A mobile network is an isolated network where the shape of traffic is controlled by the gateway of the network. In the extreme case, the traffic from an external network is saturating the bottleneck, and the gateway is not the bottleneck since the processing power of a gateway is always enough.
- Problem to solve in this Thesis: A CCA in SPGW should be able to address not only the congestion in RAN but also the possible congestion in mobile backhaul.
- Tradeoffs: recall the tradeoff in last two chapters Fig.4.10:

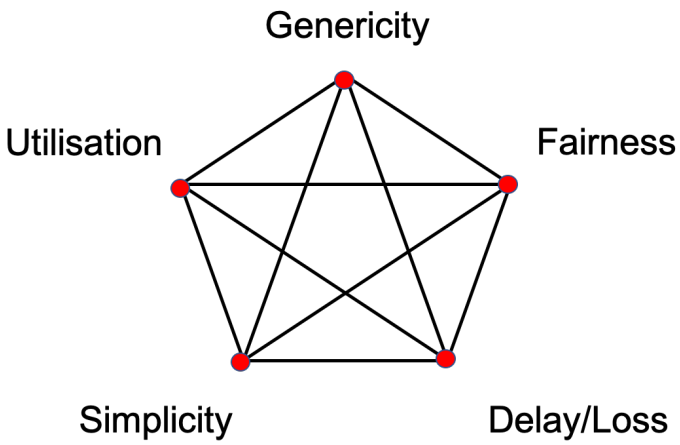



Fig. 4.10 Recalling goals and objectives

1. a fairness goal must be achievied by the distributed CCA servers at equilibrium when bottleneck is queuing up.
2. The delay caused by different operation point, as shown in Fig. 4.2, is discussed. We note that loss is the extreme expression of delay, and a certain amount of delay is the neccessary signalling for the server to understand the network.
3. last but not leastthe genericity of CCA is also discussed in this chapter: Whatever type of traffic goes into the gateway of an isolated mobile network, it will be treated as a saturated data source. The gateways should be able to manage this situation and balance the fairness not only among the the bearers in single but also the traffic among different gateways.

We discussed the tradeoff and goals of bandwidth utilisation, fairness, genericity and delay on network perspective. Combining the layered view of the tradeoff and goals set in Chapter3, we can now evaluate the existing CCA and also our own design. For simplicity of the design, the CDBE avoids the invocation of cross-layer information from an LTE chipset to the CPU for TCP/IP setup in Chapter 6. Further, CDBEv2 has a simpler client-side logic compared to CDBE in Chapter 7. In next chapter the starting point, baseline of this thesis is reviewed, implemented and validated.

# Chapter 5

# Adapt Channel Quality Indicator Control and Bottleneck Bandwidth and RTT congestion control in LTE network in NS3 simulator

## 5.1 Introduction

In this chapter, A transplant of Channel Quality Indicator Control (CQIC) in LTE from 3G/QUIC semantic to LTE/TCP with detailed congestion reaction gestures. We also review the implementation of BBR and implement it in the NS3 simulator. We compare the performance of the transplanted CQIC with two conventional CCAs Westwood and Cubic, which we expect low goodput and high RTT, although they are the most popular options in existing Linux servers. We also identify the reason for performance degradation of high data rate downloads in LTE Acknowledged Mode (AM, which is turned on by default). Furthermore, the impacts of the commonly encountered DelAck (delayed acknowledgement) feature are analysed for both congestion BBR and CQIC.

From the experience from the latter chapter, we can explain why the lower ACK rate can be beneficial to the cross-layer bandwidth report architecture. It is beneficial to reduce the BW update rate if the reported bandwidth is mismatching the real-time bandwidth experienced by the transport layer. The CQI translated theoretical BW or the resource block information calculated in DCI are different really experienced by the transport layer. The proposal in this chapter lifts the PHY layer translation (CQI to transport layer bandwidth) to the RLC/PDCP layer which takes more network factors into consideration: Queue size on Radio Access

Network(RAN), traffic status experienced on the cell or the RAN scheduler configuration, etc.. However such improvement cannot take the difference between RAN protocol stack and the traffic jitters on end-to-end transport layer into consideration.

In this chapter, we review the features of this very first cross-layer proposal. In the previous publication[93], we named our proposal TCP-CQIC-LTE. In this thesis, when we are evaluating the cross-layer solution, this old naming system is still in use since our papers use this naming system. However, in the end, we would like to change the name of this prototype proposal to DCIC. Hence in the rest of this chapter and following chapters, DCIC TCP-CQIC and CQIC are interchangeable, except as otherwise noted e.g. QUIC-CQIC, they all represent our proposal. This name is much simpler compared to the original name, and it can distinguish our proposal to the original CQIC algorithm.

For the baseline analysis, we also explore the performance of BBR in a mobile scenario. In the previous research, the loss-based congestion control algorithms give a sub-optimal performance in mobile network [26, 34, 94, 95]. *BBR,BBRStandard,BBRDeliverRateEst* is a most recent congestion based congestion control algorithms which is widely deployed in Google service/products[96]. It remarkably improves the E2E performance compared to conventional TCP variances, especially in a pure BBR data centre network. Hence, the second part of the simulation uses BBR as an up-to-date baseline for CQIC validation. First of all, we introduce our baseline algorithm BBR.

## 5.2 TCP BBR

In this research work, TCP BBR is one of the primary benchmark used. As a successor of RTT based congestion control, BBR inherits the concept of using BDP: product of BW and RTT as the last equation in equation3.2.

### 5.2.1 BBR state machine

BBR behaviour relies on four states: *Startup*, *Drain*, *ProbeBW*, *ProbeRTT* in sender. It keeps on estimating and recording the bandwidth estimations and minimum round trip time ($RTT_{min}$) in all the states upon the arrival of each ACK :

- $D_i$: the sequence of the data,
- $A_i$: the sequence of ACK corresponds to $D_i$,
- $rt_i$: reception time of $A_i$,

- $ts_i$: time when $D_i$ was sent,
- $es_i$: elapse send = $ts_i$-(send time of first data in the flight at $ts_i$),
- $ea_i$: elapse ACK of $A_i = rt_i - rt_{i-1}$,
- $sd$: amount of ACKed data during the RTT.

Then estimated bandwidth is defined as follow:

$$BW_i = \frac{sd}{max(ea_i, es_i)} \tag{5.1}$$

The calculated value $BW_i$ is updated in a window-filtered manner, where window size is 10 $RTT_{min}$. The update algorithm of $RTT_{min}$ is Bubble Sort Method [97]. According to windowed maximum bandwidth, the *BBR* sender updates the two primary attributes which control traffic emissions: CWND and pacing rate. Target CWND is calculated by multiplying max BDP with the CWND gain of the current state. Pacing rate is derived from the estimated bandwidth of the moment.

*Startup* is similar to traditional TCP slow start. *BBR* sender fills the buffer in bottleneck by an aggressive CWND gain and a pacing rate gain ($G_{cwnd}$ and $G_{pacing}$) of $2/ln2$. At this stage, the number of delivered packets increases with the growing CWND, and RTT begins to grow after the bottleneck pipe starts to build up. Hence a growth of estimated bandwidth should be observed at the very beginning of a connection. As soon as the increment of bandwidth is less than 25 percent upon three consecutive ACKs (over-send *2 BDP* in *2 RTT*), the sender enters the *Drain* state. The extra *2 BDP* of assumed buffered data is released by a $G_{pacing}(ln2/2)$ for 2 RTT, and sender proceeds to *ProbeBW*. A cycle with 8 RTT is defined in *ProbeBW*: SOUNDING (1 RTT), DRAIN (1 RTT), STEADY (6 RTT). The purpose of periodically inject an extra amount of data to the network and to probe extra potential bandwidth in the bottleneck. In SOUNDING period, extra 25% of data is allowed to be sent as a probe to sound the hidden capacity in the bottleneck. The side effect of extra data drains in next RTT. In such manner, a potential rise of available bandwidth can be captured by larger $sd$ and possibly lower $es_i$,$ea_i$ in $BW_i$ every 8 RTT. The degradation of bandwidth is caught by a maximum bandwidth sliding window timeout and will be effective when the pipe built by previous CWND will be drained. Last but not least, *BBR* sender will enter ProbeRTT for $min(RTT_{min}, 200ms)$, every 10 second. CWND in this period is only four maximum segment size (MMS). The idea of this period is to force connection back to drained pipe state to find new $RTT_{min}$ to use before next ProbeRTT session.

In summary, *BBR* trade a minor portion of time and buffer size in the bottleneck in a network for a fair optimal operation point for the majority of the time. It does not care about the random

loss or the 'hole' in the received ACK sequence. Loss is not the duty of BBR, and the only thing for BBR to do is send data at the calculated rate.

### 5.2.2 BBR capacity estimation

As observed in [98], BBR obtains better performance than conventional Cubic thanks to its smooth RTT based estimation and pacing feature. The shifting time window manner in BBR BW estimation trades off between timeliness and smoothness of the BWE. However, its reaction to bandwidth variation in mobile networks can be too slow. When the bandwidth difference is substantial, e.g. two times the adequate bandwidth, the increase may take several cycles (8 RTT per cycle) to achieve when a packet is ACKed (4 cycles in 2 times case). Furthermore, such fixed length of cycle length can make the converge to be difficult.

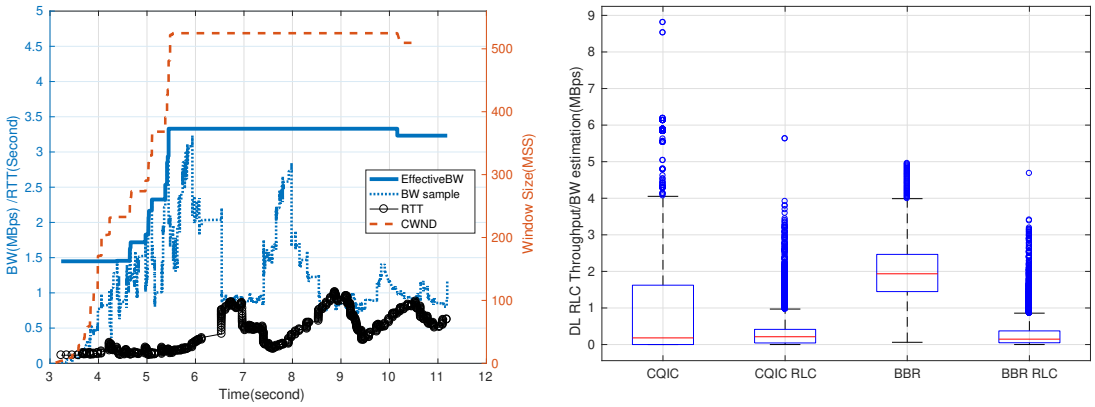
We ran a small single flow simulation using the same mobile network simulation configuraton in section 5.4.

In BBR *ProbeBW* state, the bandwidth samples are recorded in a max-filter, and the filter update round is 10. Such design will cause an overall synchronisation when The round update count is increased only when an ACK after sentinel packet is received. Due to the high delay variation in the mobile network, the maximum bandwidth update can be much slower than ten minimum round trip time. In such case, maximum bandwidth is updated when the new maximum is found, or a transmission buffer is drained. The periodical increased pacing rate may squeeze the potential bandwidth in the LTE mobile network. The observation on BBR server will be a bunch of maximum BW sample. As a result of the lag window and cyclical max BW sample, as shown in Fig.5.1a, the translated effective target *cwnd* is kept on a higher level for a relatively long period. As a result, it will keep on using overestimated active BW for some time, and the round trip time can be increased to a higher level. Another evidence is illustrated in Fig.5.1b, in BBR cases, the average BW is generally higher than RLC layer observation.

## 5.3 From CQIC in 3G to DCIC/TCP-CQIC-LTE

### 5.3.1 QUIC-CQIC in HSPA+

The original implementation of CQIC on 3G UEs are based on QUIC [99] With the assumption of the mobile link is the bottleneck of the network, the most direct way for an E2E transmitter to obtain the bottleneck capacity is to get clear feedback from the E2E receiver. To know the available lower layer capacity, a CQIC-UE should harvest the lower layer information. In[40], the necessary information in HSPA+ is CQI and the DTX ratio for QUIC-CQIC. There is a




a BBR overestimation sample

b Estimation v.s. RLC throughput

Fig. 5.1 Case study example of CQIC and BBR

direct mapping from CQI to next transport block size($tbs$), which can further be converted to the bandwidth estimation at$t_j$ by dividing the $tbs$ by 2ms interval(specified in 3G).

$$eBW_j = \frac{TbsFromCqi(CQI)}{Interval} \tag{5.2}$$

Note that the report duration of CQI in 3G network is 2ms, there will be $T_i/2$ CQI measurements in a time window $[0, T_i]$. The UE will keep on counting the downlink control information in the time window to calculate DTX ratio.

$$DTX_i = \frac{\sum_0^{T_i} \mathbb{1}(RxedDCInfo)}{T_i/2} \tag{5.3}$$

The product of average mapping results in $T_i$ and the DTX ratio is the final capacity estimation which will be sent to the data CQIC-Server every 500ms.

$$rBW_i = \frac{(\sum_1^{T_i/2} eBW_j)}{T_i/2} * DTX_i \tag{5.4}$$

where,$rBW_i$ is the reported bandwidth send to the CQIC-Server by attaching the information in a QUIC feedback(ACK/NACK). This result will be the basis of the performance for CQIC-Server before it receives the next feedback. The CQIC-Server will calculate the inter-packet transmission interval to pace every data transmission.

$$I = \frac{PktSize}{rBW_i} \tag{5.5}$$

where packet size is the QUIC packet size send to the network. The CQIC-Server will further calculate the congestion window(CWND) by calculating the BDP (bandwidth-delay product) and take the double of this size as the CWND to limit the maximum amount of data to be transmitted.

$$CWND = 2 * rBW_i * RTT_{min} \tag{5.6}$$

### 5.3.2 DCIC/TCP-CQIC in LTE

TCP-CQIC is consist of the following two parts: 1. General bandwidth estimation, according to DCI from eNB, and 2. Calculation of the initial state of TCP-CQIC by UE CQI measurement.

#### General Bandwidth estimation

Different from the 3G network, CQI is not the only variable to calculate the to the *tbs* for the next transmission time interval(TTI). Instead, *tbs* is calculated by taking both the Physical Resource Block(PRB) and CQI in the scheduler in eNB. The amount of assigned PRB is an "on-the-fly" decision made by the scheduler. It is not available to UE as a simple mapping table. Hence the original CQI-to-Bandwidth estimation in QUIC-CQIC is inapplicable in LTE. Fortunately, in most of the mobile system, the *tbs* information has been embedded in the Downlink Control Information, and specifically, such information is named Downlink Control Indicator(DCI).

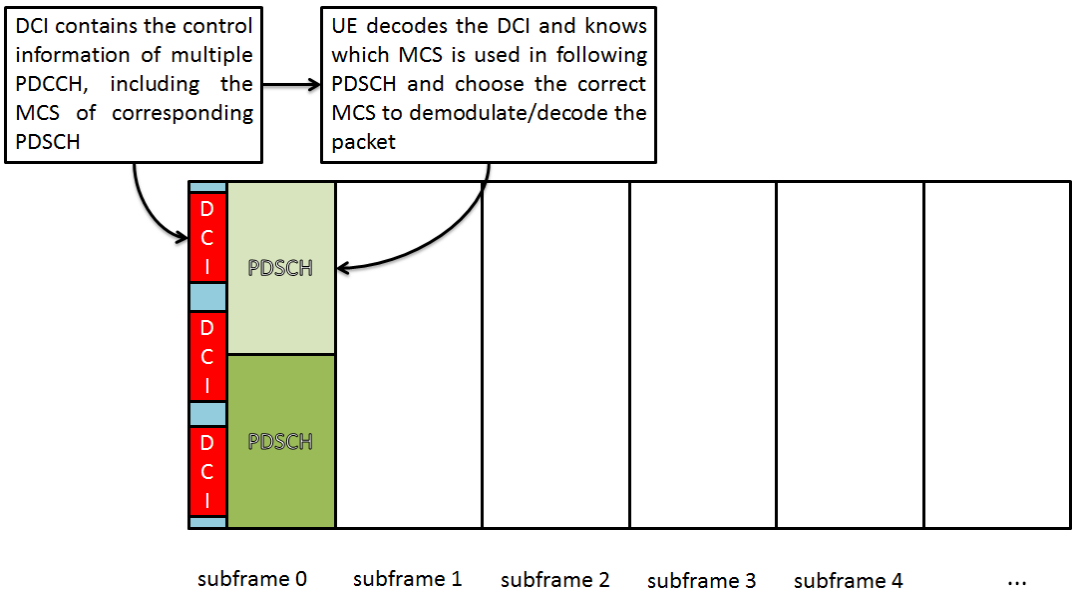



Fig. 5.2 DCI in LTE radio link

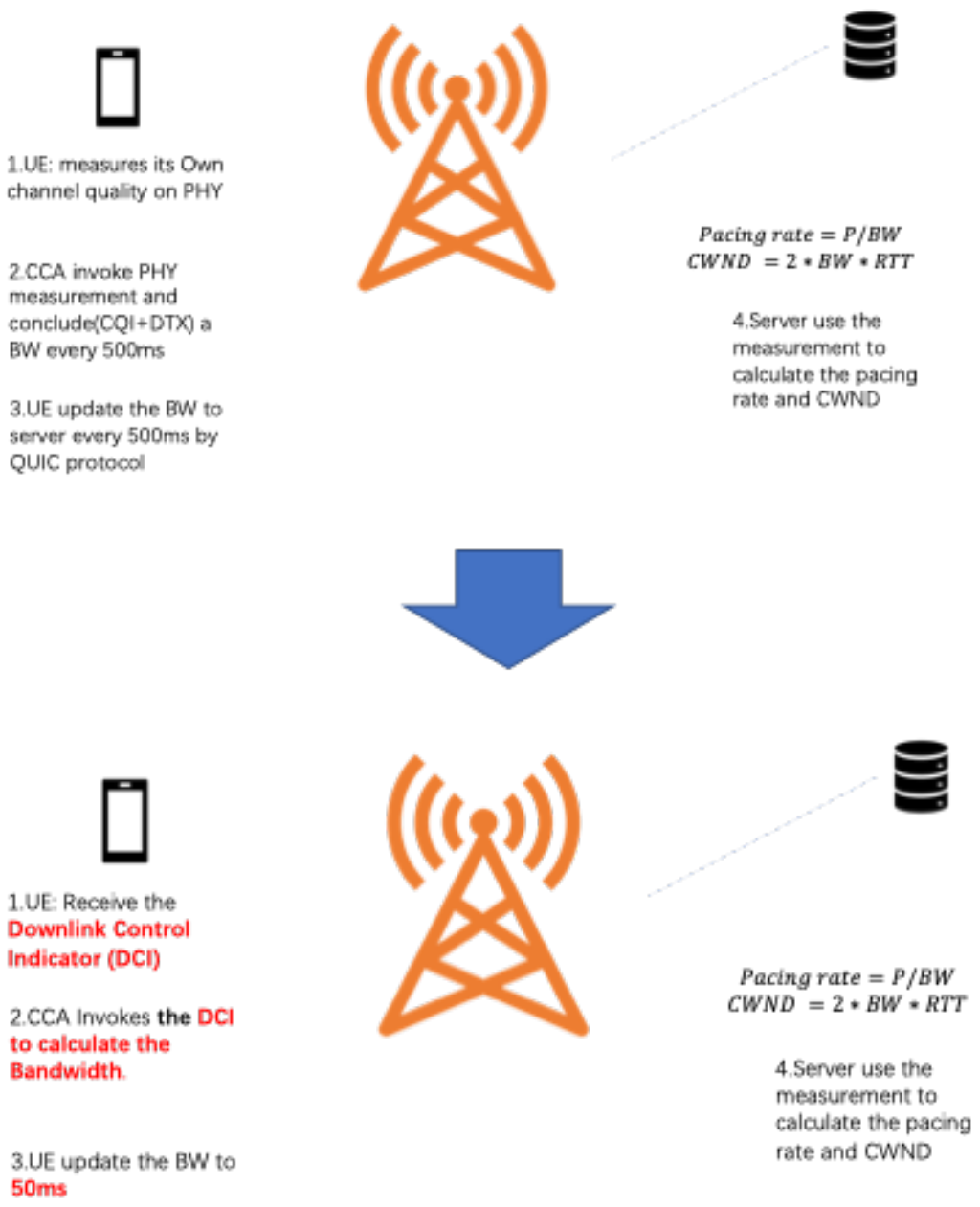



Fig. 5.3 Compare CQIC method and DCIC method

As shown in 5.2, there is data to be transmitted to the UE, eNB will schedule the resource according to the regularly updated CQI from UE and map the CQI to the corresponding MCS. The scheduler will further map MCS and scheduled PRB to *tbs*. This information will be delivered to UE by DCI. Once the UE receives this DCI, it will know which PRB to decode and in which MCS in the next TTI. The size of *tbs* in the TTI is also directly available. With

the $tbs$ size at $t_j$, the estimated bandwidth is similar to (5.2)

$$eBW_j = \frac{TBS_{DCI_j}}{Interval} \tag{5.7}$$

Note that DCI is received irregularly. The reception of DCI itself involves the DTX ratio defined in original QUIC-CQIC. Hence the reported bandwidth is :

$$rBW_i = \frac{\sum^{T_i} eBW_j}{\sum_0^{T_i} \mathbb{1}(DCI)} \tag{5.8}$$

In our implementation, the estimated bandwidth is recorded and averaged every $T_i = 50ms$ instead of 500ms proposed in QUIC-CQIC. This period can average out the considerable capacity variation in the mobile wireless network but also keep accuracy in relatively smaller time scale. A 32bit reported bandwidth will be appended at the end of the TCP header of TCP-ACK. The size of feedback can be further reduced if the level of bandwidth, rather than the exact value of capacity, is reported. On the CQIC-Server side, the inter-packet interval and the CWND with the same manner described in (5.5) and (5.6). The main difference between our proposal and the original CQIC is shown in Fig.5.3.

### Initial state of DCIC/TCP-CQIC

The problem of CQIC design is that the first bandwidth report. A TCP connection is started with the three-way handshake, and during this period, the amount of downlink data transmission is small. The prediction of bandwidth will thus be inaccurate. However, [40] is implemented on QUIC, which relies on a combined crypto and transport handshake[100]. The data transmission follows the handshake control signalling without waiting for the end of the handshake. Thus the first ACK from QUIC-CQIC can feed the sender with a relatively accurate capacity estimation if the sender is flooding the data as an initial set up. However, if sender requires any further information to set up the initial $CW$ and $Inter-Packetinterval$ before any data transmission to UE, the first report from UE will always be inaccurate since the UE is in relatively low data load.

The above-mentioned initial status is not mentioned in the original paper. Hence, We propose the following method to estimate the capacity of first bandwidth to report to TCP-CQIC-LTE sender. Since UE will estimate the CQI every 2ms in UE, the measurement can be mapped to the Approximate theoretical maximum capacity:

$$\begin{aligned} eCapacity = SpecEffi(cqi_j) nDataSymbol * nPRB \\ * nRE * nSubFrame * nLayer \end{aligned} \tag{5.9}$$

where $SpecEffi$ is the mapping from CQI to spectral efficiency in specification [3], $nDataSymbol$ is the amount of OFDM symbol for data transmission in a subframe. Its typical value is: $14-3=11$. $nPRB$ is the amount of PRB which is related to the channel bandwidth of the network and the assignment of the scheduler in eNB. We also assume 50% of the PRB is expected from scheduler if there are no recent update from eNB. In such case, the $nPRB = \frac{nPRB_{max}}{2}$ .After the initial status,$nPRB$ is the latest released value from eNB. $nRE = 12$ is the amount of Resource Element (RE) in a PRB. $nSlot$ is the amount of total subframe in a second and $nLayer$ states the spatial stream. We further assume 50% of DTX ratio, and the expectation of bandwidth can be calculated as$eBW_0 = eCapacity * 50\%$.

The server keeps on tracking this $eBW_0$ and calculates average $rBW_0$ after the data transmission from a sender is activated. In our following implementation, $PktSize$ is treated as TCP Segment size, since in our experiment, CQIC is a congestion control algorithm of TCP. Now we describe our implementation of TCP-CQIC in LTE. When $rBW_i = 0$, we can also replace the reported value of $rBW_0$ to avoid the dumb sender problem.

#### Potential Issue on PHY capacity based capacity estimation

There is the fact that the PHY bandwidth is not the actual capacity shown to transport layer. There are some retransmissions in the MAC layer and RLC layer among those scheduled PHY transmissions. These retransmissions will degrade the end-to-end throughput and fill up the queues in RLC layers, which will introduce more delay on RTT as bufferbloat phenomenon [32]. Fig.5.4 shows the difference among the PHY capacity, the RLC capacity, and TCP-CQIC estimation in some typical case. This is also the hint for latter design to calculate The actual deliver-rate from the LTE network to the UE transport layer is the average RLC capacity shown on the figure. It contains a certain amount of retransmissions. However, we should still report the PHY capacity instead of higher layer capacity back to the sender since the retransmission mechanism, and buffers in lower layers should take good care of data delivery by trade capacity with delay.

### 5.3.3 Why Delayed ACK

As mentioned above, the RLC and MAC level retransmission will introduce the extra delay on packet delivery and further prompt the RTT experienced by the transport layer. The right estimation of downlink capacity should improve the throughput, thus cause a large amount of ACK to be transmitted in the uplink. Among which, a certain amount of retransmission exists as described above. The retransmission in up-link is more detrimental than that in the downlink. Different from TTI level of scheduling result allocation, the retransmission can introduce a

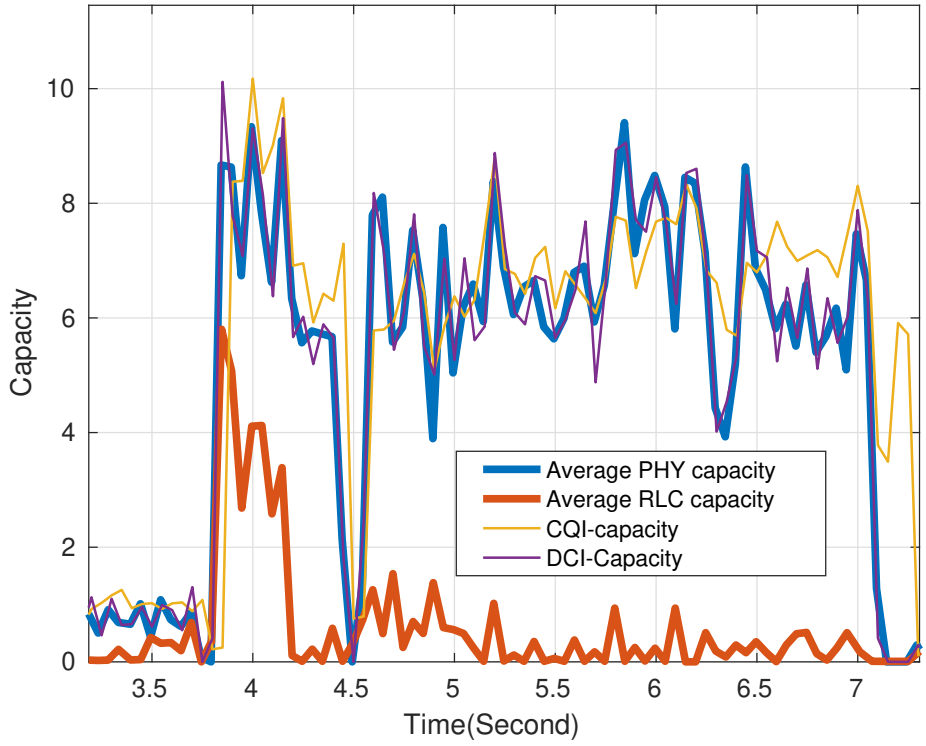



Fig. 5.4 Capacity difference among PHY, RLC and estimations

delay of at least, typically, 8ms[34]. Each re-segmentation of retransmission will cause a delay of at least 16ms if the original PDU is an incomplete segment of original data. Furthermore, since the size of ACK is relatively small, RLC layer shall assemble several PDCP SDUs for a better utilisation the available MAC transmission time slot. In the vast varying channel situation, such a large packet may increase the risk of RLC retransmission and re-segmentation for the reason that the degradation of channel quality may cause the decrease of MAC transmission slot size.

The delayed ACK timer (DelAckTimer)value in TCP allows up to 500ms of waiting time to send the ACK. The delayed ACK counter(DelAckCount) value is another constraint which regulates the performance of delayed ACK performance. In our implementation, DelAckTimer is set as Linux Cubic default (40ms), and DelAckCount is set statically as 8. This static set up works well for our experiment.

## 5.4 DCIC/TCP-CQIC implementation on NS3

The LTE simulation is available in NS3 simulator thanks to LENA project[101] . In DCIC and BBR, both algorithms require pacing feature on server side. Hence In 3.26 version of NS3 simulator, we made following changes to apply the features we need for DCIC and BBR simulation:

- For DCIC implementation a DCI report function in the PHY layer of User Equipment in LTE LENA,on receiver side. When a DCI is received from the eNB, a notify is sent from PHY to Transport layer. This function is implemented by adding a callback functor *LteUePhy::NotifyCqiMeasured* in *lte-ue-phy* module. On the Transport layer, *tcp-socket-base* module on client side, use this value to calculate the Bandwidth report(as shown in equation 5.8) to send to server side.

- An bandwidth report option field is attached on the end of each TCP ACK header as shown in Fig.5.5

- Second change is on TCP server side. We introduce the pacing transmission feature into the NS3 tcp-socket-base model. The *CqicLte::CalInterval* function is added to Congestion control functions and an interval feature/variable is added to Transmission Control Block(TCB). Every received bandwidth report on server side will trigger invoke the update of transmission interval *CqicLte::CalInterval* according to Equation 5.5.

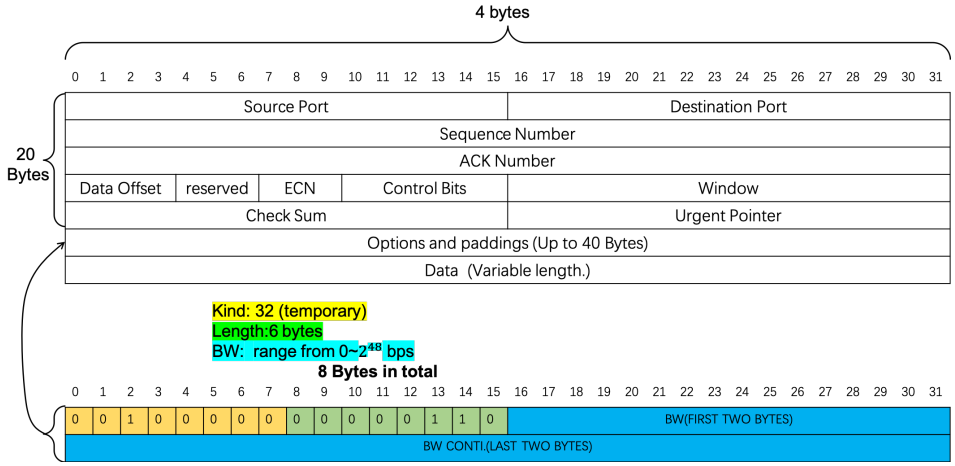



Fig. 5.5 CQIC Header report design.

With these feature implemented in NS3 simulator, the DCIC and BBR

## 5.5 DCIC/TCP-CQIC-LTE v.s. TCP Westwood and Cubic

We evaluate the performance of TCP-CQIC-LTE in ns-3 simulator (v3.26). We modified the LTE module and Internet module to implement RLC AM re-segmentation and drop tail queue feature. We have considered a typical outdoor scenario attached to one single eNB locating on the centre of a disc. UEs are moving at random walking speed and random initial position in 80 different drops of topology. The realistic channel model of path-loss and traced fading is also enabled, which creates a varying radio environment. The available experiments measures that the overall buffer size on 3G/4G is about 4MB per bearer[28, 29]. Hence we assume that RLC buffer should be lower than this value. In our simulation, the selected RLC layer buffer is 1MB per UE in both uplink and downlink. This buffer size can store roughly 10 PDUs with maximum PDU size.

The detailed simulation scheme is shown in Table. 5.1:

| | |
|---|---|
| Topology model | |
| Number of UEs | 5 |
| Number of eNBs | 1 |
| Mobility model | Random walk model |
| UE Velocity interval (m/s) | [1, 5] |
| LTE RAN | |
| Path-Loss Model | Cost231 |
| Fading model | e-EPA |
| Tx Power(dBm) | 46(eNB),24(UEs) |
| Noise Figure | 5(eNB),9(UEs) |
| Scheduler type | Proportional Fair |
| Number of Resource Block | 100(DL/UL) |
| RLC configuration | AM(Acknowledged Mode) |
| RLC buffer size | (1Mb per flow) |
| DL/UL frequency | 2120 / 1930 MHz |
| Core Network | |
| Wired Network delay (ms/hop) | 30 |
| Wired Network Capacity (GBps) | 10 |

Table 5.1 Simulation Configuration for DCIC/TCP-CQIC validation

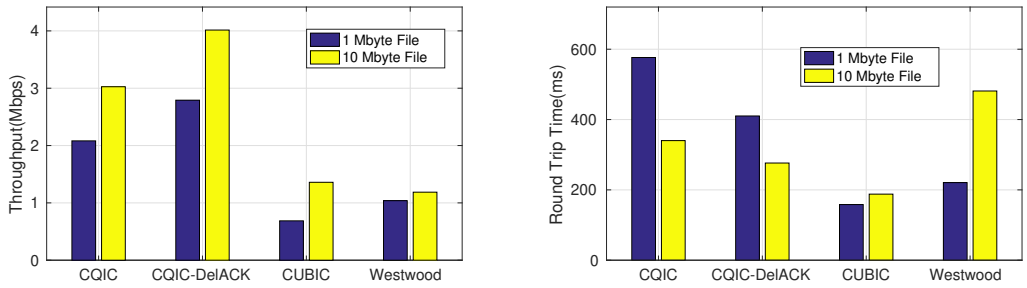
In the simulation, we have a remote server and all 5 UEs making FTP from remote server download simultaneously. The connection starts from 3rd second, and a whole simulation lasts

15 seconds. Cubic, Westwood is a benchmark used in the experiment. Basic LTE TCP-CQIC and TCP-CQIC-DelAck are tested in the simulation.

### 5.5.1 Result and discussion

Firstly we present flow level throughput. It is calculated by dividing the total amount of transmitted data(Mb) by the time interval from the transmission of the first packet to the reception of the last packet.

On average, as shown in Fig.5.6a, the throughput of Cubic is lower than that of Westwood on downloading a 1MB file from the server. The reason is Westwood probes the bandwidth by RTT-based estimation; timeout happens the CWND can be merged to an estimated *ssthresh* faster in an exponential manner. TCP-CQIC can outperform both of them by 203.1%/100.6%. When DelAck is enabled, a further improvement of 34.1%, on average. Cubic surpasses Westwood upon 10MB TCP download. The reason is the misunderstanding of RTT jitters as described in [34],



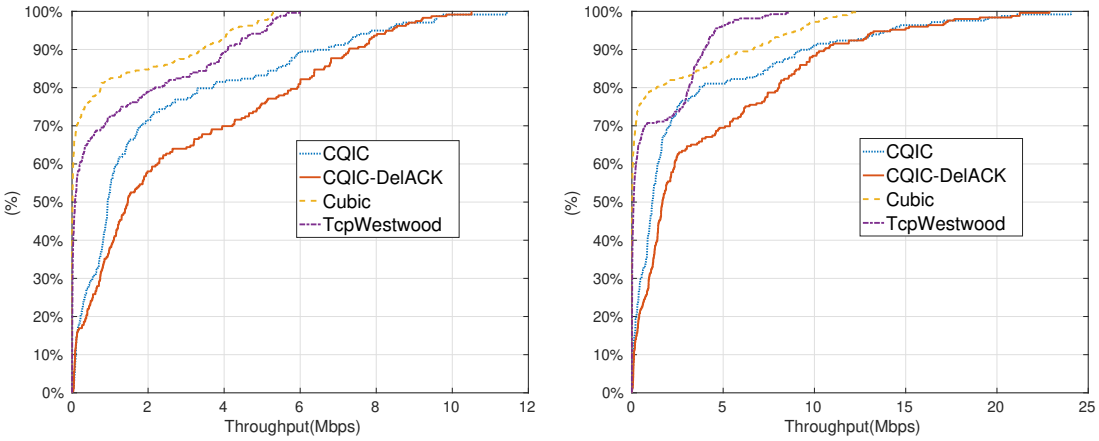

a b

Fig. 5.6 Average Throughput(a), and(b)Average Round Trip Time

As illustrated in Fig.5.6b, Round Trip Time(RTT) of the CQIC data sets are generally higher than that on Cubic and Westwood upon downloading smaller files. When a larger file is in transmission, TCP-CQIC and TCP-CQIC-DelAck have lower RTT than Westwood, but it is still higher than Cubic. Fig.5.8a and Fig.5.8b states that such a situation is always true, which is due to the facts that:

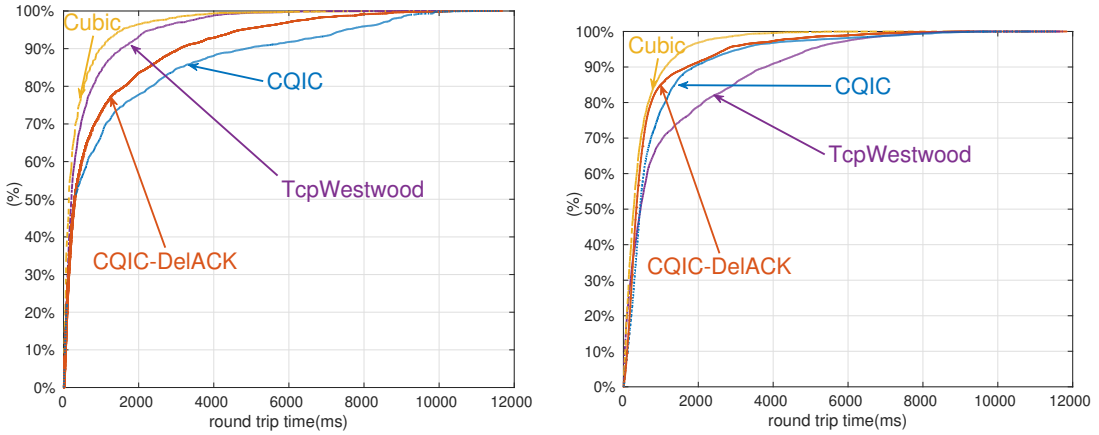
1. CQIC sender is sending a large amount of data which fits the physical capacity 'one-way delay' ago and the data reaches eNB after one extra 'one-way-delay'. The timeliness of prediction is 1-RTT behind current radio condition, and
2. PHY and MAC/RLC capacity mismatch and
3. up-link re-transmission.

Hence TCP data packets fill up the queue in eNB and cause an extra delay. The average and cumulative RTT is seen in Fig.5.8a and Fig.5.6b states that RLC UL re-transmission can be a bottleneck of the network when the amount of data transmission is large and frequent. Note that the amount of ACK in 1MB TCP-CQIC data sets is lower than that in 10MB TCP-CQIC data sets. The extra ACKs in 10MB data set, which is transmitted after 1M experienced different radio condition. Hence the higher average and cdf RTT seen on 1MB CQIC data-sets are just statistical phantoms. Further analysis of this phenomenon, according to Fig.5.10 is presented later in this section.




a b

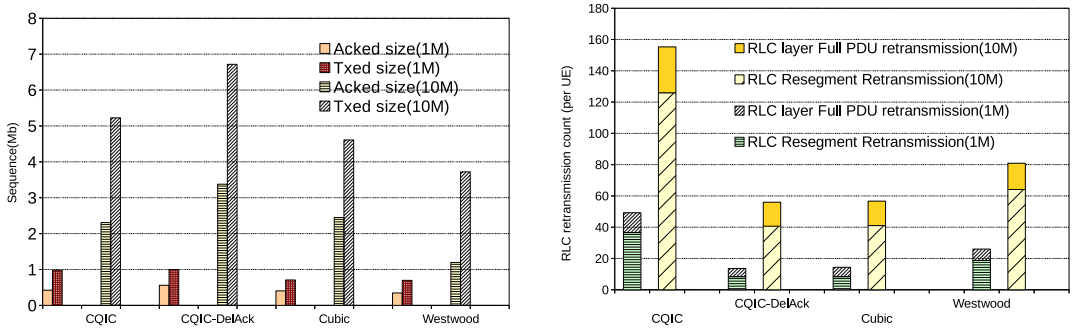
Fig. 5.7 Throughput Cumulative Distributive Function of1MB(a),10MB(b) download




a b

Fig. 5.8 RTT cdf of 1MB(a),10MB(b) download

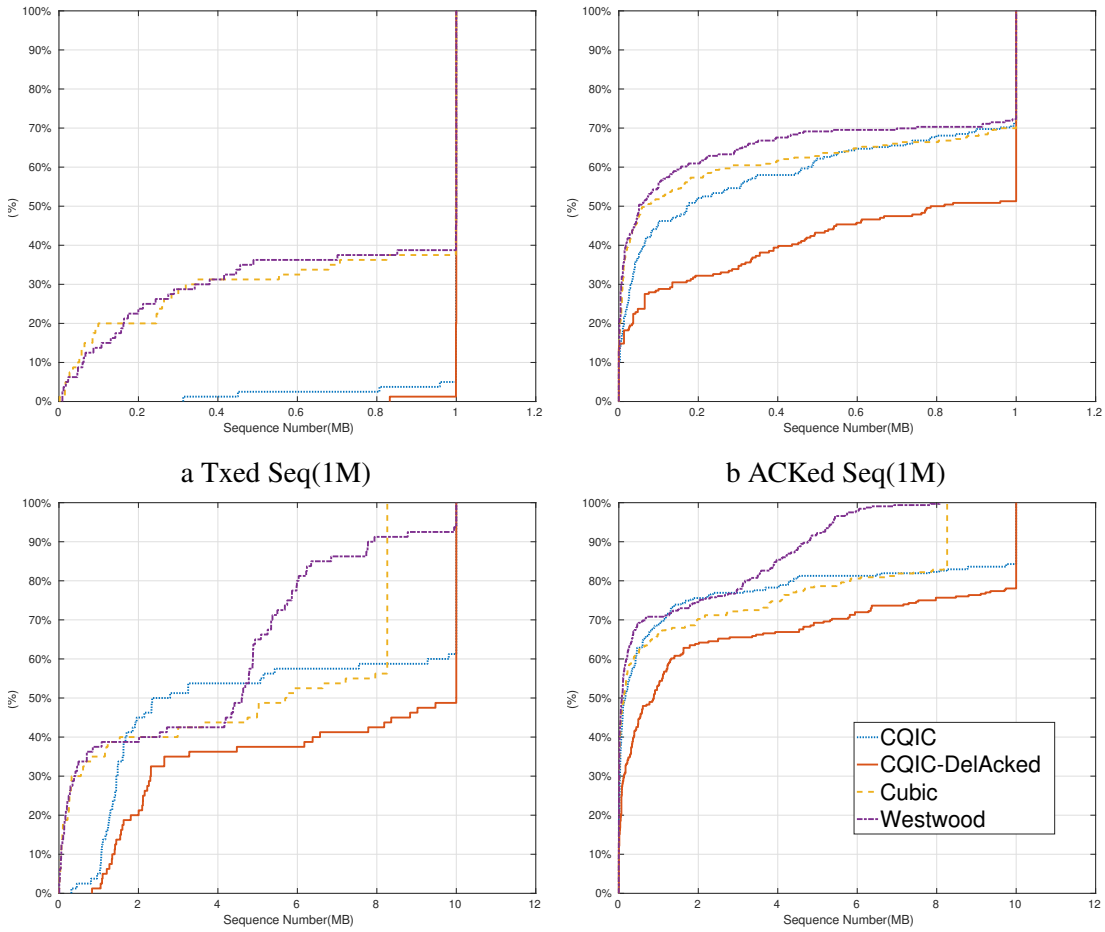
As shown in Fig.5.9a and Fig.5.10, not all the transmitted data are ACKed and not all the data are transmitted in simulation duration. For 1$MB$ data set, TCP-CQIC and TCP-CQIC-DelAck both nearly transmitted all the data on average while that in legacy TCP congestion control algorithms are limited. The ratio of ACKed Sequence and Transmitted Sequence in CQIC, CQIC-DelAck, Cubic and Westwood are 42.28%,59.62%, 54.69%, and 49.74% respectively. For 10MB data set, that proportion becomes 42.12%, 48.73%, 45%, and 33.2% respectively. The TCP-CQIC-DelAck has both maximum amounts of the transmitted data packet and ACKed packet in the simulation. Though TCP-CQIC can transmit a larger amount of total data(as throughput in Fig.5.6a) and higher amount of valid data (as bare transmission without retransmission, shown as $Txedsize$ in Fig.5.9a) than legacy TCP. The amount of ACKed data (As goodput, depicted in $ACKedSize$ in Fig.5.9a) has no obvious advantage(7.2%/9.5% in 1MB/10MB gain) compare to Cubic. While the Del-ACK enabled TCP-CQIC gives compelling growth on ACKed data(53.64%/57.8% in 1MB/10MB gain) compare to Cubic.




a b

Fig. 5.9 (a),Average RLC UL re-transmission (per UE) and (d)Average RLC UL re-transmission (per UE)

Fig 5.10a can explain the aforementioned high RTT in TCP-CQIC data sets: more than 90% of flows can fully transmit their 1MB data, and the minimum amount of transmission is more than 250KB/800KB. Only less than 65% of legacy TCP congestion control flows finished their 1 MB transmission. This means TCP-CQIC and TCP-CQIC-DelAck allow more UEs which may on the edge of the cell transmit lat least some amount of data. As shown in 5.10b, though TCP-CQIC flows to deliver a higher amount of data, only 30% of flows can finish the transmission, which is similar to traditional TCP. When delay ACK is applied with TCP-CQIC, extra data are ACKed per flow, and 50% of flows can finish the transmission. Similar situation happens in 10$MB$ data set in Fig.5.10c and Fig.5.10d. Note that Cubic cannot finish delivering data into the network. The reason for this phenomenon is that the $ssthresh$ of Cubic follows the legacy TCP AIMD manner and stays in CA mode if there is no retransmission. CQIC,




a Txed Seq(1M)

b ACKed Seq(1M)

c Txed Seq(10M)

d ACKed Seq(10M)

Fig. 5.10 The Maximum sequence of transmitted a,c and ACKed data sequenced,b

however, is not giving an outstanding performance in the 10MB data set. About 20% of flows, TCP-CQIC transmits less data than Cubic. For TCP-CQIC-DelAck, the overall transmitted data per flow is the highest among all, and more than 50% of flows pour the full 10 MB data into the network. As illustrated in Fig.5.10d, even if some flows transmit all the data, Westwood gives no finished transmission. Cubic flows have no finished transmission due to the CA limit, and TCP-CQIC has about 16% successful transmission. Once again, the TCP-CQIC-DelAck stands out. It outnumbers the rest of the group in all case in Fig.5.10d and allows about 22% of flows to finish transmission.

Moreover, as highlighted in Fig.5.9b, even if TCP-CQIC offers a large amount of data into the network. Among that transmission, the busy reception of data packet from CQIC-Server will cause a crowded TCP up-link where the amount of RLC layer is more than tripled compared to Cubic and the proportion of re-segmentation among total up-link retransmission is higher than that of Cubic.The total amount of TCP-CQIC-DelAck re-transmission reduced to

slightly lower than the amount in Cubic. As a result of crowded up-link, the benefit of extra data seen in Fig.5.9a of TCP-CQIC is negligible since they are not ACKed and re-transmission proportion is higher (6.52% re-transmission/total data transmission in TCP-CQIC against 1.67% in Cubic). TCP-CQIC-DelAck can keep low LTE-UL in a low re-transmission and re-segmentation manner. The benefit of this feature is to reduce data load in LTE-UL when the massive amount of ACK caused by the high-speed connection-oriented transport protocol. Note that Cubic and Westwood have a lower proportion of transmission, the delayed ACK may introduce several drawback[102].

## 5.6 DCIC/TCP-CQIC v.s. TCP BBR

After we compare the performance of CQIC with conventional TCP variations, now we compare the performance of CQIC and BBR. Since at this stage, CQIC has no state machine in the server side, we do not expect CQIC has better performance than BBR.

| | *BBR* | *DCIC* | *BBR DelAck* | *DCIC DelAck* |
|---|---|---|---|---|
| **End-to-end downlink(1MB and *10MB*)** | | | | |
| DL TP (MBps) | 1.10<br>*2.21* | 1.75<br>*2.59* | 1.30<br>*2.51* | 2.03<br>*3.04* |
| DL Delay (ms) | 63.68<br>*74.63* | 109.37<br>*117.64* | 67.96<br>*81.04* | 109.92<br>*130.52* |
| **End-to-end Up-Link** | | | | |
| UL Delay (Second) | 1.602<br>*1.764* | 2.422<br>*2.7065* | 1.421<br>*1.452* | 2.144<br>*2.274* |
| **End-to-end RTT** | | | | |
| Goodput (Kbps) | 160.59<br>*317.05* | 264.22<br>*360.60* | 181.87<br>*351.01* | 290.84<br>*412.24* |
| RTT(ms) | 597.51<br>*424.92* | 1219.6<br>*584.18* | 601.90<br>*392.30* | 1188.8<br>*512.26* |
| Retx (time/flow) | 0.6120<br>*0.6462* | 0.5356<br>*0.5771* | 0.6432<br>*0.6371* | 0.4809<br>*0.5703* |

Table 5.2 Compare the *DCIC* versions with BBR

The main result is shown in Table 5.2. With DelAck option enabled, both DCIC/TCP-CQIC and BBR generally can get better performance, in both goodput and RTT, compared to non-

DelAck cases. The supportive Cumulative Distributive plot is shown in Fig.5.11. DCIC has higher DL throughput and goodput. CQIC performance is better than BBR, but, similar to results in [93], the improvement is more obvious in smaller file transmission. As a consequence of more transmitted DL data shown in Fig.5.11a, 5.11d, UL traffic to acknowledge the data is also increased. As shown in Fig.5.11a, Fig.5.11d, Fig.5.11b, Fig.5.11b, in the same DelAck setup, DCIC has more finished transmission and ACKed data. The amount of retransmission per flow is also relatively lower. It is because of:

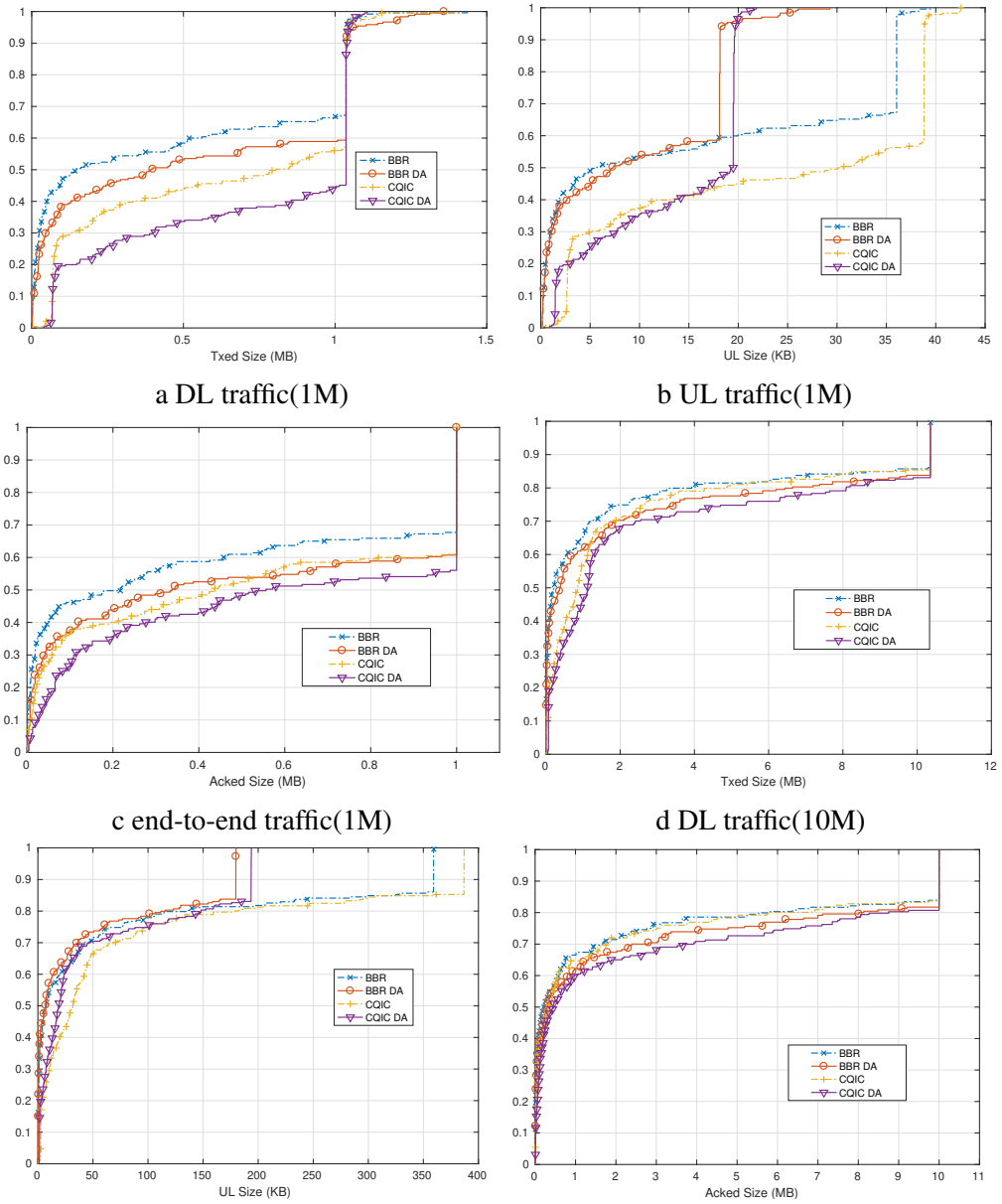
- Skips Slow start stage: Slow start manner is no more in DCIC server. Hence the time wasted in the Slow start is utilised.
- More accurate capacity estimation: Though there is a possibility of overestimation as stated in Section 5.3.2, the Capacity estimation is somewhat more accurate as statistics in Fig.5.1b indicated.

The delay of DCIC in both DL and UL are higher than that of BBR due to the timeliness lag facts analysed in Section 5.3.2. However, if the delay is averaged to each transmitted byte, the delay per byte on both DL and UL are on the same level (below ms, and 10ms level for DL and UL respectively). In fact, 32 extra bits of reported value in each ACK will cause 16.67% extra LTE up-link traffic. Such per ACK UL report manner can be further improved to reduce the amount of UL traffic. We distinguish the additional traffic caused by CQIC ACK header overhead by the CDF plot in Fig.5.11b and Fig.5.11e: for the completed transmission of 1MB and 10MB file, DCIC causes about 3KB and 30KB or extra traffic respectively. For those unfinished FTP connection, the DCIC UL traffic is also generally higher since a higher amount of data is received by UE.

Under the same DelAck condition, DCIC/TCP-CQIC has the higher amount of Acked data transmission, and from statistical view in Table 5.2, the goodput of CQIC is also higher: 64.53%/59.92% and 10.71%/14.32% improvement for an-DelAck/DelAck cases in 1MB and 10MB respectively. Note that the RTT record of CQIC is higher than that of BBR. Firstly larger RTT can be caused by the mismatch of PHY estimation, and RLC throughput due to timeliness and ARQ explained in section 5.3.2. Secondly, in TCP protocol, RTT is only recorded from the valid DATA-ACK pairs, which means the round trip delay of DUPACKs is not taken into account.

Hence, combining the individual case studies in section 5.2.2, we conclude that

1. CQIC can improve the end-to-end performance in LTE-AM network compared to latest TCP-BBR by introducing a limited amount of delay.
2. DelAck should be turned on for higher e2e performance in a mobile network, and




a DL traffic(1M)

b UL traffic(1M)

c end-to-end traffic(1M)

d DL traffic(10M)

e UL traffic(10M)

f end-to-end traffic(10M)

Fig. 5.11 DL, UL and End-to-End traffic

3. A method to reduce the UL traffic is necessary for CQIC, while a more agile target BW adaptation method should be applied to BBR for a faster reaction in the mobile network.

| | Advantages | Disadvantages |
|---|---|---|
| DCIC/ TCP-CQIC | • Fast reaction to BW variation,<br>• Takes only DS traffic into consideration.<br>• Easier and more accurate bandwidth estimation compared to CQIC | • May report over-estimated BWE to the server,<br>• Need UE hardware support,<br>• Cannot take bottleneck other than RAN last hop into consideration. |
| BBR | • Simple deployment (Server update only),<br>• Server is able to probe and simply monitor the bottleneck queue status. | • Slow reaction to rapid bandwidth variation,<br>• Lack of mechanism to cope with the bottleneck buffer,<br>• Takes both DL/UL bottleneck into consideration. |

Table 5.3 Pros and cons of using DCIC(TCP-CQIC)/BBR

## 5.7 Conclusion

In this chapter, we adapted the CQIC algorithm in [40] from the 3G network to 4G and test its performance against Cubic and Westwood on a more crowded and more dynamical radio environment than that in [40]. We also coded BBR into NS3 simulator and tested the performance of BBR. The result of first part shows that the TCP-CQIC can remarkably improve the throughput on LTE network but also introduce some delay. Further investigation shows that goodput improvement is limited which means the proportion of valid transmission in TCP-CQIC is limited. There are several reasons to cause this problem:

1. The PHY capacity is not fully equivalent to RLC capacity due to MAC/RLC retransmission

2. The timeliness of bandwidth estimation has 1-RTT delay, and

3. The crowded up-link caused by a large number of data restraints the benefit of TCP-CQIC.

We propose that the Delay ACK option should be turned on for CQIC to both improve throughput and reduce delay. The cooperation of DelAck and TCP-CQIC liberates the true potential of high-speed end-to-end transmission and mitigates the potential problems introduced by high-speed end-to-end transport protocol design. Based on the evaluation and analysis, we conclude that the CQIC can only achieve performance gain by working with appropriate Delayed ACK scheme; otherwise, the improvement on throughput is invalid(no exceptional improvement on goodput).

TCP-CQIC and BBR can both take over traffic shaping role to match the sending traffic to the $BW_{Btlnck}$. The main difference is the BWE: CQIC uses radio level report from UE while BBR probes and estimates the $BW_{Btlnck}$ through data ACKed/send rate. Furthermore, the slow start manner was abandoned by CQIC, to very quickly grasp available bandwidth even at the very beginning of the connection. The advantage of using DCIC, compared to BBR, is that the timeliness of estimation is higher since it is a real-time value instead of a 3-tiered-max time filter. Using the bandwidth calculated on the UE side can also get rid of the upstream bottleneck compared to BBR. However, the requirement of lower-layer support makes the deployment of CQIC difficult in the nearer future: In the original CQIC proposal, the implementation is achieved by connecting a mobile phone to a laptop and QXDM software is then able to capture the radio-layer traces in the diagnostic mode [40]. What is more, the lower layer translated capacity can be inaccurate due to RLC and MAC retransmission [103]. Last but not least, CQIC assumes the bottleneck locates in mobile RAN, which is not always true. How to cope with the bandwidth estimation result of CQIC with bottlenecks other than the radio link can be a critical issue. The pros and cons of using BBR and DCIC are summarised in Table. 5.3.

In the second half of this chapter, the performance of DCIC(TCP-CQIC) and TCP BBR are compared in both with or without Delayed ACK cases. The results show that TCP-CQIC can remarkably improve overall performance on LTE network while introducing some little amount of delay. However, CQIC has no state machine on the server-side. In the next chapter, we decide to apply a state transition logic we explored in previous chapters to CQIC for better performance. Furthermore, we assume that the high delay is caused by the mismatching of bandwidth report from PHY layer and Transport layer, and at the same time, the RAN is the bottleneck of the network. To also simplify the design and also avoid the mismatch of throughput observation from a different layer, a client-side bandwidth estimation method is proposed in the next chapter from the transport layer perspective.

# Chapter 6

# Toward client driven bandwidth estimation architecture for end-to-end congestion control: a protocol design

## 6.1 Introduction

Based on the discussions and analysis on BBR and DCIC/TCP-CQIC, we turn back rethink about the general design principle of congestion control mentioned in previous chapters: To probe more, to share bandwidth, but not to fully occupy buffers. In conventional TCP, the typical congestion control signal to discipline the end-to-end (E2E) traffic loss. As a extreme expression of delay, when loss is detected by out the duplicated ACK feedback,the congestion window (CWND) and slow start threshold (*ssthresh*) are the two main features for a server to govern the TCP transmission pattern. A congestion control algorithm (CCA) updates the CWND and *ssthresh* by a reaction to each received Acknowledgement (ACK) in SS and CA or by a Re-transmission Timeout (RTO). Most of the frequently used TCP is loss-based and top up the queue to cause bufferbloat. Some existing TCP CCA, e.g. Westwood [60], Vegas [45], and most recent BBR [96], may roughly probe the network capacity and use it as the aid of congestion control. They are tested in the previous chapter and cannot outperform the CQIC algorithm which is the starting point of our CCA design. Note that BBR is the first widely deployed congestion-based CCA without a strict additive increase multiplicative decrease (AIMD) behaviour. As a consequence, BBR gives TCP a better capacity to react to BW fluctuations than typical AIMD CCA [39]. In previous chapter, BBR has similar performance as CQIC. But still its performance is not optimum in Cellular network.Further, the evolution and coexistence of generations of cellular networks raises a new challenge for

BWE based CCA design. On the one hand, an appropriate choice of scheduling method and modulation and coding scheme (MCS) can reduce the possibility of radio transmission failure. This strategy achieves higher radio resource utilisation from media access (MAC) perspective of view. However, from per UE perspective of view, they should experience a fluctuation of instantaneous radio DL BW according to eNB scheduling result [2]. On the other hand, the lossy nature of the radio channel is handled by ARQ/HARQ in Radio Link Control (RLC) and MAC layers in the LTE network. Multiple levels of retransmission mechanism can introduce a certain amount of delay according to RAN configuration [2, 104]. Hence from the E2E point of view, a delay variation instead of the loss is observed. As a result, the loss detection based CCA may cause bufferbloat [95] and can not perform well in such scenario [26, 34, 94, 105]. Furthermore, the LTE uplink and downlink are asymmetric. The uplink uses the robust transmission strategy [106] and power-saving configuration [33], which can be a constraint of the data communication.

In this chapter, a TCP client-driven BWE (CDBE) method for the fast BW variation is proposed to address the problems. It is design and validated in NS3 simulator under the TCP skeleton. So we still call it TCP CDBE, but essentially it is an CCA. Thereby it can be applied to any type of network architecture for a role of end-to-end traffic control.

In this CCA architecture, the client and server of the TCP connection can cooperate. A CDBE client can engage in end-to-end CCA decision loop by offering BWE feedback. The benefit of doing so is that the non-application limited DS traffic can be delivered without considering the upstream bottleneck. When the last hop RAN is the bottleneck of the network, a fairly accurate last-hop BWE is achieved by dual-window BWE method. While the bottleneck is elsewhere in DS, the BWE method can also reflect the core network bottleneck BW. The measurement result is further piggybacked to the server by introducing a new "BW" optional field on TCP header in our experiment. The server then exploits the estimation result to calculate traffic shaping elements, CWND and pacing interval (*PI*). Different gains are applied to computed elements, according to the DS delay, to avoid bufferbloat or to push more data into the network if there are available BW in the bottleneck.

## 6.2 The pros and cons of tested CCAs

From Chapter3, 4 and 5 we have the conclusion of CCAs in Table 6.1. The main existing problem of CQIC is fully relying on the Client-side Radio Link(RL) capacity report and does not take the traffic condition, scheduling result or the bottleneck, etc. somewhere elsewhere than eNB into consideration. Hence we decide to follow the thought of the general CCA logic in Chapter.4 to improve the logic of CQIC. Further, The translation from RL bandwidth can

| | *Westwood/Reno/Cubic* | *DCIC/CQIC* | *BBR* |
|---|---|---|---|
| Type of CCA | Event based | feedback based | feedback based |
| | | **How to** | |
| Find current $BW_{Btlnck}$ | slow-start and loss-based event-driven method; (the maximum queue length in network is found.) | Base on the client side feedback. | Startup described in Chapter5 |
| Avoid congestion | Time-out loss event, fast-retransmission mechanism. | Base on the client-side feedback. | Regular house-keeping oriented low gain in both Startup and ProbBW stage |
| probe more BW | reserved congestion-avoidance method which allows minimum units of CWND growth | Totally base on the client-side feedback. | regular growth-oriented gain in ProbBW stage |

Table 6.1 Compare the technical details in the baselines and DCIC

be misleading for a hybrid network, e.g. 2G, 3G, 4G, 5G or even different version of Wi-Fi. Last but not least, the usage of lower layer processing power can be power inefficient, and the mobile throughput guidance [70] is more energy efficient for the portable UEs compared to this method. Concerning these challenges, CDBE is proposed. In this cooperative CCA architecture, a CDBE client can engage offers a BWE feedback to the server. When the pipeline is on the optimum point as described in [107], the feedback is the BW from the bottleneck. The benefit of doing so is that the non-application limited DS traffic can be delivered without considering the upstream bottleneck. When the last hop RAN is the bottleneck of the network, a reasonably accurate last-hop BWE is achieved by dual-window BWE method. While the bottleneck is elsewhere in DS, the BWE method can also reflect the core network bottleneck BW. The measurement result is further piggybacked to the server by introducing a new "BW" optional field on TCP header in our experiment. The server then exploits the estimation result to calculate traffic shaping elements, CWND and pacing interval (*PI*). Different gains are applied to computed elements, according to the DS delay, to avoid bufferbloat or to push more data into the network if there are available BW in the bottleneck.

As for other up to date researches we have investigates Verus, TCP-CDG and GCC as a plus: Verus [108] uses the Round Trip Time (RTT) gradient to infer the congestion degree in the network and evolve the current sending window size from the previous window size.

The TCP CDG[109] also uses the RTT gradient as a hint of loss in the Loss-Based CCA semantic. A shadow CWND will be maintained during the LOSS-detected period, and the value will replace the dropped CWND to maintain the utilisation of pipeline for higher throughput. This may still not be efficient to mitigate the buffer-bloat issue.

The google congestion control (GCC) [110] is the main congestion control for WebRTC. It evolves the BW report on the client-side by a factor. The server uses the report to update the transmission pace for the current video frame quality. All these foregoers inspire the design of this original prototype of CDBE.

## 6.3 Bandwidth estimation method in UE

Different from the existing server-side BWE methods listed in [**?** ], the algorithm runs on the UE side, and it is a continuous measurement. The CDBE BWE aims to follow the BW variation of the mobile network in ms level. Hence in this work, all the TCP clients run in UEs, and the following UEs also represent CDBE clients.

### 6.3.1 Principle of design

The arriving rate can be simply abstracted as follows:

$$R(t) = \frac{\sum Pkts}{Duration} \tag{6.1}$$

The duration can be calculated from the system timer or timestamp value. Use a hard-coded constant duration to calculate arriving rate can cause the following issues: 1. involve an unnecessary volume of time to cause underestimation, since we do not know how much spare time is involved in the head and tail of the window, or, 2. over-estimate due to the bursty arrival caused by PDCP in order delivery mechanism. When this constant value is set too large, the underestimation in the former point is more significant, and when the window is too small, the latter overestimation is more obvious. Hence, the choice of the constant window as duration may cause inaccurate rate calculation on the client side like that in GCC, CQIC, piStream, etc.

The principle of the design is to filter out the unnecessary time gap in the BW calculation but not to eliminate the gap caused by the delay. We propose a two-window based BWE method to address the issue in this first prototype version of CDBE: The CDBE client is responsible for measuring the BW samples ($sBW$) of each sample windows ($SW$) and use their average to

update the report ($rBW$) in every long window($LW$) epoch. The $SW$ captures the total amount of received data segments to calculate $sBW$, whereas the latter epoch considers all the included $sBW$ to calculate $rBW$. Once the $rBW$ is updated, the value is converted to 32 bit BW option in ACK. The time limit of $SW$ is $W_s$, and for $LW$ it is denoted as $W_l$. The size of $W_s$ and $W_l$ can affect the accuracy of estimation. The $W_l$ is five times $W_s$ period in the current configuration. This allows no more than 5 $sBW$ in each $LW$. A $LW$ begins with the head of its first $SW$ and ends with the termination of its last $SW$. To guarantee the timeliness of the $rBW$, the $W_l$ is set to the same as RTT since it takes at least one RTT to reflect the current $rBW$ back to the client.

The implementation of CDBE client requires to turn on the timestamp option for a better RTT accuracy. The redesign of TCP TS field may be required [111]: More than five different granularity exists in a current TCP implementation. In the evaluation of this work, the TS granularity is synchronised to 1ms.

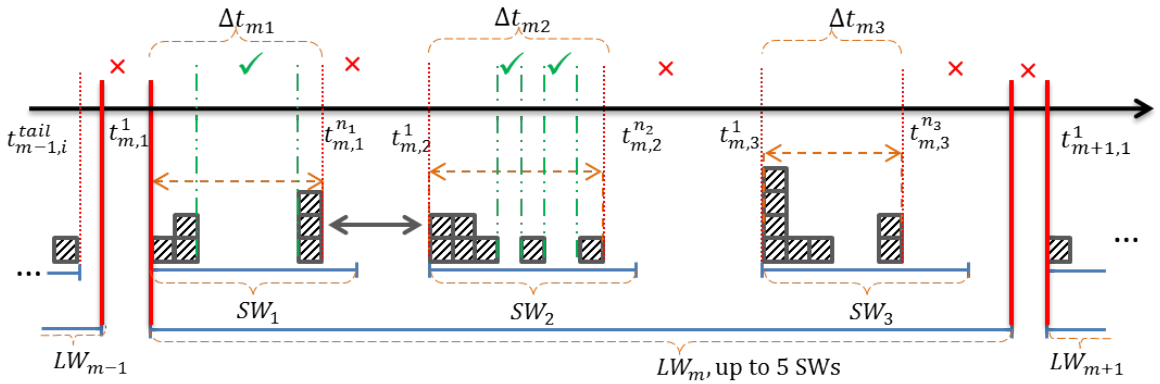



Fig. 6.1 Illustration of CDBE filter function on UE

### Detail of BWE method

Fig. 6.1 is used to aid the understanding of the proposed BWE method. A cube in Fig. 6.1 represents an arriving TCP data segment. Each $LW$ epoch begins with the first arriving TCP data segment of its first sample window $SW_1$ at $t_{11}$. Upon the arrival of each data segment, a CDBE client calculates the time difference ($\Delta t$) between now and the head of the current $SW_i$. Once the received data segment falls outside of $SW_i$, it becomes the head of the next $SW_{i+1}$. A BW sample $sBW_i$ is calculated using the equation below:

$$sBW_i = \frac{\sum_1^n P_n}{time_i} \tag{6.2}$$

where $n$ is the total number of segments received in $SW_i$, and $P_n$ is the size of the $n^{th}$ packet in $SW_i$. $time_i$ is selected from one of the three possible value: minimal duration ($\Delta t_{min}$), $W_s$ and

explicit time difference between the first and the last segment in $SW$ ($\Delta t_i = t_{in} - t_{i1}$):

$$time_i = \begin{cases} W_s & \text{if } \Delta TS_i >= 2 * max(\Delta t_i, \Delta t_{min}) \\ \Delta t_{min} & \text{if } \Delta t_i == 0 \\ \Delta t_i & Otherwise \end{cases} \tag{6.3}$$

TCP timestamp is used to support the decision making. When the timestamp difference ($\Delta TS_i = TS_{in} - TS_{i1}$) of corresponding data segments is larger than $\Delta t_i$, it means there must be some level of delay in the network. Some part of the delay may comes from the HARQ/ARQ mechanism in LTE which guarantees the in order delivery of its data packet according to our observation: Once a burst of segment reception happens, there is a possibility that the delivery of the first $(j-1)^{th}$ segments is obstructed by $j^{th}$ segment due to several rounds of unsuccessful retransmission of $(j-1)^{th}$. This phenomenon is irrelevant to the BW reduction, and it can be averaged out by taking mean of samples in a $LW$. We also propose that when $\Delta TS_i$ is more than twice of $max(\Delta t_i, \Delta t_{min})$, we should use $W_s$ instead of $\Delta t$ to avoid potential over-estimation in current $SW_i$. This approach tackles most of the over-shooting problem and performs well in our simulation. However, in some extreme cases, if the RLC/PDCP caused bursty reception of data is severe, the $\Delta t_i$ can be very small. As a result, a large number of packets can arrive with $\Delta t = 0$. A minimum time duration $t_{min}$ is introduced to make sure the valid calculation in equation 6.2. In our current implementation, $t_{min}$ is a constant value of 2ms.

Once the current segment falls outside the current $LW$ epoch, the UE will calculate the average of the BW samples in $LW$. Further, the valid latest $rBW$ is calculated as the smooth averaged with the prior report value ($rBW_{m-1}$):

$$rBW_m = \beta * rBW_{m-1} + (1-\beta) * \overline{rBW}_m \tag{6.4}$$

where,

$$\beta = \begin{cases} \dfrac{0.4}{0.4 + (t_m^{first} - t_{m-1}^{last})} & \text{if } t_m^{first} - t_{m-1}^{last} > W_S \\ 0.4 & Otherwise \end{cases} \tag{6.5}$$

This low pass filter manner reports a smoother $rBW$ and mitigates the sharp variation of CWND and $PI$ when RTT is small. If $rBW_{m-1}$ record is too old, it should not affect the current result too much, hence if the time difference of two $LW$ is more substantial than $W_s$, $\beta$ is formulated as an inverse proportional to the time gap between the two $LWs$. Last but not least, under the current configuration, the initial BW report is 1 MBps.

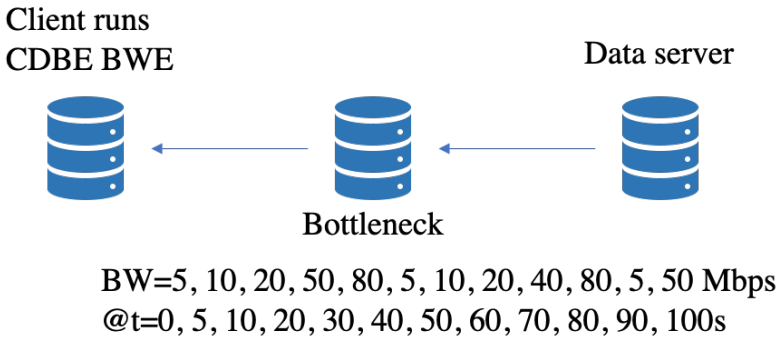



Fig. 6.2 Fixed network topology for CDBE BWE validation

### 6.3.2 Validate CDBE BWE method with saturated traffic

To validate the proposed method, we first test the CDBE client BWE algorithm in a fixed network. We simulated a two-hop wired network with one varying bottleneck and minimum round trip delay of 80ms. The bottleneck BW changes in the following sequence: 5, 10, 20, 50, 80, 5, 10, 20, 40, 80, 5, 50 Mbps, at 5, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100 second respectively. The second hop has 10Gbps BW, which guarantees the bottleneck limits the traffic. The topology is demonstrated in Fig.6.2. The host nodes make UDP download to saturate the bottleneck, which guarantees the BWE in the client can reflect the maximum BW in the bottleneck. The result is shown in Fig. 6.3: regardless of lower-layer overhead, the BWE tracks the discrete downstream BW variation well.

We further simulate the CDBE BWE in five mobile UEs under UDP-download scenario for 30 seconds. Five co-cell UEs locate on 50, 100, 120, 150, 180 metres away from eNB. The download starts from $5^{th}$ second and the UEs move away from eNB for first 10 seconds at 6m/s. After that, the UEs then move toward eNB at the velocity of 10 m/s for the next 10 seconds. In the last five second of simulation, the UEs leave eNB again, but their speed is 20m/s. As illustrated in Fig. 6.4, the CDBE algorithm in UEs can keep track of continuous BW variation. Hence we confirm that the proposed algorithm is capable of following the track of saturated traffic.

## 6.4 CDBE state transition module in server

In non-application limited mode, the CDBE server uses received $rBW$ to calculate the CWND and $PI$:

$$PI = \frac{pktSize}{G_{pacing} * rBW} \tag{6.6}$$

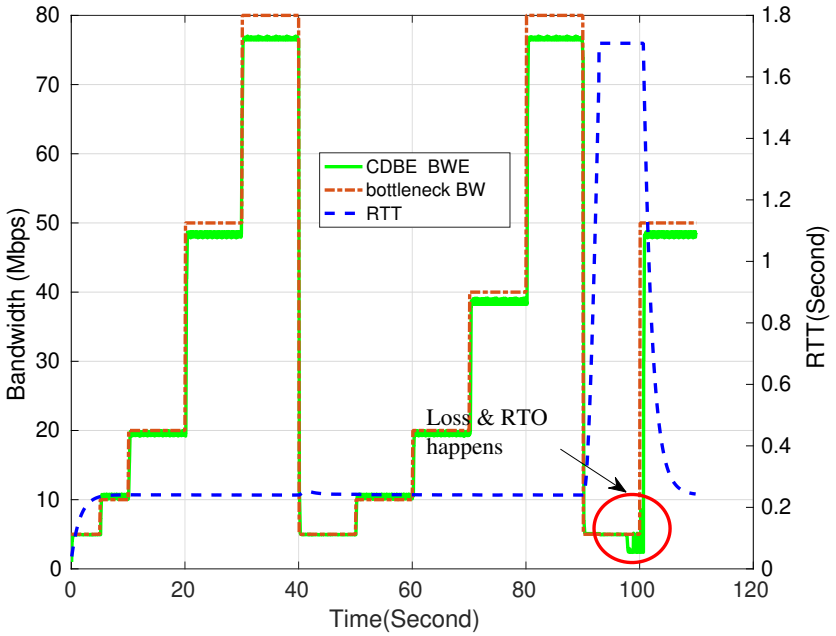



Fig. 6.3 Validate CDBE BWE method in a wired network with UDP traffic

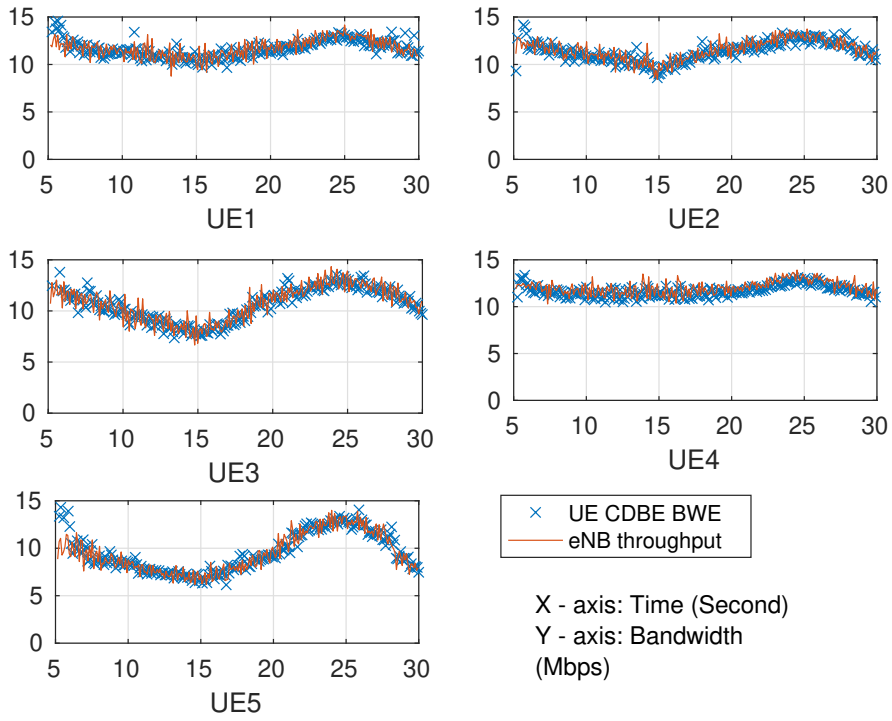



Fig. 6.4 Validate CDBE BWE method in LTE with UDP traffic

$$cwnd = G_{cwnd} * rBW * RTT_{min} \tag{6.7}$$

Pacing interval is the main character to shape the traffic, while CWND is the minor congestion control factor to guarantee that the amount of traffic does not exceed scaled BW delay product according to current state. According to the analysis in Section 6.3.2, it is necessary for the server to send enough backpressure for the bottleneck to reach the optimum working point for the BWE method to track the BW variation accurately. In the meantime, the queue should also be maintained on a relatively low level. The server uses the continuous probing method to feed the bottleneck with enough data while also take care of the queue in the bottleneck buffer. We will discuss transition logic and the definition of the states in the following two subsections.

### 6.4.1 State definition and impact of parameters

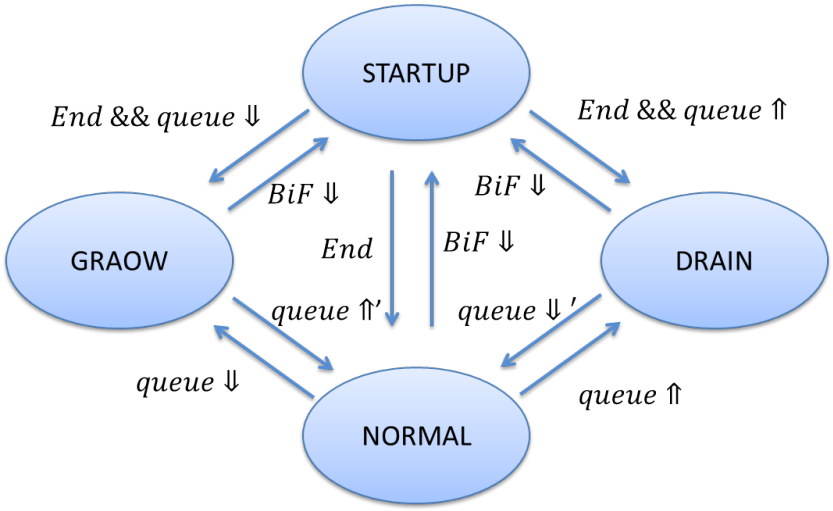



Fig. 6.5 CDBE server state transition

The CDBE server considers the downstream delay (DSDL) variation as the main indicator of queue state in eNB. Hence in our current implementation, the factor to optimise is DSDL, which is directly available from TCP TS option. The DSDL growth caused by the server's own traffic relates to an eNB DL queue build up while DSDL drop corresponds to a draining eNB DL buffer. The server should *increase*, *decrease* and *maintain* the calculated CWND and *PI* due to the queue size variation. In such case, the states defined for CDBE are: *STARTUP*, *NORMAL*, *DRAIN*, *GROW* and their transition is illustrated in Fig. 6.5. To control the amount of traffic, pairs $G_{pacing}$ and $G_{cwnd}$ are designed for each states. These value of the pairs are inherited from BBR: $2/ln2, 1, 0.75, 1.25$ are the $G_{pacing}$ for *STARTUP*, *NORMAL*, *DRAIN*, *GROW* respectively and the $G_{cwnd}$ for corresponding states are $2/ln2, 2, 2, 2$.

Sender travels among the states in a time-windowed ($TW$) manner ($W_{update} = RTT_{min}$) based on the sign of DSDL variation. The detail of each state is given below:

■ ***STARTUP***: This is the first stage a sender/server enters. The server assumes that there is no pipe in the eNB or not enough data in the flight for the current connection. Hence CDBE sender uses more aggressive $G_{pacing}$ and $G_{cwnd}$ to build up the pipe in the eNB as fast as possible. This approach squeezes more and more of the available BW for $m$ round trip time. Once the startup-count is larger than $m$ or the growth of received BW report is less than 10% for two continuous minimum RTT. Similar to BBR startup, this is a dynamical variant of slow-start and keeps similar TCP SS performance for short flow. In *STARTUP*, the sender has begun keeping on monitoring the DSDL and BWE for an earlier preparation for state transition among the other state.

■ ***GROW***: The main purpose of the *GROW* state is to probe the available BW in a BW varying network. For example, in LTE cellular, if the radio and co-cell traffic condition is good enough, we should observe no queue up when we use the original *rBW* from UE to shape traffic. After $n$ period of stable DSDL, the server will enter *GROW* to send at a gain of 1.25. The UE will observe the higher amount of received data segments if there are some available spare BW, and the corresponding *rBW* will increase exponentially. Before the sign of $queue\Uparrow'$ is met, server will stay in *GROW*. This manner allows for faster progress of BW increment compared to BBR. The conditions marked in Fig. 6.5 will be explained in next subsection.

■ ***DRAIN***: *DRAIN* reduces the bottleneck queue. $queue\Uparrow$ corresponds to the inflating buffer caused by the server's own traffic. Once the condition of DRAIN is met, the server will try to ease the potential bufferbloat caused. Under the current configuration, the server will keep draining the queue until $queue\Downarrow'$ is shown. The server will use *NORMAL* for one $TW$ to wait and see the dropping trend of DSDL. If DSDL does not fall back to $DSDL_{min}$, a new $DSDL_{min}$ may be updated. This approach stables the DSDL at a reasonable level, and the server will go to *NORMAL* to maintain the amount of data.

■ ***NORMAL***: *NORMAL* is used to avoid the oscillation between *GROW* and *DRAIN*. It allows the server to observe how the network reacts to its gain pairs variation. In *NORMAL* state, the server will observe the DSDL for a couple of $TW$. As long as the trend of DSDL in last $TW$ does not vary continuously on the same trend, the sender assumes the buffer in the bottleneck buffer is about stable, and sender stays in *NORMAL*.

### 6.4.2 Condition of state transition in CDBE server

The server maintains the record of DSDL and the received *rBW* to appraise the result of the queueing inspection. The DSDL trend, the change of *rBW*, and current state decide the next state. The DSDL can be obtained by calculating the difference between TSecr and TSval of TCP timestamp option for each received segment. Due to the bursty scheduling manner in the LTE network, the packet arriving delay may vary widely [108]. To observe the steady trend of DSDL variation in the last RTT, the sender should check the average of the DSDL sample recorded in each recorded each $TW$ :

$$A_{DSDL_i} = \sum_{j=1}^{T_i} \frac{DSDL_j}{T_i} \tag{6.8}$$

The average distance of DSDL samples from minimum $DSDL_{min}$ is also considered to eliminate the DSDL variation:

$$\Delta DSDL_i = \sum_{j=1}^{T_i} (DSDL_j - A_{DSDL_{i-1}}) \tag{6.9}$$

Further, minimum DSDL ($DSDL_{min}$) and *BDP* are also use as a clue. The sender decides its state in the next sample period according to these evidence shown below:

□ **Obvious Queue growth (*queue*↑)**:

$A_{DSDL_i} > (1+\alpha) * A_{DSDL_{i-1}}$ and $\Delta DSDL_i > 0$,
or $A_{DSDL_i} > (1+\beta) * DSDL_{min}$

When the DSDL shows the trend of growth, it means the queue is possibly building up. When the average DSDL is larger than $(1+\alpha) * DSDL_{min}$, it will also be considered as one potential hint for *DRAIN*. Concerning the queue up sign may be the result of the traffic from other traffic sources, $\alpha$ is set to 0.1 to tolerate the queuing pressure from other traffic sources. It not only guilds the queue length built by the server's own traffic but also prevents the server from draining the queue frequently as Vegas when facing cross traffic.

□ **Obvious Queue decrease(*queue*↓)**:

$A_{DSDL_i} < (1-\alpha) * A_{DSDL_{i-1}}$ and $\Delta DSDL_i < 0$,
or $A_{DSDL_i} < (1+\alpha) * DSDL_{min}$

When the DSDL shows the trend of decrease, it means the queue is possibly draining, or the route has changed. When the average DSDL is lower than $(1+\alpha) * DSDL_{min}$, it will also be counted as one hint for potential *GROW*: we can probe for more available BW. We record this trend to confirm the further firm Queue decrease.

□ **Firm Queue growth(*queue*⇑)**:

1. $A_{DSDL_i} >= (1+\beta) * A_{DSDL_{i-1}}$, or
2. n continuous (queue↑),

Any of the two conditions above is met, server will go to *DRAIN* . The reasons of setting $n = 2$ are: 1.after two continuous queue↑, the DSDL growth is more than 21% which is close to $\beta = 0.25$ in the first condition above, and 2. average DSDL continuously less than minimum DSDL record for two RTT periods. $\beta = 0.25$ is also conjugated with the 25% gain value used in ***GROW***.

□ **Firm Queue decline or Probe (*queue*⇓)**:

1. $A_{DSDL_i} < (1-\beta) * A_{DSDL_{i-1}}$, or
2. m continuous (queue↓)

Any of one the two conditions above is met, server will go to *DRAIN* to use more aggressive $G_{pacing}$ to capture the potential BW growth in bottleneck.

□ ***GROW* caused queue growth(*queue*⇑′)**:

*queue*↑ or $A_{DSDL_i} >= (1+\beta) * A_{DSDL_{Gstart}}$

Server quit *GROW* after it observes an certain amount of DSDL growth (compared to the $A_{DSDL_{Gstart}}$ at the beginning of current *GROW* period).

□ ***DRAIN* caused queue decline (*queue*⇓′)**:

*queue*↓ or $A_{DSDL_i} <= (1-\beta) * A_{DSDL_{Dstart}}$

Server quit *Drain* after it observes reduction of a certain amount of DSDL reduction (compared to the $A_{DSDL_{Dstart}}$ at the beginning of current *DRAIN* period).

□ **Bytes in flight lower than expectation (*BiF*⇓)**:

BiF $< \gamma * BDP$

When the amount of bytes in flight is lower than expectation,e.g. after a application limited transmission period, the server can Choose to enter *STARTUP* again with a reserved *StartupCount* value.

□ **To exit *STARTUP* (*END*)**:

Server exit *STARTUP* when counter reaches *StartupCount* or the data increase is less than $\alpha$ for two continuous minimum RTT.

For the rest of the situation, the server will consider BN queue statues as stable and stay in ***NORMAL*** for most of the time. One more benefit of using this module is that it can work with any BWE algorithm (on TCP client or server). It is proved later in the simulation section.

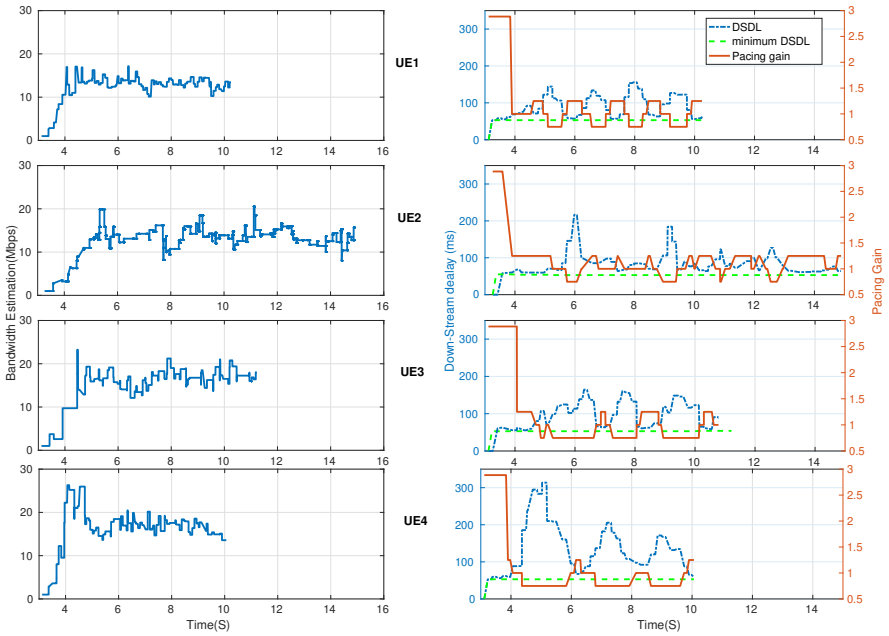



Fig. 6.6 Bandwidth estimation (left), Gain, DSDL and $DSDL_{min}$ in one experiment (right)

## 6.5 Simulation and result discussion

### 6.5.1 Simulation configuration

An NS3 simulation is conducted to measure the performance of CDBE server and the CDBE architecture as a whole. In the simulation, = 5 UEs are downloading from the remote server throughout the simulation. The size of the download is set as 10MB and 1MB to test the performance of the traffic amount. The remote server is connected to the P-gate way directly to simulate the CDN download scenario. The connection starts from 3rd second, and each full simulation lasts 15 seconds. The end-to-end performance of CDBE is evaluated against CQIC, CQIC client+CDBE server (CQIC-S), and TCP BBR. We have shown that CQIC can outperform Cubic and Westwood in this scenario, hence BBR is a reasonable TCP baseline compared to legacy TCP CCA. Note that the performance of CQIC-S is also used to validate the assumption that CDBE server can cooperate with any TCP client with BWE report capability and improve their performance by taking care of the Queue states in the bottleneck. All the TCP clients in simulation enable delay ack with the configuration of DelAckCount=2, DelAckTimer=40ms. LTE module and Internet module have been customised to carry out RLC Acknowledged Mode (AM) re-segmentation, RLC drop tail buffer and CDBE implementations. We consider a typical outdoor scenario that all the UEs are attached to one single eNB locating on the centre of a disc with a radius of 400m. 5 UEs are moving at a random walking speed

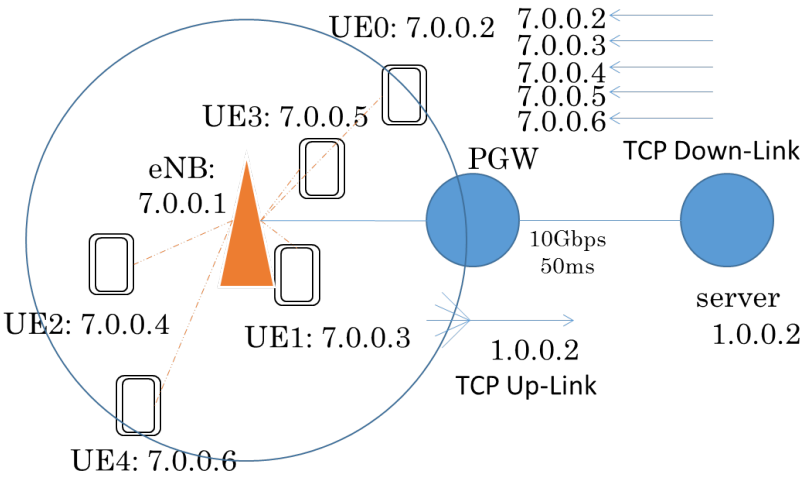



Fig. 6.7 Simulation topology and concept illustration

of 4m/s, and 250 different drops of an arbitrary initial position are tested for the statistic. The detail of the configuration is shown in Table 6.2 and the topology is shown in Fig.6.7.

### 6.5.2 Performance of CDBE server

The main simulation result is listed in Table 6.3. To validate the performance of CDBE server, we use CQIC enabled UE to report the BWE. It is known that CQIC reports an aggressive last hop BWE as evaluated in [103]. When CQIC is armed with state transition enabled server, the overall performance improves in both 1MB and 10MB cases. This result shows that enabling CDBE state machine can be beneficial to a UE which is capable of making BW report. The exceptions are CQIC-S has higher DSDL in the 1MB case, and the DS throughput of CQIC-S is lower in 10MB case. The former is due to the CQIC implementation has more aggressive initial BW report than CDBE client, plus the offensive gain used by CDBE server in *STARTUP*. While the latter is caused by the frequent use of *DRAIN* gain to suppress the CQIC BWE, which involves MAC overhead but ignores retransmission. The benefit is that the reviled eNB load and the reduced loss/RTO in this case. Overall, the goodput and RTT of CQIC type of client feedback BWE mechanism is improved by CDBE server since it somehow addresses the overestimation problem located by [103]. This result shows that CDBE server can cooperate with the client with aggressive BWE report and avoid bufferbloat to some degree.

| | | |
|---|---|---|
| Topology | Number of UEs | 5 |
| | Number of eNBs | 1 |
| | Mobility model | Random walk |
| | UE Velocity interval (m/s) | [1, 5] |
| LTE RAN | Path-Loss Model | Cost231 |
| | Fading model | EPA |
| | Tx Power (dBm) | 46 (eNB),24 (UEs) |
| | Noise Figure | 5 (eNB),9 (UEs) |
| | Scheduler type | Proportional Fair |
| | Number of Resource Block | 100 (DL/UL) |
| | RLC configuration | AM |
| | RLC buffer size | (1Mb per flow) |
| | DL/UL frequency | 2120 / 1930 MHz |
| Core Network | Delay (ms/hop) | 50 |
| | Capacity (GBps) | 10 |

Table 6.2 Simulation Configuration for CDBE validation

| | *BBR* | *CQIC* | *CQIC-S* | CDBE |
|---|---|---|---|---|
| **End-to-end Downstream(1MB and *10MB*)** | | | | |
| DS TP (Mbps) | 22.73<br>37.01 | 35.82<br>58.98 | 37.01<br>58.88 | 32.88<br>56.3 |
| DSDL (ms) | 66.45<br>115.87 | 127.1221<br>153.93 | 134.22<br>144.9 | 90.17<br>99.91 |
| **End-to-end overall** | | | | |
| Goodput (Mbps) | 3.1427<br>6.47 | 5.08<br>7.50 | 5.32<br>7.56 | 4.69<br>7.26 |
| RTT(ms) | 0.6235<br>0.7708 | 1.0630<br>1.1741 | 1.0425<br>1.0366 | 0.7399<br>0.9130 |
| Fairness (Goodput) | 0.7907<br>0.6135 | 0.8164<br>0.6360 | 0.8250<br>0.6445 | 0.8093<br>0.6332 |

Table 6.3 Compare the CDBE with Baselines

### 6.5.3 Per-flow analysis

Fig. 6.6 plots the tracks of one experiment in the data set. In this specific set of experiment, one of the UE is out of the connection range, and the distance of the rest of UEs are 46.55m,187.12m,143.57m and 109.13m away from eNB. As we can see, the BW grows exponentially with *STARTUP* gains. Once *STARTUP* is over, the server will change its state according to the manner introduced in Section 6.4. When the DSDL is small enough (close to $DSDL_{min}$) for a continuous period, the server then tries to send more data to the network for UE to prob and grasp higher potential BW. When DSDL goes up, the server will go back to *NORMAL* to see whether this DSDL stays in a reasonable range once the DSDL is too high, the server should use reserved $G_{pacing}$ to send so that the bottleneck can digest the queue. The result shows that this CCA architecture meets the objective of DS delay management.

### 6.5.4 System level result statistics, evaluation and discussion

The following subsections will discuss CDBE CCA architecture as a whole to compares to the benchmarks in four aspects:

#### Goodput and DS throughput

Compared to BBR, CDBE has better average goodput. According to the CDF plot of the result in Fig. 6.8a, the improvement in both 1MB and 10MB cases are not caused by the outliers. In 10MB case, most of BBR goes to *steady* and most of CDBE goes to its tour among *GROW*, *DRAIN*, and *NORMAL* states.

The performance of CDBE is close to CQIC-S regarding throughput and goodput, but there is still a gap. The gaps are shown in Fig. 6.8a come from the fact that the BW report value of CQIC is more aggressive than other CCAs.

#### RTT and DSDL

CDBE can outperform CQIC concerning DSDL and RTT. This result indicates that the higher BW value from CQIC BWE algorithm may not fit the queue well. Note that the configuration of parameters, e.g. $\alpha, \beta$,etc., in section6.4 can affect the performance of CDBE. However, the DSDL is always better than CQIC or BBR in larger file case. This result shows that the state transition can minimise DSDL while keeping the DS throughput at a higher level.

BBR has a slower reaction since it will need four 8-minimum-RTT cycles to double to higher BW and need to wait for the 3-tier filter time out to decrease the BWE. The reactions, as a consequence, are on the second level. As for CDBE, as long as the server enters *GROW*, it

can keep on growing as long as the hint of DSDL increase does not show. More importantly, the reported value can take effect immediately instead of using time 3-tiered filtered manner. This is the reason why CDBE has lower DSDL and higher DS throughput and higher goodput than BBR.

From the RTT point of view, CDBE has roughly 18% extra RTT for both 1MB and 10MB cases compared to BBR. It is mainly caused by a higher amount of ACK while the size of CDBE ACK is also significant. This extra amount of delay mainly comes from upstream, or more specifically in LTE uplink, due to the robust MCS choice [106]. One of the clues to support this point of view is that DSDL of CDBE is less than BBR in the 10MB case while it is the reverse in 1MB case. This result is because that the BW growth of CDBE happens one RTT faster than BBR. After one $W_l$, the *rBW* can reflect the growth of BW, and after $W_l + ULDL$ (roughly a little bit more than one RTT) the grown BW becomes effective in CDBE server. As for BBR, the time for the first BW to take effect in the BBR startup period is 2RTT. Further, the in 1MB case, the connection stays mainly in STARTUP for both BBR and CDBE and missing of necessary drain period right after startup, CDBE can reveal higher DSDL and higher DS throughput. The RTT can be improved by reducing the frequency of BW report. Currently, we report *rBW* every ACK. This strategy results in an extra 32bits of traffic per ACK in Up-stream. It can be reduced to report in the period manner or upon every a few ACK.

### Fairness

Concerning the fairness of the network, the intra-protocol fairness of CDBE can be seen as an approximation to achieve equilibrium of proportion fair share of network capacity as described in [112]. When the bottleneck is in RAN, the UE side is capable of capture the variation of capacity and report the value to CDBE server. The server will reshape the traffic to fit the current capacity share among the UE and consider the potential bufferbloat. When the bottleneck is in somewhere else in the network, the capacity estimation of CDBE client is not limited by RAN BW share. Hence the BWE in the client is reflecting the BW of the actual bottleneck and try to approach the equilibrium point of actual congestion node. Thus, the client report manner can avoid the over-aggressive behaviour of TCP BBR mentioned in [113].

For Inter-protocol fairness, we still need more experiment to discover test the friendliness of CDBE to legacy CCAs. The weakness of vegas has been taken into account and the second condition of $queue\uparrow$ in section 6.4.2 makes sure that the ***DRAIN*** only happens to take care of the bufferbloat caused by the server's traffic. The CCA algorithm module in the server can be further altered to avoid too aggressive or too conservative performance in future research.

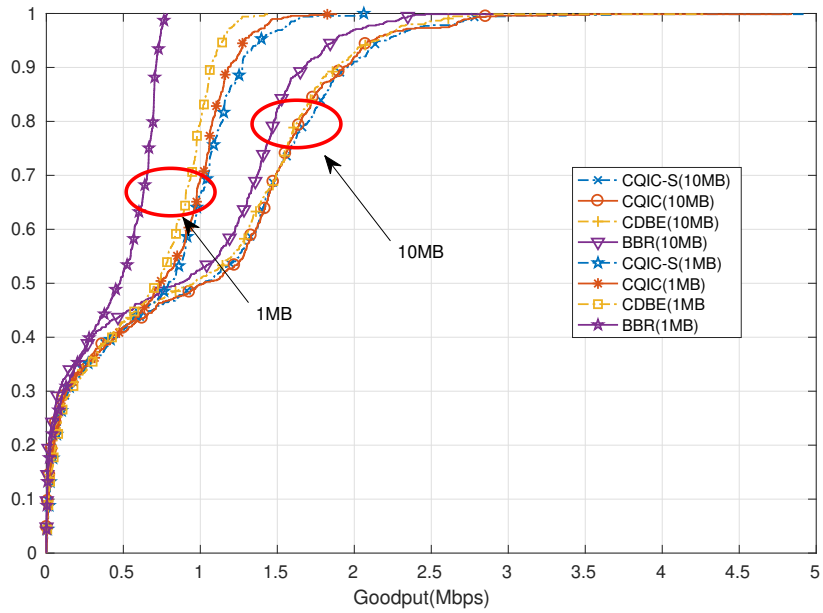
### Deployment

Last but not least, the deployment of CDBE needs the change from both the TCP server and TCP client, but only on the code in the transport layer. It simplifies the deployment compared to CQIC or other cross-layer design which requires the hardware changes. When compare CDBE to BBR, CDBE needs the support of the transport layer on the client-side. Concerning the benefit mentioned earlier, we believe it worth deploying this interactive CCA architecture for connection endpoints to better utilise the future network capacity in both mobile and wired network for the future network. It is possible to adapt the bandwidth estimation method in congestion control module in QUIC protocol as CDBE client.

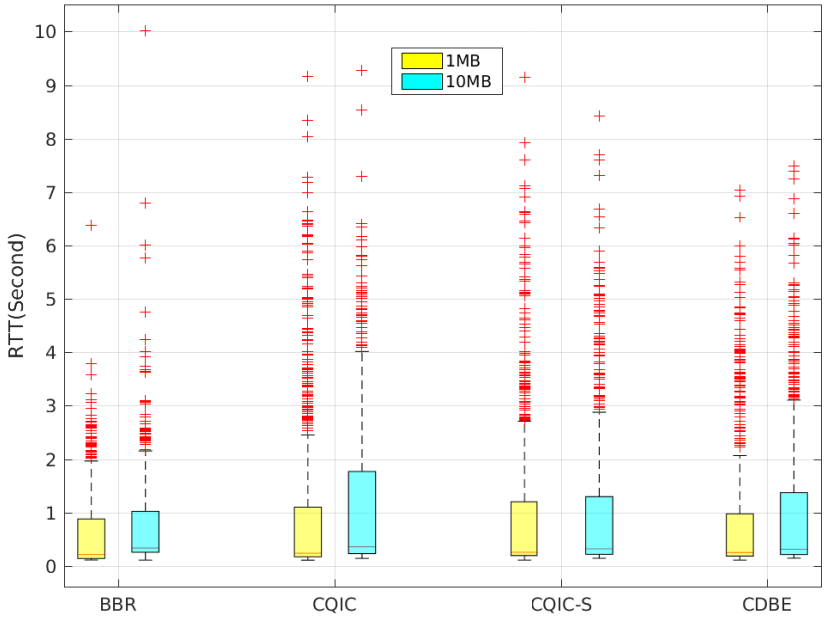
## 6.6 Conclusion

In this chapter, a novel congestion control algorithm, named CDBE, is proposed. Both the client and the server play roles in this CCA. The two components of the proposed CCA architectures are bandwidth client and a state transition module in the TCP server. The CDBE client uses a filtered windowed algorithm to calculate the bandwidth and send it back the server. The state machine in UE can decide how to use the reported value by judging the information ( in this work, downstream delay) extracted from TCP timestamp The advantages of this CCA are: 1.mitigate the influence of LTE uplink or bottleneck in up-stream path; 2. promptly estimation of the bandwidth, 3. Compatible CDBE server for different kinds of UEs, and 4. simple BWE algorithm on UE.

However, this state machine cannot guarantee the bandwidth convergence if the bottleneck is in the backhaul or other part of the fixed network. The convergence of several CDBE flows relies entirely on the scheduling algorithm in the BS. The second version of CDBE is proposed in chapter 7.




a




b

Fig. 6.8 (a) Goodput CDF of CDBE, CQIC, CQIC-S and BBR and (b) RTT samples with 75% and 99% percentiles

# Chapter 7

# CDBEv2: Toward ubiquitous congestion control in a mobile network

## 7.1 Introduction

In this chapter, the technical details of CDBE on both server state machine and UE side are both redefined. By referring the following discussions in previous chapters:

- The equilibrium achieved by AIMD as mentioned in chapter 4,
- the client side bandwidth estimation introduced in CQIC, DCIC and CDBE in chapter 3,5,6
- the queue management concept in TCP Vegas, BBR and CDBE, in chapter 3,5,6 and
- the features in a mobile network in both Radio Access Network and Packet core network(backhaul) in background review3,

our design of CDBE version 2 (CDBEv2) should be able to achieve both fast convergence in the fixed network and the mobile network. Meanwhile, the bottleneck bandwidth is fairly shared by the flows while the buffer depth should also be low.

The client-side bandwidth estimation is Further simplified. Now the client will only report a simply calculated bandwidth sample and piggyback the BW sample back to the server. A dynamic low pass filtering method takes place on the server-side. To configure the parameters of the design, we propose two models for the STARTUP state to find out the optimum parameter configurations. The formula used in the two models can be further adapted to mobile networks with different traffic statistics or RAN, EPC configurations. In some saturated cases, we proposed the more effective draining method and gaining compensation method to lower the

level of congestion and also improve fairness. The Bytes in flight instead of RTT is used as a hint of state transition.

To validate our design, we practised several simulations on both fixed and mobile network to show the merit of CDBEv2.

As expected, the simulation result shows that the combination of all improvements in this chapter allow multiple flows to share the bandwidth in a balanced manner in both the fixed network and the 4G mobile network. The bottleneck queue size is also small since the design treats the buffer in the network as an operation redundancy instead of the resource to occupy. As a result, CDBEv2 achieves high goodput, low RTT and low loss rate in both the fixed network and the mobile network.

## 7.2 From CDBE to CDBEv2

### 7.2.1 Simplified Client-Side bottleneck bandwidth estimation

The original multiple window manner plus low pass filtering manner in CDBE is discarded.

Instead, a single time window(TW) manner is used for Bandwidth estimation (BWE) method. The calculation shall include not only the valid data in sequence but the data with repeat sequence number should also be taken into consideration. Meanwhile, the TCP header size is not taken into account since the algorithm is calculating the throughput on the transport layer. The equation for the BWE method is shown below:

$$rBW(t_i) = \frac{totalBytesReceived}{\Delta t} \tag{7.1}$$

where $\Delta t$ is the difference between the first received packets and the last packet in the current time window. The TW used in client-side should cover at least one potential 7ms RLC layer retransmission in a 4G network when the Physical layer link is suffering from interference, and at the same time, it should also be smaller than the typical minimum Round Trip Time($RTT_{min}$) in a mobile network which is 20ms. Hence our choice is 10ms. For better utilization of this reported value, a low-pass filter manner is employed in the server-side, which will be further explained in the next section. The report is further sent back to server once the calculation is triggered. The BW option is added to the header roughly every $TW_{client} = 10ms$ if the arriving data rate is enough or the gaps between arrival exceeds TW. The structure of the BW option is shown in Fig. 7.1. As shown in the figure, the temporary KIND of the option is 32, and the LENGTH of BW option is six, and the range of BW option field varies from 0bps to $2^{48}$ bps. This option is appended after TS options. Once the reported bandwidth is returned to

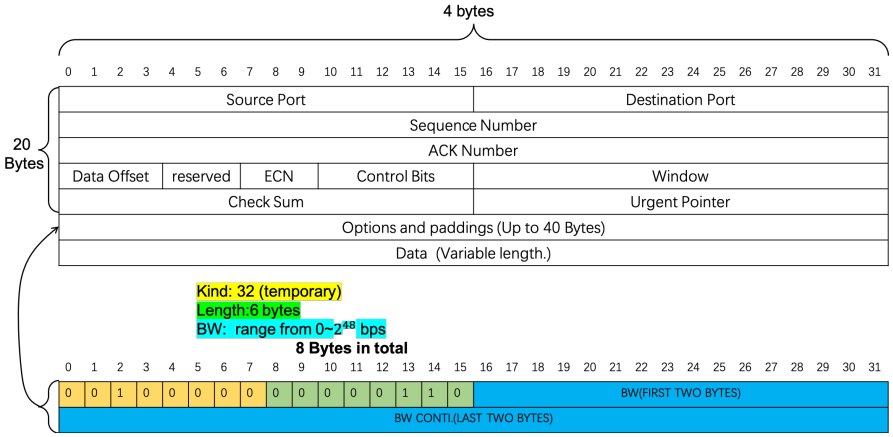



Fig. 7.1 The structure of BW option field in TCP header.

the server, the server can extract the bandwidth option and send process the real-time reported value $rBW(t_i)$.

### 7.2.2 Server Side Windowed BWE report utilisation with a dynamic low-pass filter

#### Bandwidth sample and its bound

As described in chapter 6, principally the CDBEv2 still use the BW reports from the UE side to calculate its pacing rate and the CWND to limit the traffic: The real time sample pacing interval between the packets are calculated by dividing the segment size by the product of pacing gain($G_{pacing}$)bandwidth sample, and the CWND sample is calculated by multiplying BDP with the CWND gain ($G_{cwnd}$)A lower bound of bandwidth($minBW$) is added to the calculation to avoid the underestimation caused by the radio link loss:

$$P_i = \frac{Seg}{G_{pacing} * max(BW(t_i), minBW)} \tag{7.2}$$

$$CWND_i = G_{cwnd} * BW(t_i) * RTT_{min} \tag{7.3}$$

where the $P_i$ and $CWND_i$ is the pacing rate and congestion window at time $t_i$, $RTT_{min}$ is the minimum round trip time in use at the moment. $BW(t_i)$ is the bandwidth in use. During the STARTUP and GROW state, $BW(t_i)$ is equivalent to the recorded maximum value $BW_{max}$. Similarly in the TCP state of RECOVERY/LOSS, the $BW_{max}$ in last Cycle also takes effect. For the rest of the time $BW(t_i)$ is updated in low-pass filtered manner. The full bandwidth update algorithm is shown below:

**Algorithm 7.1:** Update $BW(t_i)$ in use

```
/* Server receives a BW report rBW(t_i) at time i, it triggers BW(t_i)
   update                                                          */
input  : rBW(t_i),tcpState
output : BW(t_i)
1 if rBW(t_i) > BW_max then
      // BW_max will be reset to BW(t_i) at the beginning of each cycle
2     BW_max = rBW(t_i);
3 if (tcpState==RECOVERY or LOSS) then
4     BW(t_i) = BW_max;
5 else
6     if rBW(t_i) > RTT_min then
7         BW(t_i) = (1 − β) ∗ BW(t_{i−1}) + β ∗ rBW(t_{i−1});
```

At the beginning of each DRAIN cycle, a maximum value of bandwidth $BW_{max}$, is reset to $BW(t_i)$ The BW is also lower-bounded by 300kbps, which is selected according to the minimal physical layer bandwidth provided by the 4G network and also takes the IPV4 Maximum segment size in consideration:

$$minBW \geq max(\frac{MSS*8}{TW}, 300kbps) \tag{7.4}$$

where the MSS is 576 octets[114] and TW is the time window(TW) used in receiver side is 20ms. The full design, mentioned above, prevents the unexpected underestimation/overestimation of BWE. It also leaves all the filtering and smoothing job to the data server with less power constrain compared ta o mobile device.

### Dynamic low pass filtering parameter for a range of RTT

In the first draft of CDBE, we calculate, update and filter the bandwidth sample in Client-side. This can be an extra resource consumption for a mobile device as the power supply of such equipment is using a portable battery. Hence in this version of CDBEv2, the client only report the Bandwidth sample back to the server. The server can use the report in a similar low-pass filtered manner:

$$BW(t_i) = (1-\beta)*BW(t_{i-1}) + \beta * rBW(t_{i-1}) \tag{7.5}$$

Whenever the server received a BW option, the BW update is triggered. We take the elimination of jitters in the bandwidth report, which may be caused by the bursty new-comer or random packet discard or other unexpected situation, into our consideration. Hence, our goal is to make

sure the new rate will be updated as close to an actual new value as possible every two $RTT_{min}$. Assuming the BW report($rBW$) frequency and the $rBW$ value is a stable constant before the change of transmission interval on the server-side. The bandwidth in use will approach this constant. For the "converge" from $n^{th}$ report, we have:

$$\begin{aligned} BW(1) &= (1-\beta) * BW(t_0) + \beta * rBW(t_0) \\ BW(2) &= (1-\beta) * BW(t_1) + \beta * rBW(t_2) \\ &\dots \\ BW(t_n) &= (1-\beta) * BW(t_{n-1}) + \beta * rBW(t_{n-1}) \end{aligned} \tag{7.6}$$

$BW(t_0)$ is the initial bandwidth value, $rBW(t_0), rBW(t_1) \dots rBW(t_n)$ are all equivalent. substitute

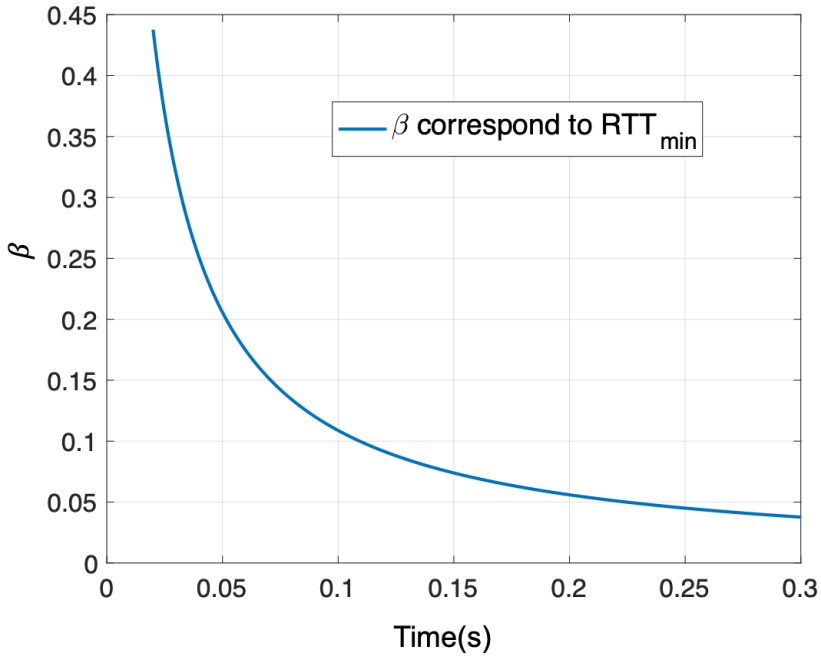



Fig. 7.2 How does $\beta$ change with RTT on interval $RTT_{min} \in [20ms, 300ms]$

the equations above step by step we have:

$$\begin{aligned} BW(t_n) &= (1-\beta)^n * BW(t_0) + \sum_{n=1}^{N} (1-\beta)^n \beta * rBW(t_0) \\ &= (1-\beta)^n * BW(t_0) + (1-(1-\beta)^{n-1}) rBW(t_{n-1}) \\ &= (1-\beta)^{n-1} * ((1-\beta) BW(t_0) - rBW(t_{n-1})) + rBW(t_{n-1}) \end{aligned} \tag{7.7}$$

Since the report time window is set to 10ms for the mobile network, and the dominate term which affects the convergence is $(1-\beta)^{n-1}$, we have the following equation to help our dynamic lowpass-filter design:

$$\begin{aligned} & (1-\beta)^{n-1} < thres \\ >> \quad & \beta > 1-(thres)^{1/n} \end{aligned} \tag{7.8}$$

The *thres* we have here is 10%, which means n-1 number of $TW = 10ms$ reports latter, the bandwidth in use will be at least 90% of the new stable value . Hence n is calculated by:

$$n = \frac{RTT_{min}}{TW} \tag{7.9}$$

To clearly observe how the value for $\beta$ in equation 7.8 evolve with the $RTT_{min} \in [20ms, 300ms]$, we plot the curve in Fig.7.2. As we can see that for 10ms case $\beta = 0.437$ will guarantee the probe method used in GROW can get the new maximum bandwidth at two reports interval time while at 300ms $RTT_{min}$ case, 30 reports later the bandwidth in use should approach 90% of new bandwidth while the extra jitter will be eliminated.

### 7.2.3 Advanced features in CDBEv2 state machine

Similar to the first version of CDBE, the state machine includes four states: STARTUP, STEADY, GROW and DRAIN. The four states correspond to the tasks for CCAs we presented in earlier chapters. The technical details are different, and they are defined as follows:

#### STARTUP

STARTUP is the period which provides the server with the capability to probe the maximum available bottleneck bandwidth without hitting through the buffer depth in single flow case. When multiple flows coexist in a path, the STARTUP should also guarantee a reasonable level of bandwidth share. Hence the quitting condition of this state is critical. In Loss-based congestion control algorithms, such quitting condition is to hit through the buffer. BBR, BBR' and BBR plus uses stable maximum bandwidth for several continuous $RTT_{min}$ as the quitting condition, which may cause severe loss. To evaluate the quitting condition, we modelled a single flow scenario and two/multiple flow scenario. The quitting decision of the STARTUP is made as long as any of the following conditions are met:

Where the two counter related threshold is explained further in the following sections. G is the RTT gradient:$G(t_i) = RTT(t_i) - RTT(t_{i-1})$ which is to avoid the bursty transmission of accumulated un-successful packets in RLC layer. Generally speaking, when the TCP state goes

**Algorithm 7.2:** Quitting condition for STARUP state

```
/* Server receives a new ACK at time i in STARTUP state, The latest DSDL
   reported by the TIMESTAMP option is the input of this algorithm to
   decide whether to quit STARTUP                                      */
input  : tcpState,DSDL(t_i)
output : BW(t_i)
1 if DSDL(t_i) − DSDL_min > Thres1&&G(t_i) > 0 then      // Thres1 = 2 ∗ DSDL_min
2     thres1Counter++;
3 if DSDL(t_i) − DSDL_min > Thres2&&G(t_i) > 0 then      // Thres2 = 1Mb/BW_max
4     thres2Counter++;
5 if (tcpState==RECOVERY or LOSS)
6     ||thres1Counter > N
7     ||thres2Counter > N
8  then                                                  // N = 5
9     QuitSTARTUP();
```

to RECOVERY due to multiple duplicated ACK, the server will quit STARTUP stage. When the five or more times that DSDL from TIMESTAMP option carried by ACKs indicates that the DSDL exceeds a certain value, the server should also quit STARTUP.

The choice of $Thres1 = 2 * DSDL_{min}$ and $hres2 = \frac{1Mb}{BW_{max}}$ depends on the choice of

1. STARTUP gain pairs: $G_{pacing}$ and $G_{CWND}$,

2. Facts that the $RTT_{min} \in [20ms, 300ms]$ and $BW_{Btlnck} \in [1Mbps, 500Mbps]$,

3. Facts that the buffer size in bottleneck is about 3MB to 6MB((24Mb to 48Mb)), and

4. The Goal fair share of Bandwidth and low loss possibility when quitting STARTUP state

Firstly the choice of STARTUP gain is based on the following analysis:

- **Gains in STARTUP**

**1.Single flow analysis**

In the single flow case, the buffer size before the first DRAIN contains two parts: unavoidable delay and signalling delay: the mismatch of $BW(t_i)$ and bottleneck $BW$ is the reason of unavoidable delay, and its amount in our set up is:

$$Delay_u = \frac{(G^n_{pacing} * BW(t_0) - BW_{Btlnck})}{BW_{Btlnck}} * RTT_{min} \tag{7.10}$$

where,

$$n = floor(\log_{G_{pacing} * BW(t_0)}^{BW_{Btlnck}}) \tag{7.11}$$

is the $n^{th}$ STARTUP cycle before the maximum bandwidth is found. Since, before the $n^{th}$ STARTUP cycle, the $BW_i$ in use multiplied by the STARTUP gain is lower than $BW_{Btlnck}$, there will not be any queue accumulation in the bottleneck buffer. The queuing up delay is reflected as growing RTT to the server while the latest maximum bandwidth is also reported. As described in the quitting STARTUP algorithm 7.2, the threshold 4 is preset as $Q_{Thres} = 1Mb$ This limits the signalling delay we defined below:

$$Delay_s = \frac{Q_{Thres}}{BW_{Btlnck}} \tag{7.12}$$

In the quitting STARTUP round of single flow case, the maximum bandwidth is found. Hence, the signalling delay is defined by the reported $BW_n = BW_{Btlnck}$. The related Queuing size in the network for a single flow is:

$$\begin{aligned} Q_{total} &= Delay_u * BW_{Btlnck} + \frac{Q_{Thres}}{BW_{Btlnck}} * G_{pacing} * BW_{Btlnck} \\ &= (G_{pacing}^n * BW(t_0) - BW_{Btlnck}) * RTT_{min} + G_{pacing} * Q_{Thres} \end{aligned} \tag{7.13}$$

which, is the amount for housekeeping for the DRAIN state. Together combines 7.10 and 7.13,we have the loss condition for typical mobile network is:

$$Q_{total} < 6MB \tag{7.14}$$

Since the last $Q_{Thres}$ is a constant and the $RTT_{min}$ is simply a coefficient, the loss condition for different $BW_{Btlnck}$ depends on $BW(t_0)$ and $G_{pacing}$. The optimal pair of $BW(t_0)$ and $G_{pacing}$ can provide a minimum average delay and minimum loss rate on all possible value of $BW_{Btlnck}$.

Since in BBR series, they try to fit the pacing rate growth to the curve of 2.88 or 2.77, as the derivation in chapter 4 while the latest BBR2 reduced this STARTUP gain to 2. Hence the essential meaning of gain parameter is just to fit the growth of pacing rate into a certain curve. Hence our goal is to find out a gain factor between 2 and 2.88 which can minimize both the Queue length and the total delay. In which the two factors we can control are $G_{pacing}$ and $BW(t_0)$.

Using the model above, we firstly investigate the typical value of $G_{pacing} = 2, 2.77, 2.88$. Fig.7.3 shows the plot of Queuing delay and the Queue size calculated by the single flow model when $BW(t_0) = 1Mbps$. The typical maximum Buffer size is 6MB. When the queue size is

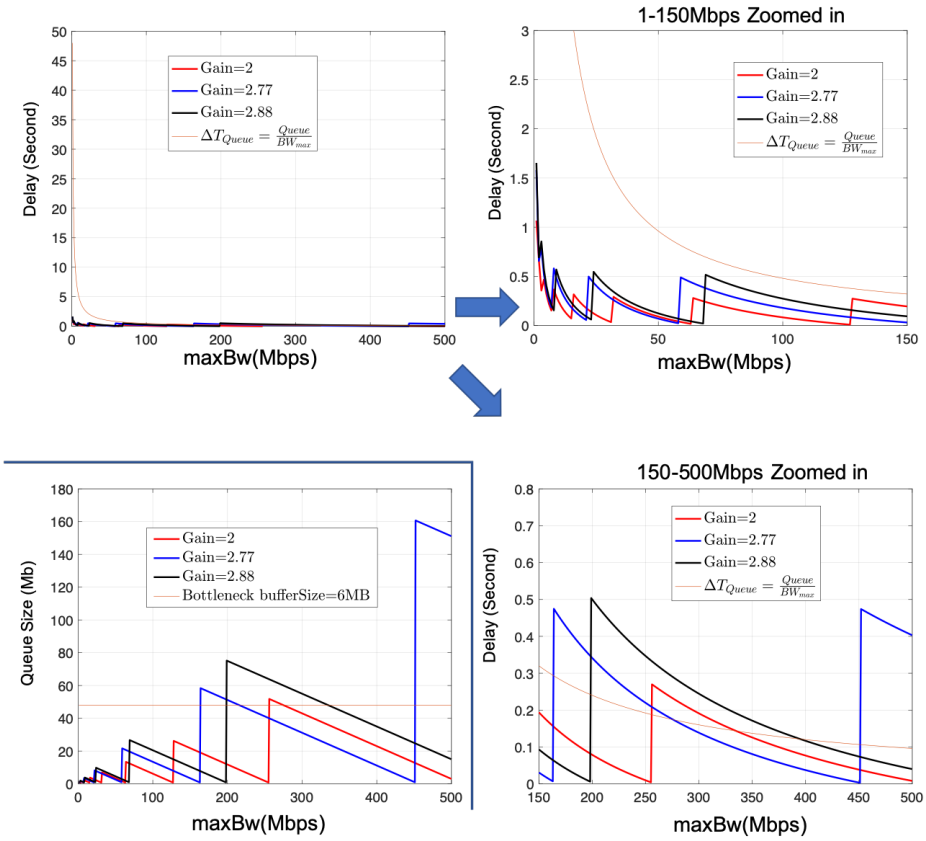



Fig. 7.3 Queuing delay caused by STARTUP stage for $G_{pacing} = 2, 2.77, 2.88$ with $BW(t_0) = 1Mbps$

larger than 6MB, the loss happens. As shown in the up left subplot, the maximum lossless queuing delay$max\Delta T_{Queue}$ decreases with the growing maximum bottleneck bandwidth. On the up right subplot, it is the zoomed-in version for maximum BW from 1Mbps to 150Mbps. The delay caused by the STARTUP period, in this $G_{pacing}$ range, does not exceed the delay caused by the $max\Delta T_{Queue}$. On the bottom right subplot, it is the zoomed-in version for bottleneck bandwidth($BW_{Btlnck}$) from 150Mbps to 500Mbps. We can see that different $G_{pacing}$ will result in the delay to exceed $max\Delta T_{Queue}$ for different $BW_{Btlnck}$. Though the Gain of 2 has the lowest loss rate among the three classic $G_{pacing}$ value, it may not be the most time-efficient option to search for the maximum BDP. Meanwhile, the initial $BW(t_0)$ is also a factor which, combined with $G_{pacing}$, will tackle the curves of delay together.

To observe on which degree, the cooperation of $G_{pacing}$ and $BW(t_0)$ will cause the packet loss on the STARTUP stage. The delay for a full range of bottleneck bandwidth( from 1Mbps to 500Mbps) is averaged and plotted on Fig. 7.4 corresponds to each $G_{pacing}$ and $BW(t_0)$ pair.

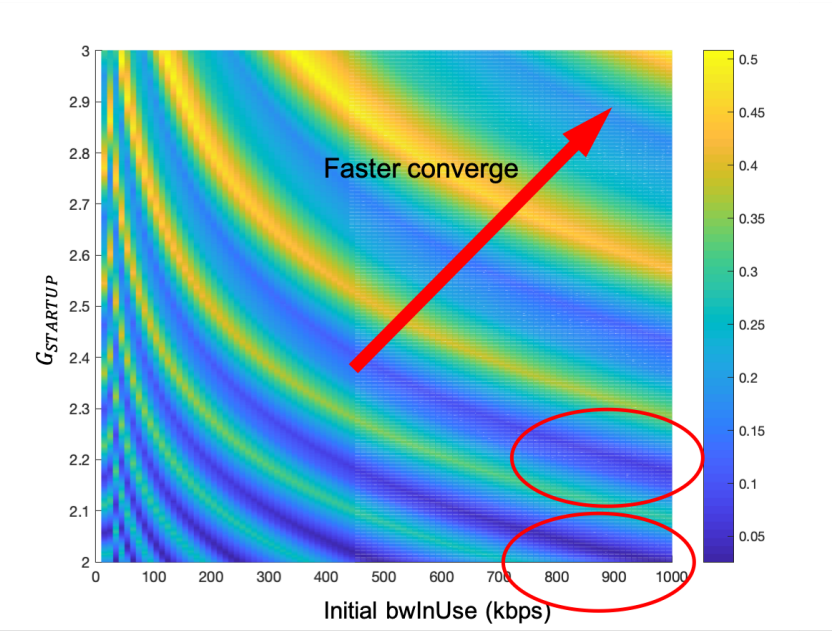



Fig. 7.4 Overall mean delay for initial bandwidth from 100Kbps to 1Mbps and the STARTUP Gain from 2 to 3 respectively.

Obviously, the larger the $G_{pacing}$ and $BW(t_0)$, the faster the STARTUP will find the maximum BDP for the pipeline. Hence for the higher searching efficiency, the upright conner is the direction for us to look for the optimal pair.

In Fig. 7.4, the yellow colour indicates the higher mean delay while the blue implies the lower delay. In the figure, we have circled the two possible areas where the overall delay is low. Roughly we can tell that area A ($\Phi A$)$G_{pacing} \in [2, 2.1]$ with the range from $BW(t_0) \in [800Kbps, 1Mbps]$, has lower delay while area B ($\Phi B$) $G_{pacing} \in [2.2, 2.3]$ with the range from $BW(t_0) \in [800Kbps, 1Mbps]$ has similar effect.

To further observe the performance of different configuration, the mean, maximum, and minimum Queue size, corresponds to different $BW_{Btlnck}$, caused by the STARTUP stage with different pair of $G_{pacing}$ and $BW(t_0)$ is plotted in Fig. 7.5 and Fig. 7.6. According to the result shown in Fig.7.3, the detail observation also proceeds in two $BW_{Btlnck}$ scope: 0-150Mbps and 150-500Mbps: the sensitivity of RTT and loss are different in the two BW region.

Another important measure is the percentage of loss, which is calculated by dividing the number of corresponding delay which exceeds the $max\Delta T_{Queue}$ by the total number of delay calculated throughout the full range of $BW_{Btlnck}$ (1Mbps, 1.1Mbps, 1.2Mbps...500Mbps).

As we can see in Fig.7.5 , the blue strips for smaller $BW_{Btlnck}$ range follows the similar trends as in overall $BW_{Btlnck}$ range. However, the exact $G_{pacing} = 2$ and $BW(t_0) = 1Mbps$ is no longer the best option as the mean and maximum delay in this $BW_{Btlnck}$ indicated. As for the

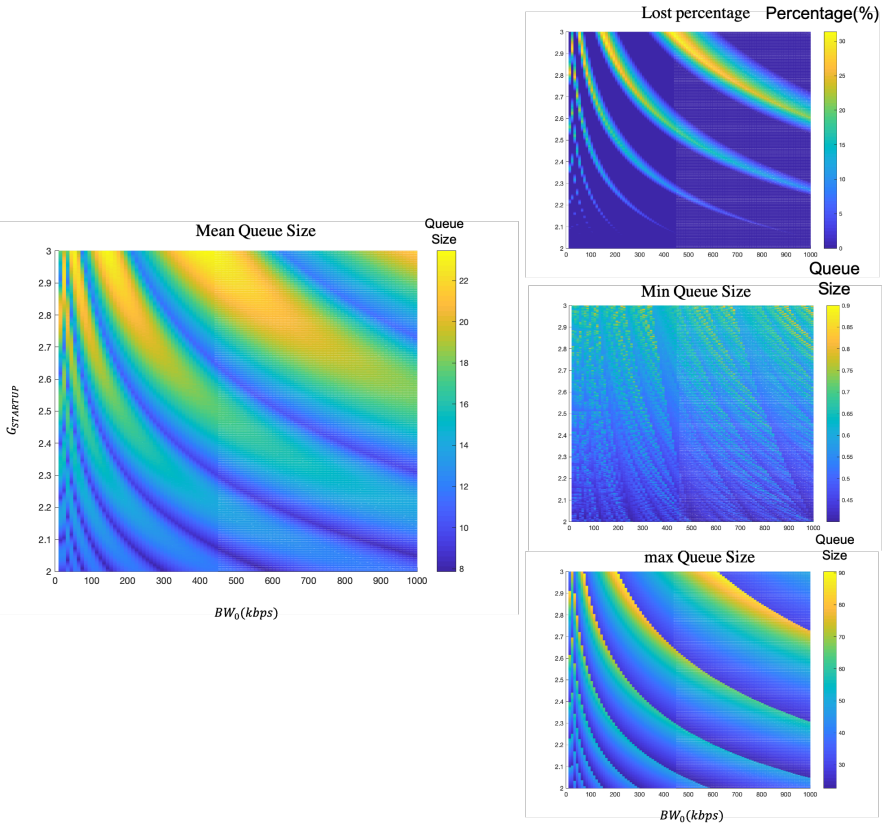



Fig. 7.5 The the mean, maximum, minimum delay and lost percentage caused by different pairs of $G_{pacing}$ and $BW(t_0)$ for $BW_{Btlnck} \in [1Mbps, 150Mbps]$

loss caused by over transmission as the *Lost percentage,* on up-right conner shows, the selected $\Phi A$ and $\Phi B$ has no loss in this $BW_{Btlnck}$ scope.

For the Large $BW_{Btlnck}$ dimension in Fig.7.6, the same trend of stripes are also observed. Though the bound of rough areas $\Phi A$ and $\Phi B$ is slightly shifted back to more similar to the $\Phi A$ and $\Phi B$ in Fig.7.3. This is not only caused by the range of $[150Mbps, 500Mbps]$ is more significant than $[1Mbps, 150Mbps]$ but also the value of Queue size and the corresponding delay is substantially larger than that in the small $BW_{Btlnck}$ dimension.

The loss percentage in Large $BW_{Btlnck}$ is also considerable, from about 25% to 80% while this measure is from 0% to 30% when $BW_{Btlnck}$ is small. Still, the two concerned areas have similar average low Queue size, and low loss percentage in $\Phi A$ and $\Phi B$ are no more than 35%.

Concerning the minimum delay, the optimal area is between $\Phi A$ and $\Phi B$.

With these supporting evidence, we are now going to pick the practical operation points and some reference points for a mobile network with RTT range from 20ms to 300ms and $BW_{Btlnck}$ from 1Mbps to 500Mbps.

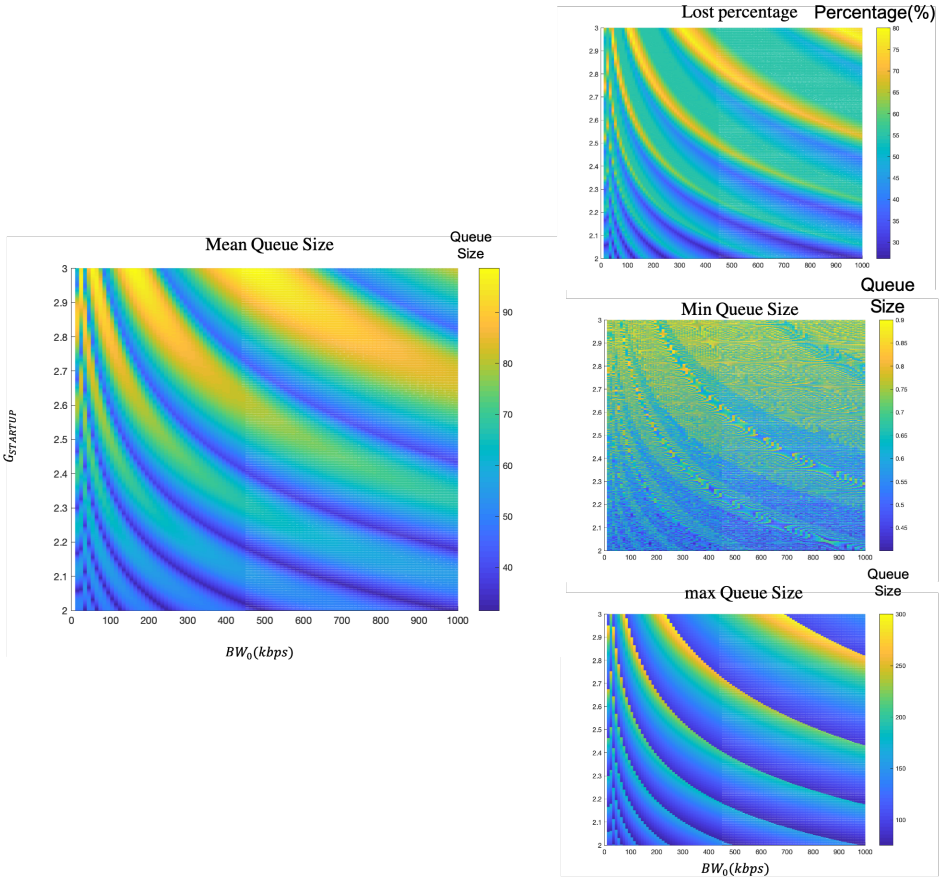



Fig. 7.6 The the mean, maximum, minimum delay and lost percentage caused by different pairs of $G_{pacing}$ and $BW(t_0)$ for $BW_{Btlnck} \in [150Mbps, 500Mbps]$

As shown in Fig. 7.7, four pairs of practical $G_{pacing}$ and $BW(t_0)$ sets are selected and their properties are shown in the table below:

A reference pair is also shown in the last row of the table 7.1, which has larger Queue size, corresponding delay and the higher Loss percentage in our model, which was the configuration in BBR. The final choice is $G_{pacing} = 2.175$ and $BW(t_0) = 1Mbps$ for the mobile network parameter setup since it has the medium convergence speed while the delay and average loss probability for larger BW set up with 6MB buffer size is relatively low and acceptable.

The quitting condition for multiple flows is also typical to avoid too much loss and too much delay.

**2.Multiple flow analysis** In this analysis, we assume there is an existing flow which fully utilizing the full $BW_{Btlnck}$ at $t_0$, denotes this bandwidth as $BW_0(t_0)$. A new-comer flow is probing the maximum available bandwidth with the pacing gain of 2.175 and initial bandwidth $BW_1(t_0) = 1Mbps$. The maximum buffer size is still 6MB.

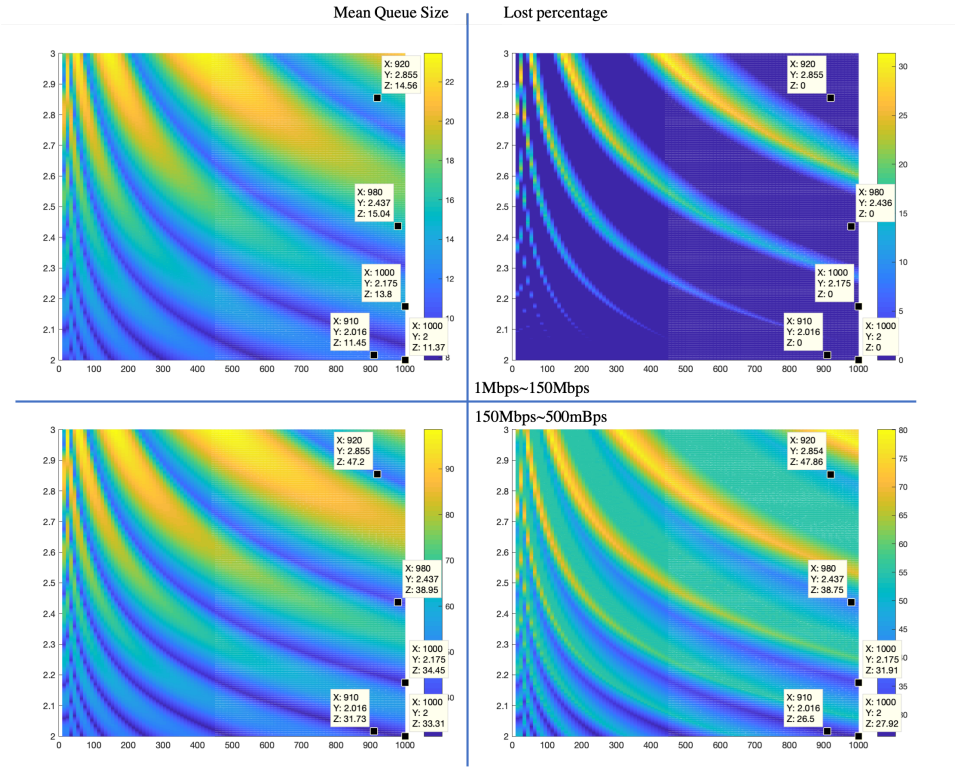



Fig. 7.7 Summary of STARTUP Gain pair selection

Suppose that the buffer is allocating the $BW_{Btlnck}$ in a fair manner which is basically true in the mobile network since the proportional fair is the baseline used [82]. Hence there is a moment $t_1$ when the allocated bandwidth is reflected back to the transmitter:

$$\begin{aligned} BW_1(t_1) &= \frac{G * BW_1(t_0)}{G * BW_1(t_0) + \alpha * BW_0(t_0)} * BW_{Btlnck} \\ BW_0(t_1) &= \frac{\alpha * BW_0(t_0)}{G * BW_1(t_0) + \alpha * BW_0(t_0)} * BW_{Btlnck} \end{aligned} \tag{7.15}$$

| Pair value | $\overline{Q}_{total}$ in $BW^{small}_{Btlnck}$ | $\overline{Q}_{total}$ in in $BW^{large}_{Btlnck}$ | Loss in $BW^{large}_{Btlnck}$ | Converge speed | Type |
|---|---|---|---|---|---|
| $G_{pacing} = 2$ $BW(t_0) = 1Mbps$ | 11.37Mb | 33.31Mb | 27.92% | Slowest | Back Up Choice |
| $G_{pacing} = 2.016$ $BW(t_0) = 910Kbps$ | 11.45Mb | 31.37Mb | 26.5% | Slower | Back Up Choice |
| $G_{pacing} = 2.175$ $BW(t_0) = 1Mbps$ | 13.8Mb | 34.45Mb | 31.91% | Medium | Selected |
| $G_{pacing} = 2.437$ $BW(t_0) = 980Kbps$ | 15.04Mb | 38.95Mb | 38.75% | fast | Back Up Choice |
| $G_{pacing} = 2.855$ $BW(t_0) = 920Kbps$ | 14.37Mb | 47.2Mb | 47.86% | Fastest | Reference |

Table 7.1 Performance of different pair of Gain and initial BW.

where $G$ is the simplified notation for both$G_{CWND}$ and $G_{pacing}$ and $\alpha$ is the pacing gain used for current flow. the reflected bandwidth of following time moments for the newcomer flow are:

$$\begin{aligned} BW_1(t_2) &= \frac{G * BW_1(t_1)}{G * BW_1(t_1) + \alpha * BW_0(t_1)} * BW_{Btlnck}, \\ BW_1(t_3) &= \frac{G * BW_0(t_2)}{G * BW_1(t_2) + \alpha * BW_0(t_2)} * BW_{Btlnck}, \\ &\ldots \\ BW_1(t_n) &= \frac{G * BW_0(t_{n-1})}{G * BW_1(t_{n-1}) + \alpha * BW_0(t_{n-1})} * BW_{Btlnck} \end{aligned} \tag{7.16}$$

substitute the equations above step by step we have the formula of bandwidth in use of new-comer flow:

$$BW_1(t_n) = \frac{G^n * BW_1(t_1)}{G^n * BW_1(t_1) + \alpha^n * BW_0(t_1)} * BW_{Btlnck} \tag{7.17}$$

Similarly, the formula for current flow at $t_n$ is:

$$BW_0(t_n) = \frac{\alpha^n * BW_1(t_1)}{G^n * BW_1(t_1) + \alpha^n * BW_0(t_1)} * BW_{Btlnck} \tag{7.18}$$

In the current analysis, $\alpha$ in use is 1 for convenience. Based on this assumption, it is practical to represent the multiple flow case with two flow equation since the sum and trend of change of multiple I flow is the same term as a single existing flow:

$$BW_0(t_n) = \sum_{i=1}^{I} BW_i(t_n)$$

According to our initial assumption, at $t_1$, the buffer has built up to a certain level. The bottleneck drains the buffer at a rate of $BW_{Btlnck}$, and the two flows are feeding the buffer at an initial constant rate. Hence the Queue size at $t_1$ is as follows:

$$Q(t_1) = (G * BW_0(t_0) + \alpha * BW_1(t_0)) * RTT_(t_0) \tag{7.19}$$

$$\begin{aligned} RTT(t_1) &= \frac{Q(t_1)}{BW_{Btlnck}} \\ &= \frac{G * BW_0(t_0) + \alpha * BW_1(t_0)}{BW_{Btlnck}} * RTT_{min} \end{aligned} \tag{7.20}$$

where $BW_1(t_0) = BW_{Btlnck}$ and $RTT_{t_0} = RTT_{min}$. With $\alpha = 1$, $BW_0(t_1)$ and $BW_0(t_1)$ shown in equation 7.15, for the moment $t2$ to $t_n$ we have:

$$\begin{aligned} Q(t_2) &= (G * BW_0(t_1) + \alpha * BW_1(t_1)) * RTT_(t_1), \\ &= \frac{G^2 * BW_1(t_1) + \alpha^2 * BW_0(t_1)}{G * BW_1(t_1) + \alpha * BW_0(t_1)} * BW_{Btlnck} * RTT(t_1) \\ Q(t_3) &= (G * BW_0(t_2) + \alpha * BW_1(t_2)) * RTT_(t_2), \\ &\ldots \\ Q(t_n) &= (G * BW_0(t_n) + \alpha * BW_1(t_n)) * RTT_(t_n) \\ RTT(t_n) &= \frac{Q(t_n)}{BW_{Btlnck}} \end{aligned}$$

substitute the equation above step by step and apply Equation 7.18 and 7.17, we have:

$$\begin{aligned} Q(t_n) &= (G * BW_0(t_{n-1}) + \alpha * BW_1(t_{n-1})) * RTT_(t_{n-1}), \\ &= (G^n * BW_0(t_0) + \alpha^n * BW_1(t_0)) * RTT_{min} \\ RTT(t_n) &= \frac{G^n * BW_0(t_0) + \alpha^n * BW_1(t_0)}{BW_{Btlnck}} * RTT_{min} \end{aligned} \tag{7.21}$$

The evolution of bandwidth of existing data flow and the new-comer flow with time shows can be analyzed in the quantitively. This model can also trace the corresponding Queue size and RTT trend.

According to the model, we can figure out the rough conditions which can use to quit the STARTUP stage. Our goal is to make sure that, in a homogenous network, when the new-comer quits the STARTUP, the share of bandwidth is relatively fair, and the loss possibility is minimized.

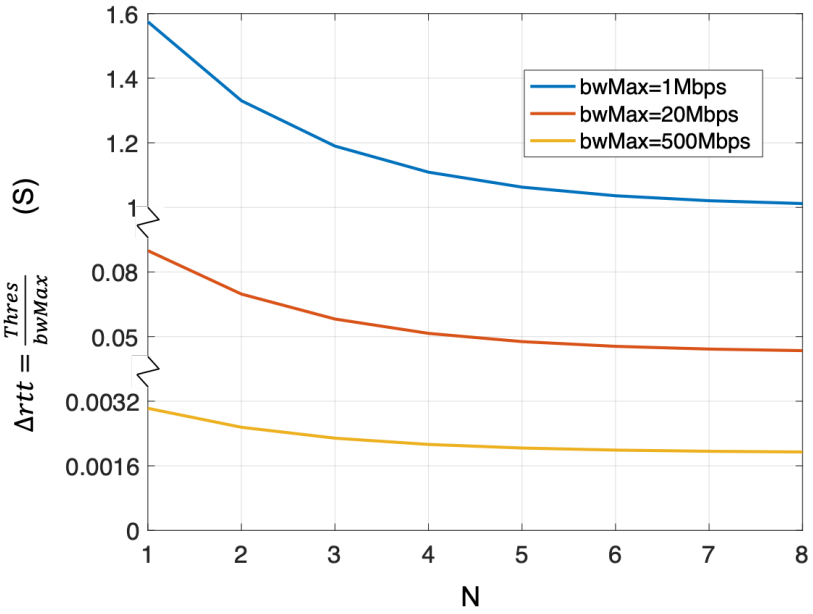



Fig. 7.8 Performance of thresholds, evolution of $BW_0/BW_1$ and Queue size in bottlneck in $BW_{Btlnck} = 1Mbps$

Heuristically, the delay caused by the new-comer should not exceed a certain level. The first threshold come to connect the BW feedback to a certain level of buffer depth. This threshold can also provide a certain level of fairness[115, 116]. However, this condition is bandwidth sensitive, as shown in Fig.7.8. As we can see that the lower bound of BW can allow $\Delta t$ up to more than 1 second which can be un-reachable even if the new-comer flow fully occupies the bandwidth and use $G_{pacing} * BW_{Btlnck}$ to transmit. This is the nature of BDP based congestion control. Hence, a dual threshold-based quitting conditions proposed for new-comer flow to quit STARTUP are:

$$\begin{aligned} &\Delta RTT > Thres1 = \gamma * RTT_{min}, \\ &\qquad or \\ &\Delta RTT > Thres2 = \frac{Q_{Thres}}{BW_1} \end{aligned} \tag{7.22}$$

Where $\gamma$ and $Q_{Thres}$ the two parameters which are going to be discussed. As shown in Fig. 7.9,7.10,7.11,7.12 and 7.13, the Growth of Queue size are different due to the maximum bandwidth feedback loop strategy used by CDBE.

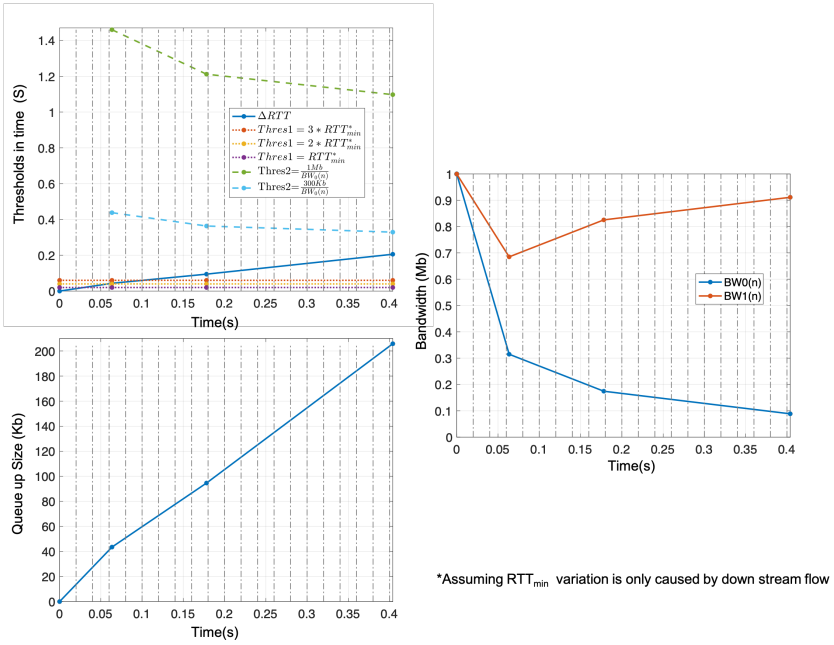



Fig. 7.9 Performance of thresholds, evolution of $BW_0/BW_1$ and Queue size in bottlneck in $BW_{Btlnck} = 1Mbps$

In the low $BW_{Btlnck}$ cases, the bandwidth can grow slowly. Before the queue is built-up, the newcomer bandwidth $BW_1$ will surpass $BW_0$. Further, since the bandwidth growth is limited below $2.175 * BW_0$, the queue size is not large enough to trigger our second threshold($Thres2$). This is also validated in Fig 7.9 and 7.10

The $Q_{Thres} = 300Kb$ lines are way above the $\gamma = 3$ lines for both $BW_{Btlnck} = 1Mbps$ case and $BW_{Btlnck} = 5Mbps$ case. It will also cause higher delay compared to $\gamma = 1$ of $Thres1$ even in $BW_{Btlnck} = 20Mbps$ cases. Hence it is logical to use the $Thres1$ as the constraint of $BW_1$ growth and bandwidth exploration.

When bandwidth is large enough, the growth of RTT will not reach the multiple of RTT on a reasonable time or before the loss ($QueueSize > 6MB$), even if $\gamma = 1$. In this range, $Thes2$ can take over the control of quit STARTUP constraint.

Now we look at each case individually. On the $BW_{Btlnck} = 1Mbps$ shown in Fig.7.9, when $\Delta RTT$ reaches $1 * RTT_{min}$ line, the time just passes less than two $RTT_{min}$. When $\Delta RTT >= 3 * RTT_{min}$ , $BW_1$ is more than twice the value of $BW_0$.

For $BW_{Btlnck} = 5Mbps$ case in Fig.7.10, the $Thres1 = RTT_{min}$ allows $BW_1$ to get close to $BW_0$ at about $3 * RTT_{min}$, while $Thres1 = RTT_{min}$ forced the $BW_1$ to be three times larger

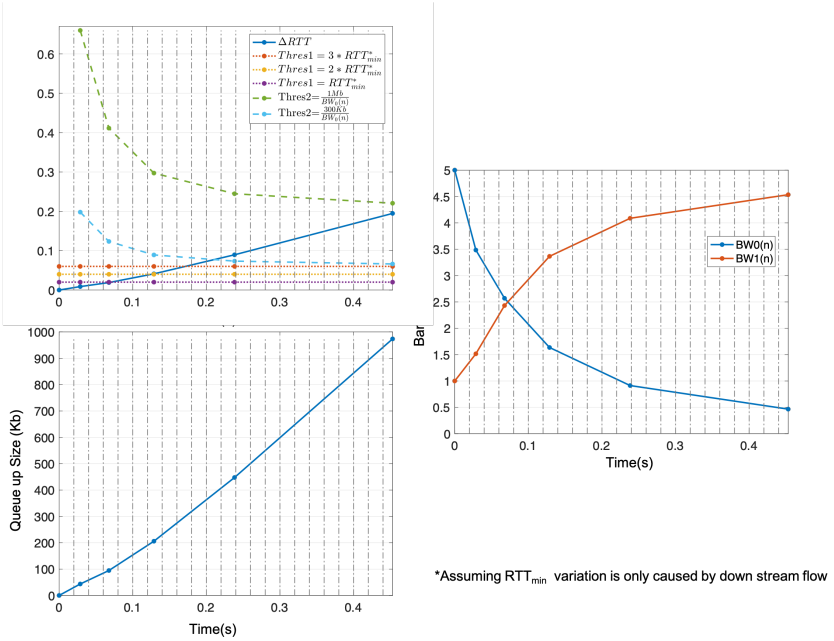



Fig. 7.10 Performance of thresholds, evolution of $BW_0/BW_1$ and Queue size in bottlneck in $BW_{Btlnck} = 5Mbps$

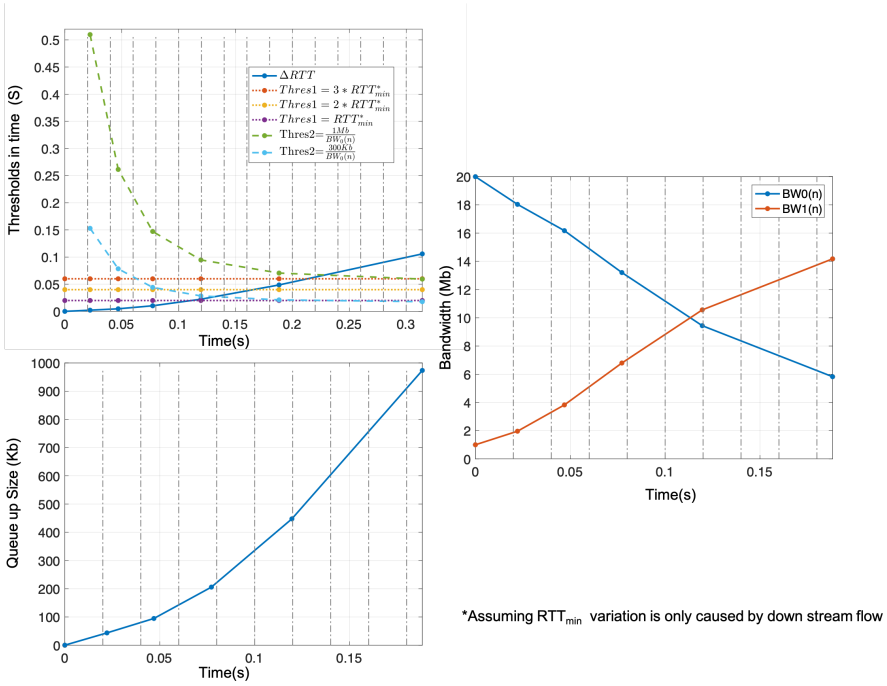



Fig. 7.11 Performance of thresholds, evolution of $BW_0/BW_1$ and Queue size in bottlneck in $BW_{Btlnck} = 20Mbps$

than $BW_0$. $Thres2$ with $Q_{Thres} = 300Kb$ has similar effect as $Thres1 = 3 * RTT_{min}$ in this case. The Queue size caused by $Thres1$ is roughly about 200Kb to 300 Kb while that in $BW_{Btlnck} = 5Mbps$ is less than 80Kb. Note that the the $Thres1 = 2 * RTT_{min}$ will make quit STARTUP to at about $7 * RTT_{min}$

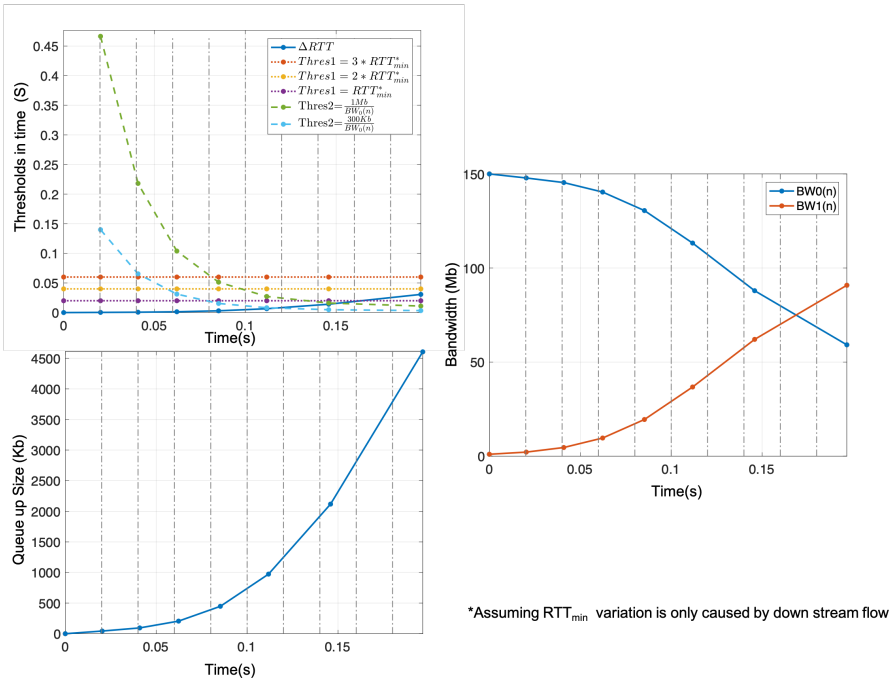



Fig. 7.12 Performance of thresholds, evolution of $BW_0/BW_1$ and Queue size in bottlneck in $BW_{Btlnck} = 150Mbps$

In $BW_{Btlnck} = 20Mbps$ case in Fig.7.11, the $Thres1 = RTT_{min}$ has similar performance to $Thres2$ with $Q_{Thres} = 300Kb$ which allows $BW_1$ to be slightly higher than $BW_0$ after about $6 * RTT_{min}$, while the effect of $Thres1 = 3 * RTT_{min}$ is similar to the effect of $Thres2$ with $Q_{Thres} = 1Mbb$. Which will make the quitting $BW_1$ to be more than twice of the value of $BW_0$. As for the $Thres1 = 2 * RTT_{min}$ will make quit STARTUP to at around $9 * RTT_{min}$ .

From $BW_{Btlnck} = 150Mbps$ case in Fig. 7.12, the performance of $Thres2$ with $Q_{Thres} = 1Mb$ is roughly equivalent to the performance of minimal $Thres1$ test case. When at the quitting is triggered, $BW_1$ is slightly lower than $BW0$ while the Queue size is about 2Mb. From 150Mbps and above $Thres1$ will no longer be effective.

As for the upper-bound of $BW_{Btlnck} = 500Mbps$, in this case, $Q_{Thres} = 1Mb$ and $Q_{Thres} = 300Kb$ will trigger the DRAIN after STARTUP roughly at about $6 * RTT_{min}$ and $8 * RTT_{min}$ respectively. Since at the high $BW_{Btlnck}$ range, the newcomer bandwidth $BW_1$ cannot surpass the bandwidth of current flow $BW0$, we should choose a relatively more aggressive $Q_{Thres}$ constant to make sure that $BW_1$ is close to $BW0$ while keep the Queue size on a relatively low level.

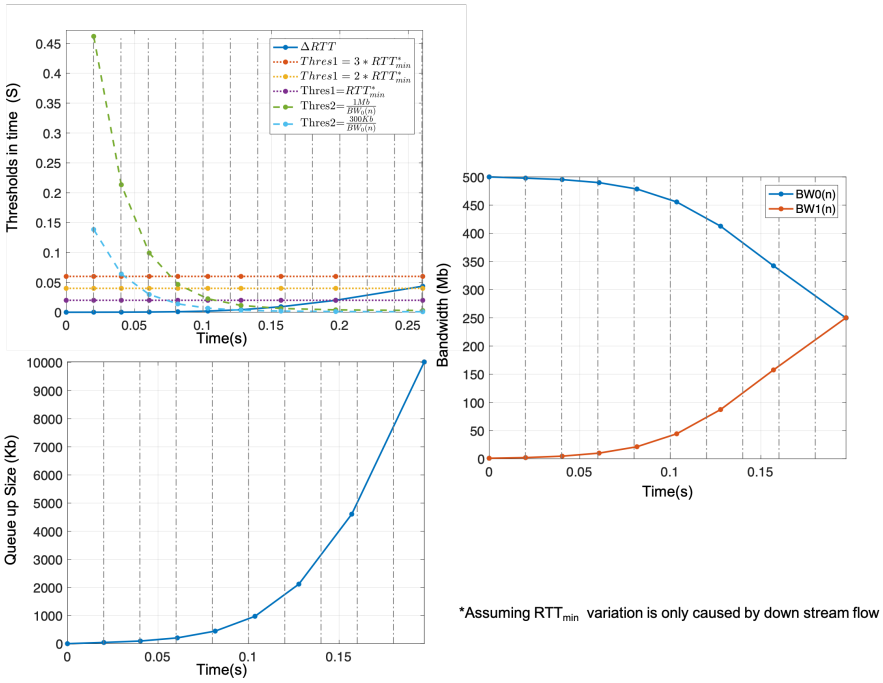



Fig. 7.13 Performance of thresholds, evolution of $BW_0/BW_1$ and Queue size in bottlneck in $BW_{Btlnck} = 500Mbps$

To sum up, our choice of $Q_{Thres}$ for high bottleneck bandwidth case. Taking both factors into consideration, we choose 1Mbps as our constant to calculate $Thres2$. As for the low bottleneck bandwidth case, we choose the medium value of $Thres1 = 2 * RTT_{min}$.

**DRAIN, DEEPDRAIN**

**1.** ***Normal DRAIN***

The DRAIN period is to housekeeping the extra amount of data transmitted to the pipeline by GROW or STARTUP state. Typically, DRAIN will reduce the Bytes in Flight to the maximum real-time BDP level for the past cycle by using a lower pacing gain ($G_{pacing} = 0.75$). The condition to quit DRAIN of $i^{th}$ cycle is:

$$BIF - 2 * Seg_{min} <= \begin{cases} 75\% * BDP_i^{max} & \text{if Deep Drain} \\ BDP_i^{max} & \text{else} \end{cases} \tag{7.23}$$

There is a Probe RTT state in BBR series of CCA once every 10 seconds or 2.5 seconds for BBRv1 and BBRv2 respectively. The former is more aggressive to reduce the BIF down to 4 Segments, and the latter has a 75% reduction coefficient for BDP.

**2.** ***DEEP DRAIN(DD)***

In our experiment, such design cannot guarantee the efficiency of convergence, and it may result in one new-comer flow with higher RTT to dominate the flow, or multiple new-comer

flows with higher RTT to dominate the pipeline and the buffer. This is also partly mentioned in [117]. Hence we design our DD cycle as follows:

1. When the RTT measurement cannot reach the minimum RTT for up to N Cycle or 2.5 seconds, which are targeting the lower and upper bound of RTT in $RTT_{min} \in [20, 300]$. The expectation of each full cycle is $(4+2)*20ms*5 = 600ms$ Seconds, where Deep DRAIN Limit $N_{DD} = 5$, in the lower bound case, t. For the higher limit of $4.5*300ms* 10 = 13.5s$ while in this range, the 2.5 seconds constraint will take over the control.

2. When it comes to DD, a factor of 75% is in use for the flows to reveal the minimum RTT of the route, and this period will last for two DRAIN state, where the STEADY(9) state between the two DRAINs will last for 1 $RTT_{min}$.

3. Condition to DD is asymmetric(ADD): When a flow is in DD state, a flow can quit DD and proceed to the following STEADY state when the received TIMESTAMP reports that RTT reaches the recorded minimum for five times. However, the counter will be reset to zero if the reported RTT down cross recorded $RTT_{min}$. Such a crossover of $RTT_{min}$ means current flow may be dominating the inflight pipeline, and it should not quit DD earlier. This method to make ADD is called Fair Quit Method (FQM).

4. During the sandwich STEADY between DDs, once the RTT reaches or crossover the recorded minimum, the second continuous DD will be skipped.

The dual continuous DD design is to differentiate DRAIN shape to avoid the RTT unfairness in BDP based congestion control.

**GROW**

The GROW state is to probe for the higher potential bandwidth with the cost of Growth of RTT, which can be eliminated by the following DRAIN state. In GROW state, the pacing gain is set to 125% to allow more packets in the pipeline to see whether there is more bandwidth available. The quitting condition of GROW state is:

$$BIF > \begin{cases} BIF + C & \text{if } BIF + C > 125\% BDP_i^{max} \\ 125\% BDP_i^{max} & \text{else} \end{cases} \tag{7.24}$$

where C is calculated as follows:

$$C = 50KBps * min(300ms, RTT_{min}) \tag{7.25}$$

When $BW_{Btlnck}$ increases, BIF will not shoot over the 125% threshold before the new available BDP pipeline is fully filled. The reason why we can apply this complement as the

growth compensation(GC) is that the minimum bandwidth in a Mobile network is 200KBps and the one forth of such value is 50KBps which can avoid the flow with RTT disadvantage to send enough packet to compete for a fair share of bandwidth. The effect of C will be validated in the following section.

**STEADY**

The STEADY state is a state between a DRAIN state and a GROW state. Once the server quit DRAIN, the server will stay in STEADY state with $G_{pacing} = 1$ for a random period of time. The range of this random period varies from 2 to 6 $RTT_{min}$. This is to avoid the global synchronization of all flow, which avoids several flows enters GROW at the same time. According to equation 7.17 and 7.18, synchronized GROW period cannot improve the fair share of the $BW_{Btlnck}$ network. Such random access avoids the collision of the GROWs from different flows and allows the flows with lower bandwidth to have a chance to converge faster. However, the converge depends not only on GROW state but also on the DD period since the new-comer flows rely on DD to $RTT_{min}$ and share the pipeline in a fair BDP, and allow the flows with BDP advantages to drain more aggressively.

The full life cycle of a CDBEv2 is illustrated in Fig 7.14. A simulated life cycle of two CDBEv2 flows will further be demonstrated in the following section.

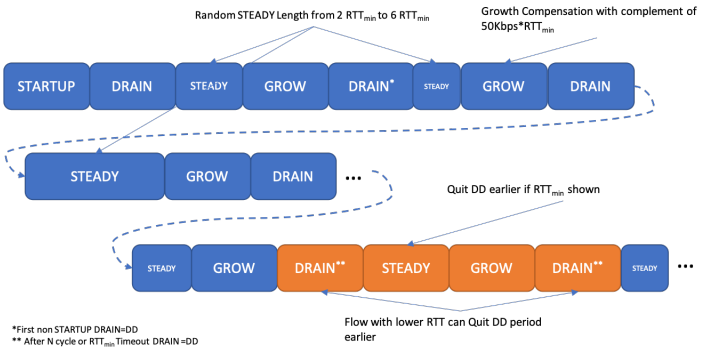



Fig. 7.14 State transition of CDBEv2 with features of each states

## 7.2.4 Simulation and analysis of CDBEv2

### Simulation configuration

In this section two simulation scenarios are configured:

- A Fixed network with a bottleneck configured from 1Mbps to 500Mbps with a six hop round trip where the RTT can be configured to vary from 20ms to 300ms. The buffer depth is set to 6MB, which is the typical value in a Mobile network. Such large buffer size is also the source of buffer bloat when the $BW_{Btlnck}$ is low as Algorithm.7.22 tell.

- A mobile network with exact the same infrastructure configuration as that in Chapter 6 while three mobility cases are different:

  1. four mobile devices standstill at the point which is $\sqrt{50+5(height)}$ metres apart from the BS
  2. four mobile devices moving at the same speed moving along the same trajectory shown in Fig.7.15
  3. four mobile devices randomly dropped around the base station and moving random direction with a constant speed of 3m/s

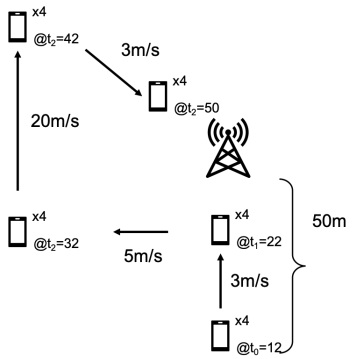



Fig. 7.15 Trajectory of moving mobile device for mobile simulation case 2.

### Analysis on Fixed network

### 1.Overall state transition

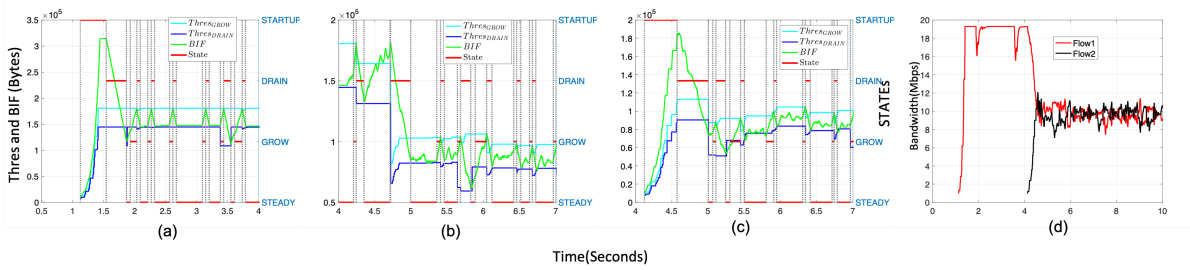



Fig. 7.16 State transition, BIF and thresholds variations for two flows in simulation. Left: Flow1 from 1s-4s;Middle:Flow1 from 4s-7s;Right: Flow2 from 4s-7s $RTT_{min}$ =60ms $BW_{Btlnck} = 20Mbps$

In this case, we analyze the two flow compete for in a fixed network with $BW_{Btlnck} = 20Mbps$. The first flow (F1) starts from 1st second and the second flow starts from 4th second.

As shown in Fig. 7.16, F1 probs the maximum available bandwidth for about 500ms and housekeeping the extra BIF queued up in BN buffer for about 300ms. For the following 2.5 seconds, normal GROW-DRAIN state pairs follow a STEADY state of random length. 2.5 seconds later, since no $RTT_{min}$ is revealed, A DD period is triggered, and the server DRAIN the BIF to 80% of current BDP. After the 4th second, F2 starts to probe for available bandwidth and the condition defined in Algo. 7.2 stops F2 STARTUP probe at a relatively fair. Meanwhile, the BW reported to F1 swiftly reflect the share of $BW_{Btlnck}$ and converge is able to achieve after the first DRAIN.

**2.multiple flows in a fixed network with varying BW**

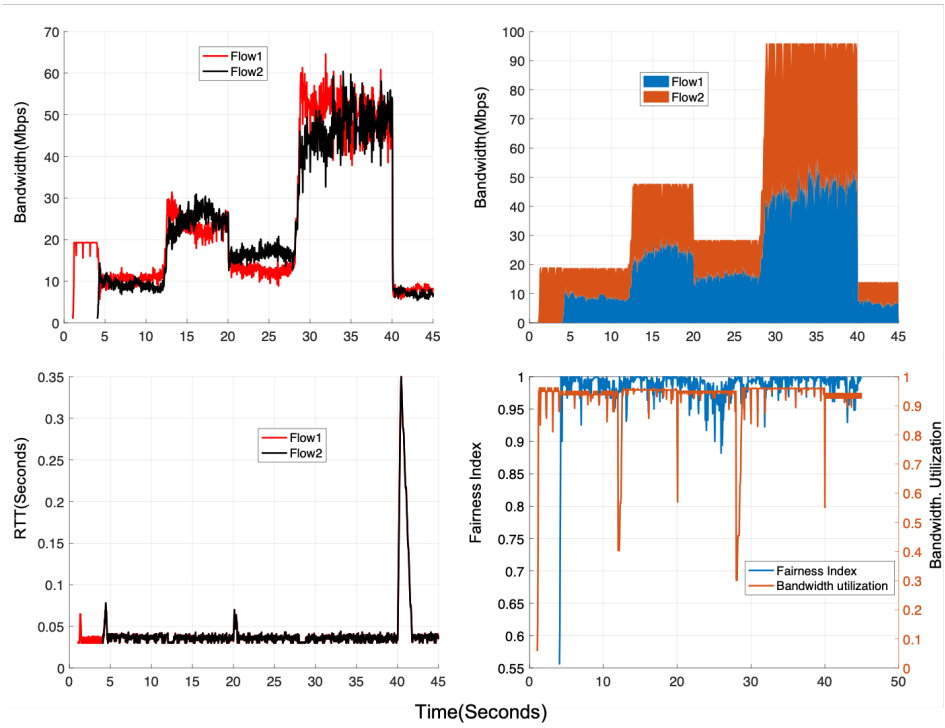



Fig. 7.17 Performance of complete CDBEv2 for varying $BW_{Btlnck}$, $RTT_{min}$ =24ms, 2 flows. Top left:Bandwidth variation; Top right:share of Bandwidth of two flow; bottom left:RTT variation; bottom right:BW utilization and Fairness of BW share;

In this simulation scenario, the bandwidth of BN is set to 20Mbps, 50Mbps, 30Mbps, 100Mbps, 15Mbps at the moment of 0, 12, 20, 28, 40 seconds respectively. The $RTT_{min}$ is set to 24ms. Flow 1 starts at 1 second, and the following flows start every 3 seconds later. As shown in both Fig.7.17 and Fig.7.18, the bandwidth can converge to the new $BW_{Btlnck}$ with little RTT jitters when $BW_{Btlnck}$ grows and limited amount of RTT growth when $BW_{Btlnck}$ decreases. When $BW_{Btlnck}$ varies enormously, the deep buffer and the BDP limited transmission manner

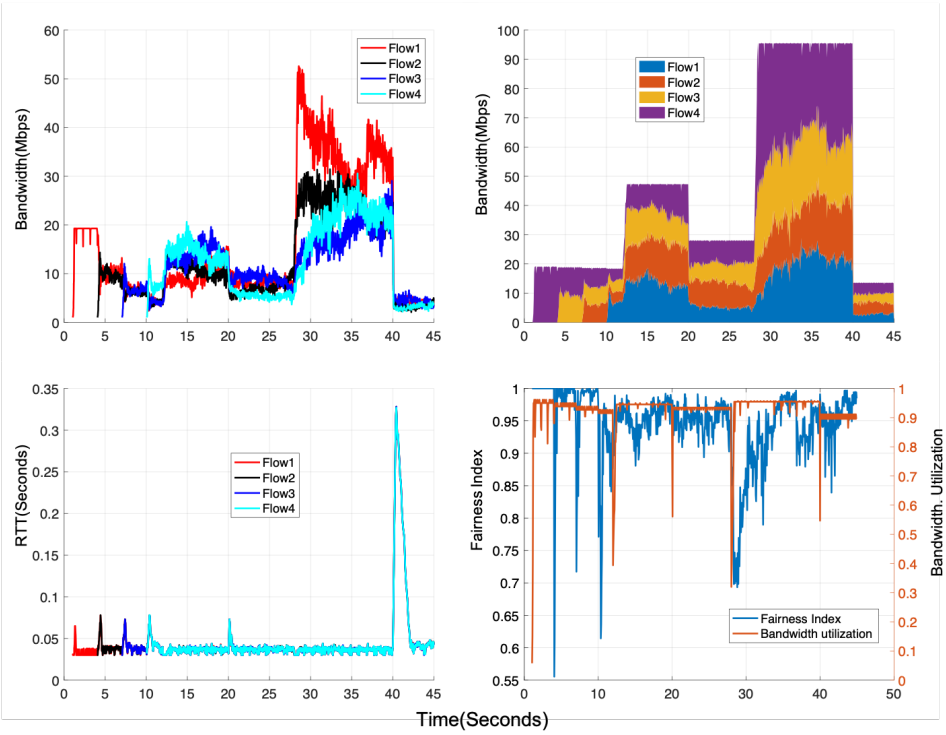



Fig. 7.18 Performance of complete CDBEv2 for varying $BW_{Btlnck}$, $RTT_{min}$ =24ms, 4 flows. Top left:Bandwidth variation; Top right:share of Bandwidth of two flow; bottom left:RTT variation; bottom right:BW utilization and Fairness of BW share

guarantees a low loss rate and the built-up Queue is digested in roughly a couple of seconds in both 2-flow and 4-flow cases.

As for the fairness and bandwidth utilization, the bottom right subplots of Fig.7.17 and Fig.7.18 shows that the bandwidth utilization is above 90% (Upstream ACK included in the same route). An acceptable period of underutilization appears ever since the bandwidth changes since it takes time to converge to the new bandwidth.

The fair share of $BW_{Btlnck}$ among flows can be seen on the top two subplots in Fig.7.17 and Fig.7.18. For the lower $BW_{Btlnck}$ value the convergence of fair share dynamic equilibrium is faster and more stable and for the larger $BW_{Btlnck}$(100Mbps), it takes some time to achieve a new equilibrium, because the BDP grows with the $BW_{Btlnck}$ and the flow which grows earlier can get the advantage in transmission. This unfairness is merged by the cooperation of random STEADY period, compensated GROW and asymmetric DD(ADD). To precisely measure the fairness, the bottom right subplot also prints the Jain's fairness [116] index. The conclusion is pretty much the same for CDBEv2:

- The higher the *BDP*, the more likely it will be unfair in BW share,

- The more coexisting flows, the more likely it will be unfair in BW share
- The two factors above determines the average time for CDBEv2 to converge in a Fixed network with typical BW and RTT of Mobile network

**3.Evaluation of GC and DD configuration**

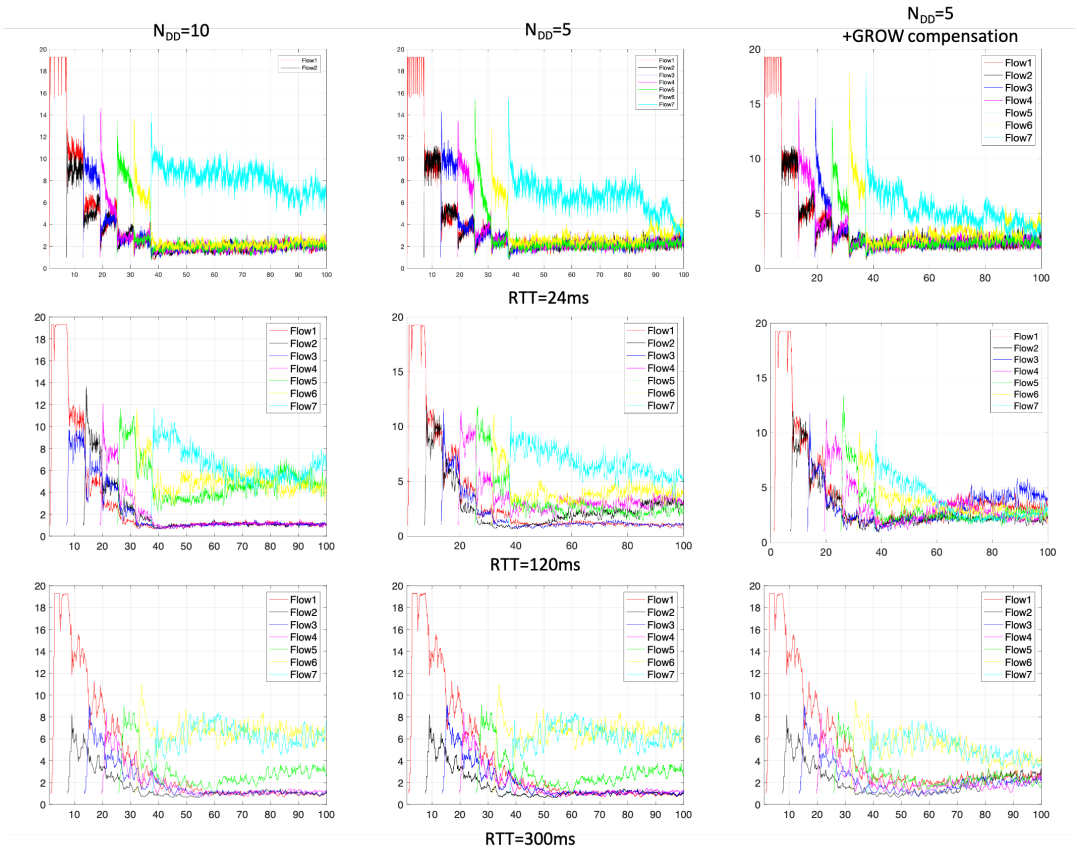



Fig. 7.19 Bandwidth performance in Fixed network with different configurations. Left column: Deep Drain Limit is 10. Middle column: Deep Drain Limit is 5. Right column: Deep Drain Limit is 5 with Growth compensation. $RTT_{min}$ =24ms.

In this scenario, seven flows are initiated with 6 seconds interval. In Fig.7.19, the configuration of different $N_{DD}$ values. For the left column, we can see the simply a $N_{DD} = 10$ DD method cannot relax the congestion it made, and the RTT in $RTT_{min} = 24ms$ case keeps on growing in the Lower BW cases. On the one hand, Such congestion is more and more severe if the traffic made by BDP based congestion control is saturating the link, and the new-comer traffic is not able to overshoot the buffer. On the other hand, the new-comer with higher RTT record will dominate the pipeline and Queue which will make the Flows with lower RTT record to remain on using a low share of minimum BW, and the 125% of such value is not enough to raise the share of starving flows.

To deal with the two problems mentioned above, we propose to use a smaller $N_{DD}$ value: $N_{DD} = 5$and a GROW compensation method as described in 9 The result of implementing these methods shows in the right column of Fig.7.19. When only $N_{DD} = 5$ takes effect, the low RTT starts to go lower. However, the trend of convergence is not clear, and in $RTT_{min} = 300ms$ case, the lower BW share flows are still suppressed by large RTT flows. When GROW compensation is enabled, all the tested saturated transmission cases with different $RTT_{min}$ converges. Since

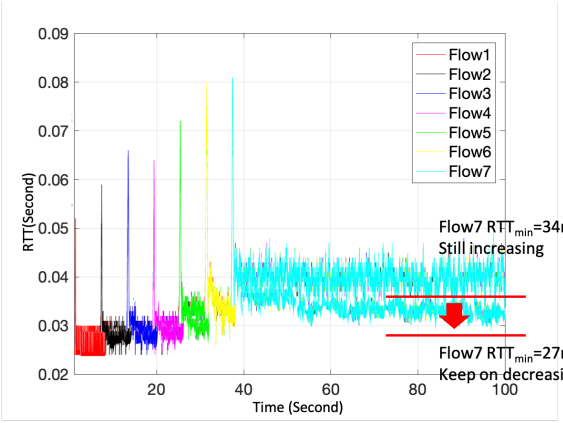



Fig. 7.20 RTT performance evolution in Fixed network 7 flows.$RTT_{min}$ =24ms.

the low RTT cases are shows the most severe congestion problem, even in this case, as shown in Fig.7.20, the trend of $RTT_{min}$ is reversed from growing to decreasing, and the recorded $RTT_{min}$, after the seventh flow comes in, degrades from 34ms to 27ms, which is 25% lower BIF for overflowing flow. Combination of the two methods offer the underdog flows a better chance to grab more BW share.

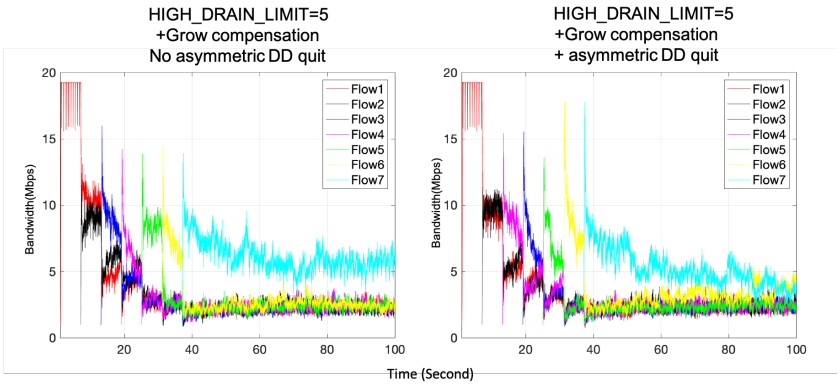



Fig. 7.21 Bandwidth performance in Fixed network:
Left column: DDL=5 + GC. Right column: DDL=5 + GC + FQM. $RTT_{min}$ =24ms.

The results in the right column of Fig.7.19 and in Fig.7.20 should also thank the ADD quitting feature. Without ADD, even though the flows are trying to probe minimum RTT, the

BW superior flows and disadvantaged flows are making the same proportion of BIF. Since the entrance of DD is approximately synchronized, the share of BIF will no be re-allocated according to equation 7.17 and 7.18. In contrast, ADD enabled advantageous flows can release more BDP and Queue share as expected. Last but not least, to make sure that the dual

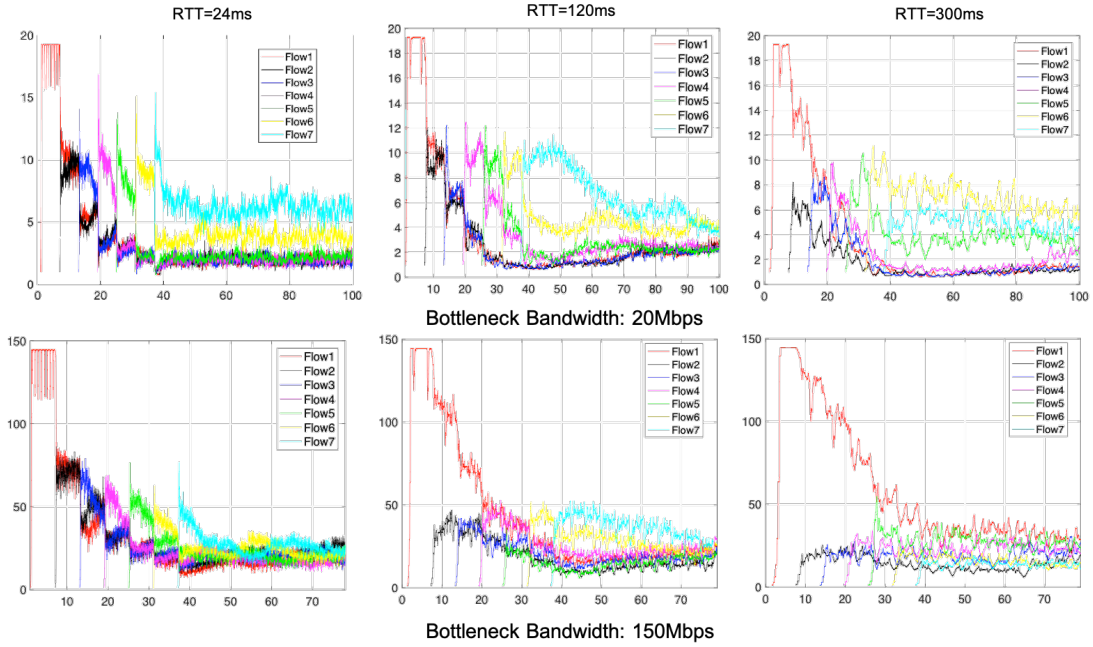



Fig. 7.22 Bandwidth performance in Fixed network:
First row: $BW_{Btlnck} = 20Mbps$. Second row: $BW_{Btlnck} = 150Mbps$. $RTT_{min}$ =24ms, 120ms and 300ms for 1st, second and 3rd column, respectively.

continuous DD is necessary, the BW performance of GC + ADD enabled, but no secondary continuous DD case is plotted in Fig. 7.21 . As we can see on the top row, compared to continuous DD configuration, the single DD case cannot achieve the same convergence speed and RTT reduction effect as Dual DD method.

**4.Performance of CDBEv2 on 2,4,7 flows in Fixed network**

For the full feature enabled CDBEv2, we present all the combination of $RTT_{min} \in 24ms, 120ms, 300ms$ in $BW_{Btlnck} \in 20Mbps, 150Mbps$ to validate the performance in low and high $BW_{Btlnck}$ indicated in STARTUP modeling section.

In a two flow case, the BW fair share can be achieved right after the STARTUP of second flow, and the RTT seen on the server-side is stable on no more than 35ms. For a more congested four flow cases, the fair BW share convergence happens in a few seconds while their RTT is also dynamically stabled on a low level.

For the saturated scenarios of 7 flows contention case in low $BW_{Btlnck}$ when $RTT_{min} =$ is also low, the BDP based congestion control with inappropriate BIF management state can

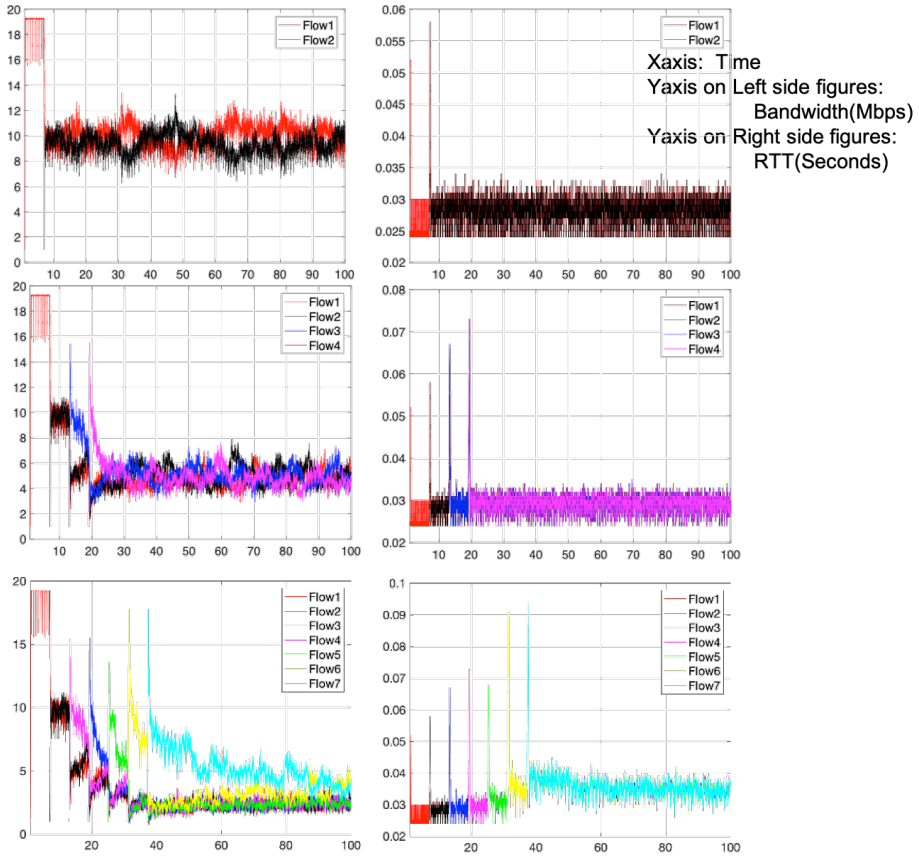



Fig. 7.23 Performance of complete CDBEv2 for $BW_{Btlnck}$ =20Mbps, $RTT_{min}$ =24ms

cause the unfairness issue as described earlier. With our Dual DD + ADD+GC method, the advantageous flows can consciously give way once a smaller $RTT_{min}$ is discovered by the DD method. The overall trend is toward $RTT$ fall back to the minimum, and BW converges toward fair share.

For $RTT_{min} = 120ms$ case in Fig.7.24, despite the two flow converge case is less stable compared to the low RTT case, the BW share is still fair overall, and the $RTT_{min}$ is also revealed regularly by the DD method. For the four flow cases, the situation is almost the same, a little bit more variation compared to low $RTT_{min}$ scenarios. The RTT can still regularly fall back to $RTT_{min}$. In saturated seven flow cases on the bottom of Fig.7.24, the convergence speed, in this case, is faster since a longer pipe BDP dilutes the new-comer portion of BIF so that the over transmission due to the high BW feedback from higher minimum RTT record is partly mitigated.

For the highest $RTT_{min}$ case in Fig.7.25, the overall convergence speed is even slower due to the following reasons: Larger BIF pipe alleviates the effect of packets sent from new-comer

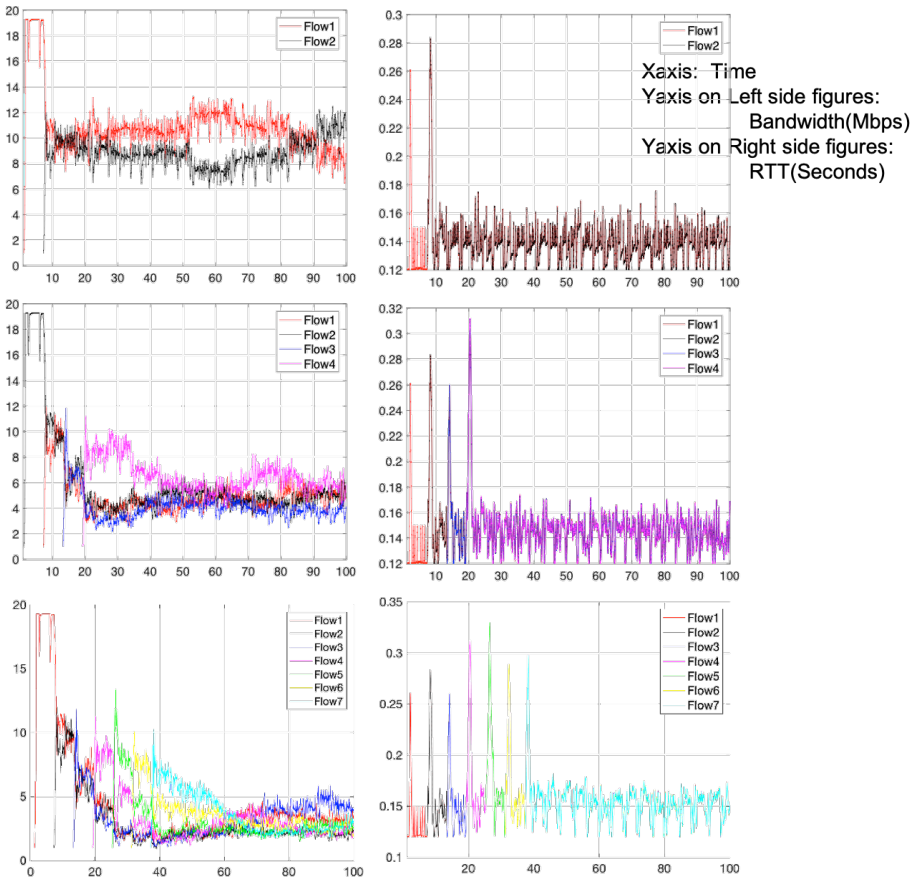



Fig. 7.24 Performance of complete CDBEv2 for $BW_{Btlnck}$ =20Mbps, $RTT_{min}$ =120ms

flows from STARTUP so that the convergence is achieved by the cooperation of Randomized STEADY, GROW and DD states. Also, the RTT can fall back to the minimum regularly even on seven flows case. The convergence of BW can on seven flow cases has lower unfairness since the portion of data from the new-comer for buffer schedule is lower than that in lower $RTT_{min}$ cases. Such a phenomenon is more remarkable in $BW_{Btlnck}$ =150Mbps, $RTT_{min}$ =300ms case shown in Fig. 7.32 attached in Conclusion and future work. However, still, the longer convergence time consumption trend is the same for seven-flow cases. Note that there is a loss that happens for the first flow. Similar trends can also be found in 150Mbps cases. The Figure for 150Mbps cases are listed at the end of this chapter as Conclusion and future work(Fig.7.30,7.31,7.32).

**5.Performance of CDBEv2 in Mobile network compare with BBR and Cubic**

In the Mobile network, a single Basestation and four mobile User Devices(UEs). The network configurations are the same as that in Section 6.5 in Chapter 6. In the first scenarios, all the UEs are standing still on the 50 metres south away from the BS.

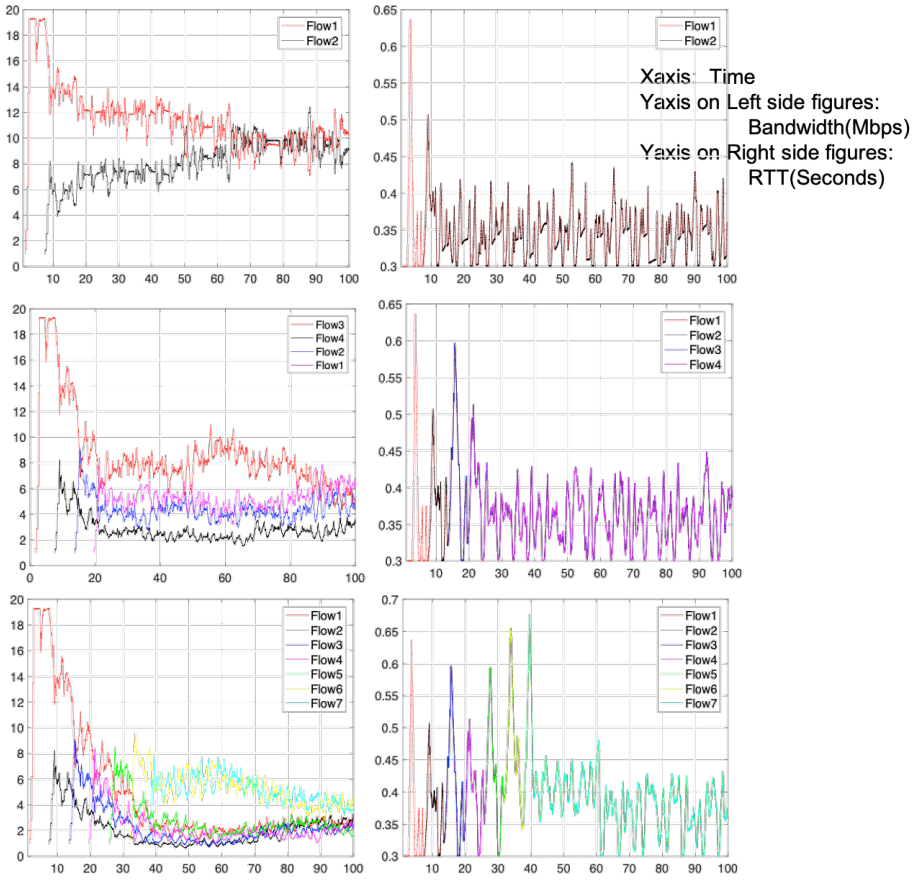



Fig. 7.25 Performance of complete CDBEv2 for $BW_{Btlnck}$ =20Mbps, $RTT_{min}$ =300ms

As shown in Fig.7.26, It seems that CUBIC has the highest Receiving Data Rate samples while BBR and CDBEv2 have similar Receiving Data Rate samples on the Client side. Since CUBIC constantly shoot over the buffer, its RTT is the highest. As for RTT samples in BBR is a little bit higher, especially on the period when new-comer flows are starting up.

A similar trend is shown in moving on trajectory case in Fig.7.27. In this simulation scenario, all the UEs are located at the same initial position. Once after all the flows are transmitting after STARTUP state, all the four UEs move along the same trajectory shown in Fig.7.15. Hence in the first two simulation scenarios, four UEs has exact the same radio condition. Hence we can check the fairness performance of different CCAs. The average goodput RTT and fairness are given in Fig.7.28. The bar chart shows that CDBEv2, on average, has the highest goodput, lowest RTTand best fairness index on both mobility cases and both RTT and BW among four Flows.

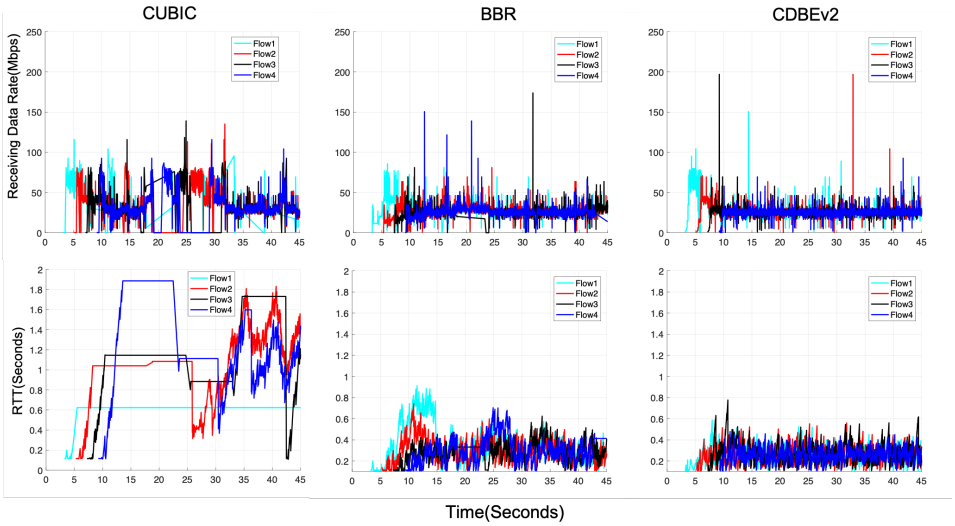



Fig. 7.26 Receiving Data Rate(above) and RTT(bottom) performance of CUBIC, BBR, CDBEv2 in LTE network: constant position

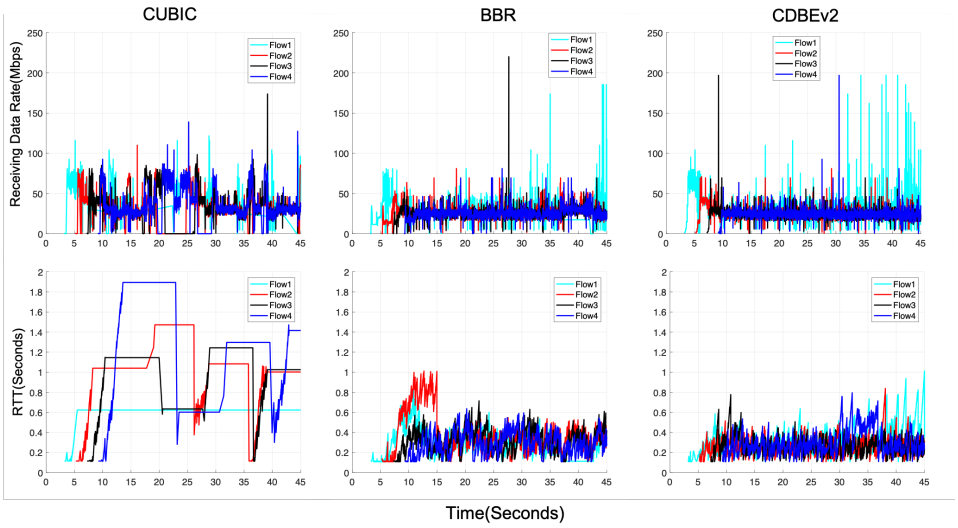



Fig. 7.27 Receiving Data Rate(above) and RTT(bottom) performance of CUBIC, BBR, CDBEv2 in LTE network: predefined trajectory

The last mobile simulation scenario is 40 sets of random initial position and random walking direction at the moving speed of 3m/s. Since the radio condition is different for each UE, the fair share of the radio resource is fully managed by the schedulers in BS. The result is statistically shown in the boxplots in Fig.7.29. The upper and lower bound of the box is 75% and 25% percentile of the data set, and the red cross beyond the 99.3% whiskers are the samples with an extreme case. The green diamond is the average of data, and the red line in the box is the median. Statistically, CDBEv2 has about 5% improvement on meanwhile a lower jitter

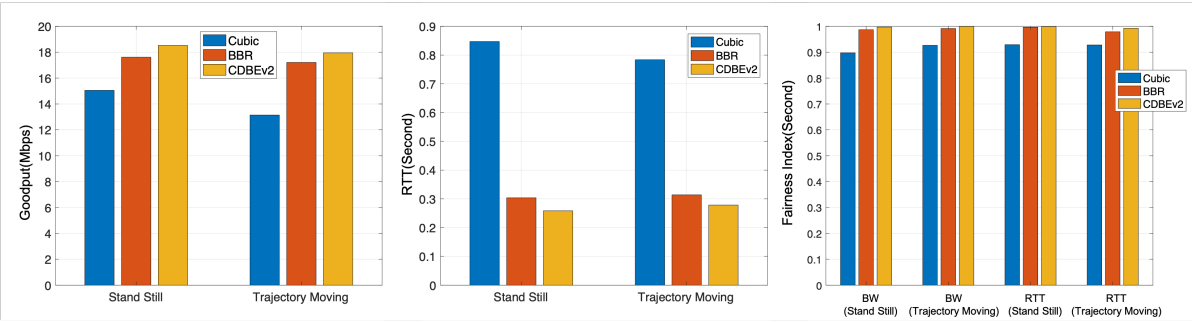



Fig. 7.28 Average Goodput(left), RTT(middle) and FairnessIndx(right) performance for Stand still and Trajectory moving cases

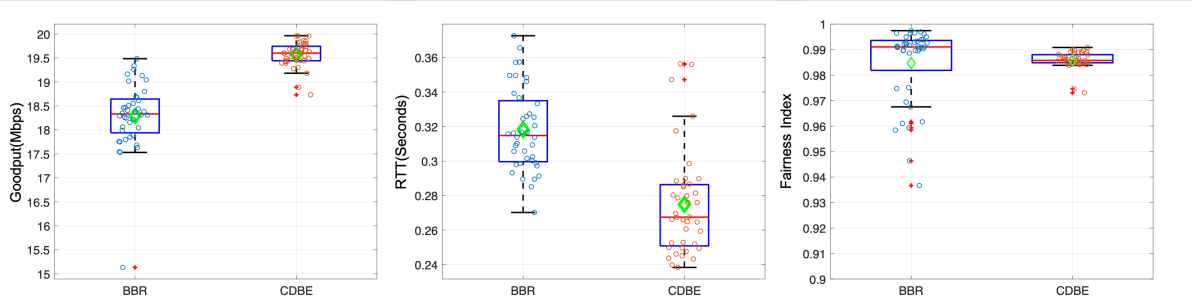



Fig. 7.29 Statistics of Goodput(left) , RTT(middle) and fairness(right)t for BBR, CDBEv2 in LTE network: 40 sets of different random initial location and random trajectory

on Goodput performance. Concerning RTT the 18.5% less RTT, on average, is observed on CDBEv2 compared to BBR. Last but not least, CDBEv2 can achieve slightly better fairness on average among 40sets of simulation.

## 7.3 Conclusion and future work

To address the un-converge problem of CDBE in the 4G mobile network, CDBEv2 is proposed in this chapter. This algorithm can take care of both bottlenecks in mobile radio link in Base station and the fixed-mobile backhaul as well. The two theoretical models are applied to analyse the STARTUP performance on both single flow and multiple/two flow. The configuration and quitting conditions are designed according to the theoretical analysis. Note that the analysis is based on the average statistics we find from existing research. For a different type of network, the two proposed formula can be customised to achieve an equilibrium with a low delay according to the traffic statistics or the radio access network configuration. The technical details of the state machine of CDBEv2 design are validated on 5 different types of simulation:

- Scenario 1 is the combination of different value of fixed bottleneck bandwidth and RTT,

- Second scenario is several flows sharing the time-varying bottleneck bandwidth, which is to see the convergence performance on a controlled dynamic environment.
- Scenario 3 is to compare the CUBIC BBR and CDBE v2 under the same fixed position mobile scenario, and
- Scenario 4 us ti test their performance difference under a specific trajectory. And
- Finally, in the fifth scenario, the statistic result on Goodput, delay and fairness of baseline and proposals of the random initial position, random walking UE scenarios are collected.

The simulations on both Fixed network and Mobile network on several different cases show that CDBEv2 can surpass the performance of CUBIC and BBR on Goodput, RTT and fairness.

The future work is to design and improve the coexistence capability of CDBEv2. To deploy the CDBE in an Orange real-world network can be one possible direction to further validate our design. For the analysis part, the proposed formula has the potential to improve

## Appendix

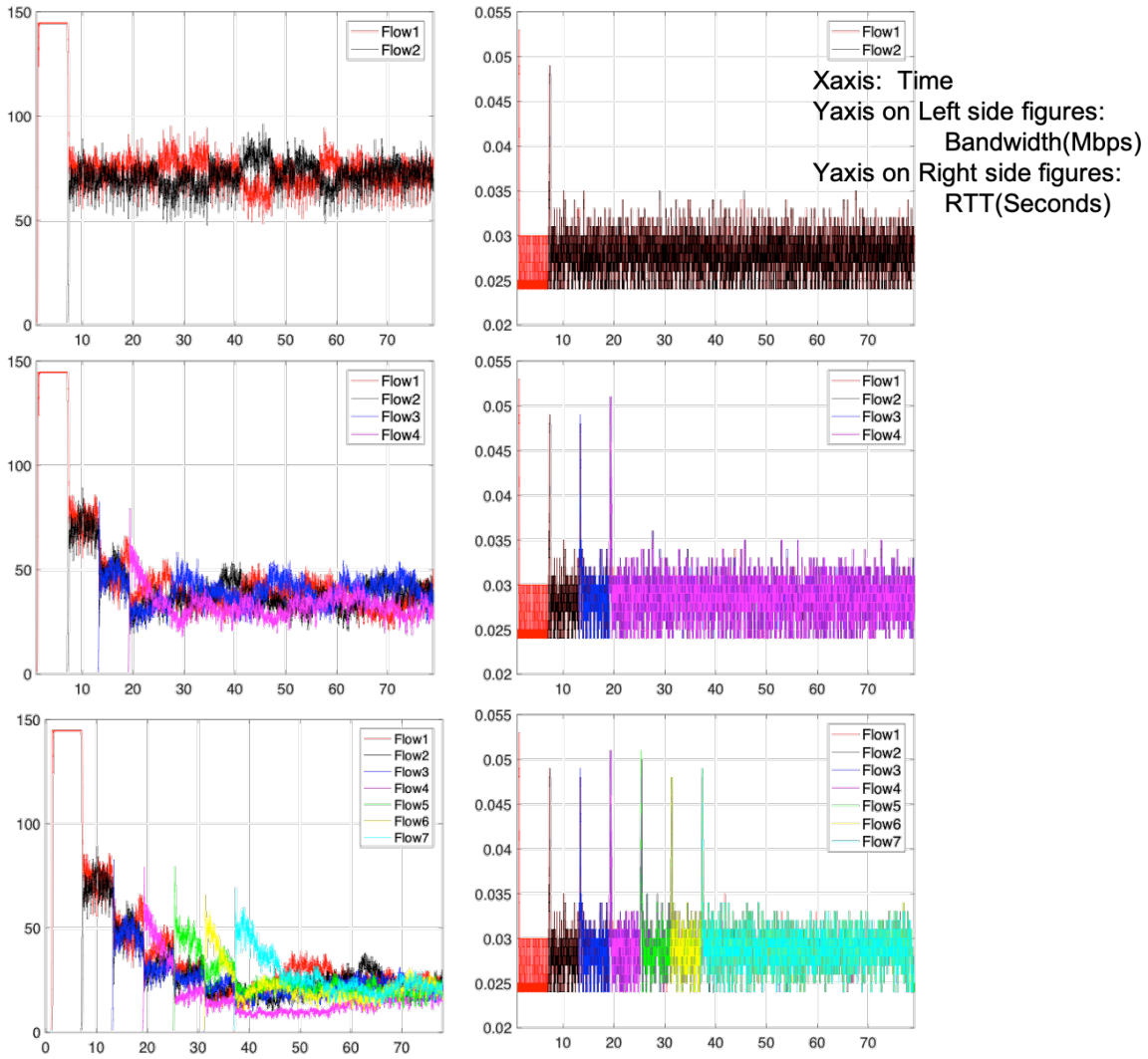



Fig. 7.30 Performance of complete CDBEv2 for $BW_{Btlnck}$ =150Mbps, $RTT_{min}$ =24ms

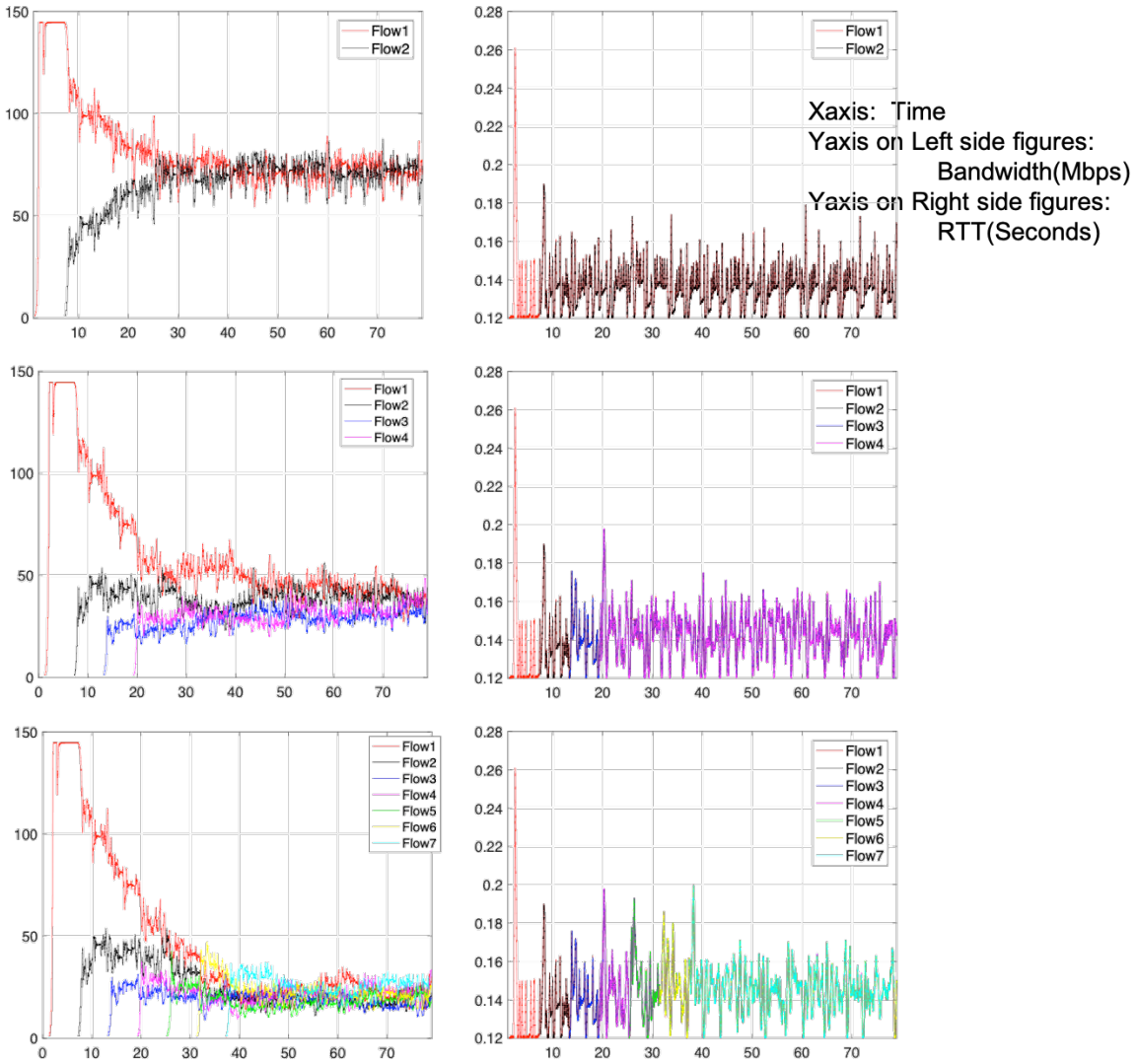



Fig. 7.31 Performance of complete CDBEv2 for $BW_{Btlnck}$ =150Mbps, $RTT_{min}$ =120ms

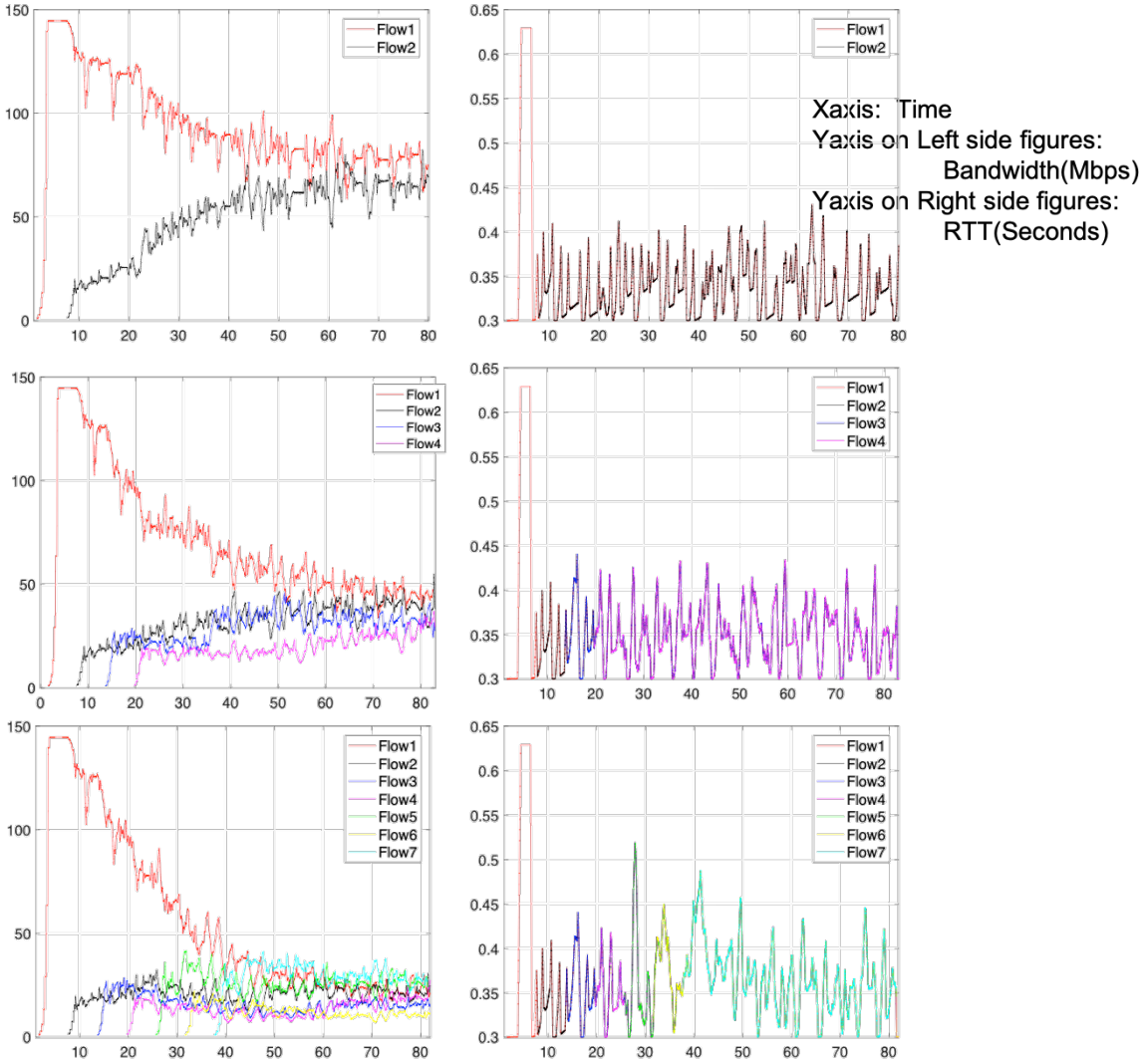



Fig. 7.32 Performance of complete CDBEv2 for $BW_{Btlnck}$ =150Mbps, $RTT_{min}$ =300ms

# Chapter 8

# Conclusion and Future work

## 8.1 Thesis conclusion

In this thesis, we explored the possibility of congestion control in the mobile network. Firstly we improved the LTE modules in NS3 simulator to a more realistic simulation scenario. Secondly, we transplanted 3G-CQI oriented cross-layer Congestion control, and BBR congestion control in NS3 simulator and evaluated their performance against conventional TCP CCAs. Based on the tradeoffs and goals proposed in the first three chapters, We proposed a client-driven bandwidth-estimation congestion control algorithm. Last but not least, we proposed an improvement on CDBE to address:

- The low BW utilisation problem caused by fast varying bandwidth
- Address the bufferbloat caused by low capacity and large buffer size mismatching, and
- To cover the hybrid bottleneck possibility to make in a mobile edge network.

Following the methods and objectives in the introduction, we have achieved the following goals in several chapters:

1. review the state of the art of mobile cellular network
2. review the literature of existing congestion control strategies
3. transplant the origin CCA, CQIC from 3G Context to 4G context for NS3 simulation and identified its pros and cons
4. implement BBR for NS3 simulation and identified its pros and cons

5. Designed the prototype of Client Driven Bandwidth Estimation architecture for congestion control in mobile network.

In Chapter 3, the characteristics of mobile network PHY/MAC layer is reviewed. This chapter also pointed out the essential features which should be taken into consideration when designing a congestion control algorithm. Besides, the state of the art of related algorithms is also summarised in this part: the solutions to improve CCA performance is classified by four crossing factors: End-to-End, Cross-layer, middlebox and conventional TCP solutions. The state of the art review and the following research of CQIC in Chapter 5 is the starting point of this thesis, proves that the single lower layer oriented, or a single type of bottleneck oriented congestion control algorithm can be harmful to the overall performance of the network.

Chapter 4, further discuss the details of the congestion control algorithm. The configurations and the traffic equilibrium from a network perspective are analysed and summarised: necessary models, tools, technical details of loss-based congestion control and traffic on an SPGW or a bottleneck. First of all, we clarify the relationship between bufferbloat effect and the AIMD loss-based congestion control algorithms. Secondly, we discuss duplicated-ACK and Acknowledgement noise filtering. Thirdly, the reason for using pacing is explained in the context of queuing theory. Last but not least, a TCP trace guided design tool is also proposed to guide the design of our CCA algorithm.

Despite the weakness identified latter in the thesis of CQIC, In Chapter 5, we present the implementation, test and measurement of CQIC in NS3 platform for LTE. The implementation of CQIC in LTE is different from a 3G network. Original CQIC uses CQI mapping for UE to match the radio link bandwidth. In our experiment, DCI distributed from eNB is used for bandwidth calculation. The use of this DCI can take queuing state, traffic prediction and all the other factors on RLC and PDCP layer into consideration. Though in the publication we call this proposal TCP-CQIC-LTE, here in the thesis, we correct the name to be DCIC. The result shows that DCIC can provide lower loss rate, lower RTT, and higher throughput compares to Westwood and Cubic.

To compare the performance of proposed DCIC and design the full route oriented congestion control algorithm, we further implemented the latest BBR algorithm in NS3 platform in Chapter 5. The tool proposed in Chapter 4 is also used to analyse the operation of BBR CCA. This part also identifies the problems of CQIC and BBR. Based on this research, we start to propose our own solution to the problem.

In Chapter 6, we described, implemented, and evaluated the proposed CCA named Client-Driven-Bandwidth-Estimation(CDBE) Congestion control. The idea combines the DCIC and BBR, which provides the resistance to a route change, bandwidth variation, and asymmetric routing. The essential nature of CDBE is simulated and validated in a wired network. Compared

to BBR and CUBIC, the overall performance of CDBE is validated in the simulation of a mobile network.

Last but not least, an improvement of CDBE, named CDBEv2, is proposed. Two models are proposed for the STARTUP state to configure the parameters. Based on the analysis, we designed the CDBEv2 state machine and the features of the new feedback structure. The simulation on five different scenarios is used to validate the performance of CDBEv2: two fixed network scenarios, and three mobile scenarios. The five types of simulations validate the capability of fair share and high bandwidth utilisation of CDBEv2. CDBEv2 can outperform BBR and Cubic in the mobile network in bandwidth RTT and fairness.

Technically, the five metrics of evaluating a CCA in this thesis are bandwidth utilisation, end-to-end delay/loss, protocol simplicity, intra-protocol fairness and the genericity of the algorithm. In chapter 3 and 4, we define the goal and tradeoff metrics from layered and network perspective of view, respectively. Evidence in Chapter 5 proof that CDBE improves the genericity, utilisation, delay simplicity compared to CQIC and the fairness and simplicity is further enhanced in CDBEv2.

Generally speaking, the purpose of congestion control is not to fight until the only winner dominates the arena. It is to design a distributed traffic management system to reach an overall harmony while maintaining the capability to identify, react and self-defence with the possibility of wild, aggressive and fast varying environment. The experiment, analysis practised in this thesis is a meaningful explore toward this goal.

## 8.2 Perspectives and future direction

Looking ahead, we believe that several exciting new research directions remain to explore. Firstly, the configuration method proposed in chapter 7 can be customised to adapt the real networks with different traffic statistics. CDBEv2 with such customised parameters can be further implemented in a real system to discover the challenges in the real world. For example,

- to see whether the proposed formula and analysis fits the real traffic statistics and the network configuration,
- to see whether the ACK loss rate or the ACK delay rate will affect the performance of CDBE.

Additionally, we should also address the coexistence between conventional TCP and CDBEv2 if the applied network does not fit our *isolated mobile network*assumption. For example, the classic delay-based Vegas algorithm can be too polite in network resource competition. In such

case, BBR and the further version is trying to be more aggressive when it is trying to survive in the network full of other wild network congestion control algorithm and such design cause unfairness between BBR and CUBIC, and this manner is improved on BBRv2. Thirdly, the merge of CDBE and BBR can be a point to get a guaranteed Dul-bandwidth estimation system which allows the server to get more information about the network status.

CDBE user-side bandwidth-estimation algorithm can be improved to a per-packet basis. What is more, the client-side bandwidth estimation can also be reviewed to adapt to various types of network, include not only the 4G network but also the 5G and or IoT scenario. We should also identify the performance of CDBE in a real-world environment, where the bandwidth variation, queuing management, traffic variation is more practical than the simulator.

Last but not least, though the 5G network is the next generation of mobile network, the underlying architecture and protocol stack has a similar structure. By installing the server-side logic in the mobile gateway, we can implement CDBEv2 in 4G and 5G network. For client-side logic, it can be achieved by adapting the bandwidth estimation module in QUIC protocol for the next phase.

# References


[1] 3GPP, "Evolved Universal Terrestrial Radio Access (E-UTRA); Radio Resource Control (RRC); Protocol specification," Technical Specifications and Technical Reports for a GERAN-based 3GPP system, Technical Specification (TS), 97. [Online]. Available: https://www.3gpp.org/technologies/keywords-acronyms/102-gprs-edge

[2] *Evolved Universal Terrestrial Radio Access (E-UTRA); Physical layer procedures*, 3GPP Std. [Online]. Available: https://www.3gpp.org/technologies/keywords-acronyms/98-lte

[3] *Evolved Universal Terrestrial Radio Access (E-UTRA); Physical layer procedures*, 3GPP Std. [Online]. Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=2427

[4] E. G. Larsson, O. Edfors, F. Tufvesson, and T. L. Marzetta, "Massive mimo for next generation wireless systems," *IEEE Communications Magazine*, vol. 52, no. 2, pp. 186–195, 2014.

[5] "Industry specification group non ip networking," https://www.etsi.org/technologies/non-ip-networking, accessed: 2020-04-10.

[6] "Ns3 simulator." [Online]. Available: https://www.nsnam.org/

[7] E. Metsälä and J. Salmelin, *LTE Backhaul: Planning and Optimization*. Wiley, 2015. [Online]. Available: https://books.google.fr/books?id=adCzrQEACAAJ

[8] B. Sklar, *Digital Communications: Fundamentals and Applications*, ser. Prentice Hall Communications Engineering and Emerging Techno. Prentice-Hall PTR, 2001. [Online]. Available: https://books.google.fr/books?id=Bh4fAQAAIAAJ

[9] H. Balakrishnan, S. Seshan, and R. H. Katz, "Improving reliable transport and handoff performance in cellular wireless networks," *Wirel. Netw.*, vol. 1, no. 4, pp. 469–481, Dec. 1995. [Online]. Available: http://dx.doi.org/10.1007/BF01985757

[10] H. Balakrishnan, V. N. Padmanabhan, S. Seshan, and R. H. Katz, "A comparison of mechanisms for improving tcp performance over wireless links," *SIGCOMM Comput. Commun. Rev.*, vol. 26, no. 4, pp. 256–269, Aug. 1996. [Online]. Available: http://doi.acm.org/10.1145/248157.248179

[11] M. Gast, *802.11ac: A Survival Guide*, 1st ed. O'Reilly Media, Inc., 2013.

[12] "Wi-fi alliance," https://www.wi-fi.org/.

## References


[13] L. Khoukhi, A. El Masri, A. Sardouk, A. Hafid, and D. Gaiti, "Toward fuzzy traffic adaptation solution in wireless mesh networks," *IEEE Transactions on computers*, vol. 63, no. 5, pp. 1296–1308, 2012.

[14] L. Khoukhi and S. Cherkaoui, "Fuzzyccg: A fuzzy logic qos approach for congestiosn control in wireless ad hoc networks," in *Proceedings of the 1st ACM international workshop on Quality of service & security in wireless and mobile networks*, 2005, pp. 105–111.

[15] A. El Masri, A. Sardouk, L. Khoukhi, A. Hafid, and D. Gaiti, "Neighborhood-aware and overhead-free congestion control for ieee 802.11 wireless mesh networks," *IEEE Transactions on Wireless Communications*, vol. 13, no. 10, pp. 5878–5892, 2014.

[16] G. Miao, J. Zander, K. W. Sung, and S. Ben Slimane, *Fundamentals of Mobile Data Networks*. Cambridge University Press, 2016.

[17] W. Diego, I. Hamchaoui, and F. Guillemin, "Slomo: An implicit cross-layer mechanism for a better experience on mobile networks," in *2016 IEEE International Conference on Communications (ICC)*, 2017.

[18] I. Hamchaoui, W. Diego, and S. Jobert, "Ip centric qos model for mobile networks — packet based qos management for intra-bearer arrangements," in *2014 IEEE Wireless Communications and Networking Conference (WCNC)*, 2014, pp. 2653–2658.

[19] W. Diego, I. Hamchaoui, and X. Lagrange, "Cross-layer design and performance evaluation for ip-centric qos model in lte-epc networks," in *2015 8th IFIP Wireless and Mobile Networking Conference (WMNC)*, 2015, pp. 88–95.

[20] B. D. Van Veen and K. M. Buckley, "Beamforming: a versatile approach to spatial filtering," *IEEE ASSP Magazine*, vol. 5, no. 2, pp. 4–24, 1988.

[21] P. Benko, G. Malicsko, and A. Veres, "A large-scale, passive analysis of end-to-end tcp performance over gprs," in *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, vol. 3, March 2004, pp. 1882–1892 vol.3.

[22] R. Chakravorty and I. Pratt, "Performance issues with general packet radio service," *Journal of Communications and Networks*, vol. 4, no. 4, pp. 266–281, Dec 2002.

[23] K. Liu and J. Y. B. Lee, "On improving tcp performance over mobile data networks," *IEEE Transactions on Mobile Computing*, vol. 15, no. 10, pp. 2522–2536, Oct 2016.

[24] S. Parkvall, E. Englund, P. Malm, T. Hedberg, M. Persson, and J. Peisa, "Wcdma evolved-high-speed packet-data services," *Ericsson Review*, vol. 2, pp. 56–65, 2003.

[25] C. Cox, *An Introduction to LTE: LTE, LTE-Advanced, SAE and 4G Mobile Communications*, 1st ed. Wiley Publishing, 2012.

[26] J. Huang, F. Qian, Y. Guo, Y. Zhou, Q. Xu, Z. M. Mao, S. Sen, and O. Spatscheck, "An in-depth study of lte: Effect of network protocol and application behavior on performance," *SIGCOMM Comput. Commun. Rev.*, vol. 43, no. 4, pp. 363–374, Aug. 2013. [Online]. Available: http://doi.acm.org/10.1145/2534169.2486006

[27] J. Sommers and P. Barford, "Cell vs. wifi: on the performance of metro area mobile connections," in *Proceedings of the 2012 ACM conference on Internet measurement conference*. ACM, 2012, pp. 301–314.

[28] J. Huang, Q. Xu, B. Tiwana, Z. M. Mao, M. Zhang, and P. Bahl, "Anatomizing application performance differences on smartphones," in *Proceedings of the 8th international conference on Mobile systems, applications, and services*. ACM, 2010, pp. 165–178.

[29] J. Huang, F. Qian, A. Gerber, Z. M. Mao, S. Sen, and O. Spatscheck, "A close examination of performance and power characteristics of 4g lte networks," in *Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys '12. New York, NY, USA: ACM, 2012, pp. 225–238. [Online]. Available: http://doi.acm.org/10.1145/2307636.2307658

[30] A. Gurtov, M. Passoja, O. Aalto, and M. Raitola, "Multi-layer protocol tracing in a gprs network," in *Vehicular Technology Conference, 2002. Proceedings. VTC 2002-Fall. 2002 IEEE 56th*, vol. 3. IEEE, 2002, pp. 1612–1616.

[31] M. C. Chan and R. Ramjee, "Tcp/ip performance over 3g wireless links with rate and delay variation," in *Proceedings of the 8th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '02. New York, NY, USA: ACM, 2002, pp. 71–82. [Online]. Available: http://doi.acm.org/10.1145/570645.570655

[32] H. Jiang, Y. Wang, K. Lee, and I. Rhee, "Tackling bufferbloat in 3g/4g networks," in *Proceedings of the 2012 ACM Conference on Internet Measurement Conference*, ser. IMC '12. New York, NY, USA: ACM, 2012, pp. 329–342. [Online]. Available: http://doi.acm.org/10.1145/2398776.2398810

[33] E. Dahlman, S. Parkvall, and J. Skold, *4G: LTE/LTE-Advanced for Mobile Broadband*. Elsevier Science, 2011. [Online]. Available: https://books.google.fr/books?id=DLbsq9GD0zMC

[34] I. Johansson, "Congestion control for 4G and 5G access," Internet Engineering Task Force, Internet-Draft draft-johansson-cc-for-4g-5g-02, Jul. 2016, work in Progress. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-johansson-cc-for-4g-5g-02

[35] A. O. Allen, *Probability, Statistics, and Queueing Theory with Computer Science Applications*. San Diego, CA, USA: Academic Press Professional, Inc., 1990.

[36] X. Liu, A. Sridharan, S. Machiraju, M. Seshadri, and H. Zang, "Experiences in a 3g network: Interplay between the wireless channel and applications," in *Proceedings of the 14th ACM International Conference on Mobile Computing and Networking*, ser. MobiCom '08. New York, NY, USA: ACM, 2008, pp. 211–222. [Online]. Available: http://doi.acm.org/10.1145/1409944.1409969

[37] B. Sardar and D. Saha, "A survey of tcp enhancements for last-hop wireless networks," *IEEE Communications Surveys Tutorials*, vol. 8, no. 3, pp. 20–34, rd 2006.

[38] X. Xie, X. Zhang, S. Kumar, and L. E. Li, "pistream: Physical layer informed adaptive video streaming over lte," in *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '15. New York, NY, USA: ACM, 2015, pp. 413–425. [Online]. Available: http://doi.acm.org/10.1145/2789168.2790118

[39] N. Cardwell, Y. Cheng, S. H. Yeganeh, and V. Jacobson, *BBR Congestion Control draft-cardwell-iccrg-bbr-congestion-control-00*, Google, Inc Std., July 2017. [Online]. Available: https://tools.ietf.org/html/draft-cardwell-iccrg-bbr-congestion-control-00

[40] F. Lu, H. Du, A. Jain, G. M. Voelker, A. C. Snoeren, and A. Terzis, "Cqic: Revisiting cross-layer congestion control for cellular networks," in *Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications*, ser. HotMobile '15. New York, NY, USA: ACM, 2015, pp. 45–50. [Online]. Available: http://doi.acm.org/10.1145/2699343.2699345

[41] J. H. Saltzer, D. P. Reed, and D. D. Clark, "End-to-end arguments in system design," *ACM Trans. Comput. Syst.*, vol. 2, no. 4, pp. 277–288, Nov. 1984. [Online]. Available: http://doi.acm.org/10.1145/357401.357402

[42] J. Postel *et al.*, "User datagram protocol," 1980.

[43] M. Allman, V. Paxson, W. Stevens *et al.*, "Tcp congestion control," 1999.

[44] W. R. Stevens and T. Narten, "Unix network programming," *ACM SIGCOMM Computer Communication Review*, vol. 20, no. 2, pp. 8–9, 1990.

[45] L. S. Brakmo and L. L. Peterson, "Tcp vegas: end to end congestion avoidance on a global internet," *IEEE Journal on Selected Areas in Communications*, vol. 13, no. 8, pp. 1465–1480, Oct 1995.

[46] V. Jacobson, "Congestion avoidance and control," *SIGCOMM Comput. Commun. Rev.*, vol. 18, no. 4, pp. 314–329, Aug. 1988. [Online]. Available: http://doi.acm.org/10.1145/52325.52356

[47] V. P. M. Allman, and and W. Stevens, "Tcp congestion control," Network Working Group, Tech. Rep., 1999, rFC2581.

[48] S. Floyd and T. Henderson, "The new reno modification to tcp's fast recovery algorithm," Network Working Group, Tech. Rep., 2004, rFC 2582.

[49] S. Ha, I. Rhee, and L. Xu, "Cubic: A new tcp-friendly high-speed tcp variant," *SIGOPS Oper. Syst. Rev.*, vol. 42, no. 5, pp. 64–74, Jul. 2008. [Online]. Available: http://doi.acm.org/10.1145/1400097.1400105

[50] L. Xu, K. Harfoush, and I. Rhee, "Binary increase congestion control (bic) for fast long-distance networks," in *IEEE INFOCOM 2004*, vol. 4. IEEE, 2004, pp. 2514–2524.

[51] K. N. Srijith, L. Jacob, and A. L. Ananda, "Tcp vegas-a: solving the fairness and rerouting issues of tcp vegas," in *Conference Proceedings of the 2003 IEEE International Performance, Computing, and Communications Conference, 2003.*, April 2003, pp. 309–316.

[52] U. Hengartner, J. Bolliger, and T. Gross, "Tcp vegas revisited," in *Proceedings IEEE INFOCOM 2000. Conference on Computer Communications. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (Cat. No. 00CH37064)*, vol. 3. IEEE, 2000, pp. 1546–1555.

[53] R. J. La, J. Walrand, and V. Anantharam, *Issues in TCP vegas*.

[54] K. Xu, Y. Tian, and N. Ansari, "Tcp-jersey for wireless ip communications," *IEEE J.Sel. A. Commun.*, vol. 22, no. 4, pp. 747–756, Sep. 2006. [Online]. Available: http://dx.doi.org/10.1109/JSAC.2004.825989

[55] ——, "Improving tcp performance in integrated wireless communications networks," *Comput. Netw.*, vol. 47, no. 2, pp. 219–237, Feb. 2005. [Online]. Available: http://dx.doi.org/10.1016/j.comnet.2004.07.006

[56] J. Kim, J. Koo, and H. Choo, *TCP NJ+: Packet Loss Differentiated Transmission Mechanism Robust to High BER Environments*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 380–390. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-72606-7_33

[57] E. H.-K. Wu and M.-Z. Chen, "Jtcp: Jitter-based tcp for heterogeneous wireless networks," *IEEE J.Sel. A. Commun.*, vol. 22, no. 4, pp. 757–766, Sep. 2006. [Online]. Available: http://dx.doi.org/10.1109/JSAC.2004.825999

[58] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "Rtp: A transport protocol for real-time applications," Network Working Group, Tech. Rep., 2003.

[59] S.-Y. Chen, E. H.-K. Wu, and M.-Z. Chen, "A new approach using time-based model for tcp-friendly rate estimation," in *Communications, 2003. ICC'03. IEEE International Conference on*, vol. 1. IEEE, 2003, pp. 679–683.

[60] S. Mascolo, C. Casetti, M. Gerla, M. Y. Sanadidi, and R. Wang, "Tcp westwood: Bandwidth estimation for enhanced transport over wireless links," in *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '01. New York, NY, USA: ACM, 2001, pp. 287–297. [Online]. Available: http://doi.acm.org/10.1145/381677.381704

[61] C. P. Fu and S. C. Liew, "Tcp veno: Tcp enhancement for transmission over wireless access networks," *IEEE J.Sel. A. Commun.*, vol. 21, no. 2, pp. 216–228, Sep. 2006. [Online]. Available: http://dx.doi.org/10.1109/JSAC.2002.807336

[62] S. Shalunov, "Low extra delay background transport (ledbat)," Internet Engineering Task Force (IETF), Tech. Rep., mar 2009.

[63] A. Bakre and B. R. Badrinath, "I-tcp: indirect tcp for mobile hosts," in *Distributed Computing Systems, 1995., Proceedings of the 15th International Conference on*, May 1995, pp. 136–143.

[64] A. V. Bakre and B. R. Badrinath, "Implementation and performance evaluation of indirect tcp," *IEEE Transactions on Computers*, vol. 46, no. 3, pp. 260–278, Mar 1997.

[65] R. Yavatkar and N. Bhagawat, "Improving end-to-end performance of tcp over mobile internetworks," in *Mobile Computing Systems and Applications, 1994. Proceedings., Workshop on*, Dec 1994, pp. 146–152.

[66] K.-Y. Wang and S. K. Tripathi, "Mobile-end transport protocol: an alternative to tcp/ip over wireless links," in *INFOCOM '98. Seventeenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 3, Mar 1998, pp. 1046–1053 vol.3.

[67] K. Brown and S. Singh, "M-tcp: Tcp for mobile cellular networks," *SIGCOMM Comput. Commun. Rev.*, vol. 27, no. 5, pp. 19–43, Oct. 1997. [Online]. Available: http://doi.acm.org/10.1145/269790.269794

[68] W. G. Zeng and L. Trajkovic, "Tcp packet control for wireless networks," in *WiMob'2005), IEEE International Conference on Wireless And Mobile Computing, Networking And Communications, 2005.*, vol. 2, Aug 2005, pp. 196–203 Vol. 2.

[69] M. Ivanovich, P. W. Bickerdike, and J. C. Li, "On tcp performance enhancing proxies in a wireless environment," *IEEE Communications Magazine*, vol. 46, no. 9, pp. 76–83, September 2008.

[70] A. Jain, A. Terzis, H. Flinck, N. Sprecher, S. Arunachalam, K. Smith, V. Devarapalli, and R. B. Yanai, "Mobile Throughput Guidance Inband Signaling Protocol," Internet Engineering Task Force, Internet-Draft draft-flinck-mobile-throughput-guidance-04, Mar. 2017, work in Progress. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-flinck-mobile-throughput-guidance-04

[71] N. S. S. A. K. S. V. D. R. B. Y. A. Jain, A. Terzis, "Mobile throughput guidance inband signaling protocol." Internet Engineering Task Force, 2017, p. 1.

[72] I. Sodagar, "The mpeg-dash standard for multimedia streaming over the internet," *IEEE Multimedia*, vol. 18, no. 4, pp. 62–67, 2011.

[73] X. K. Zou, J. Erman, V. Gopalakrishnan, E. Halepovic, R. Jana, X. Jin, J. Rexford, and R. K. Sinha, "Can accurate predictions improve video streaming in cellular networks?" in *Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications*, ser. HotMobile '15. New York, NY, USA: ACM, 2015, pp. 57–62. [Online]. Available: http://doi.acm.org/10.1145/2699343.2699359

[74] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson, "On the self-similar nature of ethernet traffic (extended version)," *IEEE/ACM Transactions on Networking*, vol. 2, no. 1, pp. 1–15, Feb 1994.

[75] M. E. Crovella and A. Bestavros, "Self-similarity in world wide web traffic: Evidence and possible causes," *IEEE/ACM Trans. Netw.*, vol. 5, no. 6, pp. 835–846, Dec. 1997. [Online]. Available: http://dx.doi.org/10.1109/90.650143

[76] K. Winstein, A. Sivaraman, and H. Balakrishnan, "Stochastic forecasts achieve high throughput and low delay over cellular networks," in *Proceedings of the 10th USENIX Conference on Networked Systems Design and Implementation*, ser. nsdi'13. Berkeley, CA, USA: USENIX Association, 2013, pp. 459–472. [Online]. Available: http://dl.acm.org/citation.cfm?id=2482626.2482670

[77] E. H.-K. Wu, Y.-C. Huang, and G.-K. Chang, "Ejtcp: Enhanced jitter-based tcp for wireless broadband networks," *Journal of Information Science and Engineering*, vol. 23, no. 6, pp. 1663–1679, 2007.

[78] A. Andreadis, S. Rizzuto, and R. Zambon, "A cross-layer jitter-based tcp for wireless networks," *EURASIP Journal on Wireless Communications and Networking*, vol. 2016, no. 1, p. 191, 2016. [Online]. Available: http://dx.doi.org/10.1186/s13638-016-0695-0

[79] K. Ramakrishnan and S. Floyd, "A proposal to add explicit congestion notification (ecn) to ip," AT&T Labs Research, Tech. Rep., jan 1999, rFC2481.

[80] B. S. Bakshi, P. Krishna, N. H. Vaidya, and D. K. Pradhan, "Improving performance of tcp over wireless networks," in *Distributed Computing Systems, 1997., Proceedings of the 17th International Conference on*. IEEE, 1997, pp. 365–373.

[81] H. Balakrishnan, S. Seshan, E. Amir, and R. H. Katz, "Improving tcp/ip performance over wireless networks," in *Proceedings of the 1st Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '95. New York, NY, USA: ACM, 1995, pp. 2–11. [Online]. Available: http://doi.acm.org/10.1145/215530.215544

[82] R. Kwan, C. Leung, and J. Zhang, "Proportional fair multiuser scheduling in lte," *IEEE Signal Processing Letters*, vol. 16, no. 6, pp. 461–464, June 2009.

[83] M. Proebster, C. M. Mueller, and H. Bakker, "Adaptive fairness control for a proportional fair lte scheduler," in *21st Annual IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, Sept 2010, pp. 1504–1509.

[84] A. H. Zahran, J. J. Quinlan, K. K. Ramakrishnan, and C. J. Sreenan, "Impact of the lte scheduler on achieving good qoe for dash video streaming," in *2016 IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN)*, June 2016, pp. 1–7.

[85] J. Fabini and T. Zseby, "The right time: Reducing effective end-to-end delay in time-slotted packet-switched networks," *IEEE/ACM Transactions on Networking*, vol. 24, no. 4, pp. 2251–2263, Aug 2016.

[86] Y. Huo, X. Dong, and W. Xu, "5g cellular user equipment: From theory to practical hardware design," *IEEE Access*, vol. 5, pp. 13 992–14 010, 2017.

[87] M. Zhang, M. Polese, M. Mezzavilla, J. Zhu, S. Rangan, S. Panwar, and M. Zorzi, "Will tcp work in mmwave 5g cellular networks?" *IEEE Communications Magazine*, vol. 57, no. 1, pp. 65–71, 2019.

[88] "Bbr development google group." [Online]. Available: https://groups.google.com/forum/#!forum/bbr-dev

[89] N. Cardwell, Y. Cheng, C. S. Gunn, S. H. Yeganeh, and V. Jacobson, "Bbr: Congestion-based congestion control," *Queue*, vol. 14, no. 5, pp. 50:20–50:53, Oct. 2016. [Online]. Available: http://doi.acm.org/10.1145/3012426.3022184

[90] W. R. Stevens, *TCP/IP illustrated vol. I: the protocols*. Pearson Education India, 1994.

[91] A. Aggarwal, S. Savage, and T. Anderson, "Understanding the performance of tcp pacing," in *Proceedings IEEE INFOCOM 2000. Conference on Computer Communications. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (Cat. No. 00CH37064)*, vol. 3. IEEE, 2000, pp. 1157–1165.

## References


[92] J. Cao, W. S. Cleveland, D. Lin, and D. X. Sun, "Internet traffic tends toward poisson and independent as the load increases," in *Nonlinear estimation and classification*. Springer, 2003, pp. 83–109.

[93] Z. Zhong, I. Hamchaoui, and R. Khatoun, "Perils of using cqic in lte network and a quick fix with delayed ack," in *15th IEEE Annual Consumer Communications and Networking Conference (CCNC 2018) - Posters Accepted*, January 2018.

[94] B. Nguyen, A. Banerjee, V. Gopalakrishnan, S. Kasera, S. Lee, A. Shaikh, and J. Van der Merwe, "Towards understanding tcp performance on lte/epc mobile networks," in *Proceedings of the 4th Workshop on All Things Cellular: Operations, Applications, & Challenges*, ser. AllThingsCellular '14. New York, NY, USA: ACM, 2014, pp. 41–46. [Online]. Available: http://doi.acm.org/10.1145/2627585.2627594

[95] H. Jiang, Z. Liu, Y. Wang, K. Lee, and I. Rhee, "Understanding bufferbloat in cellular networks," in *Proceedings of the 2012 ACM SIGCOMM Workshop on Cellular Networks: Operations, Challenges, and Future Design*, ser. CellNet '12. New York, NY, USA: ACM, 2012, pp. 1–6. [Online]. Available: http://doi.acm.org/10.1145/2342468.2342470

[96] N. Cardwell, Y. Cheng, C. S. Gunn, S. H. Yeganeh, and V. Jacobson, "Bbr: Congestion-based congestion control," *Queue*, vol. 14, no. 5, pp. 50:20–50:53, Oct. 2016. [Online]. Available: http://doi.acm.org/10.1145/3012426.3022184

[97] J. E. Hopcroft and J. D. Ullman, *Data structures and algorithms*, 1983.

[98] Y. Huang and B. Hu, "Enhanced dctcp to explicitly inform of packet loss," in *2015 IEEE International Conference on Communications (ICC)*, June 2015, pp. 5511–5516.

[99] J. Iyengar and I. Swett, *QUIC: A UDP-Based Secure and Reliable Transport for HTTP/2 draft-tsvwg-quic-protocol-00*, Network Working Group Std. [Online]. Available: https://tools.ietf.org/html/draft-tsvwg-quic-protocol-00

[100] R. Hamilton, J. Iyengar, I. Swett, and A. Wilk, *QUIC: A UDP-Based Multiplexed and Secure Transport, draft-hamilton-quic-transport-protocol-01*, Network Working Group Std. [Online]. Available: https://tools.ietf.org/html/draft-hamilton-quic-transport-protocol-01

[101] G. Piro, N. Baldo, and M. Miozzo, "An lte module for the ns-3 network simulator." in *SimuTools*, 2011, pp. 415–422.

[102] S. Landström and L.-A. Larzon, "Reducing the tcp acknowledgment frequency," *SIGCOMM Comput. Commun. Rev.*, vol. 37, no. 3, pp. 5–16, Jul. 2007. [Online]. Available: http://doi.acm.org/10.1145/1273445.1273447

[103] Z. Zhenzhe, H. Isabelle, K. Rida, and S. Ahmed, "Performance evaluation of cqic and tcp bbr in mobile network," in *Proceedings 21st Conference on Innovation in Clouds, Internet and Networks (ICIN 2018)*, Paris, France, 2018.

[104] B. Levasseur, M. Claypool, and R. Kinicki, "Impact of acknowledgments on application performance in 4g lte networks," in *2015 International Conference on Computing, Networking and Communications (ICNC)*, Feb 2015, pp. 1034–1038.

[105] H. S. Park, J. Y. Lee, and B. C. Kim, "Tcp performance issues in lte networks," in *ICTC 2011*, Sept 2011, pp. 493–496.

[106] M. Wang, Z. Zhong, and Q. Liu, "Resource allocation for sc-fdma in lte uplink," in *Proceedings of 2011 IEEE International Conference on Service Operations, Logistics and Informatics*, July 2011, pp. 601–604.

[107] L. Kleinrock, "Power and deterministic rules of thumb for probabilistic problems in computer communications," in *ICC'79; International Conference on Communications, Volume 3*, vol. 3, 1979, pp. 43–1.

[108] Y. Zaki, T. Pötsch, J. Chen, L. Subramanian, and C. Görg, "Adaptive congestion control for unpredictable cellular networks," in *ACM SIGCOMM Computer Communication Review*, vol. 45, no. 4. ACM, 2015, pp. 509–522.

[109] D. A. Hayes and G. Armitage, "Revisiting tcp congestion control using delay gradients," in *International Conference on Research in Networking*. Springer, 2011, pp. 328–341.

[110] G. Carlucci, L. De Cicco, S. Holmer, and S. Mascolo, "Analysis and design of the google congestion control for web real-time communication (webrtc)," in *Proceedings of the 7th International Conference on Multimedia Systems*. ACM, 2016, p. 13.

[111] B. Veal, K. Li, and D. Lowenthal, "New methods for passive estimation of tcp round-trip times," in *International Workshop on Passive and Active Network Measurement*. Springer, 2005, pp. 121–134.

[112] S. H. Low, L. L. Peterson, and L. Wang, "Understanding tcp vegas: a duality model," *Journal of the ACM (JACM)*, vol. 49, no. 2, pp. 207–235, 2002.

[113] M. Hock, R. Bless, and M. Zitterbart, "Experimental evaluation of bbr congestion control," in *2017 IEEE 25th International Conference on Network Protocols (ICNP)*, Oct 2017, pp. 1–10.

[114] J. Postel, "The tcp maximum segment sizeand related topics." Network Working Group, 1983, p. 1.

[115] A. K. Parekh and R. G. Gallager, "A generalized processor sharing approach to flow control in integrated services networks: the single-node case," *IEEE/ACM Transactions on Networking*, vol. 1, no. 3, pp. 344–357, June 1993.

[116] R. K. Jain, D.-M. W. Chiu, and W. R. Hawe, "A quantitative measure of fairness and discrimination," 1984.

[117] S. Zhang, "An evaluation of bbr and its variants," *arXiv preprint arXiv:1909.03673*, 2019.

ECOLE DOCTORALE

**Titre :** Contrôle de congestion 'cross-layer' et qualité de service dans les réseaux mobiles

**Mots clés :** réseaux mobiles, contrôle de congestion, LTE, estimation de la bandwidth réseau d'accès radio.

**Résumé :** Le réseau mobile est un réseau hybride avec une partie d'accès radio et le réseau central de liaison fixe. Les algorithmes de contrôle de congestion (CCA) conçus pour un type de système spécifique peuvent ne pas fonctionner aussi bien dans l'autre type de réseau, en particulier le réseau avec un dispositif de fonctionnalité hybride comme le réseau de périphérie mobile. Généralement, le goulot d'étranglement dans un réseau mobile est la partie accès radio. Cependant, ce n'est pas toujours le cas puisque plusieurs stations de base radio ou passerelle de réseau de livraison de paquets peuvent partager le même goulot d'étranglement dans le backhaul de livraison de paquets. Dans cette thèse, nous partons d'une méthode cross-layer et abordons le problème avec une solution omniprésente. Le premier algorithme que nous avons analysé est appelé CQIC, qui implique la couche PHY de l'UE dans la conception du contrôle de la congestion. Une amélioration du scénario 3G CQIC au scénario LTE est proposée sous le nom de DCIC. Cet algorithme utilise l'indicateur de commande de liaison descendante (DCI) au lieu de l'indicateur de qualité de canal (CQI) pour économiser la puissance de calcul sur l'UE et prendre en compte le résultat de la planification d'eNB. En outre, nous avons évalué l'algorithme BBR actuel, qui se concentre sur le réseau du centre de données, dans le scénario mobile. La plupart des CCA conventionnels ne prennent pas en compte la dégradation du BW de liaison montante et les autres caractéristiques du système cellulaire dans sa méthode d'estimation de la largeur de bande. Sur la base de cet examen, nous avons proposé les cinq objectifs de compromis pour guider la conception de l'ACC dans un type de réseau hybride mobile: utilisation de la bande passante, délai (où la perte est l'expression extrême du retard), équité, simplicité et généricité. Sur la base des compromis et des objectifs, nous avons proposé le CDBE, une estimation de la bande passante pilotée par le client TCP (CDBE) et une boucle de rétroaction de rapport. La méthode d'estimation BW côté client ne prend que peu de capacité de calcul dans la deuxième version, par rapport à la première version ou CQIC et DCIC. Coopérez avec la transition d'état côté serveur améliorée CDBE peut atteindre une part équitable de BW dans le réseau central à paquets fixes ou le réseau mobile avec un coût de RTT inférieur à celui des CCA conventionnels. Aucune unité / application de boîtier de médiation ou de périphérie n'est requise dans l'architecture CDBE.

**Title :** Cross-layer congestion control and quality of service in mobile networks

**Keywords :** Mobile networks, Congestion Control, LTE, bandwidth estimation, Radio Access Network.

**Abstract :** The mobile network is a hybrid network with Radio Access part and the fixed backhaul core network. The congestion control algorithms(CCA) designed for a specific type of system may not work as well in the other kind of network, especially the network with hybrid feature device like the mobile edge network. Generally, the bottleneck in a mobile network is the Radio access part. However, this is not always the case since multiple radio base stations or packet delivery network gateway can be sharing the same bottleneck in the packet delivery backhaul. In this thesis, we start from a cross-layer method and address the issue with a ubiquitous solution. The first algorithm we analysed is called CQIC, which get the PHY layer of UE involved in the congestion control design. An improvement from 3G CQIC to LTE scenario is proposed named DCIC. This algorithm uses the Downlink Control Indicator(DCI) instead of Channel Quality Indicator(CQI) to save the computation power on UE and take the scheduling result of eNB into consideration. Further, we evaluated current BBR algorithm, which focuses on the data centre network, in the mobile scenario. Most conventional CCA does not take the uplink BW degradation and other features in the cellular system into consideration in its bandwidth estimation method. Based on the review, we proposed the five tradeoff objectives to guide the CCA design in a mobile hybrid type of network: Bandwidth Utilisation, Delay(where loss is the extreme expression of delay), Fairness, Simplicity and Genericity. Based on the tradeoffs and goals, we proposed CDBE, a TCP client-side driven bandwidth estimation(CDBE) and report feedback loop. The client-side BW estimation method takes only little computation capability in the second version, compared to its first version and the DCIC. Cooperate with the enhanced server-side state transition CDBE can achieve a fair share of BW in both fixed packet core network or mobile network with a lower cost of RTT compared to conventional CCAs. No extra middlebox or edge computing unit/applications is required in CDBE architecture.

**Institut Polytechnique de Paris**
91120 Palaiseau, France