ระบบการส่งวีดิทัศน์ที่ปรับขนาดได้หลายผู้รับบนระบบเครือข่ายที่กำหนดโดยซอฟต์แวร์

Scalable Video Coding Multicast over Software Defined Networking

ปิยวิทย์ ตันติสาครเขต

Piyawit Tantisarkhornkhet

วิทยานิพนธ์นี้สำหรับการศึกษาตามหลักสูตรปริญญา
วิศวกรรมศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมคอมพิวเตอร์
มหาวิทยาลัยสงขลานครินทร์

A Thesis Submitted in Fulfillment of the Requirements for the Degree of
Master of Engineering in Computer Engineering
Prince of Songkla University
2560

ชื่อวิทยานิพนธ์ ระบบการส่งวีดิทัศน์ที่ปรับขนาดได้หลายผู้รับบนระบบเครือข่ายที่กำหนดโดยซอฟต์แวร์

ผู้เขียน นายปิยวิทย์ ตันติสาครเขต

สาขาวิชา วิศวกรรมคอมพิวเตอร์

อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก

.....

(ผู้ช่วยศาสตราจารย์ ดร.วโรดม วีระพันธ์)

คณะกรรมการสอบ

.....ประธานกรรมการ

(ดร.จิรวัฒน์ แท่นทอง)

.....กรรมการ

(ผู้ช่วยศาสตราจารย์ ดร.วโรดม วีระพันธ์)

.....กรรมการ

(ผู้ช่วยศาสตราจารย์ ดร.อภิชาติ หีดนาคราม)

.....กรรมการ

(ผู้ช่วยศาสตราจารย์ ดร.พงษ์พิสิฐ วุฒิดิษฐโชติ)

บัณฑิตวิทยาลัย มหาวิทยาลัยสงขลานครินทร์ อนุมัติให้นับวิทยานิพนธ์ฉบับนี้สำหรับการศึกษา ตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมคอมพิวเตอร์

.....

(รองศาสตราจารย์ ดร.ธีระพล ศรีชนะ)

คณบดีบัณฑิตวิทยาลัย

ขอรับรองว่า ผลงานวิจัยนี้มาจากการศึกษาวิจัยของนักศึกษาเอง และได้แสดงความขอบคุณบุคคลที่มีส่วนช่วยเหลือแล้ว

ลงชื่อ.....
(ผู้ช่วยศาสตราจารย์ ดร.วโรดม วีระพันธ์)
อาจารย์ที่ปรึกษาวิทยานิพนธ์

ลงชื่อ.....
(นายปิยวิทย์ ตันติสาครเขต)
นักศึกษา

ข้าพเจ้าขอรับรองว่า ผลงานวิจัยนี้ไม่เคยเป็นส่วนหนึ่งในการอนุมัติปริญญาในระดับใดมาก่อน และไม่ได้ถูกใช้ในการยื่นขออนุมัติปริญญาในขณะนี้

ลงชื่อ.....
(นายปิยวิทย์ ตันติสาครเขต)
นักศึกษา

ชื่อวิทยานิพนธ์ ระบบการส่งวีดิทัศน์ที่ปรับขนาดได้หลายผู้รับบนระบบเครือข่ายที่กำหนดโดยซอฟต์แวร์

ผู้เขียน นายปิยวิทย์ ตันติสาครเขต

สาขาวิชา วิศวกรรมคอมพิวเตอร์

ปีการศึกษา 2559

บทคัดย่อ

วิทยานิพนธ์นี้นำเสนอรูปแบบการส่งข้อมูลประเภทวีดิทัศน์ปรับขนาดได้แบบหลายผู้รับบนระบบเครือข่ายที่กำหนดโดยซอฟต์แวร์ ผู้วิจัยได้เลือกพัฒนา 2 ส่วน ในส่วนที่ 1 คือ ส่วนการจัดการการส่งข้อมูล และส่วนที่ 2 คือ ส่วนผู้ส่งและผู้รับ (ส่วนที่ 1) ในส่วนการจัดการการส่งข้อมูล เป็นการจัดการระบบเครือข่ายด้วยซอฟต์แวร์ ผู้วิจัยได้ออกแบบรูปแบบการหาเส้นทางที่เน้นในการส่งข้อมูลประเภทวีดิทัศน์สตรีมมิ่งโดยใช้ระดับขีดแบ่งที่แตกต่างกันตามความต้องการของระดับขนาดวีดิทัศน์ที่แตกต่างกันโดยเลือกคุณภาพเส้นทางในการส่งข้อมูลแต่ละด้านที่สมดุลกันซึ่งมีชื่อว่า Quantized Level Balance (QLB) ผู้วิจัยได้นำรูปแบบดังกล่าวเปรียบเทียบกับรูปแบบการหาเส้นทางจากระบบเครือข่ายดั้งเดิมโดยใช้รูปแบบ Open Shortest Path First (OSPF) ซึ่งผลลัพธ์จากการแสดงผลจะเห็นได้ว่า QLB สามารถควบคุมคุณภาพการแสดงผลได้ดีกว่ารูปแบบการหาเส้นทางแบบ OSPF (ส่วนที่ 2) ผู้วิจัยได้ออกแบบรูปแบบการรับส่งข้อมูลแบบหลายขนาดและหลายผู้รับ โดยปรับปรุงรูปแบบการส่งข้อมูลวีดิทัศน์ประเภท Scalable Video Coding-Multiple Session Transmission (SVC-MST) เนื่องจากรูปแบบในการรับส่งข้อมูลในโหมดของ SVC-MST เป็นรูปแบบที่สามารถส่งข้อมูลโดยการสร้างเส้นทางที่แตกต่างให้กับแต่ละชั้นวีดิทัศน์ที่แตกต่างกัน จึงสามารถลดข้อจำกัดของขนาดแบนด์วิดท์เพื่อแก้ปัญหาการชะงักซึ่งเป็นปัญหาที่เกิดจากการส่งข้อมูลอย่างไม่เพียงพอ ผู้วิจัยได้เปรียบเทียบรูปแบบการส่งข้อมูลของ SVC-MST แบบปรับปรุงในชื่อ SVC-MST BWQLB กับรูปแบบการส่งข้อมูลของ Advanced Video Coding (AVC), Scalable Video Coding-Single Session Transmission (SVC-SST) และ SVC-MST แบบดั้งเดิม ผลปรากฏว่า SVC-MST BWQLB สามารถลดความล่าช้าในการส่งข้อมูลแบบกลุ่มได้สูงสุด 46.77% (640x360 และ 1280x720) และ 66.64% (640x360, 1280x720 และ 1920x1080) ตามลำดับเมื่อเปรียบเทียบกับ SVC-SST อย่างไรก็ตามผู้รับวีดิทัศน์ประเภท SVC-MST จะได้รับวีดิทัศน์ที่ไม่ตรงกับลำดับที่ถูกต้องทำให้มีการแสดงผลที่ผิดพลาด ผู้วิจัยจึงได้สร้างบัฟเฟอร์ที่ใช้ในการจัดเรียงข้อมูล

คำสำคัญ: ระบบเครือข่ายที่กำหนดโดยซอฟต์แวร์, การเข้ารหัสวีดิทัศน์ประสิทธิภาพสูง, การเข้ารหัสวีดิทัศน์ปรับขนาดได้, มัลติคลาสต์, รูปแบบการส่งแบบเส้นทางเดียว, รูปแบบการส่งข้อมูลหลายเส้นทาง

Thesis Title Scalable Video Coding Multicast over Software Defined Networking
Author Mr.Piyawit Tantisarkhornkhet
Major Program Computer Engineering
Academic Year 2016

ABSTRACT

This thesis presents the methods of Scalable Video Coding (SVC) Multicast over Software Defined Networking. We choose to develop 2 parts. In the first part is related to the transmission management, and the second part is described a sender and a receiver. In the first part, we implement the routing algorithm using SDN which focuses on the transmission of video streaming data by using different thresholds according to requirement of different video resolutions and choose the route to transmit based on balance quality of service parameters, called Quantized Level Balance (QLB). We examine this model with traditional network dynamic routing by using the Open Shortest Path First (OSPF) routing. The results are revealed that QLB routing can control the display quality better than OSPF routing. In second part, we design the transmission of multiple video resolutions and receivers by modified SVC-MST (Multi-session transmission) called SVC-MST BWQLB since the SVC-MST mode transmission is used to send different video resolutions to several paths. SVC-MST can reduce the usage bandwidth for freezing solution, which is a problem caused by bandwidth congestion. We compare the transmission of the modified SVC-MST with the Advanced Video Coding (AVC), the SVC-SST (Single-session transmission) mode and the tranditional SVC-MST. The maximum delay can be reduced by 46.77% (640x360 and 1280x720) and 66.64% (640x360, 1280x720 and 1920x1080) compared to SVC-SST. However, SVC-MST receiver gets the video that does not match with the correct sequence, resulting in encoded error. Eventually, we introduce buffer to solve the problem of defragmentation.

Keywords: Software Defined Networking, Advanced Video Coding, Scalable Video Coding, Multicast, Single-session transmission, Multi-session transmission

กิตติกรรมประกาศ

ขอกราบขอบพระคุณ ผู้ช่วยศาสตราจารย์ ดร.วโรดม วีระพันธ์ อาจารย์ที่ปรึกษาวิทยานิพนธ์ ผู้ซึ่ง เสนอแนวทาง ชี้แนะข้อบกพร่อง ดูแลให้คำปรึกษา ให้ความรู้ในด้านต่าง ๆ อีกทั้งยังสนับสนุนอุปกรณ์ ในการทำวิจัย ตลอดจนตรวจแก้ไขวิทยานิพนธ์จนกระทั่งสำเร็จลุล่วง

ขอบพระคุณ ดร.จิรวัฒน์ แท่นทอง ที่กรุณาให้เกียรติเป็นประธานกรรมการสอบวิทยานิพนธ์ อีกทั้งยังให้คำเสนอแนะ ดูแลให้คำปรึกษาและแก้ไขวิทยานิพนธ์กระทั่งสำเร็จลุล่วง

ขอขอบพระคุณ ผู้ช่วยศาสตราจารย์ ดร. อภิชาติ หีดนาคราม ที่กรุณาให้เกียรติเป็นกรรมการสอบวิทยานิพนธ์ อีกทั้งเสนอแนะแนวทางในการวิจัย สนับสนุนความรู้ต่าง ๆ ให้คำปรึกษาเป็นอย่างดีเสมอมา ตลอดจนตรวจแก้ไขวิทยานิพนธ์ให้ดำเนินไปอย่างสมบูรณ์

ขอขอบพระคุณ ผู้ช่วยศาสตราจารย์ ดร. พงษ์พิสิฐ วุฒิดิษฐโชติ จากมหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ ที่กรุณาให้ เกียรติเป็นกรรมการสอบวิทยานิพนธ์ และให้คำชี้แนะอันเป็นประโยชน์ยิ่งต่อการวิจัย ตลอดจนตรวจ แก้ไขวิทยานิพนธ์ให้ดำเนินไปอย่างสมบูรณ์

ขอขอบพระคุณภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ ที่สนับสนุนทุนค่าธรรมเนียมการศึกษา ระดับบัณฑิตศึกษา ด้วยเงินรายได้โครงการศิษย์ก้นกุฏิสาขาวิชาวิศวกรรมคอมพิวเตอร์ ณ วิทยาเขตหาดใหญ่

ขอขอบพระคุณคณาจารย์และบุคลากรทุกท่านในภาควิชาวิศวกรรมคอมพิวเตอร์ที่อบรมสั่งสอน ให้คำปรึกษาตลอดมา ตลอดจนสนับสนุนอุปกรณ์ที่จำเป็นต้องใช้ในงานวิจัย

ขอบคุณนักศึกษาภาควิชาวิศวกรรมศาสตร์ คณะวิศวกรรมคอมพิวเตอร์ ทั้งที่วิทยาเขตหาดใหญ่และวิทยาเขตภูเก็ตทุกท่าน ที่ได้ให้ความช่วยเหลือ ให้กำลังใจ และคำแนะนำที่ดีเสมอมา

สุดท้ายนี้ขอน้อมรำลึกถึงพระคุณบิดา มารดา และครอบครัวที่เอื้อเฟื้อ ให้การสนับสนุน ให้กำลังใจ ให้ความช่วยเหลือ และส่งเสริมการศึกษาเป็นอย่างดีเสมอมาจนทำให้ข้าพเจ้าประสบผลสำเร็จ

ปิยวิทย์ ตันติสาครเขต

สารบัญ

หน้า

สารบัญ

หน้า

รายการตาราง

หน้า

รายการภาพประกอบ

หน้า

รายการภาพประกอบ

หน้า

สัญลักษณ์ ตัวย่อ และคำย่อ

SDN	Software Defined Networking
AVC	Advanced Video Coding
SVC	Scalable Video Coding
SST	Single Session Transmission
MST	Multiple Session Transmission
GOP	Group of Pictures
RIP	Routing Information Protocol
IGRP	Interior Gateway Routing Protocol
EIGRP	Enhanced Interior Gateway Routing Protocol
OSPF	Open Shortest Path First
QLB	Quantized Level Balance
NALU	Network Abstraction Layer Unit
PSNR	Peak Signal-to-Noise Ratio
SVEF	Scalable Video-streaming Evaluation Framework

บทที่ 1
บทนำ

1.1 ความสำคัญและที่มาของหัวข้อวิจัย

ในปัจจุบันการส่งข้อมูลประเภทวีดิทัศน์ถูกใช้มากถึงประมาณ 73 เปอร์เซ็นต์ของแบนด์วิดท์ (Bandwidth) ทั่วโลก ในปี พ.ศ. 2559 และมีแนวโน้มที่จะเพิ่มขึ้นถึง 82 เปอร์เซ็นต์ในปี 2564 [1] การจัดการและควบคุมการส่งข้อมูลประเภทวีดิทัศน์จึงมีความจำเป็นและสำคัญเป็นอย่างมาก ปัจจุบันมีมาตรฐานการเข้ารหัสวีดิทัศน์ที่มีความยืดหยุ่นในการส่งข้อมูลชื่อว่า วีดิทัศน์ปรับขนาดได้ (Scalable Video Coding) ซึ่งเป็นส่วนขยายของมาตรฐานวีดิทัศน์ประสิทธิภาพสูง (Advanced Video Coding) ในวีดิทัศน์ปรับขนาดได้ประกอบด้วยชั้นวีดิทัศน์พื้นฐาน (Base layer) และชั้นคุณภาพของวีดิทัศน์ขยาย (Enhancement layer) ที่เพิ่มคุณภาพวีดิทัศน์หลากหลายตามความต้องการได้ โดยไม่เกิดการใช้แบนด์วิดท์ที่ทับซ้อนกัน อย่างไรก็ตามการส่งข้อมูลของวีดิทัศน์ปรับขนาดได้ก็มีความซับซ้อนมาก ทำให้ต้องมีการจัดการระบบการส่งข้อมูลที่มีประสิทธิภาพ ณ ขณะนี้ได้มีรูปแบบการจัดการระบบเครือข่ายที่มีประสิทธิภาพในการจัดการและน่าสนใจที่มีชื่อว่า เครือข่ายที่กำหนดโดยซอฟต์แวร์

เครือข่ายที่กำหนดโดยซอฟต์แวร์ (Software Defined Networking) คือ รูปแบบการจัดการส่งข้อมูลโดยการโปรแกรมผ่านตัวควบคุม (Controller) ซึ่งมีแนวคิดในการแยกส่วนระดับชั้นควบคุม (Control plane) และส่วนระดับชั้นข้อมูล (Data plane) ออกจากกัน ทำให้สั่งการเพียงครั้งเดียวก็สามารถจัดการอุปกรณ์ทั้งหมดในระบบที่ตัวควบคุมเชื่อมต่ออยู่ได้ เครือข่ายที่กำหนดโดยซอฟต์แวร์เป็นการจัดการระบบเครือข่ายที่มีความน่าสนใจเป็นอย่างมากในการจัดการส่งข้อมูลของวีดิทัศน์ปรับขนาดได้ ซึ่งได้มีงานวิจัยที่ใช้เครือข่ายที่กำหนดโดยซอฟต์แวร์จัดการทำงานของวีดิทัศน์ปรับขนาดได้ ทั้งการส่งข้อมูลแบบ ยูนิคาสต์ (Unicast) และ มัลติคาสต์ (Multicast) ในส่วนของยูนิคาสต์ทั้งงานวิจัยของ S. Laga et al. [2] และ Tsung-Feng Yu et al. [3] มีการทำงานส่วนใหญ่จะใช้การแยกเส้นทางของชั้นวีดิทัศน์พื้นฐานและชั้นวีดิทัศน์ขยายแล้วส่ง โดยหลักการทำงานของ [2] จะแยกชั้นการส่งข้อมูลด้วยระดับคุณภาพของการบริการ (Quality of service level หรือ QoS-level) ส่วน [3] แยกชั้นการส่งข้อมูลด้วยความสำคัญ (Priority) โดยทั้ง 2 งานทำให้ลดอาการชะงักของวีดิทัศน์ได้ 77.3% และ 72% ตามลำดับ ส่วนการส่งข้อมูลด้วยมัลติคาสต์ได้มีงานวิจัยของ E. Yang et al. [4] สร้างรูปแบบที่มีตัวควบคุม การจัดการระบบหลายผู้ส่งและหลายผู้รับของ J. Yang et al. [5] และ N.Xue et al. [6] ก็ได้สร้างรูปแบบการจัดการการส่งข้อมูลแบบมัลติคาสต์โดยสร้างเซิร์ฟเวอร์จัดการ (Management server) ช่วยในการจัดการระบบ แต่การทำงานของวีดิทัศน์ปรับขนาดได้ ใน

รูปแบบมัลติคาสต์ข้างต้นยังคงเป็นการส่งข้อมูลด้วยหนึ่งเส้นทาง ซึ่งต่างจากการส่งแบบยูนิคาสต์ที่แยกเส้นทางในการส่งแต่ละชั้นวีดิทัศน์ได้

ในงานวิจัยนี้ นำเสนอการสร้างวีดิทัศน์ปรับขนาดได้ในรูปแบบมัลติคาสต์ที่แยกการคำนวณเส้นทางให้แต่ละชั้นวีดิทัศน์ โดยกำหนดให้ชั้นวีดิทัศน์พื้นฐานมีค่าความสำคัญสูงสุด ทำให้ได้รับการคำนวณเส้นทางก่อน ส่วนชั้นวีดิทัศน์ขยายในชั้นต่อไปจะได้รับการคำนวณต่อตามลำดับ โดยแต่ละชั้นจะถูกส่งค่าเส้นทางของชั้นวีดิทัศน์ก่อนหน้าและค่าแบนด์วิดท์ของชั้นวีดิทัศน์ของชั้นนั้น ๆ ในการพิจารณาก่อนส่งข้อมูลไปพร้อมกัน นอกจากนี้ผู้วิจัยได้ออกแบบบัฟเฟอร์สำหรับการแก้ปัญหาลำดับข้อมูลในฝั่งผู้รับที่เกิดจากการแยกเส้นทางส่งข้อมูล

1.2 การตรวจเอกสาร

ในหัวข้อนี้เป็นส่วนการนำเสนองานการจัดการการส่งข้อมูลวีดิทัศน์ปรับค่าได้บนระบบเครือข่ายที่กำหนดโดยซอฟต์แวร์ โดยจะแบ่งออกป็น 2 หัวข้อย่อย ได้แก่ การส่งข้อมูลแบบยูนิคาสต์ (Unicast) และแบบมัลติคาสต์ (Multicast)

1.2.1 การส่งข้อมูลแบบยูนิคาสต์

1) **Optimizing Scalable Video Delivery Through OpenFlow Layer-based Routing [2]**

งานวิจัยนี้เป็นการเสนอการส่งข้อมูลประเภทวีดิทัศน์ โดยการเปรียบเทียบ AVC และ SVC โดยการส่งข้อมูลแบบยูนิคาสต์ซึ่งมีการคำนวณด้วยวิธีการไดก์สตราที่พิจารณา Load และความล่าช้า

$$W(p) = Wdelay + Wload \quad (1)$$

โดยที่ W คือ น้ำหนักโดยรวม

W_{delay} คือ น้ำหนักเกิดจากความล่าช้า

W_{load} คือ น้ำหนักเกิดจาก Load

การส่งจะแยกเส้นทางชั้นวีดิทัศน์พื้นฐาน (Base layer) และชั้นวีดิทัศน์เพิ่มประสิทธิภาพ (Enhancement layer) ออกจากกัน

2) **Adaptive Routing for Video Streaming with QoS Support over SDN Networks [3]**

งานวิจัยนี้เป็นการนำเสนอการส่งข้อมูลประเภทวีดิทัศน์ปรับค่าได้ แบบยูนิคาสต์ การส่งข้อมูลวีดิทัศน์ปรับค่าได้จะถูกจัดลำดับความสำคัญให้ชั้นวีดิทัศน์โดยให้ Base layer มีความสำคัญในระดับหนึ่ง (QoS level-1) และ Enhancement layer ความสำคัญระดับสอง (QoS level-2) ทั้งสองชั้นประเภทวีดิทัศน์จะถูกส่งไปยังเส้นทางที่มีความแตกต่างกันและจะมีการตรวจสอบและเปรียบเทียบเส้นทางทั้งสอง เมื่อมีการเปลี่ยนแปลงของระบบเครือข่ายโดยจะให้ชั้นวีดิทัศน์พื้นฐานได้รับเส้นทางที่ดีที่สุด

1.2.2 การส่งข้อมูลแบบมัลติคาสต์

1) **Demonstration of OpenFlow-Controlled Network Orchestration for Adaptive SVC Video Manycast [4]**

งานวิจัยนี้เป็นการจำลองระบบเครือข่ายโดยการจัดการผู้ส่งและผู้รับหลายตำแหน่งที่มีตัวควบคุมศูนย์กลางในการจัดการระบบ ซึ่งตัวควบคุมศูนย์กลางจะเลือกผู้ส่งและผู้รับ จากนั้นจะคำนวณเส้นทางที่สั้นที่สุดจากความล่าช้าและการไหวของเวลา ส่วนการจัดการศูนย์กลางประกอบด้วย

(1) **Path Provision Module (PPM)** เป็นส่วนการจัดการ การร้องขอที่เข้ามา
(2) **Path Computation Module (PCM)** เป็นส่วนเลือกผู้ส่งและคำนวณเส้นทาง
(3) **Path Switching Module (PSM)** เป็นส่วนการจัดการเมื่อมีการเปลี่ยนแปลงของระบบให้มีการปรับค่าใหม่
(4) **Topology Management Module (TMM)** เป็นส่วนตรวจสอบระบบเครือข่ายถึงการเปลี่ยนแปลง
(5) **Traffic Engineering Database (TED)** เป็นส่วนเก็บข้อมูลสถานะต่าง ๆ

2) A Multicast Architecture of SVC Streaming Over OpenFlow Networks [5]

งานวิจัยนี้ได้ออกแบบระบบที่ใช้จัดการส่งข้อมูลวีดิทัศน์ปรับค่าได้ด้วยรูปแบบ Multicast โดยมีส่วนประกอบดังนี้

(1) Management Server ส่วนนี้จะเป็นผู้รับผิดชอบสำหรับการจัดการระบบหลายผู้รับทั้งโต้ตอบกับ ตัวแทนผู้ใช้, ตัวควบคุม และผู้บริการสื่อ การทำงานของผู้บริการการจัดการจะรับการร้องขอของตัวแทนผู้ใช้ และตรวจสอบวีดิทัศน์ที่ต้องส่งว่ามีหรือไม่ แล้วแจ้งให้ผู้บริการสื่อส่งข้อมูลออกไป และแจ้งผู้บริการการจัดการ เพื่อคำนวณและจัดการการส่งข้อมูล

(2) Media Server ทำหน้าที่ส่งข้อมูลโดยจะระบุที่อยู่ IP ผู้ส่ง และ Port ผู้ส่ง โดย Port เป็นตัวกำหนดชั้นวีดิทัศน์

(3) User Agent (UA) ผู้รับวีดิทัศน์ปลายทาง

(4) Controller ส่วนการควบคุมการส่งข้อมูล

การคำนวณเส้นทางใช้ Prim's algorithm แล้วส่งตามจำนวนชั้นวีดิทัศน์ที่อุปกรณ์ปลายทางต้องการ

3) SDM^2 Cast An OpenFlow-Based, Software-Defined Scalable Multimedia Multicast Streaming Framework [6]

งานวิจัยนี้เป็นพัฒนาต่อจาก E. Yang et al. [5] โดยเพิ่มการจัดการควบคุมการส่งข้อมูลชั้นวีดิทัศน์เพิ่มประสิทธิภาพ (Enhancement layer) ในระบบเครือข่ายที่มีความคับคั่งสูง โดยการระงับการส่งข้อมูลในชั้นวีดิทัศน์เพิ่มประสิทธิภาพ (Enhancement layer) ทำให้การส่งข้อมูลชั้นวีดิทัศน์ระดับล่างถูกส่งได้อย่างสมบูรณ์มากขึ้น

1.3 วัตถุประสงค์

1. เพื่อออกแบบการหาเส้นทางที่ควบคุมคุณภาพแต่ละขนาดวีดิทัศน์ให้พอเหมาะอย่างสมดุลด้วยระบบเครือข่ายที่กำหนดโดยซอฟต์แวร์
2. เพื่อเพิ่มประสิทธิภาพจากการกระจายเส้นทางส่งข้อมูลแบบมัลติคาสต์สำหรับการจัดการการเข้ารหัสวีดิทัศน์ปรับขนาดได้
3. เพื่อแก้ปัญหาลำดับข้อมูลวีดิทัศน์ที่ไม่จัดเรียงกันจากการกระจายเส้นทาง

1.4 ขอบเขตของงานวิจัย

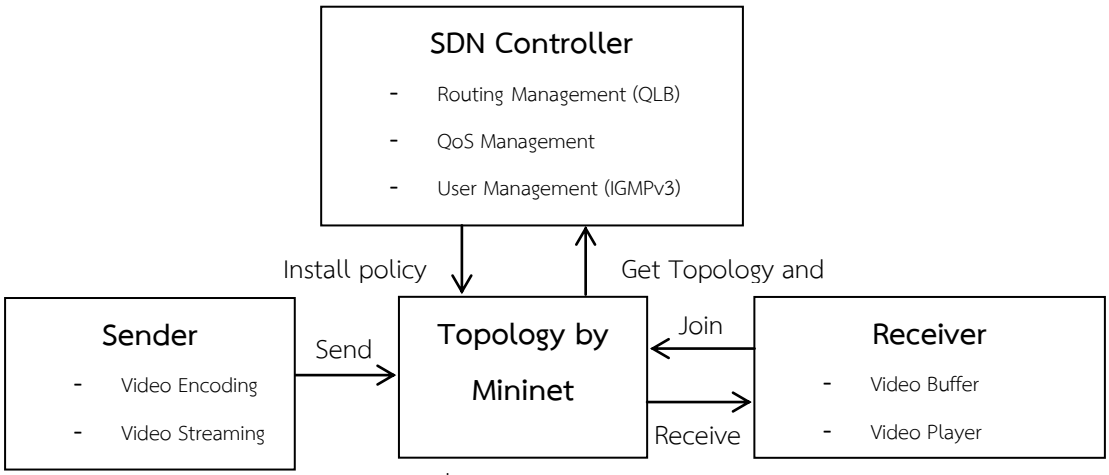
1. สร้างการส่งข้อมูลแบบมัลติคาสต์บนระบบเครือข่ายที่กำหนดโดยซอฟต์แวร์
2. สร้างวิธีการหาเส้นทางโดยการใช้ตารางความน่าจะเป็นที่พิจารณาแบนด์วิดท์, การไหวของเวลา (Jitter) และการสูญหาย แบบน้ำหนักรวม และแบบน้ำหนักสมดุล
3. ออกแบบระบบการส่งข้อมูลวีดิทัศน์ด้วยมาตรฐาน H.264/AVC และ SVC ในขนาดวีดิทัศน์ 640x360, 1280x720 และ 1920x1080

1.5 ประโยชน์ที่คาดว่าจะได้รับ

1. สามารถเลือกเส้นทางการส่งข้อมูลที่เหมาะสมกับการส่งข้อมูลประเภทวีดิทัศน์โดยเลือกระดับคุณภาพของเส้นทางที่ใช้ในการส่งอย่างสมดุล
2. สามารถลดปัญหาความคับคั่งของแบนด์วิดท์ไม่เพียงพอ เมื่อส่งวีดิทัศน์ที่มีคุณภาพสูงโดยแยกเส้นทางในการส่งข้อมูล
3. สามารถแสดงผลวีดิทัศน์อย่างถูกต้อง เมื่อมีการส่งข้อมูลแบบหลายเส้นทางด้วยบัฟเฟอร์สำหรับจัดเรียงข้อมูล

1.6 ภาพรวมของระบบ

ในหัวข้อนี้ ผู้วิจัยได้แบ่งส่วนของระบบออกเป็น 4 ส่วน ดังนี้ ผู้ส่ง (Sender) ผู้รับ (Receiver) ส่วนควบคุมระบบเครือข่าย (SDN Controller) และระบบเครือข่าย (Topology) ซึ่งแต่ละส่วนมีความสัมพันธ์ ดังรูปที่ 1.1



รูปที่ 1.1 ภาพรวมของระบบ

1) ผู้ส่ง (Sender)

(1) Video Encoding คือ การเข้ารหัสมาตรฐาน H.264 แบ่งออกเป็น 2 ประเภท ได้แก่

- AVC เข้ารหัสด้วยโปรแกรม ffmpeg
- SVC เข้ารหัสด้วย Joint Scalable Video Model Reference Software (JSVM) [7]

(2) Video Streaming ส่งข้อมูลด้วยรูปแบบมัลติคลาสต์

2) ผู้รับ (Receiver)

(1) Video sorted Buffer ใช้สำหรับ SVC-MST

(2) Video player ใช้โปรแกรม Mplayer

3) ส่วนควบคุมระบบเครือข่าย (SDN Controller)

(1) User Management (IGMPv3) ใช้จัดการสมาชิกในระบบเครือข่าย

(2) Routing Management ใช้หาเส้นทางสำหรับส่งข้อมูล

(3) QoS Management ใช้ตรวจสอบและเก็บข้อมูลคุณภาพเส้นทางในระบบเครือข่าย

4) ระบบเครือข่าย (Topology)

ในงานนี้ ผู้วิจัยใช้โปรแกรม Mininet ในการจำลองระบบเครือข่ายโดยเก็บข้อมูลต่าง ๆ ดังนี้ แบนด์วิดท์ การไหวของเวลา และการสูญหาย

1.7 โครงสร้างของวิทยานิพนธ์

วิทยานิพนธ์นี้ได้จัดวางโครงสร้างเป็น 6 บท ดังต่อไปนี้

บทที่ 1 เป็นบทนำ ซึ่งกล่าวถึงความสำคัญและที่มาของปัญหาวิจัย วัตถุประสงค์ ขอบเขตงานวิจัย การตรวจเอกสารและทบทวนวรรณกรรมที่เกี่ยวข้องกับงานวิจัย

บทที่ 2 กล่าวถึงทฤษฎี หลักการ ขั้นตอนวิธีที่จำเป็นต่อการศึกษา วิทยานิพนธ์นี้ และงานวิจัยที่เกี่ยวข้อง

บทที่ 3 – 5 กล่าวถึงการพัฒนาและการทดลองต่าง ๆ ระหว่างการวิจัย ทั้งนี้บทที่ 3 จะแสดงการเปรียบเทียบรูปแบบการส่งข้อมูลที่เกิดจากการจัดการระบบเครือข่ายแบบทั่วไปและระบบเครือข่ายที่กำหนดโดยซอฟต์แวร์ที่ผู้วิจัยพัฒนาขึ้น บทที่ 4 จะเปรียบเทียบคุณสมบัติด้านการเก็บข้อมูลและด้านการส่งข้อมูลของการเข้ารหัสวีดิทัศน์ AVC และ SVC และบทที่ 5 จะเน้นเรื่องการแก้ปัญหาที่เกิดขึ้นจากการส่งข้อมูล SVC-MST

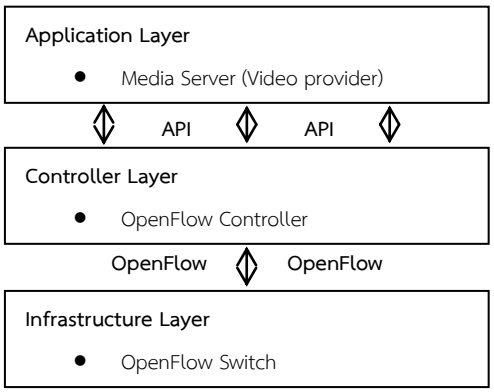
บทที่ 6 เป็นบทสรุปการวิจัยเพื่อวิทยานิพนธ์

บทที่ 2
ทฤษฎีและหลักการ

ในบทนี้จะกล่าวถึงทฤษฎีและหลักการอันเป็นความรู้พื้นฐานที่สำคัญต่อการศึกษาและการทำความเข้าใจในบทถัด ๆ ไป ซึ่งมีเนื้อหาที่เน้นการอธิบายในด้านระบบเครือข่าย และรูปแบบวีดิทัศน์มาตรฐาน H.264

2.1 เครือข่ายที่กำหนดโดยซอฟต์แวร์

เครือข่ายที่กำหนดโดยซอฟต์แวร์ (SDN) [8] คือ ระบบเครือข่ายที่มีการจัดการโดยซอฟต์แวร์ รูปแบบที่แตกต่างจากระบบเครือข่ายแบบดั้งเดิมคือการแยกส่วนระดับชั้นควบคุม (Control plane) ออกจากส่วนระดับชั้นข้อมูล (Data plane) เพื่อรวมการจัดการเครือข่ายทั้งหมดด้วยการเขียนโปรแกรมควบคุมผ่านโอเพนโฟลว์โพรโทคอล (OpenFlow Protocol) ในอุปกรณ์เครือข่าย ทำให้มีความยืดหยุ่นสูงในการจัดการอุปกรณ์ที่หลากหลาย การควบคุมระบบเครือข่ายจะถูกควบคุมเพียงจุดเดียวหรือหลายจุดตามจำนวนตัวควบคุมในระบบที่ได้สร้างขึ้น และเมื่อมีอุปกรณ์เข้ามาใหม่ก็ไม่ต้องกำหนดค่าใหม่ทีละอุปกรณ์ แต่สามารถปรับเปลี่ยนการกำหนดค่าด้วยโปรแกรมบางส่วนแล้วปรับปรุง (Update) จากตัวควบคุมได้



รูปที่ 2.1 SDN Architechture

จากรูปที่ 2.1 คือ สถาปัตยกรรมของ SDN ประกอบด้วย 3 ชั้น ดังนี้

ชั้นที่ 1 ชั้นการใช้งาน (Application Layer) คือ ชั้นที่ผู้จัดการจะติดตั้งการทำงานต่าง ๆ เช่น การบริการจัดการระบบเครือข่าย (Manage network service) การกำหนดเส้นทาง (Routing) การควบคุมการเข้าถึง (Access control) และการบริการวีดิทัศน์สตรีมมิ่งแบบมัลติคาสต์ (Video multicast streaming) เป็นต้น ในส่วนนี้จะติดต่อกับ ชั้นการควบคุม (Control Layer) ด้วยอินเตอร์เฟซการใช้งาน

ชั้นที่ 2 ชั้นการควบคุม (Control Layer) คือ ชั้นที่จัดการและควบคุมข้อมูลต่าง ๆ โดยจะทำการกำหนดกฎการส่งข้อมูลต่าง ๆ ของ Data plane ทั้งการปรับปรุงตารางการกำหนดเส้นทาง (Update routing table) ที่เปลี่ยนแปลงตามสภาพของเครือข่าย ในชั้นโครงสร้างพื้นฐาน (Infrastructure Layer) โดยการส่งข้อมูลการสั่งการกับชั้นโครงสร้างพื้นฐานนั้นผ่านโอเพนโฟลว์โพรโทคอล

ชั้นที่ 3 ชั้นโครงสร้างพื้นฐาน (Infrastructure Layer) คือ ชั้นของอุปกรณ์ที่จะประกอบไปด้วยอุปกรณ์ต่าง ๆ เช่น โอเพนโฟลว์สวิตซ์ (OpenFlow switch) แม่ข่าย (Host) และอุปกรณ์อื่น ๆ ในชั้นนี้ถูกสั่งการด้วยข้อความโอเพนโฟลว์ (OpenFlow message) ไปยังอุปกรณ์ที่รองรับโอเพนโฟลว์โพรโทคอล เช่น โอเพนโฟลว์สวิตซ์ เป็นต้น

2.2 การเข้ารหัสวีดิทัศน์ปรับขนาดได้

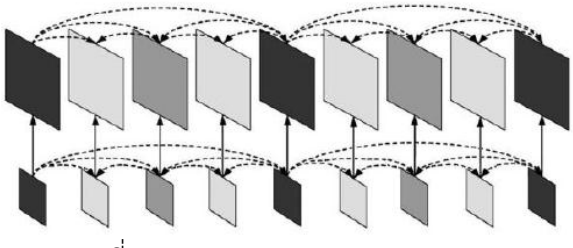
การเข้ารหัสวีดิทัศน์ปรับขนาดได้ (SVC) [9] คือ ส่วนขยายของมาตรฐานการเข้ารหัสวีดิทัศน์ H.264/AVC ซึ่งเป็นส่วนให้การส่งข้อมูลประเภทวีดิทัศน์ที่มีความยืดหยุ่นสูง โดยจะทำการปรับขยายคุณภาพของวีดิทัศน์ได้ โดยการนำวีดิทัศน์มาแบ่งเป็นชั้น ซึ่งชั้นของวีดิทัศน์จะมี 2 ประเภท ได้แก่

ประเภทที่ 1 ชั้นพื้นฐาน (Base Layer) ใน 1 วีดิทัศน์จะมีเพียง 1 ชั้นเท่านั้น โดยจะมีคุณภาพที่ต่ำสุดซึ่งทุกผู้รับที่ต้องการวีดิทัศน์จะได้รับทั้งหมด

ประเภทที่ 2 ชั้นขยาย (Enhancement Layer) เกิดจากการทำนายระหว่างชั้น (Inter layer prediction) เป็นชั้นทางเลือกของผู้รับวีดิทัศน์เนื่องจากเป็นชั้นที่ใช้สำหรับขยายคุณภาพการแสดงผลของวีดิทัศน์ แม้ว่าผู้รับจะไม่ได้รับชั้นวีดิทัศน์ในส่วนนี้ก็ยังคงแสดงผลได้ เมื่อมีการใช้งานชั้นวีดิทัศน์จะถูกนำไปรวมกับชั้นก่อนหน้า

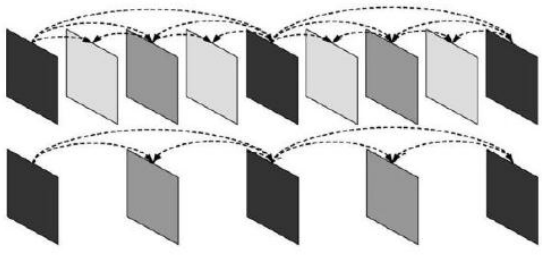
รูปแบบการปรับขยายนั้น มี 3 รูปแบบ ได้แก่

แบบที่ 1 ปรับขนาดเชิงพื้นที่ (Spatial Scalability) ใช้สำหรับความต้องการขนาดของเฟรม (Frame size) ที่แตกต่างกัน โดยชั้นพื้นฐานจะมีขนาดที่เล็กที่สุด และชั้นขยายจะถูกส่งเพื่อขยายขนาดของเฟรมให้ใหญ่ขึ้น แสดงดังรูปที่ 2.2



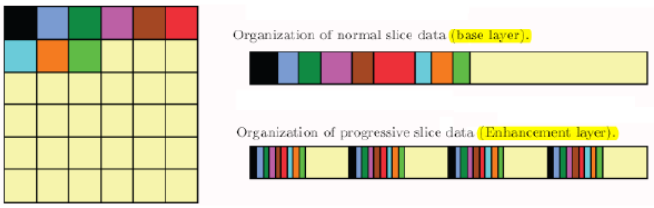
รูปที่ 2.2 Spatial Scalability of SVC

แบบที่ 2 ปรับขนาดเชิงเวลา (Temporal Scalability) ใช้สำหรับความต้องการจำนวนเฟรมในหนึ่งวินาทีที่แตกต่างกัน โดยชั้นพื้นฐานจะถูกกำหนดให้มีจำนวนเฟรมที่น้อยที่สุด และชั้นขยายจะทำหน้าทีเพิ่มจำนวนเฟรมให้มากขึ้นโดยไม่ซ้ำกับชั้นก่อนหน้า แสดงดังรูปที่ 2.3



รูปที่ 2.3 Temporal Scalability of SVC

แบบที่ 3 ปรับขนาดเชิงคุณภาพ (Quality Scalability) ใช้สำหรับคุณภาพของวีดิทัศน์ที่แตกต่างกัน โดยชั้นพื้นฐานจะส่งวีดิทัศน์ที่มีคุณภาพต่ำที่สุด และชั้นขยายจะเพิ่มคุณภาพให้กับชั้นวีดิทัศน์ก่อนหน้าโดยการส่งเฉพาะส่วนข้อมูลให้มีความละเอียดมากขึ้น แสดงดังรูปที่ 2.4



รูปที่ 2.4 Quality Scalability of SVC

2.3 โพรโทคอลในการ streaming

โพรโทคอลที่ใช้ในการ streaming มีหลากหลาย เช่น Real-time Transport Protocol (RTP), Hypertext Transfer Protocol (HTTP), Real Time Streaming Protocol (RTSP), Real Time Messaging Protocol (RTMP) และ HTTP Live Streaming (HLS) เป็นต้น ซึ่งแต่ละโพรโทคอลสนับสนุนการใช้งานที่แตกต่างกัน

2.3.1 Real-time Transport Protocol (RTP)

RTP [10] คือ โพรโทคอลที่ช่วยในการส่งข้อมูลประเภท Real-Time ที่ต้องการการตอบสนองโดยทันทีในการส่งทั้งวีดิทัศน์และเสียง โดยปกติ RTP จะส่งข้อมูลโดยใช้ User Datagram Protocol (UDP) ซึ่งทำให้สามารถส่งในรูปแบบของมัลติคลาสต์ไปยังหลายปลายทางได้ โดยมีการกำหนดตัวแปรต่าง ๆ ดังรูปที่ 2.5

<table>
<tr><td>0
1</td><td>0
2</td><td>0
3</td><td>0
4</td><td>0
5</td><td>0
6</td><td>0
7</td><td>0
8</td><td>0
9</td><td>1
0</td><td>1
1</td><td>1
2</td><td>1
3</td><td>1
4</td><td>1
5</td><td>1
6</td><td>1
7</td><td>1
8</td><td>1
9</td><td>2
0</td><td>2
1</td><td>2
2</td><td>2
3</td><td>2
4</td><td>2
5</td><td>2
6</td><td>2
7</td><td>2
8</td><td>2
9</td><td>3
0</td><td>3
1</td><td>3
2</td></tr>
<tr><td colspan="2">V</td><td>P</td><td>X</td><td colspan="4">CC</td><td>M</td><td colspan="7">Packet Type</td><td colspan="16">Sequence Number</td></tr>
<tr><td colspan="32">Time stamp</td></tr>
<tr><td colspan="32">Synchronization source (SSRC) identifier</td></tr>
<tr><td colspan="32">contributing source (CSRC) identifiers
...</td></tr>
</table>

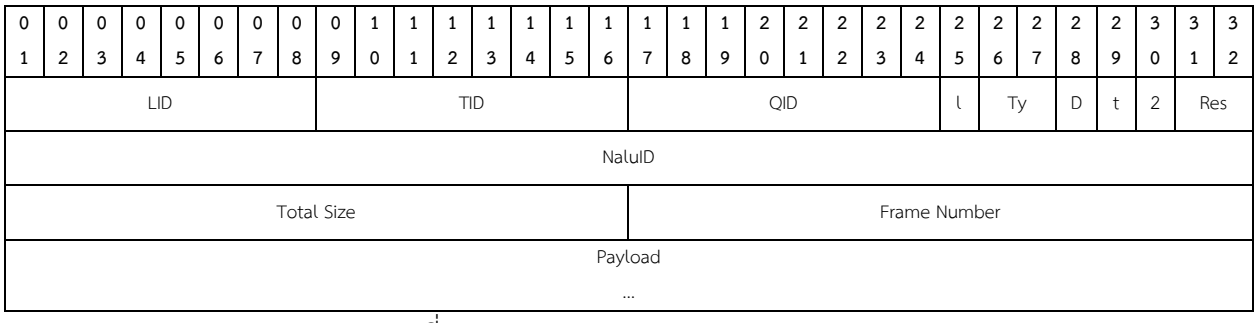
รูปที่ 2.5 RTP header

จากโครงสร้าง header ของ RTP มีขนาดโดยทั่วไปเท่ากับ 12 ไบต์ มีคุณสมบัติของตัวแปร ดังนี้

- Version (V) (2 บิต) คือ เวอร์ชันของโพรโทคอล RTP ที่ใช้ซึ่งปัจจุบันเป็นเวอร์ชัน 2
- Padding (P) (1 บิต) คือ ฟิลด์ที่บอกว่าแพ็กเก็ตนั้นได้ถูกเติมด้วยข้อมูลที่เพิ่มให้พอดีกับขนาดของแพ็กเก็ตหรือไม่ และใช้ในกระบวนการการเข้ารหัส
- Extension (X) (1 บิต) คือ ฟิลด์ที่บอกว่าเฮดเดอร์มีการขยายหรือไม่
- CSRC Count (CC) (4 บิต) คือ ส่วนบอกจำนวนของ Contribution Source Identifier ในแพ็กเก็ตโดยมีค่า CSRC ได้ตั้งแต่ 0-15
- Marker (M) (1 บิต) คือ บิตที่ใช้สำหรับระบุโดย Profile และ Specification
- Packet type (7 บิต) คือ ส่วนในการระบุชนิดของข้อมูลภายในเพย์โหลด ซึ่งถูกกำหนดจากการรูปแบบของการบีบอัดข้อมูล
- Sequence Number (16 บิต) คือ ส่วนในการบอกลำดับของแพ็กเก็ต เพื่อให้ผู้รับตรวจสอบการสูญหายของแพ็กเก็ต และสามารถใช้ในการเรียงลำดับใหม่ได้
- Timestamp (32 บิต) คือ ค่าของเวลาที่ระบุในช่วงที่ส่งออกมา ซึ่งถูกใช้ในการคำนวณการไหวของเวลา และค่า (Round Trip Time Delay: RTT)
- SSRC (32 บิต) คือ เลขประจำตัว Session เพื่อระบุผู้ส่ง
- CSRC (32 บิต ต่อ 1 ค่า) คือ ส่วนที่ใช้สำหรับผสมสื่อ ถูกใช้สำหรับกระบวนการประชุม

2.3.2 H.264 SVC header ที่กำหนดโดยโปรเจค SVEF

เนื่องจาก SVC เป็นรูปแบบวีดิทัศน์ที่มีคุณสมบัติภายในแตกต่างจากวีดิทัศน์โดยทั่วไปที่ทำให้ต้องมีส่วนในการระบุคุณสมบัติของวีดิทัศน์ SVC ซึ่งมีงานวิจัยชื่อว่า SVEF [11] กำหนดรูปแบบคุณสมบัติของข้อมูลที่ทำการส่งโดยมีการจัดเรียงตัวแปร ดังรูปที่ 2.6



รูปที่ 2.6 SVC header by SVEF

จากโครงสร้าง SVC header มีขนาดเท่ากับ 12 ไบต์ มีคุณสมบัติของตัวแปร ดังนี้

- LID, TID, QID (3 * 8 บิต) คือ ID ของชั้น SVC วีดิทัศน์ ซึ่งประกอบด้วย LID ปรับขนาดเชิงพื้นที่, TID ปรับขนาดเชิงเวลา, QID ปรับขนาดเชิงคุณภาพ
- l (1 บิต) คือ Network Abstraction Layer Unit (NALU) หน่วยสุดท้าย
- Ty (2 บิต) คือ ประเภทของ NALU
- d (1 บิต) คือ บอกความสามารถการยกเลิก NALU
- t (1 บิต) คือ บอกความสามารถการตัดการส่ง NALU
- 2 (1 บิต) คือ ชุดควบคุม NAL แบบสั้น ๆ จะถูกส่งไปในชุดเดียวกันกับ NALU ที่ติดตาม
- Res (2 บิต) คือ ส่วนสำรองสำหรับการใช้งานอื่น ๆ ในอนาคต
- NaluID (4 บิต) คือ การชดเชยของ NALU ในวีดิทัศน์ต้นแบบ
- Total size (16 บิต) คือ ขนาดของข้อมูลวีดิทัศน์รวมถึง header ด้วย
- Frame number (32 บิต) คือ จำนวนของเฟรม

2.5 โพรโทคอลในการหาเส้นทาง

โพรโทคอลในการหาเส้นทาง (Routing Protocol) คือ กระบวนการหาเส้นทางเพื่อให้ส่งข้อมูลไปยังปลายทางได้อย่างถูกต้อง ซึ่งสามารถทำได้ 2 วิธี ได้แก่

2.5.1 เส้นทางแบบคงที่ (Static Route)

เส้นทางแบบคงที่ คือ การเพิ่มเส้นทางในตารางการส่งข้อมูล (Routing Table) โดยผู้ดูแลระบบเครือข่าย (Network) เพื่อบอกให้เราเตอร์ (Router) ทราบว่าถ้าต้องการจะส่งข้อมูลไปที่อยู่เครือข่ายย่อย (Subnet Address) ที่เท่าไรและจะต้องส่งผ่านเราเตอร์ตัวไหน ค่าที่ถูกป้อนเข้าไปในตารางเลือกเส้นทางนี้มีค่าที่ตายตัว

2.5.2 เส้นทางแบบไดนามิก (Dynamic Route)

เส้นทางแบบไดนามิก คือ การใช้ซอฟต์แวร์ที่ติดตั้งมากับเราเตอร์เพื่อทำหน้าที่แลกเปลี่ยนข้อมูลข่าวสารที่เกี่ยวกับการเลือกเส้นทางระหว่างเราเตอร์ หลักการทำงาน คือเราเตอร์จะส่งตารางการส่งข้อมูลที่สมบูรณ์ของตัวเองให้กับเราเตอร์เพื่อนบ้านเรียกว่ามี โพรโทคอลในการหาเส้นทาง (Routing Protocol) ที่ใช้ในการแลกเปลี่ยนตารางการส่งข้อมูล โดยที่ผู้ดูแลเครือข่ายไม่ต้องแก้ไขข้อมูลตารางการส่งข้อมูลด้วยตัวเอง การหาเส้นทางแบบไดนามิกสามารถแบ่งออกเป็น Distance Vector และ Link State ดังตารางที่ 2.1 [12]

ตารางที่ 2.1 ความสามารถของโพรโทคอลหาเส้นทางแบบไดนามิก

Properties	Distance Vector				Link State	
	RIPv1	RIPv2	IGRP	EIGRP	OSPF	IS-IS
Speed of Covergence	Slow	Slow	Slow	Fast	Fast	Fast
Scalability	Small	Small	Small	Large	Large	Large
Use of VLSM	No	Yes	No	Yes	Yes	Yes
Resource Usage	Low	Low	Low	Medium	High	High
Implementation	Simple	Simple	Simple	Complex	Complex	Complex

หมายเหตุ:

- **Speed of Convergence** คือ ความเร็วในการหาเส้นทาง
- **Scalability** คือ ความสามารถในการรองรับระบบเครือข่ายขนาดใหญ่

- **Classless** จะรองรับ Variable Length Subnet Masking (VLSM) และการทำสรุปเส้นทาง (Route summarization) เพื่อลดขนาดของตารางการส่งข้อมูล
- **Resource Usage** คือ ทรัพยากรที่ใช้ในการหาเส้นทาง
- **Implementation and Maintenance** คือ ความรู้และทักษะที่จำเป็นต้องรู้ในการดูแลและจัดการ

1) เวกเตอร์ระยะทาง

เวกเตอร์ระยะทาง (Distance Vector) คือ รูปแบบที่เราเตอร์สามารถเรียนรู้โครงสร้างระบบเครือข่ายและเครือข่ายย่อยปลายทางต่าง ๆ โดยอาศัยการแลกเปลี่ยนตารางการส่งข้อมูลกับตารางการส่งข้อมูลการส่งข้อมูลของเพื่อนบ้าน เพื่อที่จะได้เรียนรู้ว่าเราเตอร์เพื่อนบ้านของมันรู้จักกับเครือข่ายย่อยอะไรบ้าง เพื่อที่จะอัพเดตตารางการส่งข้อมูลของตนเองว่า ถ้ามีแพ็กเก็ตที่มีที่อยู่ปลายทางเป็นเครือข่ายย่อยที่เราเตอร์เพื่อนบ้านรู้จักก็จะส่งต่อแพ็กเก็ตนั้นไปให้เราเตอร์เพื่อนบ้านตัวดังกล่าวเลย ซึ่งตัวอย่างโพรโทคอลประเภทนี้ ได้แก่

1. **Routing Information Protocol (RIP)**

ใช้เมตริก (Metric) แบบ นับจำนวนฮอป (Hop Count) ในการเลือกเส้นทาง

2.**Interior Gateway Routing Protocol (IGRP) และ Enhanced Interior Gateway Routing Protocol (EIGRP)**

ใช้ส่วนประกอบ ทั้ง 5 ตัวแปร ในการคำนวณ ได้แก่

K1= Bandwidth, K2= Load, K3= Delay, K4= Reliability และ K5 = MTU

$$Weight\,(p) = 256 * [\frac{K1*BW+(K2*BW)}{(256-Load)+K3xDelay} + \frac{K5}{(Rel+K4)}] \quad (2)$$

โดยที่ BW คือ แบนด์วิดท์ต่ำสุดเส้นทาง

Load คือ ภาระของเส้นทาง

Delay คือ ความล่าช้าของเส้นทาง

Rel คือ ความเชื่อถือได้ (Reliability)

K-Value: K1=K3=1 และ K2=K4=K5=0 และนำมาเข้าสมการ ดังนี้

$$Weight\,(p) = 256 * [\frac{K1*BW}{(256)+K3*Delay}] \quad (3)$$

2) สถานะการเชื่อมโยง

สถานะการเชื่อมโยง (Link State) คือ การที่เราเตอร์จะส่งข้อมูลอินเตอร์เฟสทั้งหมดของมันไปให้กับเราเตอร์เพื่อนบ้าน เพื่อให้เราเตอร์เพื่อนบ้านคำนวณหาเส้นทางที่ดีที่สุดเอง เราเตอร์จะไม่รู้จักแค่เราเตอร์เพื่อนบ้านแต่จะรู้จักเราเตอร์ข้างเคียงด้วยทำให้เราเตอร์สามารถเห็นภาพรวมทั้งหมดของระบบเครือข่ายเป็นอย่างดี ซึ่งตัวอย่างโพรโทคอลประเภทนี้ได้แก่ Open Shortest Path First (OSPF) และ Intermediate System to Intermediate System (IS-IS)

ใช้เมตริก คือ ค่าเฉลี่ยของความล่าช้า (Average delay) หรือแบนด์วิดท์ โดยวีดิทัศน์จะมีการแบบค่าใช้จ่าย (Cost) ดังตารางที่ 2.2

ตารางที่ 2.2 ค่าใช้จ่ายแต่ละแบนด์วิดท์

Bandwidth	Cost
100 gbps	1
40 gbps	1
10 gbps	1
1 gbps	1
100 mbps	1
10 mbps	10
1.544 mbps	64
768 kbps	133
384 kbps	266
128 kbps	781

2.6 การประกันคุณภาพการให้บริการ

การประกันคุณภาพการให้บริการ (Quality of Service: QoS) เป็นสิ่งจำเป็นสำหรับการให้บริการบนเครือข่าย เนื่องจากการประกันคุณภาพการให้บริการ QoS สามารถที่จะจัดการกับหลายปัญหา ซึ่งตัวแปรของการประกันคุณภาพการให้บริการ มีดังนี้

1) ช่องสัญญาณที่ส่งได้ (Throughput) คือ การส่งข้อมูลจากจุดหนึ่งไปยังอีกจุดหนึ่งในช่วงเวลาที่กำหนดได้สำเร็จ มีหน่วยเป็นบิตต่อวินาที (bps) ผู้ใช้บริการอินเทอร์เน็ตจึงใช้ค่าช่องสัญญาณที่ส่งได้ (Throughput) ในการรับประกันขนาดของวงจรสื่อสารหรือช่องสัญญาณน้อยที่สุดที่จะจัดสรรให้ใช้ได้ (Minimum Throughput Guarantee)

$$Throughput = \frac{\sum sent\ data\ (bit)}{time\ data\ delivery\ (s)} \quad (4)$$

2) เวลาลาเทนซี่ (Latency) หรือที่รู้จักกันคือค่าล่าช้า (Delay) ซึ่งเป็นค่าล่าช้าของการเดินทางของแพ็กเก็ตจากต้นทางไปยังปลายทาง

$$Latency = \frac{packet\ length\ (bit)}{link\ bandwidth\ (bit/s)} \quad (5)$$

3) การสูญหายของแพ็กเก็ต (Packet loss) คือ ความผิดพลาดในระดับบิตของแพ็กเก็ตในขั้นตอนการส่งข้อมูลไม่ว่าจะเป็นการสูญหายของแพ็กเก็ตระหว่างการเดินทาง หรือการถูกดรอป (Drop) ทิ้งเมื่อเกิดความคับคั่งขึ้นในเครือข่าย

$$Packet\ loss = \frac{packet\ sent - packet\ received}{packets\ sent} x\ 100\% \quad (6)$$

4) การไหวของเวลา (Jitter) คือ ค่าที่เกิดจากความผันผวนของเวลาล่าช้า (Delay variation) หรือเป็นความแตกต่างของเวลาความล่าช้าที่เกิดขึ้นกับแพ็กเก็ต

$$Jitter = \frac{\sum variation\ delay}{\sum packet\ received} \quad (7)$$

ตัวอย่าง [13] ได้ระบุตัวแปรของการประกันคุณภาพการให้บริการแบบเฉพาะ โดยมีระดับคุณภาพ ดังตารางที่ 2.3

ตารางที่ 2.3 ระดับคุณภาพของแต่ละปัจจัยของคุณภาพในแบบต่าง ๆ

Category	Throughput	Jitter	Packet loss	Delay
Excellent	100%	-	0%	-
Good	75%	< 20 ms	3%	< 150 ms
Medium	50%	20 – 50 ms	15%	150 – 400 ms
Poor	< 25%	> 50 ms	25%	> 400 ms

2.7 ขั้นตอนวิธีของไดก์สตรา

ขั้นตอนวิธีของไดก์สตรา (Dijkstra's shortest path Algorithm) [14] คือ วิธีการในการหาเส้นทางที่สั้นที่สุดวิธีหนึ่งที่ได้รับความนิยมโดยจะทำการพิจารณาที่น้ำหนักในสาย (*ew*) และเปรียบเทียบเส้นทางเพื่อเลือกเส้นทางที่มี *ew* น้อยที่สุด

จากรูปที่ 2.7 คือ การแสดง วิธีการหาเส้นทางที่สั้นที่สุดด้วยวิธีไดก์สตรา โดยมีการกำหนดตัวแปร ดังนี้

V = กลุ่มของอุปกรณ์ที่เชื่อมต่อ, *E* = กลุ่มของสายเชื่อมต่อ, *s* = ผู้ส่ง (Server), *MG* = กลุ่มผู้รับ (Client of group), *d[u]* = เส้นทางสั้นสุดของที่พิจารณา จาก *s* ถึง *u* (Distance), *p[u]* = เส้นทางสั้นสุดของที่พิจารณาก่อนหน้า จาก *s* ถึง *u* (Previous Distance)

Dijkstra_Algorithm function

Input: *G=(V,E), ew, s, MG*

Output: *MT*

```
1:  T={s}; d[s] ⟵ 0; d[u] ⟵ ∞ and p[v] ⟵ 0 for each u ≠ s, u ⊂ V
2:  insert u with key d[u] into the priority queue Q, for each u ⊂ V
3:  while (Q!=null)
4:      u ⟵ Minimum_distance(Q)
5:      for each v adjacent to u do
6:          if d[v] > d[u]+ew[u,v] then
7:              d[v] ⟵ d[u]+ew[u,v]
8:              p[v] ⟵ d[u]
9:              add v into Tree
10: Return MT, the subtree of Tree rooted at as associsted with MG
```

รูปที่ 2.7 Dijkstra shortest path Algorithm

วิธีไดก์สตรา ถูกใช้ในโพรโทคอลในการหาเส้นทางของระบบเครือข่ายดั้งเดิม ทั้ง RIP, OSPF และ IS-IS แต่โพรโทคอลในการหาเส้นทาง ทั้ง 3 แบบ ใช้การคำนวณหาเส้นทางโดยการพิจารณาเพียง Hop count, ความล่าช้า หรือแบนด์วิดท์จึงไม่เพียงพอต่อการจัดการส่งข้อมูลวีดิทัศน์ที่มีความต้องการการพิจารณาตัวแปรในเครือข่ายที่มากขึ้นอย่างการไหวของเวลา และการสูญหาย [15] ด้วย ดังตารางที่ 2.4

ตารางที่ 2.4 ความต้องการของคุณภาพในแต่ละประเภทการใช้งาน

Application Type	Throughput	Latency	Jitter	Packet loss
Email	Low	High	High	High
Web browsing	Low	High	High	High
File transfer (FTP)	Low-High	High	High	High
Chart (IM)	Low	Medium	Medium	Medium
Video Streaming	Medium-High	Medium	Medium	Medium
Video on Demand	High	Medium	Medium	Low
Voice over IP / WiFi	Low	Low	Low	Low
Video conferencing	Medium-High	Low	Low	Low

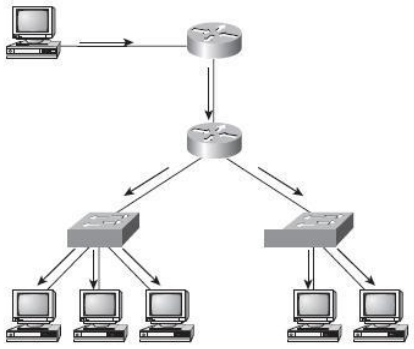
มีงานวิจัย [16] ที่นำค่าแบนด์วิดท์, ความล่าช้า และความสูญหายมารวมเข้าด้วยกัน ดังสมการที่ (8)

$$Inverse\ weight\ (p) = \frac{Bandwidth\,(p)}{Delay(p) * Loss\,(p)} \quad (8)$$

จากสมการข้างต้น ไม่มีการพิจารณาค่าการไหวของเวลาและเป็นการรวมค่าตัวแปรเข้าด้วยกันให้เป็นสูตรสำเร็จซึ่งทำให้ไม่สามารถระบุความแตกต่างของแต่ละค่าได้

2.4 มัลติคาสต์

มัลติคาสต์ (Multicast) คือ วิธีที่ลดปริมาณของข้อมูลที่ส่งผ่านเครือข่าย หนึ่งผู้ส่งไปยังหลายผู้รับอย่างมีประสิทธิภาพมากขึ้น เหมาะกับการส่งข้อมูลแบบเสียงและวีดิทัศน์ เช่น การประชุม ฝึกอบรม และการแถลงการณ์ เป็นต้น มัลติคาสต์จะส่งข้อมูลจากผู้ส่งไปยังกลุ่มเฉพาะของผู้รับที่เข้าร่วมกลุ่ม (Subscribe) โดยจะส่งแพ็กเก็ต (Packet) เพียงชุดเดียว โดยจะถูกส่งในชั้น Layer 3 แต่ละเส้นทางในเครือข่ายที่จะต้องจัดส่งข้อมูลช่วยให้ใช้ประสิทธิภาพของวีดิทัศน์ต่ำสุด และการประมวลผลสำหรับอุปกรณ์ทั้งหมด เครือข่ายมีการส่งสำเนาของข้อมูลและอุปกรณ์ ตัวอย่างการส่งข้อมูล ดังรูปที่ 2.7



รูปที่ 2.8 มัลติคาสต์

มัลติคาสต์ถูกนำมาใช้ในระบบเครือข่ายที่กำหนดโดยซอฟต์แวร์ได้อย่างมีประสิทธิภาพ เนื่องจากเครือข่ายที่กำหนดโดยซอฟต์แวร์เป็นรูปแบบที่สามารถจัดการระบบเครือข่ายแบบศูนย์กลางได้ จึงทำให้รูปแบบการส่งข้อมูลแบบมัลติคลาสต์ที่ได้รับการเพิ่มความปลอดภัยในระบบมากขึ้น เช่น GroupFlow Model [17] เป็นรูปแบบการส่งข้อมูลแบบมัลติคาสต์ ในระบบเครือข่ายที่จัดการโดยซอฟต์แวร์รูปแบบหนึ่งที่มีการเพิ่มปลอดภัยโดยสร้างโทเค็น (Token) ที่ใช้ยืนยันตัวตนระหว่างผู้รับและผู้ส่ง ซึ่ง Token จะถูกรวมกับ IGMP join message ส่งผ่านตัวควบคุมศูนย์กลางและส่งให้ผู้ส่งเพื่อให้ผู้ส่งใช้ในการตรวจสอบตัวตนก่อนจะอนุญาตและส่งข้อมูลระหว่างกัน

ในงานนี้จะปรับปรุงการทำงานภายใน GroupFlow Model ในส่วนของขั้นตอนการหาเส้นทางที่สั้นที่สุดด้วยวิธีของไดก์สตรา (Dijkstra's Shortest Past Algorithm) ให้มีความสามารถในการพิจารณาการกำหนดน้ำหนัก ทั้งเวลาลาเทนซี่ (Latency) การสูญหายของแพ็กเก็ต (Packet Loss) และ การไหวของเวลา (Jitter) สำหรับการส่งข้อมูลประเภทวีดิทัศน์สตรีมมิ่ง

2.8 GroupFlow Model

กรุ๊ปโฟร์โมเดล (GroupFlow Model) [17] เป็นรูปแบบการจัดการเครือข่ายแบบมัลติคาสต์ที่สร้างด้วย pox SDN ที่ใช้การไดก์สตรา โดยพิจารณาค่าในสายจากแบนด์วิดท์ที่เหลืออยู่และนับจำนวนของอุปกรณ์เครือข่าย นอกจากนี้การจัดการสมาชิกที่ใช้ IGMPv3 [18] กรุ๊ปโฟร์โมเดลที่ทำงานกับ OpenFlow ประกอบด้วย 4 ส่วนหลัก ดังนี้

1) GroupFlow เป็นส่วนที่ใช้ในการคำนวณหาเส้นทางให้กับสมาชิกที่ได้จาก IGMP Manager Module และใช้ข้อมูลใน FlowTracker module ตรวจสอบเส้นทางที่ใช้ในการคำนวณ นอกจากนี้เมื่อทำการคำนวณเส้นทางแล้ว ส่วนควบคุมจะติดตั้งตารางส่งข้อมูลให้กับแต่ละสวิตช์ที่ใช้ในการส่งข้อมูล

ฟังก์ชันที่ใช้ในการคำนวณหาเส้นทางประกอบด้วยฟังก์ชัน ดังนี้

- **calc_link_weights** ใช้คำนวณน้ำหนักทุกการเชื่อมต่อในเครือข่าย แต่ละการเชื่อมต่อโดยนำค่าจาก flow_tracker module (อ่านเพิ่มใน pox.openflow.flow_tracker) และ รูปแบบของน้ำหนัก มี 2 ประเภท: linear และ exponential และจะถูกนำ weighted_topo_graph มาใช้ใน calc_path_tree_dijkstras
- **Calc_path_tree_dijkstras** คำนวณหาเส้นทางโดยวิธีไดก์สตรา แล้วเก็บค่าไว้ใน path_tree_map และจะถูกใช้ใน install_openflow_rules เพื่อเลือกเส้นทางจากผู้ส่ง
- **Install_openflow_rules** เป็นส่วนใช้ในการติดตั้ง การทำงานให้กับสวิตช์แต่ละอันโดยใช้ข้อความ flowmod (of.ofp_flow_mod()) โดยดั้งเดิมจะมีระบุเพียงจุดขาออก จะไม่รู้ถึงจุดขาเข้า
- **remove_openflow_rules** ลบตารางการทำงานหรือตั้งค่าใหม่ด้วยตัวแปร of.OFPFC_DELETE
- **update_flow_placement** เป็นส่วนที่ใช้เรียกการหาเส้นทางของ calc_path_tree_dijkstra ใหม่ และติดตั้งตารางการทำงานด้วยตัวแปร install_openflow_rules ใหม่

2) IGMP Manager Module เป็นส่วนที่ใช้จัดการสมาชิก ด้วย IGMPV3 การทำงานต่าง ๆ ตั้งค่าด้วย RFC 3376 [19]

ประกอบด้วย 5 classes ดังนี้

(1) IGMPManager เก็บการตั้งค่าของ IGMP settings และ จัดการการเชื่อมโยงของเราเตอร์ IGMPv3

(2) IGMPv3Router () เป็นส่วนจัดการคำสั่งของตัวเราเตอร์ IGMPv3

(3) MulticastGroupEvent () แสดงเหตุการณ์ทำงานของทุกการเชื่อมต่อและกลุ่มบนตัวเราเตอร์

(4) MulticastMembershipRecord () ใช้เก็บการทำงานของเราเตอร์

(5) MulticastTopoEvent () เก็บเหตุการณ์ของการเปลี่ยนแปลงเครือข่าย

3) FlowTracker module เป็นส่วนที่ใช้ในการตรวจสอบ สภาพของระบบเครือข่าย โดยเน้นที่ตรวจตรวจสอบปริมาณการไหลของข้อมูลและใช้ในการพิจาณาหาเส้นทางด้วยวิธีการของ GroupFlow module แหล่งที่อยู่: pox.openflow.flow_tracker
ประกอบด้วย 2 ส่วนใหญ่ คือ

(1) FlowStats จะพบเมื่อมีการส่งข้อมูล มันมีความสามารถในการตรวจจับข้อมูลด้วยตัวแปร flow.bype และ flow.packet

(2) PortStats ใช้นับข้อมูลด้วยตัวแปร port_stat.rx_bytes ขึ้นอยู่กับการทำงานของจุดเชื่อมต่อ

ฟังก์ชันที่ใช้งานเพื่อดูค่าการทำงาน

- **get_flow_utilization_normalized** ใช้บอกอัตราส่วนร้อยละระหว่างการใช้งานและขนาดของการเชื่อมต่อมีค่าระหว่าง 0 ถึง 1
- **get_link_utilization_mbps** ใช้เก็บข้อมูลของแต่ละการเชื่อมต่อที่มีการไหลของข้อมูลจากแต่ละสวิตช์และจุดเชื่อมต่อขาออก
- **get_link_utilization_normalized** ใช้ด้วยการหารด้วยขนาดข้อมูล
- **get_max_flow_utilization** ใช้ประเมินค่าการไหลของข้อมูลสูงสุดที่ใช้
- **output_peak_usage** ใช้เก็บข้อมูลใน log.info

4) misc.groupflow_event_tracer เป็นส่วนที่เก็บเวลาการทำงานของขั้นตอนต่าง ๆ เป็นส่วนที่จะเก็บข้อมูลใน log files มี 2 ส่วนใหญ่ ดังนี้

(1) Groupflow (GroupFlowTraceEvents) จะเก็บเวลาในการหาเส้นทางมีข้อมูล ดังนี้

- **igmp_trace_event** ทริกเกอร์เหตุการณ์การหาเส้นทางด้วยแพ็กเก็ต IGMP
- **tree_calc_time** คือ เวลาในการคำนวณหาเส้นทาง ประกอบด้วย 2 ส่วน คือเวลาเริ่มต้นและเวลาสิ้นสุด
- **route_processing_time** คือ เวลาในการประมวลผล ของการหาเส้นทางประกอบด้วย 2 ส่วน คือ เวลาเริ่มต้นและเวลาสิ้นสุด

- **flow_installation_time** คือ เวลา OpenFlow rule installation ของการหาเส้นทางประกอบด้วย 2 ส่วน เวลาเริ่มต้นและเวลาสิ้นสุด
- **multicast_group** คือ ที่อยู่มัลติคาสต์ที่ใช้งานอยู่
- **src_ip** คือ ที่อยู่มัลติคาสต์ของผู้ส่ง

ฟังก์ชันที่สามารถใช้งาน

- **Get_flow_installation_time()** แสดงเวลาการติดตั้งการทำงานของการหาเส้นทาง
- **Get_route_processing_time()** แสดงเวลาการเลือกเส้นทาง
- **Get_tree_calc_time()** แสดงเวลาการคำนวณในการหาเส้นทาง

(2) IGMP_manager (IGMPTraceEvent) จะเก็บเวลาประมวลผลของแพ็กเก็ต IGMP ดังนี้

- **Router_dpid:** เราเตอร์ที่รับแพ็กเก็ต
- **Igmp_msg_type:** ประเภทของการประมวลผลข้อความ IGMP (กำหนดใน ox.lib.packet.igmpv3)
- **Igmp_group_records:** เก็บรายการ ประเภทของ IGMP ของแต่ละ multicast group
- **num_igmp_group_records:** จำนวนของกลุ่มมีอยู่ในแพ็คเก็ต IGMP
- **igmp_prcessing_time:** คือ เวลาในการประมวลผล ของ igmp ประกอบด้วย 2 ส่วน คือ เวลาเริ่มต้นและเวลาสิ้นสุด

ฟังก์ชันที่สามารถใช้งาน

- **get_igmp_processing_time()** แสดงเวลาประมวลผลของ IGMP

บทที่ 3
วิธีการส่งข้อมูลบนระบบเครือข่ายกำหนดโดยซอฟต์แวร์

ในบทนี้ ผู้วิจัยทดลองการทำงานของการส่งข้อมูลบนระบบเครือข่ายที่ควบคุมด้วยซอฟต์แวร์ โดยเปรียบเทียบรูปแบบการคำนวณคุณภาพที่ผู้วิจัยได้สร้างขึ้นกับรูปการกำหนดคุณภาพแบบระบบเครือข่ายดั้งเดิม โดยหารูปแบบที่เหมาะสมสำหรับการส่งข้อมูลประเภทวีดิทัศน์สตรีมมิ่งในแต่ละขนาดวีดิทัศน์

3.1 ความเป็นมา

ปัจจุบันการส่งข้อมูลประเภทวีดิทัศน์มีอัตราเพิ่มขึ้นอย่างต่อเนื่อง ทำให้การจัดการการส่งข้อมูลประเภทวีดิทัศน์เป็นที่น่าสนใจในการพัฒนามากขึ้น รูปแบบการหาเส้นทางของระบบเครือข่ายแบบดั้งเดิมมีการใช้โปรโทคอลหาเส้นทางแบบรวมค่าคุณภาพเส้นทางด้วยสมการอย่างโพรโทคอล Interior Gateway Routing Protocol (IGRP) และ Enhanced Interior Gateway Routing Protocol (EIGRP) หรือ สร้างตารางคุณภาพด้วยขนาดของแบนด์วิดท์ต่าง ๆ ด้วยวิธีของ Open Shortest Path First (OSPF) ซึ่งยังไม่ตอบสนองความต้องการของเส้นทางวีดิทัศน์สตรีมมิ่งที่มีความต้องการเส้นทางที่มีคุณภาพวีดิทัศน์ด้านต่าง ๆ ได้แก่ ความล่าช้า การไหวของเวลา และการสูญหาย ที่เหมาะสมในแต่ละขนาดวีดิทัศน์อย่างสมดุลกัน

ผู้วิจัยได้ออกแบบรูปแบบการคำนวณน้ำหนักของเส้นทางที่ใช้สำหรับวีดิทัศน์สตรีมมิ่งโดยการพิจารณาเส้นทางที่มีระดับคุณภาพของเส้นทางด้านต่าง ๆ อย่างสมดุลที่มีชื่อว่า (Quantized Level Balance: QLB) โดยทำการสร้างบนระบบเครือข่ายที่กำหนดโดยซอฟต์แวร์และเปรียบเทียบกับรูปแบบการหาเส้นทางของระบบเครือข่ายแบบดั้งเดิม

3.2 วิธีการทดลอง

ในการทดลองนี้ ผู้วิจัยเลือกรูปแบบการหาเส้นทางแบบ Open Shortest Path First (OSPF) เป็นตัวแทนการเลือกเส้นทางของรูปแบบระบบเครือข่ายแบบดั้งเดิม มาเปรียบเทียบกับ QLB ซึ่งใน QLB นั้นก็มีรูปแบบในการกำหนดคุณภาพอยู่ 2 แบบ ได้แก่ รูปแบบน้ำหนักรวม และรูปแบบน้ำหนักสมดุล ในการเปรียบเทียบคุณภาพวีดิทัศน์ในแต่ละรูปแบบนั้น ผู้วิจัยได้เลือกใช้วีดิทัศน์ประเภท AVC ในการทดสอบโดยมีขนาดวีดิทัศน์ดังนี้ 640x360, 1280x720 และ 1920x1080

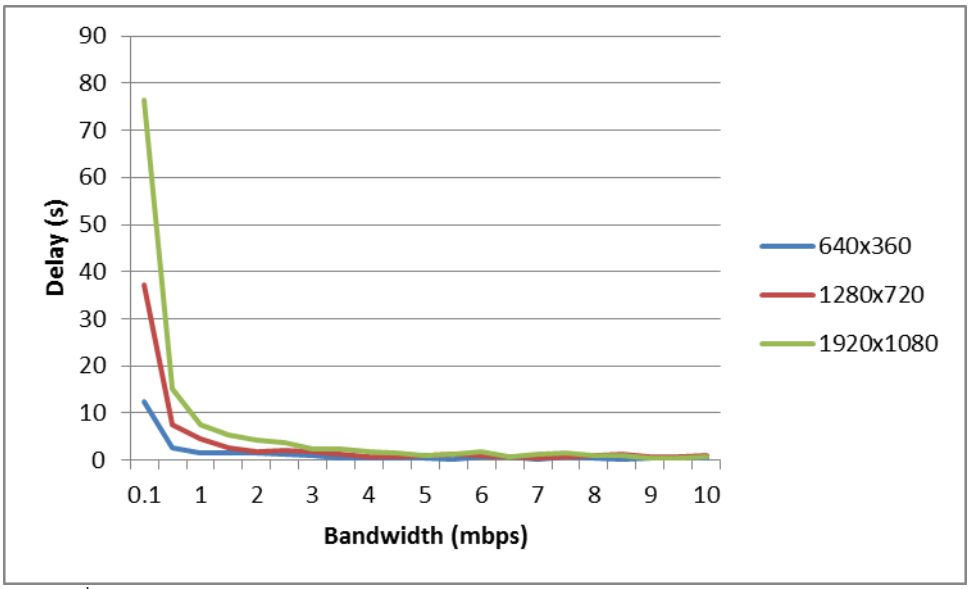
3.3 Quantized Level Balance (QLB)

ในส่วนนี้ เป็นส่วนอธิบายหลักการและการทำงานของการพิจารณาของเส้นทางด้วยวิธีการ Quantized Level Balance (QLB)

3.3.1 หลักการทำงานของ QLB

QLB คือ การหาเส้นทางด้วยคุณภาพต่าง ๆ ในระบบเครือข่ายโดยคำนวณคุณภาพเส้นทางโดยการแบ่งระดับคุณภาพด้วยค่าขีดแบ่งที่แตกต่างกันในแต่ละขนาดวีดิทัศน์ ซึ่งจะแบ่งออกเป็น 3 ระดับ ได้แก่ ระดับดี, ระดับปานกลาง และระดับต่ำ จากนั้นจะนำระดับคุณภาพที่ได้ไปพิจารณาเป็นหนึ่งระดับน้ำหนัก ซึ่งผู้วิจัยได้แบ่งการพิจารณาออกเป็น 2 วิธี ได้แก่ วิธีการหาแบบน้ำหนักรวม และวิธีการหาแบบน้ำหนักสมดุล ในงานวิจัยนี้ เป็นการพิจารณาตารางคุณภาพของแบนด์วิดท์ การไหวของเวลา และความสูญหาย โดยแต่ละค่าผู้วิจัยวัดโดยทดลองส่งวีดิทัศน์แบบเพียร์ทูเพียร์ในข้อจำกัดที่เพิ่มขึ้นและแบ่งคุณภาพแต่ละส่วน ดังนี้

1) แบนด์วิดท์ คือ ข้อจำกัดในการส่งข้อมูล หากวีดิทัศน์ได้รับการส่งข้อมูลในแบนด์วิดท์ที่ไม่เพียงพอต่อการแสดงผลก็จะทำให้เกิดปัญหาการชะงัก ซึ่งผู้วิจัยได้วัดระดับค่าแบนด์วิดท์ตั้งแต่ 0.1 เมกะบิต ถึง 10 เมกะบิต โดยวัดค่าความล่าช้าจากแพ็กเก็ตสุดท้ายของ GOP ที่มีการเข้ารหัสจัดกลุ่มข้อมูลที่ 47 เฟรม เนื่องจากสามารถคำนวณเวลาของการแสดงผลอย่างต่อเนื่องของ 47 เฟรมต่อ 1 GOP โดยใช้ตัวอย่างวีดิทัศน์จาก [20] ซึ่งแสดงผลด้วยขนาดอัตราเร็วที่ 24 เฟรมต่อวินาที นั้นหมายความว่าการรับแพ็กเก็ตหรือเฟรมสุดท้ายจะต้องใช้เวลาไม่เกิน 2 วินาที ในเฟรมสุดท้ายจึงทำให้สามารถแสดงผลได้โดยไม่มีปัญหาการชะงัก



รูปที่ 3.1 กราฟเปรียบเทียบความล่าช้าของการรับข้อมูลในแต่ละขนาดแบนด์วิดท์

จากรูปที่ 3.1 ระบุตำแหน่งของแบนด์วิดท์ที่สามารถส่งข้อมูลในแพ็กเก็ตสุดท้ายต่ำกว่า 2 วินาที และสามารถระบุระดับคุณภาพได้ ดังตารางที่ 3.1

ตารางที่ 3.1 การแบ่งระดับคุณภาพจากค่าแบนด์วิดท์

ขนาดวีดิทัศน์	ดี (Good)	ปานกลาง (Medium)	ต่ำ (Poor)
640x360	มากกว่า 0.8 เมกะบิต	0.6 - 0.8 เมกะบิต	ต่ำกว่า 0.6 เมกะบิต
1280x720	มากกว่า 2.2 เมกะบิต	2.0 - 2.2 เมกะบิต	ต่ำกว่า 2.0 เมกะบิต
1920x1080	มากกว่า 4.8 เมกะบิต	4.0 - 4.8 เมกะบิต	ต่ำกว่า 4.0 เมกะบิต

2) การไหวของข้อมูล คือ ข้อจำกัดในด้านของการมาถึงของข้อมูล ซึ่งจะส่งผลต่อการแสดงผลและการถอดรหัสวีดิทัศน์ที่ต้องอ้างอิงข้อมูลวีดิทัศน์จากเฟรมข้างเคียง ได้แก่ P-frame และ B-frame ซึ่งการตรวจสอบ ถึงแม้ว่าค่าความแปรปวนที่เกิดจากการไหวของเวลาสามารถแก้ไขด้วย การสร้างขนาดบัฟเฟอร์แต่หากมีขนาดบัฟเฟอร์มากก็ทำให้เกิดค่าความล่าช้ามากไปด้วย ดังนั้นผู้วิจัยจึงกำหนดขนาดบัฟเฟอร์ของแต่ละขนาดวีดิทัศน์ของ 640x360, 1280x720 และ 1920x1080 คือ 0.1 กิโลไบต์ 0.6 กิโลไบต์ และ 1.5 กิโลไบต์ ตามลำดับ [21], [22] และกำหนดขนาดวีดิทัศน์เป็นขนาดที่เพียงพอต่อการส่งข้อมูลโดยอ้างอิงค่าปานกลางจากตารางที่ 3.1

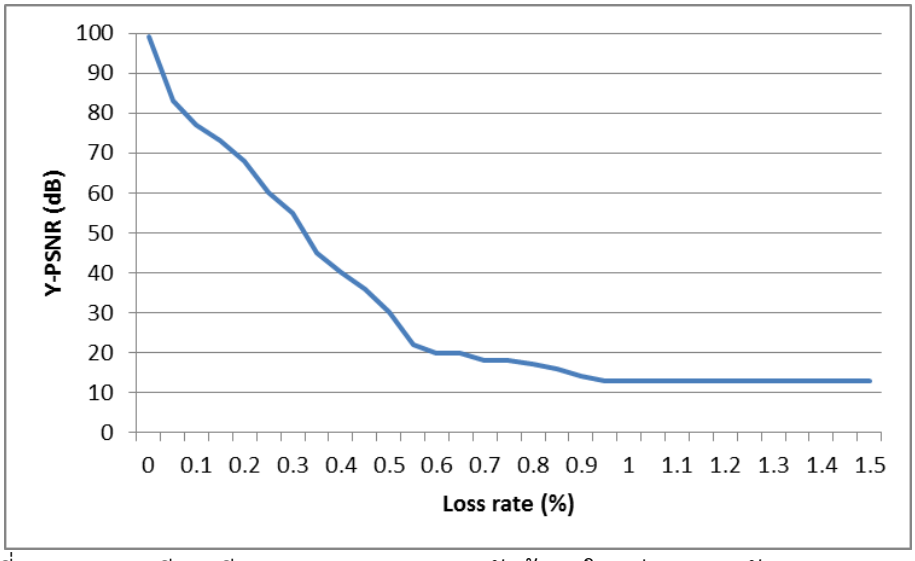
$$Buffer\ delay = \frac{Packet\ size}{Bandwidth} \quad (9)$$

ในส่วนของค่าการไหวของเวลาเป็นค่าที่ใช้การคำนวณดังสมการ (9) ซึ่งจะได้ผลลัพธ์ของแต่ละขนาดวีดิทัศน์ของ 640x360, 1280x720 และ 1920x1080 คือ 8 วินาที 10.62 วินาที และ 7.2 วินาที ตามลำดับ นอกจากนี้ปัญหาการไหวของเวลาสามารถเกิดมากเมื่อมีการส่งข้อมูลที่มีขนาดต่างกัน เช่น การส่งข้อมูลที่เกินระหว่าง I-frame ที่มีขนาดใหญ่กว่า B-frame และ P-frame ของ Group of Pictures (GOP)

ตารางที่ 3.2 การแบ่งระดับคุณภาพจากค่าการไหวของเวลา

ขนาดวีดิทัศน์	**ดี (Good)**	**ปานกลาง (Medium)**	**ต่ำ (Poor)**
640x360	ต่ำกว่า 8 วินาที	8.0 - 8.3 วินาที	มากกว่า 8.5 วินาที
1280x720	ต่ำกว่า 10.7 วินาที	10.7 – 11 วินาที	มากกว่า 11 วินาที
1920x1080	ต่ำกว่า 7.11 วินาที	7.2 – 7.5 วินาที	มากกว่า 7.5 วินาที

3) ความสูญหาย คือ ข้อจำกัดที่ส่งผลต่อการแสดงผลของวีดิทัศน์เป็นอย่างมาก เพราะเป็นส่วนที่ทำให้ได้รับข้อมูลวีดิทัศน์ที่ไม่ครบถ้วน ซึ่งผู้วิจัยได้เลือกใช้ PSNR ในการวัดระดับค่าในการแบ่งระดับขีดแบ่ง เนื่องจากเป็นค่าที่ได้รับความนิยมใช้ในการประเมินคุณภาพของวีดิทัศน์ อย่างไรก็ตามความสูญหายจากหนึ่งแพ็กเก็ตจะมีค่า PSNR ที่เกิดขึ้นเป็นช่วงกว้าง เนื่องจากตำแหน่งในการสูญหายข้อมูลของหนึ่ง GOP นั้นมีการส่งผลต่อไปยังข้อมูลในเฟรมต่อไปที่ถอดรหัสแสดงผลโดยอ้างอิงข้อมูลจากเฟรมก่อนหน้าเพื่อลดขนาดของข้อมูลวีดิทัศน์ GOP นั้น ๆ ในงานนี้ผู้วิจัยได้เลือกขนาด GOP เป็น 47 เฟรมในการทดสอบ



รูปที่ 3.2 กราฟเปรียบเทียบของ PSNR ของการรับข้อมูลในแต่ละขนาดอัตราความสูญหาย

จากรูปที่ 3.2 แสดงระดับคุณภาพที่เกิดจากการสูญหายของแต่ละขนาดวีดิทัศน์ที่แตกต่างกันมีผลลัพธ์ที่ใกล้เคียงกัน เนื่องจากการระบุค่าการสูญหายถูกกำหนดโดยเปอร์เซ็นต์ ดังนั้นจึงสามารถระบุค่าระดับคุณภาพ ดังตารางที่ 3.3

ตารางที่ 3.3 การแบ่งระดับคุณภาพจากค่าความสูญหาย

ขนาดวีดิทัศน์	**ดี** (Good)	**ปานกลาง** (Medium)	**ต่ำ** (Poor)
640x360	ต่ำกว่า 0.2 เปอร์เซ็นต์	0.2 - 0.3 เปอร์เซ็นต์	มากกว่า 0.3 เปอร์เซ็นต์
1280x720			
1920x1080			

3.3.2 การพิจารณาระดับคุณภาพของเส้นทาง

การพิจารณาแต่ละจุดเชื่อมต่อ (Node) จะเก็บค่าน้ำหนัก (Weight) ในตัวแปรรูปแบบตารางความน่าจะเป็น 3 ตำแหน่ง ซึ่งประกอบด้วยแบนด์วิดท์ การไหวของเวลา และความสูญหายตามลำดับ โดยเลขแต่ละลำดับมีค่า ดังนี้ 1 คือ คุณภาพดี 2 คือ คุณภาพปานกลาง และ 3 คือ คุณภาพต่ำ

ตารางที่ 3.4 ตารางความน่าจะเป็นของ 3 คุณภาพ และ 3 ระดับ

[1][1][1]	[1][2][1]	[1][3][1]
[1][1][2]	[1][2][2]	[1][3][2]
[1][1][3]	[1][2][3]	[1][3][3]
[2][1][1]	[2][2][1]	[2][3][1]
[2][1][2]	[2][2][2]	[2][3][2]
[2][1][3]	[2][2][3]	[2][3][3]
[3][1][1]	[3][2][1]	[3][3][1]
[3][1][2]	[3][2][2]	[3][3][2]
[3][1][3]	[3][2][3]	[3][3][3]

จากตารางที่ 3.4 ในงานวิจัยนี้ ได้กำหนดข้อจำกัดตามลำดับ ดังนี้ แบนด์วิดท์ การไหวของเวลา และความสูญหาย เนื่องจากการกำหนดค่าคุณภาพของการไหวของเวลาเป็นค่าที่ถูกสร้างโดยได้รับการอ้างจากค่าแบนด์วิดท์คุณภาพปานกลางของแต่ละระดับขนาดคุณภาพวีดิทัศน์ ดังนั้นการกำหนดของกลุ่มระดับข้อมูลในกลุ่มค่าแบนด์วิดท์สูงและต่ำจะถูกพิจารณาแยกออกไป โดยให้สามารถกำหนดค่าแบนด์วิดท์คุณภาพดีและมีค่าการไหวของเวลาระดับดีถึงปานกลาง จะหมายความว่าเส้นทางนั้นจะไม่มีปัญหาที่เกิดขึ้นกับเวลาในการส่งข้อมูลที่เกิดจาก แบนด์วิดท์และการไหวของเวลาในทางกลับกันในกลุ่มค่าแบนด์วิดท์คุณภาพต่ำและมีค่าการไหวของเวลาระดับปานกลางถึงต่ำ จะหมายความว่าเส้นทางนั้นมีปัญหาที่เกิดขึ้นกับเวลาในการส่งข้อมูลที่เกิดจากแบนด์วิดท์และการไหวของเวลาดังนั้นจึงสามารถกำหนดให้มีคุณภาพที่ต่ำเช่นกัน อย่างไรก็ตามในส่วนค่าแบนด์วิดท์ที่ดีแต่มีระดับค่าคุณภาพการไหวของเวลาที่ต่ำหรือมีค่าแบนด์วิดท์ที่ต่ำแต่มีระดับค่าคุณภาพการไหวของเวลาที่สูง จากค่าในการวัดจะไม่สามารถระบุค่าคงที่ให้กับคุณภาพได้อย่างชัดเจน ดังนั้นจึงกำหนดให้มีระดับคุณภาพปานกลางแบบพิเศษและแสดงระดับตารางคุณภาพได้จากการวิเคราะห์นี้สามารถแปลงระดับคุณภาพใหม่ได้ ดังตารางที่ 3.5

ตารางที่ 3.5 ตารางความน่าจะเป็นระดับคุณภาพที่กำหนดขึ้น

[1][1][1]	[1][1][1]	[2][2][1]
[1][1][2]	[1][1][2]	[2][2][2]
[1][1][3]	[1][1][3]	[2][2][3]
[2][1][1]	[2][2][1]	[2][3][1]
[2][1][2]	[2][2][2]	[2][3][2]
[2][1][3]	[2][2][3]	[2][3][3]
[2][2][1]	[3][3][1]	[3][3][1]
[2][2][2]	[3][3][2]	[3][3][2]
[2][2][3]	[3][3][3]	[3][3][3]

จากตารางที่ 3.5 สามารถจัดลำดับความสำคัญได้ 2 แบบ ดังนี้

1) รวมน้ำหนัก (Weight Total) คือ การนำตำแหน่งมารวมเข้าด้วยกัน ดังตารางที่ 3.5

Total_Weight function

Input: ew

Output: ew

1: ew[][][P+1] ⟵ sum(each ew[0])

2: ew[][][P+1] ⟵ (ew[][][P+1] – (L-1)) / MG

รูปที่ 3.3 ฟังก์ชันการจัดเรียงของแบบน้ำหนักรวม

สามารถแยกได้ 7 กลุ่ม ดังนี้

กลุ่มที่ 1 ผลรวมเป็น 3 ได้แก่ [1][1][1]

กลุ่มที่ 2 ผลรวมเป็น 4 ได้แก่ [1][1][2], [1][2][1], [2][1][1]

กลุ่มที่ 3 ผลรวมเป็น 5 ได้แก่ [1][1][3], [1][2][2], [1][3][1], [2][1][2], [2][2][1], [3][1][1]

กลุ่มที่ 4 ผลรวมเป็น 6 ได้แก่ [1][2][3], [1][3][2], [2][1][3], [2][2][2], [2][3][1], [3][1][2], [3][2][1]

กลุ่มที่ 5 ผลรวมเป็น 7 ได้แก่ [1][3][3], [2][2][3], [2][3][2], [3][1][3], [3][2][2], [3][3][1]

กลุ่มที่ 6 ผลรวมเป็น 8 ได้แก่ [2][3][3], [3][2][3], [3][3][2]

กลุ่มที่ 7 ผลรวมเป็น 9 ได้แก่ [3][3][3]

จากการจับกลุ่มข้างต้นทำให้แสดงลำดับกลุ่มการพิจารณาจากน้ำหนักต่ำไปสูงได้ ดังตารางที่ 3.6

ตารางที่ 3.6 ลำดับน้ำหนักของการพิจารณาแบบน้ำหนักรวม

1	1	3
2	2	4
3	3	5
2	3	6
3	4	7
6	7	9
3	5	5
4	6	6
5	7	7

2) สมดุลน้ำหนัก (weight balance) คือ การจัดกลุ่มการตรวจสอบลำดับเลขภายในตารางความน่าจะเป็นที่เรียงกันซึ่งเป็นลักษณะของการจัดเรียงที่ต้องการควบคุมคุณภาพของแต่ละด้านให้ไม่แตกต่างกันมากจนเกินไป โดยสามารถแสดงการทำงานได้ ดังรูปที่ 3.4 และ 3.5

Check_level_parameter function

Input: **ew, P, L**

Output: count

```
1:   for i=1 to P do
2:       for j=1 to L do
3:           if ew[j] = i
4:               count[i]+1
```

รูปที่ 3.4 ฟังก์ชันนับจำนวนระดับคุณภาพ

รูปที่ 3.4 คือ ฟังก์ชันที่ใช้ในการนับจำนวนของระดับคุณภาพเพื่อใช้ในการตัดสินใจในฟังก์ชันการจัดเรียงของแบบน้ำหนักสมดุลที่มีการทำงาน ดังรูปที่ 3.5 ซึ่งการทำงานของฟังก์ชันการจัดเรียงแบบน้ำหนักสมดุลนั้นจะแบ่งการพิจารณาออกเป็นจำนวนของระดับที่กำหนดขึ้นในที่นี้มีค่าสูงสุดที่ 3 กลุ่ม ในกลุ่มที่ 1 จะใช้เงื่อนไขในบรรทัดที่ 2, กลุ่มที่ 2 จะใช้เงื่อนไขในบรรทัดที่ 4 และกลุ่มที่ 3 จะใช้เงื่อนไขในบรรทัดที่ 8 ในแต่ละเงื่อนไขจะถูกแยกการพิจารณาด้วยวิธีจัดลำดับคุณภาพซึ่งสามารถจำแนกได้ ดังนี้

Balance_Weight function

Input: ew

Output: ew

```
1:   count[]  ⟵ Check_level_parameter(ew, P, L)
2:   if count[1] = L
3:       ew[][][P+1] ⟵ 1
4:   if count[2] > 0 and count[3] = 0
5:       for n=1 to P do
6:           if count[1] = P-n and count[2] = n
7:               ew[][][P+1] ⟵ n+1
8:   if count[3] > 0
9:       for n=1 to P+1 do
10:          if count[3] = n
11:              for m= 0 to P-1
12:                  if count[1] = P-n-m and count[2] = m
13:                      if n=1
14:                          ew[][][P+1] ⟵ P+m+n
15:                      else
16:                          ew[][][P+1] ⟵ P+m+(2*n)
```

รูปที่ 3.5 ฟังก์ชันการจัดเรียงแบบน้ำหนักสมดุล

กลุ่มใหญ่ที่ 1 มีเพียงเลข 1

กลุ่มที่ 1 คือ 1 = 3 ตัว ได้แก่ [1][1][1]

กลุ่มใหญ่ที่ 2 มี 2 ในตำแหน่ง

กลุ่มที่ 2 คือ 1 = 2 ตัว และ 2 = 1 ตัว ได้แก่ [1][2][1], [1][1][2], [2][1][1]

กลุ่มที่ 3 คือ 1 = 1 ตัว และ 2 = 2 ตัว ได้แก่ [1][2][2], [2][1][2], [2][2][1]

กลุ่มที่ 4 คือ 1 = 0 ตัว และ 2 = 3 ตัว ได้แก่ [2][2][2]

กลุ่มใหญ่ที่ 3 มี 3 ในตำแหน่ง

กลุ่มที่ 5 คือ 1 = 2 ตัว, 2 = 0 ตัว และ 3 = 1 ตัว ได้แก่ [1][1][3], [1][3][1], [3][1][1]

กลุ่มที่ 6 คือ 1 = 1 ตัว, 2 = 1 ตัว และ 3 = 1 ตัว ได้แก่ [1][2][3], [1][3][2], [2][1][3], [2][3][1], [3][1][2], [3][2][1]

กลุ่มที่ 7 คือ 1 = 0 ตัว, 2 = 2 ตัว และ 3 = 1 ตัว ได้แก่ [2][2][3], [2][3][2], [3][2][2]

กลุ่มที่ 8 คือ 1 = 1 ตัว, 2 = 0 ตัว และ 3 = 2 ตัว ได้แก่ [1][3][3], [3][1][3], [3][3][1]

กลุ่มที่ 9 คือ 1 = 0 ตัว, 2 = 1 ตัว และ 3 = 2 ตัว ได้แก่ [2][3][3], [3][2][3], [3][3][2]

กลุ่มที่ 10 คือ 1 = 0 ตัว, 2 = 0 ตัว และ 3 = 3 ตัว ได้แก่ [3][3][3]

จากการจับกลุ่มข้างต้น สามารถแสดงลำดับกลุ่มการพิจารณาจากน้ำหนักต่ำไปสูงได้ดังตารางที่ 3.7 โดยอ้างอิงตารางระดับคุณภาพ จากตารางที่ 3.5

ตารางที่ 3.7 ลำดับน้ำหนักของการพิจารณาแบบน้ำหนักสมดุล

1	1	3
2	2	4
5	5	7
2	3	6
3	4	7
6	7	9
3	8	8
4	9	9
7	10	10

จากตารางที่ 3.6 และ 3.7 เนื่องจากกลุ่มที่มีสีเทาเป็นกลุ่มที่มีค่าปานกลางของแบนด์วิดท์และการไหวของเวลาแบบปานกลางพิเศษซึ่งไม่สามารถกำหนดค่าได้อย่างชัดเจนจึงถูกใช้เป็นการพิจารณาในลำดับที่ 2 หลังจากถูกพิจารณาในค่าอื่น ๆ

3.4 การหาเส้นทางของ QLB บน GroupFlow

Quantized Level Balance (QLB) คือ การหาเส้นทางที่คำนวณเส้นทางโดยการเลือกเส้นทางที่มีคุณภาพของแต่ละด้านที่ไม่แตกต่างกันมากจนเกินไป โดยการนำค่าของแต่ละด้านมาแบ่งตามค่าขีดแบ่ง (Threshold) ที่กำหนดไว้ การปรับแต่ง GroupFlow ประกอบด้วย 2 ส่วน ดังนี้

1. GroupFlow module

1) ในส่วนของ install_openflow_rules

โดยดั้งเดิมจะเป็นการกำหนดการไหลของข้อมูลให้กับข้อความ Flowmod เพียง action.port แต่การตรวจสอบ คุณภาพของการเชื่อมต่อระหว่างอุปกรณ์จะต้องเปรียบเทียบ ข้อมูลขาออกของสวิตช์ที่ 1 และข้อมูลขาเข้าของสวิตช์ที่ 2 ดังนั้นจะต้องเก็บข้อมูลขาเข้าที่ได้ข้อมูลจากการเชื่อมต่อ สวิตช์ที่ 1 ข้อมูลขาออก และสวิตช์ที่ 2

2) กำหนดน้ำหนักบนเส้น calc_link_weights

โดยดั้งเดิมจะเป็นการกำหนดตัวแปร link_weight เท่ากับ 1 เพื่อใช้เป็นวิธีการ Hop count แต่ในการทำงานของ QLB พิจารณาคุณภาพที่หลากหลายจึงเก็บค่าแบนด์วิดท์ที่พร้อมใช้งาน การไหวของเวลา และความสูญหายในส่วนนี้เป็นการนำค่าจาก FlowTracker module

3) คำนวณเส้นทาง calc_path_tree_dijkstras

ดั้งเดิมจะพิจารณาโดยการ กำหนดเส้นทางโดยวิธีไดก์สตรา แต่ในการทำงานของ QLB จะมีการสร้าง balance_weight และ total_weight ที่ใช้ตัดสินใจเลือก path

(1) **quantized_to_3level** ใช้ในการแบ่งระดับคุณภาพ โดยมีค่าของแต่ละค่าขีดแบ่งที่แตกต่างกัน

(2) **balance_weight** พิจารณาน้ำหนักโดยใช้ quantized_to_3level ในการแบ่งค่าขีดแบ่งและ check_level_parameter ใช้ในการนับจำนวนระดับคุณภาพเพื่อให้ balance_weight สามารถจำแนกกลุ่มน้ำหนักได้

(3) **total_weight** พิจารณาน้ำหนักโดยใช้ quantized_to_3level ในการแบ่งค่าขีดแบ่งเหมือนกับ balance_weight และจะนำผลของการแบ่งมารวมกันเพื่อเป็นกลุ่ม

2. FlowTracker module

โดยดั้งเดิมจะมีการพิจารณาเพื่อเก็บเพียงการใช้งานวีดิทัศน์ในแต่ละการเชื่อมต่อจากข้อมูลที่ไหลออกด้วยการตรวจสอบของ Flow state ในแต่ละจุดเชื่อมต่อของแต่ละสวิตช์ในส่วนของ ความล่าช้า การไหวของเวลา และความสูญหาย โดยใช้ flowstate เช่นกัน ดังนี้

- Delay คือ การนับเวลาของการพบ ข้อมูลขาออกของสวิตช์ที่ส่งออกจนถึงเวลาข้อมูลขาเข้าสวิตช์ที่รับข้อมูล
- Jitter คือ เวลาที่เปลี่ยนแปลงของความล่าช้าในการเชื่อมต่อนั้น ๆ
- Loss คือ การเปรียบเทียบระหว่างข้อมูลขาออกของสวิตช์ที่ส่งข้อมูลออก กับข้อมูลขาเข้าของสวิตช์ที่รับข้อมูล

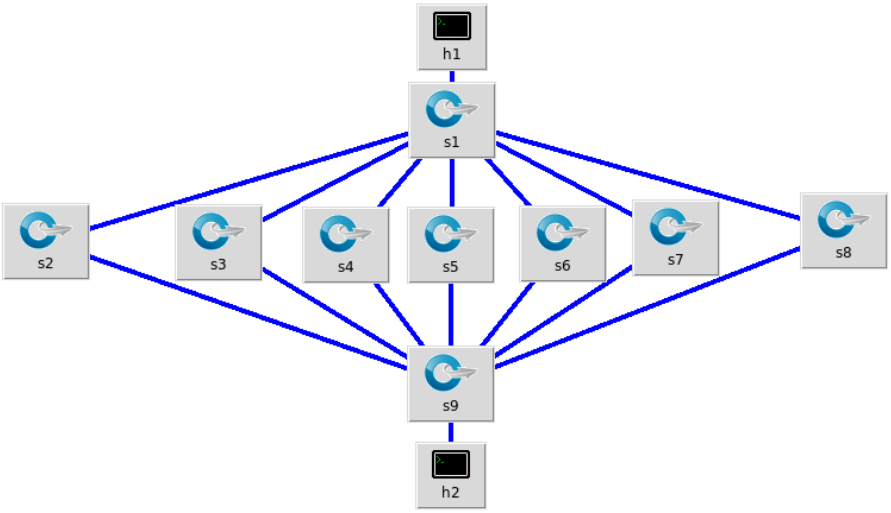
3.5 เปรียบเทียบผลการทำงานระหว่าง OSPF และ QLB

ในหัวข้อนี้ผู้วิจัยนี้เป็นการเปรียบเทียบการแสดงผลลัพธ์ของวีดิทัศน์ในฝั่งผู้รับ ซึ่งผู้วิจัยได้สร้างระบบเครือข่ายด้วยโดยแกรม Mininet ที่สามารถตั้งค่าคุณภาพของเส้นทางได้ ดังนี้ แบนด์วิดท์ ความล่าช้า การไหวของเวลา และความสูญหาย ซึ่งผู้วิจัยได้เลือกกลุ่มคุณภาพที่แตกต่างกันเพื่อให้เห็นถึงการเลือกเส้นทางที่แตกต่างกันของแต่ละรูปแบบ โดยการตัดลำดับค่าคุณภาพที่มีน้ำหนักเท่ากันของแบบน้ำหนักรวมในตารางที่ 3.3 และแบบน้ำหนักสมดุลในตารางที่ 3.4 จากน้อยไปมากและจากมากไปน้อย นอกจากนี้ก็ยังตัดค่าที่เป็นแบนด์วิดท์และการไหวของเวลาแบบปานกลางพิเศษ เนื่องจากการวัดค่าเหล่านี้ยังคงเป็นค่าที่ไม่คงที่ ดังนั้นจะเหลือค่าที่นำมาเปรียบเทียบเฉพาะกรอบสีขาว ดังตารางที่ 3.8

ตารางที่ 3.8 เส้นทางที่ใช้ในการทดลอง

[1][1][1]	[1][2][1]	[1][3][1]
[1][1][2]	[1][2][2]	[1][3][2]
[1][1][3]	[1][2][3]	[1][3][3]
[2][1][1]	[2][2][1]	[2][3][1]
[2][1][2]	[2][2][2]	[2][3][2]
[2][1][3]	[2][2][3]	[2][3][3]
[3][1][1]	[3][2][1]	[3][3][1]
[3][1][2]	[3][2][2]	[3][3][2]
[3][1][3]	[3][2][3]	[3][3][3]

จากค่าผลรวมของแต่ละเส้นทางในแต่ละระดับคุณภาพ 7 เส้นทาง ดังรูปที่ 3.6 ซึ่งมีการกำหนดระดับคุณภาพ ดังนี้ [1][1][3], [1][2][3], [2][3][1], [2][2][2], [2][3][2], [2][1][3] และ [2][2][3] ผู้วิจัยทดลองโดยการส่งข้อมูลจาก h1 ไปยัง h2 ด้วยรูปแบบในการหาเส้นทาง ได้แก่ OSPF, QLB แบบน้ำหนักรวม และ QLB แบบน้ำหนักสมดุล



รูปที่ 3.6 แบบจำลองระบบเครือข่าย

1. OSPF มีการจัดระดับคุณภาพได้ 2 ระดับ ดังนี้

กลุ่มที่ 1 คือ [1][1][3] และ [1][2][3]

กลุ่มที่ 2 คือ [2][3][1], [2][2][2], [2][3][2], [2][1][3] และ [2][2][3]

OSPF ได้เลือกเส้นทาง [1][1][3] หรือคุณภาพต่ำกว่า [1][2][3] ซึ่งเป็นเส้นทางที่ไม่มีปัญหาความล่าช้าในการส่งวีดิทัศน์หรือปัญหาการชะงักของข้อมูลแต่ข้อมูลวีดิทัศน์จะมีปัญหาการสูญหายและแสดงผลไม่ครบโดยมีค่าเฉลี่ย PSNR เท่ากับ 24 เดซิเบล โดยมีภาพตัวอย่างความสูญเสียดังรูปที่ 3.8

2. QLB แบบน้ำหนักรวม มีการจัดระดับคุณภาพได้ 3 ระดับ ดังนี้

กลุ่มที่ 1 คือ [1][1][3] และ [1][2][3]

กลุ่มที่ 2 คือ [2][2][2] [2][3][1] และ [2][1][3]

กลุ่มที่ 3 คือ [2][3][1] และ [2][2][3]

QLB แบบน้ำหนักรวม ได้เลือกเส้นทางที่มีผลรวมของคุณภาพเส้นทางที่ดีที่สุด คือ [1][1][3] หรือ [1][2][3] ซึ่งเป็นเส้นทางที่ไม่มีปัญหาความล่าช้าในการส่งวีดิทัศน์หรือปัญหาการชะงักของข้อมูลแต่มีปัญหาในด้านการสูญหายของข้อมูลเหมือนกับรูปแบบการส่งข้อมูลของ OSPF จึงมีค่าเฉลี่ย PSNR ที่เท่ากับ 24 เดซิเบล

3. QLB แบบน้ำหนักสมดุล มีการจัดระดับคุณภาพได้ 4 ระดับ ดังนี้

กลุ่มที่ 1 คือ [2][2][2]

กลุ่มที่ 2 คือ [1][1][3] และ [1][2][3]

กลุ่มที่ 3 คือ [2][3][1] และ [2][1][3]

กลุ่มที่ 4 คือ [2][2][3] และ [2][3][2]

QLB แบบน้ำหนักสมดุล ได้เลือกเส้นทางควบคุมคุณภาพอย่างสมดุล คือ [2][2][2] ซึ่งเป็นเส้นทางที่ไม่มีปัญหาความล่าช้าในการส่งวีดิทัศน์ ปัญหาการชะงักของข้อมูลและมีปัญหาในด้านการสูญหายของข้อมูลในระดับปานกลางที่ผู้บริการกำหนดขึ้นในการแบ่งระดับคุณภาพ โดยมีค่าเฉลี่ย PSNR เท่ากับ 62 เดซิเบล ดังมีภาพตัวอย่างความสูญเสีย ดังรูปที่ 3.9

รูปที่ 3.7 ภาพวีดิทัศน์ตัวอย่างขนาด 1280x720 จากผู้ส่ง

รูปที่ 3.8 ภาพวีดิทัศน์ตัวอย่างขนาด 1280x720 จากการสูญหายมาก (PSNR = 24 dB)

รูปที่ 3.9 ภาพวีดิทัศน์ตัวอย่างขนาด 1280x720 จากการสูญหายเพียงเล็กน้อย (PSNR = 62 dB)

3.6 วิเคราะห์และสรุปผล

จากการทดลองการส่งข้อมูลของ OSPF, QLB แบบน้ำหนักรวม และ QLB แบบน้ำหนักสมดุล ผลลัพธ์คือ OSPF มีการพิจารณาเพียงแบนด์วิดท์ทำให้ไม่สามารถรับรู้ถึงคุณภาพการไหวของเวลาและความสูญหาย จึงทำให้การแสดงผลของวีดิทัศน์ไม่สมบูรณ์ ส่วนการพิจารณาในรูปแบบ QLB แบบน้ำหนักรวม ก็ยังคงมีปัญหาที่ไม่สามารถแบ่งแยกการเลือกเส้นที่มีคุณภาพด้านใดด้านหนึ่งมีคุณภาพที่ดี แต่กลับมีปัญหาในคุณภาพอีกด้านหนึ่งทำให้รูปแบบ QLB แบบน้ำหนักสมดุลสามารถเลือกเส้นทางได้ดีกว่า OSPF และ QLB แบบน้ำหนักรวม เนื่องจากการจัดระดับคุณภาพวีดิทัศน์ QLB แบบน้ำหนักสมดุล แบ่งลำดับที่ให้ความสำคัญกับการควบคุมซึ่งดีกว่าระดับวีดิทัศน์การเลือกเส้นทางที่มีคุณภาพด้านใดด้านหนึ่งที่ดีมากเกินไปแต่กลับมีคุณภาพอีกด้านที่ต่ำ

บทที่ 4
วิธีการ SVC-MST แบบพิจารณาเส้นทางก่อนหน้า

ในบทนี้ จะเปรียบเทียบคุณสมบัติของการเข้ารหัสวีดิทัศน์มาตรฐาน H.264 โดยเปรียบเทียบระหว่าง Advanced Video Coding (AVC) และ Scalable Video Coding (SVC) ซึ่งเน้นที่การเก็บข้อมูลและความต้องการของการส่งข้อมูลในระดับต่าง ๆ ดังนั้นผลการทดลองจะแสดงประโยชน์ด้านการใช้งานวีดิทัศน์ประเภทต่าง ๆ ที่เหมาะสม ตามด้วยบทวิเคราะห์ และสรุปผล

4.1 ความเป็นมา

การเข้ารหัสวีดิทัศน์มาตรฐาน H.264 แบบดั้งเดิมคือ AVC และมีส่วนขยายรูปแบบที่เรียกว่า SVC เป็นส่วนขยายที่มีความสามารถในการแบ่งคุณภาพวีดิทัศน์ได้ถึง 3 คุณสมบัติ ได้แก่ ปรับขนาดเชิงพื้นที่ ปรับขนาดเชิงเวลา และปรับขนาดเชิงคุณภาพ ทำให้ SVC เป็นหนึ่งในรูปแบบวีดิทัศน์ที่มีความสามารถในการส่งข้อมูลที่มีความยืดหยุ่นมากขึ้น ทำให้ SVC สามารถแบ่งรูปแบบการส่งข้อมูลออกเป็น 2 ประเภท ได้แก่ Scalable Video Coding-Single Session Transmission (SVC-SST) และ Scalable Video Coding-Multiple Session Transmission (SVC-MST) ซึ่งรูปแบบการส่งข้อมูลแบบ SVC-SST จะมีการทำงานที่เหมือนกับรูปแบบการส่งข้อมูลโดยทั่วไป ซึ่งจะเป็นการส่งข้อมูลตามขนาดความต้องการของผู้รับในขนาดใดขนาดหนึ่ง เมื่อผู้รับมีความต้องการคุณภาพที่หลากหลายคุณภาพ การส่งข้อมูลแบบ SVC-SST ก็จะทำการส่งหลายขนาดวีดิทัศน์ที่เป็นข้อมูลเต็มของขนาดวีดิทัศน์นั้น ๆ ทำให้เกิดการทับซ้อนกันของข้อมูลเหมือนกับการส่งข้อมูลแบบทั่วไปอย่าง AVC แต่การทำงานในรูปแบบ SVC-MST จะแยกชั้นของวีดิทัศน์ออกแล้วทำการส่งเมื่อมีผู้รับที่ต้องการคุณภาพวีดิทัศน์ที่หลากหลายก็จะเลือกรับจำนวนชั้นวีดิทัศน์นั้น ๆ ตามความต้องการได้เลย อย่างไรก็ตาม SVC-MST มีรูปแบบการทำงานที่ซับซ้อนมากกว่าการส่งข้อมูลแบบทั่วไป อย่าง AVC และ SVC-SST แม้ว่าในงานวิจัยส่วนใหญ่ที่พัฒนาการส่งข้อมูลแบบ SVC-MST สามารถกำหนดเส้นทางให้แต่ละชั้นวีดิทัศน์ได้ [4]–[6] แต่การพิจารณาแต่ละชั้นวีดิทัศน์ในชั้นถัดไปไม่ได้รับการคำนวณโดยพิจารณาเส้นทางในชั้นก่อนหน้า ทำให้ยังคงมีปัญหาในกรณีการส่งข้อมูลในขนาดแบนด์วิดท์ที่ต่ำอยู่ ดังนั้นผู้วิจัยจึงได้สร้าง SVC-MST แบบพิจารณาเส้นทางก่อนหน้าและนำเสนอการเปรียบเทียบคุณสมบัติกับ AVC, SVC-SST และ SVC-MST แบบไม่พิจารณาเส้นทางก่อนหน้า

4.2 วิธีการทดลอง

ผู้วิจัยได้เปรียบเทียบรูปแบบการส่งข้อมูลของ SVC-MST แบบพิจารณาเส้นทางก่อนหน้ากับการส่งข้อมูลแบบ AVC, SVC-SST และ SVC-MST แบบไม่พิจารณาเส้นทางก่อนหน้า โดยผู้วิจัยได้แบ่งการเปรียบเทียบออกเป็น 2 ส่วน ได้แก่ ปริมาณการเก็บข้อมูล และการส่งข้อมูลโดยเปรียบเทียบในระดับความละเอียดที่แตกต่างกัน 3 ระดับ ได้แก่ 640x360, 1280x720 และ 1920x1080 และการส่งข้อมูลแบบมัลติคลาสต์ ที่ผู้รับมีความต้องการแบบเพียงหนึ่งขนาดวีดิทัศน์และมีความต้องการแบบหลายขนาดในเวลาเดียวกัน

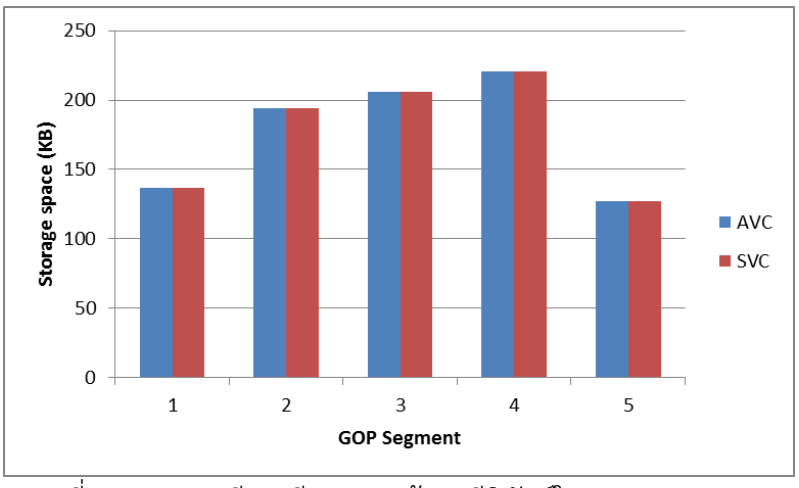
4.3 เปรียบเทียบวีดิทัศน์

4.3.1 ปริมาณการเก็บข้อมูล

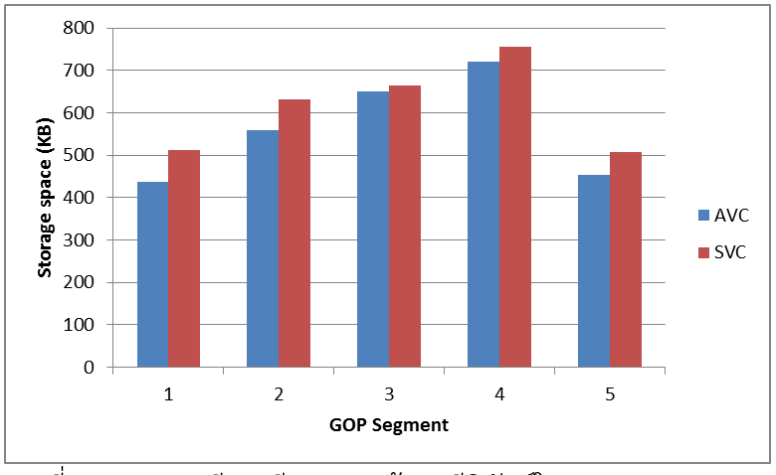
ผู้วิจัยได้ทดสอบเปรียบเทียบขนาดการเก็บข้อมูลวีดิทัศน์ของ AVC และ SVC จำนวน 5 กลุ่มลำดับข้อมูลที่มีการจัดกลุ่มประเภทของเฟรมที่เหมือนกัน โดยทดสอบ 3 ขนาดวีดิทัศน์ ได้แก่ 640x360, 1280x720 และ 1920x1080 โดยแบ่งการทดลองออกเป็น 2 กรณี ได้แก่ การเก็บวีดิทัศน์เพียงหนึ่งขนาด และการเก็บวีดิทัศน์หลายขนาด

1) กรณีการเก็บวีดิทัศน์เพียงหนึ่งขนาด

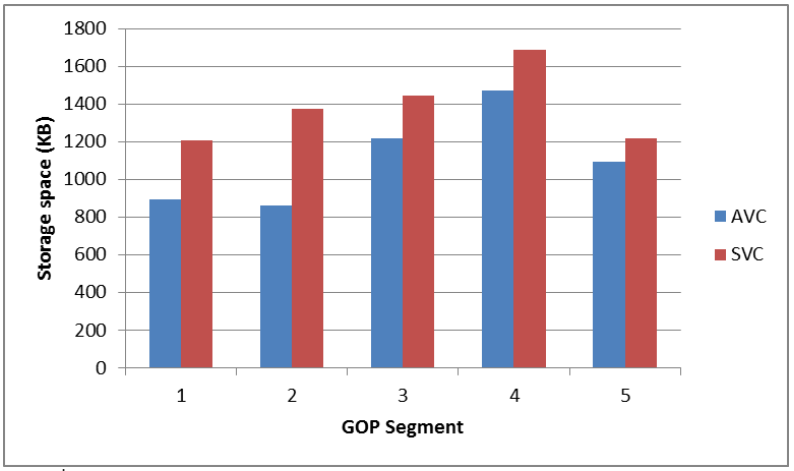
ในกรณีนี้เป็นการทดสอบโดยเปรียบเทียบการเก็บข้อมูลวีดิทัศน์ที่มีขนาดแตกต่างกัน 3 ขนาด ได้แก่ 640x360, 1280x720 และ 1920x1080 ซึ่งมีผลลัพธ์ ดังรูปที่ 4.1 จะเห็นได้ว่าขนาด 640x360 จะไม่มีความแตกต่างกันระหว่าง AVC และ SVC เนื่องจากชั้นพื้นฐานของ SVC ก็คือวีดิทัศน์ประเภท AVC ที่มีคุณภาพต่ำที่สุด ในรูปที่ 4.2 ของขนาด 1280x720 จะเริ่มเห็นความแตกต่างของขนาดวีดิทัศน์ของ SVC ที่มากกว่า AVC และมีขนาดความห่างที่มากขึ้น ในรูปที่ 4.3 ของขนาด 1920x1080 เนื่องจากภายในเฟรมของ SVC จะต้องมี overhead ที่ใช้ในการแบ่ง Network Abstraction Layer Unit (NALU) ออกเป็นชั้นคุณภาพวีดิทัศน์ทำให้เมื่อรวมข้อมูลเข้าด้วยกันก็จะมีขนาดที่ใหญ่กว่าขนาดวีดิทัศน์มาตรฐานอย่าง AVC เมื่อคำนวณค่าเฉลี่ยของการเพิ่มขึ้นเป็นเปอร์เซ็นต์จะได้ผลลัพธ์ของแต่ขนาดวีดิทัศน์ของ 5 กลุ่มลำดับข้อมูล ได้แก่ 640x360, 1280x720 และ 1920x1080 เท่ากับ 0%, 8.93% และ 28.72% ตามลำดับ



รูปที่ 4.1 กราฟเปรียบเทียบขนาดข้อมูลวีดิทัศน์ในขนาด 640x360



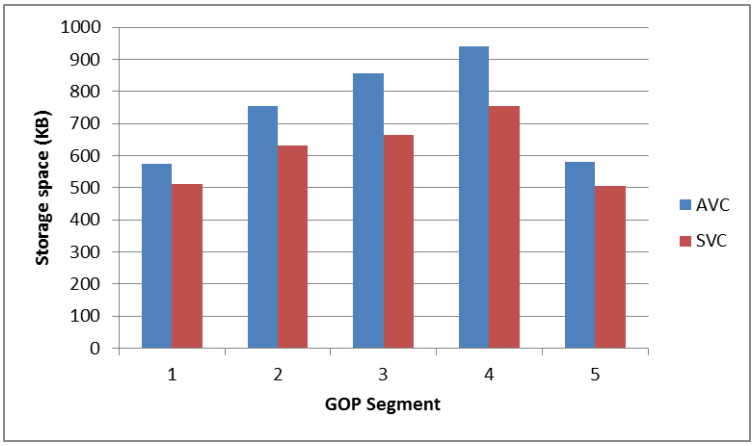
รูปที่ 4.2 กราฟเปรียบเทียบขนาดข้อมูลวีดิทัศน์ในขนาด 1280x720



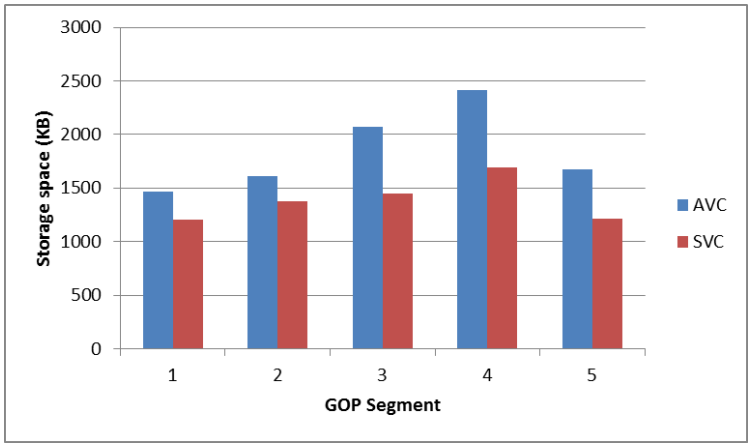
รูปที่ 4.3 กราฟเปรียบเทียบขนาดข้อมูลวีดิทัศน์ในขนาด 1920x1080

2) กรณีการเก็บวีดิทัศน์หลายขนาด

ในกรณีนี้เป็นการทดสอบโดยเปรียบเทียบการเก็บกลุ่มข้อมูลวีดิทัศน์ที่มีขนาดแตกต่างกัน ได้แก่ กลุ่ม (640x360 และ 1280x720) และกลุ่ม (640x360, 1280x720 และ 1920x1080) ซึ่งมีผลลัพธ์ ในรูปที่ 4.4 ของกลุ่ม (640x360 และ 1280x720) และ รูปที่ 4.5 ของกลุ่ม (640x360, 1280x720 และ 1920x1080) แสดงให้เห็นว่าการเก็บข้อมูลของ SVC มีการเก็บข้อมูลโดยรวมที่น้อยกว่า AVC เนื่องจากการเก็บข้อมูลของ AVC จะเก็บข้อมูลเต็มของขนาดวีดิทัศน์นั้น ๆ ทำให้เกิดปัญหาข้อมูลที่ทับซ้อนกัน แต่ในส่วนการเก็บข้อมูลที่มีคุณภาพวีดิทัศน์ที่หลากหลายของ SVC จะเก็บข้อมูลโดยใช้ขนาดสูงสุดของวีดิทัศน์นั้น ๆ เมื่อคำนวณค่าเฉลี่ยของการเพิ่มขึ้นเป็นเปอร์เซ็นต์จะได้ผลลัพธ์ของแต่ละขนาดวีดิทัศน์ของ 5 กลุ่มลำดับ แสดงให้เห็นว่า SVC สามารถลดพื้นที่การเก็บข้อมูลที่หลากหลายได้ดีกว่า AVC ในกลุ่ม (640x360 และ 1280x720) และ กลุ่ม (640x360, 1280x720 และ 1920x1080) เท่ากับ 22.14% และ 42.22% ตามลำดับ



รูปที่ 4.4 กราฟเปรียบเทียบขนาดข้อมูลวีดิทัศน์ในแต่ละกลุ่มข้อมูล (640x360 และ 1280x720)

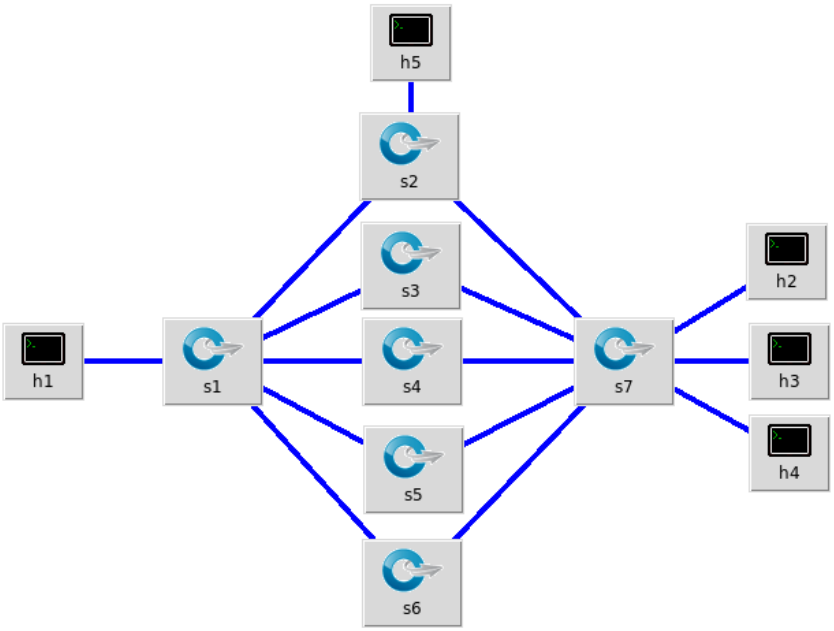


รูปที่ 4.5 กราฟเปรียบเทียบขนาดข้อมูลวีดิทัศน์ในแต่ละกลุ่มข้อมูล (640x360, 1280x720 และ 1920x1080)

จากการทดลองเปรียบเทียบขนาดพื้นที่ของทั้ง AVC และ SVC จะเห็นได้ว่า AVC มีการใช้พื้นที่ในกรณีการเก็บวีดิทัศน์ในขนาดใดขนาดหนึ่งดีกว่า SVC อย่างไรก็ตาม SVC กลับมีการใช้พื้นที่ในกรณีการเก็บวีดิทัศน์หลายขนาดดีกว่า AVC เนื่องจาก AVC จะเก็บข้อมูลเต็มของทุกขนาดวีดิทัศน์ต่างกับ SVC จะมีขนาดเท่ากับขนาดสูงสุดของกลุ่มวีดิทัศน์นั้น ๆ

4.3.2 การใช้งานแบนด์วิดท์ของ AVC, SVC-SST และ SVC-MST

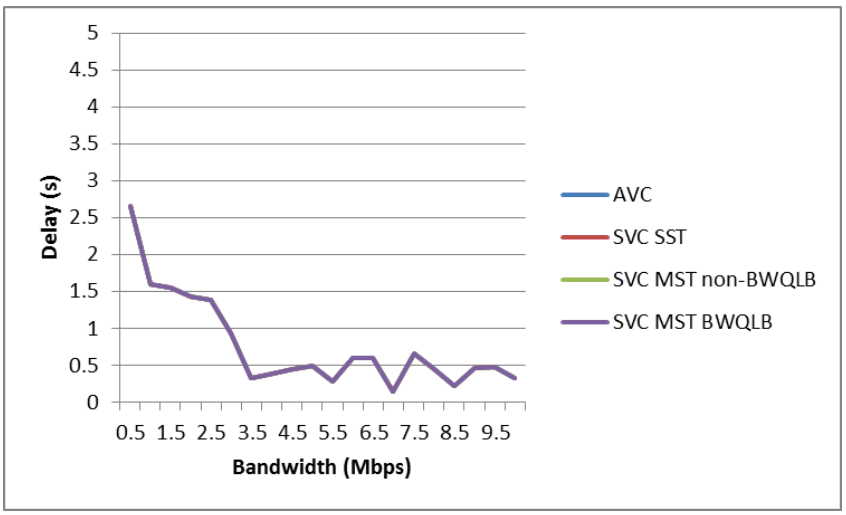
ในหัวข้อนี้ ผู้วิจัยทดลองวัดค่าความล่าช้าของการส่งข้อมูลวีดิทัศน์ AVC, SVC-SST และ SVC-MST จาก h1 ไปยัง h2, h3 และ h4 บนระบบเครือข่าย ดังรูปที่ 4.6 โดยกำหนดให้ h2, h3 และ h4 ร้องขอข้อมูลที่มีขนาดวีดิทัศน์แตกต่างกัน ผู้วิจัยแบ่งการทดลองออกเป็น 2 กรณี ได้แก่ กรณีส่งวีดิทัศน์เพียงหนึ่งขนาด และกรณีส่งวีดิทัศน์หลายขนาด ซึ่งผู้วิจัยได้เปรียบเทียบกับค่าความล่าช้าที่เกิดขึ้นในแต่ละขนาดแบนด์วิดท์ที่แตกต่างกัน



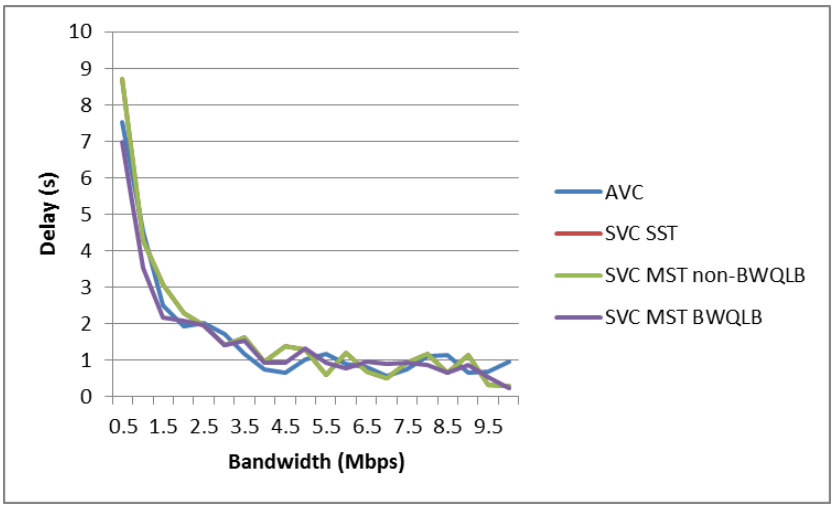
รูปที่ 4.6 แบบจำลองระบบเครื่องข่ายในการทดลอง SVC-MST BWQLB

1) กรณีส่งวีดิทัศน์ในขนาดใดขนาดหนึ่ง

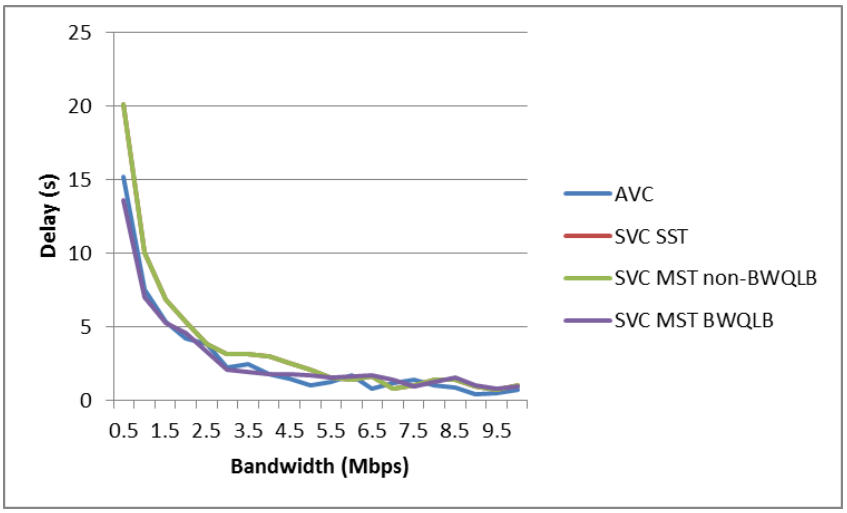
ในกรณีนี้จะเป็นการเปรียบเทียบแต่ละคุณภาพวีดิทัศน์ 640x360, 1280x720 และ 1920x1080 ของการส่งข้อมูล AVC, SVC-SST, SVC-MST แบบไม่พิจารณาเส้นทางก่อนหน้า (SVC-MST non-BWQLB) และ SVC-MST แบบพิจารณาเส้นทางก่อนหน้า (SVC-MST BWQLB) ซึ่งมีผลการวัดค่า ดังนี้



รูปที่ 4.7 กราฟเปรียบเทียบความล่าช้าของวีดิทัศน์ในแต่ละขนาดแบนด์วิดท์ของ 640x360



รูปที่ 4.8 กราฟเปรียบเทียบความล่าช้าของวีดิทัศน์ในแต่ละขนาดแบนด์วิดท์ของ 1280x720

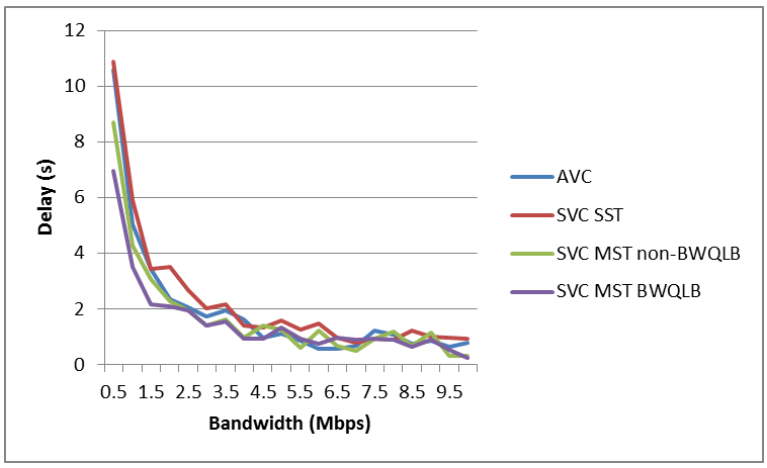


รูปที่ 4.9 กราฟเปรียบเทียบความล่าช้าของวีดิทัศน์ในแต่ละขนาดแบนด์วิดท์ของ 1920x1080

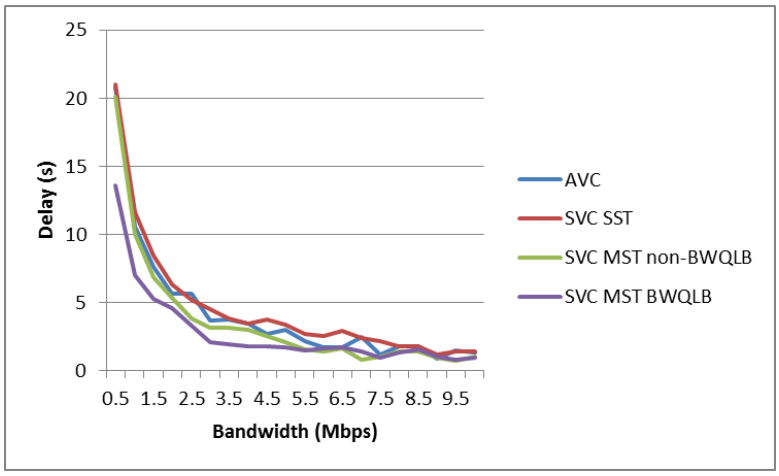
จากผลลัพธ์การทดลอง รูปที่ 4.7 ของขนาด 640x360 จะไม่มีความแตกต่างกันของประเภทวิดีทัศน์และเส้นทางที่ใช้ในการส่งของแต่ละรูปแบบ เนื่องจากการส่งข้อมูลวีดิทัศน์ขนาดต่ำสุดของ SVC เป็นการส่งข้อมูล AVC ที่มีคุณภาพต่ำสุด และเป็นการส่งข้อมูลเพียงเส้นทางเดียวกันทั้งหมด ทำให้รูปแบบทั้งหมดมีรูปแบบเหมือนกับ AVC และแสดงผลเป็นกราฟเส้นเดียว ในส่วนรูปที่ 4.8 ของขนาด 1280x720 และรูปที่ 4.9 ของขนาด 1920x1080 จะเห็นความแตกต่างของการส่งข้อมูลในขนาดวิดีทัศน์ที่ใหญ่ขึ้น เนื่องจากขนาดวีดิทัศน์ 1280x720 และ 1920x1080 ของ AVC และ SVC จะมีขนาดที่แตกต่างกัน ในส่วนเส้นทางการส่งนั้น SVC-SST และ SVC-MST non-BWQLB จะมีการส่งข้อมูลในเส้นทางเดียวกับ AVC แต่มีขนาดวีดิทัศน์ที่ใหญ่กว่า ซึ่งจะแสดงอัตราส่วนนั้นในหัวข้อที่ 4.3.1 ทำให้ SVC-SST และ SVC-MST non-BWQLB มีรูปแบบเดียวกันและมีความล่าช้าสูงกว่า AVC แม้ว่า SVC-MST BWQLB จะเป็นการส่งข้อมูล SVC ที่มีขนาดโดยรวมใหญ่กว่า AVC แต่มีรูปแบบการส่งข้อมูลแบบแยกเส้นทางให้แต่ละชั้นวีดิทัศน์เมื่อเส้นทางในการส่งข้อมูลวีดิทัศน์มีขนาดแบนด์วิดท์ที่ไม่เพียงพอต่อชั้นนั้น ๆ เมื่อเปรียบเทียบเป็นเปอร์เซ็นต์โดยรวมพบว่า SVC-MST BWQLB มีค่าความล่าช้าโดยเฉลี่ยในช่วง 0 ถึง 10 เมกะบิต โดยรวมน้อยกว่า AVC ในขนาด 640x360, 1280x720 และ 1920x1080 เท่ากับ 0%, 5.33% และ 0.136% ตามลำดับ และน้อยกว่า SVC-SST และ SVC-MST non-BWQLB เท่ากับ 0%, 11.62% และ 30.84% ตามลำดับ การกำหนดให้มีการแยกเส้นทางของ SVC-MST BWQLB ในการส่งข้อมูลของขนาด 1280x720 และ 1920x1080 คือ ขนาดแบนด์วิดท์ 1.75 เมกะบิต และ 3 เมกะบิต ตามลำดับ ซึ่งเป็นขนาดแบนด์วิดท์ที่เพียงพอต่อการแสดงผลที่ 24 เฟรมต่อวินาที ทำให้ SVC-MST BWQLB มีการกระจายเส้นทางในการส่งข้อมูลของ 1280x720 และ 1920x1080 ช่วงแบนด์วิดท์ต่ำกว่า 1.75 เมกะบิต และ 3 เมกะบิต ตามลำดับ เพื่อแสดงประสิทธิภาพของ SVC-MST BWQLB โดยเปรียบเทียบ SVC-MST BWQLB กับรูปแบบการส่งข้อมูล AVC, SVC-SST และ SVC-MST non-BWQLB ในขนาด 1280x720 ช่วงแบนด์วิดท์ 0 ถึง 1.75 เมกะบิต และขนาด 1920x1080 ช่วงแบนด์วิดท์ 0 ถึง 3 เมกะบิต SVC-MST BWQLB มีความล่าช้าน้อยกว่า AVC เท่ากับ 11.67% และ 7.03% , น้อยกว่า SVC-SST และ SVC-MST non-BWQLB เท่ากับ 24.36% และ 37.53% ตามลำดับ

2) กรณีส่งวีดิทัศน์หลายขนาด

ในกรณีนี้จะเป็นการเปรียบเทียบกลุ่มคุณภาพวีดิทัศน์ ได้แก่ (640x360 และ 1280x720) และ (640x360, 1280x720 และ 1920x1080) และยังคงเป็นการเปรียบเทียบรูปแบบการส่งข้อมูลของ SVC-MST BWQLB กับ AVC, SVC-SST และ SVC-MST non-BWQLB ซึ่งมีผลการวัดค่า ดังนี้



รูปที่ 4.10 กราฟเปรียบเทียบความล่าช้าของกลุ่มวีดิทัศน์ในแต่ละขนาดแบนด์วิดท์ของ (640x360 และ 1280x720)



รูปที่ 4.11 กราฟเปรียบเทียบความล่าช้าของกลุ่มวีดิทัศน์ในแต่ละขนาดแบนด์วิดท์ของ (640x360, 1280x720 และ 1920x1080)

จากผลลัพธ์การทดลองรูปแบบการส่งข้อมูลใน รูปที่ 4.10 กลุ่มขนาด (640x360 และ 1280x720) และรูปที่ 4.11 กลุ่มขนาด (640x360, 1280x720 และ 1920x1080) จะแสดงให้เห็นถึงประสิทธิภาพของคุณสมบัติการแยกเส้นทางของชั้นคุณภาพวีดิทัศน์ที่แตกต่างกันของ SVC-MST BWQLB กับรูปแบบอื่น เมื่อเปรียบเทียบเป็นเปอร์เซ็นต์โดยรวมพบว่า SVC-MST BWQLB มีค่าความล่าช้าโดยเฉลี่ยในช่วง 0 ถึง 10 เมกะบิต โดยรวมน้อยกว่า AVC ในกลุ่มขนาด (640x360 และ 1280x720) และ (640x360, 1280x720 และ 1920x1080) เท่ากับ 25.84% และ 50.73% ตามลำดับ น้อยกว่า SVC-SST เท่ากับ 46.77% และ 66.64% ตามลำดับ และน้อยกว่า SVC-MST non-BWQLB เท่ากับ 11.62% และ 30.84% ตามลำดับ เนื่องจากคุณสมบัติของ SVC-MST จะส่งข้อมูลในรูปแบบชั้นคุณภาพวีดิทัศน์จึงทำให้การส่งข้อมูลทั้ง SVC-MST BWQLB และ SVC-MST non-BWQLB แสดงประสิทธิภาพได้ดีกว่ารูปแบบการส่งข้อมูลของ AVC และ SVC-SST ที่ส่งข้อมูล

เต็มของขนาดวีดิทัศน์นั้นจึงทำให้มีขนาดวีดิทัศน์ที่ทับซ้อนกัน นอกจากนี้ SVC-MST BWQLB ยังสามารถแสดงประสิทธิภาพในการลดความล่าช้าได้มากกว่า SVC-MST non-BWQLB เนื่องจากรูปแบบการส่งข้อมูลที่จะแยกเส้นทางในการส่งชั้น 1280x720 ที่แบนด์วิดท์ต่ำกว่า 1.75 เมกะบิต และ 1920x1080 ที่แบนด์วิดท์ต่ำกว่า 3 เมกะบิต ดังนั้นผู้วิจัยจึงเปรียบเทียบความล่าช้าของการส่งข้อมูลของ 1280x720 และ 1920x1080 ช่วงแบนด์วิดท์ต่ำกว่า 1.75 เมกะบิต และ 3 เมกะบิต ตามลำดับ เพื่อแสดงให้เห็นถึงประสิทธิมากขึ้น SVC-MST BWQLB ซึ่ง SVC-MST BWQLB มีความล่าช้าน้อยกว่า AVC เท่ากับ 45.22% และ 50.20%, น้อยกว่า SVC-SST เท่ากับ 60.74% และ 59.22% และน้อยกว่า SVC-MST non-BWQLB เท่ากับ 24.36% และ 37.53% ตามลำดับ

4.4 วิเคราะห์และสรุปผล

รูปแบบการเข้ารหัสวีดิทัศน์ H.254 ในส่วนการเก็บข้อมูล AVC มีความสามารถในการเก็บข้อมูลได้ดีกว่า SVC ในกรณีการเก็บวีดิทัศน์ในขนาดใดขนาดหนึ่งที่ขนาดวีดิทัศน์ 640x360, 1280x720 และ 1920x1080 เท่ากับ 0%, 8.93% และ 28.72% ตามลำดับ แต่ในกรณีการเก็บวีดิทัศน์หลายขนาดที่ขนาดวีดิทัศน์ (640x360 และ 1280x720) และ (640x360, 1280x720 และ 1920x1080) SVC กลับมีการใช้พื้นที่น้อยกว่า AVC เท่ากับ 22.14% และ 42.22% ตามลำดับ เนื่องจากความสามารถในการเก็บข้อมูลของ SVC สามารถแยกส่วนคุณภาพได้ต่างจาก AVC ดังนั้น SVC จึงเหมาะสมกับการเก็บข้อมูลของผู้ให้บริการที่ต้องการส่งข้อมูลวีดิทัศน์เดียวกันแต่มีคุณภาพที่แตกต่างกัน

ในส่วนการทดลองการใช้แบนด์วิดท์ของ AVC, SVC-SST, SVC-MST non-BWQLB และ SVC-MST BWQLB โดยการวัดค่าความล่าช้า ซึ่งผลลัพธ์ของประสิทธิภาพจากค่าความล่าช้าจากน้อยไปมากในกรณีส่งวีดิทัศน์ในขนาดใดขนาดหนึ่ง ได้ดังนี้ SVC-MST BWQLB, AVC , SVC-MST non-BWQLB และ SVC-SST ตามลำดับ เมื่อเปรียบเทียบเป็นเปอร์เซ็นต์โดยรวมพบว่า SVC-MST BWQLB มีค่าความล่าช้าโดยเฉลี่ยในช่วง 0 ถึง 10 เมกะบิต โดยรวมน้อยกว่า AVC ในขนาด 640x360, 1280x720 และ 1920x1080 เท่ากับ 0%, 5.33% และ 0.136% ตามลำดับ น้อยกว่า SVC-SST และ SVC-MST non-BWQLB เท่ากับ 0%, 11.62% และ 30.84% ตามลำดับ จะเห็นได้ว่าในกรณีนี้ การส่งข้อมูลของ SVC-SST และ SVC-MST non-BWQLB จะมีค่าเท่ากันเนื่องจากมีรูปแบบการส่งเส้นทางเหมือนกัน

ในส่วนกรณีส่งวีดิทัศน์หลายขนาด เมื่อจัดเรียงประสิทธิภาพจากค่าความล่าช้าจากน้อยไปมาก ได้ดังนี้ SVC-MST BWQLB, SVC-MST non-BWQLB, AVC และ SVC-SST ตามลำดับ เมื่อเปรียบเทียบเป็นเปอร์เซ็นต์โดยรวมพบว่า SVC-MST BWQLB ยังคงมีค่าความล่าช้าโดยเฉลี่ยในช่วง 0 ถึง 10 เมกะบิต น้อยกว่า AVC ในกลุ่มขนาด (640x360 และ 1280x720) และ (640x360,

1280x720 และ 1920x1080) เท่ากับ 25.84% และ 50.73% ตามลำดับ, น้อยกว่า SVC-SST เท่ากับ 46.77% และ 66.64% ตามลำดับ และน้อยกว่า SVC-MST non-BWQLB เท่ากับ 11.62% และ 30.84% ตามลำดับ

จากผลลัพธ์ข้างต้น จะเห็นได้ว่า SVC-MST BWQLB มีประสิทธิภาพดีที่สุดทั้งในด้านการเก็บข้อมูลแบบหลายขนาดวีดิทัศน์และการส่งข้อมูลทั้งรูปแบบการส่งข้อมูลแบบเพียงหนึ่งขนาดวีดิทัศน์และรูปแบบการส่งข้อมูลแบบหลายขนาดวีดิทัศน์ เนื่องจากคุณสมบัติของ SVC-MST BWQLB สามารถแยกการส่งข้อมูลไปหลายเส้นทางได้โดยขึ้นอยู่กับจำนวนระดับคุณภาพที่ต้องการของผู้รับต่างกับรูปแบบการส่งข้อมูลของ AVC และ SVC-SST ที่มีรูปแบบการส่งข้อมูลเพียงเส้นทางเดียว เมื่อเส้นทางการส่งข้อมูลนั้นไม่เพียงพอต่อความต้องการของระดับขนาดของวีดิทัศน์นั้น ๆ ก็ทำให้เกิดความคับคั่งสูง อย่างไรก็ตาม SVC-MST ทั้งแบบ non-BWQLB และ BWQLB มีปัญหาการรับข้อมูลที่อาจจะไม่ตรงตามลำดับข้อมูลอย่างถูกต้อง เนื่องจากส่งหลายกลุ่มและหลายเส้นทาง ซึ่งผู้วิจัยได้ออกแบบบัฟเฟอร์เพื่อแก้ปัญหาลำดับข้อมูลไว้ในบทที่ 5 ต่อไป

บทที่ 5
การทดลองเพื่อแก้ไขปัญหา SVC-MST Sequence

ในบทนี้เป็นการแสดงการแก้ปัญหาลำดับวีดิทัศน์ที่ไม่ตรงกันของ SVC-MST โดยการออกแบบตัวแปรใหม่ 2 ตัวแปรให้กับ RTP Header และสร้าง Sorted buffer โดยวัดขนาดที่เหมาะสมให้กับแต่ละขนาดวีดิทัศน์ โดยจะทำการทดลองใช้กับรูปแบบการหาเส้นทาง แบบพิจารณาเส้นทางก่อนหน้าและแบบไม่พิจารณาเส้นทางก่อนหน้า ดังนั้นผู้วิจัยจะแบ่งเนื้อหาในบทนี้เป็นความเป็นมา รายละเอียดของการสร้าง RTP Header และสร้าง Sorted buffer ขนาดความล่าช้าที่เกิดขึ้นกับขนาดของ Sorted buffer จากนั้นจะวิเคราะห์ผลลัพธ์และสรุปผล

5.1 ความเป็นมา

จากการเปรียบเทียบการทำงานของการส่งข้อมูลของ AVC, SVC-SST และ SVC-MST จะเห็นได้ว่า SVC-MST จะมีการทำงานที่ดีกว่าในข้อจำกัดของแบนด์วิดท์ที่น้อยกว่า จากการแยกเส้นทางในการส่งข้อมูล แต่ผลลัพธ์ของวีดิทัศน์ที่เกิดขึ้นคือลำดับวีดิทัศน์เกิดการผิดพลาดขึ้น ผู้วิจัยจึงได้การออกแบบบัฟเฟอร์และตัวแปรใหม่ให้กับ RTP header ที่ใช้ในการจัดเรียงลำดับวีดิทัศน์อย่างถูกต้องแสดงผลวีดิทัศน์ นอกจากนี้รูปแบบการส่งข้อมูล SVC-MST โดยทั่วไปไม่มีการคำนวณเส้นทางโดยคำนึงถึงการใช้งานแบนด์วิดท์ของชั้นวีดิทัศน์ก่อนหน้าของการส่งหลายขนาดวีดิทัศน์ในเวลาเดียวกัน ผู้วิจัยจึงทำการเปรียบเทียบการคำนวณเส้นทางการหาเส้นทางทั้งแบบพิจารณาเส้นทางก่อนหน้าและแบบไม่พิจารณาเส้นทางก่อนหน้า

5.2 วิธีการทดลอง

ในการทดลองนี้ เป็นการทดลองสร้างบัฟเฟอร์สำหรับรองรับการทำงานของ SVC-MST ที่แยกการส่งข้อมูลหลายเส้นทางทำให้เกิดปัญหาข้อมูลไม่จัดเรียงกัน นอกจากนี้ ผู้วิจัยได้เปรียบเทียบรูปแบบการส่งข้อมูลแบบทั่วไปที่ไม่คำนึงถึงการใช้แบนด์วิดท์ของวีดิทัศน์ที่คำนวณก่อนหน้าและพิจารณาเส้นทางของชั้นข้อมูลก่อนหน้า ผู้วิจัยได้ใช้ขนาดวีดิทัศน์ SVC ทั้ง 3 ขนาด ได้แก่ 640x360, 1280x720 และ 1920x1080

5.3 การออกแบบการจัดเรียงข้อมูลในฝั่งผู้รับ

ในส่วนนี้จะแสดงการสร้าง RTP header แบบเฉพาะที่รองรับ SVC-MST และ Sorted buffer ในการจัดเรียงข้อมูล ซึ่งผู้วิจัยได้สร้างการจัดเรียงข้อมูลออกเป็น 3 ส่วน ได้แก่ จัดเรียงภายในเฟรมย่อย จัดเรียงภายในชั้น และจัดเรียงระหว่างชั้น

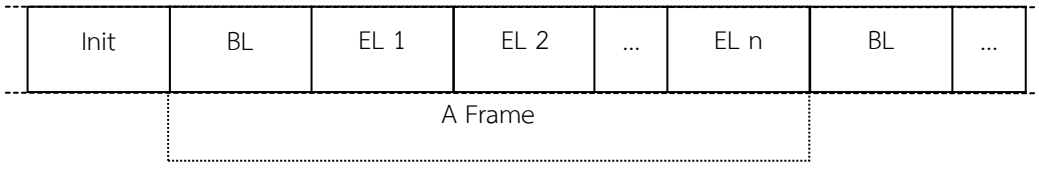
5.3.1 Modified RTP header

ในส่วนนี้ ผู้วิจัยจะแสดงการแยกชั้นวีดิทัศน์ปรับขนาดได้ ดังรูปที่ 5.1 และแสดงการแยกเส้นทางการส่งข้อมูลในแต่ละเส้นทางของชั้นวีดิทัศน์นั้น ๆ ดังรูปที่ 5.2 ซึ่งจะมีประเภทของ NALU ดังนี้

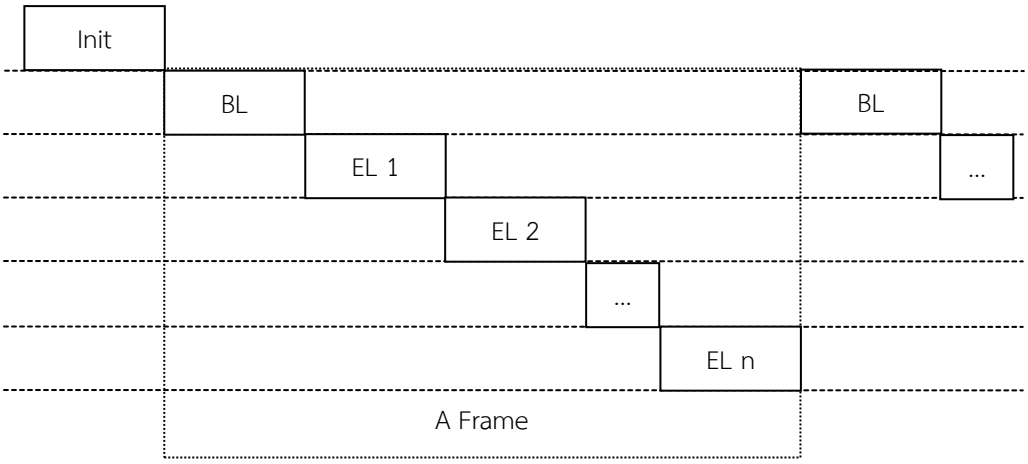
1) ข้อมูลเริ่มต้น Initial data (Init) เป็นส่วนเก็บข้อมูลองค์ประกอบต่าง ๆ ของวีดิทัศน์ Network Abstraction Layer Units (NALU) ประเภทไม่ใช่ชั้นวีดิทัศน์เข้ารหัส Non-Video Coding Layer (non-VCL)

2) ชั้นพื้นฐาน Base layer (BL) เป็นส่วนฐานของวีดิทัศน์ที่สามารถแสดงผลวีดิทัศน์ด้วยตัวมันเองแต่เป็นวีดิทัศน์ที่มีคุณภาพต่ำสุด และมีเพียงไฟล์เดียวเท่านั้น NALU ประเภทชั้นวีดิทัศน์เข้ารหัส Video Coding Layer (VCL) ที่เท่ากับ 1 หรือ 14

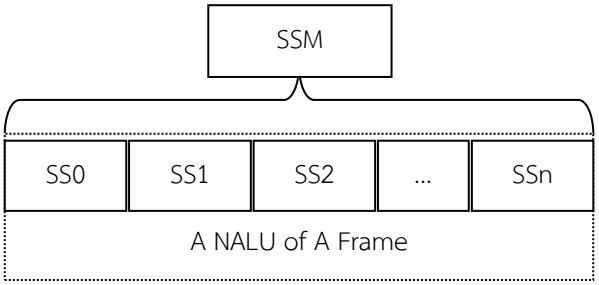
3) ชั้นขยาย Enhancement layer (EL) เป็นส่วนขยายคุณภาพวีดิทัศน์เป็นหลายระดับตามความต้องการของผู้ใช้ ทำให้ไฟล์คุณภาพถูกแบ่งออกเป็นหลายไฟล์เป็นลำดับ NALU ประเภทชั้นวีดิทัศน์เข้ารหัส VCL ที่เท่ากับ 20



รูปที่ 5.1 ข้อมูลวีดิทัศน์ปรับขนาดได้แบบทั่วไป



รูปที่ 5.2 ข้อมูลวีดิทัศน์ปรับขนาดได้แบบแยกส่วนคุณภาพ



รูปที่ 5.3 ข้อมูล NALU ของวีดิทัศน์ปรับขนาดได้แบบแยกส่วน

เนื่องจากการส่งข้อมูลวีดิทัศน์จะมีการจำกัดขนาดแพ็กเก็ต ผู้วิจัยจึงต้องแบ่งย่อย NALU เป็นหน่วยย่อย ดังรูปที่ 5.3 เพื่อให้สามารถสามารถส่งข้อมูลได้ และเพิ่มตัวแปรใน RTP header เพื่อกำหนดลำดับภายในให้กับแต่ละเฟรมย่อยที่ถูกแบ่งออกมาโดยเพิ่มตัวแปร ได้แก่ Sub Sequence Number (SSn) ขนาด 16 บิต และ Sub Sequence Max number (SSM) 16 บิต ตั้งแต่บิตที่ 33 ถึง 48 และ 49 ถึง 64 ตามลำดับ ดังรูปที่ 5.4

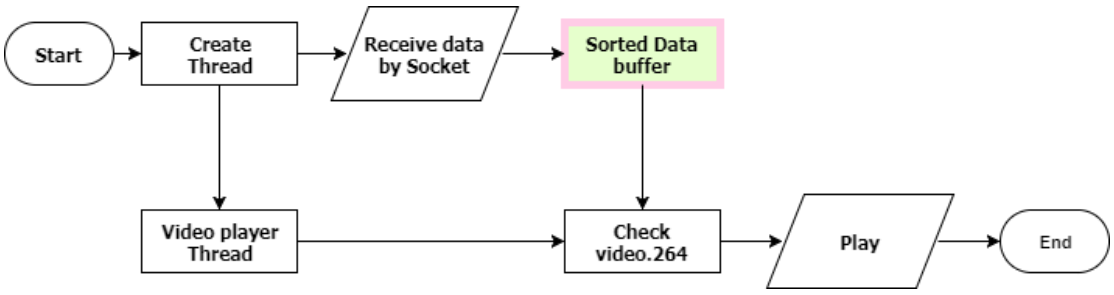
01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
V		P	X	CC				M	Packet Type							Sequence Number															
Sub Sequence Number																**Sub Sequence Max Number**															
Time stamp																															
Synchronization source (SSRC) identifier																															
contributing source (CSRC) identifiers ...																															

รูปที่ 5.4 Modified RTP header

จากรูปที่ 5.4 ผู้วิจัยใช้ modified RTP header ที่เพิ่มตัวแปรในการจัดเรียงข้อมูล โดยมีการใช้ตัวแปร ดังนี้ Packet Type ถูกใช้ในการแบ่งประเภทของชั้นวีดิทัศน์ Sequence Number ถูกใช้ในการบอกลำดับเฟรม Sub Sequence Number ถูกใช้ในการบอกลำดับของเฟรมที่ถูกแบ่งย่อยออก และ Sub Sequence Max Number ถูกใช้ในการบอกจำนวนเฟรมที่แบ่งออกมา

5.3.2 การสร้าง Sorted buffer

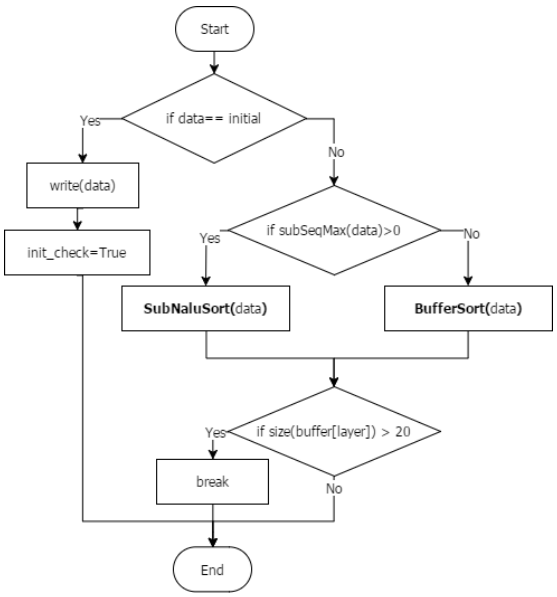
ในส่วนนี้ ผู้วิจัยแสดงการสร้าง Sorted buffer ที่ใช้แก้ปัญหาการไม่เรียงกันของลำดับข้อมูลในการส่งข้อมูลของ SVC-MST ที่ฝั่งผู้รับได้รับข้อมูลมีการจัดเรียงข้อมูลที่แตกต่างกัน โดยรูปแบบการทำงานของฝั่งผู้รับแสดง ดังรูปที่ 5.5



รูปที่ 5.5 ภาพรวมการทำงานของฝั่งผู้รับ

Sorted Data buffer มีการแบ่งฟังก์ชันได้ ดังนี้

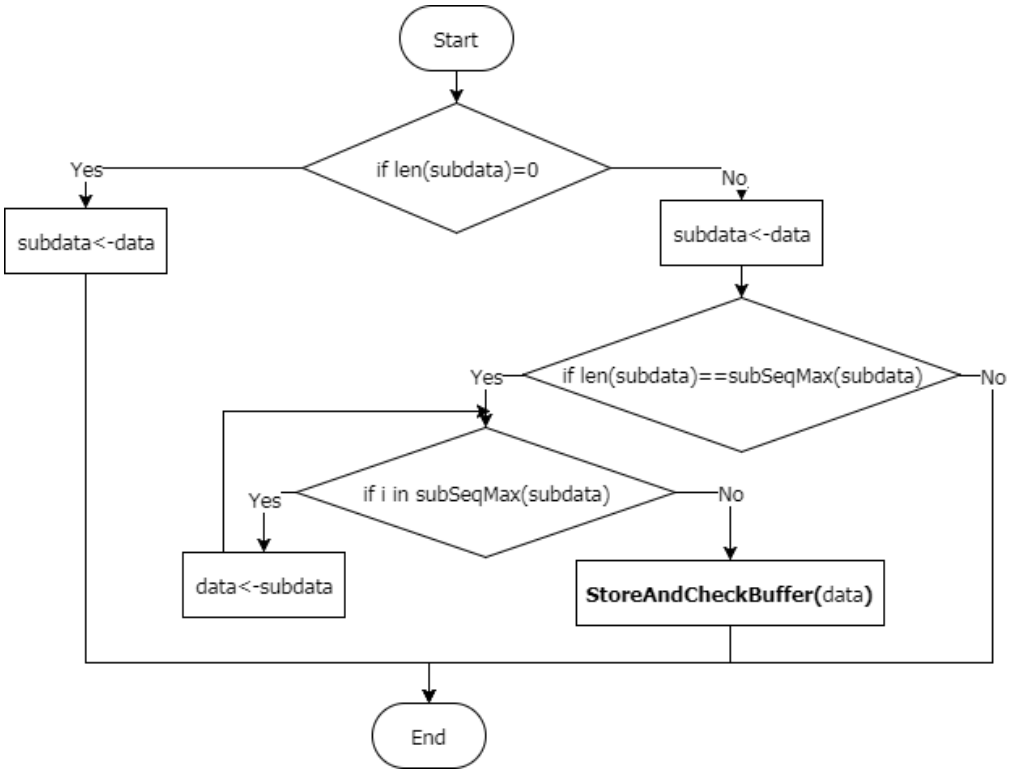
1) InitialChecker (data) function



รูปที่ 5.6 InitialChecker function

ฟังก์ชันนี้ คือ ส่วนในการตรวจสอบการมาถึงของ Initial packet ที่เก็บข้อมูล NALU ประเภท non-VCL ซึ่งเป็นส่วนในการตั้งค่าการแสดงผลวีดิทัศน์ โดยการตรวจสอบ Packet Type ซึ่งผู้วิจัยกำหนดให้เป็น 101 แสดงการทำงาน ดังรูปที่ 5.6

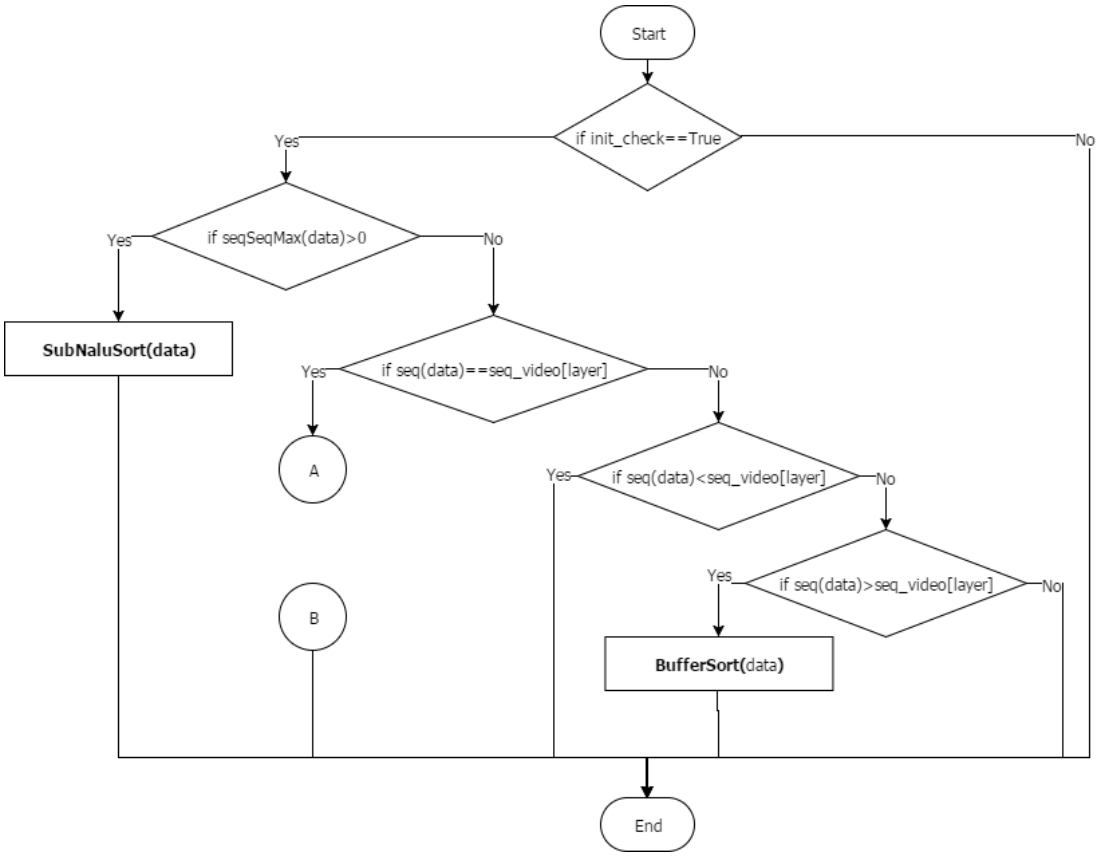
2) SubAccessUnitSort (data) function



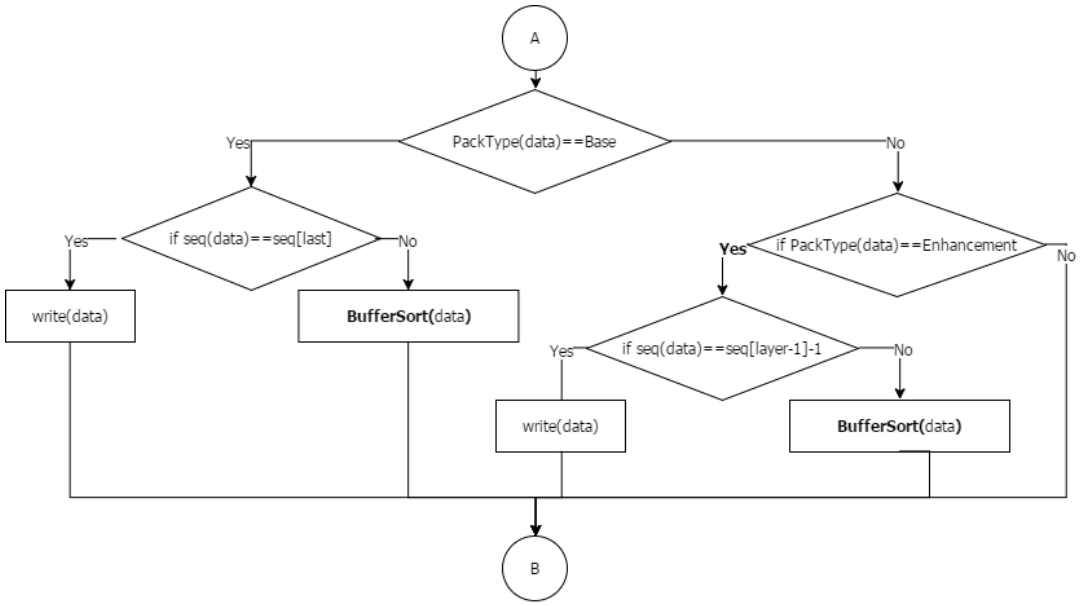
รูปที่ 5.7 SubAccessUnitSort function

ฟังก์ชันนี้ คือ ส่วนในการจัดเก็บและจัดเรียงข้อมูลที่ถูกแบ่งย่อยเฟรมก่อนนำไปจัดเรียงที่ AccessUnitSort (data) function โดยตรวจสอบการใช้ฟังก์ชันจากตัวแปร Sub Sequence Max Number ที่สร้างขึ้นใน RTP header หากมีค่ามากกว่า 0 ก็หมายความว่า ข้อมูลที่ถูกส่งเข้ามาถูกแบ่งย่อย โดยฟังก์ชันนี้จะรวมข้อมูลที่มี Packet Type และ Sequence Number เดียวกัน และจัดเรียงข้อมูลด้วย Sub Sequence Number การตรวจสอบจำนวนเฟรมที่สมบูรณ์โดยเปรียบเทียบกับ Sub Sequence Max Number ว่าเท่ากันแล้วหรือยัง แสดงการทำงาน ดังรูปที่ 5.7

3) AccessUnitSort (data) function



(ก)

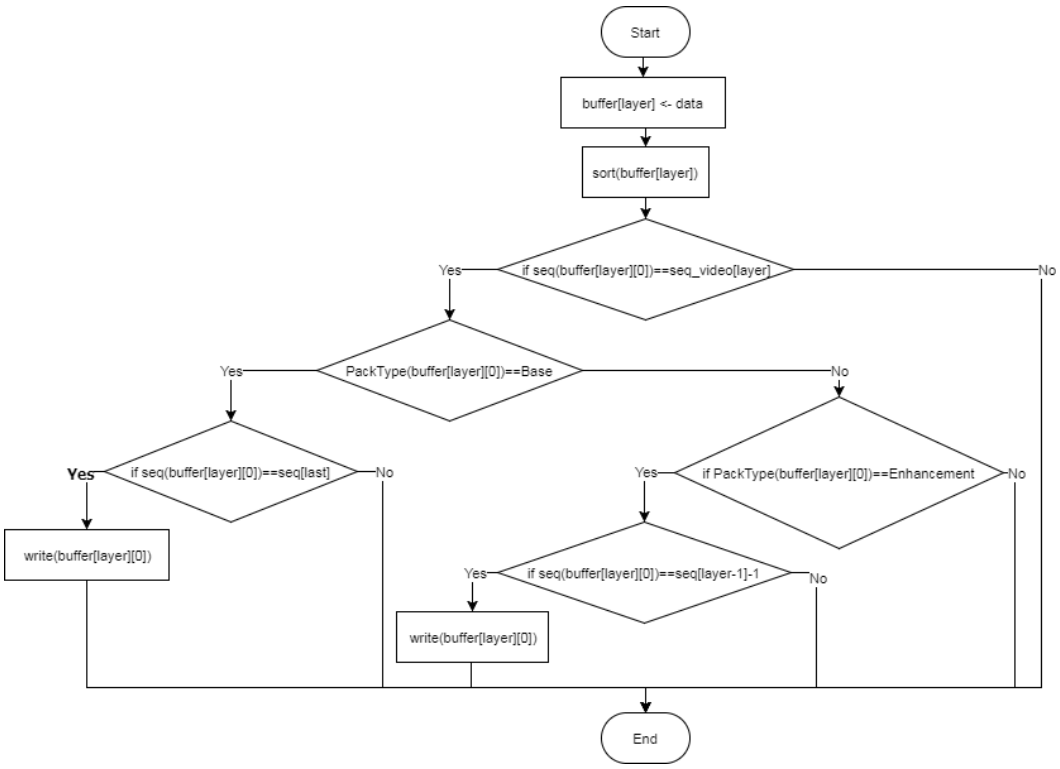


(ข)

รูปที่ 5.8 AccessUnitSort function (ก) ตรวจสอบลำดับภายในชั้น (ข) ตรวจสอบลำดับระหว่างชั้น

ฟังก์ชันนี้ คือ ส่วนหลักในการจัดเรียงข้อมูลโดยตรวจสอบลำดับข้อมูลที่เข้ามาด้วย Sequence Number การตรวจสอบลำดับในฟังก์ชันนี้สามารถแบ่งออกเป็น 2 ส่วน ได้แก่ ตรวจสอบลำดับภายในชั้น ดังรูปที่ 5.8 (ก) และตรวจสอบลำดับระหว่างชั้น ดังรูปที่ 5.8 (ข) ในการตรวจสอบลำดับภายในชั้นมีการพิจารณา ดังนี้ หากลำดับข้อมูลที่เข้ามาตรงกับลำดับวีดิทัศน์ที่ต้องการ จะได้รับการพิจารณาในการตรวจสอบลำดับระหว่างชั้นต่อไป, หากลำดับข้อมูลที่เข้ามาช้ากว่าลำดับวีดิทัศน์ที่บันทึกลงบัฟเฟอร์ก่อนแสดงผลแล้ว จะถูกตัดข้อมูลนั้นออกไป และหากข้อมูลที่เข้ามาเร็วกว่าลำดับที่ถูกต้อง ก็จะถูกส่งไปเก็บในบัฟเฟอร์ และในการจัดเรียงลำดับระหว่างชั้นข้อมูล สามารถแบ่งออกเป็น 2 กลุ่มข้อมูล ได้แก่ ข้อมูลประเภทพื้นฐาน (Base) จะตรวจสอบชั้นวีดิทัศน์ในชั้นสุดท้ายว่าได้รับข้อมูลแล้วหรือไม่ และข้อมูลประเภทขยาย (Enhancement) จะตรวจสอบการรับข้อมูลของวีดิทัศน์ในชั้นก่อนหน้า หากข้อมูลที่เข้ามาไม่ตรงกับลำดับของข้อมูลที่ถูกต้องก็จะถูกส่งไปจัดเก็บและจัดเรียงภายในบัฟเฟอร์ด้วย BufferSort (data) function

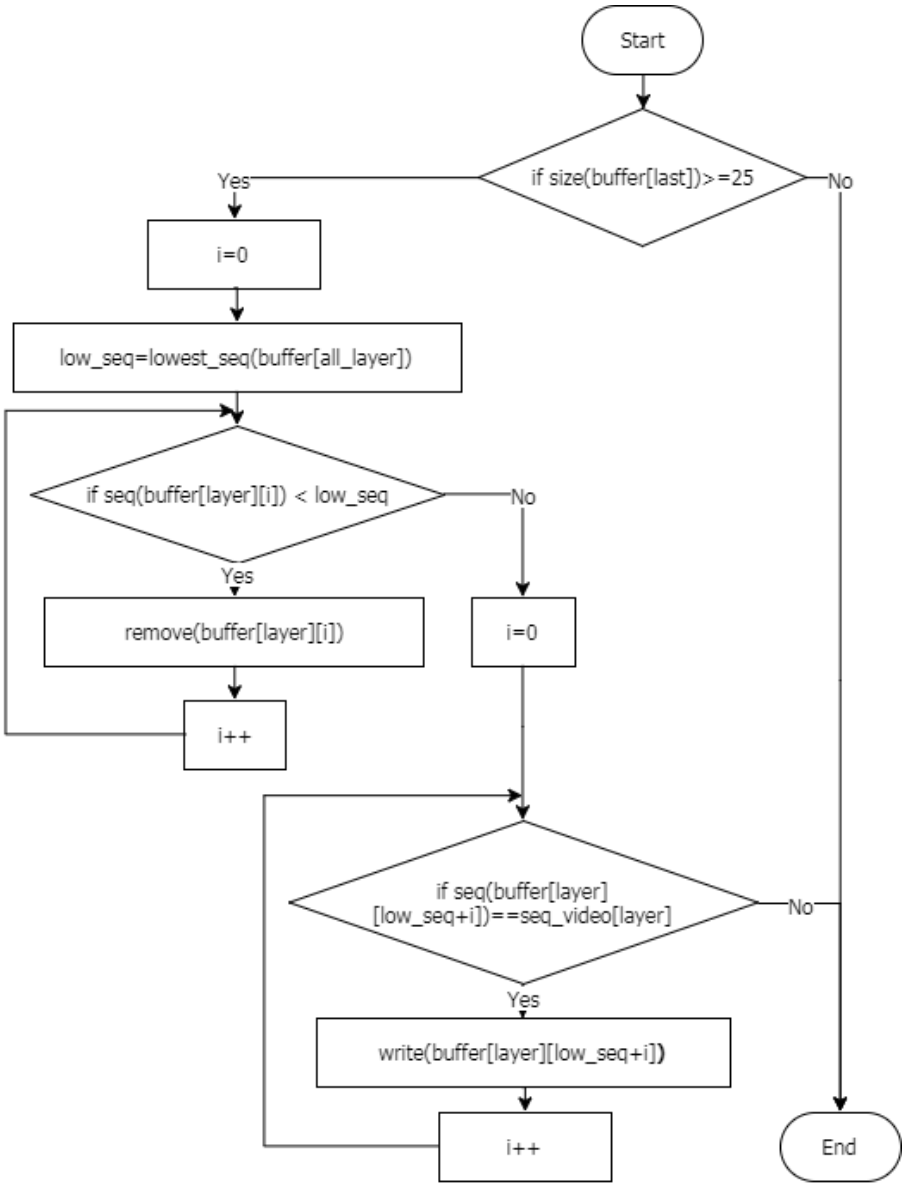
4) BufferSort (data) function



รูปที่ 5.9 BufferSort function

ฟังก์ชันนี้ คือ ส่วนในจัดเก็บข้อมูลที่เข้ามาแต่ไม่ตรงกับลำดับวีดิทัศน์ที่ถูกต้อง ตำแหน่งในการจัดเก็บจะถูกจัดเรียงก่อนถูกเก็บข้อมูล เมื่อจัดเก็บแล้วจะตรวจสอบลำดับแรกสุดของชั้นวีดิทัศน์ในบัฟเฟอร์ว่าตรงตามความต้องการวีดิทัศน์หรือไม่ แสดงการทำงาน ดังรูปที่ 5.9

5) BufferFullChecker(data) function



รูปที่ 5.10 BufferFullChecker function

ฟังก์ชันนี้ คือ ส่วนที่ถูกใช้เมื่อบัฟเฟอร์เต็ม แต่หากลำดับของวีดิทัศน์ที่ต้องการยังไม่มาถึงก็จะถูกตัดการทำงานออกและเลื่อนลำดับวีดิทัศน์ที่ต้องการเป็นลำดับต่ำสุดที่มีในทุกชั้นวีดิทัศน์ เพื่อบันทึกในบัฟเฟอร์ที่ใช้แสดงผล จากนั้นจึงนำข้อมูลที่มีในบัฟเฟอร์เขียนลงในวีดิทัศน์ที่พร้อมแสดงผลตามลำดับที่ถูกต้อง หากมีลำดับที่ไม่ตรงกับที่ต้องการก็จะหยุดทำงาน แสดงการทำงาน ดังรูปที่ 5.10

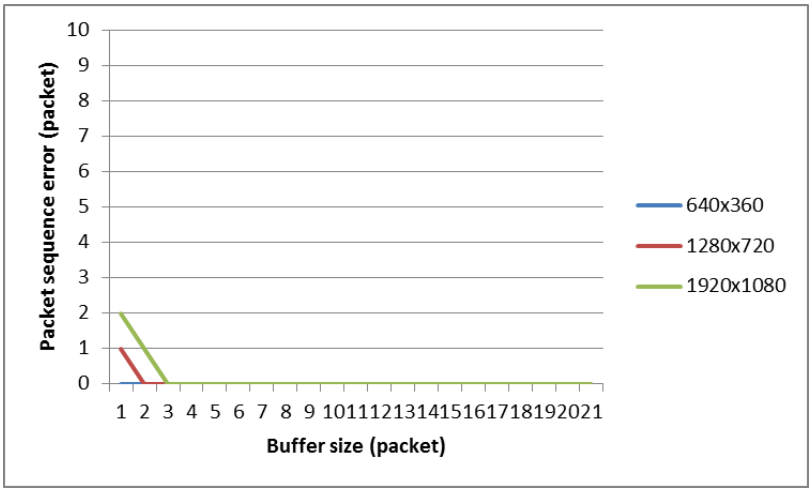
5.3.3 ทดลองหาขนาด Sorted buffer ที่เหมาะสมและเวลาสูงสุดที่เกิดขึ้นในบัฟเฟอร์

ในหัวข้อนี้ ผู้วิจัยทดสอบการส่งข้อมูลโดยทดลองกับการส่งข้อมูลบนระบบเครือข่ายที่มีเส้นทางไปยังปลายทาง 3 เส้นทาง ที่มีขนาดแบนด์วิดท์สูงสุดที่ 0.6 เมกะบิต สำหรับการส่งข้อมูล 640x360, 2.2 เมกะบิต สำหรับการส่งข้อมูล 1280x720 และ 4.8 เมกะบิต สำหรับการส่งข้อมูล 1920x1080 และเพื่อให้เห็นผลแตกต่างจากผลลัพธ์ของการส่งข้อมูลในแบบพิจารณาเส้นทางก่อนหน้าจึงปรับขนาดแบนด์วิดท์ สูงสุดที่ 0.6 เมกะบิต สำหรับการส่งข้อมูล 640x360, 1.75 เมกะบิต สำหรับการส่งข้อมูล 1280x720 และ 3 เมกะบิต สำหรับการส่งข้อมูล 1920x1080 ค่าแบนด์วิดท์ที่ได้เกิดจากการคำนวณ ดังสมการที่ (10)

$$Bandwidth = \frac{Videosize * FPS}{GOPFrame} \quad (10)$$

1 SVC-MST แบบไม่พิจารณาเส้นทางก่อนหน้า (SVC-MST non-BWQLB)

ในส่วนนี้ เป็นรูปแบบที่ถูกใช้โดยทั่วไปในหลายงานวิจัย ซึ่งข้อมูลที่ถูกส่งจะถูกรวมชั้นวีดิทัศน์เข้าด้วยกัน จากการส่งข้อมูลแสดงขนาดของ Sorted buffer ที่เหมาะสมกับแต่ละขนาดวีดิทัศน์ ได้แก่ 640x360, 1280x720 และ 1920x1080 ซึ่งมีผลลัพธ์ ดังรูปที่ 5.11



รูปที่ 5.11 จำนวนข้อผิดพลาดในแต่ละขนาดบัฟเฟอร์แบบไม่พิจารณาเส้นทางก่อนหน้า

จากรูปที่ 5.11 แสดงขนาดบัฟเฟอร์ที่เหมาะสมที่ไม่ทำให้วีดิทัศน์มีปัญหาในการแสดงผลของวีดิทัศน์ในขนาด 640x360, 1280x720 และ 1920x1080 คือ 0, 1 และ 2 ตามลำดับ เมื่อคำนวณเวลาที่เสียกับขนาดของบัฟเฟอร์ที่เหมาะสมต่อ 1 หน่วย โดยการส่งข้อมูลต่ำสุดที่ทำให้วีดิทัศน์สามารถแสดงผลโดยที่ไม่ทำให้เกิดปัญหาการชะงักโดยอ้างอิงค่าแบนด์วิดท์โดยคำนวณจากสมการที่

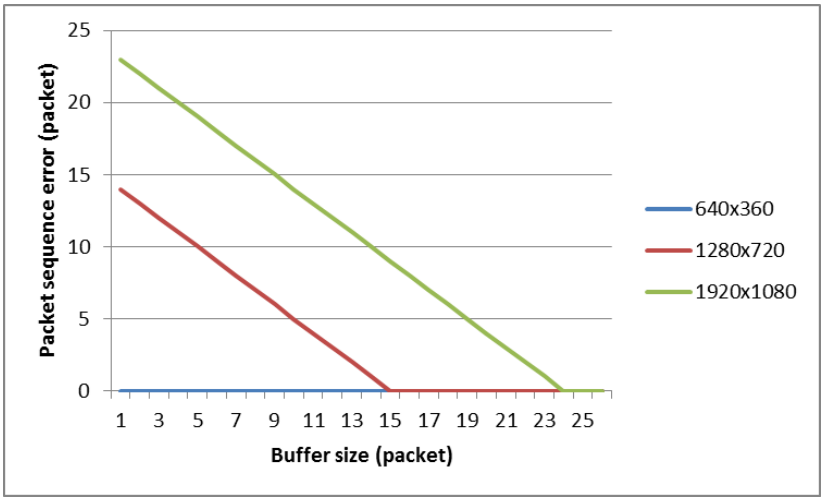
(10) ซึ่งได้ค่าแบนด์วิดท์ของแต่ละขนาดวีดิทัศน์ 640x360, 1280x720 และ 1920x1080 เท่ากับ 0.6 เมกะบิต, 2.2 เมกะบิต และ 4.8 ซึ่งมีการคำนวณเวลาต่อหนึ่งแพ็กเก็ต ได้ดังสมการที่ (11)

$$Buffer\ delay = \frac{Packet\ size}{Bandwidth} \quad (11)$$

ซึ่งผลลัพธ์ของการคำนวณทำให้ทราบว่า เวลาสูงสุดที่ไม่ทำให้การแสดงผลวีดิทัศน์เกิดปัญหาการชะงักของแต่ละเฟรมแพ็กเก็ต มีเวลาประมาณ 0.041 วินาที สำหรับการแสดงวีดิทัศน์ 24 เฟรมต่อวินาที ดังนั้นจะต้องกำหนดขนาดเวลาที่เกิดขึ้นกับของการแสดงผลของวีดิทัศน์ในขนาด 640x360, 1280x720 และ 1920x1080 เท่ากับ 0 วินาที, 0.041 วินาที และ 0.082 วินาที ตามลำดับ อย่างไรก็ตาม เวลาในบัฟเฟอร์ที่เกิดขึ้นนั้นเป็นเพียงค่าเวลาสูงสุดที่เกิดขึ้นกับการแสดงผลของวีดิทัศน์ที่ปกติ หากวีดิทัศน์นั้น ๆ ได้รับการส่งข้อมูลในแบนด์วิดท์ที่สูงขึ้นสามารถลดเวลาที่เกิดขึ้นกับบัฟเฟอร์ได้

2 SVC-MST แบบพิจารณาเส้นทางก่อนหน้า (SVC-MST BWQLB)

ในส่วนนี้ เป็นรูปแบบที่ผู้วิจัยสังเกตเห็นจากการส่งข้อมูลในหลายงานวิจัย ที่สามารถเห็นผลกระทบของการคำนวณแบบไม่พิจารณาเส้นทางก่อนหน้าในการส่งข้อมูล เมื่อขนาด แบนด์วิดท์ถึงขีดจำกัดที่สามารถส่งข้อมูลรวมกันในเส้นทางใดเส้นทางหนึ่ง ทำให้แก้ปัญหาการชะงักโดยการแยกเส้นทางไปยังเส้นทางอื่นทำให้มีการกระจายการส่งข้อมูล แต่ผลลัพธ์จากการส่งข้อมูลโดยแยกเส้นทางก็จะเกิดปัญหาลำดับของข้อมูลที่มาถึงไม่ตรงตามการแสดงผล ในส่วนนี้จึงได้ทดลองเพื่อวัดขนาดของ Sorted buffer ที่เหมาะสมกับแต่ละขนาดวีดิทัศน์ ได้แก่ 640x360, 1280x720 และ 1920x1080 ในหนึ่ง GOP ที่มีขนาด 47 เฟรม โดยมีผลลัพธ์ ดังรูปที่ 5.12



รูปที่ 5.12 จำนวนข้อผิดพลาดในแต่ละขนาดบัฟเฟอร์แบบพิจารณาเส้นทางก่อนหน้า

จากรูปที่ 5.12 แสดงขนาดบัฟเฟอร์ที่เหมาะสมที่ทำให้การแสดงผลของวีดิทัศน์ในขนาด 640x360, 1280x720 และ 1920x1080 สมบูรณ์ คือ 0, 14 และ 23 ตามลำดับ เมื่อคำนวณเวลาที่เสียกับการเพิ่มขนาดของบัฟเฟอร์ที่เหมาะสม ซึ่งผู้วิจัยใช้ขนาดแบนด์วิดท์ค่าต่ำสุดที่ได้จากการคำนวณของสมการที่ (10) ซึ่งได้ค่าแบนด์วิดท์ของแต่ละขนาดวีดิทัศน์ 640x360, 1280x720 และ 1920x1080 เท่ากับ 0.6 เมกะบิต, 1.75 เมกะบิต และ 3 เมกะบิต ซึ่งเป็นการใช้แบนด์วิดท์ที่ต่ำกว่าในการทดลองของ SVC-MST แบบไม่พิจารณาเส้นทางก่อนหน้า เนื่องจากเป็นการคำนวณจากขนาดแต่ละชั้นวีดิทัศน์ เท่ากับ 0.041 วินาที สำหรับการแสดงวีดิทัศน์ 24 เฟรมต่อวินาที เช่นกัน ดังนั้นเวลาที่เกิดขึ้นกับของการแสดงผลของวีดิทัศน์ในขนาด 640x360, 1280x720 และ 1920x1080 ของรูปแบบพิจารณาเส้นทางก่อนหน้าคือ 0 วินาที, 0.574 วินาที และ 0.943 วินาที ตามลำดับ

5.4 วิเคราะห์และสรุปผล

จากการทดลองใช้งานบัฟเฟอร์จัดเรียงลำดับข้อมูลวีดิทัศน์ในรูปแบบการส่งข้อมูลของ SVC-MST แบบไม่พิจารณาเส้นทางก่อนหน้า (SVC-MST non-BWQLB) และ SVC-MST แบบพิจารณาเส้นทางก่อนหน้า (SVC-MST BWQLB) พบว่ารูปแบบพิจารณาเส้นทางก่อนน่าจะต้องใช้ขนาดบัฟเฟอร์ที่มากกว่าเนื่องจากการแยกเส้นทางให้กับแต่ละชั้นวีดิทัศน์ จะไม่มีการควบคุมอัตราแบนด์วิดท์ระหว่างเส้นทางที่ส่งด้วยกัน และแต่ละชั้นของวีดิทัศน์ก็มีขนาดของเฟรม ในการส่งออกที่แตกต่างกันออกไปด้วย จากการวัดค่าในแต่ละขนาดวีดิทัศน์ พบว่าข้อมูลวีดิทัศน์ที่มาไม่ตรงในแต่ละชั้นที่ขนาดแบนด์วิดท์ที่เท่ากันทุกชั้นจะมีลำดับการมาถึงที่ไม่จัดเรียงกันตามอัตราขนาดข้อมูลระหว่างชั้น ถึงแม้ว่า SVC-MST BWQLB มีขนาดบัฟเฟอร์ที่ใหญ่ แต่เมื่อคำนวณเป็นเวลาความล่าช้าที่เกิดขึ้นก่อนการแสดงผล ก็พบว่าเป็นขนาดเวลาที่น้อยมาก

บทที่ 6
สรุปผลและข้อเสนอแนะ

ในบทนี้จะกล่าวสรุปผลการวิจัยสำหรับวิทยานิพนธ์ ข้อเสนอแนะต่าง ๆ อันเป็นประโยชน์ต่องานวิจัยที่เกี่ยวข้องกับรูปแบบการส่งข้อมูลประเภทวีดิทัศน์บนระบบเครือข่ายกำหนดโดยซอฟต์แวร์และข้อเสนอแนะและแนวทางการพัฒนาต่อ

6.1 สรุปผลการวิจัย

ในงานวิจัยนี้ เป็นการพัฒนาระบบการส่งข้อมูลแบบวีดิทัศน์สตรีมมิ่งที่ช่วยลดปัญหาความคับคั่งของการส่งข้อมูลจากหนึ่งผู้ส่งไปยังหลายผู้รับบนระบบเครือข่ายที่กำหนดโดยซอฟต์แวร์ โดยออกแบบการหาเส้นทางที่ควบคุมคุณภาพของการส่งข้อมูลวีดิทัศน์สตรีมมิ่ง จากนั้นผู้วิจัยได้เปรียบเทียบคุณภาพของเส้นทางที่ได้จากการหาเส้นทางที่ได้พัฒนาขึ้นกับการหาเส้นทางแบบดั้งเดิม นอกจากนี้ ผู้วิจัยยังได้เลือกใช้การเข้ารหัสวีดิทัศน์แบบปรับขนาดได้ในการส่งข้อมูลเพื่อลดขนาดการใช้เส้นทางสำหรับการส่งข้อมูลหลายขนาดวีดิทัศน์ไปยังหลายผู้รับที่มีความต้องการของคุณสมบัติวีดิทัศน์ที่แตกต่างกัน

การพัฒนารูปแบบการหาเส้นทางแบบควบคุมคุณภาพ QLB แบบน้ำหนักรวม และ QLB แบบน้ำหนักสมดุล ผู้วิจัยได้นำการหาทั้งเส้นทางทั้ง 2 รูปแบบไปเปรียบเทียบกับการหาเส้นทางบนระบบเครือข่ายแบบดั้งเดิมโดยใช้ OSPF เป็นตัวแทนในการเปรียบเทียบระบบเครือข่ายที่สร้างขึ้นโดยมีคุณภาพของเส้นทางที่ส่งผลต่อคุณภาพวีดิทัศน์ที่แตกต่างกัน โดยมีผลลัพธ์ดังนี้

1. การจัดกลุ่มระดับคุณภาพของ OSPF: มีการพิจารณาเพียงคุณสมบัติเส้นทางเพียงแบนด์วิดท์จึงทำให้สามารถเกิดปัญหาจากการไหวของเวลาและความสูญหายได้
2. การจัดกลุ่มระดับคุณภาพของ QLB แบบน้ำหนักรวม: จะเลือกผลรวมของน้ำหนักที่น้อยที่สุด ทำให้การพิจารณาเลือกเส้นทางไม่สามารถควบคุมคุณภาพที่เหมาะสมต่อการใช้งานได้
3. การจัดกลุ่มระดับคุณภาพของ QLB แบบน้ำหนักสมดุล: จะทำการเลือกเส้นทางที่ควบคุมคุณภาพแม้ว่าคุณภาพเส้นทางด้านใดด้านหนึ่งดีมากแต่กลับมีปัญหาในใดด้านอื่น การจัดลำดับในเส้นทางนั้นจะถูกพิจารณาให้มีน้ำหนักเส้นทางมากกว่าข้อมูลในระดับปานกลางทั้งหมด ดังนั้นการส่งข้อมูลจะอยู่ในขอบเขตที่สามารถแสดงผลได้ปกติมากขึ้น

การทดลองและพัฒนาการจัดการข้อมูลวีดิทัศน์เข้ารหัสมาตรฐาน H.264 ผู้วิจัยได้เปรียบเทียบคุณภาพของมาตรฐานวีดิทัศน์ AVC และ SVC ในส่วนปริมาณการเก็บข้อมูล และการส่งข้อมูลบนระบบเครือข่ายกำหนดโดยซอฟต์แวร์ มีผลลัพธ์ดังนี้

1. ในกรณีการเก็บวีดิทัศน์ในขนาดใดขนาดหนึ่ง: AVC มีความสามารถในการเก็บข้อมูลได้ดีกว่า SVC เนื่องจากภายในเฟรมของ SVC จะต้องมี overhead ที่ใช้ในการแบ่ง NALU ออกเป็นชั้นคุณภาพวีดิทัศน์ ทำให้เมื่อรวมข้อมูลเข้าด้วยกันก็จะมีขนาดที่ใหญ่กว่าขนาดวีดิทัศน์มาตรฐาน AVC ดังนั้น AVC เป็นประเภทวีดิทัศน์ที่เหมาะกับการเก็บข้อมูลในส่วนที่ไม่ต้องการความยืดหยุ่นของการแสดงผลจึงเหมาะกับการเก็บในอุปกรณ์แสดงผลนั้น ๆ
2. ในกรณีการเก็บข้อมูลวีดิทัศน์หลายขนาด: AVC ใช้พื้นที่มากกว่า SVC เนื่องจาก AVC จะเก็บข้อมูลทั้งหมดที่มีในกลุ่ม แต่ SVC จะเก็บข้อมูลขนาดสูงสุดของกลุ่มข้อมูล ดังนั้น SVC มีคุณสมบัติในการเก็บข้อมูลได้ดีในกรณีข้อมูลหลายคุณภาพจึงเหมาะกับผู้ให้บริการวีดิทัศน์

ในส่วนการทดลองการส่งข้อมูลของ AVC และ SVC เนื่องจาก SVC เป็นการเข้ารหัสวีดิทัศน์ที่มีความสามารถในการแบ่งชั้นคุณภาพวีดิทัศน์ได้ จึงมีรูปแบบการส่งข้อมูล ดังนี้ SVC-SST และ SVC-MST ในงานวิจัยนี้ ผู้วิจัยสนใจการจัดการของรูปแบบ SVC-MST แต่ในการทำงานดั้งเดิม (SVC-MST non-BWQLB) เมื่อมีการคำนวณเส้นทางของแต่ละชั้นวีดิทัศน์ การคำนวณในชั้นถัดไปจะไม่รับรู้เส้นทางในการส่งข้อมูลก่อนหน้าทำให้ข้อมูลที่แบ่งชั้นถูกส่งไปยังเส้นทางเดียวกัน เมื่อมีกรณีที่ขนาดแบนด์วิดท์ไม่เพียงพอต่อการส่งข้อมูลสำหรับทุกชั้นก็ยังคงทำให้เกิดปัญหาความคับคั่งและอาจเกิดการสูญหายของข้อมูล ดังนั้นผู้วิจัยจึงได้นำเสนอรูปแบบที่พิจารณาเส้นทางก่อนหน้า (SVC-MST BWQLB) ดังนั้น การทดลองการเปรียบเทียบการส่งข้อมูล ได้แก่ AVC, SVC-SST , SVC-MST non-BWQLB และ SVC-MST non-BWQLB ผู้วิจัยได้แบ่งการทดลองออกเป็น 2 กรณี ได้แก่ กรณีการส่งข้อมูลวีดิทัศน์ในขนาดใดขนาดหนึ่ง และกรณีการส่งข้อมูลแบบหลายขนาดวีดิทัศน์ ซึ่งมีผลลัพธ์ดังนี้

1. กรณีการส่งข้อมูลวีดิทัศน์ในขนาดใดขนาดหนึ่ง มีผลกระทบต่อข้อจำกัดรูปแบบการส่งข้อมูลของ SVC-MST non-BWQLB เนื่องจากข้อจำกัดของรูปแบบการส่งให้ปลายทางที่สามารถส่งข้อมูลเพียงหนึ่งเส้นทางทำให้มีรูปแบบที่เหมือนกับ SVC-SST และ AVC แต่ AVC มีขนาดวีดิทัศน์ที่เล็กกว่า SVC จึงทำให้มีค่าเฉลี่ยความล่าช้าต่ำกว่าทั้ง SVC-SST และ SVC-MST non-BWQLB แม้ว่า SVC-MST BWQLB จะเป็น

การส่งข้อมูล SVC แต่คุณสมบัติการแยกเส้นทางของ SVC SVC-MST BWQLB จึงทำให้สามารถลดความล่าช้าโดยรวมได้มากที่สุด

2. กรณีการส่งข้อมูลแบบหลายขนาดวีดิทัศน์ จะแสดงคุณสมบัติของ SVC-MST BWQLB ได้อย่างมีประสิทธิภาพ เนื่องจากเป็นการรวมคุณสมบัติการแยกชั้นวีดิทัศน์ได้ของ SVC-MST และการแยกเส้นทางได้จากการแบ่งระดับคุณภาพจากวิธีการ BWQLB ทำให้การส่งข้อมูลแบบกลุ่ม SVC-MST BWQLB จะทำการส่งเพียงวีดิทัศน์ขนาดสูงสุดเหมือนกับ SVC-MST non-BWQLB แต่จะมีความแตกต่าง เมื่อเกิดความคับคั่งในเส้นทางก็จะทำการแยกชั้นวีดิทัศน์ออกไปยังเส้นทางที่หลากหลาย ดังนั้น SVC-MST BWQLB จึงลดความล่าช้าได้มากที่สุด

แม้ว่า SVC-MST ทั้งแบบ non-BWQLB และ BWQLB จะเป็นรูปแบบการส่งข้อมูลที่สามารถลดความล่าให้กับการส่งข้อมูลได้ แต่ผลลัพธ์ในการรับข้อมูลวีดิทัศน์นั้นไม่ถูกต้องกับการจัดเรียงข้อมูลที่ใช้ในการแสดงผล ดังนั้นผู้วิจัยจึงได้ออกแบบบัฟเฟอร์ที่ใช้ในการจัดเรียงข้อมูลก่อนส่งไปยังบัฟเฟอร์ที่ใช้แสดงผล ผู้วิจัยได้แบ่งการตรวจสอบจากการทดลองเพิ่มขนาดบัฟเฟอร์จนทำให้ข้อมูลการแสดงผลไม่เกิดความผิดพลาด โดยมีผลการตรวจสอบดังนี้

1. การตรวจสอบใน SVC-MST non-BWQLB เกิดปัญหาลำดับที่ไม่ตรงกันเพียงเล็กน้อยเนื่องจากรูปแบบการส่งข้อมูลยังคงอยู่ในเส้นทางเดียวกัน แต่แยกชั้นวีดิทัศน์เป็นกลุ่มเท่านั้น
2. การตรวจสอบใน SVC-MST BWQLB ในกรณีการแยกเส้นทางอย่างสมบูรณ์ จะเกิดลำดับของข้อมูลที่ไม่ตรงกันอย่างเห็นได้ชัด เนื่องจากการแยกชั้นวิดีทัศน์หลากหลายเส้นทางจะทำให้เกิดความเร็วของการมาถึงของข้อมูลในแต่ละชั้นตามอัตราขนาดแต่ละชั้นวีดิทัศน์นั้น ๆ อย่างไรก็ตามขนาดของบัฟเฟอร์ที่เกิดขึ้นเมื่อเปรียบเทียบกับขนาดบัฟเฟอร์โดยทั่วไป ก็ยังคงมีขนาดที่ไม่สูงมาก

แม้ว่า SVC-MST BWQLB มีขนาดบัฟเฟอร์ที่สูงกว่า SVC-MST non-BWQLB เมื่อมีการแยกเส้นทางของชั้นคุณภาพวีดิทัศน์ออกจากกันที่ขนาดแบนด์วิดท์ต่ำจนทำให้เกิดปัญหาความคับคั่ง แต่การส่งข้อมูลแบบ SVC-MST non-BWQLB ก็เกิดปัญหาการสูญหายของข้อมูล ดังนั้น SVC-MST non-BWQLB จึงมีคุณภาพของการทำงานที่ต่ำกว่า SVC-MST BWQLB นอกจากนี้ SVC-MST BWQLB ที่แยกเส้นทางจะมีความล่าช้าของการส่งข้อมูลวิดิทัศน์ที่ต่ำกว่า SVC-MST non-BWQLB เป็นอย่างมาก

6.2 ประโยชน์ต่อวงวิชาการ

1. นำเสนอวิธีการพิจารณาเส้นทางการหาข้อมูลอย่างสมดุล
2. แสดงคุณสมบัติของคุณภาพด้านต่าง ๆ ของเส้นทางที่เกิดขึ้นเมื่อทำการคำนวณเส้นทางการส่งข้อมูลประเภทวีดิทัศน์
3. แสดงการเปรียบเทียบรูปแบบการทำงานของการจัดการระบบเครือข่ายแบบดั้งเดิมและการจัดการระบบเครือข่ายด้วยซอฟต์แวร์ที่พัฒนาขึ้น
4. แสดงการวิเคราะห์การเก็บข้อมูลประเภท AVC และ SVC โดยวัดปริมาณข้อมูลหลากหลายขนาดวีดิทัศน์
5. แสดงการเปรียบเทียบรูปแบบการส่งข้อมูลในแบบ AVC, SVC-SST และ SVC-MST
6. แสดงรูปแบบในการใช้งาน SVC-MST แบบพิจารณาเส้นทางก่อนหน้า
7. แสดงประโยชน์ในการเห็นความสำคัญของเส้นทางที่ถูกคำนวณไม่ต่ำกว่า 3 เส้นทางก่อนส่งในเวลาเดียวกันของ SVC 3 ชั้นคุณภาพ
8. แสดงการแก้ปัญหาลำดับข้อมูลที่เกิดจากการส่งข้อมูลแบบ SVC-MST ที่ไม่ถูกต้อง

6.3 ข้อเสนอแนะและแนวทางการพัฒนาต่อ

จากการพัฒนาระบบการส่งวีดิทัศน์ที่ปรับขนาดได้หลายผู้รับบนระบบเครือข่ายที่กำหนดโดยซอฟต์แวร์ ผู้วิจัยมีข้อเสนอแนะและแนวทางการพัฒนาต่อดังต่อไปนี้

1. ทดลองในรูปแบบการปรับขนาด SVC ในรูปแบบอื่น เช่น ปรับขนาดเชิงเวลา และปรับขนาดเชิงคุณภาพ
2. พัฒนามาตรฐานวีดิทัศน์จาก H.264 เป็น H.265 โดยเปลี่ยน AVC เป็น HEVC และ SVC เป็น SHVC
3. ขยายขนาดวีดิทัศน์ที่ใช้การทดลองให้มีความหลากหลายมากขึ้น
4. ขยายระบบเครือข่ายให้มีขนาดใหญ่และหลากหลายรูปแบบมากขึ้น
5. พิจารณาการแบ่งระดับคุณภาพของวีดิทัศน์ให้หลากหลายมากขึ้น
6. ทดลองบนอุปกรณ์จริงและทดลองในระบบไร้สาย

7. หาความสัมพันธ์ของแบนด์วิดท์ที่ส่งผลกับค่าความล่าช้า, การไหวของเวลา และความสูญหาย ในรูปแบบอื่น ๆ
8. ขยายความสามารถของบัฟเฟอร์จัดเรียงโดยการสร้างการจัดเรียงให้กับกลุ่มในระดับ GOP
9. พัฒนาการส่งข้อมูลแบบ SVC-MST แบบพิจารณาเส้นทางก่อนหน้าในส่วนของการจัดการส่งข้อมูลโดยควบคุมอัตราเร็วในการส่งของแต่ละชั้น เพื่อลดการทำงานของบัฟเฟอร์ในฝั่งผู้รับ

เอกสารอ้างอิง

[1] "White paper: Cisco VNI forecast and methodology, 2015-2020," *Cisco*. [Online]. Available: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/complete-white-paper-c11-481360.html. [Accessed: 06-Jun-2017].

[2] S. Laga *et al.*, "Optimizing scalable video delivery through OpenFlow layer-based routing," in *Network Operations and Management Symposium (NOMS), 2014 IEEE*, 2014, pp. 1–4.

[3] T.-F. Yu, K. Wang, and Y.-H. Hsu, "Adaptive routing for video streaming with QoS support over SDN networks," in *Information Networking (ICOIN), 2015 International Conference on*, 2015, pp. 318–323.

[4] E. Yang, Y. Ran, S. Chen, and J. Yang, "A multicast architecture of SVC streaming over OpenFlow networks," in *Global Communications Conference (GLOBECOM), 2014 IEEE*, 2014, pp. 1323–1328.

[5] J. Yang, E. Yang, Y. Ran, and S. Chen, "SDMˆ2 cast an OpenFlow-based, software-defined scalable multimedia multicast streaming framework," *IEEE Internet Comput.*, vol. 19, no. 4, pp. 36–44, 2015.

[6] N. Xue, X. Chen, L. Gong, S. Li, D. Hu, and Z. Zhu, "Demonstration of OpenFlow-controlled network orchestration for adaptive SVC video manycast," *IEEE Trans. Multimed.*, vol. 17, no. 9, pp. 1617–1629, 2015.

[7] J. S. V. Model, "JSVM software, joint video team, Doc," *JVT-X203 Geneva Switz.*, 2007.

[8] W. Xia, Y. Wen, C. H. Foh, D. Niyato, and H. Xie, "A survey on software-defined networking," *IEEE Commun. Surv. Tutor.*, vol. 17, no. 1, pp. 27–51, 2015.

[9] S. Z., C. M., and P. W., "Scalable video coding effective video coding for multimedia applications." 2011.

[10] V. Jacobson, R. Frederick, S. Casner, and H. Schulzrinne, "RTP: A transport protocol for real-time applications," 2003.

[11] A. Detti *et al.*, “SVEF: an open-source experimental evaluation framework for H. 264 scalable video streaming,” in *Computers and Communications, 2009. ISCC 2009. IEEE Symposium on*, 2009, pp. 36–41.

[12] “Enhanced interior gateway routing protocol (EIGRP),” *Cisco*. [Online]. Available: http://www.cisco.com/c/en/us/error/404.html. [Accessed: 06-Jun-2017].

[13] W. Sugeng, J. E. Istiyanto, K. Mustofa, and A. Ashari, “The impact of QoS changes towards network performance,” *Int. J. Comput. Netw. Commun. Secur.*, vol. 3, no. 2, pp. 48–53, 2015.

[14] S. Skiena, “Dijkstra’s algorithm,” *Implement. Discrete Math. Comb. Graph Theory Math. Read. MA Addison-Wesley*, pp. 225–227, 1990.

[15] J. Hintersteiner, “Quality of Service - A Simple Explanation.” .

[16] Z. Wang and J. Crowcroft, “Quality-of-service routing for supporting multimedia applications,” *IEEE J. Sel. Areas Commun.*, vol. 14, no. 7, pp. 1228–1234, 1996.

[17] “GroupFlow: Multicast routing in software defined networks — GroupFlow 0.2.0.” [Online]. Available: http://alexcraig.github.io/GroupFlow/. [Accessed: 06-Jun-2017].

[18] H. Holbrook, B. Cain, and B. Haberman, “Using internet group management protocol version 3 (IGMPv3) and multicast listener discovery protocol version 2 (MLDv2) for source-specific multicast,” 2006.

[19] B. Cain, S. Deering, I. Kouvelas, B. Fenner, and A. Thyagarajan, “Internet group management protocol, version 3,” 2002.

[20] “DASH SVC dataset | dynamic adaptive streaming over HTTP with Scalable Video Coding.” [Online]. Available: http://concert.itec.aau.at/SVCDataset/. [Accessed: 06-Jun-2017].

[21] “Understanding buffer misses and failures - Cisco.” [Online]. Available: http://www.cisco.com/c/en/us/support/docs/interfaces-modules/channel-interface-processors/14620-41.html. [Accessed: 06-Jun-2017].

[22] “Customizing the size of the streaming buffer,” *BrightSign Support*. [Online]. Available: http://brightsign.zendesk.com/hc/en-us/articles/218067097-Customizing-the-size-of-the-streaming-buffer. [Accessed: 06-Jun-2017].

ภาคผนวก ก สรุปขั้นตอนวิธีที่ใช้ในงานวิจัย

Dijkstra_Algorithm function

Input: *G=(V,E), ew, s, MG*

Output: *MT*

```
1:  T={s}; d[s] ⟵ 0; d[u] ⟵ ∞ and p[v] ⟵ 0 for each u ≠ s, u ⊂ V
2:  insert u with key d[u] into the priority queue Q, for each u ⊂ V
3:  while (Q!=null)
4:      u ⟵ Minimum_distance(Q)
5:      for each v adjacent to u do
6:          if d[v] > d[u]+ew[u,v] then
7:              d[v] ⟵ d[u]+ew[u,v]
8:              p[v] ⟵ d[u]
9:              add v into Tree
10: Return MT, the subtree of Tree rooted at as associsted with MG
```

Modify_Dijkstra_Algorithm function

Input: G, s

Output: dis[V], pre[V]

```
1:  for each v in V
2:      dis[v] ⟵ ∞
3:      pre[v] ⟵ null
4:  Q ⟵ set(node.keys)
5:  while (Q!=null)
6:      u ⟵ Modify_Minimum_distance(d,Q)
7:      for each v adjacent to u do
8:          adj[v][][0] ⟵ Min (dis[u][0], ew[u,v][][0])
9:          adj[v][][1] ⟵ dis[u][1]+ew[u,v][][1]
10:         adj[v][][2] ⟵ dis[u][2]+ew[u,v][][2]
11:         adj[]⟵ Quantized_to_3levels (adj[])
12:         adj[]⟵ Total_Weight (adj[]) or
                                    Balance_Weight(adj[])
13:     if dis[v][0] = adj[v][][P] then
14:         if dis[v][1] > adj[v][][P+1] then
15:             dis[v][0] ⟵ adj[v][][P]
16:             dis[v][1] ⟵ adj[v][][P+1]
17:             pre[v][0] ⟵ dis[v][0]
18:             pre[v][1] ⟵ dis[v][1]
19:     else if dis[v][0] > adj[v][][P] then
20:         dis[v][0] ⟵ adj[v][][P]
21:         dis[v][1] ⟵ adj[v][][P+1]
22:         pre[v][0] ⟵ dis[v][0]
23:         pre[v][1] ⟵ dis[v][1]
```

Modify_Minimum_distance function

Input: distance, Q

Output: node, min

```
1:   min_level ⟵ ∞ ;
2:   min_mix ⟵ ∞ ;
3:   node ⟵ 0
4:   for v in Q do
5:       adj_node[] ⟵ Quantized_to_3levels (distance[])
6:       adj_node[] ⟵ Total_Weight (distance []) or
                                  Balance_Weight(distance [])
7:       if adj_node [v][][P] = min_level then
8:           if adj_node [v][][P+1] < min_mix then
9:               min_level ⟵ adj_node [v][][P]
10:              min_mix ⟵ adj_node [v][][P+1]
11:              node ⟵ v
12:      else if adj_node [v][][P] < min[0] then
13:          min_level ⟵ adj_node [v][][P]
14:          min_mix ⟵ adj_node [v][][P+1]
15:          node ⟵ v
```

Total_Weight function

Input: ew

Output: ew

```
1: ew[][][P+1] ⟵ sum(each ew[0])
2: ew[][][P+1] ⟵ (ew[][][P+1] – (L-1)) / MG
```

Check_level_parameter function

Input: **ew**, **P, L**

Output: count

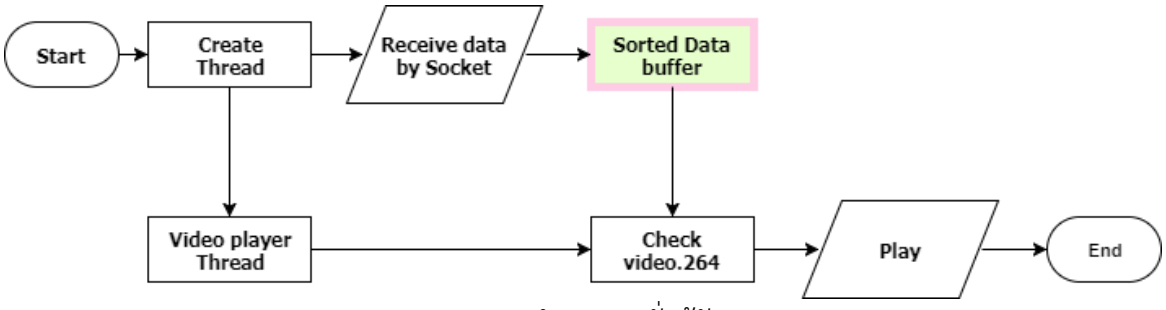
```
1:   for i=1 to P do
2:       for j=1 to L do
3:           if ew[j] = i
4:               count[i]+1
```

Balance_Weight function

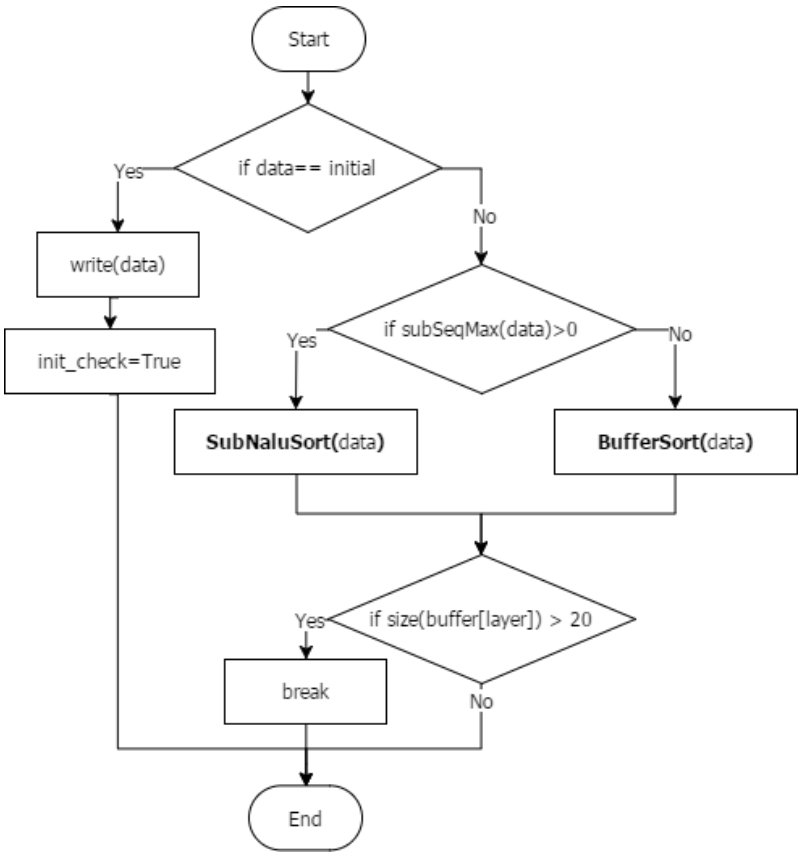
Input: ew

Output: ew

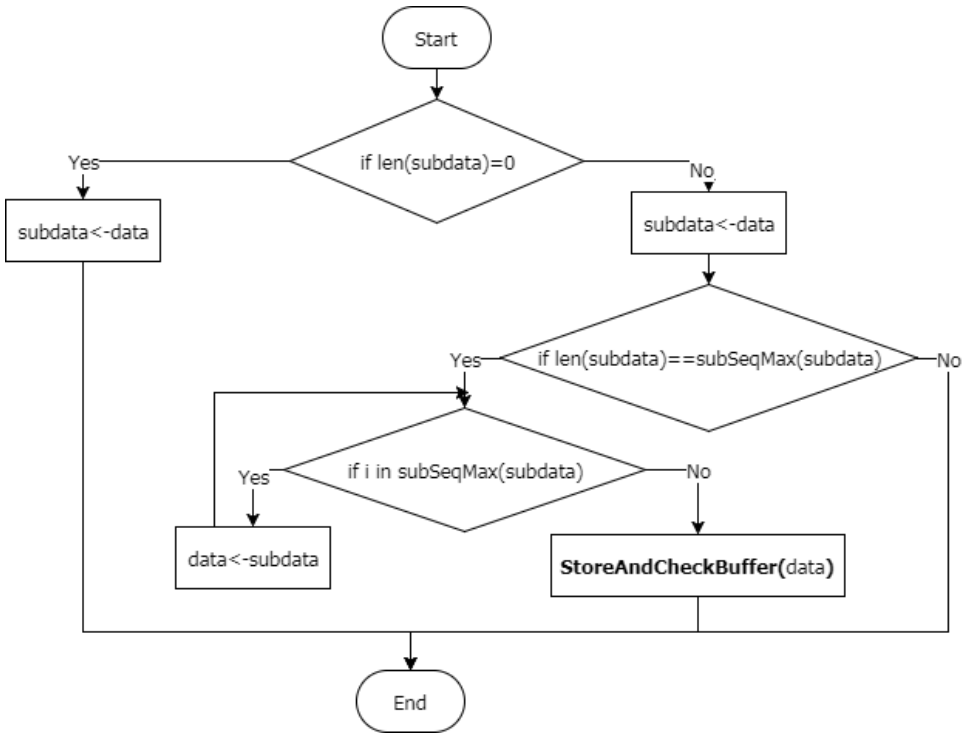
```
1:   count[]  ⟵ Check_level_parameter(ew, P, L)
2:   if count[1] = L
3:       ew[][][P+1] ⟵ 1
4:   if count[2] > 0 and count[3] = 0
5:       for n=1 to P do
6:           if count[1] = P-n and count[2] = n
7:               ew[][][P+1] ⟵ n+1
8:   if count[3] > 0
9:       for n=1 to P+1 do
10:          if count[3] = n
11:              for m= 0 to P-1
12:                  if count[1] = P-n-m and count[2] = m
13:                      if n=1
14:                          ew[][][P+1] ⟵ P+m+n
15:                      else
16:                          ew[][][P+1] ⟵ P+m+(2*n)
```



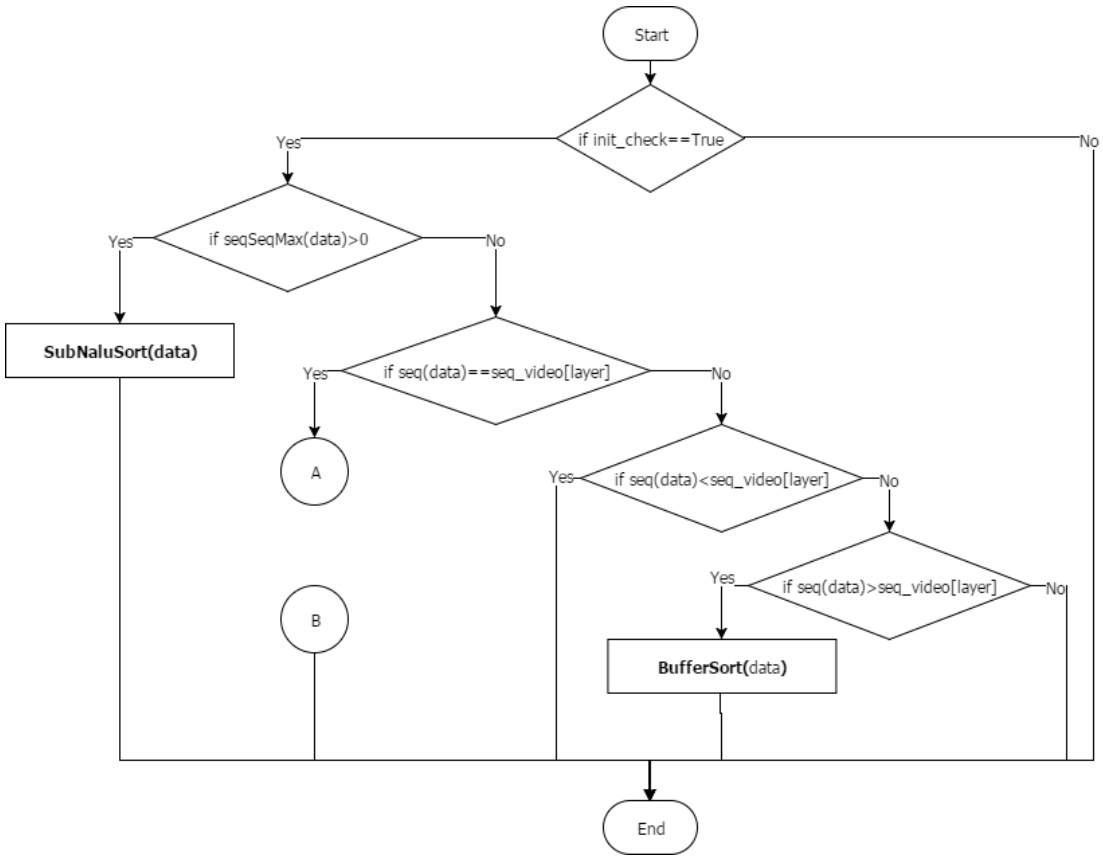
ภาพรวมการทำงานของฝั่งผู้รับ



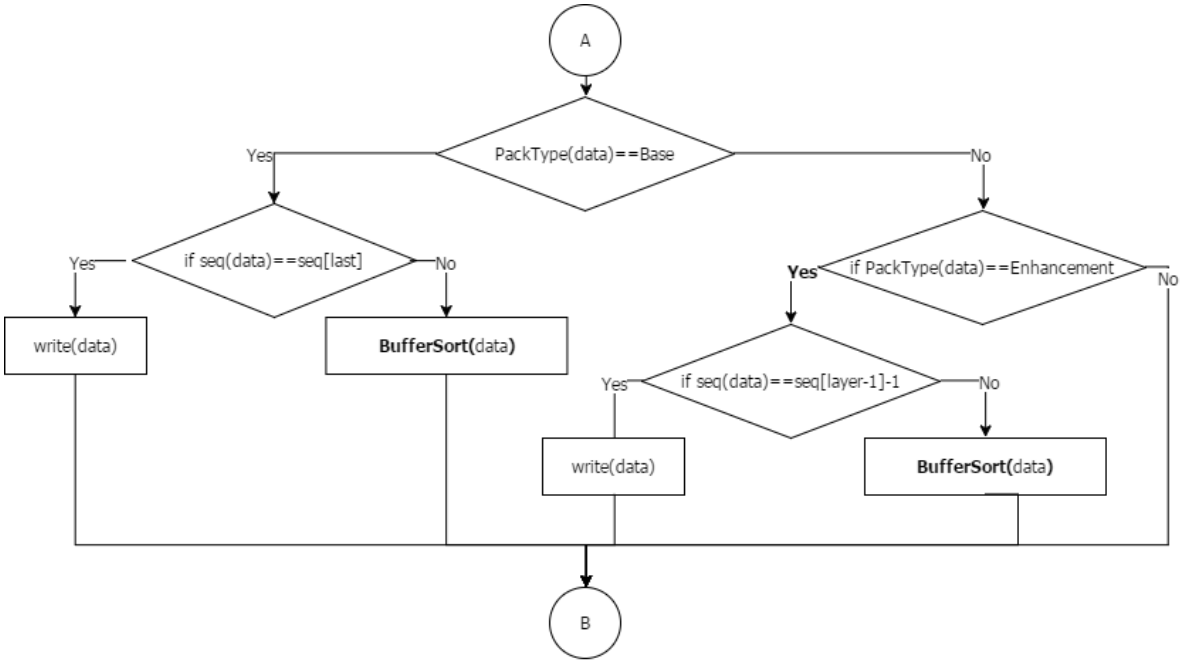
InitialChecker function



SubAccessUnitSort function

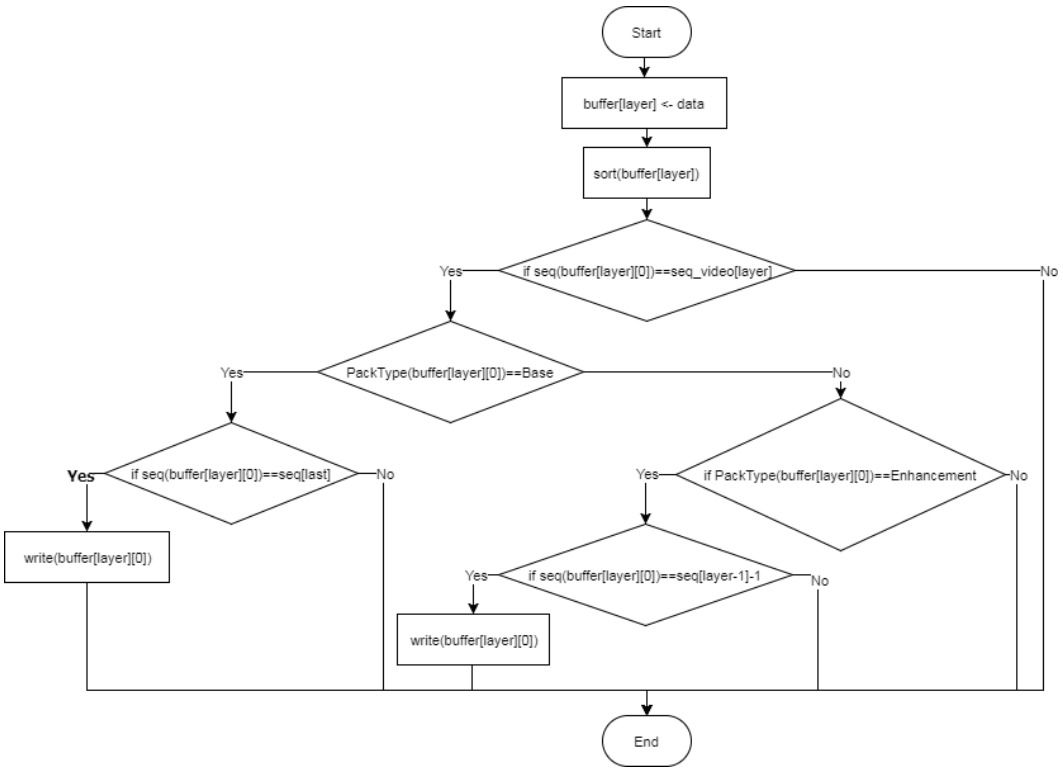


(ก)

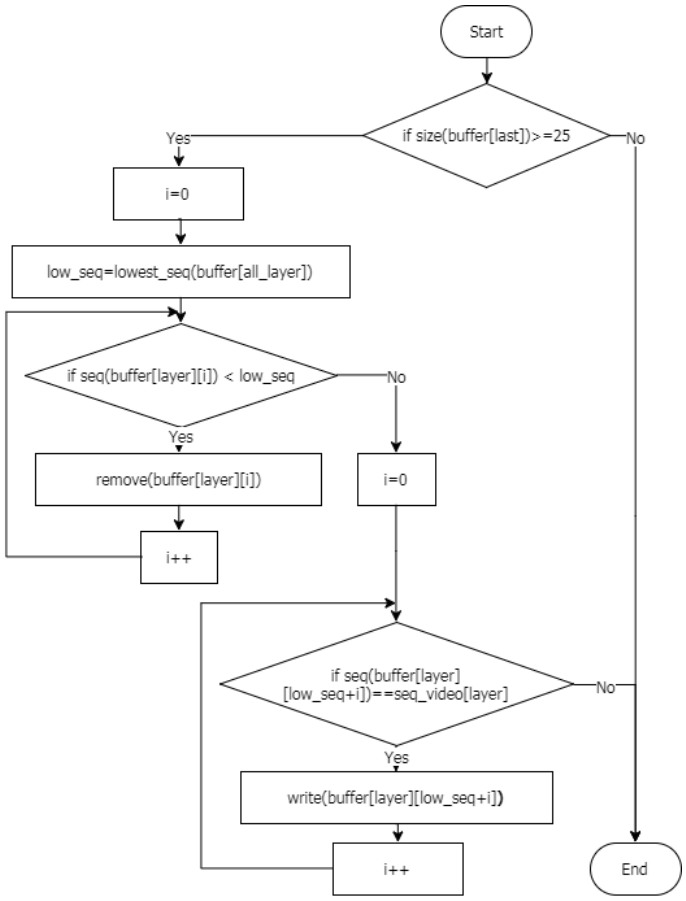


(ข)

AccessUnitSort function (ก) ตรวจสอบลำดับภายในชั้น (ข) ตรวจสอบลำดับระหว่างชั้น



BufferSort function



BufferFullChecker function

ภาคผนวก ข ผลงานที่ได้รับการตีพิมพ์

SDN Experimental on the PSU Network

Piyawit Tantisarkhornkhet, Warodom Werapun
Department of Computer Engineering,
Faculty of Engineering, Prince of Songkla University
Phuket, Thailand
Email: {5810120060@email.psu.ac.th,
warodom@coe.phuket.psu.ac.th}

Beatrice Paillassa
University of Toulouse, IRIT Laboratory,
INP – ENSEEIHT
Toulouse, France
Email: beatrice.paillassa@enseeiht.fr

***Abstract*— The Prince of Songkla University (PSU) is the top five universities of Thailand that still uses traditional networking. Presently, there are Internet applications that are able to provide dynamic, manageable, and adaptable features such as Software-defined Networking (SDN). SDN is a new concept of programmable networks that decouples the control plane and data plane of all network devices. It can be programmed via an open interface which is interesting to implement SDN because of their various benefits such as centralized network provisioning, lower operating costs etc. In this paper, we propose comparison between traditional PSU network which is defined Static routing (Static routing non-SDN) and SDN which are using programmable algorithm such as Bellman-Ford SDN (BFSDN) unicast, Dijkstra SDN (DSDN) on both unicast and multicast in order to determine worthiness of migration from traditional network to SDN. In part of topology emulator, we have replicated topology by Mininet. Its performance is examined in terms of throughput, latency, jitter, and packet loss.**

Keywords—Software-defined networking; Unicast; Multicast; OpenFlow; programmable networks; Dijkstra; Bellman-Ford;

I. INTRODUCTION

In a traditional network, it is built from different network devices such as routers, switches, and hubs, etc. with various protocols implemented on them [1]. Many device types lead to several configuring policies. It makes network management and performance tuning rather complex and error occurred. Moreover, the current Internet applications and services are more complex. They are imperative that the Internet is able to address these new challenges.

Software Define Networking is a new emerging networking approach under network programmable concept. It is an idea to separate the control plane and data plane of all network devices that can be programmed via an open interface such as ForCES [2], OpenFlow [3], etc. It is simplified network management and enable innovation plus evolution. It serves a network manager to be centralized, manage and increase granular security. In a commercial aspect, it allows to reduce operating costs, save hardware and minimize capital expenditures [4]. However, all transmission devices must support SDN protocol likes OpenFlow. On the other hand, the most traditional transmission device does not support it.

In this paper, we simulate and compare traditional PSU network, BFSDN unicast, DSDN both unicast and multicast in order to explore the value of changing from traditional networking to SDN.

The rest of this paper is structured as follows. In the next section, we present preliminaries on the work, such as Software Defined Networking, Communications and Routing algorithm, Mininet and PSU network topology. In Section III, we illustrate the simulation results. Finally, this paper is concluded with Section IV.

978-1-5090-0629-8/16/$31.00 ©2016 IEEE

II. PRELIMINARIES

A. *Software Defined Networking*

Software Defined Networking was developed to facilitate innovation and enable simple programmatic control of the network data-path. It isolates controller hardware from forwarding hardware. The SDN structure enables the consolidation of device which is simpler policy management, new functionalities.

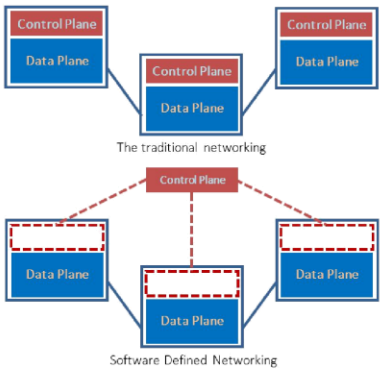


Fig. 1. Traditional network and Software-Defined Network

SDN architectures have ForCES [2] and OpenFlow [3]. Both OpenFlow and ForCES follow the basic SDN principle of separation between the control and data planes. They standardize information exchange between planes. In this paper, we focus on OpenFlow architecture. SDN-OpenFlow architecture consists of application layer, control layer, and infrastructure layer.

- **Application layer or management layer** is the top layer. It consists of network operation tools and user interfaces that management of the network via the control layer by OpenFlow protocol.
- **Control layer** is the middle layer using for control all devices in infrastructure layer via OpenFlow protocol. SDN controller such as NOX/POX [5], Ryu [6], Floodlight [7] etc.
- **Infrastructure layer** is the lowest layer. It includes all devices in traffic forwarding that all devices must support OpenFlow protocol.

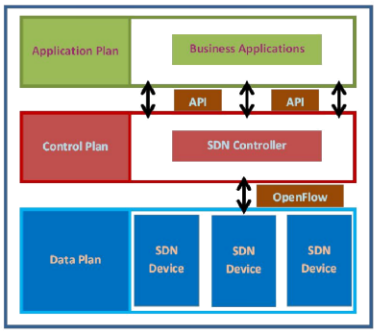


Fig. 2. The SDN architecture

OpenFlow is a flow-based switch specification designed to enable researchers to run experiments in live networks. It is based on a simple Ethernet flow switch which exposes a standardized interface for adding and removing flow entries. OpenFlow protocol is an open protocol defined between the control plane device and the data plane device [8].

B. Communications

In this section, two types of communications as unicast and multicast are reviewed. They have advantages and disadvantages which are depended on situation of communication.

1) Unicast

Unicast is a type of communication where data is sent from one source to one destination. Unicast uses protocol of transport layer such as Transmission Control Protocol (TCP) and User Datagram Protocol (UDP), which are session-based protocols. In this work, we use Dijkstra shortest path and Bellman-Ford shortest path which are popular algorithms [9].

a) Dijkstra's algorithm

The Dijkstra method is algorithm for finding the optimum path between nodes in a directed graph G= (V, E), where V is the set of nodes and E is the set of edges. Dijkstra's algorithm shows in Fig 3, Initially, d[s]=0, d[u]=∞ for u∈V, u≠s, and p[u]=null for u∈V.

```
Dijkstra's Algorithm
Input: G=(V, E), s
Output: d[V], p[V]
1: For each v in V
2:     d[v] ← ∞
3:     p[v] ← null
4: insert u with key d[u] into the priority queue Q, for each
u∈V
5: while (Q!=null)
6:    u← Minimum_distance(Q)
7:    for each v adjacent to u
8:       if d[v] > d[u]+ew[u,v] then
9:          d[v]←d[u]+ew[u,v]
10:         p[v]←d[u]
```

Fig. 3. The Dijkstra's Algorithm

In [10] Jehn-Ruey Jiang and team propose Extending Dijkstra's algorithm that adds the node weight analysis in their algorithms. From Fig 3, add the node weight (nw) in line 6 and line 7 of Dijkstra's algorithm.

b) Bellman-Ford's algorithm

Bellman-Ford method is an algorithm to find the optimum path between nodes in a directed graph likes Dijkstra's algorithm. However, it is able to analyze negative edge weights. It provides more features than Dijkstra algorithm. In contrast, if Bellman-Ford is used over non-negative weights, its performance is slower than Dijkstra. From Fig. 4, shows Bellman-Ford's algorithm which consists of three main steps as initialize graph, Relaxation of edges and checking negative cycle.

```
Bellman-ford's Algorithm
Input: G=(V, E), s
Output: d[V], p[V]
1) Initialize graph
1: For each v in V
2:          d[v] ← ∞
3:          p[v] ← null
2) Relaxation of edges
1: For i from 1 to IVI-1
2:          For each [u,v] with ew in E
3:                    If d[u]+w < d[v]
4:                              d[v] ← d[u]+w
5:                              p[v] ← u
3)Checking negative cycle
1: For each ew(u ,v) in E
2:          If d[u]+ew < d[v]
3:          Print "Negative cycle"
```

Fig. 4. Bellman-ford's Algorithm

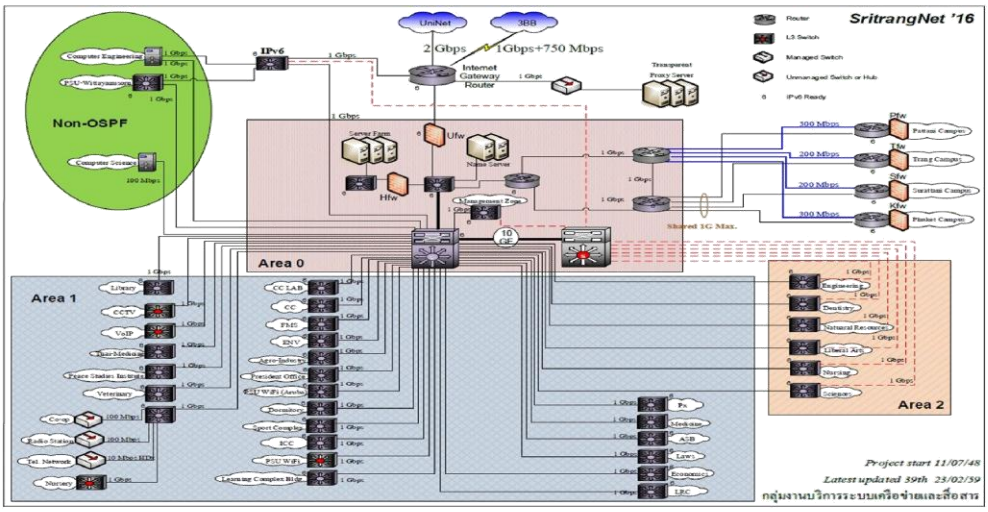


Fig. 5. PSU network topology

2) Multicast

Multicast is the communication network ability of sending IP packets to member groups. It can reduce transmission overhead from an application to send copies of data to all group members. An advantage over than unicast is to minimize network traffic and optimize bandwidth where multiple users are accessing the same live video source.

GroupFlow [11] is one SDN-driven approach to IP multicast routing that has already been proposed. It is based on CastFlow [12]. SDN controller of GroupFlow builds on the POX framework. Routing protocol of GroupFlow is Protocol Independent Multicast (PIM) which constructs shared trees centered at a rendezvous-point (RP) router. The PIM, there are two varieties as PIM dense mode (PIM-DM) and PIM sparse mode (PIM-SM) only establishes routes to the designated router (DRs) reactively to group joins.

There is some difference between multicast non-SDN and multicast SDN. Multicast SDN can add more membership security to the network which is a normal problem in multicast non-SDN. In SDN, a client joins a multicast group, its membership is assumed to authenticate at a server to a controller. GroupFlow (Fig. 6) adds more membership security in IGMP JOIN phase. GroupFlow multicast model uses Dijkstra's algorithm to calculate a shortest path from the multicast source to all multicast members in the network. Each shortest path is added to multicast tree construction.

C. Tools

Mininet [13] is a topology emulator program that supports SDN based on Linux kernel. It can create network virtual devices as hosts, OpenFlow switches, legacy switches, routers, links, and controllers. It is suitable for researching, testing, and debugging etc. Mininet brings many scripts with user-friendly GUI. In addition, BRITE [14] has been used to create our tested topologies. It supports multiple generation models and can be used as input topologies in Mininet. Although the real network devices would have some cross-traffic, CPU and bus architectures specific to each hardware component that Mininet does not have, Mininet reveals the trend of performance efficiently.

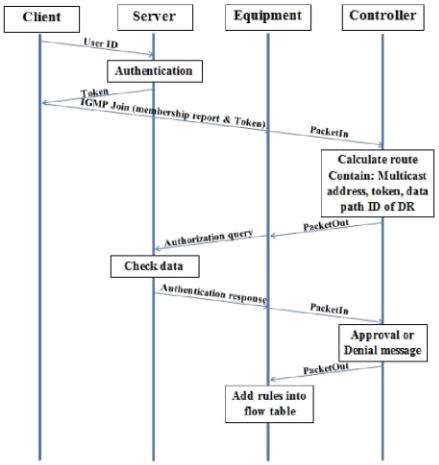


Fig. 6. IGMP Join of GroupFlow model

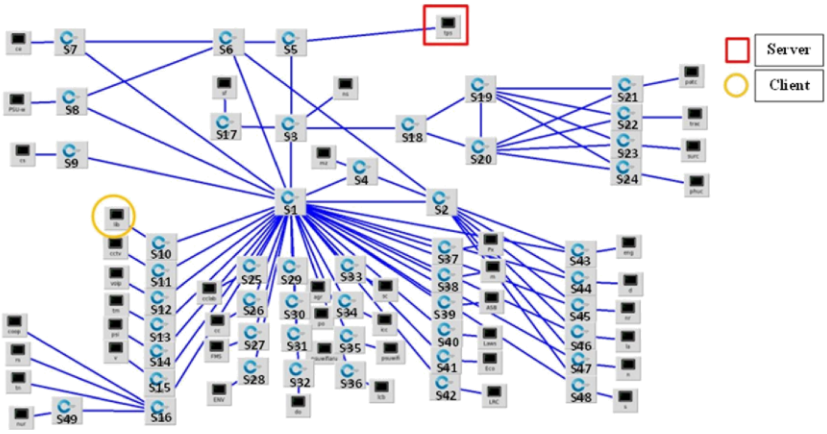


Fig. 7. PSU Topology on Mininet

D. PSU network topology

The Prince of Songkla University (PSU) uses traditional network that is the extended star topology. This real existed topology can be a representative of some organizations or companies on a similar scale. Fig. 5 shows PSU topology [15], it consists of different transmission node types such as router, L2 switch, managed switch and unmanaged switch or hub. Most bandwidth between L2 switches is 1 Gbps. We present two previously described transmission types such as unicast and multicast to test network performance since the PSU network topology has various transmission types.

III. EXPERIMANTAL RESULTS

We set up POX [16] OpenFlow controller and PSU topology in the Mininet while all switches are linked to the controller. Fig. 7 shows a PSU network topology which is simulated from Mininet. It provides most 1 Gbps links between peripheral nodes to the central nodes [15]. In order to evaluate the performance of the PSU networking used SDN and Static non-SDN, we use Iperf [17] to test them such as throughput, latency, jitter and packet loss. Experiments are conducted on a single computer with CPU Intel ® core TM i7-4790 3.60 GHz processor and DDR3 8 GB RAM.

TABLE I. SIMULATION SETTINGS

Parameter	Value
Number of hosts	45
Number of nodes	49
Number of links	109
Test times	100 seconds
Controller	POX 0.2.0 support OpenFlow 1.0
Test tools	Mininet

In the simulation experiments, we measure the performance for Static routing non-SDN and routing algorithm SDN. We use Iperf to create TCP and UDP. We assign one server (circle) and one client (square) because they have many paths such as s10-s1-s3-s5, s10-s1-s6-s5 and s10-s1-s7-s6-s5 etc. Their minimum bandwidth is 1 Gbps.

Fig 8 shows throughput of TCP. The results of Static routing non-SDN, BFSDN unicast and DSDN unicast are 943.25 mbps, 954.50 mbps and 954.69 mbps, respectively. DSDN unicast is slightly better than BFSDN unicast. Static routing non-SDN has lowest throughput. Therefore, BFSDN unicast and DSDN unicast manage bandwidth better than Static routing non-SDN in TCP transmission.

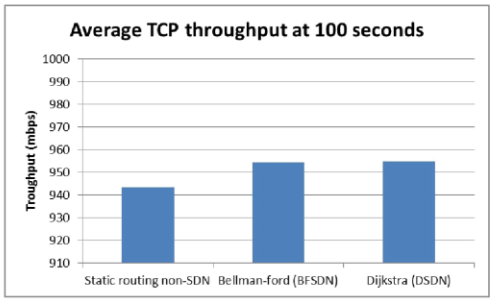


Fig. 8. The average TCP throughput

Fig 9 shows latency of Static routing non-SDN, BFSDN unicast and DSDN unicast by executing Ping command which sends ICMP request messages. Their average latencies are 0.0907 ms, 0.3934 ms and 0.3785 ms respectively. Comparison between algorithms related to SDN, BFSDN unicast latency is larger than DSDN unicast latency, which is usually caused by the depending inverse

throughput. Static routing non-SDN has the lowest latency. Both BFSDN unicast and DSDN unicast on the SDN controller have to take the time calculating the route and install suitable paths on OpenFlow switches. Therefore, more amount of data transfer takes place between nodes in a specific time period for SDN than traditional network.

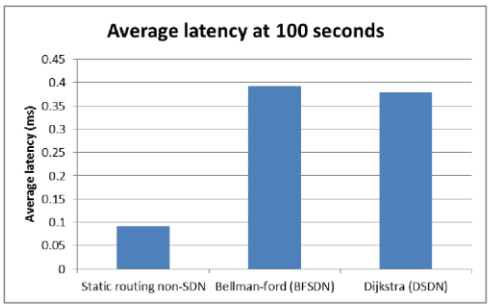


Fig. 9. The average latency

Their initial latency values are 0.9227 ms, 32.8987 ms and 31.7150 ms respectively. When we calculate average latency without an initial value as shown in Fig. 10, their average latency will be 0.0823 ms, 0.0651 ms and 0.0620 ms. Static routing non-SDN has worse latency than others. The initial latency values of SDN affects to average latency. In addition, we examine interval time that SDN has average latency lower than Static routing non-SDN by changing the experiment time from 100 seconds to 2000 seconds. The average latency of DSDN unicast and BFSDN unicast are lower than Static routing non-SDN at 990 seconds and 1619 seconds respectively. However, we do not include them in the figure since it would make the data impossible to read. Therefore, SDN is greater than Static routing non-SDN in several performance metrics on the long term used.

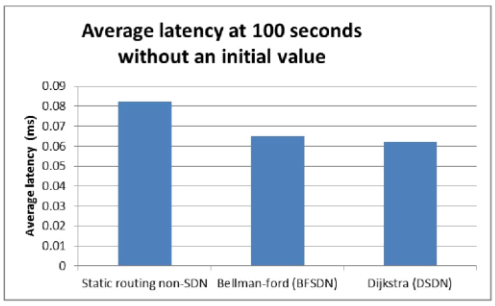


Fig. 10. Average Latency without an initial value

The result of UDP transmission as shown in Figure 11-13, each bandwidth requirement (x-axis) takes 100 seconds for experiment time. Static routing non-SDN, BFSDN unicast, DSDN unicast and Dijkstra multicast SDN (GroupFlow) have been tested for throughput of UDP as shown in Fig 11. All routing are fully well performed until BFSDN unicast began down and stable at 750 mbps, DSDN unicast began down and stable at 800 mbps and Static routing non-SDN and GroupFlow began down and stable at 850 mbps.

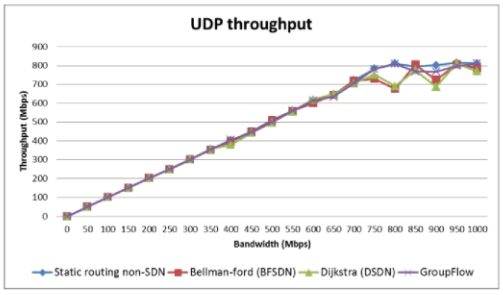


Fig. 11. The UDP throughput

Main factors of jitter and packet loss are depended on bandwidth congestion. At the maximum bandwidth, BFSDN unicast, DSDN unicast and GroupFlow (SDN-related) have high jitter as shown in Fig. 12. Average jitter algorithm are ranked as follow as BFSDN unicast, DSDN unicast and GroupFlow. Static routing non-SDN has the lowest jitter. Therefore, Static routing non-SDN stably runs at fully bandwidth congestion.

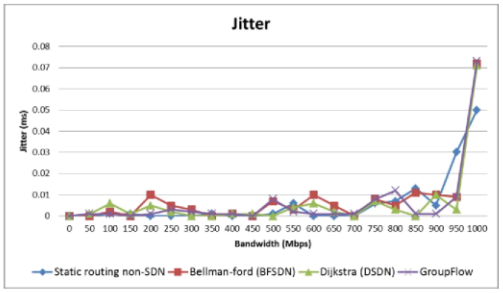


Fig. 12. The UDP jitter

Packet loss is measured to verify the performance of different bandwidth such as jitter. However, it has another factor, called the quality of the network device installation, that affected to the packet loss rate as shown in Fig. 13. The simulation results illustrate that the order of packet loss rate are DSDN unicast, BFSDN unicast, GroupFlow and Static routing non-SDN respectively. Static routing non-SDN slightly swings and has the lowest average packet loss compared with other routing algorithms. BFSDN unicast, DSDN unicast and GroupFlow have the packet loss rate lightly different. They begin to swing a value at 450 mbps and increasing continuously. At high bandwidth usage, we can see different packet loss rate between SDN and Static routing non-SDN obviously since transmission of SDN requires controlled messages by connected switches (a sender side) to a centralize controller while Static routing

non-SDN has a control plane inside a device. Therefore, Static routing non-SDN has an advantage on packet loss for fully bandwidth usage because SDN has more network installation steps due to the separating of the control plane and data plane. However, SDN packet loss percentage is fallen in medium category of phone quality standard [18] which is still adequate.

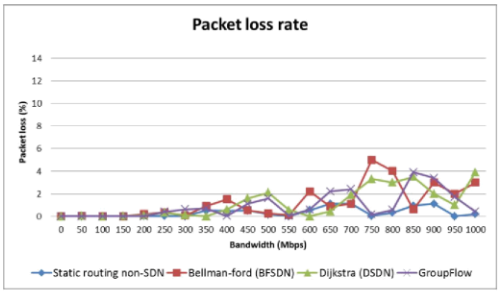


Fig. 13. Packet loss rate

IV. Conclussion and Future Work

The Software Defined Network is an efficiency programmable network which has more advantages than traditional network such as SDN handles data from a centralize controller, SDN allows reducing deployment configuration, SDN reduces the amount of spending required on infrastructure etc. Thus, SDN is an interesting network system that determines worthiness of migration traditional network.

In this paper, we use POX to implement BFSDN unicast, DSDN unicast, and using GroupFlow multicast model to compare performance analysis with Static routing non-SDN in terms of throughput, latency, jitter, and packet loss under the PSU network topology with the Mininet emulation tool. Not only SDN benefits previously stated above but also the experiment results show that DSDN unicast and BFSDN unicast have more outperformed than Static routing non-SDN in term of TCP throughput. DSDN and BFSDN unicast have latency lower than Static routing non-SDN when transmission time is over than 990 seconds and 1619 seconds respectively.

Furthermore, we found that the quality decreases at high bandwidth usage since packets are sent to a receiver while controlled messages are required sending to a centralized controller. Thus, SDN has more bandwidth congestion than Static routing non-SDN at same bandwidth requirement. However, SDN offers suitable quality of service requirements such as chat, video streaming, video on demand, video over IP, and video conferencing. The result of this research shows that replacing a traditional network by SDN has benefits. Moreover, we are able to develop various applications which can be used to manage and control network effectively. Eventually, the contribution of this paper is to examine the benefits of migrating from traditional network to SDN-aware network.

For the future work, we plan to conduct simulation experiments for video streaming unicast and multicast transmission over real topologies and build horizontal scaling devices. We are going to simulate edge weights which may be changed during the experimental run and measure overhead of OpenFlow messages at a central controller.

Acknowledgment

This research has been supported by the Engineering Post-Graduate Scholarship, Faculty of Engineering, Prince of Songkla University, Thailand.

References

[1] B. A. Nunes, M. Mendonca, X.-N. Nguyen, K. Obraczka, and T. Turletti, "A survey of software-defined networking: Past, present, and future of programmable networks," *Commun. Surv. Tutor. IEEE*, vol. 16, no. 3, pp. 1617–1634, 2014.

[2] A. Doria, J. H. Salim, R. Haas, H. Khosravi, W. Wang, L. Dong, R. Gopal, and J. Halpern, "Forwarding and control element separation (ForCES) protocol specification," 2010.

[3] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: enabling innovation in campus networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, 2008.

[4] D. Kreutz, F. M. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proc. IEEE*, vol. 103, no. 1, pp. 14–76, 2015.

[5] "noxrepo (NOX Repo)," *GitHub*. [Online]. Available: https://github.com/noxrepo. [Accessed: 10-Jul-2016].

[6] "osrg/ryu," *GitHub*. [Online]. Available: https://github.com/osrg/ryu. [Accessed: 10-Jul-2016].

[7] "floodlight/floodlight," *GitHub*. [Online]. Available: https://github.com/floodlight/floodlight. [Accessed: 10-Jul-2016].

[8] S. Sezer, S. Scott-Hayward, P. K. Chouhan, B. Fraser, D. Lake, J. Finnegan, N. Viljoen, M. Miller, and N. Rao, "Are we ready for SDN? Implementation challenges for software-defined networks," *IEEE Commun. Mag.*, vol. 51, no. 7, pp. 36–43, 2013.

[9] S. Upadhaya and G. Devi, "Characterization of QoS based routing algorithms," *Int. J. Comput. Sci. Emerg. Technol.*, vol. 133, 2010.

[10] J.-R. Jiang, H.-W. Huang, J.-H. Liao, and S.-Y. Chen, "Extending Dijkstra's shortest path algorithm for software defined networking," in *Network Operations and Management Symposium (APNOMS), 2014 16th Asia-Pacific*, 2014, pp. 1–4.

[11] "alexcraig/GroupFlow," *GitHub*. [Online]. Available: https://github.com/alexcraig/GroupFlow. [Accessed: 10-Jul-2016].

[12] "caioviel/CastFlow," *GitHub*. [Online]. Available: https://github.com/caioviel/CastFlow. [Accessed: 10-Jul-2016].

[13] M. Team, *Mininet: An instant virtual network on your laptop (or other PC)*. 2012.

[14] "BRITE: Boston university Representative Internet Topology gEnerator." [Online]. Available: http://www.cs.bu.edu/brite/. [Accessed: 10-Jul-2016].

[15] "PSU Network Diagram." [Online]. Available: http://netserv.cc.psu.ac.th/images/phocadownload/blackbone/psunet.png. [Accessed: 10-Jul-2016].

[16] "noxrepo/pox," *GitHub*. [Online]. Available: https://github.com/noxrepo/pox. [Accessed: 10-Jul-2016].

[17] "iPerf - The TCP, UDP and SCTP network bandwidth measurement tool." [Online]. Available: https://iperf.fr/. [Accessed: 10-Jul-2016].

[18] W. Sugeng, J. E. Istiyanto, K. Mustofa, and A. Ashari, "The Impact of QoS Changes towards Network Performance," *Int. J. Comput. Netw. Commun. Secur.*, vol. 3, no. 2, pp. 48–53, 2015.

QLB: QoS Routing Algorithm for Software-Defined Networking

Piyawit Tantisarkhornkhet, Warodom Werapun
Department of Computer Engineering,
Faculty of Engineering, Prince of Songkla University
Phuket, Thailand
Email: {5810120060@email.psu.ac.th, warodom@coe.phuket.psu.ac.th}

***Abstract*—Software-Defined Networking (SDN) is a new efficiently idea of programmable networks that separates the control plane from data plane of all network devices. Internet service provider is responsible for all the control decisions and communication among the forwarding elements from centralized controller. SDN provides the various optimized services. Quality of service (QoS) routing is a path computation method that is suitable for the different traffics generated by several applications, while utilization of network resources has increased. This agreement of service is defined by QoS requirements such as throughput, delay, jitter and packet loss etc. Multimedia applications often require assured from multi QoS constrained, causing the NP-complete problem which cannot be simply solved in polynomial time and high management complexity in the transition network. SDN is able to reduce complexity and it is used to efficiently implement traffic, hence SDN significantly values to development QoS routing. In this paper, we propose QoS routing algorithm called Quantized Level Balance (QLB) for SDN that considers one or many QoS parameters relating to the network application. To satisfy the requirements, QLB selects QoS parameters depending to the level of appropriate application service quality. We have replicated our algorithm on simulate topology with Scalable Video-streaming Evaluation Framework (SVEF). We measure the Peak Signal-to-Noise Ratio (PSNR) and Mean Opinion Score (MOS) of Scalable Video Coding (SVC) at the receiver. Our propose algorithm is improved than single-metric approach that may choose poor QoS parameter paths.**

Keywords—Software-defined networking; multimedia; routing algorith; Dijkstra;

I. INTRODUCTION

Presently, there is a new efficiently network management called Software-defined networking (SDN) [1] which manages network by programmable concept. It offers important advantages over than traditional network by decoupling the control plane from data plane of all SDN network devices [2] to bring more flexible and manageable. Internet service provider is responsible for all the control decisions and communication among the forwarding elements from centralized controller. Consequently, it can apply SDN to lead control software relaying on network resources and routing.

Quality of Service (QoS) allows various applications or services operating as expected [3]. There are several problems of network performance such as the problems of throughput, delay, jitter and packet loss, which are called QoS parameter, in common network. The QoS constraints have three basic composition rules for complete path with respect to each QoS parameter such as Additive Metric (delay, hop count, cost, and jitter), Multiplicative metric (reliability and loss) and Concave metric (bandwidth) [4]. Each QoS parameter uses different rules. The evolution of QoS is more complexity in routing as the QoS requirements specified by the various multimedia applications service. Moreover, QoS routing is suitable for multimedia applications such as voice over IP, video streaming and video conferencing etc. They are applications that require Multiple Constraint Path (MCP) [5]. On the other hand, some traditional routing protocols work on the network with a single mixed metric approach [6],[7] which is inadequate for multimedia applications. Additionally, they are increasingly used on the network.

In this paper, we propose QoS routing algorithm that is considered as one or many QoS parameters depending on the network application. To satisfy the requirements, our proposed algorithm takes QoS parameters by examining the level of appropriate quality for application services. We simulate our algorithm on simulate topology with SVEF which is an open-source SVC framework. We measure the PSNR and MOS of SVC at the receiver. It offers a better performance than only combination of equation that may choose some poor paths with unsatisfied QoS parameter.

The remainder of this paper is structured as follows. Literature review is discussed in Section II. Section III explains Quality of Service and SVC. Techniques of work are presented in Section IV. Finally, we summarize this work in Section V.

II. LITERATURE REVIEW

This section shows some different routing algorithms for NP-complete solutions such as Heuristic, Approximation and Randomization [8] as follows:

A. Heuristics algorithms

Heuristics algorithms were proposed to solve a NP-complete problem by reducing the complexity of path computation. The algorithm is fast whereas, it is not efficient to implement a best solution with acceptable probability. Heuristics algorithms can be classified into three types as follows:

978-1-5090-0629-8/16/$31.00 ©2016 IEEE

1) Linear composition

Linear composition is an algorithm that combines additive metrics. Y. Cui [9] and J. M. Jaffe [10] proposed converting any two additive weights to a single metric. Z. Wang [6], [7] proposed a single mixed metric approach for bandwidth, delay or loss. Although it can be an indicator in path selection, it is not sufficient for reliable QoS routing.

2) Lagranges relaxation Linear compositions

Lagranges relaxation Linear compositions is an algorithm that calculates lower bound and finds good solutions. It combined the two weights in terms of α to linear equation forms as an aggregate weight $w=w_1+ \alpha w_2$. In SDN, the linear equations form was used in [11]. They had low time complexity. A. Juttner [12] proposed an aggregated concept that considers cost and delay. It was used with SDN for service Scalable Video Coding (SVC) transmission in [13].

3) Non-Linear

Non-Linear is an algorithm that combined multi-constrained to a single weight by using non-linear formation. It was suitable for the metrics that are not correlated. T. Korkmaz [14] proposed H_MCOP algorithm that is the current best Non-Linear Heuristics algorithm. It uses Dijkstra algorithm two times including reverse direction by a linear function and forward direction by a non-linear function. This algorithm has also been proposed in SDN [15].

B. Approximation algorithms

Approximation algorithms were those heuristic that also implement some error bounds. They are efficient in arbitrary specified precision. However, they have high time complexity. G. Xue [16] proposed an Approximation algorithm that approximates all k-constraints without enforcing any one constraint.

C. Randomization algorithms

Randomization algorithms have randomness concept that are used to avert sudden problems when routing for a feasible path. Although they can be executed for inaccurate or dynamic networks, they are unfriendly with small probability. T. Korkmaz [13] implemented Randomized Breadth First Search (BFS) invents nodes from a chance node to a final destination node.

III. QUALITY OF SERVICE

QoS is needed to deliver uninterrupted multimedia services. There are many researches on how to satisfy the QoS requirements [17] such as throughput, delay, jitter, and packet loss etc.

A. QoS Requirements

The multimedia applications offer many services [18] such as video streaming, video on demand, voice over IP and video streaming etc. Each multimedia application uses a mechanism to provide differentiated QoS shows in TABLE I.

Voice over IP application uses low bandwidth. However, it is a real-time service required low latency, jitter and loss. In a real-time video application such as video conference, they should be effective in all QoS constraints as well as being high bandwidth, low latency, low jitter and low loss. Therefore, this application strongly needs suitable QoS parameters regarding to its service quality.

TABLE I. QOS REQUIREMENTS FOR TYPICAL APPLICATIONS

Application Type	Throughput Demand	Latency Tolerance	Jitter Tolerance	Packet Loss Tolerance
Email	Low	High	High	High
Web browsing	Low	High	High	High
File transfer (FTP)	Low-High	High	High	High
Chart (IM)	Low	Medium	Medium	Medium
Video streaming	Medium-High	Medium	Medium	Medium
Video on demand	High	Medium	Medium	Low
Voice over IP / WiFi	Low	Low	Low	Low
Video conferencing	Medium-High	Low	Low	Low

B. Scalable Video coding

Scalable Video Coding (SVC) is a video compression standard that is extension of the Advanced Video Coding (AVC). It translates data stream and conversely translates video into a bit stream. The SVC offers many benefits such as more flexibility, leading to higher storage and reduced redundancy. SVC was used in diverse video applications such as video on demand, video conferencing and video streaming etc.

Currently, there are many SVC frameworks such as SVEF [19], myEvalSVC-Mininet [20] and DASH-SVC-Toolchain [21]etc. In this paper, we use SVEF for video streaming application since SVEF has been used in several frameworks. Therefore, we measure average PSNR and MOS from 4CIF YUV video at 30 fps that was encoded in SVC format with cross-layer scheduling by Joint Scalable Video Model (JSVM) in four constraints such as bandwidth, delay, jitter and loss. Since delay and jitter have low effect on PSNR, we show only MOS that use for Quantized function as follows:

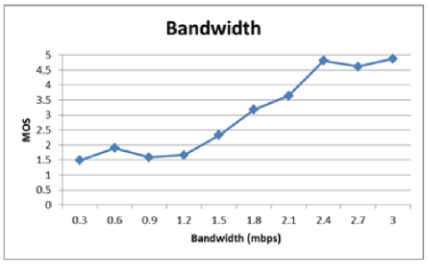


Fig. 1. MOS of received SVC stream from bandwidth constraint

From Fig. 1, in too low bandwidth, traffic will become congested and has high jitter and loss. Thus, users feel annoyed when they watch the video stream at low bandwidth. In Fig. 2 shows loss rate is important effect on PSNR and MOS down. It makes distortion in video frame.

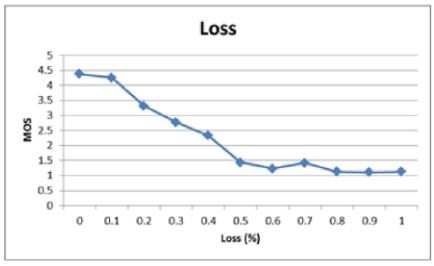


Fig. 2. MOS of received SVC stream from loss constraint

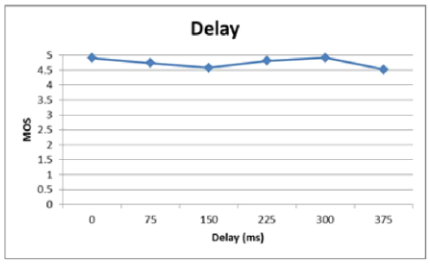


Fig. 3. MOS of received SVC stream from delay constraint

Delay has low effect on PSNR and MOS as show in Fig. 3. We set the delay at 100 ms (fair rating) because there is no difference between the videos with difference fixed delay [22] as show in Fig. 4.

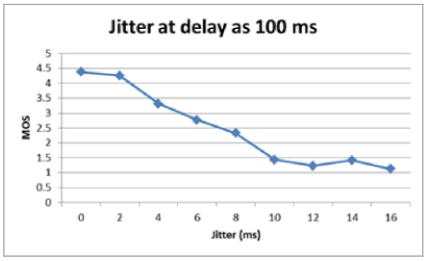


Fig. 4. MOS of received SVC stream from jitter constraint

These graphs are examining by calculating the average MOS values for each constraint. The user's perceived quality of watching a video can be analyzed. We use the MOS for divide scale which the MOS scale is used to rate the quality of videos from 1 to 5 but this work uses 3 rates. Thus, we define lower 3 in MOS is 3 in QLB, between 3 and 4 in MOS is 2 in QLB and upper 4 in MOS is 1 of QLB.

In Table II, we provide some summarizing results of constrained level such as bandwidth, loss and jitter. Constraint Level (Table II) is used in *Quantized_to_3levels()*. However, all constrained varies with the types of codec used and video motion.

TABLE II. CATEGORY REFERENCES OF QUALITY PARAMETER

QoS Constraints	Levels		
	1 Good (MOS > 4)	2 Medium (4 > MOS > 3)	3 Poor (MOS < 3)
Bandwidth level	> 2.2	1.6-2.2	< 1.6
Loss level	< 0.2	0.2-0.3	> 0.3
Jitter level	< 4	4-6	> 6

IV. TECHNIQUES

This section shows the single mixed metric [6], [7] problem and proposes our algorithm called QLB that improved the single mixed metric. The mixed QoS parameter method is a tempting Heuristic. It consists of bandwidth, delay and loss rate as expressed in Equation (1).

$$f(p) = \frac{bandwidth(p)}{delay(p)*loss(p)} \quad (1)$$

In order to compare the single mixed metric to our approach, the equation (1) must be inverted into weight from (2). This equation will be applied in our simulation.

$$Weight(p) = \frac{\text{delay(p)*loss(p)}}{\text{bandwidth(p)}} \quad (2)$$

From the above equation (1), we simply define 2 mbps bandwidth with delay and packet loss following in TABLE III, and then an example of single mixed metric result can be calculated as follows:

For example: $f(p) = \frac{2000}{100\times25} = \frac{2000}{200\times12.5} = \frac{2000}{500\times5} = 0.8$

TABLE III. QOS PARAMETER VALUE EXAMPLES

No.	Delay	Packet Loss
1	Good (100)	Poor (25)
2	Medium (200)	Good (12.5)
3	Poor (500)	Good (5)

The single mixed metric combines QoS parameters. Although different QoS parameters are used, the single mixed metric may produce the same result. Nevertheless, a good routing algorithm must select a path that has balance-QoS parameter values as second row of TABLE III for multimedia applications which should not have any poor QoS parameter.

A. The algorithm description

Our proposed algorithm takes QoS parameters suitable for application services. Selection of QoS parameters depends on TABLE I and level for QoS parameters regarding on TABLE II e.g., video streaming should analyze four parameters such as bandwidth, delay, jitter and loss since TABLE I illustrate that video streaming is able to work on medium throughput, delay, jitter and loss. However, from constraint measure results in Fig. 1-4 show that delay does not has effect on the video quality too much. Thus, we use throughput, jitter and loss for this work.

Steps:

1. Use *Dijkstra's algorithm* for QLB
2. Quantized all QoS parameters to a desired level and use the mixed metric weight.
3. In this step, there are two choices for classify each QoS parameters to level QoS parameters as follows:
    a. Total Weight
    b. Balance Weight

B. Dijkstra's algorithm for QLB

Dijkstra's algorithm is used to search the optimum shortest paths between nodes in a directed graph G (V, E), denote: set of nodes (V) and set of edges (E). We modify Dijkstra's algorithm to consider three constraints and consider weights as follows: ew[] is node, ew[][] are raw QoS parameters, ew[][][] are quantized QoS parameters, level weight (ew[][][P]) and the mixed single metric weight (ew[][][P+1]). Denote: Number of Level of category (L) and Number of QoS parameter (P).

Function of Dijkstra's algorithm consists of two functions as follows:

Modify_Dijkstra_Algorithm function

Input: G, s
Output: dis[V], pre[V]

```
1:  for each v in V
2:      dis[v] ← ∞
3:      pre[v] ← null
4:  Q ← set(node.keys)
5:  while (Q!=null)
6:      u← Minimum_distance(d,Q)
7:      for each v adjacent to u do
8:          adj[v][][0] ← Min (dis[u][0], ew[u,v][][0])
9:          adj[v][][1] ← dis[u][1]+ew[u,v][][1]
10:         adj[v][][2] ← dis[u][2]+ew[u,v][][2]
11:         adj[] ← Quantized_to_3levels (adj[])
12:         adj[] ← Total_Weight (adj[]) or
                                Balance_Weight(adj[])
13:     if dis[v][0] = adj[v][][P] then
14:         if dis[v][1] > adj[v][][P+1] then
15:             dis[v][0]← adj[v][][P]
16:             dis[v][1]← adj[v][][P+1]
17:             pre[v][0]←dis[v][0]
18:             pre[v][1]←dis[v][1]
19:     else if dis[v][0] > adj[v][][P] then
20:         dis[v][0]← adj[v][][P]
21:         dis[v][1]← adj[v][][P+1]
22:         pre[v][0]←dis[v][0]
23:         pre[v][1]←dis[v][1]
```

Fig. 5. Modify_Dijkstra's Algorithm for QLB

Our *Modify_Dijkstra_Algorithm()* (Fig. 5) is enhanced from an original algorithm that considers QoS parameters while collect weight from source to node v in adj[v][][0] (bandwidth), adj[v][][1] (loss) and adj[v][][2] (jitter). It uses *Quantized_to_3levels()* for calculating a mixed single metric (adj[][][P+1]) and quantizing each QoS parameter and uses *Total_Weight ()* or *Balance_Weight ()* for calculating weight level (adj[][][P]). Additional consideration is to focus on the level weight (adj[][][P]) and a mixed single metric (adj[][][P+1]), respectively. Its time complexity is O(E) as same as an original Dijkstra's algorithm.

Modify_Minimum_distance function

Input: distance, Q
Output: node, min

```
1:  min_level ← ∞ ;
2:  min_mix ← ∞ ;
3:  node ← 0
4:  for v in Q do
5:      adj_node[] ← Quantized_to_3levels (distance[])
6:      adj_node[] ← Total_Weight (distance []) or
                            Balance_Weight(distance [])
7:      if adj_node [v][][P] = min_level then
8:          if adj_node [v][][P+1] < min_mix then
9:              min_level ← adj_node [v][][P]
10:             min_mix ← adj_node [v][][P+1]
11:             node ← v
12:     else if adj_node [v][][P] < min[0] then
13:         min_level ← adj_node [v][][P]
14:         min_mix ← adj_node [v][][P+1]
15:         node ← v
```

Fig. 6. Modify_Minimum_distance function

Modify_Minimum_distance() is a component of Dijkstra's algorithm used to retrieve a node that has the first minimum level weight [line11]. If the minimum level weight is equal to the weight being compared [line 6], the next weight has to be checked [line 7].

C. The Quantized of QoS parameters

This section presents the conversion of the QoS parameter to a desired level by the quality of threshold that is defined in TABLE II., Definition: Good=1, Medium=2 and Poor=3. *Quantized_to_3levels()* (Fig. 7) is used to calculate a mixed metric equation and divide three threshold levels.

In calculate mixed metric equation, we adapt (2) that consist of bandwidth, delay and loss without jitter. Jitter is added into (3) in order to calculate mixed metric.

$$Weight_{\,mixed\ metric} = \frac{\text{delay} * \text{jitter} * \text{loss}}{bandwidth} \tag{3}$$

From (3), we calculate each QoS parameter for weight of path by different rules. Delay and jitter are additive metric and loss is multiplicative metric. However, loss constraint of path in Mininet simulator, calculate from additive metric during links (4).

$$Weight_{\,mixed\ metric\ path} = \frac{\sum \text{delay} * \sum \text{jitter} * \sum loss}{bandwidth} \tag{4}$$

In addition, it is serviced for QoS parameters which are stored at edge weights (ew[][]). It begins from Bandwidth [line 2-4], Jitter [line 5-7] and Loss. [line 8-10]. Thresholds in Fig. 7, they are defined from TABLE II which we can take them to a multimedia service appropriately.

Denote: ew[][0] is edge weight of delay level (unused in this work), ew[][1] is edge weight of bandwidth level, ew[][2] is edge weight of loss level and ew[][3] is edge weight of jitter level.

Quantized_to_3levels function
Input: ew
Output: ew

```
1:  ew[][][P+1]=mixed metric equation (4)
2:  if ew[][1] > 2.2 then ew[][][1] ← 1
3:  else if ew[][1] ≥ 1.6 then ew[][][1] ← 2
4:  else if ew[][1] < 1.6 then ew[][][1] ← 3
5:  if ew[][2] < 0.2 then ew[][][2] ← 1
6:  else if ew[][2] ≤ 0.3 then ew[][][2] ← 2
7:  else if ew[][2] > 0.3 then ew[][][2] ← 3
8:  if ew[][3] < 4 then ew[][][3] ← 1
9:  else if ew[][3] ≤ 6 then ew[][][3] ← 2
10: else if ew[][3] > 6 then ew[][][3] ← 3
```

Fig. 7. Quantized_to_3levels function

D. Total Weight

This method is to sum the levels of each QoS parameters. It requires time complexity lower than Balance Weight.

Fig.8 shows *Total_Weight()* function. It is used for grouped of addition level of each QoS parameter (ew[][][P+1]) [line 1] and subtract the summation of edge weight from (L-1) for weight level [line 2].

Total_Weight function
Input: ew
Output: ew

```
1: ew[][][P+1] ← sum(each ew[0])
2: ew[][][P+1] ← (ew[][][P+1] – (L-1)) / MG
```

Fig. 8. Total_Weight function

Total Weight has a maximum number of groups (MG) that depends on Number of Level of category (L) and Number of QoS Parameter (P). We can predict a maximum number of groups from equation (5), (6).

$$MG_{Total\ Weight} = 1, \quad where\ L=1 \tag{5}$$

$$MG_{Total\ Weight} = L + [(P-1)*2^{(L-2)}],\ where\ L > 1 \tag{6}$$

E. Balance Weight

This method classifies by examining each QoS category level and sorts desired levels from low to high. It is used to avoid some poor QoS parameters.

Fig. 9 presents *Check_level_parameter()* that supports *Balance_Weight()* for counting the numbers in each level of QoS parameters. Its time complexity is O(P*L).

Check_level_parameter function
Input: ew, P, L
Output: count

```
1:  for p=1 to P do
2:      for l=1 to L do
3:          if ew[l] = p
4:              count[p]+1
```

Fig. 9. Check_level_parameter function

Balance_Weight function
Input: ew
Output: ew

```
1:  count[] ← Check_level_parameter(ew, P, L)
2:  if count[1] = L
3:      ew[][][P+1] = 1/MG
4:  if count[2] > 0 and count[3] = 0
5:      for n=1 to P do
6:          if count[1] = P-n and count[2] = n
7:              ew[][][P+1] = n+1/MG
8:  if count[3] > 0
9:  for n=1 to P do
10:         if count[3] = n
11:             for m= 0 to P-1
12:                 if count[1] = P-n-m and count[2] = m
13:                 ew[][][P+1] = P+m+n+1/MG
```

Fig. 10. Balance_Weight

Fig. 10 illustrates *Balance_Weight()* that uses *Check_level_parameter()* for counting number of each QoS parameter level in link. This function supports the maximum level is 3. If QoS parameter level has only 1, it is found in [line 2-3]. If QoS parameter level has 1 and 2, it is found in [line 4-7]. If QoS parameter level has 1, 2 and 3, it is found in [line 9-13]. The time complexity *Balance_Weight()* function is depended on maximum metric level.

We can predict each maximum number of groups from equations (7) - (10).

$$MG_{Balance} = 1, \quad where\ L=1 \tag{7}$$

$$MG_{Balance} = 1+P, \quad where\ L=2 \tag{8}$$

$$MG_{Balance} = 1 + P + \sum_{k=1}^{P} k,\ where\ L=3 \tag{9}$$

$$MG_{Balance} = 1 + P + \sum_{k=1}^{P} k + MG_{Balance\ (P-1)},\ where\ L=4 \tag{10}$$

F. Simulate

In our proposed algorithm, we simulate on the topology and information regarding to Bandwidth (B), jitter (J) and loss (L) by using three threshold levels in Mininet. We measure PSNR from 4CIF YUV video at 30 fps that is encoded in SVC format by Joint Scalable Video Model (JSVM).

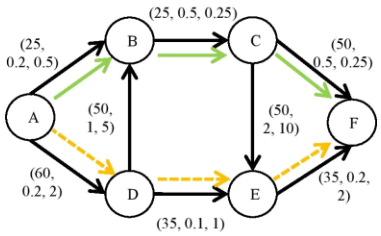


Fig. 11. Simple topology

Fig. 11 shows simple topology with three constrained such as bandwidth (mbps), loss rate (percent) and jitter (ms) at delay as 100 ms and results from calculating links and paths by Total Weight and Balance Weight, respectively. We show different cases for path choosing between Total Weight and Balance Weight. Total Weight function selects the top path (A-B-C-F) since total level weight is lowest (PSNR is 17 dB and MOS is 1.7). Balance Weight function takes the bottom path (A-D-E-F) which is the lowest balance level weight (PSNR is 20 dB and MOS is 2.3). The comparison results between total weight and balance weight reveal that balance weight has about 3 dB PSNR and 0.6 MOS greater than total weight.

V. Conclusion

Multimedia application has been continuously increased in the world network. QoS routing is popular requested for multimedia application. Furthermore, the most QoS transmission have the NP-complete problem since they consider multi-constrained. SDN is another network management concept which is developed continuously for the reason that it has advantages more than traditional network. Therefore, it is suitable for various services, particularly multimedia services.

In this paper, we propose an algorithm called QLB with SDN-aware. It is used to consider MCP which uses quantized and classifies into groups. We arrange into two types: 1) Total Weight and 2) Balance Weight. Total Weight is summarized QoS category levels, whereas Balance Weight is sorted each QoS category level in ascending order. Although time complexity of Balance Weight is more than Total Weight, Balance Weight is suitable for generally multimedia applications since the majority of multimedia services required each suitable QoS category with well path selection. Future work, we will quantize five levels from Mean Opinion Score (MOS). Moreover, we are going to simulate various videos with frame sizes on different large topologies that have dynamic update of QoS parameters

Acknowledgment

This research has been supported by the Engineering Post-Graduate Scholarship, Faculty of Engineering, Prince of Songkla University, Thailand.

References

[1] H. Farhady, H. Lee, and A. Nakao, "Software-defined networking: A survey," *Comput. Netw.*, vol. 81, pp. 79–95, 2015.

[2] B. A. Nunes, M. Mendonca, X.-N. Nguyen, K. Obraczka, and T. Turletti, "A survey of software-defined networking: Past, present, and future of programmable networks," *Commun. Surv. Tutor. IEEE*, vol. 16, no. 3, pp. 1617–1634, 2014.

[3] M. Curado and E. Monteiro, "A survey of QoS routing algorithms," in *Proceedings of the International Conference on Information Technology (ICIT 2004), Istanbul, Turkey*, 2004.

[4] P. Karkazis, P. Trakadas, H. C. Leligou, L. Sarakis, I. Papaefstathiou, and T. Zahariadis, "Evaluating routing metric composition approaches for QoS differentiation in low power and lossy networks," *Wirel. Netw.*, vol. 19, no. 6, pp. 1269–1284, 2013.

[5] P. Van Mieghem, F. A. Kuipers, T. Korkmaz, M. Krunz, M. Curado, E. Monteiro, X. Masip-Bruin, J. Solé-Pareta, and S. Sánchez-López, "Quality of service routing," in *Quality of Future Internet Services*, 2003, pp. 80–117.

[6] Z. Wang and J. Crowcroft, "Quality-of-service routing for supporting multimedia applications," *Sel. Areas Commun. IEEE J. On*, vol. 14, no. 7, pp. 1228–1234, 1996.

[7] Z. Wang and J. Crowcroft, "Bandwidth-delay based routing algorithms," in *Global Telecommunications Conference, 1995. GLOBECOM'95., IEEE*, 1995, vol. 3, pp. 2129–2133.

[8] S. Upadhaya and G. Devi, "Characterization of QoS based routing algorithms," *Int. J. Comput. Sci. Emerg. Technol.*, vol. 133, 2010.

[9] Y. Cui, K. Xu, and J. Wu, "Precomputation for multiconstrained QoS routing in high-speed networks," in *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, 2003, vol. 2, pp. 1414–1424.

[10] J. M. Jaffe, "Algorithms for finding paths with multiple constraints," *Networks*, vol. 14, no. 1, pp. 95–116, 1984.

[11] S. Laga, T. Van Cleemput, F. Van Raemdonck, F. Vanhoutte, N. Bouten, M. Claeys, and F. De Turck, "Optimizing scalable video delivery through OpenFlow layer-based routing," in *Network Operations and Management Symposium (NOMS), 2014 IEEE*, 2014, pp. 1–4.

[12] A. Jüttner, B. Szviatovski, I. Mécs, and Z. Rajkó, "Lagrange relaxation based method for the QoS routing problem," in *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, 2001, vol. 2, pp. 859–868.

[13] T. Korkmaz and M. Krunz, "Multi-constrained optimal path selection," in *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, 2001, vol. 2, pp. 834–843.

[14] Y. Tsung-Feng, K. Wang, and Y.-H. Hsu, "Adaptive routing for video streaming with QoS support over SDN networks," in *Information Networking (ICOIN), 2015 International Conference on*, 2015, pp. 318–323.

[15] L. Sheng, Z. Song, and J. Yang, "A multi-constrained routing algorithm for software defined network based on nonlinear annealing," *J. Netw.*, vol. 10, no. 6, pp. 376–384, 2015.

[16] G. Xue, A. Sen, W. Zhang, J. Tang, and K. Thulasiraman, "Finding a path subject to many additive QoS constraints," *IEEEACM Trans. Netw. TON*, vol. 15, no. 1, pp. 201–211, 2007.

[17] W. Sugeng, J. E. Istiyanto, K. Mustofa, and A. Ashari, "The Impact of QoS Changes towards Network Performance," *Int. J. Comput. Netw. Commun. Secur.*, vol. 3, no. 2, pp. 48–53, 2015.

[18] Y. Chen, T. Farley, and N. Ye, "QoS requirements of network applications on the Internet," *Inf. Knowl. Syst. Manag.*, vol. 4, no. 1, pp. 55–76, 2004.

[19] A. Detti, G. Bianchi, W. Kellerer, and others, "SVEF: an Open-Source Experimental Evaluation Framework," in *In Proc. of IEEE MediaWIN 2009, Sousse, Tunisia*, 2009.

[20] C.-H. Ke, "myEvalSVC: an Integrated Simulation Framework for Evaluation of H. 264/SVC Transmission.," *TIIS*, vol. 6, no. 1, pp. 379–394, 2012.

[21] C. Kreuzberger, D. Posch, and H. Hellwagner, "A scalable video coding dataset and toolchain for dynamic adaptive streaming over HTTP," in *Proceedings of the 6th ACM Multimedia Systems Conference*, 2015, pp. 213–218.

[22] V. D. Bhamidipati and S. Kilari, "Effect of delay/delay variable on QoE in video streaming," *M Thesis Sch. Comput. Blekinge Inst. Technol. Swed.*, 2010.

SVC-MST BWQLB Multicast over Software Defined Networking

Piyawit Tantisarkhornkhet, Warodom Werapun

Department of Computer Engineering, Faculty of Engineering, Prince of Songkla University Phuket, Thailand

Abstract

This paper presents the method of Scalable Video Coding (SVC) Multicast over Software Defined Networking (SDN). We choose to develop 2 parts. The first part is concerned to the transmission management, and the second part is related to sender and receiver models. In the first part, we modify the Quantized Level Balance (QLB) routing algorithm using SDN which focuses on the transmission of video streaming. We measure different video resolution thresholds and routing to transmit based on balance quality of service parameters. We examine this model compare with the OSPF routing on traditional network. The results are revealed that QLB routing can control the display quality better than OSPF routing. In the second part, we design the transmission of multiple video resolution and video receivers by modified SVC-MST (Multi-session transmission) with bandwidth threshold of QLB called SVC-MST BWQLB. The SVC-MST BWQLB can reduce the usage bandwidth for freezing solution, which is a problem caused by bandwidth congestion. We experiment our SVC-MST BWQLB with 640x360, 1280x720 and 1920x1080 video resolution respectively. The transmission of the modified SVC-MST with the Advanced Video Coding (AVC), the SVC SST (Single-session transmission) mode and the traditional SVC-MST are compared. However, SVC-MST receiver gets the video that does not match with the correct sequence, resulting in encoded error. Buffers for defragmentation are implemented. Eventually, the contribution of this paper is designing QLB routing in order to reduce both storage space and bandwidth congestion problem in multiple video resolution. The maximum delay can be reduced with 46.77% (640x360 and 1280x720) and 66.64% (640x360, 1280x720 and 1920x1080) compared to SVC-SST. Furthermore, sorted buffer has been proposed for SVC-MST defragmentation problem. Our proposed solution can be applied for designing another SDN network management that is required different quality thresholds for the variable application services.

Keywords:
Software Defined Networking, Advanced Video Coding, Scalable Video Coding, Multicast, Single-session transmission, Multi-session transmission

1. Introduction

In 2016, 73 percent of global Internet traffic is video traffic. The Cisco traffic researcher predicted future video traffic that will be up to 82 percent in 2021 Cisco (2016). Thus, multimedia management service is an interesting and important implementation now. Presently, there is an extension of Advanced Video Coding (AVC) Sullivan et al. (2004) that has more flexibility than AVC called Scalable Video Coding (SVC) Heiko Schwarz et al. (2007). The SVC consists of a base layer that is AVC video and many enhancement layers that are a part of extend video quality.

The SVC transmission can be separated into 2 modes such as single session transmission (SST) mode and multiple session transmission (MST) mode. The SVC SST mode works like traditional video transmission that transfers a single video in a single session. The SVC-MST mode is transmission model that can transfer an SVC video layer to various sessions, so it can reduce lag length or freezing problem in many resolution transmissions. However, efficacious SVC video service management is considerably complex. Thus, it requires to has an excellent network management too. Software Defined Networking (SDN) Hu et al. (2014) is an excellent network transmission management that bases on network programmable concept. The SDN administrator can manage network policy via a control plan at a central controller that is separated from a data plan. Therefore, we chose to implement the SVC video streaming service on SDN.

The traditional SVC-MST on SDN, there are both unicast and multicast. The several SVC unicast models propose to divide the SVC layer to different sessions and paths which use various traffic management techniques. Although several SVC multicast models can detach video layers to many session, the routing management calculates only a current layer and ignore previous calculated video layers. If there are links that have too low bandwidth for sending many layers, traditional SVC-MST may send videos at the congestion links.

In this paper, we propose SVC-MST multicast management model on SDN that uses bandwidth thresholds QLB concept for dividing layer sessions or paths called SVC-MST BWQLB. The SVC-MST BWQLB compute routing from a base layer to a highest enhancement layer since a base layer is the most important for video display. Each layer calculated path will be

Email addresses: 5810120060@email.psu.ac.th (Piyawit Tantisarkhornkhet), warodom@coe.phuket.psu.ac.th (Warodom Werapun)

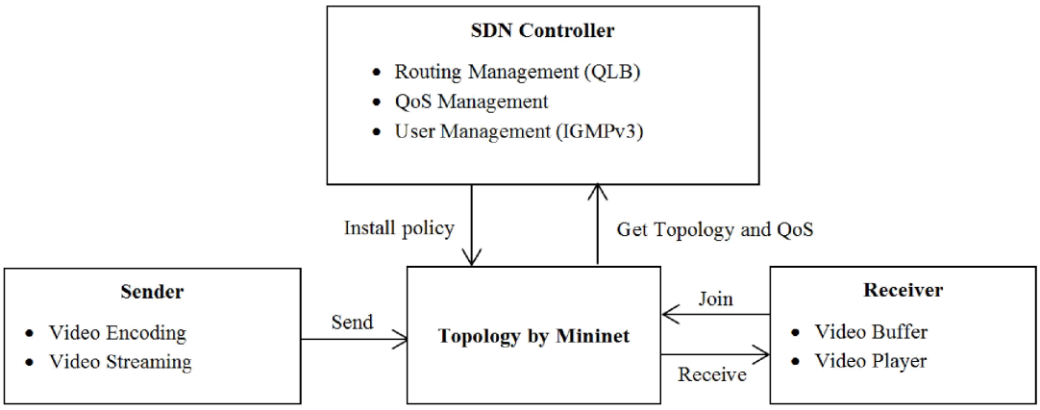


Figure 1: Design of the SVC-MST BWQLB system

recoded and passed in order to define bandwidth threshold of QLB routing. However, SVC-MST packet sequences may be wrong since each video layer has different access unit packet sizes and all video layer sessions are not related. Therefore, we implement sorted buffer in order to solve packet sequences error at video receivers.

The remainder of this paper is structured as follows. Literature review is discussed in Section II. Section III explains design of system. Modified QLB for multiple AVC videos are presented in Section IV. We propose SVC-MST BWQLB model in Section V. Finally, we summarize this work in Section VI.

2. Literature Review

There are various researchers who have worked on scalable video coding (SVC) over Software Defined networking (SDN). Two transmissions can be categorized as unicast and multicast.

In unicast transmission part, Laga et al. (2014) separated video layers by Quality of service levels which can reduce freezing problem 77.3%. Yu et al. (2015) defined priority to each video layer which could reduce freezing problem 72%. Civanlar et al. (2010) proposed a Quality-of-Service (QoS) architecture for SVC video. Egilmez et al. (2013) offered the model that manages the forwarded layer controlling by dynamic QoS. Detti et al. (2009) proposed open-source framework of SVC video streaming called Scalable Video-streaming Evaluation Framework (SVEF). Then, Ke (2012) implemented SVEF on SDN called myEvalSVC.

In multicast transmission part, Yang et al. (2014) presented a multiple sender and receiver model of SVC video transmission. Yang et al. (2015) and Xue et al. (2015) illustrated SVC video architecture for multicast. Although each SVC layer will be sent at different sessions, these SVC layers are still delivered in the same path. When a receiver has transmission problems, Jian Yang reduced some high enhancement layers for decreasing the lag length problem.

Additionally, there were interesting SVC transmissions on traditional network such as Wenger et al. (2011) solution by using several RTP payload formats and usages for SVC. Their packetization modes consist of SVC-SST and SVC-MST. SVC-MST is interesting transmission since it can be used on various ways.

3. Design of System

This session presents the architecture which consists of a sender, a SDN controller, a receiver and a topology as shown in Fig. 1.

1. Sender takes H.264 video encoding for video streaming. SVC is encoded by JSVM (Joint Scalable Video Model) and a video streaming model for stream video data by multicast transmission Kreuzberger et al. (2015).
2. Receiver is used for video rendering from a sender included video downloading. Moreover, the receiver has been proposed and implemented the video sorted buffer for SVC-MST. Mplayer media player program is selected for a receiver as a video player.
3. In SDN controller, we modified GroupFlow model Craig (2014) which consists of 3 modules as follow:
   a) User Management is used for managing members in multicast group by IGMPv3 Holbrook et al. (2006)
   b) Routing Management is used for managing routes from servers to clients. The traditional GroupFlow routing module use Dijkstra's routing. In this work, we modified QLB routing.
   c) QoS Management is used for recording multiple constraints from network traffic.
4. Topology is conducted by the network emulator program called Mininet. It can define various QoS constraints such as bandwidth, delay, jitter and loss rate etc.

4. Modified QLB for AVC video

This section offers QLB Tantisarkhornkhet and Werapun (2016) thresholds of AVC for 3 resolution such as 640x360, 1280x720 and 1920x1080. We measure 3 constraints by peer to peer on SDN network as follows:

4.1. Bandwidth

Bandwidth is the transmission limitation. If the video is transmitted at insufficient bandwidth, it will cause freezing problem during video rendering. We measured last packet delay of a Group of pictures (GOP) that consists of coded video frame types such as I-frame, P-frame and B-frame in bandwidth range from 0.1 Mbps to 10 Mbps as shown in Fig. 2. We calculate good delay of last packet that play 24 fps of 47 frames GOP about 1.96 seconds completely. Therefore, threshold table is defined from Fig. 2 and Table 1.

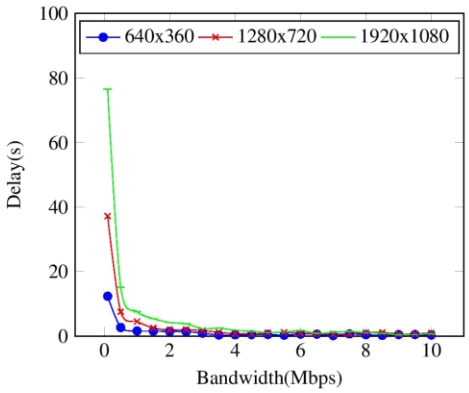


Figure 2: Delay of 3 resolution AVC

Table 1: Bandwidth threshold of 3 resolution

Resolution	Good	Medium	Poor
640x360	> 0.8 Mbps	0.6-0.8 Mbps	< 0.6 Mbps
1280x720	> 2.2 Mbps	2.0-2.2 Mbps	< 2.0 Mbps
1920x1080	> 4.8 Mbps	4.0-4.8 Mbps	< 4.0 Mbps

4.2. Jitter

Jitter is a variation delay constraint. It is a lag length problem factor in video display. Although jitter is solved by buffer, large buffer is directly affected to preparation display delay. Therefore, we use buffer size from Cisco Troubleshooting TechNotes (2005) for different video resolution such as 640x360, 1280x720 and 1920x1080. These are 0.1 MByte, 0.6 MByte and 1.5 MByte respectively. In addition, we define jitter threshold by (1) equation and use medium bandwidth threshold parameters from Table 1. Denote: B is buffer size, PS is packet size and Bw is Bandwidth.

$$B = \frac{PS}{Bw} \tag{1}$$

The jitter thresholds are calculated for (1) equation which has each resolution result such as 640x360, 1280x720 and 1920x1080. These are 8 seconds, 10.62 seconds and 7.2 seconds respectively as show Table 2. Furthermore, jitter may occur by frame type arrival time like transmission difference of I-frame, B-frame and P-frame that have different packet sizes.

Table 2: Jitter threshold of 3 resolution

Resolution	Good	Medium	Poor
640x360	< 8 s	8.0-8.3 s	> 8.5 s
1280x720	< 10.7 s	10.7-11 s	> 11 s
1920x1080	< 7.11 s	7.2-7.5 s	> 7.5 s

4.3. Loss rate

Loss is an impact of incomplete video data receiving. If there are some loss packets at the first GOP frame or I-frame, it will increase encoded problem. Loss rate threshold is measured using PSNR between source videos and receiver videos. Since loss rate is a random packet value, PSNR has weight range. Eventually, threshold is determined in Table 3 and Fig. 3.Loss rate (Fig 3) of several video resolution is overlapped as one line graph due to the same sending path.

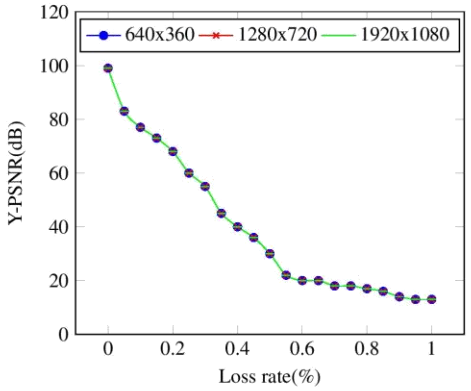


Figure 3: Loss of 3 resolution AVC

Table 3: Loss rate threshold of 3 resolution

Resolution	Good	Medium	Poor
640x360 1280x720 1920x1080	< 0.2 %	0.2-0.3 %	> 0.3 %

4.4. Possible Threshold

We offer possible threshold of 3 constraints as bandwidth, jitter and loss rate respectively. These parameters will be quantized in 3 levels as follows: (1) is good, (2) is medium and (3) is poor as shown Table 4.

Table 4: Possible threshold of 3 constraints and 3 levels

(1)(1)(1)	(1)(2)(1)	(1)(3)(1)
(1)(1)(2)	(1)(2)(2)	(1)(3)(2)
(1)(1)(3)	(1)(2)(3)	(1)(3)(3)
(2)(1)(1)	(2)(2)(1)	(2)(3)(1)
(2)(1)(2)	(2)(2)(2)	(2)(3)(2)
(2)(1)(3)	(2)(2)(3)	(2)(3)(3)
(3)(1)(1)	(3)(2)(1)	(3)(3)(1)
(3)(1)(2)	(3)(2)(2)	(3)(3)(2)
(3)(1)(3)	(3)(2)(3)	(3)(3)(3)

Possible threshold in Table 2 is measured and quantized from medium bandwidth values in Table 1. Since jitter is directly related to bandwidthTherefore, we classified new good and poor bandwidth thresholds are newly classified. The good bandwidth and good-medium jitter threshold group mean the group does not have delay time problem. On the other handother hand, the poor bandwidth and medium-poor jitter threshold group will have delay problem. Thus, we set new threshold of 3 QoS constraints and 3 levels as shown in Table 5.

Table 5: Possible threshold of 3 constraints and 3 levels (jitter from medium bandwidth)

(1)(1)(1)	(1)(1)(1)	(2)(2)(1)
(1)(1)(2)	(1)(1)(2)	(2)(2)(2)
(1)(1)(3)	(1)(1)(3)	(2)(2)(3)
(2)(1)(1)	(2)(2)(1)	(2)(3)(1)
(2)(1)(2)	(2)(2)(2)	(2)(3)(2)
(2)(1)(3)	(2)(2)(3)	(2)(3)(3)
(2)(2)(1)	(3)(3)(1)	(3)(3)(1)
(2)(2)(2)	(3)(3)(2)	(3)(3)(2)
(2)(2)(3)	(3)(3)(3)	(3)(3)(3)

From Table 5, we can group QLB cost by referencing threshold possible table, for example of the total weight, (1)(1)(1) from Table 5 has lowest summation. Thus, (1)(1)(1) is group '1', next (1)(1)(2) or (1)(2)(1) or (2)(1)(1) is group '2' etc respectively as shown in Table 6.

The balance weight cost is determined by sorting possible threshold from Table 5 to balance each constraint. For example of the balance weight, (1)(1)(1) from Table 5 has only a 'good' level. Thus, (1)(1)(1) is group '1', next (1)(1)(2) or (1)(2)(1) or (2)(1)(1) has a 'medium' constraint. They are at group '2' respectively. (1)(1)(3) or (1)(3)(1) or (3)(1)(1) is group '5' since they have a 'poor' constraint as shown in Table 7. However, (1)(1)(3) or (1)(3)(1) or (3)(1)(1) of total weight has summation as 5. Therefore, they are group '3' in total weight.

Table 6: Total weight cost

1	1	3
2	2	4
3	3	5
2	3	4
3	4	5
6	7	6
3	5	5
4	6	6
5	7	7

Table 7: Balance weight cost

1	1	3
2	2	4
5	5	7
2	3	6
3	4	7
6	7	9
3	8	8
4	9	9
7	10	10

The gray boxes are ambiguous and variant weights for using since the weights (bandwidth and jitter) are changed from Table 4 to Table 5. Therefore, we consider them to a second weight that can be chosen when they are lower weight than normal weight (white box).

4.5. Experimental

This part compares among QLB total weight, QLB balance weight and OSPF (Traditional network agent) by sending data from h1 to h2 on Mininet topology that has 7 paths as shown in Fig. 4. The 7 paths are defined by Table 4 that removes same highest and lowest weight thresholds between QLB total weight and QLB balance weight in order to clearly show their different routing. In the 7 paths, there are quality thresholds like (1)(1)(3), (1)(2)(3), (2)(3)(1), (2)(2)(2), (2)(3)(2), (2)(1)(3) and (2)(2)(3). We use 3 AVC video resolution such as 640x360, 1280x720 and 1920x1080.

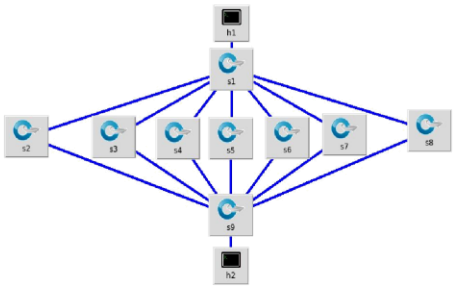


Figure 4: The 7 paths topology

1. The OSPF analyses only bandwidth, so it can be classified 2 weights as follows:
   Group 1: (1)(1)(3) and (1)(2)(3)
   Group 2: (2)(3)(1), (2)(2)(2), (2)(3)(2), (2)(1)(3) and (2)(2)(3)
   The OSPF chose 2 parts that has threshold such as (1)(1)(3) and (1)(2)(3). Although both 2 paths have good bandwidth and jitter, they have a high loss problem. When we measure loss rate that has a problem of paths, the average Y-PSNR is 24 dB.
2. The QLB total weight analyses by summation of thresholds, so it can be classified 3 weights as follows:
   Group 1: (1)(1)(3) and (1)(2)(3)
   Group 2: (2)(2)(2), (2)(3)(1) and (2)(1)(3)
   Group 3: (2)(3)(1) and (2)(2)(3)
   QLB total weight chose best paths such as (1)(1)(3) or (1)(2)(3) like OSPF which they still have loss rate problem in the path.
3. The QLB balance weight analyses thresholds by sorting group that has 1 to 3 (more information in Tantisarkhornkhet and Werapun (2016)), so it can be classified 4 weights as follows:
   Group 1: (2)(2)(2)
   Group 2: (1)(1)(3) and (1)(2)(3)
   Group 3: (2)(3)(1) and (2)(1)(3)
   Group 4: (2)(2)(3) and (2)(3)(2)
   QLB balance weight chose the best path as (2)(2)(2) which is the balance threshold path. The (2)(2)(2) path can stream video normally since it has normal bandwidth and jitter threshold range of the video. The loss threshold of the path is defined by SDN controller administrator. The average Y-PSNR is 62 dB.

5. SVC-MST BWQLB model

This section presents SVC-MST BWQLB system. The SVC-MST BWQLB is a model that applies bandwidth threshold of QLB for SVC-MST. Many SVC layers are delivered to several paths when the line has too low bandwidth from sending multiple SVC layers. Additionally, bandwidth of previous SVC layers is examined when the shortest path is calculated for higher SVC layers in order to analyse traffic properly. SVC-MST BWQLB architecture is illustrated as follows.

5.1. Sender

1. Storage space
   We compare AVC and SVC storage space of 5 GOPs that has the same encoded format from Kreuzberger et al. (2015). 3 video resolution such as 640x360, 1280x720 and 1920x1080 are measured and divided into 2 cases such as the single resolution storage space case and the multiple resolution storage space case.
   a) Single resolution storage space case
      This case compares 3 resolution such as 640x360, 1280x720 and 1920x1080. In Fig. 5 shows 640x360, AVC and SVC that have the same video size since the base layer of SVC is equal to the lowest quality AVC. In Fig. 6 shows 1280x720, there are different video sizes in all segments between AVC and SVC. There are more different video sizes of 1920x1080 as shown in Fig. 7 SVC frame has overhead for dividing video quality layer. Therefore, 5 GOP segments of SVC size larger than AVC size such as 640x360, 1280x720 and 1920x1080 are 0%, 8.93% and 28.72% respectively.

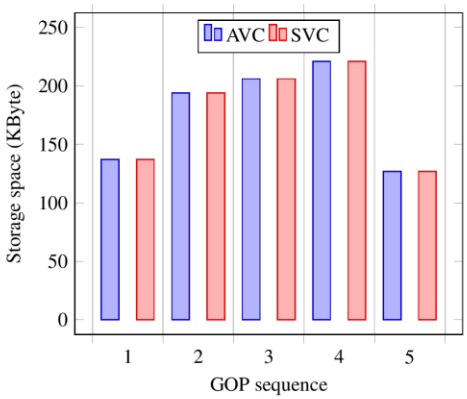


Figure 5: Comparison of video group size between AVC and SVC 640x360

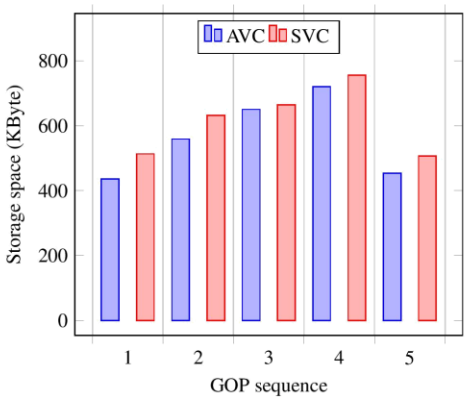


Figure 6: Comparison of video group size between AVC and SVC 1280x720

   b) Multiple resolution storage space case
      This case compares 2 resolution groups such as (640x360 and 1280x720) and (640x360, 1280x720 and 1920x1080). Fig. 8 (640x360 and 1280x720) and Fig. 9 (640x360, 1280x720 and 1920x1080) show that SVC has

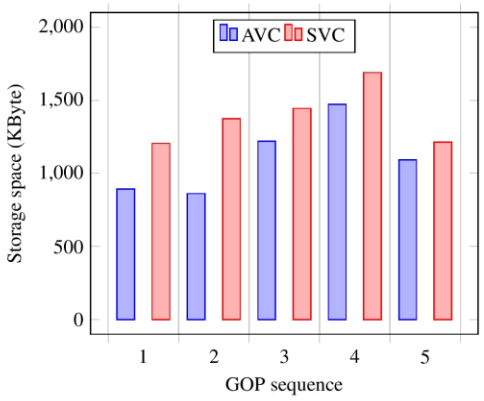


Figure 7: Comparison of video group size between AVC and SVC 1920x1080

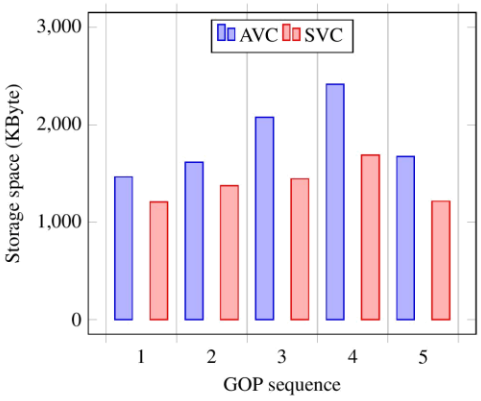


Figure 9: Comparison of video group size between AVC and SVC (640x360, 1280x720 and 1920x1080)

storage space lower than AVC since AVC storage space collects full videos of all resolution which cause duplicated video data. However, SVC avoids storing duplicated video data by collecting only highest quality video with some small metadata of each video layer. Therefore, SVC can save disk spaces more than AVC in multiple resolution storage space case. Additional, we compare 5 GOP segments percent of (640x360 and 1280x720) and (640x360, 1280x720 and 1920x1080). The storage space results are reduced 22.14% and 42.22% respectively.
AVC takes storage space slightly less than SVC in the single resolution storage space case. AVC is suitable for storing a video in a device that requests to the specific video quality. On the other hand, SVC is greater than AVC for storing a video with various video qualities.

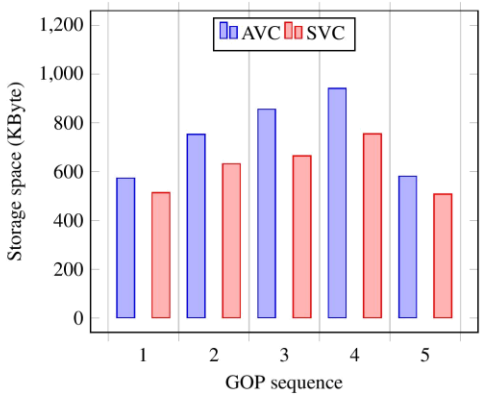
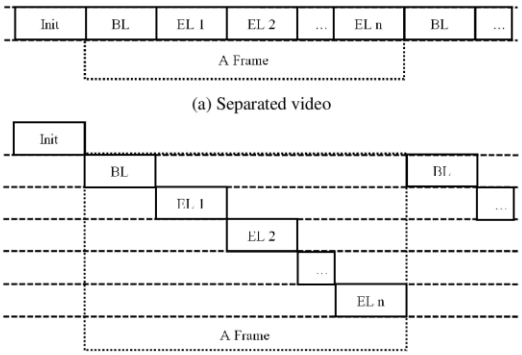


Figure 8: Comparison of video group size between AVC and SVC (640x360 and 1280x720)

2. Separated SVC

In this section, Separated NALU of SVC is described as shown in Fig. 10a. When a media server streams video layers to the network as shown in Fig. 10b which we can classify each NALU type as follows:



(a) Separated video

(b) Video streaming

Figure 10: SVC video

type 1: Initial data (Init) is any associated additional information of video or Non-Video Coding Layer (non-VCL).
type 2: Base layer (BL) is the lowest video data quality. It can display without enhancement layer. It is Video Coding Layer (VCL) that has NALU as 1 or 14.
type 3: Enhancement layer (EL) is to increase performance part for videos. If a client requests high video quality, a server will send many enhancement layers to a client. It is VCL that has NALU as 20.

3. Modified RTP header
From Fig. 11, we show separation of a NALU for creating RTP packet that has stable and limit packet size. Separated

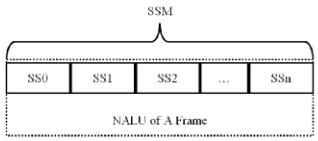


Figure 11: A separated NALU

packet sequence is defined as Sub Sequence Number (SSn), and separated packet number is Sub Sequence Max Number (SSM) which has value as n+1.

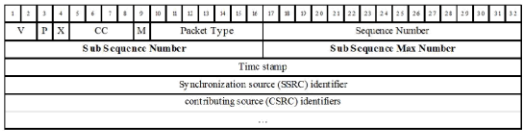


Figure 12: Our modified RTP header

Therefore, we build modified RTP header as shown in Fig. 12 by adding two parameters such as Sub Sequence Number 16 bits (since 33 to 48) and Sub Sequence Max number 16 bits (since 49 to 64). In additional, we use other parameters such as Packet Type which is used for classifying video type, Sequence Number is a frame sequence, Sub Sequence Number is divided packet sequence and Sub Sequence Max Number is used to separate packet number.

5.2. SDN controller

1. SVC available Bandwidth threshold
   This part presents calculated bandwidth threshold of SVC-SST and SVC-MST non-BWQLB in Table 8 and SVC-MST BWQLB in Table 9.

Table 8: SVC SST and SVC-MST non-BWQLB threshold

Resolution	Good	Medium	Poor
640x360	> 0.8 Mbps	0.6-0.8 Mbps	< 0.6 Mbps
1280x720	> 2.7 Mbps	2.3-2.7 Mbps	< 2.3 Mbps
1920x1080	> 5.5 Mbps	5.1-5.5 Mbps	< 5.1 Mbps

Table 9: SVC-MST BWQLB threshold

Resolution	Good	Medium	Poor
640x360	> 0.8 Mbps	0.6-0.8 Mbps	< 0.6 Mbps
1280x720	> 2.1 Mbps	1.7-2.1 Mbps	< 1.7 Mbps
1920x1080	> 3.4 Mbps	3.0-3.4 Mbps	< 3.0 Mbps

2. Algorithm design of SVC-MST BWQLB
   We modified BWQLB_Dijkstra_Algorithm() in Fig. 1 by enhancing bandwidth at Line 8 to 10. The bandwidth will be used to check paths of previous layers. For example, the base layer is the first path to calculate, so it will be examined normally by analyzing only topology constraints. Then, calculated base layer path will be recoded in path(l) and considered in other layers. All enhancement layers will be checked with calculated part in path(l) at Line 9. If the link between u and v node is calculated path part, the available bandwidth of the link will be minus the layer usage bandwidth weight. All enhancement layer is still recorded to path(l) for higher layer calculation. We have new parameters such as L which is a maximum layer number, path(l) is a calculated path of a layer and lw(l) is weight of a layer.

Algorithm 1 BWQLB_Dijkstra Algorithm

Input: G, s, L, path, lw
Output: dis[V], pre[V]

```
1:  procedure BWQLB_Dijkstra Algorithm
2:      for v in V do
3:          dis[v] ← ∞.
4:          pre[v] ← null.
5:      Q ← set(node.keys)
6:      while Q!=null do
7:          u ← Minimum_distance(d, Q)
8:          for v adjacent in u do
9:              for l in L do
10:                 if [u,v] in path(l) then
11:                     ew[u, v][][0] ← ew[u, v][][0] − lw(l).
12:             adj[v][][0] ← Min(dis[u][0], ew[u, v][][0])
13:             adj[] ← Quantized_to_3levels(adj[])
14:             adj[] ← Total_Weight(adj[])orBalance_Weight(adj[])
15:         if dis[v][0] = adj[v][][P] then
16:             if dis[v][1] > adj[v][][P+1] then
17:                 dis[v][0] ← adj[v][][P]
18:         if dis[v][0] > adj[v][][P] then
19:             dis[v][0] ← adj[v][][P]
```

5.3. Receiver

Fig. 13 is the receiver flowchart that consists of 2 threads such as receiver socket thread and video player thread. Although SVC-MST has unordered sequence problem, we designed the Sorted Data Buffer module in a receiver socket thread which consists of 5 functions as follows.

1. InitialChecker (data) function
   This function is used for checking Initial packet arrival or non-VCL NALU which is associated additional information. We set a packet type of initial packet as 101.
2. SubAccessUnitSort (data) function
   This function is used for collecting and sorting incomplete NALU packets that are separated from a sender as shown in Fig. 12. Moreover, this function will check NALU by Sub Sequence Max Number at modified RTP header and sort the incomplete NALU packets by Sub Sequence Number that has the same Packet Type and Sequence Number. The complete NALU packet will be sent to AccessUnitSort (data) function for analysing in Sequence Number term.

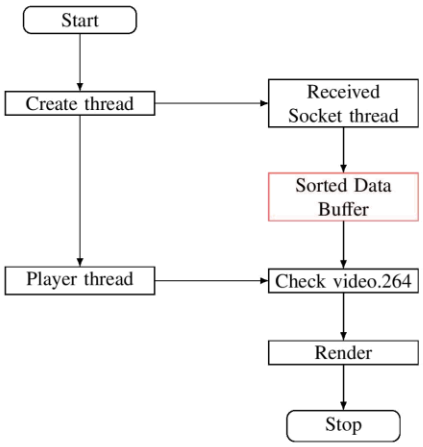


Figure 13: Receiver flowchart

3. AccessUnitSort (data) function
   This function is the main function that is used for sorting complete Access Unit by Sequence Number. If some received packets have wrong sequence, packets will be sent to BufferSort function. Conversely, if packet sequence matched, they will be collected in a render buffer. The function checked sequences: the base layer will check final layer sequence and enhancement layer will check previous layer sequence.
4. BufferSort (data) function
   This function is used for storing and sort NALU packets that were wrong sequence and were sent from AccessUnitSort function. It checks lowest buffer sequence that matches with a render buffer at every packet arrival.
5. BufferfullChecker(data) function
   This function is used for examining sorted buffer size. If a sorted buffer is full, this function will slide render buffer to sequence that has all layers in a sorted buffer.

5.4. Experiment

This section measures transmission of AVC, SVC-SST, SVC-MST non-BWQLB and SVC-MST BWQLB by streaming video from h1 to h2, h3 and h4 on a mesh topology as shown in Fig. 14. We define h2, h3 and h4 that request different video resolution. We measure transmission delay from bandwidth 0.1 Mbps to 10 Mbps. We use 3 video resolution such as 640x360, 1280x720 and 1920x1080. Moreover, this experiment consists of 2 cases such as single resolution transmission and multiple resolution transmission.

1. Single resolution transmission
   This case measures last packet delay of GOP that streams data from a server to clients. Then, we use resolution in a time that consists of 640x360, 1280x720 and 1920x1080 as show from Fig. 15 to 17.

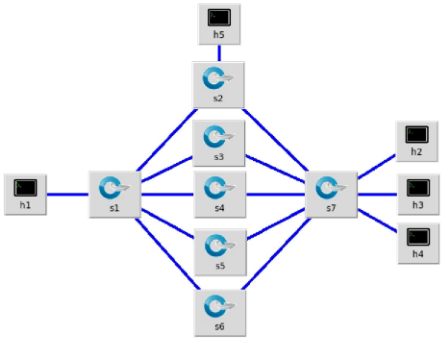


Figure 14: The 5 paths topology

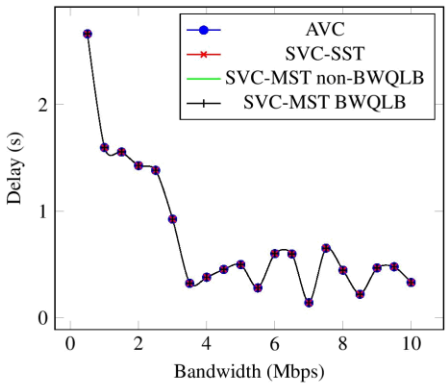


Figure 15: Comparison of 640x360 video delays

From Fig. 15 (640x360), the transmission delay of AVC, SVC-SST, SVC-MST non-BWQLB and SVC-MST BWQLB have the same delay due to base layer of SVC. Note that SVC base layer video is AVC video with the lowest resolution and all transmissions choose a single path. Thus, these delay results are overlapped as seen at the one line graph. In Fig. 16 (1280x720) and 17 (1920x1080), although the transmission of SVC-SST and SVC-MST non-BWQLB chose path that has the same quality with SVC-MST BWQLB, the SVC-SST and SVC-MST non-BWQLB have different delays of all transmissions since 1280x720 and 1920x1080 resolution of AVC and SVC have different sizes as show Fig. 5 to 7. Therefore, the SVC-SST and SVC-MST non-BWQLB have same transmission delay and have transmission delay more than AVC. Even though SVC-MST BWQLB streams SVC video that has video size more than AVC, the transmission of SVC-MST BWQLB splits layer to different paths when single path has insufficient bandwidth to stream all layers. The comparison of all transmissions in percent term from 0.1 to 10 Mbps reveals

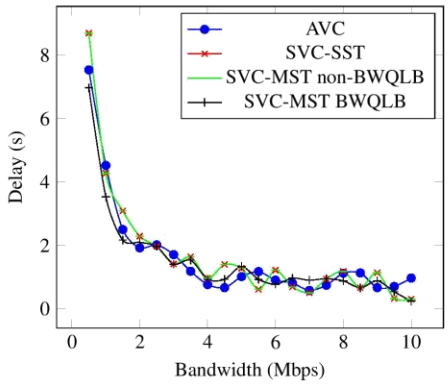


Figure 16: Comparison of 1280x720 video delays

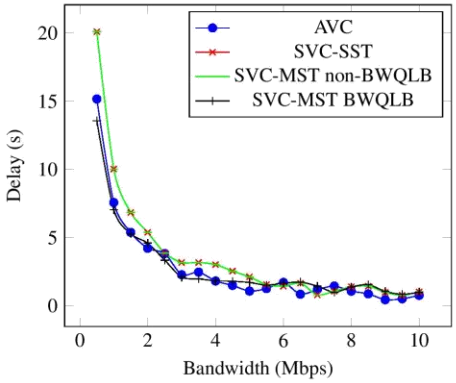


Figure 17: Comparison of 1920x1080 video delays

that the SVC-MST BWQLB has average delay lower than AVC such as 640x360, 1280x720 and 1920x1080 which are 0%, 5.33% and 0.136% respectively. It is lower than SVC-SST and SVC-MST non-BWQLB which are 11.62% and 30.84% respectively. Moreover, the comparison of all transmissions in percent term from 0.1 to 1.75 Mbps for 1280x720 and from 0.1 to 3 Mbps for 1920x1080 will show more SVC-MST BWQLB performance since both 1.75 and 3 Mbps are bandwidth threshold of SVC-MST BWQLB 1280x720 and 1920x1080 respectively. From 0.1 to 1.75 Mbps of 1280x720, SVC-MST BWQLB has transmission delay lower than AVC as 11.67% and lower than SVC-SST and SVC-MST non-BWQLB which are 11.67%. From 0.1 to 3 Mbps of 1920x1080, SVC-MST BWQLB still has transmission delay lower than AVC as 7.03% and lower than SVC-SST and SVC-MST non-BWQLB are 37.53%.

2. Multiple resolution transmission
   This case measures last packet delay of GOP that streams data from a server to clients. Then, we use a resolution of group in a time that consists of (640x360 and 1280x720) and (640x360, 1280x720 and 1920x1080) as shown in Fig. 18 and 19.

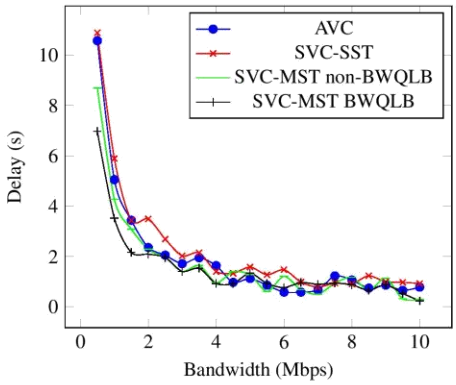


Figure 18: Comparison of video group delay (640x360 and 1280x720)

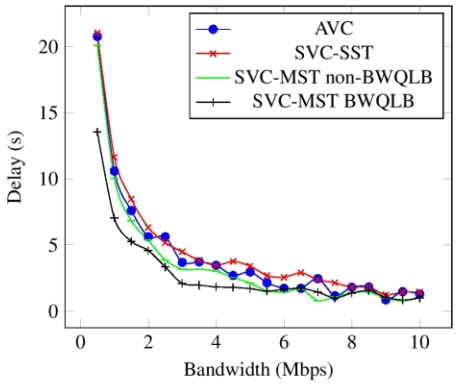


Figure 19: Comparison of video group delay (640x360, 1280x720 and 1920x1080)

From Fig. 18 (640x360 and 1280x720 group) and Fig. 19 (640x360, 1280x720 and 1920x1080 group), they show SVC-MST BWQLB manage bandwidth efficiently because streaming requests low bandwidth. When a server streams video groups of AVC, SVC-SST and SVC-MST non-BWQLB in low bandwidth, the transmission gets high congestion. However, SVC-MST BWQLB solves high congestion problem by dividing SVC video layers to various paths. In the comparison of all transmissions in percent term from 0.1 to 10 Mbps, The SVC-MST BWQLB has average delay lower than other transmission in (640x360 and 1280x720 group) and (640x360, 1280x720 and 1920x1080 group). The SVC-MST BWQLB has transmission delay lowers than AVC as 25.84% and 50.73% respectively. The SVC-MST

BWQLB has transmission delay lowers than SVC-SST as 46.77% and 66.64% respectively. The SVC-MST BWQLB has transmission delay lowers than SVC-MST non-BWQLB as 11.62% and 30.84% respectively. In addition, we show comparison of all transmissions in percent term from 0.1 to 1.75 Mbps for (640x360 and 1280x720 group) and from 0.1 to 3 Mbps for (640x360, 1280x720 and 1920x1080 group) will show more SVC-MST BWQLB performance. From 0.1 to 1.75 Mbps of (640x360 and 1280x720 group), SVC-MST BWQLB has transmission delay lower than AVC as 45.22%, lower than SVC SST as 60.74% and lower than SVC-MST non-BWQLB as 24.36%. From 0.1 to 3 Mbps of (640x360, 1280x720 and 1920x1080 group), SVC-MST BWQLB still has transmission delay lower than AVC as 50.20%, lower than SVC SST as 59.22% and lower than SVC-MST non-BWQLB as 37.53%.

6. Conclusion and Future work

In the paper, we design QLB for 3 video resolution such 640x360, 1280x720 and 1920x1080 that define jitter threshold from medium bandwidth relation. In the experiment, we compare results from OSPF (traditional network routing), QLB total weight and QLB balance weight. Moreover, we propose the SVC multicast streaming model based on SDN that reduces bandwidth congestion by using BWQLB call SVC-MST BWQLB. The SVC-MST BWQLB uses QLB that defines only bandwidth threshold for SVC layer.

In the comparison of OSPF, QLB total weight and QLB balance weight experiment; we set new QLB weights for 3 AVC resolution. We compare the new QLB weights with OSPF which uses it as the traditional network routing comparator. We define 7 paths on the topology that create for comparing between QLB total weight and QLB balance weight. The experimental results reveal that OSPF analyses only a bandwidth constraint, so it has video loss problem. The QLB total weight selects path similar to OSPF since it chooses a path by disregarding poor video quality. However, The QLB balance weight controls quality range with constraints. Thus, it gets the lowest loss rate which brings media rendering video normally.

The H.264 video compression or code experiment, we compare AVC with SVC by checking 3 resolution such as 640x360, 1280x720 and 1920x1080 of 5 GOPs. Regarding the storage space part, AVC uses disk space lower than SVC as 0%, 8.93% and 28.72% respectively in a single video resolution storage space case since a single video resolution of SVC has divided video layer overheads. On the other hand, When we compare several resolution such as (640x360 and 1280x720) and (640x360, 1280x720 and 1920x1080), SVC uses disk space lower than AVC in multiple resolution storage space. Since SVC can divide large video resolution to several lower video resolution, only highest video resolution with their video layer metadata is stored. Therefore, AVC storage space suitable for specific video resolution device but SVC storage space suitable for a media server that service flexible and many video resolution.

In AVC and SVC transmission experiment, the SVC transmission consists of SVC-SST and SVC-MST. In this paper, we focus in SVC-MST implementation. Our SVC-MST is developed by using bandwidth thresholds QLB analysis for dividing layer session call SVC-MST BWQLB. Additionally, the each video layer routing of SVC-MST BWQLB will be recorded the calculated paths and used them for next video layer routing. Since SVC-MST BWQLB can find insufficient links in order to deliver many video layers, it can choose better paths for sending high layers. The traditional SVC-MST (SVC-MST non-BWQLB) can divide layer session. However, it can analyze only constraints from current topology traffic. If there are links that have too low bandwidth to send many layers, SVC-MST non-BWQLB may send a video in the congestion links. In this experiment, we compare SVC-MST BWQLB with AVC, SVC-SST and SVC-MST non-BWQLB.

The transmission experiment consists of single resolution transmission and multiple resolution transmission. We measure delays of different bandwidths from 0.1 to 10 Mbps. From the single resolution transmission results, the SVC-MST BWQLB has average delay lower than 640x360, 1280x720 and 1920x1080 of AVC as 0%, 5.33% and 0.136% respectively, lower than SVC SST and SVC-MST non-BWQLB as 0%, 11.62% and 30.84% respectively. The multiple resolution transmission results, the SVC-MST BWQLB still has the lowest delay. SVC-MST BWQLB lower than (640x360 and 1280x720) and (640x360, 1280x720 and 1920x1080) of AVC are 25.84% and 50.73% respectively, lower than SVC-SST as 46.77% and 66.64% respectively and lower than SVC-MST non-BWQLB as 11.62% and 30.84% respectively. The SVC-MST BWQLB is a model that uses storage space efficiently for multiple resolution case and has the lowest delay in single resolution transmission case and multiple resolution transmission case. However, received videos of SVC-MST BWQLB and SVC-MST non-BWQLB has wrong packet sequence since they divide video layers to different sessions. Therefore, we designed sorted buffer for SVC-MST receiver that can be used for sorting intra-NALU, intra-layer and inter-layer.

For future work, we will test this model on real network devices with wireless environment. Bandwidth, delay, jitter and loss rate relation for several QLB thresholds are taken in to account to implement bitrate management of SVC-MST BWQLB video layer for reducing sorted buffer size. In the media management, we will measure this model with different video qualities, frame sizes and several motion videos, upgrade from H.264 to H.265 by changing AVC to HEVC and SVC to SHVC, and enhancing sorted buffer management to GOP analysis.

Bibliography

Piyawit Tantisarkhornkhet is a master's degree student, Department of Computer Engineering, Faculty of Engineering at Prince of Songkla University, Thailand under the supervision of Asst. Prof. Dr. Warodom Werapun. His research interests include adaptive video transmission, software-defined networking, and future networks. Contact him at 5810120060@email.psu.ac.th.

Warodom Werapun is an assistant professor in computer networks Department of Computer Engineering, Faculty of Engineering at Prince of Songkla University, Thailand. He received the Ph.D. degree from Institute National Polytechnique of Toulouse (INPT), France. His current areas of research are network architecture, IoTs and SDN. Contact him at warodom@coe.phuket.psu.ac.th.

Acknowledgments

This research has been supported by the Engineering Post-Graduate Scholarship, Faculty of Engineering, Prince of Songkla University, Thailand.

References

Cisco, V. N. I., 2016. Forecast and Methodology, 2015-2020. White Paper, Cisco.

Cisco Troubleshooting TechNotes, 2005. Understanding Buffer Misses and Failures - Cisco.
URL http://www.cisco.com/c/en/us/support/docs/interfaces-modules/channel-interface-processors/14620-41.html

Civanlar, S., Parlakisik, M., Tekalp, A. M., Gorkemli, B., Kaytaz, B., Onem, E., 2010. A qos-enabled openflow environment for scalable video streaming. In: GLOBECOM Workshops (GC Wkshps), 2010 IEEE. IEEE, pp. 351–356.

Craig, A., 2014. GroupFlow: Multicast Routing in Software Defined Networks — GroupFlow 0.2.0.
URL http://alexcraig.github.io/GroupFlow/

Detti, A., Bianchi, G., Kellerer, W., others, 2009. SVEF: an Open-Source Experimental Evaluation Framework. In: In Proc. of IEEE MediaWIN 2009, Sousse, Tunisia.

Egilmez, H. E., Civanlar, S., Tekalp, A. M., 2013. An optimization framework for QoS-enabled adaptive video streaming over OpenFlow networks. IEEE Transactions on Multimedia 15 (3), 710–715.

Heiko Schwarz, Marpe, D., Wiegand, T., 2007. Overview of the scalable video coding extension of the H. 264/AVC standard. IEEE Transactions on circuits and systems for video technology 17 (9), 1103–1120.

Holbrook, H., Cain, B., Haberman, B., 2006. Using internet group management protocol version 3 (IGMPv3) and multicast listener discovery protocol version 2 (MLDv2) for source-specific multicast. Tech. rep.

Hu, F., Hao, Q., Bao, K., 2014. A survey on software-defined network and openflow: From concept to implementation. IEEE Communications Surveys & Tutorials 16 (4), 2181–2206.

Ke, C.-H., 2012. myEvalSVC: an Integrated Simulation Framework for Evaluation of H. 264/SVC Transmission. TIIS 6 (1), 379–394.

Kreuzberger, C., Posch, D., Hellwagner, H., 2015. A scalable video coding dataset and toolchain for dynamic adaptive streaming over HTTP. In: Proceedings of the 6th ACM Multimedia Systems Conference. ACM, pp. 213–218.

Laga, S., Van Cleemput, T., Van Raemdonck, F., Vanhoutte, F., Bouten, N., Claeys, M., De Turck, F., 2014. Optimizing scalable video delivery through OpenFlow layer-based routing. In: Network Operations and Management Symposium (NOMS), 2014 IEEE. IEEE, pp. 1–4.

Sullivan, G. J., Topiwala, P., Luthra, A., 2004. The H. 264/AVC advanced video coding standard: Overview and introduction to the fidelity range extensions. In: Proc. of SPIE Vol. Vol. 5558. p. 455.

Tantisarkhornkhet, P., Werapun, W., 2016. QLB: QoS routing algorithm for Software-Defined Networking. In: Intelligent Signal Processing and Communication Systems (ISPACS), 2016 International Symposium on. IEEE, pp. 1–6.

Wenger, S., Wang, Y., Schierl, T., 2011. A. Eleftheriadis," RTP Payload Format for Scalable Video Coding. Tech. rep., RFC 6190, May.

Xue, N., Chen, X., Gong, L., Li, S., Hu, D., Zhu, Z., 2015. Demonstration of OpenFlow-controlled network orchestration for adaptive SVC video manycast. IEEE Transactions on Multimedia 17 (9), 1617–1629.

Yang, E., Ran, Y., Chen, S., Yang, J., 2014. A multicast architecture of SVC streaming over OpenFlow networks. In: Global Communications Conference (GLOBECOM), 2014 IEEE. IEEE, pp. 1323–1328.

Yang, J., Yang, E., Ran, Y., Chen, S., 2015. SDM^ 2 Cast An OpenFlow-Based, Software-Defined Scalable Multimedia Multicast Streaming Framework. IEEE Internet Computing 19 (4), 36–44.

Yu, T.-F., Wang, K., Hsu, Y.-H., 2015. Adaptive routing for video streaming with QoS support over SDN networks. In: Information Networking (ICOIN), 2015 International Conference on. IEEE, pp. 318–323.

ประวัติผู้เขียน

ชื่อ สกุล นายปิยวิทย์ ตันติสาครเขต

รหัสประจำตัวนักศึกษา 5810120060

วุฒิการศึกษา

วุฒิ	ชื่อสถาบัน	ปีที่สำเร็จการศึกษา
วิศวกรรมศาสตรบัณฑิต (วิศวกรรมคอมพิวเตอร์)	มหาวิทยาลัยสงขลานครินทร์	2557

ทุนการศึกษา (ที่ได้รับในระหว่างการศึกษา)

ทุนสนับสนุนทุนค่าธรรมเนียมการศึกษา ระดับบัณฑิตศึกษา ด้วยเงินรายได้โครงการทุนศิษย์ก้นกุฏิ สาขาวิชาวิศวกรรมคอมพิวเตอร์ ณ วิทยาเขตหาดใหญ่

ประสบการณ์การเรียนรู้ระหว่างการศึกษา

เป็นผู้ช่วยสอนทั้งในวิชาสายซอฟต์แวร์และสายฮาร์ดแวร์

ร่วมเข้าฟังการบรรยายเรื่อง เครือข่ายที่กำหนดโดยซอฟต์แวร์ โดยบริษัท Cisco วันที่ 10 เดือน มีนาคม พ.ศ. 2559

การตีพิมพ์เผยแพร่ผลงาน

1. P. Tantisarkhornkhet, W. Werapun, B. Paillassa, “SDN experimental on the PSU network,” In Intelligent Signal Processing and Communication Systems (ISPACS), 2016 International Symposium on *(IEEE 2016.)*, Phuket, Thailand, Oct. 2016, pp 1-6.
2. P. Tantisarkhornkhet, W. Werapun, “QLB: QoS routing algorithm for Software-Defined Networking,” in Intelligent Signal Processing and Communication Systems (ISPACS), 2016 International Symposium on *(IEEE 2016.)*, Phuket, Thailand, Oct. 2016, pp 1-6.
3. P. Tantisarkhornkhet, W. Werapun, “SVC-MST BWQLB Multicast over Software Defined Networking,” Computer Communications (Submitted to Journal of Computer Communications)