สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

อัลกอริธึมการแทนที่ข้อมูลสำหรับเว็บแคชเซิร์ฟเวอร์

REPLACEMENT ALGORITHM FOR WEB CACHE SERVER

T052462

ภาวิน สพโชค
PAWIN SOPECHOKE

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมไฟฟ้า
บัณฑิตวิทยาลัย
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
พ.ศ. 2547

ISBN 974-15-1049-7

วพ.
ว4748
2547
เลขหมู่.....
เลขทะเบียน 52462
วัน,เดือน,ปี 14 ก.ย. 2547

b.11182824
i.....

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

REPLACEMENT ALGORITHM FOR WEB CACHE SERVER

PAWIN SOPECHOKE

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENT FOR THE DEGREE OF
MASTER OF ENGINEERING IN ELECTRICAL ENGINEERING
SCHOOL OF GRADUATE STUDIES
KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG
2004

ISBN 974-15-1049-7

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

COPYRIGHT 2004
SCHOOL OF GRADUATE STUDIES
KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หัวข้อวิทยานิพนธ์	อัลกอริธึมการแทนที่ข้อมูลสำหรับเว็บแคชเซิร์ฟเวอร์
นักศึกษา	นาย ภาวิน สพโชค
รหัสนักศึกษา	42061088
ปริญญา	วิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชา	วิศวกรรมไฟฟ้า
พ.ศ.	2547
อาจารย์ผู้ควบคุมวิทยานิพนธ์	ดร. ศักดิ์ชัย ทิพย์จักษุรัตน์
อาจารย์ผู้ควบคุมวิทยานิพนธ์ร่วม	รศ. บรรจง ปิยธำรง

บทคัดย่อ

วิทยานิพนธ์นี้นำเสนอการปรับแต่งเว็บแคชเซิร์ฟเวอร์ โดยจะพิจารณาการพัฒนาปรับปรุงในส่วนของอัลกอริธึมการแทนที่ข้อมูล (Replacement Algorithm) โดยค่าที่นำมาวิเคราะห์คือ ปริมาณการร้องขอข้อมูลจากไคลเอนต์ ขนาดข้อมูลถ่ายโอน ประเภทของข้อมูลที่ร้องขอ เนื่องจากค่าต่าง ๆ เหล่านี้จะเปลี่ยนแปลงไปตามกลุ่มผู้ใช้งาน จากการศึกษาเราพบว่าอัลกอริธึมการแทนที่ข้อมูลแต่ละชนิด จะเหมาะสมกับชนิดของไฟล์ที่มีขนาดแตกต่างกัน จากคุณสมบัติดังกล่าว เราได้พัฒนาและปรับปรุงอัลกอริธึมการแทนที่ข้อมูลในเว็บแคชเซิร์ฟเวอร์ ให้เหมาะสมกับขนาดที่แตกต่างกันของข้อมูลที่มีการร้องขอ เพื่อปรับปรุงค่าของ Byte-Hit ของเว็บแคชเซิร์ฟเวอร์ให้สูงขึ้น และลดปริมาณการคับคั่งของข้อมูลภายในระบบเครือข่าย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Thesis Title	Replacement Algorithm for Web Cache Server
Student	Mr. Pawin Sopechoke
Student ID.	42061088
Degree	Master of Engineering
Programme	Electrical Engineering
Year	2004
Thesis Advisor	Dr. Sakchai Thipchaksurat
Thesis Co-Advisor	Assoc. Prof. Banjong Piyatamrong

ABSTRACT

This thesis described the improvement of web cache server by scoping in replacement algorithm of data which were collected from the clients. Data that we used for analysis were the requests from all clients, size of object and type of request which all of data would be changed depending on users. However, we found that each replacement algorithm was suitable for each type of data. Therefore, we developed web cache server to have the suitable replacement algorithm according to the size of data which were requested from the clients. As the result, the value of Byte-Hit ratio of web cache server could be increased and the congestion in network could be reduced.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กิตติกรรมประกาศ

คุณความดีอันใดที่บังเกิดจากวิทยานิพนธ์ฉบับนี้ ขอมอบแด่บิดาและมารดาของผู้วิจัย ผู้ที่คอยห่วงใย เข้าใจและให้การสนับสนุนในการศึกษามาโดยตลอด

วิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงลงได้ด้วยดี เนื่องจากได้รับความกรุณาจาก ดร. ศักดิ์ชัย ทิพย์จักษุรัตน์ อาจารย์ที่ปรึกษาและ รศ. บรรจง ปิยธำรง อาจารย์ที่ปรึกษาร่วม ที่ได้ช่วยเหลือในการให้คำแนะนำ ความรู้ทางทฤษฎีต่าง ๆ ที่ใช้ และชี้แนะแนวทางในการแก้ปัญหาต่าง ๆ อย่างทุ่มเท รวมทั้งฝึกฝนผู้วิจัยให้มีความสามารถในการทำวิจัยและพัฒนาได้อย่างมีประสิทธิภาพ ผู้วิจัยรู้สึกซาบซึ้งในความอนุเคราะห์จากท่านและขอกราบขอบพระคุณเป็นอย่างสูง

ขอขอบพระคุณ ดร. วรวัฒน์ ลิ้มโภคา, ดร.ชุติเมษฎ์ ศรีนิลทา, ดร. สุรินทร์ กิตติธรกุล และ ดร. สมศักดิ์ วลัยรัชต์ ซึ่งเป็นกรรมการสอบวิทยานิพนธ์ฉบับนี้ที่ได้กรุณาให้คำแนะนำที่มีคุณค่าซึ่งทำให้วิทยานิพนธ์ฉบับนี้สมบูรณ์ยิ่งขึ้น

ขอบคุณพี่ ๆ เพื่อน ๆ และน้อง ๆ นักศึกษาทุกคนรวมทั้งเพื่อน ๆ หลาย ๆ คนในโลกอินเทอร์เน็ตที่ช่วยเหลือให้คำแนะนำต่าง ๆ และให้กำลังใจแก่ผู้วิจัยตลอดมา

ขอขอบคุณ คุณประเสริฐ อัศวรุ่งสกุล, คุณ วราภรณ์ เลิศสุวรรณ และ ดร. ณยศ คุรุกิจโกศลสำหรับการให้คำปรึกษาด้านคณิตศาสตร์สำหรับแนวทางการวิเคราะห์ข้อมูลที่ได้มาจากการทดลอง

ภาวิน สพโชค

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ

หน้า

บทคัดย่อภาษาไทย I

บทคัดย่อภาษาอังกฤษ II

กิตติกรรมประกาศ III

สารบัญ IV

สารบัญตาราง XI

สารบัญรูป XII

บทที่ 1 บทนำ 1

1.1 กล่าวนำ 1

1.2 วัตถุประสงค์ของวิทยานิพนธ์ 2

1.3 หลักการใหม่ของวิทยานิพนธ์ 2

1.4 รายละเอียดของวิทยานิพนธ์ 3

บทที่ 2 หลักการทำงานของเว็บแคชเซิร์ฟเวอร์ 5

2.1 หลักการทำงานและการพัฒนาของเว็บแคชเซิร์ฟเวอร์ 5

2.2 รูปแบบความสัมพันธ์ระหว่างเว็บแคชเซิร์ฟเวอร์ 6

2.3 Squid เว็บแคชเซิร์ฟเวอร์ 11

2.3.1 ขั้นตอนการทำงานของ Squid เว็บแคชเซิร์ฟเวอร์ 17

2.3.1.1 ช่วงเริ่มต้นระบบ (System Initialize Phase) 17

2.3.1.2 ช่วงให้บริการ (Service Phase) 17

2.3.2 ไฟล์ Squid.conf กับพารามิเตอร์ที่มีผลต่อการทำงานของเว็บแคชเซิร์ฟเวอร์ 24

2.3.3 ไฟล์ Log ของ Squid เว็บแคชเซิร์ฟเวอร์ 27

2.3.3.1 รายละเอียดของ Log Tag 28

2.3.3.2 รายละเอียดของ Hierarchy Data 30

บทที่ 3 อัลกอริธึมการแทนที่ข้อมูลที่นำมาพิจารณาในงานวิจัย 32

3.1 ชนิดของอัลกอริธึมการแทนที่ข้อมูล 32

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ (ต่อ)

หน้า

3.2 การพิจารณาอัลกอริธึมการแทนที่ข้อมูลแบบ Least Frequency Used with Dynamic Aging (LFUDA) 33

3.2.1 อัลกอริธึมการแทนที่ข้อมูลแบบ Least Frequency Used (LFU) 34

3.2.2 อัลกอริธึมการแทนที่ข้อมูลแบบ Least Frequency Used-Aging (LFU-Aging) 34

3.2.3 อัลกอริธึมการแทนที่ข้อมูลแบบ Least Frequency Used with Dynamic Aging (LFUDA) 34

3.3 การพิจารณาอัลกอริธึมการแทนที่ข้อมูลแบบ Greedy-Dual-Size Frequency (GDSF) 35

3.3.1 อัลกอริธึมการแทนที่ข้อมูลแบบ Greedy-Dual (GD) 35

3.3.2 อัลกอริธึมการแทนที่ข้อมูลแบบ Greedy-Dual-Size (GD-Size) 36

3.3.3 อัลกอริธึมการแทนที่ข้อมูลแบบ Greedy-Dual-Size-Frequency (GDSF) 38

3.3.3.1 ขั้นตอนการทำงานของอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF 39

3.3.3.2 คุณสมบัติของอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF 41

บทที่ 4 ขั้นตอนการปรับแต่งอัลกอริธึมการแทนที่ข้อมูลของ Squid เว็บแคชเซิร์ฟเวอร์ 42

4.1 การวัดประสิทธิภาพของเว็บแคชเซิร์ฟเวอร์ 42

4.2 การวิเคราะห์อัลกอริธึมการแทนที่ข้อมูลที่นำมาพิจารณา 43

4.3 การจำลองการทำงานของเว็บแคชเซิร์ฟเวอร์ 46

4.3.1 เครื่องมือสำหรับการทดสอบการทำงานของเว็บแคชเซิร์ฟเวอร์ 46

4.3.2 ข้อมูลที่นำมาทดสอบ 46

4.3.3 โมเดลของการจำลองการทำงานของเว็บแคชเซิร์ฟเวอร์ในฟังก์ชั่นของอัลกอริธึมการแทนที่ข้อมูล 47

4.4 การปรับแต่งอัลกอริธึมการแทนที่ข้อมูลของ Squid เว็บแคชเซิร์ฟเวอร์ 48

4.5 ผลการทดสอบการทำงานของอัลกอริธึมการแทนที่ข้อมูลที่ถูกปรับแต่ง 52

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ (ต่อ)

หน้า

4.5.1 การเปรียบเทียบอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF กับอัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 10 Kbytes ซึ่งใช้ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังโดยตั้งแคชที่ขนาด 64 Mbytes 53

4.5.2 การเปรียบเทียบอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF กับอัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 100 Kbytes ซึ่งใช้ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังโดยตั้งแคชที่ขนาด 64 Mbytes 54

4.5.3 การเปรียบเทียบอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF กับอัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 1 Mbytes ซึ่งใช้ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังโดยตั้งแคชที่ขนาด 64 Mbytes 55

4.5.4 การเปรียบเทียบอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF กับอัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 10 Mbytes ซึ่งใช้ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังโดยตั้งแคชที่ขนาด 64 Mbytes 56

4.5.5 การเปรียบเทียบอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF กับอัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 10 Kbytes ซึ่งใช้ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังโดยตั้งแคชที่ขนาด 128 Mbytes 57

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ (ต่อ)

หน้า

4.5.6 การเปรียบเทียบอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF กับอัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 100 Kbytes ซึ่งใช้ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังโดยตั้งแคชที่ขนาด 128 Mbytes 58

4.5.7 การเปรียบเทียบอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF กับอัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 1 Mbytes ซึ่งใช้ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังโดยตั้งแคชที่ขนาด 128 Mbytes 59

4.5.8 การเปรียบเทียบอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF กับอัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 10 Mbytes ซึ่งใช้ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังโดยตั้งแคชที่ขนาด 128 Mbytes 60

4.5.9 การเปรียบเทียบอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF กับอัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 10 Kbytes ซึ่งใช้ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังโดยตั้งแคชที่ขนาด 256 Mbytes 61

4.5.10 การเปรียบเทียบอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF กับอัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 100 Kbytes ซึ่งใช้ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังโดยตั้งแคชที่ขนาด 256 Mbytes 62

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ (ต่อ)

หน้า

4.5.11 การเปรียบเทียบอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF กับอัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 1 Mbytes ซึ่งใช้ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังโดยตั้งแคชที่ขนาด 256 Mbytes.....63

4.5.12 การเปรียบเทียบอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF กับอัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 10 Mbytes ซึ่งใช้ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังโดยตั้งแคชที่ขนาด 256 Mbytes.....64

4.5.13 การเปรียบเทียบอัลกอริธึมการแทนที่ข้อมูลให้ค่า Threshold มีค่า 10 Kbytes ซึ่งใช้ข้อมูลจาก Clarknet โดยตั้งแคชที่ขนาด 64 Mbytes.....65

4.5.14 การเปรียบเทียบอัลกอริธึมการแทนที่ข้อมูลให้ค่า Threshold มีค่า 100 Kbytes ซึ่งใช้ข้อมูลจาก Clarknet โดยตั้งแคชที่ขนาด 64 Mbytes.....66

4.5.15 การเปรียบเทียบอัลกอริธึมการแทนที่ข้อมูลให้ค่า Threshold มีค่า 1 Mbytes ซึ่งใช้ข้อมูลจาก Clarknet โดยตั้งแคชที่ขนาด 64 Mbytes.....67

4.5.16 การเปรียบเทียบอัลกอริธึมการแทนที่ข้อมูลให้ค่า Threshold มีค่า 10 Mbytes ซึ่งใช้ข้อมูลจาก Clarknet โดยตั้งแคชที่ขนาด 64 Mbytes.....68

4.5.17 การเปรียบเทียบอัลกอริธึมการแทนที่ข้อมูลให้ค่า Threshold มีค่า 10 Kbytes ซึ่งใช้ข้อมูลจาก Clarknet โดยตั้งแคชที่ขนาด 128 Mbytes.....69

4.5.18 การเปรียบเทียบอัลกอริธึมการแทนที่ข้อมูลให้ค่า Threshold มีค่า 100 Kbytes ซึ่งใช้ข้อมูลจาก Clarknet โดยตั้งแคชที่ขนาด 128 Mbytes.....70

4.5.19 การเปรียบเทียบอัลกอริธึมการแทนที่ข้อมูลให้ค่า Threshold มีค่า 1 Mbytes ซึ่งใช้ข้อมูลจาก Clarknet โดยตั้งแคชที่ขนาด 128 Mbytes.....71

4.5.20 การเปรียบเทียบอัลกอริธึมการแทนที่ข้อมูลให้ค่า Threshold มีค่า 10 Mbytes ซึ่งใช้ข้อมูลจาก Clarknet โดยตั้งแคชที่ขนาด 128 Mbytes.....72

4.5.21 การเปรียบเทียบอัลกอริธึมการแทนที่ข้อมูลให้ค่า Threshold มีค่า 10 Kbytes ซึ่งใช้ข้อมูลจาก Clarknet โดยตั้งแคชที่ขนาด 256 Mbytes.....73

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ (ต่อ)

หน้า

4.5.22 การเปรียบเทียบอัลกอริธึมการแทนที่ข้อมูลให้ค่า Threshold มีค่า 100 Kbytes ซึ่งใช้ข้อมูลจาก Clarknet โดยตั้งแคชที่ขนาด 256 Mbytes....74

4.5.23 การเปรียบเทียบอัลกอริธึมการแทนที่ข้อมูลให้ค่า Threshold มีค่า 1 Mbytes ซึ่งใช้ข้อมูลจาก Clarknet โดยตั้งแคชที่ขนาด 256 Mbytes...75

4.5.24 การเปรียบเทียบอัลกอริธึมการแทนที่ข้อมูลให้ค่า Threshold มีค่า 10 Mbytes ซึ่งใช้ข้อมูลจาก Clarknet โดยตั้งแคชที่ขนาด 256 Mbytes...76

4.6 การวิเคราะห์ผลการทดสอบ....77

4.6.1 การวิเคราะห์ผลการปรับเปลี่ยนขนาดของแคช....77

4.6.2 การวิเคราะห์ผลการปรับเปลี่ยนค่า Threshold....78

4.7 การวิเคราะห์หาค่า Threshold ที่เหมาะสมสำหรับการปรับแต่งอัลกอริธึมการแทนที่ข้อมูล....79

บทที่ 5 การประยุกต์ใช้งานอัลกอริธึมการแทนที่ข้อมูลที่มีการปรับแต่งกับระบบเว็บแคชเซิร์ฟเวอร์....89

5.1 เครื่องมือที่ใช้ในการทดสอบ....89

5.2 ข้อมูลที่ใช้ในการทดสอบ....90

5.3 เงื่อนไขการพิจารณาข้อมูลที่ได้จากไฟล์ Log ของ Squid เว็บแคชเซิร์ฟเวอร์....91

5.4 ผลการทดสอบอัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่ง....92

5.4.1 การเปรียบเทียบอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF กับอัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 100 Kbytes....93

5.4.2 การเปรียบเทียบอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF กับอัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 1 Mbytes....94

5.4.3 การเปรียบเทียบอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF กับอัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 10 Mbytes....96

5.5 การวิเคราะห์ผลการทดสอบ....97

บทที่ 6 บทสรุป....99

6.1 สรุปงานวิจัยที่นำเสนอ....99

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ (ต่อ)

หน้า

6.2 สรุปผลการทดลอง 99

6.3 ปัญหาและอุปสรรค 100

6.4 แนวทางการปรับปรุงงานวิจัยในอนาคต 100

เอกสารอ้างอิง 102

ภาคผนวก 104

ภาคผนวก ก ตัวอย่างไฟล์ Configuration ของ Squid ในส่วนที่มีการตั้งค่าสำหรับการเรียกใช้งานอัลกอริธึมการแทนที่ข้อมูล 104

ภาคผนวก ข ผลงานวิจัยในระหว่างการศึกษาที่ได้รับการตีพิมพ์เผยแพร่ 112

ภาคผนวก ค ผลงานวิจัยในระหว่างการศึกษาที่ได้รับการตอบรับและกำลังรอการตีพิมพ์ 120

ประวัติผู้เขียน 138

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญตาราง

ตารางที่ หน้า

2.1 แสดงฟิลด์ต่าง ๆ ของไฟล์ Log ที่เกิดจากการประมวลผลการร้องขอของ Squid 27

4.1 แสดงค่า $\overline{\Delta H_i(gdsf)}$, $\overline{\Delta H_i(lfuda)}$, $\overline{\Delta H_b(gdsf)}$, $\overline{\Delta H_b(lfuda)}$, $\Delta gdsf$ และ $\Delta lfuda$ ของข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าลาดกระบัง 82

4.2 แสดงค่า $\overline{\Delta H_i(gdsf)}$, $\overline{\Delta H_i(lfuda)}$, $\overline{\Delta H_b(gdsf)}$, $\overline{\Delta H_b(lfuda)}$, $\Delta gdsf$ และ $\Delta lfuda$ ของข้อมูลจาก Clarknet 85

5.1 แสดงเปอร์เซ็นต์ของข้อมูลที่ร้องขอแบ่งตามชนิดของไฟล์ 91

5.2 แสดงเปอร์เซ็นต์ของข้อมูลที่ร้องขอแบ่งตามช่วงขนาดของไฟล์ 91

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูป

รูปที่		หน้า
1.1	โมเดล ไคลเอนต์-เซิร์ฟเวอร์ ของเว็บแคชเซิร์ฟเวอร์และไคลเอนต์	1
2.1	แสดงระบบเว็บแคชเซิร์ฟเวอร์แบบ Single Proxy	7
2.2	แสดงระบบเว็บแคชเซิร์ฟเวอร์ที่มีความสัมพันธ์แบบ Parent-Child	8
2.3	แสดงระบบเว็บแคชเซิร์ฟเวอร์ที่มีความสัมพันธ์แบบ Sibling	9
2.4	แสดงระบบเว็บแคชเซิร์ฟเวอร์ที่มีความสัมพันธ์แบบผสม	9
2.5	แสดงรายละเอียดรูปแบบของ ICP Message	10
2.6	โฟลว์ชาร์ตสรุปการทำงานเมื่อ Squid ได้รับการร้องขอจากไคลเอนต์	19
2.7	แสดงผลของการร้องขอที่พบข้อมูลในเมโมรีแคช	20
2.8	แสดงผลของการร้องขอที่พบข้อมูลในเมโมรีแคชแต่หมดอายุ	21
2.9	แสดงผลของการร้องขอที่พบข้อมูลในดิสก์แคช	21
2.10	แสดงผลของการร้องขอที่พบข้อมูลในดิสก์แคชแต่หมดอายุ	22
2.11	แสดงผลของการร้องขอที่ไม่พบข้อมูลในแคชของ Squid	23
2.12	แสดงการจัดเก็บข้อมูลในแคชของ Squid	23
4.1	กราฟแสดง %Hit ของอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF และอัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA ซึ่งใช้ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง	44
4.2	กราฟแสดง %Byte-Hit ของอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF และอัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA ซึ่งใช้ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง	44
4.3	กราฟเปรียบเทียบ %Hit ของอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF และอัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA ซึ่งใช้ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง แบ่งตามช่วงของขนาดข้อมูล	45
4.4	โมเดลของการจำลองการทำงานของเว็บแคชเซิร์ฟเวอร์ในฟังก์ชั่นของอัลกอริธึมการแทนที่ข้อมูล	47

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูป (ต่อ)

รูปที่		หน้า
4.5	ขั้นตอนการทำงานของการจำลองการทำงานของเว็บแคชเซิร์ฟเวอร์ในฟังก์ชั่นอัลกอริธึมการแทนที่ข้อมูล	48
4.6	แสดงการจัดเก็บข้อมูลในแคชของ Squid	50
4.7	การปรับแต่งอัลกอริธึมการแทนที่ข้อมูลของเว็บแคชเซิร์ฟเวอร์ที่นำเสนอ	52
4.8	กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 10 Kbytes	53
4.9	กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 10 Kbytes	53
4.10	กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 100 Kbytes	54
4.11	กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 100 Kbytes	54
4.12	กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 1 Mbytes	55
4.13	กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 1 Mbytes	55
4.14	กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 10 Mbytes	56
4.15	กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 10 Mbytes	56
4.16	กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 10 Kbytes	57
4.17	กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 10 Kbytes	57
4.18	กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 100 Kbytes	58
4.19	กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 100 Kbytes	58
4.20	กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 1 Mbytes	59
4.21	กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 1 Mbytes	59
4.22	กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 10 Mbytes	60
4.23	กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 10 Mbytes	60
4.24	กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 10 Kbytes	61
4.25	กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 10 Kbytes	61
4.26	กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 100 Kbytes	62
4.27	กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 100 Kbytes	62
4.28	กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 1 Mbytes	63
4.29	กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 1 Mbytes	63
4.30	กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 10 Mbytes	64
4.31	กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 10 Mbytes	64

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูป (ต่อ)

รูปที่		หน้า
4.32	กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 10 Kbytes	65
4.33	กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 10 Kbytes	65
4.34	กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 100 Kbytes	66
4.35	กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 100 Kbytes	66
4.36	กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 1 Mbytes	67
4.37	กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 1 Mbytes	67
4.38	กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 10 Mbytes	68
4.39	กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 10 Mbytes	68
4.40	กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 10 Kbytes	69
4.41	กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 10 Kbytes	69
4.42	กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 100 Kbytes	70
4.43	กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 100 Kbytes	70
4.44	กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 1 Mbytes	71
4.45	กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 1 Mbytes	71
4.46	กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 10 Mbytes	72
4.47	กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 10 Mbytes	72
4.48	กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 10 Kbytes	73
4.49	กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 10 Kbytes	73
4.50	กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 100 Kbytes	74
4.51	กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 100 Kbytes	74
4.52	กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 1 Mbytes	75
4.53	กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 1 Mbytes	75
4.54	กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 10 Mbytes	76
4.55	กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 10 Mbytes	76
4.56	กราฟแสดงผลต่างระหว่าง Δ*gdsf* และ Δ*lfuda*	84
4.57	กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 2 Mbytes	84
4.58	กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 2 Mbytes	85
4.59	กราฟแสดงผลต่างระหว่าง Δ*gdsf* และ Δ*lfuda*	87

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูป (ต่อ)

รูปที่		หน้า
4.60	กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 300 Kbytes	87
4.61	กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 300 Kbytes	88
5.1	รูปแบบการเชื่อมต่อของระบบที่ใช้ในการทดสอบ	90
5.2	กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 100 Kbytes	93
5.3	กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 100 Kbytes	93
5.4	กราฟเปรียบเทียบ %Hit ของการปรับค่า Threshold มีค่า 100 Kbytes แบ่งตามช่วงของขนาดข้อมูล	94
5.5	กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 1 Mbytes	94
5.6	กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 1 Mbytes	95
5.7	กราฟเปรียบเทียบ %Hit ของการปรับค่า Threshold มีค่า 1 Mbytes แบ่งตามช่วงของขนาดข้อมูล	95
5.8	กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 10 Mbytes	96
5.9	กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 10 Mbytes	96
5.10	กราฟเปรียบเทียบ %Hit ของการปรับค่า Threshold มีค่า 10 Mbytes แบ่งตามช่วงของขนาดข้อมูล	97

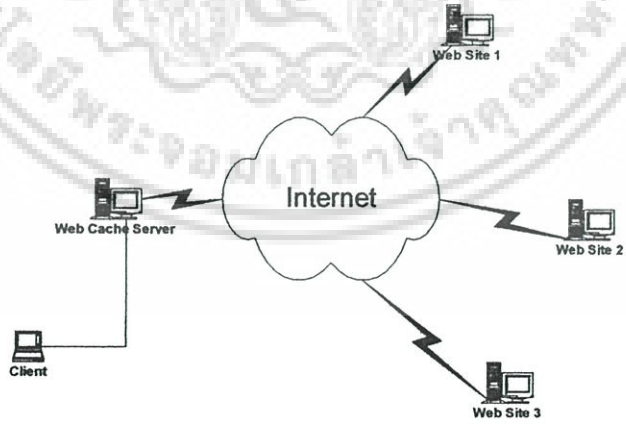
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 1
บทนำ

1.1 กล่าวนำ

ในปัจจุบันอัตราการเติบโตของการใช้เวิลด์ไวด์เว็บได้เพิ่มขึ้นอย่างรวดเร็ว ซึ่งรูปแบบของการติดต่อจะเป็นแบบไคลเอนต์-เซิร์ฟเวอร์ โดยไคลเอนต์จะติดต่อไปยังเซิร์ฟเวอร์ต้นทางของเว็บไซต์เพื่อขอข้อมูลเว็บเพ็จ ข้อมูลเว็บเพ็จที่ได้รับ จะประกอบด้วยข้อมูลหลายชนิดทั้งส่วนของรูปภาพ ตัวอักษร ไฟล์วีดีโอ ภาพเคลื่อนไหว ซึ่งข้อมูลเหล่านี้จะมีลักษณะที่แตกต่างกันออกไป และจากการเติบโตของการใช้งานนี้ทำให้เกิดปัญหาการคับคั่งของข้อมูลบนเครือข่าย ผลที่เกิดขึ้นคือ การล่าช้าของการได้รับข้อมูลที่ต้องการ

ดังนั้นเพื่อลดปัญหาดังกล่าวเว็บแคชเซิร์ฟเวอร์จึงได้ถูกพัฒนาขึ้น หลักการทำงานของเว็บแคชเซิร์ฟเวอร์จะแสดงได้ดังรูปที่ 1.1 คือ เว็บแคชเซิร์ฟเวอร์จะรอรับการร้องขอข้อมูลจากไคลเอนต์ ถ้าเว็บแคชเซิร์ฟเวอร์ค้นพบข้อมูลที่ร้องขอในแคช (Cache) ซึ่งเป็นหน่วยความจำที่เว็บแคชเซิร์ฟเวอร์ใช้สำหรับเก็บข้อมูลที่ได้รับจากเซิร์ฟเวอร์ต้นทางของเว็บไซต์ และยังไม่หมดอายุ (เวลาที่กำหนดไว้ เพื่อเก็บข้อมูลไว้ในแคชของเว็บแคชเซิร์ฟเวอร์) จะทำการส่งข้อมูลที่เก็บอยู่ในแคชไปให้ไคลเอนต์ แต่ถ้าไม่พบข้อมูลหรือข้อมูลที่พบหมดอายุไปแล้ว จะทำการร้องขอไปยังเซิร์ฟเวอร์ต้นทางของเว็บไซต์ เพื่อให้เซิร์ฟเวอร์ต้นทางของเว็บไซต์ส่งข้อมูลใหม่มาให้เว็บแคชเซิร์ฟเวอร์ และเว็บแคชเซิร์ฟเวอร์จะทำการส่งข้อมูลกลับไปให้ไคลเอนต์ที่ร้องขอ พร้อมกับทำการสำเนาเก็บเอาไว้ในแคชด้วยเพื่อให้บริการไคลเอนต์ ในกรณีที่มีการร้องขอข้อมูลชุดเดิมอีกในโอกาสต่อไป



รูปที่ 1.1 โมเดล ไคลเอนต์-เซิร์ฟเวอร์ ของเว็บแคชเซิร์ฟเวอร์และไคลเอนต์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เนื่องจากเว็บแคชเซิร์ฟเวอร์มีขนาดของแคชที่จำกัด และเนื้อที่แคชของเว็บแคชเซิร์ฟเวอร์จะมีค่าลดลง เมื่อมีการเก็บข้อมูลที่นำมาจากเซิร์ฟเวอร์ต้นทางของเว็บไซต์ทุกครั้ง หลังจากที่ได้ส่งข้อมูลไปให้ไคลเอนต์ เมื่อข้อมูลที่เก็บอยู่ในแคชของเว็บแคชเซิร์ฟเวอร์มีค่ามากขึ้นจนถึงค่ากำหนดขนาดของแคช เว็บแคชเซิร์ฟเวอร์จะเรียกใช้ฟังก์ชั่นที่ทำหน้าที่เลือกข้อมูลที่เก็บอยู่ในแคชและนำข้อมูลที่เลือกไว้ออกจากแคชของเว็บแคชเซิร์ฟเวอร์จนกระทั่งขนาดของแคชมีค่าเท่ากับค่ากำหนดที่เว็บแคชเซิร์ฟเวอร์ตั้งไว้ ก็จะหยุดเรียกใช้งานฟังก์ชั่นและเริ่มทำการเก็บข้อมูลลงไปในแคชอีกครั้ง โดยฟังก์ชั่นที่เว็บแคชเซิร์ฟเวอร์เรียกใช้จะมีชื่อว่า อัลกอริธึมการแทนที่ข้อมูล (Replacement Algorithm) [1]

1.2 วัตถุประสงค์ของวิทยานิพนธ์

งานวิจัยนี้มุ่งเน้นการเพิ่มสมรรถนะของเว็บแคชเซิร์ฟเวอร์เพื่อให้เว็บแคชเซิร์ฟเวอร์มีค่าของเปอร์เซ็นต์การพบข้อมูลแบบไบต์สูงขึ้น และมีค่าของเปอร์เซ็นต์การพบข้อมูลที่ใกล้เคียงกับข้อมูลเดิม โดยศึกษาอัลกอริธึมการแทนที่ข้อมูล ของเว็บแคชเซิร์ฟเวอร์ให้มีความเหมาะสมกับรูปแบบของข้อมูลที่วิ่งอยู่ในระบบเครือข่าย โดยจะวิเคราะห์ตามประเภทของข้อมูลแต่ละชนิดเช่น ข้อมูลรูปภาพ ข้อมูลเท็กซ์ และข้อมูลอื่น ๆ เนื่องจากข้อมูลแต่ละชนิดจะมีช่วงขนาดของข้อมูลไม่เหมือนกัน แม้ในปัจจุบันงานวิจัยส่วนใหญ่ที่พัฒนาส่วนของอัลกอริธึมการแทนที่ข้อมูลได้พยายามหาตัวแปรที่มีผลกระทบต่อประสิทธิภาพการทำงานของเว็บแคชเซิร์ฟเวอร์ [2] แต่อัลกอริธึมการแทนที่ข้อมูลแต่ละชนิดจะให้ข้อดีในแต่ละด้านกัน ดังนั้นในการจัดทำวิทยานิพนธ์เรื่อง "อัลกอริธึมการแทนที่ข้อมูลสำหรับเว็บแคชเซิร์ฟเวอร์" (Replacement Algorithm for Web Cache Server) จึงได้กำหนดวัตถุประสงค์ในการทำวิทยานิพนธ์ไว้ดังนี้

1) ศึกษาค้นคว้าและพัฒนาเทคนิควิธีการใหม่ สำหรับการปรับแต่งอัลกอริธึมการแทนที่ข้อมูลของเว็บแคชเซิร์ฟเวอร์ เพื่อให้เว็บแคชเซิร์ฟเวอร์มีค่าของอัตราการพบข้อมูลแบบไบต์ที่สูงขึ้นและเหมาะสมกับระบบที่นำไปประยุกต์ใช้
2) นำลักษณะการทำงานของอัลกอริธึมแต่ละชนิดมาพิจารณาถึงความสัมพันธ์เพื่อหาตัวแปรที่นำไปประยุกต์สำหรับการออกแบบอัลกอริธึมการแทนที่ข้อมูล
3) สามารถนำอัลกอริธึมที่ออกแบบไปประยุกต์ใช้งานได้อย่างถูกต้อง
4) เหมาะสมในการนำไปประยุกต์ใช้งานกับระบบเว็บแคชเซิร์ฟเวอร์ที่มีอยู่ทั่วไปในระบบ

1.3 หลักการใหม่ของวิทยานิพนธ์

ในการจัดทำวิทยานิพนธ์ฉบับนี้ ได้มีการนำเสนอการปรับแต่งอัลกอริธึมการแทนที่ข้อมูลของเว็บแคชเซิร์ฟเวอร์ โดยนำลักษณะการทำงานของอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF และอัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA มาผสมเข้าด้วยกันโดยใช้ขนาดของไฟล์ที่ถูกเก็บไว้ใน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แคชของเว็บแคชเซิร์ฟเวอร์เป็นตัวแปรที่จะนำมาใช้เลือกชนิดของอัลกอริธึมการแทนที่ข้อมูลที่เรียกใช้ ซึ่งผลที่ได้คือไฟล์ที่มีขนาดใหญ่และมีการร้องขอจากไคลเอนต์หลายครั้ง จะยังถูกเก็บไว้ในแคชของเว็บแคชเซิร์ฟเวอร์ ซึ่งจะทำให้ค่าของอัตราการพบข้อมูลแบบไบต์มีค่าสูง และได้ทำการจำลองรูปแบบของอัลกอริธึมการแทนที่ข้อมูลที่มีการปรับแต่งเข้ากับชุดข้อมูลที่นำมาทดสอบเพื่อหาขนาดของค่าที่จะนำมาใช้เป็นค่าที่เหมาะสม นอกจากนี้ยังได้ทำการประยุกต์อัลกอริธึมที่มีการปรับแต่งเข้ากับระบบของเว็บแคชเซิร์ฟเวอร์จริงเพื่อทดสอบเปรียบภายในระบบเครือข่ายจริง เพื่อยืนยันถึงประสิทธิภาพของเว็บแคชเซิร์ฟเวอร์ที่ได้จากการปรับแต่งอัลกอริธึมการแทนที่ข้อมูล

1.4 รายละเอียดของวิทยานิพนธ์

วิทยานิพนธ์ฉบับนี้ เป็นการนำเสนอการปรับแต่งอัลกอริธึมการแทนที่ข้อมูลซึ่งประยุกต์ข้อดีของอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF และอัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA เข้ามารวมกันเพื่อให้ได้ค่าของ %Byte-Hit ที่ดีขึ้นและค่าของ %Hit เปลี่ยนแปลงน้อยที่สุด โดยรายละเอียดต่าง ๆ ภายในวิทยานิพนธ์ฉบับนี้ได้จัดแบ่งในส่วนของเนื้อหาออกเป็น 6 บท ซึ่งแต่ละบทมีหัวข้อและเนื้อหาดังต่อไปนี้

บทที่ 1 บทนำ

อธิบายถึงวัตถุประสงค์และหลักการใหม่ ที่ได้นำเสนอไว้ภายในวิทยานิพนธ์ รวมไปถึงรายละเอียดเนื้อหาโดยสรุปของแต่ละบท

บทที่ 2 เว็บแคชเซิร์ฟเวอร์

อธิบายถึงระบบการทำงานของเว็บแคชเซิร์ฟเวอร์ ชนิดของเว็บแคชเซิร์ฟเวอร์ การทำงานของ Squid เว็บแคชเซิร์ฟเวอร์ ซึ่งเป็นเว็บแคชเซิร์ฟเวอร์ที่นำมาใช้ในงานวิจัย และส่วนประกอบที่สำคัญของ Squid เว็บแคชเซิร์ฟเวอร์

บทที่ 3 อัลกอริธึมการแทนที่ข้อมูล (Replacement Algorithm)

อธิบายถึงที่มา การทำงานของอัลกอริธึมการแทนที่ข้อมูลต่าง ๆ ที่เว็บแคชเซิร์ฟเวอร์ใช้ในการจัดการกับข้อมูลที่อยู่ภายในแคชของเว็บแคชเซิร์ฟเวอร์ และทำการทดสอบเพื่อหาข้อดีและข้อเสียของอัลกอริธึมการแทนที่ข้อมูลที่ได้รับความนิยมนำไปประยุกต์ใช้งานในเว็บแคชเซิร์ฟเวอร์ต่าง ๆ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 4 การวิเคราะห์และปรับแต่งอัลกอริธึมการแทนที่ข้อมูล

อธิบายถึงการวิเคราะห์ข้อดีข้อเสียของอัลกอริธึมการแทนที่ข้อมูลแค้ละชนิด และขั้นตอนการนำข้อดีของอัลกอริธึมการแทนที่ข้อมูลแต่ละชนิดมาปรับแต่งเป็นอัลกอริธึมการแทนที่ข้อมูลที่ออกแบบไว้

บทที่ 5 การทดสอบและประยุกต์ใช้งานอัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่ง

แสดงถึงผลการทดสอบของอัลกอริธึมการแทนที่ข้อมูลที่ได้ทำการปรับแต่งไว้เทียบกับอัลกอริธึมก่อนที่จะนำมาปรับแต่งโดยข้อมูลอ้างอิงที่นำมาทดสอบ

บทที่ 6 บทสรุป

สรุปและวิจารณ์ผลที่ได้จากการทดลอง พร้อมทั้งปัญหาที่เกิดขึ้น ตลอดจนข้อเสนอแนะต่าง ๆ ในการทำวิจัยต่อไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 2
หลักการทำงานของเว็บแคชเซิร์ฟเวอร์

ในบทนี้จะกล่าวถึงหลักการทำงานทั่วไปของเว็บแคชเซิร์ฟเวอร์ [3] หลังจากนั้นจะกล่าวถึงเว็บแคชเซิร์ฟเวอร์ที่ได้มีการพัฒนาขึ้นมาใช้งาน หลังจากนั้นจะกล่าวถึงรูปแบบความสัมพันธ์ของเว็บแคชเซิร์ฟเวอร์ และสุดท้ายจะกล่าวถึง Squid [4] ซึ่งเป็นเว็บแคชเซิร์ฟเวอร์ที่สนใจจะศึกษา โดยจะอธิบายถึงส่วนประกอบต่าง ๆ หลักการทำงาน พารามิเตอร์ที่สำคัญในการปรับตั้งการทำงานของ Squid และ รูปแบบของไฟล์ Log และพารามิเตอร์ที่มีอยู่ในไฟล์ Log เพื่อให้เข้าใจและทราบถึงภาพรวมในการทำงานของเว็บแคชเซิร์ฟเวอร์

2.1 หลักการทำงานและการพัฒนาของเว็บแคชเซิร์ฟเวอร์

การทำงานของเว็บแคชเซิร์ฟเวอร์หรือพรอกซี จะใช้หลักการของไคลเอนต์/เซิร์ฟเวอร์ที่มีตัวเครื่อง เว็บแคชเซิร์ฟเวอร์เป็นผู้ให้บริการแก่ไคลเอนต์ซึ่งก็คือ HTTP Client โดยบริการกล่าวถึงคือการเป็นตัวแทนของไคลเอนต์เหล่านั้นในการไปดึงข้อมูลจากภายนอกมาให้แก่ไคลเอนต์ตามที่ได้รับการร้องขอมา ทั้งนี้เว็บแคชเซิร์ฟเวอร์ยังสามารถทำการจัดเก็บข้อมูลเหล่านั้นไว้ชั่วระยะเวลาหนึ่ง (การทำแคช) ซึ่งถ้ามีการร้องขอข้อมูลเดียวกันเข้ามาเว็บแคชเซิร์ฟเวอร์ก็สามารถนำข้อมูลที่จัดเก็บไว้ในแคชส่งให้ไคลเอนต์ได้เลยโดยไม่ต้องไปดึงมาจากภายนอกอีก

แอพพลิเคชั่นเว็บแคช ที่เป็นที่นิยมในปัจจุบันมีอยู่อย่างหลากหลายทั้งที่พัฒนาขึ้นเพื่อจุดประสงค์ทางการค้าและไม่หวังผลตอบแทน โดยแอพพลิเคชั่นจะมีความสามารถในการทำงานที่แตกต่างกันออกไป ในที่นี้จะขอแบ่งแอพพลิเคชั่นออกตามกลุ่มของผู้พัฒนา

CERN/W3C [5] เป็นต้นฉบับของเว็บแคช โดยพัฒนาเริ่มแรกที่ CERN ต่อมาถูก W3C นำมาพัฒนาต่อ

Harvest Cache เป็นส่วนประกอบหนึ่งของ Harvest Project คือข้อมูลแคชและที่นี่เป็นแห่งแรกที่มีการพัฒนาโพรโตคอล ICP (Internet Cache Protocol) [6] โดยตัว Harvest Cache ถูกนำไปพัฒนาต่อโดย Squid และ Netscape [7] โดยสามารถรองรับการทำงานของโพรโตคอล: HTTP, FTP, gopher และ ICP

Netscape Proxy Server: ถูกพัฒนาโดยนักพัฒนาคนสำคัญที่เป็นคนสร้าง CERN Proxy โดยสามารถรองรับการทำงานของโพรโตคอลได้หลายตัว เช่น HTTP, FTP, Gopher, SSL, ICP. CARP (Cache Array Routing Protocol)

DeleGate [8] เป็นเว็บแคชเซิร์ฟเวอร์ที่ทำงานได้บนหลากหลาย Platform ของระบบปฏิบัติการ ไม่ว่าจะเป็น Unix, Windows, OS/2 และสามารถรองรับโพรโตคอลเช่น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

HTTP, FTP, Gopher, NNTP, POP, SMTP, Telnet, WAIS, X, CU-SeeMe, Socks, SSL และ ICP version 2

Microsoft proxy [9] : ทำงานภายใต้สภาพแวดล้อมของ Windows NT และทำงานร่วมกับโพรโตคอล HTTP และ CARP

Wcol/Catalyst [10] : อาศัยการทำ perfecting เพื่อลดความล่าช้าและเพิ่ม bandwidth

Novell BorderManager FastCache [11] : รองรับโพรโตคอล: HTTP, ICP ได้

Apache [12] : เป็นเว็บเซิร์ฟเวอร์ที่มีความสามารถในการทำแคชด้วย

Cacheflow [13] : ใช้เทคนิค Prefecting ในการลดเวลาแฝงในการร้องขอข้อมูล

MOWS [14] : ใช้ภาษาจาวาในการพัฒนาโปรแกรมและสามารถทำงานร่วมกับโพรโตคอล ICP ได้

นอกจากนี้ยังมีแอพพลิเคชั่นเว็บแคชบางตัวซึ่งไม่ได้กล่าวถึงเพราะว่าในงานวิจัยนี้มุ่งเน้นศึกษาการทำงานของโปรแกรม Squid เป็นหลัก เนื่องจากเป็นโปรแกรมที่ถูกนิยมนำไปใช้งานในหลาย ๆ ที่ ทั้งสถาบันการศึกษาและภายในองค์การหรือบริษัทต่าง ๆ

2.2 รูปแบบความสัมพันธ์ระหว่างเว็บแคชเซิร์ฟเวอร์

เราสามารถแบ่งประเภทของเว็บแคชเซิร์ฟเวอร์ตามลักษณะการจัดวางรูปแบบของแคชตามโทโปโลยี (Topology) ได้เป็นสองประเภท คือ Simple Web-cache และ Cooperate Web-Cache

1 Simple Web-cache ซึ่งหมายถึงการเชื่อมต่อเว็บแคชเซิร์ฟเวอร์ที่มีการทำงานต่อกันเป็นลำดับชั้น (Hierarchy) เมื่อไม่พบข้อมูลที่แคชของตัวเองก็จะทำการร้องขอข้อมูลดังกล่าวกับเว็บแคชเซิร์ฟเวอร์เครื่องอื่นที่อยู่ในระดับสูงขึ้นไป

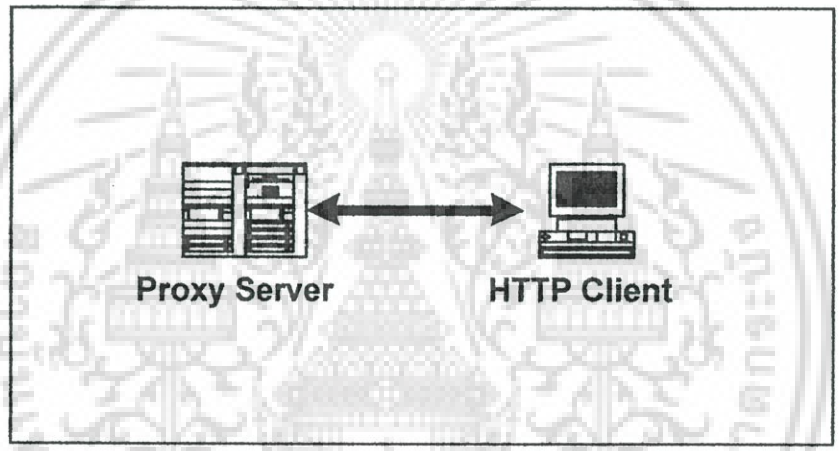
2 Cooperate Web-Cache สามารถแบ่งกระจายความรับผิดชอบในการให้ตอบสนองต่อข้อมูลที่ถูกร้องขอจากเซิร์ฟเวอร์ต้นทางที่อยู่ภายใต้กลุ่มของโดเมนเนมที่กำหนด เพื่อเป็นการลดความซ้ำซ้อนในการทำสำเนาข้อมูลในแต่ละดิสก์แคชของเว็บแคชเซิร์ฟเวอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เราสามารถอธิบายความสัมพันธ์ระหว่างเครื่องเว็บแคชเซิร์ฟเวอร์ได้ดังนี้

1 Single Proxy

เป็นระบบที่มีเว็บแคชเซิร์ฟเวอร์เพียงตัวเดียวโดยไคลเอนต์ทั้งหมดที่จะติดต่อขอใช้บริการจากเครื่องเซิร์ฟเวอร์นี้เพียงเครื่องเดียว เป็นระบบที่ง่ายต่อการติดตั้งและปรับแต่งค่าทั้งทางฝั่งเซิร์ฟเวอร์ และไคลเอนต์ ตลอดจนการควบคุมสิทธิในการใช้งานของผู้ใช้แต่การที่มีเครื่องให้บริการเพียงเครื่องเดียวอาจจะเสี่ยงแต่ความล้มเหลวของระบบ (Single point of failure) และปัญหาคอขวดที่เครื่องเว็บแคชเซิร์ฟเวอร์มีความสามารถในการให้บริการไม่เพียงพอต่อปริมาณความต้องการของไคลเอนต์ นอกจากนี้การออกแบบระบบให้มีเครื่องเว็บแคชเซิร์ฟเวอร์เพียงเครื่องเดียวนั้นทำให้ต้องใช้เครื่องที่มีประสิทธิภาพในการทำงานและทรัพยากรที่สูงซึ่งหมายความว่าจะต้องใช้งบประมาณที่สูงมากสำหรับเครื่องเซิร์ฟเวอร์นี้

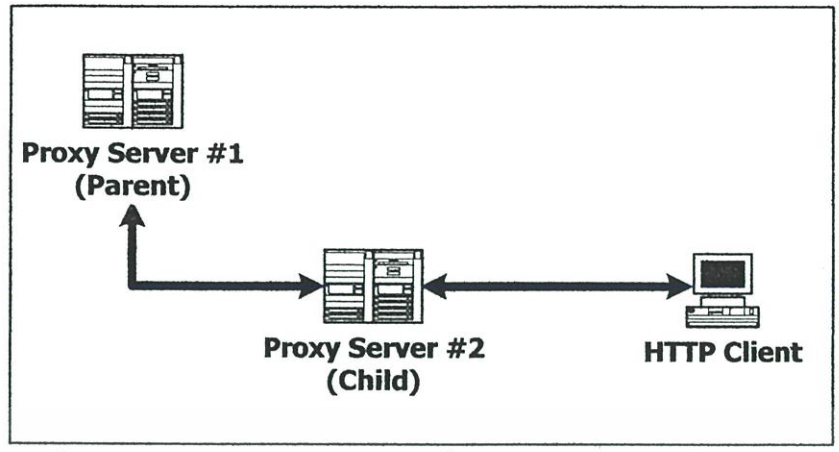


รูปที่ 2.1 แสดงระบบเว็บแคชเซิร์ฟเวอร์แบบ Single Proxy

2 Parent-Child Relation

เป็นระบบที่ประกอบไปด้วย เว็บแคชเซิร์ฟเวอร์ ตั้งแต่สองเครื่องขึ้นไป โดยจะมีเครื่องเว็บแคชเซิร์ฟเวอร์ หลักที่มีขนาดใหญ่และประสิทธิภาพสูงอยู่ที่ระดับบนสุด (Parent) ทำหน้าที่คอยรับการร้องขอบริการจากเครื่อง Proxy ที่อยู่ระดับต่ำกว่า (Child) ซึ่งโดยมากจะเป็นเครื่อง เว็บแคชเซิร์ฟเวอร์ ที่มีขนาดและประสิทธิภาพต่ำกว่าอยู่ในระดับถัดมาเพื่อทำหน้าที่รับการร้องขอบริการจากเครื่อง ไคลเอนต์ ทั้งนี้จำนวนของเครื่อง เว็บแคชเซิร์ฟเวอร์ ที่ทำหน้าที่เป็น Child นั้นสามารถมีได้หลายเครื่องเพื่อเป็นการกระจายภาระงานไปยัง เว็บแคชเซิร์ฟเวอร์ หลายตัวและเข้าถึงกลุ่มของ ไคลเอนต์ ได้อย่างหลากหลาย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



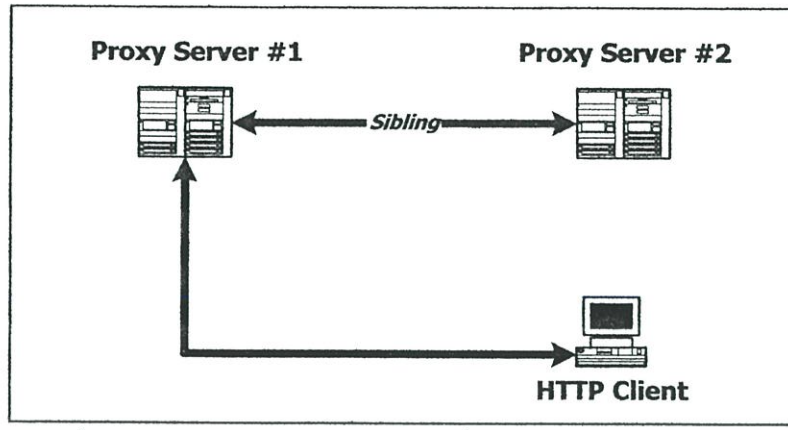
รูปที่ 2.2 แสดงระบบเว็บแคชเซิร์ฟเวอร์ที่มีความสัมพันธ์แบบ Parent-Child

ระบบเว็บแคชเซิร์ฟเวอร์ที่มีความสัมพันธ์แบบนี้เป็นตัวอย่างของระบบแบบ Simple Web-Cache นั้นคือเครื่องเว็บแคชเซิร์ฟเวอร์หมายเลข 2 ในรูปที่ 2.2 จะต้องทำการร้องขอข้อมูลผ่านเครื่องเว็บแคชเซิร์ฟเวอร์หมายเลข 1 ทุกครั้งที่ไม่พบข้อมูลในแคชที่ตัวเอง และเมื่อเครื่องหมายเลข 1 นำข้อมูลที่ถูกร้องขอส่งมาให้เครื่องหมายเลข 2 นั้นตัวเครื่องหมายเลข 1 เองก็จะทำการสำเนาข้อมูลดังกล่าวไว้ด้วย ตรงนี้เองที่ทำให้เกิดความซ้ำซ้อนในการเก็บข้อมูล ยิ่งถ้าในกรณีที่ในระบบมีเว็บแคชเซิร์ฟเวอร์หลายเครื่องและมีความสัมพันธ์แบบ Parent-Child ลดหลั่นลงมาเป็น ลำดับชั้น (Hierarchy) ก็จะทำให้เกิดความซ้ำซ้อนในการเก็บสำเนาข้อมูลมาขึ้นตามไปด้วย

3 Sibling Relation

เป็นระบบที่มีเว็บแคชเซิร์ฟเวอร์มากกว่าหนึ่งตัวประกอบกันขึ้นเพื่อรองรับการร้องขอใช้บริการของไคลเอนต์โดยเครื่องเว็บแคชเซิร์ฟเวอร์แต่ละตัวจะทำการร้องถามไปยังเว็บแคชเซิร์ฟเวอร์ที่เป็น Sibling ของมันว่ามีข้อมูลที่ต้องการเก็บไว้หรือไม่พร้อมทั้งตรวจสอบเวลาในการไปร้องขอข้อมูลที่อยู่ที่เครื่องเซิร์ฟเวอร์ต้นทางจากตัวเองเปรียบเทียบกับจากเว็บแคชเซิร์ฟเวอร์ที่เป็น Sibling ของมันเพื่อใช้ในการตัดสินใจ เครื่องเว็บแคชเซิร์ฟเวอร์ที่มีความสัมพันธ์แบบ Sibling นี้จะถือว่ามีความสัมพันธ์อยู่ในระดับ (Level) เดียวกันและสามารถปรับตั้งได้ว่าในกรณีที่เว็บแคชเซิร์ฟเวอร์ตัดสินใจร้องขอข้อมูลจาก Sibling ของมันแล้วจะให้ทำสำเนาหรือไม่ทำสำเนาข้อมูลนั้นเก็บลงในดิสก์แคชของตัวเองก็ได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

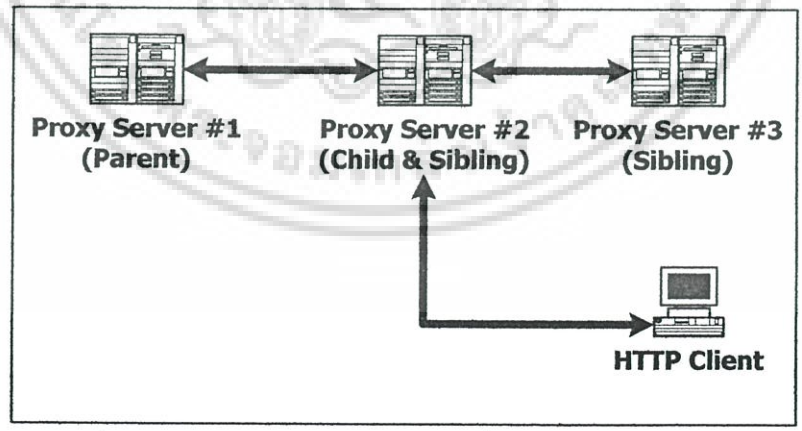


รูปที่ 2.3 แสดงระบบเว็บแคชเซิร์ฟเวอร์ที่มีความสัมพันธ์แบบ Sibling

ระบบเว็บแคชเซิร์ฟเวอร์ที่มีความสัมพันธ์แบบนี้เป็นตัวอย่างของ Cooperate Web-Cache โดยที่เราสามารถปรับตั้งให้เว็บแคชเซิร์ฟเวอร์ที่มีความสัมพันธ์กันแบบ Sibling นี้ไม่ต้องทำการสำเนาข้อมูลที่ร้องขอมาได้จาก Sibling ของตัวเอง (ไม่ใช่ร้องขอมาได้จากเซิร์ฟเวอร์ต้นทาง) ซึ่งจะไม่ทำให้เกิดความซ้ำซ้อนของการจัดเก็บข้อมูล

4 Compound Relation

เป็นระบบที่ประกอบไปด้วย เว็บแคชเซิร์ฟเวอร์ จำนวนมาก ที่มีความสัมพันธ์กันทั้งแบบ Parent-Child และ Sibling ซึ่งสามารถจะรองรับปริมาณการร้องขอใช้บริการของ HTTP Client ได้สูงและมีความอ่อนตัวในการปรับแต่ง Proxy แต่ละตัวเพื่อทำสมดุลภาระงาน (Load-balancing) ซึ่งสามารถเพิ่มจำนวนของเครื่องเว็บแคชเซิร์ฟเวอร์ได้เมื่อมีความต้องการเพิ่มมากขึ้น ตลอดจนสามารถใช้แนวคิดของ Cooperate Web-Cache เพื่อละความซ้ำซ้อนของการเก็บข้อมูลได้อีกด้วย



รูปที่ 2.4 แสดงระบบเว็บแคชเซิร์ฟเวอร์ที่มีความสัมพันธ์แบบผสม

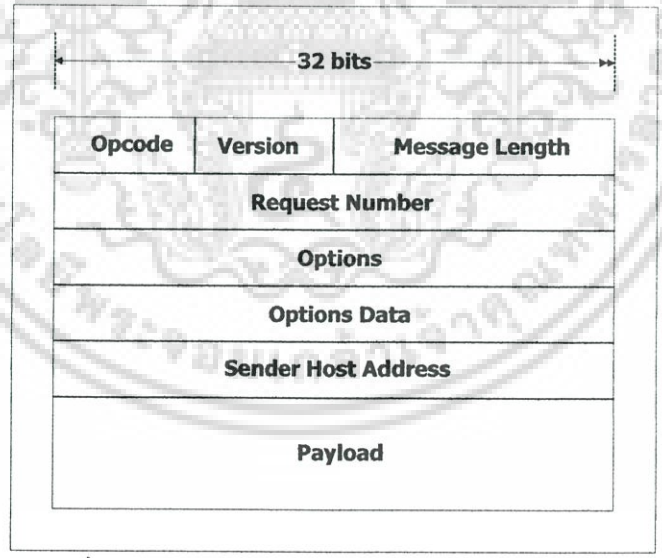
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในระบบที่มีเว็บแคชเซิร์ฟเวอร์ทำงานร่วมกันตั้งแต่ 2 เครื่องขึ้นไปจะมีการใช้โพรโตคอลที่ใช้สื่อสารและจัดการระหว่างแคช เพื่อสื่อสารข้อมูลระหว่างแคชเพื่อตรงสอบค่าต่าง ๆ ภายในกลุ่มของเว็บแคช เพื่อเพิ่มประสิทธิภาพ

โพรโตคอล ICP [6] ที่ใช้ในการสื่อสารระหว่างเว็บแคชเซิร์ฟเวอร์ เพื่อแลกเปลี่ยนข้อมูลเกี่ยวกับ URLs ที่มีอยู่ในเว็บแคชเซิร์ฟเวอร์ใกล้เคียงโดยจะเป็นการแลกเปลี่ยน ICP message (ICP-Query & ICP-Reply) เพื่อรวบรวมข้อมูลที่ใช้ในการร้องขอข้อมูลจากเว็บแคชเซิร์ฟเวอร์ที่อยู่ใกล้เคียง โดยเว็บแคชเซิร์ฟเวอร์จะส่ง ICP-Query ไปยังเว็บแคชเซิร์ฟเวอร์ที่อยู่ใกล้เคียง เมื่อได้รับแล้วจึงตอบกลับมาด้วย ICP-Reply เพื่อตอบว่า พบ (HIT) หรือไม่พบ (MISS) ข้อมูลที่ร้องขอนั้น โดยการส่ง ICP-message นี้ จะส่งโดยเลือกใช้บริการจาก TCP หรือ UDP ก็ได้ แต่การใช้บริการจาก UDP จะมีความเหมาะสมมากกว่าเพราะไม่มีการสร้าง Connection จึงทำงานเร็วกว่า TCP หากเกิดการสูญหายของ ICP-message ก็หมายความว่าเกิดความหนาแน่นหรือความเสียหายของเครือข่ายที่ใช้ติดต่อไปยังแคชที่อยู่ใกล้เคียง ในกรณีนี้จึงไม่ควรเลือกเว็บแคชเซิร์ฟเวอร์ตัวนี้ในรับข้อมูลที่ต้องการ นอกจากนี้ TCP ยังมี Overhead ที่ใหญ่กว่า UDP

รูปแบบ ICP message

ICP Message จะประกอบด้วย 2 ส่วนคือ Header ขนาด 20 Octet และ Payload ซึ่งมีขนาดเปลี่ยนแปลงได้ดังรูปที่ 2.5



รูปที่ 2.5 แสดงรายละเอียดรูปแบบของ ICP Message

Opcode: เป็นส่วนที่ใช้บอกประเภทของ ICP-message โดยมีค่า Opcode ที่ใช้ทั่วไปดังนี้คือ ICP_QUERY, ICP_MISS, ICP_HIT เป็นต้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Version: เป็นส่วนที่ใช้ระบุ version number ของ ICP protocol เพื่อใช้ตรวจสอบ โดยในปัจจุบันมีใช้งาน Version 2 และ 3

Message Length: เป็นส่วนที่ใช้บอกความยาวทั้งหมดของ ICP-message โดยจะมีความยาวไม่เกิน 16384 Octets

Request number: เป็นส่วนที่ใช้ Identify เมื่อมีการตอบ Query ค่านี้จะถูกเก็บลงใน Reply-message ด้วย

Options: มีความยาว 4 Octet เป็น Option flag ที่จะร่วมกับ Opcode เพื่อจุดประสงค์ต่าง ๆ เช่น ICP_FLAG_HIT_OBJ จะใช้ร่วมกับ ICP_QUERY เพื่อยินยอมให้ตอบกลับด้วยข้อมูลเลยหากขนาดของข้อมูลไม่เกินความยาวของคำตอบ (Reply)

Options Data: มีขนาด 4 octet ใช้รองรับการทำงานของ Optional feature ซึ่ง ICP-feature อาจจะต้องใช้ข้อมูลในฟิลด์นี้

Sender Host Address: เป็นส่วนที่ใช้เก็บ IPv4 Address ของ Host ที่ส่ง ICP-Message แต่อย่างไรก็ตามก็มี Address ที่มีพร้อมกับ Socket API อยู่แล้วซึ่งเป็นการซ้ำซ้อน ทำให้ไม่เป็นที่นิยมในการใช้งาน

Payload: เป็นส่วนของเนื้อหาซึ่งมีขนาดไม่แน่นอนโดยจะขึ้นกับ Opcode แต่โดยปรกติแล้วจะเป็นค่า Null-terminated URL String

2.3 Squid เว็บแคชเซิร์ฟเวอร์

Squid คือ Internet Proxy Caching application ซึ่งมีจุดเริ่มต้นมาจากการพัฒนาโปรแกรมของ Harvest Project ซึ่งไม่ได้มีจุดมุ่งหมายหลักในการพัฒนาแคช โดยได้รับเงินทุนสนับสนุนงานวิจัยจาก National Laboratory of Network Research (NLANR) Squid เป็นแอพพลิเคชั่นที่เผยแพร่ Source Codeโดยไม่คิดค่าใช้จ่าย จึงมีผู้นำ Squid ไปพัฒนาเพิ่มความสามารถอื่น ๆ และแก้ไขข้อผิดพลาดที่เกิดขึ้นอย่างแพร่หลายซึ่งปัจจุบันได้พัฒนาถึงเวอร์ชั่นที่ 2.5 จากเหตุผลที่กล่าวมาทำให้ Squid เป็นที่นิยมเพราะผู้ที่มีความรู้ทางด้านการเขียนโปรแกรมสามารถจะสามารถปรับปรุงประสิทธิภาพของมันตามความต้องการได้โดย Squid มีความสามารถในการทำ Proxying และ Caching โดยใช้โพรโตคอล HTTP, FTP และ URL แบบอื่น ๆ เช่น SSL และสามารถจะทำงานแบบ Cache Hierarchies โดยใช้โพรโตคอล ICP, HTCP, CRAP, Cache-digest ในการสื่อสารระหว่างกัน การที่เว็บแคชเซิร์ฟเวอร์จะสามารถทำงานได้อย่างมีประสิทธิภาพได้นั้น จะต้องประกอบด้วยประสิทธิภาพที่ดีของระบบโดยรวม ซึ่งองค์ประกอบที่สำคัญต่อประสิทธิภาพของเว็บแคชเซิร์ฟเวอร์ คือ เวลาที่ใช้ในการเข้าถึงข้อมูลในดิสก์ หน่วยความจำทั้งหมดของระบบ Disk-Throughput และความสามารถในการประมวลผลของ CPU

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การที่จะทำให้เว็บแคชเซิร์ฟเวอร์มีเสถียรภาพนั้นผู้ดูแลระบบควรจะมีการเตรียมพร้อมเพื่อรับมือกับปัญหาที่จะเกิดขึ้นไม่ว่าจะเป็นการเตรียมอะไหล่ที่จำเป็นในกรณีฉุกเฉินหรือการเตรียมระบบสำรองที่ติดตั้งระบบปฏิบัติการและ Squid เอาไว้เมื่อเกิดปัญหากับเครื่องเว็บแคชเซิร์ฟเวอร์หลักก็จะสามารถนำเครื่องสำรองมาใช้งานแทนที่ได้ทันที การตัดสินใจเลือกระบบที่จะนำมาใช้ร่วมกับ Squid นั้นมีองค์ประกอบหลาย ๆ อย่างที่ต้องคำนึง ไม่ว่าจะเป็นค่าปริมาณสูงสุดของการร้องขอต่อนาที (Peak number of Request per minute) ซึ่งจะแสดงให้เห็นจำนวนของข้อมูลที่ถูกร้องขอจากไคลเอนต์และสามารถนำมาคำนวณหาภาระงานที่แคชจะต้องแบกรับขณะทำงาน เป็นการยากที่จะคำนวณค่าการร้องขอสูงสุดนี้ ขึ้นอยู่กับลักษณะพฤติกรรมการใช้งานอินเทอร์เน็ตของผู้ใช้ เช่นเดียวกับการตัดสินใจเลือก ฮาร์ดแวร์ ที่เหมาะสม หากไม่มีสถิติการใช้งานอินเทอร์เน็ต เราอาจจะติดตั้งเว็บแคชเซิร์ฟเวอร์ ทดสอบเพื่อประเมิน การร้องขอ ที่จะเกิดขึ้นก็ได้ จะเป็นการดี หากเราประเมินฮาร์ดแวร์ ที่เราต้องการมากกว่า ความต้องการใช้งานในปัจจุบัน เพื่อให้สามารถรองรับความต้องการที่จะเพิ่มขึ้นตามมาในอนาคต

Squid สามารถทำงานร่วมกับ ระบบปฏิบัติการ Unix ได้เกือบทุก Platform เนื่องจาก Squid เขียนบน Digital Unix ที่ทำงานกับ GNU C compiler จึงสามารถนำ Squid ไปติดตั้งบน OS ที่มี compiler ดังกล่าว โดยการติดตั้งจะเลือกนำ Source code มา compile หรือนำ Binary Version มาติดตั้ง ซึ่งสามารถกระทำได้ง่ายกว่า แต่อาจจะมีปัญหาเรื่องความปลอดภัยที่เกิดจากการทำงานของ โปรแกรมย่อยต่าง ๆ ที่แอบแฝงอยู่ Squid โดยผู้ไม่หวังดีได้ และในปัจจุบันได้มีการปรับแต่งให้สามารถนำ Squid มาใช้งานได้บน Platform ของ Windows โดยใช้ชื่อว่า SquidNT ซึ่งยังใช้รูปแบบการปรับแต่งตัว Squid เป็นแบบการแก้ไขไฟล์คอนฟิกกุเรชั่น เช่นเดียวกับที่ใช้บน Platform Unix

ส่วนประกอบของ Squid มีส่วนประกอบที่สำคัญโดยทั่วไปดังนี้

1. Client side: เป็นส่วนที่ใช้รับการเชื่อมต่อจาก ไคลเอนต์ แล้วทำการตีความการร้องขอที่ได้รับ และประมวลผลการร้องขอนั้น ๆ ผลของการประมวลผลตรงส่วนนี้ก็จะระบุผลของการร้องขอที่มีเข้ามาด้วยว่ามีผลการทำงานเป็นแบบใด เช่น HIT MISS หรือ REFRESH เป็นต้น ซึ่งจะมีการเก็บข้อมูลของสถานะของแต่ละการร้องขอที่ประมวลผลเสร็จสิ้นไว้เพื่อใช้งานต่อไป เช่นการเก็บบันทึกลงไฟล์ Log

2. Server side: มีหน้าที่ส่งผ่านการร้องขอจากไคลเอนต์ที่มีผลของการร้องขอว่าไม่พบข้อมูลในแคช (Cache MISS) ไปยังเครื่องเซิร์ฟเวอร์อื่นโดยขึ้นอยู่กับประเภทของโพรโตคอลที่ไคลเอนต์ร้องขอเข้ามาอีกด้วย การส่งผ่านการร้องขอนี้อาจจะส่งผ่านไปยังเว็บเซิร์ฟเวอร์ต้นทางที่ไคลเอนต์ต้องการร้องขอข้อมูลหรืออาจจะเป็นเครื่องเว็บแคชเซิร์ฟเวอร์อื่น ๆ ก็ได้ ทั้งนี้แล้วแต่การปรับตั้งค่าของเว็บแคชเซิร์ฟเวอร์ว่าให้มีการทำงาน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ร่วมกับเว็บแคชเซิร์ฟเวอร์เครื่องอื่น ๆ ด้วยหรือไม่อย่างไร โดยที่ทุก ๆ การร้องขอจากฝั่งเซิร์ฟเวอร์ไปยังเครื่องเซิร์ฟเวอร์อื่น ๆ ดังกล่าวมาแล้วนั้นจะเป็นการส่งผ่านในรูปแบบของโพรโตคอล HTTP เท่านั้น แม้ว่าการร้องขอเริ่มต้นจากทางไคลเอนต์นั้นจะเป็นการร้องขอด้วยโพรโตตอล FTP หรือ Gopher ก็ตาม โพรโตคอล WAIS และ Gopher นั้นไม่ค่อยได้รับความสนใจมากนักเนื่องจากในปัจจุบันมันมีสัดส่วนของปริมาณของข้อมูลและการร้องขอผ่านอินเทอร์เน็ตที่น้อยมากเมื่อเทียบกับ HTTP

3. Storage Manager: ส่วนนี้จะเป็นส่วนเชื่อมประสานระหว่าง client side และ server side โดยจะทำหน้าที่จัดการกับข้อมูลที่เก็บอยู่ในแคช ทั้งการค้นหาข้อมูลจากแคชเมื่อถูกร้องขอ และการจัดการแทนที่ข้อมูลเก่าด้วยข้อมูลที่ใหม่กว่าในกรณีที่ไม่มีเนื้อที่ดิสก์เหลือพอที่จะเก็บข้อมูลใหม่โดยอาศัย อัลกอริธึมการแทนที่ข้อมูล (Replacement Algorithm) แบบต่าง ๆ เช่น Least-Recently-Used (LRU) เป็นต้น

4. Request Forwarding: เป็นส่วนที่จะทำหน้าที่ส่งผ่านการร้องขอจากไคลเอนต์ไปยังเครื่องเว็บเซิร์ฟเวอร์อื่นหรือเว็บแคชเซิร์ฟเวอร์อื่นในบางกรณีที่มีการกำหนดไว้ล่วงหน้าแล้วที่ตัว Squid เช่น ให้ทำการส่งผ่านการร้องขอของไคลเอนต์ที่ต้องการร้องขอข้อมูลจากเว็บเซิร์ฟเวอร์ที่อยู่ภายในโดเมนเดียวกัน ไปยังเว็บเซิร์ฟเวอร์นั้นโดยตรงเลย โดย Squid จะไม่ทำการร้องขอข้อมูลดังกล่าวให้เนื่องจากถือว่าการติดต่อภายในโดเมนเดียวกันนั้นสามารถทำได้อย่างรวดเร็วและมี Bandwidth ระหว่างเครื่องไคลเอนต์และเว็บเซิร์ฟเวอร์ต้นทางที่มากพอ (เช่นอาจจะอยู่บน campus network เดียวกัน)

1. Peer Selection: ทำหน้าที่ตัดสินใจในการเลือกหรือไม่เลือก เว็บแคชเซิร์ฟเวอร์ที่เหมาะสมที่จะทำการส่งผ่านการร้องขอออกไปในกรณีที่ไม่พบข้อมูลในแคชของตัวเอง

2. Access Control: ทำหน้าที่ในการรับหรือปฏิเสธการร้องขอจากไคลเอนต์ที่มีเข้ามาโดยสามารถพิจารณาจาก IP Address ของไคลเอนต์ ชื่อเครื่องเว็บเซิร์ฟเวอร์ที่เก็บข้อมูลที่ไคลเอนต์ ต้องการร้องขอ หรือ วิธีการในการร้องขอ (Request Method) โดย Squid สามารถรองรับการทำ Access Control List ได้หลายอันพร้อมกันโดยจะทำงานเรียงตามลำดับไปเรื่อย ๆ จนครบหมดทุก List

3. Network Communication: จะเป็นส่วนที่ทำหน้าที่ในการติดต่อสื่อสารผ่านเครือข่าย โดยสามารถติดต่อสื่อสารได้ทั้งในแบบ TCP และ UDP

4. File/Disk I/O: เป็นส่วนที่ทำหน้าที่ในการติดต่อกับดิสก์ในการอ่าน และเขียนข้อมูล โดยมันสามารถที่จะรวมความต้องการในการเขียนดิสก์จากหลาย ๆ การร้องขอเป็นการสั่งให้มีการเขียนดิสก์เพียงครั้งเดียวได้ เพื่อเป็นการเพิ่มประสิทธิภาพในการทำงานโดยลดการเขียนข้อมูลขนาดเล็ก ๆ หลาย ๆ ครั้งลงเป็นการเขียนข้อมูลขนาดใหญ่ขึ้นด้วย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จำนวนครั้งที่น้อยลง อันเป็นการลดการขัดจังหวะการทำงานของหน่วยประมวลผลกลาง และเพิ่มประสิทธิภาพในการใช้งานดิสก์

5. Neighbor: ทำหน้าที่เกี่ยวกับการเก็บรายชื่อของ เว็บแคชเซิร์ฟเวอร์ที่ถูกกำหนดให้ทำงานร่วมกับตัวมัน ตลอดจนทำการรับ-ส่ง ICP Message ไปยังเครื่องเว็บแคชเซิร์ฟเวอร์ทั้งหมดที่อยู่ในรายชื่อนี้เพื่อใช้ประกอบการตัดสินใจเลือกเว็บแคชเซิร์ฟเวอร์ในการที่จะทำการร้องขอข้อมูลในกรณีที่ไม่พบ ข้อมูลที่ไคลเอนต์ร้องขอในแคชของตัวเอง

6. IP/FDQN: ทำหน้าที่เก็บและดูแลตารางความสัมพันธ์ระหว่าง ชื่อเครื่องเซิร์ฟเวอร์ที่ให้บริการกับ IP Address ของเครื่องนั้น เพื่อที่จะสามารถนำมาใช้ได้โดยไม่ต้องส่งการร้องขอไปยังเครื่องที่ให้บริการ DNS ซึ่งจะสามารถลดเวลาตรงนี้ลงไปได้ส่วนหนึ่ง

7. Cache Manager: ทำหน้าที่ในการจัดการกับตัว Squid ซึ่งเป็นเว็บแคชเซิร์ฟเวอร์อยู่โดยสามารถที่จะทำการจัดการและตรวจสอบข้อมูลจาก Squid หลาย ๆ เครื่องได้จากที่เดียวโดยผ่าน cachemgr.cgi ซึ่งมีหลักการทำงานคือตัว โปรแกรม cachemgr.cgi จะทำการส่ง URL ของข้อมูลที่ต้องการในรูปของ cache_object://hostname/operation ไปยังเครื่องเว็บแคชเซิร์ฟเวอร์ Squid ที่ต้องการจะตรวจสอบข้อมูล โดยจะทำได้เพียงการอ่านข้อมูลอย่างเดียวเท่านั้น ไม่สามารถที่จะทำการปรับตั้งค่าพารามิเตอร์ใด ๆ ได้ในขณะที่ Squid เป้าหมายนั้นทำงานอยู่

8. Network Measurement Database: จะทำหน้าที่ในการเก็บและดูแลข้อมูล Round-Trip Time (RTT) ของตัว Squid กับเว็บแคชเซิร์ฟเวอร์อื่นข้างเคียงไปยังเครื่องเซิร์ฟเวอร์ต้นทาง เพื่อใช้ในการเปรียบเทียบเวลาว่าใครอยู่ใกล้เครื่อง เว็บแคชเซิร์ฟเวอร์ต้นทางมากกว่ากัน เพื่อที่จะตัดสินใจเลือกว่าจะทำการร้องขอ ข้อมูลเครื่องเซิร์ฟเวอร์ใด การวัดค่า RTT นี้สามารถทำได้โดยใช้โปรแกรม pinger ส่ง ICMP Packet ไปยังเครื่องปลายทางแล้ววัดเวลาที่ใช้ตั้งแต่เริ่มส่งจนได้รับการตอบกลับมาจากเครื่องปลายทาง

9. Redirector: ส่วนนี้มีความสามารถในการเขียนทับการร้องขอจากไคลเอนต์หลังจากที่ตรวจสอบพบว่ามีเงื่อนไขของการร้องขอตรงกับที่ระบุไว้ใน Access Control List โดยการร้องขอนี้จะถูกส่งออกไปยังแอพพลิเคชั่นตัวอื่นหรือแม้กระทั่งเว็บแคชเซิร์ฟเวอร์ตัวอื่นก็ได้ การทำ redirection นี้จะช่วยทำให้การใช้งานเว็บแคชเซิร์ฟเวอร์มีความหลากหลายและความยืดหยุ่นมากยิ่งขึ้น

10. Configuration File Parsing: เป็นส่วนที่ทำหน้าที่ตีความพารามิเตอร์ต่าง ๆ ที่ปรากฏอยู่ในไฟล์ squid.conf ซึ่งเป็นไฟล์ที่เราใช้กำหนดค่าเริ่มต้นและพารามิเตอร์ในการทำงานต่าง ๆ ของ Squid

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

11. Callback Data Database: เป็นส่วนที่ใช้จัดการฐานข้อมูลที่มีไว้เก็บและการเข้าถึงข้อมูลในหน่วยความจำหลักของเครื่องอันเนื่องมาจากการเรียกใช้ฟังก์ชั่น callback ซึ่ง Squid จะเรียกใช้บ่อยครั้งมากโดยจะเกิดขึ้นทุกครั้งที่มีการเขียนข้อมูลผ่านเครือข่าย (ในการเขียนข้อมูลลงดิสก์ก็เกิด callback ขึ้นเช่นกันหากแต่อาจจะไม่เกิดทุกครั้งเนื่องจากการทำงานในส่วนของ File/Disk I/O นั้นสามารถรวมการเขียนข้อมูลขนาดเล็กหลายครั้งเป็นการเขียนข้อมูลลงดิสก์จริงเพียงครั้งเดียวได้) การทำงานในส่วนนี้จะจัดเตรียมวิธีการในการจัดการกับข้อมูลที่เกี่ยวข้องกับการทำ callback ในหน่วยความจำ การยกเลิกการทำ callback และ การป้องกันความผิดพลาดอันเกิดจากความพยายามในการเข้าถึงข้อมูลจากหน่วยความจำหลัก

12. Debugging: เป็นส่วนที่ช่วยรายงานผลการทำงานเพื่อใช้ในการติดตามการทำงานของ Squid การตรวจหาสาเหตุของความผิดพลาดในการทำงานของ Squid โดยสามารถเลือกระดับของการแสดงผลการทำงานได้หลายระดับ

13. Error Generation: ทำหน้าที่ในการสร้างข้อความแสดงความผิดพลาด (Error Message) จากรูปแบบและพารามิเตอร์ที่กำหนด ซึ่งทำให้สามารถทำการปรับแต่งการแสดงความผิดได้ นอกจากนี้ยังสนับสนุนหลายภาษาอีกด้วย (Multilingual Support)

14. Event Queue: ทำหน้าที่เกี่ยวกับการจัดการเหตุการณ์ต่าง ๆ ที่เกิดขึ้นทุกช่วงระยะเวลา เช่นการทำ Cache Replacement การทำลบไฟล์ใน Swap Directory นอกจากนี้ยังสามารถใช้ได้กับการทำงานของฟังก์ชั่นที่ทำเพียงครั้งเดียว (one-time function) เช่น ICP query timeout

15. Filedescriptor Management: ทำหน้าที่ในการติดตามการใช้ Filedescriptor ตลอดจนจำนวนข้อมูลที่อ่านหรือเขียนไปยัง Filedescriptor แต่ละอัน

16. Hash Table Support ทำหน้าที่ในการสร้าง ตาราง Hash ในการสร้างความสัมพันธ์แบบ Hash ของข้อมูลที่เก็บในแคช

17. HTTP Anonymization เป็นส่วนที่ทำหน้าที่เกี่ยวกับการอนุญาตหรือไม่อนุญาตให้ Squid ประมวลผลการร้องขอโดยโพรโตคอล HTTP ที่มีส่วนหัว (Header) ตรงตามที่กำหนดไว้ ซึ่งโดยปรกติแล้วจะไม่มีการเรียกใช้การทำงานในส่วนนี้ นั่นคือจะยอมให้ Squid ทำการประมวลการร้องขอโดยโพรโตคอล HTTP โดยไม่มีข้อแม้ใด ๆ

18. Internet Cache Protocol: เป็นส่วนที่ทำหน้าที่เกี่ยวกับ ICP ทั้งหมดดังที่ได้กล่าวถึงมาแล้วข้างต้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

19. Ident Lookups: เป็นส่วนที่ทำหน้าที่ในการร้องขอชื่อของผู้ใช้ในแต่ละการเชื่อมต่อที่มีเข้ามายัง Squid ซึ่งในบางที่ (Site) สามารถนำไปใช้ในการตรวจสอบสิทธิในการเข้าใช้งาน และการติดตามพฤติกรรมในการใช้งานเครื่องเซิร์ฟเวอร์ของที่นั่นได้อีกด้วย

20. Memory Management: ทำหน้าที่ในการกำหนดและจัดการเนื้อที่ในหน่วยความจำหลักสำหรับใช้ในการเก็บข้อมูลหรือข้อมูลที่ถูกเรียกใช้งานบ่อย ๆ เมื่อมีการเปิดการทำงาน memory_pool ใน Squid.conf พื้นที่ในหน่วยความจำที่ยังไม่มีการนำข้อมูลเข้าไปเก็บจะไม่ถือว่าเป็นพื้นที่ว่าง แต่จะถูกการทำงานในส่วนของ Memory Management กันที่จองเอาไว้สำหรับการใช้งานที่จะเกิดขึ้นในอนาคตต่อไป ซึ่งเป็นการใช้งานหน่วยความจำหลักอย่างไม่มีประสิทธิภาพนัก

21. Multicast support: การทำงานในส่วนนี้ในปัจจุบันจะใช้รองรับการส่ง ICP-query เท่านั้นโดยจะเชื่อม UDP socket เข้ากับ Multicast group และทำการกำหนดค่า TTL สำหรับการส่งแบบ multicast (ซึ่งจะมีค่าน้อยกว่าค่า TTL ของการส่งแบบ Unicast และ Broadcast)

22. Persistent Server Connections: จะทำหน้าที่จัดการกับการเชื่อมต่อแบบ Persistent ระหว่าง Squid ไปยังเครื่องเซิร์ฟเวอร์ต้นทางหรือ เว็บแคชเซิร์ฟเวอร์ใกล้เคียง โดยจะมีการเก็บค่าดัชนีของการเชื่อมต่อแต่ละการเชื่อมต่อไว้ใน Hash Table โดยการเก็บหมายเลข IP Address และ พอร์ต โดยที่ในแต่ละ socket address (IP Address + port) จะเก็บการเชื่อมต่อแบบ Persistent ได้มากที่สุด 10 การเชื่อมต่อ ทั้งนี้จะทำการรักษาการเชื่อมต่อดังกล่าวไว้เป็นเวลาไม่เกิน 15 วินาที หากไม่มีการใช้งานการเชื่อมต่อแบบ Persistent นั้นเกิน 15 วินาทีก็จะทำการปิดการเชื่อมต่อนั้นทันที

23. Refresh Rules: ทำหน้าที่ในการตัดสินใจว่าข้อมูลที่ถูกเก็บอยู่ในดิสก์แคชนั้นยังใช้ได้หรือหมดอายุแล้ว โดยขึ้นอยู่กับพารามิเตอร์ Refresh_pattern ในไฟล์ squid.conf โดยในกรณีที่ข้อมูลที่ถูกร้องขอนั้นยังไม่หมดอายุ Squid ก็จะนำข้อมูลนั้นส่งให้กับไคลเอนต์ที่ร้องขอเข้ามา และถือว่าการร้องขอครั้งนี้เป็นการร้องขอประเภทที่พบข้อมูล HIT แต่ถ้าข้อมูลที่ถูกร้องขอเข้ามาถูกตัดสินว่าหมดอายุแล้ว Squid ก็จะต้องทำการตรวจสอบความถูกต้องของข้อมูลนั้นกับเซิร์ฟเวอร์ต้นทางด้วยการร้องขอแบบ If-Modify-Since

24. SNMP Support: เป็นส่วนที่ใช้สนับสนุนการทำงานร่วมกับโพรโตคอลSNMP โดยปัจจุบัน Squid สามารถให้ข้อมูลเกือบทั้งหมดที่สามารถหาได้จากโปรแกรม cachemgr ผ่านทาง SNMP ได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

25. URN Support: เป็นส่วนที่ทำหน้าที่ในการสนับสนุนและรองรับการระบุข้อมูลและประเภทของการบริการ แบบ Uniform Resource Name (URN) ซึ่งเป็นรูปแบบการระบุข้อมูลที่คล้ายคลึงกับ URL

2.3.1 ขั้นตอนการทำงานของ Squid เว็บแคชเซิร์ฟเวอร์

การทำงานของเว็บแคชเซิร์ฟเวอร์ในที่นี้จะหมายถึงการทำงานที่เกิดจากการสั่งงานของ Squid ซึ่งเป็นแอพพลิเคชั่นที่ทำหน้าที่เป็นพรอกซีอยู่บนตัวเครื่องเว็บแคชเซิร์ฟเวอร์ โดยแบ่งได้เป็นสองช่วง คือ ช่วงเริ่มต้นระบบ (System Initialization Phase) และช่วงให้บริการ (Service Phase)

2.3.1.1 ช่วงเริ่มต้นระบบ (System Initialize Phase)

ถ้าเป็นการเริ่มต้นการทำงานครั้งแรกตัวเว็บแคชเซิร์ฟเวอร์จะทำการเตรียมพื้นที่เก็บข้อมูลบน ดิสก์ โดยการจัดสร้างไดเรกทอรีและไดเรกทอรีย่อยตามที่ถูกตั้งค่าเอาไว้ แล้วจึงเข้าสู่ช่วงการให้บริการ

ถ้าเป็นการเริ่มต้นการทำงานครั้งต่อ ๆ ไปเว็บแคชเซิร์ฟเวอร์จะทำการตรวจสอบความถูกต้องของฐานข้อมูลที่เกี่ยวกับข้อมูลที่เก็บให้ในดิสก์แคชตลอดจนตรวจสอบและลบข้อมูลที่มีอายุ (ที่เก็บไว้ใน ดิสก์แคช) เกินที่ตั้งค่าเอาไว้ จากนั้นจึงเข้าสู่ช่วงการให้บริการ

2.3.1.2 ช่วงให้บริการ (Service Phase)

การทำงานจะเริ่มจาก

1. รับการร้องขอ: เมื่อเครื่องเว็บแคชเซิร์ฟเวอร์ได้รับ TCP Connection จากเครื่องไคลเอนต์ผ่านทางพอร์ตที่กำหนด (ส่วนมากจะเป็น พอร์ต 8080 ยกเว้น Squid ที่ใช้ พอร์ต 3128) โดยไคลเอนต์จะส่งการร้องขอเข้ามายังเว็บแคชเซิร์ฟเวอร์ซึ่งประกอบไปด้วย URL ของข้อมูลและวิธีที่ต้องการกระทำกับข้อมูลนั้น ๆ ตามมาตรฐานของ HTTP/FTP เช่น GET, POST, REFRESH เป็นต้น

2. ค้นหาข้อมูลในเมโมรีแคช: เมื่อได้รับ การร้องขอ เข้ามาเว็บแคชเซิร์ฟเวอร์จะทำการตรวจสอบข้อมูลที่ได้รับการร้องขอนั้นมีอยู่ในเมโมรีแคชหรือไม่ ถ้าพบ ข้อมูล ดังกล่าว (MEM_HIT) ก็จะทำการส่งข้อมูลนั้นให้กับไคลเอนต์แต่ถ้าไม่พบก็จะทำการค้นหาข้อมูลเหล่านั้นใน ดิสก์แคช ต่อไป

3. ค้นหาข้อมูลในดิสก์แคช : ในกรณีที่เว็บแคชเซิร์ฟเวอร์ไม่พบข้อมูลใน เมโมรีแคชก็จะทำการตรวจสอบข้อมูลที่ได้รับการร้องขอนั้นว่ามีเก็บอยู่ในดิสก์แคชของตัวเองหรือไม่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

52462

3.1 Request HIT: ถ้าข้อมูลดังกล่าวมีอยู่ในดิสก์แคชและมีอายุไม่เกินที่ตั้งค่าไว้ก็จะทำการอ่านข้อมูลขึ้นมาไว้ในเมโมรีแคชและทำการส่งข้อมูลดังกล่าวไปให้กับไคลเอนต์ที่ได้ร้องขอมา

3.2 Request MISS: ถ้าไม่พบข้อมูลในดิสก์แคช ตัวเว็บแคชเซิร์ฟเวอร์เองจะมีทางเลือกสองทางคือ

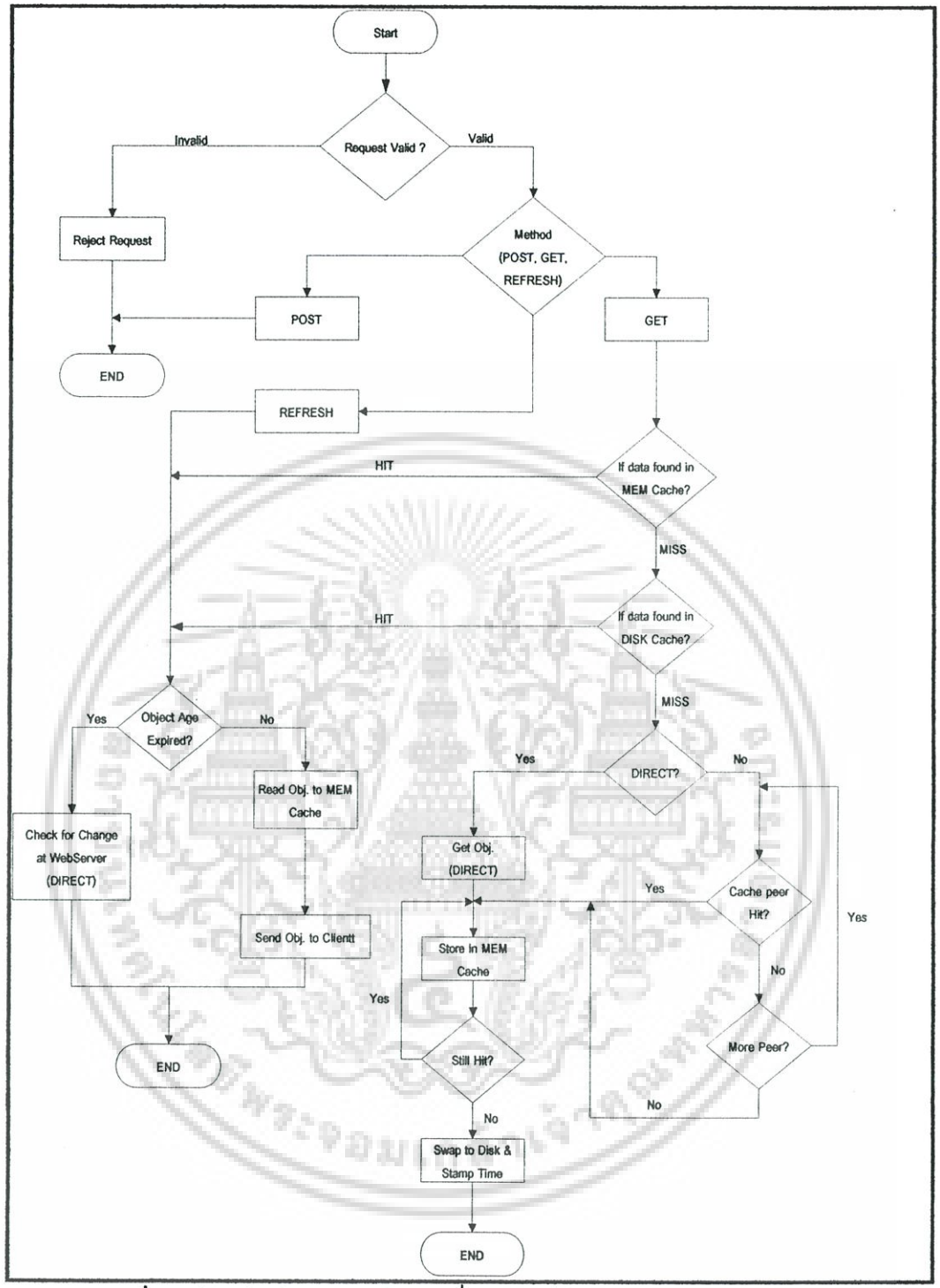
3.2.1 DIRECT: ตัวเว็บแคชเซิร์ฟเวอร์จะทำการติดต่อไปร้องขอ ข้อมูลนั้นจากเว็บเซิร์ฟเวอร์ตาม URL ที่ระบุไว้ในการร้องขอโดยตรง โดยเมื่อได้ข้อมูลดังกล่าวมาแล้วก็จะทำการจัดเก็บไว้ใน เมโมรีแคช และทำการส่งให้กับ ไคลเอนต์ที่ร้องขอมา

3.2.2 CACHE PEER: เว็บแคชเซิร์ฟเวอร์จะสร้างการติดต่อไปยังเว็บแคชเซิร์ฟเวอร์เครื่องอื่นในระบบโดยวิธีการที่จะใช้ขึ้นกับความสัมพันธ์ระหว่างเว็บแคชเซิร์ฟเวอร์คู่นั้น เช่นถ้าทั้งสองเป็น Sibling ซึ่งกันและกัน (อยู่ระดับเดียวกัน) ในกรณีของ Squid จะใช้โพรโตคอล ICP ผ่านทาง UDP พอร์ต 3130 แต่ถ้าเว็บแคชเซิร์ฟเวอร์นั้นกับมีความสัมพันธ์แบบ Parent-Child กับเว็บแคชเซิร์ฟเวอร์ที่จะร้องขอบริการเว็บแคชเซิร์ฟเวอร์ก็จะทำหน้าที่เหมือนกับเป็นไคลเอนต์ของเว็บแคชเซิร์ฟเวอร์ ที่มันไปขอใช้บริการ ซึ่งผลของการร้องขอก็จะมีทั้งพบข้อมูล (PARENT_HIT, SIBLING_HIT) และไม่พบ (PARENT_MISS, UDP_MISS) โดยถ้าพบ ข้อมูล ก็จะนำมาเก็บไว้ใน เมโมรีแคช แล้วจึงจัดส่งให้ ไคลเอนต์ ต่อไป แต่ถ้าไม่พบข้อมูลดังกล่าวก็จะทำติดต่อไปยังเว็บแคชเซิร์ฟเวอร์เครื่องต่อไปตามที่ตั้งค่าไว้จนครบ ในกรณีที่ไม่พบข้อมูลที่ต้องการจากเครื่องเว็บแคชเซิร์ฟเวอร์ทุกเครื่องที่ตั้งค่าไว้ก็จะทำการติดต่อไปยังเว็บเซิร์ฟเวอร์ตาม URL ของข้อมูลนั้น

4. เมื่อมีการค้นพบข้อมูล (HIT) ในเมโมรีแคชหรือดิสก์แคชและการสั่ง REFRESH โปรแกรม Squid จะตรวจสอบอายุของข้อมูลว่าอายุเกินกว่าค่าที่กำหนดไว้หรือยัง ในกรณีที่อายุของข้อมูลยังไม่เกินกว่าค่าที่กำหนด เว็บแคชเซิร์ฟเวอร์จะทำการนำข้อมูลไปไว้ที่เมโมรีแคชและส่งข้อมูลไปยังให้ไคลเอนต์ และในกรณีที่อายุของข้อมูลเกินกว่าค่าที่กำหนด เว็บแคชเซิร์ฟเวอร์จะติดต่อไปยังเว็บไซต์ต้นทางของข้อมูลเพื่อตรวจสอบว่าข้อมูลมีการเปลี่ยนแปลงหรือไม่ หากไม่เปลี่ยนแปลงก็จะนำข้อมูลที่มีอยู่ในแคชส่งไปให้ไคลเอนต์พร้อมกับปรับปรุงอายุของข้อมูลที่อยู่ในแคช หากข้อมูลมีการเปลี่ยนแปลงก็จะนำข้อมูลชุดใหม่มา แล้วส่งต่อให้ไคลเอนต์พร้อมกับเก็บลงไปในแคช

จากขั้นตอนการทำงานที่ได้อธิบายมาข้างต้น สามารถแสดงเป็นโฟลว์ชาร์ตได้ดังรูปที่ 2.6

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

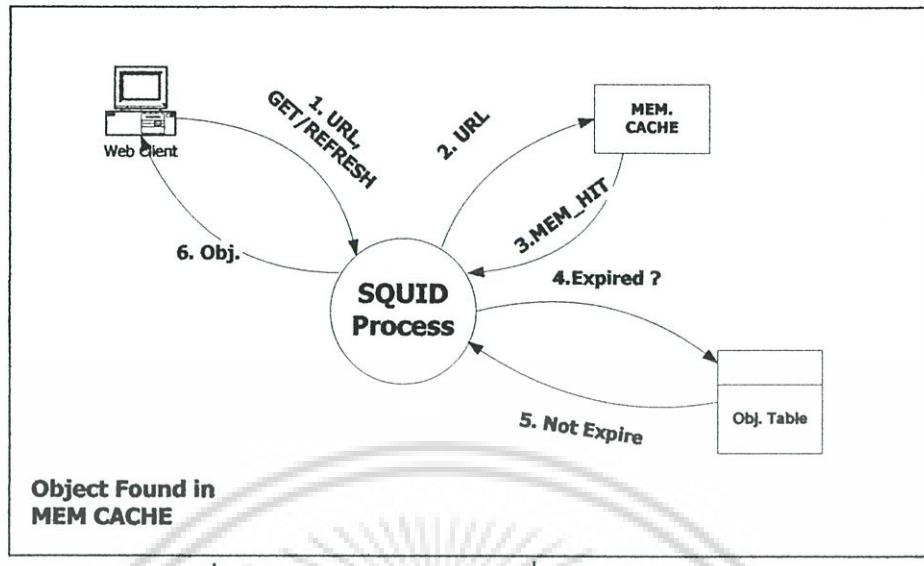


รูปที่ 2.6 โฟลว์ชาร์ตสรุปการทำงานเมื่อ Squid ได้รับการร้องขอจากไคลเอนต์

จากขั้นตอนการทำงานของ Squid ข้างต้นสามารถนำมาแยกผลของการร้องขอเพื่ออธิบายผลของการร้องขอแต่ละประเภทเพื่อแสดงถึงการไหลของพารามิเตอร์ที่เกี่ยวข้องรวมถึงองค์ประกอบในการทำงานต่าง ๆ ของ Squid โดยอาศัยแผนภาพดังต่อไปนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1. กรณีที่พบข้อมูลที่ร้องขอในเมโมรีแคช



รูปที่ 2.7 แสดงผลของการร้องขอที่พบข้อมูลในเมโมรีแคช

ในรูปที่ 2.7 การทำงานของ Squid ในกรณีที่พบข้อมูลในเมโมรีแคชจะเริ่มจากขั้นตอนที่

1. มีการร้องขอจากไคลเอนต์เข้ามา โดยไคลเอนต์จะส่ง URL ของข้อมูลและวิธีการที่ต้องการให้เว็บแคชเซิร์ฟเวอร์กระทำต่อข้อมูลนั้น เช่น สั่งให้นำมา (GET) สั่งให้ตรวจสอบการเปลี่ยนแปลง (REFRESH) ส่วนคำสั่งให้ส่งข้อมูลออกไป (POST) นั้นจำ Squid จะทำการส่งออกไปทันทีโดยไม่ได้จัดเก็บข้อมูลดังกล่าวไว้ในดิสก์แคช

2. เมื่อได้ URL มาแล้ว Squid จะทำการตรวจสอบว่ามีข้อมูลที่ระบุนั้นในเมโมรีแคชหรือไม่

3. ผลของการค้นหาปรากฏว่าพบข้อมูลในเมโมรีแคช

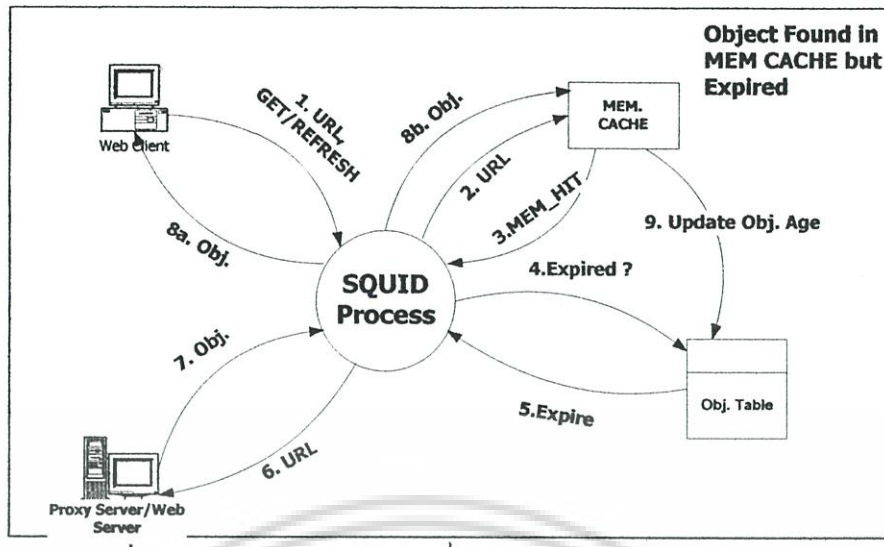
4. ทำการตรวจสอบว่าข้อมูลนั้นมีอายุในการเก็บไว้ในแคชเกินเวลาที่กำหนดไว้หรือไม่

5. ผลของการตรวจสอบพบว่าข้อมูลนั้นมีอายุในแคชยังไม่เกินเวลาที่กำหนด

6. Squid ทำการส่งข้อมูลนั้นไปให้ไคลเอนต์ตามคำร้องขอ

ในกรณีที่ผลของการตรวจสอบว่าข้อมูลที่พบในเมโมรีแคชนั้นมีอายุเกินกว่าที่กำหนดไว้ Squid ก็จะทำการร้องขอข้อมูลนั้นไปยังเซิร์ฟเวอร์ต้นทางตาม URL ของข้อมูลนั้นโดยตรง หรือถ้าหาก Squid ถูกกำหนดค่าให้มีการทำงานร่วมกับเครื่องเว็บแคชเซิร์ฟเวอร์เครื่องอื่นก็จะทำการร้องขอไปยังเว็บแคชเซิร์ฟเวอร์ดังกล่าวตามที่ได้กำหนดค่าเอาไว้ ซึ่งจะมีขั้นตอนเพิ่มตั้งแต่ขั้นตอนที่ 5 ดังแสดงในรูปที่ 2.8

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.8 แสดงผลของการร้องขอที่พบข้อมูลในเมโมรีแคชแต่หมดอายุ

5. ผลการตรวจสอบปรากฏว่าข้อมูลมีอายุที่เก็บในแคชเกินที่กำหนดไว้

6. Squid จะทำการร้องขอข้อมูลตาม URL ที่ระบุ หรือขอตรวจสอบว่าข้อมูลมีการเปลี่ยนแลงหรือไม่ไปยังเซิร์ฟเวอร์ต้นทางหรือ เว็บแคชเซิร์ฟเวอร์ที่ได้มีการกำหนดค่าไว้

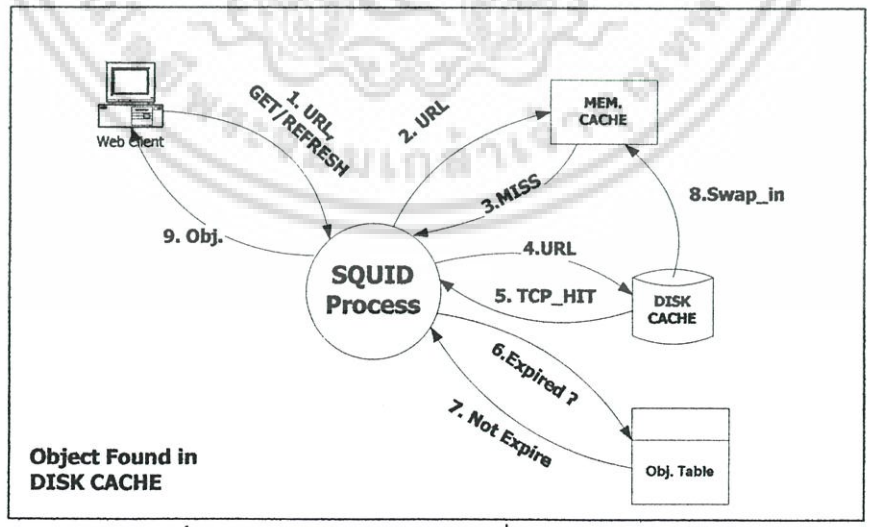
7. ได้รับข้อมูลจากเซิร์ฟเวอร์ที่ร้องขอไปในขั้นตอนที่ 6

8a. Squid จะทำการส่งข้อมูลไปให้กับไคลเอนต์ที่ร้องขอข้อมูลนั้น

8b. Squid ทำการทำสำเนาข้อมูลนั้นไปเก็บไว้ในเมโมรีแคช

9. ทำการปรับปรุง (Update) ค่าอายุของข้อมูลที่เก็บในแคช

2. กรณีที่ไม่พบข้อมูลในเมโมรีแคชแต่พบข้อมูลที่ร้องขอในดิสก์แคช



รูปที่ 2.9 แสดงผลของการร้องขอที่พบข้อมูลในดิสก์แคช

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในรูปที่ 2.9 การทำงานของ Squid ในกรณีที่พบข้อมูลในดิสก์แคชจะเริ่มจากรับการร้องขอจากไคลเอนต์เข้ามาและทำการตรวจสอบว่ามีข้อมูลดังกล่าวในเมโมรีแคชหรือไม่ (ขั้นตอนที่ 1. และ 2.) แต่กรณีนี้ไม่พบข้อมูลที่ต้องการ จึงมีขั้นตอนเพิ่มเติมดังต่อไปนี้

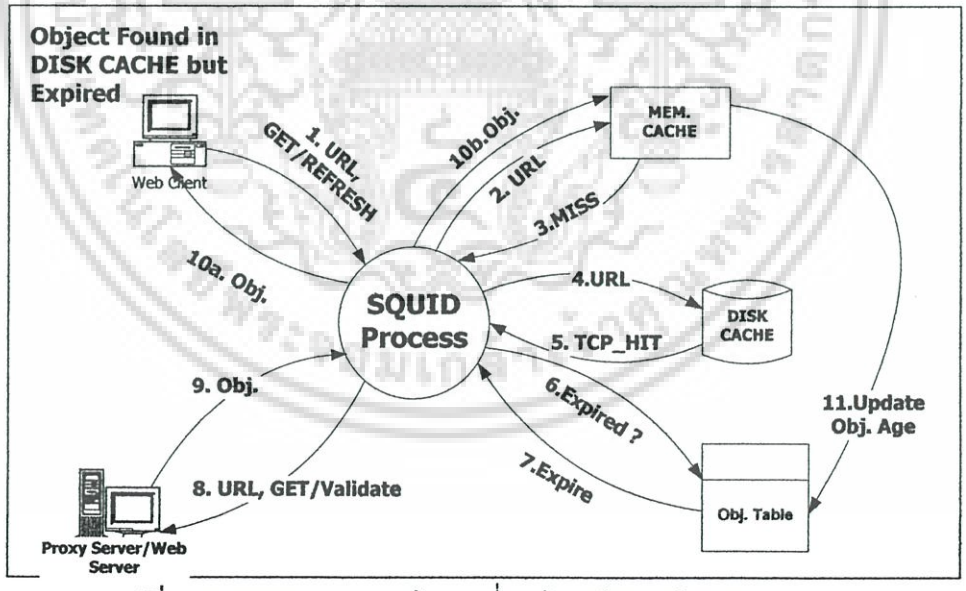
3. ผลการตรวจสอบปรากฏว่าไม่พบข้อมูลในเมโมรีแคช

4. Squid จะทำการตรวจสอบว่ามีข้อมูลตาม URL ในดิสก์แคชหรือไม่

5. ผลการตรวจสอบปรากฏว่าพบข้อมูลในดิสก์แคช

จากนั้น Squid ก็จะทำการตรวจสอบอายุของข้อมูลที่เก็บไว้ในแคชว่าเกินค่ากำหนดที่ตั้งไว้หรือไม่ ถ้าผลของการตรวจสอบปรากฏว่าข้อมูลยังมีอายุไม่เกินค่าที่กำหนด ก็จะทำการสำเนาข้อมูลนั้นไปเก็บไว้ในเมโมรีแคช (ขั้นตอนที่ 8) และส่งข้อมูลนั้นให้กับไคลเอนต์ที่ร้องขอเข้ามา (ขั้นตอนที่ 9)

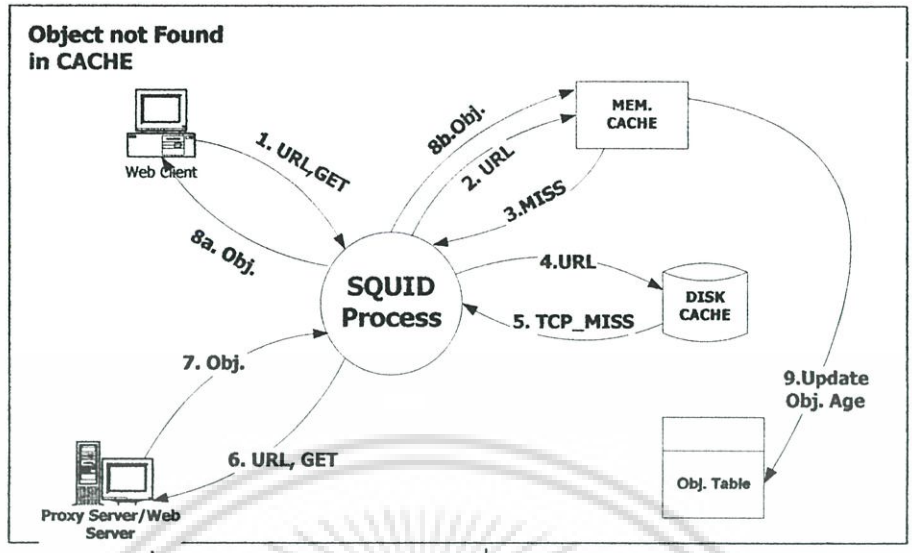
ในกรณีที่พบข้อมูลในดิสก์แคชแต่ปรากฏว่าข้อมูลนั้นมีอายุที่เก็บไว้ในแคชนานเกินกำหนดที่ตั้งไว้ ก็จะต้องเพิ่มขั้นตอนในการไปร้องขอข้อมูล (GET) หรือขอตรวจสอบว่าข้อมูลดังกล่าวมีการเปลี่ยนแปลงหรือไม่ (REFRESH) ดังกล่าว ดังแสดงไว้ในขั้นตอนที่ 8 และ 9 ในรูปที่ 2.5 ซึ่งเมื่อ Squid ได้รับการตอบรับกลับมาแล้วก็จะทำการส่งข้อมูลนั้นให้กับไคลเอนต์ที่ร้องขอเข้ามาและทำการปรับปรุงค่าอายุของข้อมูลนั้นที่เก็บไว้ในแคชใหม่



รูปที่ 2.10 แสดงผลของการร้องขอที่พบข้อมูลในดิสก์แคชแต่หมดอายุ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

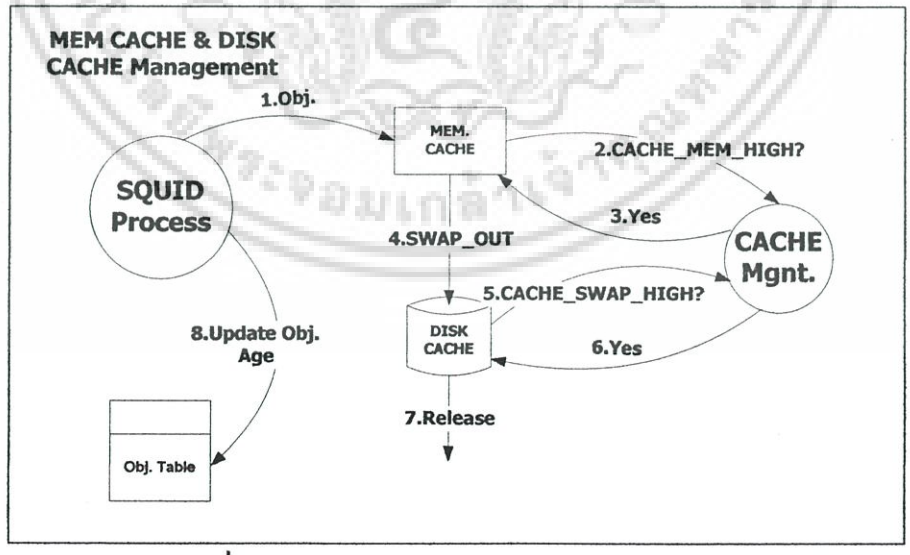
3. กรณีที่ไม่พบข้อมูลในเมโมรีแคชและดิสก์แคช



รูปที่ 2.11 แสดงผลของการร้องขอที่ไม่พบข้อมูลในแคชของ Squid

ในกรณีที่ข้อมูลที่ถูกร้องขอเข้ามานั้นไม่พบอยู่ทั้งในเมโมรีแคชและดิสก์แคชนั้น Squid ก็จะทำการร้องขอข้อมูลนั้นไปยังเซิร์ฟเวอร์ต้นทางหรือเว็บแคชเซิร์ฟเวอร์ที่ได้มีการตั้งค่าเอาไว้ ซึ่งการทำงานก็จะคล้ายกับกรณีที่ Squid เจอข้อมูลที่ถูกร้องขอในแคชของตัวเองแต่ปรากฏว่าข้อมูลนั้นหมดอายุ เพียงแต่ในขั้นตอนที่ 6 จากรูปที่ 2.11 จะเป็นการร้องขอแบบขอข้อมูล (GET) ไม่ใช่การตรวจสอบว่าข้อมูลนั้นมีการเปลี่ยนแปลงหรือไม่ (REFRESH)

ในการทำงานของ Squid จะมีการเก็บข้อมูลใหม่ที่ถูกร้องขอเข้ามาไว้ในแคชโดยจะมีลำดับขั้นตอนการทำงานและส่วนที่เกี่ยวข้องดังแสดงไว้ในรูปที่ 2.12



รูปที่ 2.12 แสดงการจัดเก็บข้อมูลในแคชของ Squid

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1. เมื่อ Squid ได้รับข้อมูลที่ถูกร้องขออันใหม่เข้ามาก็นำมาจัดเก็บไว้ในส่วนของเมโมรีแคช เป็นอันดับแรก

2. ถึง 4. เมื่อพื้นที่ที่ใช้ในการจัดเก็บข้อมูลของเมโมรีแคชมีค่ามากถึงค่าระดับที่ตั้งไว้ก็จะมีการย้ายข้อมูลจำนวนหนึ่งจากเมโมรีแคชลงไปเก็บไว้ในดิสก์แคช

5. ถึง 7. ในทำนองเดียวกันเมื่อพื้นที่ที่ใช้จัดเก็บข้อมูลของดิสก์แคชมีค่าถึงระดับที่ตั้งไว้ก็จะมีการลบข้อมูลจำนวนหนึ่งทิ้งไปเพื่อให้มีเนื้อที่เหลือพอสำหรับจัดเก็บข้อมูลที่จะเข้ามาใหม่

8. เมื่อมีการนำข้อมูลใหม่เข้ามาเก็บก็จะมีการเก็บค่าอายุที่ข้อมูลนั้นถูกเก็บไว้ในแคชด้วย

ในกรณีที่แคชเก็บข้อมูลจนถึงระดับที่ตั้งไว้ก็จะต้องมีกระบวนการเพื่อใช้ในการตัดสินใจที่จะทำการแทนที่ข้อมูลเก่าซึ่งมีอัลกอริธึมในการแทนที่ข้อมูลอยู่หลายวิธีด้วยกัน ซึ่งจะได้อธิบายอย่างละเอียดภายในบทที่ 3

2.3.2 ไฟล์ Squid.conf กับพารามิเตอร์ที่มีผลต่อการทำงานของเว็บแคชเซิร์ฟเวอร์

ไฟล์ Squid.conf เป็นไฟล์สำคัญที่ใช้เก็บค่าพารามิเตอร์ต่าง ๆ ที่ Squid ต้องใช้ในการทำงาน มีลักษณะเป็นไฟล์ตัวอักษรที่สามารถเข้าไปแก้ไขได้ด้วยโปรแกรมอิดิเตอร์ทั่วไป โดยมีพารามิเตอร์ที่สำคัญและมีผลต่อการทำงานในส่วนที่เกี่ยวข้องกับงานวิจัยดังต่อไปนี้

http_port เป็นการระบุหมายเลขพอร์ตให้ Squid ใช้ในการรอรับการร้องขอจาก HTTP ไคลเอนต์ ซึ่งการระบุนั้นสามารถทำได้ 3 แบบคือ 1 ระบุหมายเลขพอร์ตอย่างเดียว 2 ระบุชื่อเครื่องกับพอร์ต และ 3 ระบุ IP Address ของเครื่องกับพอร์ต ดังแสดงในตัวอย่างตามลำดับ

```
http_port 3218
http_port proxy.ce.kmitl.ac.th:3128
http_port 161.246.5.101:3128
```

icp_port เป็นการระบุหมายเลขพอร์ตให้ Squid ใช้ในการรับ-ส่ง ICP-Query กับเครื่องเว็บแคชเซิร์ฟเวอร์อื่น โดยสามารถทำการระบุได้ทั้ง 3 แบบเช่นเดียวกับ http_port ที่ได้แสดงไว้ข้างต้น

htcp_port เป็นการระบุหมายเลขพอร์ตให้ Squid สำหรับใช้ในการติด HTCP-Query กับเครื่องเว็บแคชเซิร์ฟเวอร์อื่น

cache_peer เป็นการระบุรายชื่อของพร็อกซีเซอร์เวอร์เครื่องอื่นที่จะกำหนดให้ทำงานร่วมกับ Squid โดยในการตั้งค่า cache_peer นี้จะมีพารามิเตอร์อื่นนอกจากชื่อของพร็อกซีเซอร์อีกดังรูปแบบต่อไปนี้

```
cache_peer hostname type http_port proxy_port options
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โดยที่

Hostname หมายถึงชื่อเครื่องเว็บแคชเซิร์ฟเวอร์ที่กำหนด

Type หมายถึงรูปแบบความสัมพันธ์ระหว่าง Squid กับเว็บแคชเซิร์ฟเวอร์ที่กำหนด โดยแบ่งเป็นความสัมพันธ์แบบ Parent หรือ Child

proxy_port หมายถึง หมายเลขพอร์ตของเว็บแคชเซิร์ฟเวอร์ที่กำหนดใช้รอรับการร้องขอ ซึ่งจะต้องเป็นหมายเลขเดียวกับ http_port ที่กำหนดไว้บนเว็บแคชเซิร์ฟเวอร์ที่กำหนดด้วย

icp_port หมายถึง หมายเลขพอร์ตของเว็บแคชเซิร์ฟเวอร์ที่กำหนดจะใช้รอรับการร้องขอ ICP-Query ซึ่งจะต้องเป็นหมายเลขเดียวกับ icp_port ที่กำหนดไว้บนเว็บแคชเซิร์ฟเวอร์ที่กำหนดด้วย

options จะเป็นค่าที่กำหนดรายละเอียดปลีกย่อยของการทำงานร่วมกับเว็บแคชเซิร์ฟเวอร์ที่กำหนด ซึ่งมีรายละเอียดดังนี้

proxy-only เป็นการกำหนดให้มีการร้องขอข้อมูลที่ต้องการไปยังเว็บแคชเซิร์ฟเวอร์ที่กำหนดไว้แต่ไม่ต้องทำสำเนาข้อมูลที่ได้รับกลับมาไว้ที่ตัวเอง

weight=n โดย n มีค่าเป็นเลขจำนวนเต็ม การกำหนดค่านี้ให้กับแต่ละเซิร์ฟเวอร์จะเป็นการบอกถึงว่าต้องการในการใช้งานเซิร์ฟเวอร์ว่ามากน้อยเพียงใด โดยที่ Squid จะเลือกใช้งานเซิร์ฟเวอร์ที่มีค่านี้มากที่สุดก่อน ตามปรกติถ้าไม่มีการกำหนดเป็นอย่างอื่นค่า weight จะมีค่าเท่ากับ 1

no-query เป็นการกำหนดว่าไม่ต้องทำการส่ง ICP-Query กับเว็บแคชเซิร์ฟเวอร์นี้

default เป็นการกำหนดให้เว็บแคชเซิร์ฟเวอร์นี้เป็นเครื่องที่ Squid จะส่งการร้องขอข้อมูลไปเมื่อไม่พบข้อมูลที่แคชของตัวเอง

round-robin เป็นการกำหนดให้เว็บแคชเซิร์ฟเวอร์นี้อยู่ในรายชื่อของกลุ่มเครื่องที่จะมีการแบ่งภาระงานแบบ round-robin

closest-only เป็นการกำหนดให้ใช้งานพรอกซีนี้เมื่อไม่มีเซิร์ฟเวอร์เครื่องอื่นตอบกลับมาจาก ICP ว่าไม่พบข้อมูลที่ร้องขอไป

login=user:password ใช้ในการส่ง Username และ Password ไปยังเว็บแคชเซิร์ฟเวอร์ที่กำหนด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตัวอย่างการกำหนดค่า cache_peer ให้เครื่อง proxy1.ce.kmitl.ac.th ทำหน้าที่เป็น parent โดยใช้ proxy_port เท่ากับ 8080 และ icp_port เท่ากับ 3130 โดยให้มี weight เท่ากับ 20 สามารถเขียนแสดงได้ดังนี้

cache_peer proxy1.ce.kmitl.ac.th parent 8080 3130 weight=20

cache_mem เป็นการกำหนดขนาดของหน่วยความจำหลักที่จะนำมาใช้เป็นเมโมรีแคช มีหน่วยเป็นเมกะไบต์ (Mbytes)

cache_dir เป็นการระบุตำแหน่งบนดิสก์และขนาดที่จะให้ Squid นำมาใช้เป็นดิสก์แคช โดยมีรูปแบบในการระบุค่าดังนี้คือ cache_dir type path size L1 L2 โดยที่

type คือประเภทของระบบดิสก์ที่ใช้ ซึ่งถ้าเป็นระบบปฏิบัติการ Unix ก็จะมีค่าเป็น ufs แต่ถ้าระบบดิสก์ที่ใช้เป็นแบบ Asynchronous I/O ก็จะมีค่าเป็น asyncufs

path คือ ตำแหน่งบนดิสก์ที่จะให้ Squid นำมาใช้ทำดิสก์แคช

size คือขนาดของดิสก์แคชมีหน่วยเป็นเมกะไบต์

L1 และ L2 คือจำนวนของไดเรกทอรีย่อยในลำดับที่ 1 และ 2 นับจาก path ลงไป

ตัวอย่างการกำหนดค่า cache_dir โดยให้ Squid ทำดิสก์แคช ที่ไดเรกทอรี /squid/cache โดยแบ่งไดเรกทอรีย่อยระดับที่ 1 และ 2 เท่ากับ 64 และ 128 ไดเรกทอรีย่อยตามลำดับและกำหนดขนาดของดิสก์แคชไว้ที่ 1 กิกะไบต์ (GigaBytes)

cache_dir ufs /squid/cache 1024 64 128

cache_swap_low และ cache_swap_high เป็นการกำหนดขอบเขตในการลบข้อมูลออกจากพื้นที่เก็บข้อมูลของแคช โดยค่าทั้งสองจะมีหน่วยเป็นเปอร์เซ็นต์ โดย cache_swap_high จะใหม่ถึงขอบเขตที่เมื่อใดก็ตามที่มีข้อมูลที่จัดเก็บในแคชคิดเป็นเปอร์เซ็นต์เท่ากับค่า cache_swap_high Squid ก็จะทำการลบข้อมูลบางส่วนออกด้วยกระบวนการในการแทนที่ข้อมูลจนข้อมูลที่เหลืออยู่มีการใช้พื้นที่ในการเก็บคิดเป็นเปอร์เซ็นต์ในแคชต่ำกว่าค่า cache_swap_in

maximum_object_size คือการกำหนดขนาดสูงสุดของข้อมูลที่อนุญาตให้เก็บไว้ในดิสก์แคชได้

cache_access_log คือการกำหนดไดเรกทอรีสำหรับใช้ในการเก็บล็อกไฟล์ (access.log) ซึ่งเป็นผลมากจากการทำงานของ Squid

reference_age คือค่าอายุอ้างอิงที่ Squid จะใช้ในการเปรียบเทียบว่าข้อมูลแต่ละข้อมูลนั้นหมดอายุหรือยังเพื่อที่จะทำการลบข้อมูลนั้นออกจากพื้นที่ของดิสก์แคชในกรณีที่มีความต้องการใช้พื้นที่นั้นในการเก็บข้อมูลใหม่ที่เข้ามา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

replacement_policy เป็นการกำหนดให้ Squid ใช้กระบวนการในการแทนที่ข้อมูลใหม่ลงในแคชโดยจะมีวิธีเลือกลบข้อมูลออกไปที่แตกต่างกัน เช่น Least-Recently Used (LRU), Greedy-Dual Size Frequency (GDSF) และ Least Frequently Used with Dynamic Aging (LFUDA)

2.3.3 ไฟล์ Log ของ Squid เว็บแคชเซิร์ฟเวอร์

เป็นไฟล์ตัวอักษร (Text file) ที่เก็บรายละเอียดในการทำงานของ Squid ให้บริการการร้องขอโดยจะเก็บหนึ่งบรรทัดต่อหนึ่งการร้องขอที่มีเข้ามาไม่ว่าจะมีผลของการทำงานต่อการร้องขอนั้นเป็นอย่างไร โดย ใน 1 บรรทัดนั้นจะแบ่งออกเป็น 10 ฟิลด์ [6] ดังแสดงในตารางที่ 2.1

ตารางที่ 2.1 แสดงฟิลด์ต่าง ๆ ของไฟล์ Log ที่เกิดจากการประมวลผลการร้องขอของ Squid

Timestamp	Elapsed	Client-address	Log-Tag/ HTTP-Code	Size	Request Method	URL	rfc931	Hierarchy Data / Hostname	Content-Type

1. Timestamp

เป็นเวลาในรูปแบบของระบบปฏิบัติการแบบ Unix โดยเริ่มนับเวลาทีละ 1 วินาทีตั้งแต่วันที่ 1 มกราคม ค.ศ.1970 (Julians Time) โดยการบันทึกเวลาลงไฟล์ Log จะกระทำก็ต่อเมื่อการเชื่อมต่อกับไคลเอนต์ถูกปิดลง

2. Elapsed Time

คือเวลาที่ใช้ในการประมวลผลการร้องขอ ตั้งแต่เริ่มการเชื่อมต่อจนสิ้นสุดการเชื่อมต่อ มีหน่วยเป็นมิลลิเซกคั่น (Millisecond, ms)

3. Client Address

คือ IP Address ของไคลเอนต์ที่ร้องขอข้อมูล ในกรณีที่เปิดการใช้งาน log_fqdn ใน file squid.conf ก็จะเก็บ Full Qualify Domain Name แทน

4. Log Tag / HTTP Code

เป็นส่วนที่ใช้อธิบายว่าการร้องขอที่เข้ามามีผลจาการทำงานเป็นอย่างไร เช่น พบ หรือ ไม่พบข้อมูลที่ร้องขอ ซึ่งจะอธิบายต่อไป ส่วน HTTP code คือข้อมูลในส่วนของ HTTP header ที่ตอบกลับมาจากเซิร์ฟเวอร์ที่พรอกซีไปร้องขอข้อมูลมา ซึ่งจะได้อธิบายแยกต่อไปด้านล่าง

5. Size

ขนาดของข้อมูลที่ถูกร้องขอ มีหน่วยเป็นไบต์

6. Request Method

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จะเก็บวิธีการกระทำต่อข้อมูลที่ร้องขอเข้ามา เช่น GET (นำมา) POST (ส่งออกไป) หรือ ICP_Query

7. URL

เก็บ URL ของข้อมูลที่ถูกร้องขอ

8. Ident

ในกรณีที่มีการใช้ความสามารถในการทำ Ident (โดยปรับตั้งค่า Ident_lookup ที่ไฟล์ squid.conf)

9. Hierarchy Data / Hostname

แสดงรายละเอียดว่า ข้อมูลนั้นถูกนำมาจากไหนและอย่างไร เช่น นำมาโดยตรงจากเซิร์ฟเวอร์ต้นทาง หรือ พบที่ Parent ของตัวมัน ซึ่งจะได้อธิบายแยกต่อไปด้านล่าง

10. Content Type

เก็บข้อมูลประเภทของข้อมูลที่ได้รับกลับมาจากเซิร์ฟเวอร์ต้นทาง เช่น เป็นข้อมูล Text, JPG, GIF เป็นต้น

2.3.3.1 รายละเอียดของ Log Tag

Log tag ที่ขึ้นต้นด้วย TCP_ จะหมายถึงการเชื่อมต่อด้วยโพรโตคอล HTTP จากไคลเอนต์เข้ามายัง Squid (โดยเข้ามาที่ พอร์ต 3128 ถ้าไม่มีการกำหนดเป็นอย่างอื่นในค่า tcp_port ในไฟล์ squid.conf)

TCP_HIT หมายถึงข้อมูลที่ไคลเอนต์ร้องขอพบอยู่ในแคชและสามารถส่งให้กับไคลเอนต์ได้

TCP_MEM_HIT หมายถึงพบข้อมูลที่ร้องขออยู่ใน เมโมรีแคช

TCP_MISS หมายถึงข้อมูลที่ไคลเอนต์ร้องขอไม่พบอยู่ในแคช

TCP_REFRESH_HIT หมายถึงข้อมูลที่ไคลเอนต์ร้องขอพบอยู่ในแคชแต่หมดอายุ จากนั้น Squid ได้ทำการส่งการร้องขอแบบ If-Modify-Since ไปยังเซิร์ฟเวอร์ต้นทางและได้รับคำตอบกลับมาทาง HTTP reply ว่า "304 Not modified" ซึ่งหมายถึงว่าข้อมูลนั้นยังไม่มีการเปลี่ยนแปลง (ขนาดและ/หรือวันที่สร้าง) จากนั้น squid จึงส่งข้อมูลที่เก็บอยู่ในแคชไปให้ไคลเอนต์พร้อมทั้งเริ่มนับเวลาที่ข้อมูลถูกเก็บในแคชใหม่

TCP_REF_FAIL_HIT หมายถึงข้อมูลที่ไคลเอนต์ร้องขอพบอยู่ในแคชแต่หมดอายุ จากนั้น Squid ได้ทำการส่งการร้องขอแบบ If-Modify-Since ไปยังเซิร์ฟเวอร์ต้นทางแต่ไม่ได้รับการตอบกลับมา Squid จึงทำการส่งข้อมูลที่พบในแคชแต่หมดอายุนั้นไปให้ไคลเอนต์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

TCP_REFRESH_MISS หมายถึงข้อมูลที่ไคลเอนต์ร้องขอพบอยู่ในแคชแต่หมดอายุ จากนั้น Squid ได้ทำการส่งการร้องขอแบบ If-Modify-Since ไปยังเซิร์ฟเวอร์ต้นทางและได้รับข้อมูลใหม่กลับมา

TCP_CLIENT_REFRESH หมายถึงไคลเอนต์ระบุการร้องขอแบบไม่ใช้แคช (คือขอให้ Squid ไปร้องขอข้อมูลนั้นโดยตรงจากเซิร์ฟเวอร์ต้นทาง)

TCP_IMS_HIT หมายถึงไคลเอนต์ส่งการร้องขอแบบ If-Modified-Since เข้ามาแต่ปรากฏว่าพบข้อมูลในแคชและข้อมูลยังไม่หมดอายุ Squid จึงทำการส่งข้อมูลที่พบในแคชนั้นให้กับไคลเอนต์

TCP_IMS_MISS หมายถึงไคลเอนต์ส่งการร้องขอแบบ If-Modified-Since เข้ามาและปรากฏว่าพบข้อมูลในแคชแต่ข้อมูลหมดอายุไปแล้ว Squid จึงต้องทำการร้องขอข้อมูลนั้นใหม่

TCP_SWAPFAIL หมายถึงข้อมูลที่ร้องขอถูกลงทะเบียนไว้ในรายชื่อของข้อมูลที่ถูกจัดเก็บอยู่ในดิสก์แคช แต่ไม่สามารถเข้าถึงได้ กรณีนี้อาจจะเกิดจากการหยุดการทำงานอย่างผิดปรกติของ Squid process ทำให้เกิดความผิดพลาดในการจัดเก็บข้อมูลในดิสก์แคช

TCP_DENIED หมายถึงการร้องขอจากไคลเอนต์นี้ถูกปฏิเสธ ซึ่งเป็นผลมาจากการทำงานในส่วนของ Access Control List

Log tag ที่ขึ้นต้นด้วย UDP_ จะหมายถึงการเชื่อมต่อด้วยโพรโตคอล ICP จากไคลเอนต์เข้ามายัง Squid (โดยเข้ามาที่ พอร์ต 3130 ถ้าไม่มีการกำหนดเป็นอย่างอื่นในค่า icp_port ในไฟล์ squid.conf)

UDP_HIT หมายถึงข้อมูลที่ไคลเอนต์ร้องขอพบอยู่ในแคชของพรอกซีปลายทาง (ที่ Squid ส่ง ICP-Query ไปร้องขอ) และสามารถส่งให้กับไคลเอนต์ได้

UDP_HIT_OBJ หมายถึงข้อมูลที่ไคลเอนต์ร้องขอถูกพบเหมือนกับกรณีของ UDP_HIT แต่ตัวข้อมูลที่ต้องการนั้นมีขนาดเล็กพอที่จะส่งกลับมาพร้อมกับคำตอบว่าพบข้อมูลนั้นเลย (ส่งกลับมาในส่วนของ TCP-Reply)

UDP_MISS หมายถึงข้อมูลที่ไคลเอนต์ร้องขอไม่พบอยู่ในแคชของพร็อกซีปลายทาง (ที่ Squid ส่ง ICP-Query ไปร้องขอ)

UDP_DENIED หมายถึงการร้องขอจากไคลเอนต์นี้ถูกปฏิเสธ ซึ่งเป็นผลมาจากการทำงานในส่วนของ Access Control List

UDP_INVALID หมายถึงการร้องขอจากไคลเอนต์ที่ไม่ถูกต้อง

UDP_RELOADING หมายถึงการร้องขอถูกปฏิเสธเนื่องจาก Squid กำลังเริ่มต้นการทำงานของตัวอยู่และยังไม่เสร็จสิ้นกระบวนการเรียกข้อมูลขึ้นมาเพื่อใช้ในการทำงาน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Log tag ที่ขึ้นต้นด้วย "ERR_" หมายถึงการร้องขอแบบ HTTP ที่ผิดพลาดไม่สามารถตีความหรือทำงานได้อย่างถูกต้อง

2.3.3.2 **รายละเอียดของ** Hierarchy Data

DIRECT หมายความว่า Squid นำข้อมูลที่ถูกร้องขอนั้นมาจากเซิร์ฟเวอร์ต้นทางตามที่ระบุไว้ใน URL โดยตรง

FIREWALL_IP_DIRECT หมายความว่า Squid นำข้อมูลที่ถูกร้องขอนั้นมาจากเซิร์ฟเวอร์ต้นทางตามที่ระบุไว้ใน URL โดยตรงเนื่องจากเซิร์ฟเวอร์ต้นทางนั้นอยู่ภายในเครือข่ายหลังไฟลวอลล์ตัวเดียวกับ Squid

FIRST_PARENT_MISS หมายความว่า Squid นำข้อมูลที่ถูกร้องขอนั้นมาจาก Parent ที่ติดต่อได้เร็วที่สุด

FIRST_UP_PARENT หมายความว่า Squid นำข้อมูลที่ถูกร้องขอนั้นมาจาก Parent ที่ว่างในรายชื่อของ parent ทั้งหมดที่มี

LOCAL_IP_DIRECT หมายความว่า Squid นำข้อมูลที่ถูกร้องขอนั้นมาจากเซิร์ฟเวอร์ต้นทางตามที่ระบุไว้ใน URL โดยตรงเนื่องจากเซิร์ฟเวอร์ต้นทางนั้นอยู่ในรายชื่อของโฮลต์ที่ระบุไว้ใน local_ip ในไฟล์ squid.conf

SIBLING_HIT หมายความว่า Squid นำข้อมูลที่ถูกร้องขอนั้นมาจากเว็บแคชเซิร์ฟเวอร์ที่อยู่ในระดับเดียวกัน (Sibling)

NO_DIRECT_FAIL หมายความว่า Squid ไม่สามารถนำข้อมูลที่ถูกร้องขอมาได้เนื่องจากไม่สามารถผ่านไฟลวอลล์หรือไม่มีเว็บแคชเซิร์ฟเวอร์ที่เป็น Parent ว่างในการรับการร้องขอจาก Squid

NO_PARENT_DIRECT หมายความว่า Squid นำข้อมูลที่ถูกร้องขอนั้นมาจากเซิร์ฟเวอร์ต้นทางอันเนื่องมากจากไม่มีรายชื่อของเว็บแคชเซิร์ฟเวอร์ที่เป็น Parent ที่กำหนดไว้สำหรับข้อมูลที่มีการร้องขอเข้ามา

PARENT_HIT หมายความว่า Squid นำข้อมูลที่ถูกร้องขอนั้นมาจากเว็บแคชเซิร์ฟเวอร์ที่เป็น Parent

SINGLE_PARENT หมายความว่าข้อมูลนี้เป็น URL ที่กำหนดไว้ว่าต้องถูกร้องขอจาก Parent ที่ระบุไว้เท่านั้น

SOURCE_FASTEST หมายความว่า Squid นำข้อมูลที่ถูกร้องขอนั้นมาจากเซิร์ฟเวอร์ต้นทางเนื่องมากจากเครื่องเซิร์ฟเวอร์ต้นทางตอบกลับมาก่อนพรอกซีเซิร์ฟเวอร์ตัวอื่น

PARENT_UDP_HIT_OBJ หมายความว่าข้อมูลที่ถูกร้องขอนั้นได้รับกลับมาในการตอบรับ UDP_HIT_OBJ reply จาก Parent เลย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

SIBLING_UDP_HIT_OBJ หมายความว่าข้อมูลที่ถูกร้องขอนั้นได้รับกลับมาในการตอบรับ UDP_HIT_OBJ reply จาก Sibling เลย

PASSTHROUGH_PARENT หมายความว่าข้อมูลถูกร้องขอผ่านเว็บแคชเซิร์ฟเวอร์ที่ระบุไว้ใน passthrough_proxy ในไฟล์ squid.conf

SSL_PARENT_MISS หมายความว่าข้อมูลถูกร้องขอผ่านเว็บแคชเซิร์ฟเวอร์ที่ระบุไว้ใน ssl_proxy ในไฟล์ squid.conf.

DEFAULT_PARENT หมายความว่าข้อมูลถูกร้องขอผ่านเว็บแคชเซิร์ฟเวอร์ที่กำหนดไว้เป็น default ใน cache_peer ในไฟล์ squid.conf โดยที่ไม่ได้มีการส่ง ICP-Message ออกไปเพื่อหาพรอกซีที่มีการตอบสนองเร็วที่สุด

ROUNDROBIN_PARENT หมายความว่าข้อมูลถูกร้องขอผ่านเว็บแคชเซิร์ฟเวอร์ที่กำหนดไว้เป็น default ใน cache_peer ในไฟล์ squid.conf และเป็นเว็บแคชเซิร์ฟเวอร์ที่มีการใช้งานน้อยที่สุดจากกระบวนการแบ่งภาระงานแบบ Round-robinโดยที่ไม่ได้มีการส่ง ICP-Message ออกไปเพื่อหาพรอกซีที่มีการตอบสนองเร็วที่สุด

CLOSEST_PARENT_MISS หมายความว่าข้อมูลถูกร้องขอผ่านเว็บแคชเซิร์ฟเวอร์ที่ทีค่า Round-trip time ไปยังเซิร์ฟเวอร์ต้นทางต่ำที่สุด ซึ่งกรณีนี้จะเกิดขึ้นก็ต่อเมื่อมีการเปิดใช้งาน query_icmp ในไฟล์ squid.conf

CLOSEST_DIRECT หมายความว่าข้อมูลถูกร้องขอไปยังเซิร์ฟเวอร์ต้นทางโดยตรงเนื่องจากตัว Squid เองเป็นผู้ที่มีค่า Round-trip time ต่ำที่สุด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 3
อัลกอริธึมการแทนที่ข้อมูลที่นำมาพิจารณาในงานวิจัย

ในขั้นตอนการทำงานของเว็บแคชเซิร์ฟเวอร์ ช่วงที่เวลาที่เว็บแคชเซิร์ฟเวอร์จะทำการบันทึกข้อมูลที่นำมาจากเว็บไซต์ต้นทางของเว็บเพ็จลงไปในแคชซึ่งเป็นพื้นที่ของฮาร์ดดิสก์หรือแรมของเว็บแคชเซิร์ฟเวอร์ เว็บแคชเซิร์ฟเวอร์จะตรวจสอบพื้นที่ว่างภายในแคชสำหรับการเก็บข้อมูล หากแคชของเว็บแคชเซิร์ฟเวอร์มีเนื้อที่สำหรับเก็บข้อมูลไม่เกินค่าที่กำหนดสำหรับการเก็บข้อมูลภายในแคช เว็บแคชเซิร์ฟเวอร์จะบันทึกข้อมูลที่ได้มาลงไปในแคช ถ้าแคชของเว็บแคชเซิร์ฟเวอร์มีเนื้อที่สำหรับเก็บข้อมูลเกินกว่าค่าที่กำหนดสำหรับการเก็บข้อมูลภายในแคช เว็บแคชเซิร์ฟเวอร์จะเลือกข้อมูลที่จะย้ายจากเมมโมรี่แคช (แคชของเว็บแคชเซิร์ฟเวอร์ในแรม) ไปยังดิสก์แคช (แคชของเว็บแคชเซิร์ฟเวอร์ในฮาร์ดดิสก์) หรือนำออกไปจากดิสก์แคช เพื่อให้พื้นที่ว่างภายในแคชมีเพียงพอสำหรับเก็บข้อมูล โดยเรียกใช้ฟังก์ชั่นที่ชื่อว่า อัลกอริธึมการแทนที่ข้อมูล

3.1 ชนิดของอัลกอริธึมการแทนที่ข้อมูล

อัลกอริธึมการแทนที่ข้อมูลเป็นฟังก์ชั่นหลักที่ต้องสร้างไว้ภายในเว็บแคชเซิร์ฟเวอร์เพื่อทำหน้าที่ในการจัดการข้อมูลภายในแคชของเว็บแคชเซิร์ฟเวอร์ ในอดีตจนถึงปัจจุบัน อัลกอริธึมการแทนที่ข้อมูล ถูกนำเสนอและถูกพัฒนาขึ้นมาหลายชนิด ซึ่งอัลกอริธึมการแทนที่ข้อมูลแต่ละชนิดจะมีรูปแบบการทำงานและตัวแปรที่นำมาพิจารณา แตกต่างกันไป อัลกอริธึมการแทนที่ข้อมูลที่ถูกนิยมนำมาประยุกต์ใช้งานในเว็บแคชเซิร์ฟเวอร์แสดงได้ดังนี้

- Least-Recently Used (LRU) จะทำการเลือกแทนที่ข้อมูลออกจากแคชของเว็บแคชเซิร์ฟเวอร์โดยพิจารณาจากอายุของข้อมูล ซึ่งเริ่มนับตั้งแต่เวลาที่ข้อมูลถูกเก็บไว้ในแคชหรือข้อมูลถูกเรียกใช้จากแคชไปจนถึงเวลาปัจจุบัน

- Least-Frequently-Used (LFU) จะพิจารณาเลือกแทนที่ข้อมูลที่มีความถี่ในการร้องขอจากไคลเอนต์น้อยที่สุดออกจากแคชของเว็บแคชเซิร์ฟเวอร์เป็นอันดับแรก

- Size [16] จะทำการเลือกแทนที่ข้อมูลที่มีขนาดใหญ่ที่สุดออกจากแคชของเว็บแคชเซิร์ฟเวอร์เป็นอันดับแรก

- LRU-Threshold [17] จะใช้หลักในการเลือกแทนที่ข้อมูลในแคชเหมือนกับวิธีแบบ LRU แต่มีเกณฑ์ของขนาดสูงสุดของข้อมูลที่จะเก็บไว้ในแคชกำกับไว้ด้วย โดยถ้าข้อมูลที่จะทำการเก็บมีขนาดใหญ่กว่าเกณฑ์ที่กำหนดก็จะไม่เก็บข้อมูลนั้นลงในแคช

- Log (Size) + LRU [17] จะใช้หลักในการเลือกแทนที่ข้อมูลในแคชเหมือนกับวิธีแบบ LRU แต่จะทำการเปรียบเทียบเฉพาะกับข้อมูลที่มีขนาดใกล้เคียงกับข้อมูลที่จะนำมาแทนที่เท่านั้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- Hyper-G [16] จะใช้หลักในการเลือกแทนที่ข้อมูลในแคชเหมือนกับวิธีแบบ LFU แต่จะนำ เวลาในการเรียกใช้ข้อมูลครั้งล่าสุดและขนาดของข้อมูลมาพิจารณาประกอบ

- Pitkow/Rocker [16] จะใช้หลักในการเลือกแทนที่ข้อมูลในแคชเหมือนกับวิธีแบบ LRU ยกเว้นในกรณีที่ข้อมูลทั้งหมดถูกร้องขอภายในวันเดียวกันทั้งหมด (ไม่มีข้อมูลใดไม่ถูกร้องขอนานเกินกว่า 1 วัน) จะเปลี่ยนมาพิจารณาแทนที่ข้อมูลที่มีขนาดใหญ่ที่สุด

- Lowest-Latency-First [18] อัลกอริธึมนี้แคชจะพยายามลดค่าเวลาแฝง (Latency time) ในการเรียกข้อมูลจากเว็บเซิร์ฟเวอร์ต้นทาง โดยจะทำการแทนที่ข้อมูลที่มีเวลาแฝงในการเรียกข้อมูลน้อยที่สุดก่อนเป็นอันดับแรก

- Least Frequently Used with Dynamic Aging (LFUDA) [19, 20] อัลกอริธึมนี้จะทำงานคล้ายกับ LFU แต่จะเพิ่มการพิจารณาอายุของข้อมูลที่ถูกเรียกใช้เข้ากับกับจำนวนครั้งที่ข้อมูลนี้ถูกเรียกใช้ด้วย และจะเลือกข้อมูลที่มีค่าจำนวนการถูกเรียกใช้ซึ่งบวกกับค่าอายุของมันแล้วมีค่าน้อยที่สุดออกไปจากแคชของเว็บแคชเซิร์ฟเวอร์

- Greedy-Dual-Size Frequency (GDSF) [19, 21] อัลกอริธึมนี้จะนำขนาดของข้อมูลมาคำนวณร่วมกับจำนวนการเรียกใช้งาน และอายุของข้อมูลในแคช โดยจะเป็นการนำอัลกอริธึมการแทนที่ข้อมูลแบบ GD-Size มาเพิ่มการพิจารณาถึงความถี่ในการร้องขอข้อมูลด้วย และจะเลือกข้อมูลที่มีค่าการพิจารณาน้อยที่สุดออกไปจากแคชของเว็บแคชเซิร์ฟเวอร์

นอกเหนือจากที่กล่าวมา ในงานวิจัยตัวอื่น ๆ [22] ได้มีการนำเสนออัลกอริธึมการแทนที่ข้อมูลอีกหลาย ๆ ชนิด และในงานวิจัยนี้ได้ศึกษาถึงอัลกอริธึมการแทนที่ข้อมูลของเว็บแคชเซิร์ฟเวอร์ที่ได้รับการนำเสนอและได้ถูกนำไปประยุกต์ใช้กับเว็บแคชเซิร์ฟเวอร์หลาย ๆ ชนิดคือ อัลกอริธึมการแทนที่ข้อมูลแบบ GDSF และ อัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA เนื่องจากอัลกอริธึมการแทนที่ข้อมูลทั้ง 2 ชนิด เกี่ยวข้องกับตัวแปรที่เหมาะสมสำหรับการพิจารณาการนำข้อมูลที่อยู่ในแคชของเว็บแคชเซิร์ฟเวอร์ออกไป

3.2 การพิจาณาอัลกอริธึมการแทนที่ข้อมูลแบบ Least Frequently Used with Dynamic Aging (LFUDA)

อัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA [19, 20] เป็นอัลกอริธึมการแทนที่ข้อมูลที่ถูกพัฒนามาจากอัลกอริธึมการแทนที่ข้อมูลแบบ Least Frequency Used (LFU) และอัลกอริธึมการแทนที่ข้อมูลแบบ Least Frequency Used-Aging (LFU-Aging) ตามลำดับ ซึ่งอัลกอริธึมการแทนที่ข้อมูลแบบ LFU และแบบ LFU-Aging จะมีรูปแบบการทำงานดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.2.1 อัลกอริธึมการแทนที่ข้อมูลแบบ Least Frequency Used (LFU)

อัลกอริธึมการแทนที่ข้อมูลแบบ LFU เป็นอัลกอริธึมการแทนที่ข้อมูลที่จะพิจารณาข้อมูลที่จะนำออกจากแคชของเว็บแคชเซิร์ฟเวอร์โดยใช้จำนวนครั้งหรือความถี่การร้องขอข้อมูลที่อยู่ภายในแคช ซึ่งข้อมูลใดที่มีจำนวนครั้งหรือความถี่ของการร้องขอน้อยครั้งที่สุด จะถูกนำออกไปจากแคชเป็นข้อมูลกลุ่มแรก

3.2.2 อัลกอริธึมการแทนที่ข้อมูลแบบ Least Frequency Used-Aging (LFU-Aging)

อัลกอริธึมการแทนที่ข้อมูลแบบ LFU-Aging [22] เป็นอัลกอริธึมการแทนที่ข้อมูลที่ถูกพัฒนามาจากอัลกอริธึมการแทนที่ข้อมูลแบบ LFU เนื่องจากในรูปแบบการแทนที่ข้อมูลแบบ LFU ข้อมูลที่มีความนิยมมากระหว่างช่วงระยะเวลาหนึ่งจะสามารถอยู่ภายในแคชของเว็บแคชเซิร์ฟเวอร์ แม้จะไม่ได้รับการร้องขอเป็นเวลานาน เนื่องจากผลรวมของจำนวนครั้งหรือความถี่ของการร้องขอที่มีค่าสูงของข้อมูลนั้น เพื่อหลีกเลี่ยงปัญหาของแคชตรงนี้ การวิเคราะห์ผลกระทบทางด้านอายุจึงได้ถูกนำเสนอขึ้นมา

การทำงานของอัลกอริธึมการแทนที่ข้อมูลแบบ LFU-Aging จะใช้ค่า Threshold เข้ามาเป็นตัวพิจารณาสำหรับการเลือกข้อมูลที่จะถูกนำออกไปจากแคชของเว็บแคชเซิร์ฟเวอร์ โดยมีเงื่อนไขการพิจารณาดังนี้ หากค่าเฉลี่ยของการนับจำนวนครั้งหรือความถี่การร้องขอทั้งหมดมีค่ามากกว่าค่า Threshold ที่กำหนด จำนวนครั้งหรือความถี่การร้องขอทั้งหมดจะถูกหารด้วย 2 และอัลกอริธึมการแทนที่ข้อมูลแบบนี้ยังใช้ค่าที่มากที่สุดสำหรับการนับจำนวนครั้งหรือความถี่การร้องขอ

3.2.3 อัลกอริธึมการแทนที่ข้อมูลแบบ Least Frequently Used with Dynamic Aging (LFUDA)

จากงานวิจัยและการพัฒนาอัลกอริธึมการแทนที่ข้อมูลแบบ LFU และ LFU-Aging จะพบว่า อัลกอริธึมการแทนที่ข้อมูลแบบ LFU จะพิจารณาเฉพาะความถี่ของการร้องขอข้อมูลเท่านั้น และอัลกอริธึมการแทนที่ข้อมูลแบบ LFU-Aging จะขึ้นอยู่กับพารามิเตอร์อย่างมาก (ค่า Threshold และ ข้อมูลที่มีความถี่สูง) อัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA พยายามแก้ไขปัญหาที่เกิดขึ้น โดยได้นำอัลกอริธึมการแทนที่ข้อมูลแบบ LFU-Aging มาพิจารณาและแทนที่ฟังก์ชั่นในส่วนของอายุข้อมูล ด้วยฟังก์ชั่นที่มีความเป็นไดนามิกส์เข้าไป ซึ่งฟังก์ชั่นที่เป็นไดนามิกส์ที่นำเข้าไปแทนที่จะเรียกว่า ฟังก์ชั่น Running Age (L) โดย L จะมีค่าเริ่มต้นที่ 0 และจะมีการปรับเปลี่ยนค่าเมื่อมีการนำข้อมูล f ออกไปจากแคชของเว็บแคชเซิร์ฟเวอร์ และค่า L ที่มีการปรับเปลี่ยนจะถูกนำไปใส่ในค่าคีย์ (K) ของข้อมูลภายในแคช: $L = K_f$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากการพัฒนาอัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA ทำให้เมื่อเว็บแคชเซิร์ฟเวอร์ต้องการพิจารณาข้อมูลที่จะนำออกไปจากแคช สามารถพิจารณาโดยคำนวณค่าคีย์ (K) ของข้อมูลแต่ละตัวภายในแคชของเว็บแคชเซิร์ฟเวอร์ โดยมีสมการสำหรับการคำนวณค่าคีย์แสดงได้ดังสมการที่ 3.1

$$K_i = (C_i * F_i) + L \tag{3.1}$$

โดย K_i คือค่าคีย์สำหรับนำมาพิจารณาข้อมูลที่จะนำออกไปจากแคชของเว็บแคชเซิร์ฟเวอร์

C_i คือค่าคงที่ซึ่งนำมารวมเมื่อนำข้อมูลเข้ามาในแคช

F_i คือจำนวนครั้งหรือความถี่ของการร้องขอข้อมูลในแคชของเว็บแคชเซิร์ฟเวอร์

L คือฟังก์ชั่น Running Age ซึ่ง จะมีค่าเริ่มต้นที่ 0 และจะมีการปรับเปลี่ยนค่าเมื่อมีการนำข้อมูล f ออกไปจากแคชของเว็บแคชเซิร์ฟเวอร์ และค่า L ที่มีการปรับเปลี่ยนจะถูกนำไปใส่ในค่าคีย์ (K) ของข้อมูลภายในแคช: $L = K_f$

3.3 การพิจารณาอัลกอริธึมการแทนที่ข้อมูลแบบ Greedy-Dual-Size Frequency (GDSF)

อัลกอริธึมการแทนที่ข้อมูลแบบ (GDSF) ได้รับการนำเสนอโดย Cherkasova [21] ซึ่งเป็นอัลกอริธึมการแทนที่ข้อมูลที่ได้รับการพัฒนามาจากอัลกอริธึมการแทนที่ข้อมูลแบบ Greedy-Dual (GD) ที่ได้รับการนำเสนอโดย Young [23] และอัลกอริธึมการแทนที่ข้อมูลแบบ Greedy-Dual-Size (GD-Size) ที่ได้รับการนำเสนอโดย Cao และ Irani [24] ซึ่งอัลกอริธึมการแทนที่ข้อมูลแบบ GD และ GD-Size จะมีรูปแบบการทำงานดังนี้

3.3.1 อัลกอริธึมการแทนที่ข้อมูลแบบ Greedy-Dual (GD)

อัลกอริธึมการแทนที่ข้อมูลแบบ GD ได้รับการนำเสนอโดย Young [22] ซึ่งจะพิจารณาเกี่ยวข้องกับกรณีที่เมื่อข้อมูลภายในแคชของเว็บแคชเซิร์ฟเวอร์มีขนาดเดียวกัน แต่มีค่าที่จะพิจารณาข้อมูลแตกต่างกันเพื่อนำข้อมูลออกมาจากแคช

อัลกอริธึมการแทนที่ข้อมูลนี้สัมพันธ์กับค่า H กับข้อมูลเพ็จ p แต่ละเพ็จที่ถูกเก็บไว้ในแคช ในเบื้องต้น เมื่อข้อมูลเพ็จถูกนำมาเก็บไว้ในแคช ค่า H ถูกกำหนดให้มีค่าที่จะนำข้อมูลเพ็จไปเก็บไว้ในแคชของเว็บแคชเซิร์ฟเวอร์ (ให้สังเกตว่าค่าจะไม่เป็นค่าลบ) เมื่อมีการทำการแทนที่ข้อมูล ข้อมูลเพ็จที่มีค่า H น้อยที่สุด ($\min_H$) จะถูกแทนที่และข้อมูลเพ็จทั้งหมดจะถูกลดค่า H

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ของข้อมูลเพ็จด้วยค่า $\min_H$ ถ้าข้อมูลเพ็จ p ถูกร้องขออีกครั้งหนึ่งค่า H ในปัจจุบันของข้อมูลเพ็จนั้นจะถูกจัดเก็บใหม่ ให้กับค่าเริ่มต้นของการนำข้อมูลเพ็จนี้เข้าสู่แคชของเว็บแคชเซิร์ฟเวอร์

ด้วยวิธีการนี้ค่า H ของข้อมูลเพ็จที่มีการร้องขอเมื่อไม่นานนี้ จะยังคงมีค่าที่มากกว่าของค่าเริ่มต้นเมื่อนำไปเปรียบเทียบกับข้อมูลเพ็จที่ไม่ได้รับการร้องขอเป็นเวลานาน อัลกอริธึมการแทนที่ข้อมูลแบบ GD จะใช้ข้อมูลเพ็จที่มีค่า H น้อยที่สุดเพื่อใช้สำหรับการแทนที่ข้อมูลในครั้งแรก ซึ่งค่านี้จะเป็นค่าที่ "แพงน้อยที่สุด" ที่จะนำไปเก็บไว้ในแคชของเว็บแคชเซิร์ฟเวอร์หรือเป็นข้อมูลเพ็จที่ไม่ได้รับการร้องขอเป็นเวลานาน

เพื่อพัฒนาและเพิ่มประสิทธิภาพการทำงานของอัลกอริธึมการแทนที่ข้อมูลแบบ GD โดยการใช้ลำดับความสำคัญและใช้ค่าชดเชย (offset value) สำหรับการปรับแต่งค่า H จึงได้มีการนำเสนออัลกอริธึมการแทนที่ข้อมูลแบบ Greedy-Dual-Size ขึ้นมา

3.3.2 อัลกอริธึมการแทนที่ข้อมูลแบบ Greedy-Dual-Size (GD-Size)

เนื่องจากเว็บแคชเซิร์ฟเวอร์จะเป็นเซิร์ฟเวอร์ที่ทำหน้าที่เกี่ยวข้องกับการเก็บข้อมลขนาดต่าง ๆ Cao และ Irani [23] ได้พัฒนาอัลกอริธึมแบบ GD ให้มีการพิจารณาเกี่ยวข้องกับตัวแปรขนาดของข้อมล โดยเปลี่ยนค่า H ไปเป็น $cost/size$ โดย $cost$ คือค่าของการนำเข้าของข้อมูล และ $size$ คือขนาดของข้อมลในหน่วยไบต์ จากกระบวนการนี้ทำให้เรียกอัลกอริธึมการแทนที่ข้อมูลที่ได้พัฒนาใหม่ว่า อัลกอริธึมการแทนทีข้อมูลแบบ Greedy-Dual-Size (GD-Size)

ผลการจำลองการทำงานที่แสดงให้เห็นโดย Cao และ Irani แสดงให้เห็นว่าอัลกอริธึมการแทนทีข้อมลทีนำเสนอทำงานได้ดีกว่าอัลกอริธึมการแทนทีข้อมูลตัวอื่นทีรู้จักกันในปัจจบัน สำหรับเมตริกซ์ประสิทธิภาพที่แตกต่างกันด้วยฟังก์ชั่น $cost$ ที่ถูกเลือก
เมตริกซ์ธรรมดา 2 แบบที่ถูกนำมาใช้ประเมินประสิทธิภาพของเว็บแคชเซิร์ฟเวอร์คือ

- Hit ratio คือ จำนวนของการร้องขอที่พบข้อมูลที่ร้องขอภายในแคชของเว็บแคชเซิร์ฟเวอร์เป็นเปอร์เซ็นต์ของการร้องขอทั้งหมด
- Byte-Hit ratio คือจำนวนไบต์ของขนาดข้อมูลที่มีการร้องขอและพบข้อมูลที่ร้องขอภายในแคชของเว็บแคชเซิร์ฟเวอร์เป็นเปอร์เซ็นต์ของจำนวนการร้องขอทั้งหมดเป็นไบต์

เพื่อให้ได้ค่า Hit ratio ที่ดีที่สุด ฟังก์ชั่น $cost$ สำหรับข้อมูลแต่ละตัวจะตั้งค่าเป็น 1 โดยวิธีนี้ ข้อมูลที่มีขนาดใหญ่กว่าจะมีค่าคีย์ความสำคัญที่เล็กกว่าข้อมูลที่มีขนาดเล็กกว่า และมีแนวโน้มที่จะถูกนำออกไปจากแคชหากไม่ได้รับการร้องขออีกในอนาคต การขยายค่า Hit ratio จะมีประโยชน์มากกว่าที่จะแทนทีข้อมูลที่มีขนาดใหญ่เพียงข้อมูลเดียว (และพลาดข้อมูลนี้หากข้อมูลนี้ได้รับการร้องขออีกครั้ง) มากกว่าที่จะนำข้อมลที่มีขนาดเล็กจำนวนมากออกไปจากแคช เพื่อนำพื้นที่ของแคชที่ใช้ขนาดเดียวกัน (และพลาดข้อมูลเหล่านี้เป็นจำนวนมากเมื่อข้อมูลเหล่านี้ถูกร้องขออีกครั้ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หนึ่ง) ค่า cost ที่เป็น 1 ข้อมูลเล็ก ๆ ที่ได้รับความนิยมและเข้าแทนที่ข้อมูลที่มีขนาดใหญ่ โดยเฉพาะสิ่งที่อ้างอิงที่หายากแหละนี้ ทำให้เรียกรูปแบบนี้ว่า GD-Size ดังนั้น GD-Size จะมีค่าของ Hit ratio ที่ดีที่สุด แต่ค่า Hit ratio ที่สูงสำหรับ GD-Size จะให้ค่าของ Byte-Hit ratio ที่ต่ำกว่า

รูปแบบ GD-Size(packets) ตั้งฟังก์ชั่น cost สำหรับข้อมูลเป็น $2+size/536$ ซึ่งเป็นค่าประมาณของแพคเก็จในเครือข่ายที่มีการส่งและรับในกรณีที่เว็บแคชเซิร์ฟเวอร์ไม่พบข้อมูลที่ถูกร้องขอ รูปแบบ GD-Size(packets) จะมีค่า Hit ratio และค่า Byte-Hit ratio ที่สูง ฟังก์ชั่น cost นี้จะให้ค่าคีย์ที่มีขนาดใหญ่กว่าสำหรับข้อมูลที่มีขนาดใหญ่มากกว่าข้อมูลที่มีขนาดเล็ก ฟังก์ชั่นนี้จะยอมให้ข้อมูลที่มีขนาดเล็กแทนที่ได้มากกว่าข้อมูลที่มีขนาดใหญ่ (โดยเฉพาะถ้าเอกสารที่มีขนาดใหญ่ถูกร้องขอบ่อย) ถ้าเอกสารที่มีขนาดใหญ่ไม่ถูกอ้างอิงอีกเลย ก็จะถูกแทนที่โดยใช้ตังแปรทางด้านอายุ

ดังนั้น GD-Size จึงมีจุดมุ่งหมายที่จะทำให้ Miss ratio ลดน้อยลง ขณะที่ GD-Size(packets) พยายามที่จะทำให้ความคับคั่งของเครือข่ายที่มีผลกระทบจากการ Miss ข้อมูลลดลง ที่จริงแล้ว ดังที่ได้พิจารณาเมตริกซ์ทั้ง 2 จะมีความขัดแย้งกัน และมีความยากมากที่รูปแบบใดรูปแบบหนึ่งจะให้ผลการทำงานของเมตริกซ์ทั้ง 2 ที่ดี โดยทั่วไปค่า Hit ratio ที่สูงเป็นทางเลือกที่ดีกว่าเพราะเป็นการแสดงว่าไคลเอนต์เป็นจำนวนมากได้ค้นพบข้อมูลที่ร้องขอจากเว็บแคชเซิร์ฟเวอร์และลดค่าเวลาแฝงเฉลี่ยของการร้องขอ แต่การลดความคับคั่งของเครือข่ายก็ยังเป็นที่ต้องการมากกว่า รูปแบบที่มีค่า Byte-Hit ratio สูงจึงถูกนำมาใช้ อัลกอริธึมการแทนที่ข้อมูลแบบ GD-Size ที่มี cost ที่เหมาะสมทำได้เหนือกว่าอัลกอริธึมการแทนที่ข้อมูลที่เป็นที่รู้จักกันในปัจจุบันเกี่ยวกับจำนวนของเมตริกซ์ของประสิทธิภาพรวมทั้งค่า Hit ratio และค่า Byte-Hit ratio

แต่อัลกอริธึมการแทนที่ข้อมูลแบบ GD-Size ยังมีข้อบกพร่องอีกประการคือ ไม่ได้มีการนับจำนวนครั้งของการร้องขอในอดีตเข้าไปในคีย์สำหรับการพิจารณาการนำข้อมูลออกไปจากแคชของเว็บแคชเซิร์ฟเวอร์ ดังนั้นจึงได้มีการพัฒนาอัลกอริธึมการแทนที่ข้อมูลแบบ GD-Size ให้มีตัวแปรที่เกี่ยวของกับจำนวนครั้งของการร้องขอข้อมูล มาเป็น อัลกอริธึมการแทนที่ข้อมูลแบบ Greedy-Dual-Size Frequency โดยปรับค่า H ให้เป็น

$$H = frequency \times \frac{cost}{size} \qquad (3.2)$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.3.3 อัลกอริธึมการแทนที่ข้อมูลแบบ Greedy-Dual-Size Frequency (GDSF)

จากการพัฒนาปรับปรุงอัลกอริธึมการแทนที่ข้อมูลแบบ GD และ แบบ GD-Size ให้เป็นอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF จะสามารถอธิบายการทำงานของอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF ได้ดังนี้

- ให้ขนาดแคชของเว็บแคชเซิร์ฟเวอร์เป็น *Total*
- ให้ *Used* เป็นจำนวนพื้นที่ของแคชที่ใช้สำหรับจัดเก็บข้อมูลถูกนำมาเก็บไว้ในเว็บแคชเซิร์ฟเวอร์
- เริ่มต้นให้ $Used = 0$
- เมื่อข้อมูลแต่ละตัว f ถูกเก็บไว้ในแคช และได้ทำการนับความถี่ $Fr(f)$: จำนวนครั้งของการร้องขอ ข้อมูล f ที่ไม่อยู่ในแคชแต่กำลังจะถูกนำมาเก็บไว้ในแคชจะมีค่าของความถี่เป็น 1: $Fr(f) = 1$

เพื่อระบุว่าข้อมูลใดจะถูกแทนที่เมื่อขนาดความจุของแคชเกินกว่าค่าที่กำหนด เราสามารถกำหนดลำดับความสำคัญของข้อมูลได้ดังนี้

ข้อมูล f จะถูกใส่เข้าไปในลำดับความสำคัญด้วยค่าคีย์ความสำคัญ $\Pr(f)$ ที่คำนวณได้ดังสมการ 3.3

$$\Pr(f) = Clock + Fr(f) \times \frac{Cost(f)}{Size(f)} \tag{3.3}$$

โดย $Clock$ คือ Running Age ที่มีค่าเริ่มต้นที่ 0 มีการปรับเปลี่ยนค่าสำหรับแต่ละการแทนที่ข้อมูล $f_{evicted}$ ให้กับคีย์ความสำคัญของข้อมูลในลำดับความสำคัญ: $\Pr(f_{evicted})$

$Fr(f)$ คือการนับความถี่การร้องขอข้อมูล f ถ้าข้อมูล f มีการ Hit (ข้อมูลถูกพบในแคชของเว็บแคชเซิร์ฟเวอร์) ค่า $Fr(f)$ จะถูกเพิ่มขึ้นมา 1 ค่า: $Fr(f) = Fr(f) + 1$ ถ้าข้อมูล f มีการ Miss (ข้อมูลไม่ถูกพบในแคชของเว็บแคชเซิร์ฟเวอร์) ค่า $Fr(f)$ จะถูกกำหนดค่าให้มีค่าเป็น 1: $Fr(f) = 1$

$Size(f)$ คือขนาดของข้อมูลในหน่วยไบต์

$Cost(f)$ คือค่าที่สัมพันธ์กับข้อมูล f เพื่อที่จะใช้นำข้อมูลมาไว้ในแคช ซึ่งได้อธิบายไว้ในเรื่องอัลกอริธึมการแทนที่ข้อมูลแบบ GD-Size

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.3.3.1 ขั้นตอนการทำงานของอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF

จะสามารถแบ่งขั้นตอนการทำงานออกเป็น 2 กรณีได้ดังนี้

1. ถ้าข้อมูล f ที่ถูกร้องขอมีค่าเป็น Hit ภายในแคชดังนั้นข้อมูลที่ร้องขอจึงถูกนำออกไปจากแคช และ

- จำนวนของแคช *Used* จะไม่มีการเปลี่ยนแปลง
- การนับความถี่ของข้อมูล $Fr(f)$ จะเพิ่มขึ้นมา 1
- คีย์ความสำคัญ $\Pr(f)$ จะถูกคำนวณใหม่อีกครั้งตามสมการที่ 4
- ข้อมูล f ที่มีการคำนวณคีย์ความสำคัญ $\Pr(f)$ จะถูกนำไปใส่ในลำดับความสำคัญอีกครั้ง เพื่อใช้สำหรับการทำงานครั้งใหม่

สิ้นสุดการทำงานของกรณีนี้

2. ถ้าข้อมูล f ที่ถูกร้องขอมีค่าเป็น Miss ภายในแคชดังนั้นข้อมูลที่ถูกร้องขอจะต้องถูกนำมาจากเซิร์ฟเวอร์ต้นทางและจะถูกเว็บแคชเซิร์ฟเวอร์สำเนาเก็บไว้ในแคชด้วย

- การนับความถี่ของข้อมูล $Fr(f)$ จะถูกตั้งค่าให้เป็น 1
- คีย์ความสำคัญ $\Pr(f)$ จะถูกคำนวณโดยใช้สมการที่ 4
- ข้อมูล f จะถูกนำไปเก็บไว้ในลำดับความสำคัญด้วยคีย์ความสำคัญ $\Pr(f)$ ที่คำนวณได้
- จำนวนของแคช *Used* จะถูกคำนวณใหม่ดังสมการที่ 3.4

$$Used = Used + Size(f) \tag{3.4}$$

หลังจากนั้นจะเกิดเหตุการณ์ใดเหตุการณ์หนึ่งใน 2 เหตุการณ์นี้จะเกิดขึ้น

- ถ้า $Used \leq Total$ หมายความว่ามีเนื้อที่เพียงพอที่จะจัดเก็บข้อมูล f และไม่มีข้อมูลที่ต้องถูกแทนที่ ข้อมูล f จะถูกจัดเก็บลงในแคชของเว็บแคชเซิร์ฟเวอร์ และเสร็จสิ้นการทำงาน
- ถ้า $Used > Total$ หมายความว่า ไม่มีเนื้อที่เพียงพอที่จะจัดเก็บข้อมูล f และข้อมูลภายในแคชของเว็บแคชเซิร์ฟเวอร์บางส่วนจะต้องมีการถูกแทนที่

เริ่มต้น เราระบุกลุ่มของข้อมูลจำนวนน้อยที่จะถูกลบออกไปจากแคชโดยใช้ขั้นตอนต่อไปนี้: ข้อมูล k ข้อมูล (k อาจมีค่าเป็น 1) ที่มีความสำคัญต่ำสุดในลำดับความสำคัญจะถูกเลือก $f_1, f_2, ..., f_k$ เพื่อให้เนื้อที่ $UsedEstimate \leq Total$ โดย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$UsedEstimate = Used - \sum_{i=1}^{k} Size(f_i) \quad (3.5)$$

(a) ถ้าข้อมูลต้นฉบับ f ที่เราจะทำการเก็บไว้ ไม่อยู่ในกลุ่มข้อมูล f_1, f_2, ..., f_k จะเกิดเหตุการณ์ดังนี้

i. ค่า *Clock* (Running Age ของลำดับความสำคัญ) จะถูกคำนวณใหม่ด้วยสมการที่ 3.6

$$Clock = \max_{i=1}^{k} \Pr(f_i) = \Pr(f_k) \quad (3.6)$$

ii จำนวนของแคช *Used* ได้รับการปรับค่าใหม่เป็น

$$Used = Used - \sum_{i=1}^{k} Size(f_i) \quad (3.7)$$

iii ข้อมูล f_1, f_2, ..., f_k ถูกนำออกไปจากแคชของเว็บแคชเซิร์ฟเวอร์

iv ข้อมูล f ถูกนำไปเก็บไว้ในแคชของเว็บแคชเซิร์ฟเวอร์

(b) ถ้าข้อมูลต้นฉบับ f ที่เราจะทำการเก็บไว้อยู่ระหว่างข้อมูล f_1, f_2, ..., f_k ซึ่งต้องถูกนำออกไปเพื่อจัดเก็บข้อมูล f ดังนั้น

i ข้อมูล f จะไม่ถูกนำไปเก็บไว้ในแคชของเว็บแคชเซิร์ฟเวอร์และค่าคีย์ความสำคัญ $\Pr(f)$ จะถูกนำออกไปจากลำดับความสำคัญ

ii ไม่มีข้อมูลในแคชของเว็บแคชเซิร์ฟเวอร์ถูกลบออก

สิ้นสุดการทำงานของกรณีนี้

จุดหลักการเปลี่ยนแปลงที่นำเสนอในส่วนนี้คือการนำเสนอการนำความถี่ของการร้องขอ $Fr(f)$ ที่มีความสัมพันธ์กับข้อมูล f แต่ละตัว ที่ถูกนำไปเก็บไว้ในแคชของเว็บแคชเซิร์ฟเวอร์รวมกันในสมการที่ 4 สิ่งนี้จะสะท้อนให้เห็นรูปแบบการเข้าถึงข้อมูลและนำเสนอการปรับปรุงอัลกอริธึมการแทนที่ข้อมูลแบบ GD-Size เข้ากับเมตริกซ์ของประสิทธิภาพแบบต่าง ๆ พร้อมด้วยการเชื่อมการทำงานเข้ากับฟังก์ชั่น cos*t*

ส่วนของอัลกอริธึมที่เกี่ยวข้องกับการลบข้อมูลออกจากแคชของเว็บแคชเซิร์ฟเวอร์ มีส่วนที่น่าสนใจสัมพันธ์กับกรณีที่เมื่อข้อมูลไม่ถูกเก็บไว้ในแคชเนื่องจากค่าคีย์มีค่าต่ำมากซึ่งได้ใส่ข้อมูลนี้ (กรณีที่ถูกเก็บไว้ในแคช) ไว้เป็นข้อมูลแรกสำหรับการแทนที่ โดยทั่วไปแล้วมันสามารถเกิดขึ้นได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อขนาดของไฟล์ใหญ่มาก กระบวนการที่นำเสนอจะห้ามไม่ให้มีกรณีเหล่านี้โดยอัตโนมัติเมื่อข้อมูลถูกเก็บลงในแคช

3.3.3.2 คุณสมบัติของอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF

เมื่อพิจารณาคุณสมบัติของการแทนที่ข้อมูลแบบ GDSF จะพบว่าในกรณีนี้คีย์ความสำคัญสำหรับข้อมูล f จะถูกคำนวณได้ดังสมการ 3.8

$$\Pr(f) = Clock + (Fr(f) \times \frac{1}{Size(f)}) \tag{3.8}$$

ดังนั้น ข้อมูลที่มีการนับความถี่มากกว่าจะมีค่าของคีย์ความสำคัญที่มากกว่า และมีโอกาสที่จะอยู่ภายในแคชของเว็บแคชเซิร์ฟเวอร์มากกว่า เมื่อเปรียบเทียบกับข้อมูลที่ไม่ค่อยมีการร้องขอ GD-Size-Frequency กำหนดให้คีย์ความสำคัญที่มีค่าสูงให้กับข้อมูลที่มีขนาดเล็กเมื่อเปรียบเทียบกับข้อมูลที่ขนาดใหญ่ โดยมีจุดมุ่งหมายให้ค่า Hit ratio สูงสุดและค่า Miss ratio น้อยสุด เมื่อข้อมูลมีการถูกแทนที่

พารามิเตอร์ *Clock* มีค่าที่ค่อย ๆ เพิ่มขึ้น (มันจะถูกทำให้เพิ่มขึ้นเมื่อข้อมูลถูกแทนที่ออกไปจากแคชของเว็บแคชเซิร์ฟเวอร์) ข้อมูลที่ไม่ได้รับการร้องขอมาเป็นระยะเวลานาน จะไม่มีการเปลี่ยนค่าคีย์ภายในลำดับความสำคัญ เมื่อถึงจุดหนึ่ง ค่าของ *Clock* มีค่าสูงเพียงพอที่ข้อมูลใหม่ใด ๆ จะถูกนำเข้ามาเก็บหลังข้อมูล "เป็นระยะเวลานานที่ไม่ได้รับการร้องขอ" เหล่านี้ โดยวิธีนี้ข้อมูลที่ขนาดเล็กและข้อมูลที่มีการนับความถี่การร้องขอมากจะถูกทำการแทนที่ ถ้าข้อมูลเหล่านี้ไม่ได้รับการร้องขออีก วิธีการ Aging นี้จะป้องกันเว็บแคชเซิร์ฟเวอร์จากปัญหาที่กล่าวมา

อัลกอริธึมการแทนที่ข้อมูลที่มีคุณสมบัติที่คล้ายคลึงกันคืออัลกอริธึมการแทนที่ข้อมูล GD-Size-Frequency(packets) โดยมีจุดมุ่งหมายที่จะได้ค่าHit ratio และค่า Byte-Hit ratio มีค่าสูง โดยมีความแตกต่างเมื่อเทียบกับอัลกอริธึมการแทนที่ข้อมูลแบบ GD-Size-Frequency เพียงอย่างเดียวคือ GD-Size-Frequency(packets) จะไม่แบ่งแยกและปฏิเสธข้อมูลที่มีขนาดใหญ่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 4
ขั้นตอนการปรับแต่งอัลกอริธึมการแทนที่ข้อมูลของ Squid เว็บแคชเซิร์ฟเวอร์

บทที่ 4 จะกล่าวถึงขั้นตอนในการปรับแต่งเว็บแคชเซิร์ฟเวอร์โดยเริ่มตั้งแต่การวัดประสิทธิภาพของเว็บแคชเซิร์ฟเวอร์, การทดสอบการทำงานของเว็บแคชเซิร์ฟเวอร์, การวิเคราะห์อัลกอริธึมการแทนที่ข้อมูลที่นำมาพิจารณา, การปรับแต่งอัลกอริธึมการแทนที่ข้อมูลของ Squid เว็บแคชเซิร์ฟเวอร์, ผลการทดสอบการทำงานของอัลกอริธึมการแทนที่ข้อมูลที่ถูกปรับแต่งและการวิเคราะห์หาค่า Threshold ที่เหมาะสมสำหรับการปรับแต่งอัลกอริธึมการแทนที่ข้อมูล

4.1 การวัดประสิทธิภาพของเว็บแคชเซิร์ฟเวอร์

เว็บแคชเซิร์ฟเวอร์เป็นระบบที่ทำหน้าที่รับการร้องขอข้อมูลจากไคลเอนต์ และจะทำการตรวจสอบข้อมูลที่อยู่ภายในแคชของเว็บแคชเซิร์ฟเวอร์ ถ้าหากข้อมูลที่ไคลเอนต์ร้องขอ ถูกพบภายในแคช และข้อมูลยังไม่หมดอายุ (ตามช่วงอายุของข้อมูลที่ได้ตั้งค่าเอาไว้) ก็จะทำการส่งข้อมูลกลับไปยังให้ไคลเอนต์ แต่หากข้อมูลที่ไคลเอนต์ร้องขอไม่ถูกพบในแคชหรือข้อมูลถูกพบในแคชแต่หมดอายุ เว็บแคชเซิร์ฟเวอร์จะติดต่อไปยังเซิร์ฟเวอร์ต้นทางของเว็บไซต์ที่ไคลเอนต์ร้องขอเพื่อให้เซิร์ฟเวอร์ต้นทางส่งข้อมูลกลับมา เมื่อเว็บแคชเซิร์ฟเวอร์ได้รับข้อมูล ก็จะส่งข้อมูลต่อไปยังไคลเอนต์ที่ร้องขอข้อมูลพร้อมกับทำการสำเนาข้อมูลลงไปในแคชของเว็บแคชเซิร์ฟเวอร์ด้วย จากหลักการทำการของเว็บแคชเซิร์ฟเวอร์ที่กล่าวมา จะแสดงให้เห็นถึงค่า 2 ค่าที่ถูกนิยมใช้ในการพิจารณาถึงประสิทธิภาพการทำงานของเว็บแคชเซิร์ฟเวอร์คือ

1. การพบข้อมูลของเว็บแคชเซิร์ฟเวอร์ (Hit) คือกรณีข้อมูลที่ไคลเอนต์ร้องขอมีการค้นพบภายในแคชของเว็บแคชเซิร์ฟเวอร์ โดยพิจารณาจากข้อมูลภายในไฟล์ Log ของ Squid เว็บแคชเซิร์ฟเวอร์ในส่วนของฟิลด์ Log Tag/HTTP Code ซึ่งค่าภายในฟิลด์ Log Tag/HTTP Code ที่นำมาพิจารณาว่าเป็นการพบข้อมูลของเว็บแคชเซิร์ฟเวอร์ (Hit) ประกอบด้วยค่า TCP_HIT, TCP_MEM_HIT, TCP_REFRESH_HIT, TCP_REF_FAIL_HIT และ TCP_IMS_HIT

2. การไม่พบข้อมูลของเว็บแคชเซิร์ฟเวอร์ (Miss) คือกรณีข้อมูลที่ไคลเอนต์ร้องขอไม่ถูกค้นพบภายในแคชของเว็บแคชเซิร์ฟเวอร์หรือข้อมูลที่อยู่ในแคชของเว็บแคชเซิร์ฟเวอร์หมดอายุแล้วทำให้เว็บแคชเซิร์ฟเวอร์ต้องทำการติดต่อไปยังเซิร์ฟเวอร์ของเว็บไซต์ต้นทางของเว็บไซต์เพื่อนำข้อมูลมาใหม่อีกครั้งหนึ่ง ค่าภายในฟิลด์ Log Tag/HTTP Code ที่นำมาพิจารณาว่าเป็นการไม่พบ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ข้อมูลของเว็บแคชเซิร์ฟเวอร์ (Miss) ประกอบด้วยค่า TCP_MISS, TCP_REFRESH_MISS, TCP_CLIENT_REFRESH , TCP_IMS_MISSม TCP_SWAPFAIL และ TCP_DENIED

ในการพิจารณาประสิทธิภาพของเว็บแคชเซิร์ฟเวอร์ สิ่งที่ได้รับความนิยมในการนำมาวัดประสิทธิภาพของเว็บแคชเซิร์ฟเวอร์คือการพบข้อมูลภายในแคชของเว็บแคชเซิร์ฟเวอร์ ซึ่งจากการค้นพบข้อมูลภายในแคชของเว็บแคชเซิร์ฟเวอร์สามารถแสดงเป็นค่าสำหรับการวิเคราะห์จำนวน 2 ค่าคือ

1.อัตราการพบข้อมูล (Hit Ratio) คืออัตราส่วนของจำนวนการร้องขอที่พบข้อมูลในแคชเปรียบเทียบกับการร้องขอทั้งหมดที่เว็บแคชเซิร์ฟเวอร์ได้รับจากไคลเอนต์ และค่าอัตราการพบข้อมูลเมื่อนำไปคูณกับค่า 100 จะถูกเรียกว่า เปอร์เซ็นต์การพบข้อมูล (%Hit)

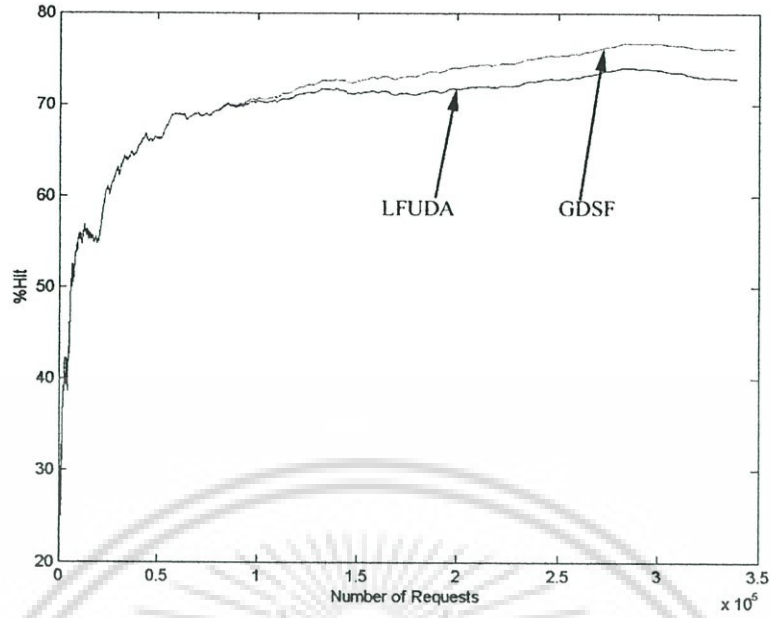
2. อัตราการพบข้อมูลแบบไบต์ (Byte-Hit Ratio) คืออัตราส่วนปริมาณข้อมูลแบบไบต์ของจำนวนการร้องขอที่พบข้อมูลในแคชเปรียบเทียบกับปริมาณข้อมูลแบบไบต์ของการร้องขอทั้งหมดที่เว็บแคชเซิร์ฟเวอร์ได้รับจากไคลเอนต์ และค่าอัตราการพบข้อมูลแบบไบต์เมื่อนำไปคูณกับค่า 100 จะถูกเรียกว่าเปอร์เซ็นต์การพบข้อมูลแบบไบต์ (%Byte-Hit)

4.2 การวิเคราะห์อัลกอริธึมการแทนที่ข้อมูลที่นำมาพิจารณา

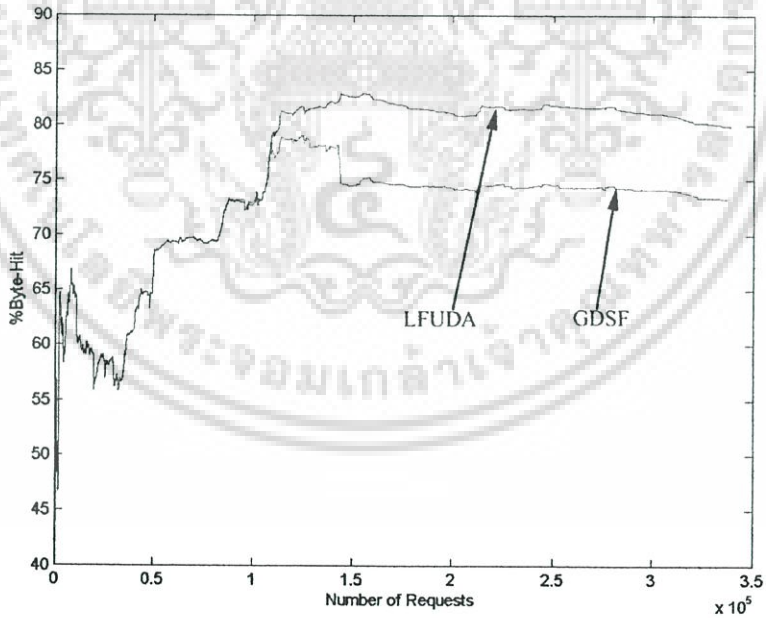
จากบทที่ 3 ในงานวิจัยนี้จะพิจารณาถึงการทำงานของอัลกอริธึมการแทนที่ข้อมูลจำนวน 2 ชนิดคืออัลกอริธึมการแทนที่ข้อมูลแบบ GDSF และ อัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA เนื่องจาก การพิจารณาลักษณะการร้องขอข้อมูลของไคลเอนต์ ข้อมูลที่มีจำนวนครั้งการร้องขอมากจะเป็นข้อมูลที่มีความเหมาะสมที่จะทำการเก็บไว้ในแคชของเว็บแคชเซิร์ฟเวอร์ และอายุของข้อมูลซึ่งจะแสดงให้เห็นถึงความทันสมัยของข้อมูลที่ไคลเอนต์ต้องการ โดยอัลกอริธึมการแทนที่ข้อมูลที่นำมาพิจารณาทั้ง 2 ชนิดนะมีตัวแปรที่เกี่ยวข้องกับสิ่งที่เราพิจารณาอยู่

การพิจารณาถึงคุณสมบัติของอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF และ แบบ LFUDA โดยใช้ระบบทดสอบการทำงานของเว็บแคชเซิร์ฟเวอร์ทำได้โดยทำการตั้งค่าอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF และ แบบ LFUDA ให้กับโปรแกรม Squid เว็บแคชเซิร์ฟเวอร์ภายในเครื่องคอมพิวเตอร์ที่เป็นเซิร์ฟเวอร์ และทำการสร้างการร้องขอด้วยโปรแกรม Cfmc บนคอมพิวเตอร์เครื่องที่เป็นไคลเอนต์ ซึ่งผลการทดสอบของข้อมูลภาควิชาวิศวกรรมคอมพิวเตอร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังแสดงได้ดังรูปที่ 4.1, 4.2 และ 4.3

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

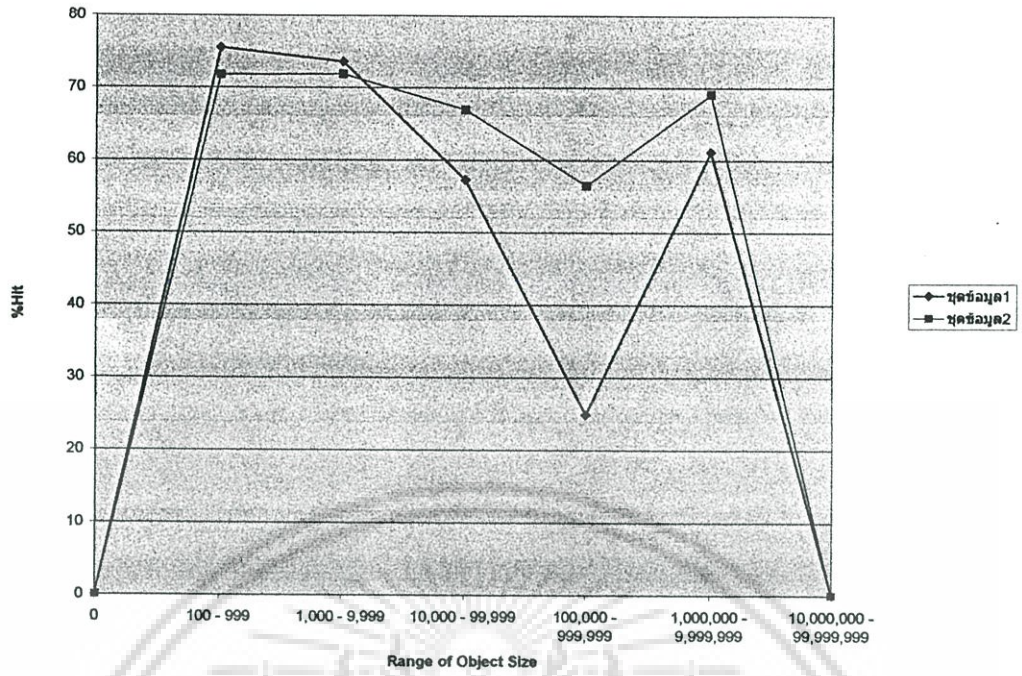


รูปที่ 4.1 กราฟแสดง %Hit ของอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF และอัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA ซึ่งใช้ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง



รูปที่ 4.2 กราฟแสดง %Byte-Hit ของอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF และอัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA ซึ่งใช้ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.3 กราฟเปรียบเทียบ %Hit ของอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF และอัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA ซึ่งใช้ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง แบ่งตามช่วงของขนาดข้อมูล

จากกราฟที่ได้เราสามารถสรุปคุณสมบัติของอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF และอัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA ได้ดังแสดงข้างล่าง

1. อัลกอริธึมการแทนที่ข้อมูลแบบ GDSF จะมีตัวแปรที่มีผลกับค่าการเลือกข้อมูลออกประกอบด้วย ขนาดของข้อมูล, จำนวนครั้งของการเรียกใช้ข้อมูลในแคช และเวลาของข้อมูลที่ถูกเก็บไว้ในแคชดังสมการที่ 3.3 ซึ่งพบว่าในกรณีของข้อมูลที่มีเวลาของข้อมูลที่เก็บไว้ในแคชเท่ากัน ขนาดของข้อมูลจะเป็นตัวแปรสำคัญของค่าการเลือกข้อมูลออกไปจากแคชเพราะ เมื่อข้อมูลมีขนาดใหญ่ค่าการเลือกข้อมูลออกจะมีค่าน้อยกว่าเมื่อเทียบกับข้อมูลที่มีขนาดเล็ก ดังนั้นข้อมูลที่มีขนาดใหญ่จึงถูกนำออกไปจากแคชของเว็บแคชเซิร์ฟเวอร์เป็นลำดับแรก

2. อัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA จะมีตัวแปรที่มีผลกับค่าการเลือกข้อมูลออกประกอบด้วย จำนวนครั้งของการเรียกใช้ข้อมูลในแคช และเวลาของข้อมูลที่ถูกเก็บไว้ในแคชดังสมการที่ 3.1 ซึ่งพบว่าในกรณีของข้อมูลที่มีเวลาของข้อมูลที่เก็บไว้ในแคชเท่ากัน จำนวนครั้งของการเรียกใช้ข้อมูลในแคชจะเป็นตัวแปรสำคัญของการเลือกข้อมูลออกไปจากแคชเพราะ ข้อมูลที่ถูกเก็บไว้ในแคชและมีการเรียกใช้งานบ่อย จะมีค่าการเลือกข้อมูลออกสูงกว่าข้อมูลที่ถูกเก็บไว้ใน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แคชและมีการเรียกใช้งานน้อย ดังนั้นข้อมูลที่ถูกเรียกใช้งานน้อยจึงถูกนำออกไปจากแคชของเว็บแคชเซิร์ฟเวอร์เป็นลำดับแรก

และเมื่อนำผลที่ได้ไปเปรียบเทียบกับการทดลองในระบบเครือข่ายจริงโดยใช้ข้อมูลชุดเดียวกัน [28] จะได้ค่าออกมาในทิศทางเดียวกัน

4.3 การจำลองการทำงานของเว็บแคชเซิร์ฟเวอร์

การจำลองการทำงานของเว็บแคชเซิร์ฟเวอร์ในฟังก์ชั่นของอัลกอริธึมการแทนที่ข้อมูลเป็นกระบวนการทดสอบ การทำงานของอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF, อัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA และอัลกอริธึมการแทนที่ข้อมูลที่นำเสนอ และเพื่อหลีกเลี่ยงการได้รับผลกระทบจากตัวแปรที่ไม่เกี่ยวข้องและมีส่วนที่ทำให้ค่าของ %Hit และ %Byte-Hit ผิดพลาดไปจากที่ต้องการ เช่น ค่า latency ของระบบเครือข่าย, การได้รับการร้องขอที่นอกเหนือจากที่กำหนด และการล่าช้าในการประมวลผลของเว็บแคชเซิร์ฟเวอร์

4.3.1 เครื่องมือสำหรับการจำลองการทำงานของเว็บแคชเซิร์ฟเวอร์

เครื่องมือใช้สำหรับการจำลองการทำงานของอัลกอริธึมการแทนที่ข้อมูลจะมีส่วนประกอบดังนี้

1. เครื่องไมโครคอมพิวเตอร์ ใช้ซีพียู Athlon 2.2 GHz หน่วยความจำ 512 MB ฮาร์ดดิสก์ความจุ 40 GB ระบบปฏิบัติการที่ใช้คือ Microsoft Windows XP Professional [25]

2. โปรแกรม Perl for windows version 5.8.0 [26] ซึ่งนำมาใช้งานสำหรับแปลงข้อมูลที่นำมาทำการทดสอบให้อยู่ในรูปแบบที่เหมาะสมสำหรับการจำลองการทำงานของเว็บแคชเซิร์ฟเวอร์ในฟังก์ชั่นของอัลกอริธึมการแทนที่ข้อมูล

4.3.2 ข้อมูลที่นำมาทดสอบ

ข้อมูลที่นำมาทดสอบกับการทดสอบการทำงานของเว็บแคชเซิร์ฟเวอร์ ประกอบด้วยข้อมูลจำนวน 2 ชุดข้อมูลโดยมีรายละเอียดดังนี้

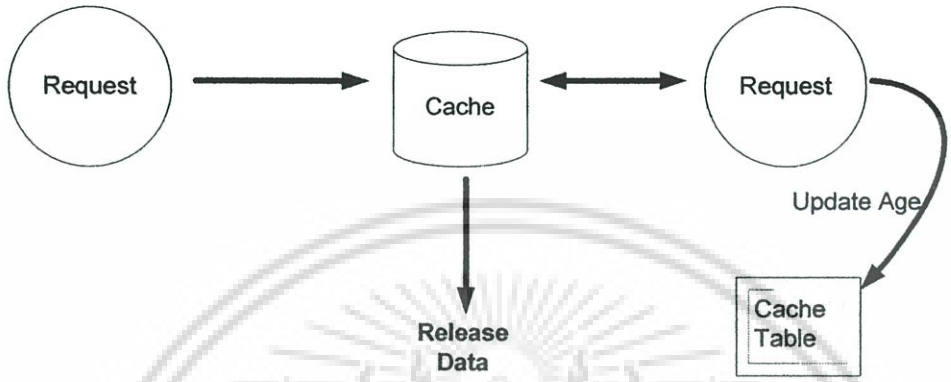
1. ข้อมูลไฟล์ Log ที่ได้จากการรวบรวมภายในระยะเวลา 1 สัปดาห์จากเว็บแคชเซิร์ฟเวอร์ของภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ประกอบด้วยการร้องขอจำนวนทั้งหมด 338,822 ครั้ง ซึ่งการร้องขอทั้งหมดจะเป็นข้อมูลที่เป็น unique จำนวน 78,580 ชุด

2. ข้อมูลไฟล์ Log ที่ได้จากการรวบรวมภายในระยะเวลา 2 สัปดาห์ของเว็บเซิร์ฟเวอร์ของ Clarknet ประกอบด้วยการร้องขอข้อมูลทั้งหมด 1,465,050 ครั้ง ซึ่งการร้องขอทั้งหมดจะเป็นข้อมูลที่เป็น unique จำนวน 35,356 ชุด โดยข้อมูลไฟล์ Log จากเว็บเซิร์ฟเวอร์ของ Clarknet สามารถดาวน์โหลดจากเว็บไซต์ http://ita.ee.lbl.gov/html/contrib/Calgary-HTTP.html [27]

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.3.3 โมเดลของการจำลองการทำงานของเว็บแคชเซิร์ฟเวอร์ในฟังก์ชั่นของอัลกอริธึมการแทนที่ข้อมูล

โมเดลของการจำลองการทำงานของเว็บแคชเซิร์ฟเวอร์ในฟังก์ชั่นของอัลกอริธึมการแทนที่ข้อมูล ดังแสดงในรูปที่ 4.4



รูปที่ 4.4 โมเดลของการจำลองการทำงานของเว็บแคชเซิร์ฟเวอร์ในฟังก์ชั่นของอัลกอริธึมการแทนที่ข้อมูล

จากรูปที่ 4.4 การทำงานจะเริ่มจาก

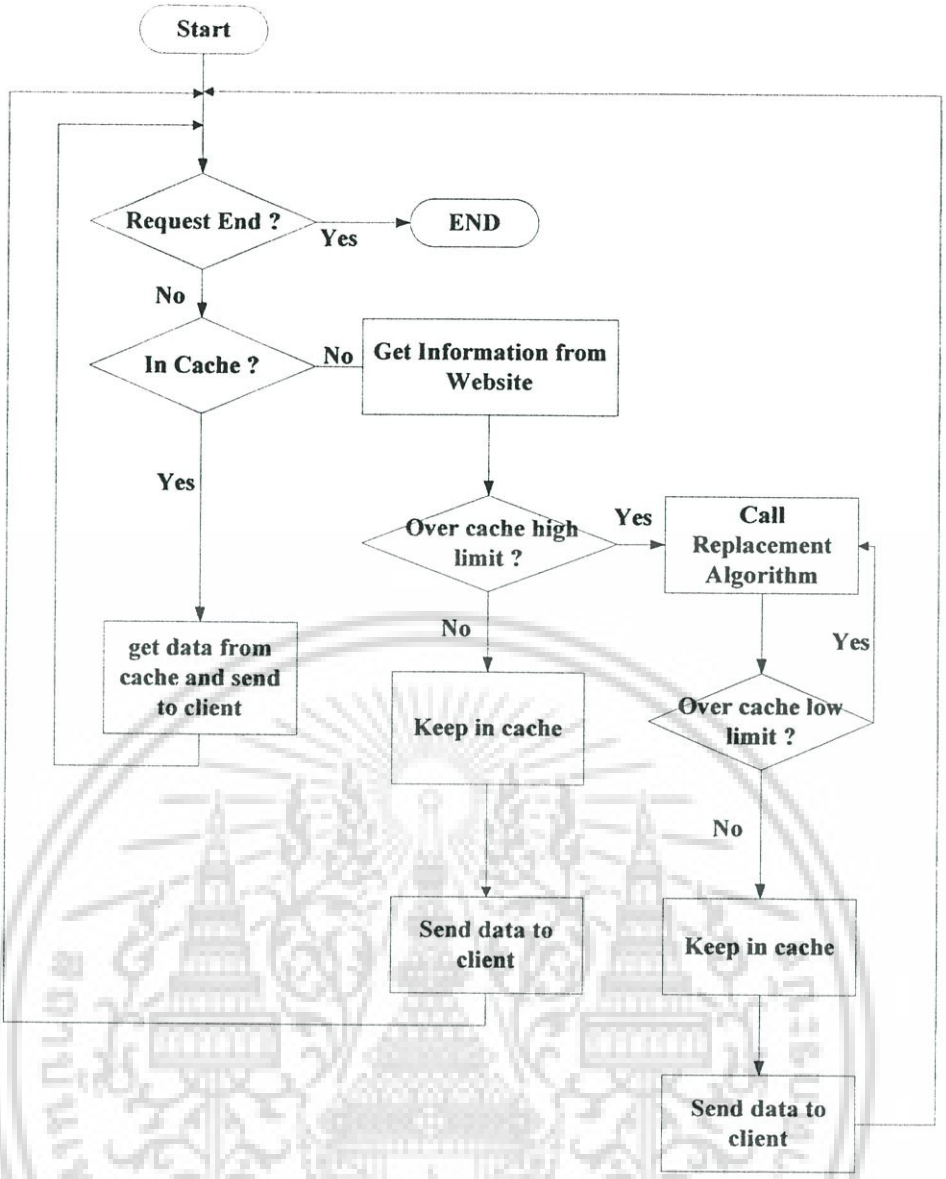
- กำหนดแคชของเว็บแคชเซิร์ฟเวอร์ โดยมีตัวแปรที่เกี่ยวข้องคือ ขนาดของแคช, ชนิดของอัลกอริธึมการแทนที่ข้อมูลที่ใช้, ค่า Higher limit ของแคช, ค่า Lower limit ของแคช และค่า Threshold สำหรับอัลกอริธึมการแทนที่ข้อมูลที่นำเสนอ

- ทำการส่งการร้องขอที่ได้จากการข้อมูลทดสอบแต่ละชุดเข้าที่ละการร้องขอ

-ตรวจสอบข้อมูลในแคช หากพบจะทำการอัพเดทค่าคีย์ความสำคัญ และรับการร้องขอต่อไป หากไม่พบจะทำการตรวจสอบขนาดของแคช ถ้าเกินค่าที่กำหนดจะเรียกใช้อัลกอริธึมการแทนที่ข้อมูลเพื่อนำข้อมูลออกไปจากแคชและทำการอัพเดทค่าอายุข้อมูลจนกระทั่งขนาดของแคชลดลงถึงค่าที่กำหนดจะหยุดเรียกใช้อัลกอริธึมการแทนที่ข้อมูลพร้อมกับเก็บข้อมูลลงในแคช และรับการร้องขอต่อไป

ขั้นตอนการทำงานของการจำลองการทำงานของเว็บแคชเซิร์ฟเวอร์ในฟังก์ชั่นของอัลกอริธึมการแทนที่ข้อมูลจะแสดงได้ดังรูปที่ 4.5

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.5 ขั้นตอนการทำงานของการจำลองการทำงานของเว็บแคชเซิร์ฟเวอร์ในฟังก์ชั่นอัลกอริธึมการแทนที่ข้อมูล

4.4 การปรับแต่งอัลกอริธึมการแทนที่ข้อมูลของ Squid เว็บแคชเซิร์ฟเวอร์

ในงานวิจัยนี้ มุ่งศึกษาถึงการเพิ่มค่าของ %Byte-Hit ของเว็บแคชเซิร์ฟเวอร์ ซึ่งค่าของ %Byte-Hit จะมีค่าสูงเมื่อมีการพบข้อมูลขนาดใหญ่ภายในแคชของเว็บแคชเซิร์ฟเวอร์เป็นจำนวนมาก จากคุณสมบัติของอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF และ LFUDA จึงได้ทำการปรับปรุงอัลกอริธึมการแทนที่ข้อมูล ดังนี้

1. กำหนดค่า Threshold ซึ่งเป็นค่าที่ใช้สำหรับเปรียบเทียบกับขนาดของข้อมูลที่เว็บแคชเซิร์ฟเวอร์ได้รับมาจากเซิร์ฟเวอร์ต้นทางของเว็บไซต์

2. ข้อมูลที่มีขนาดเท่ากับหรือมากกว่าค่า Threshold ขึ้นไปจะพิจารณาการนำข้อมูลออกจากแคชโดยใช้อัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA เนื่องจากข้อมูลที่มีขนาดใหญ่และมีการ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เรียกใช้งานบ่อย ๆ ควรจะถูกนำมาเก็บไว้ในแคชและข้อมูลขนาดใหญ่ที่ไม่มีการเรียกใช้งานบ่อย ๆ ก็ควรนำออกไปจากแคชของเว็บแคชเซิร์ฟเวอร์

3. ข้อมูลที่มีขนาดน้อยกว่าค่า Threshold ลงมาจะถูกพิจารณาการนำข้อมูลออกจากแคช โดยการใช้อัลกอริธึมการแทนที่ข้อมูลแบบ GDSF เนื่องจากการแทนที่ข้อมูลแบบ GDSF จะพิจารณาถึงข้อมูลที่มีขนาดเล็กและมีการเรียกใช้งานบ่อย ๆ เป็นหลัก

จากขั้นตอนการปรับแต่งอัลกอริธึมการแทนที่ข้อมูลที่นำเสนอ สามารถนำมาเขียนสมการของของค่า คีย์ความสำคัญ ของข้อมูลแต่ละตัวได้ดังสมการที่ 4.1

$$K_i = \begin{cases} \left(\frac{F_i * C}{S_i}\right) + L & \text{เมื่อ } S_i < \text{Threshold} \\ (F_i * C) + L & \text{เมื่อ } S_i \geq \text{Threshold} \end{cases} \quad (4.1)$$

ในการใช้งานอัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งจะทำการกำหนดให้ค่าของ C มีค่าเป็น 1 เพื่อให้ความถี่ของการร้องขอและอายุของข้อมูลที่ถูกเก็บไว้ในแคชของเว็บแคชเซิร์ฟเวอร์มีความสำคัญที่เท่าเทียมกันดังนั้นจากสมการที่ 4.1 จะสามารถเขียนได้ใหม่เป็นดังสมการที่ 4.2

เมื่อวิเคราะห์สมการที่ 4.1 พบว่า ค่าของ F_i ของอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF และอัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA จะเริ่มต้นจากค่า 1 และเพิ่มขึ้นเรื่อย ๆ เมื่อมีการร้องขอซ้ำจากไคลเอนต์

เมื่อนำมาใช้กับอัลกอริธึมการแทนที่ข้อมูลที่ทำการปรับแต่งพบว่า $1 \leq S_i < \alpha$ ดังนั้นเมื่อพิจารณากรณีที่ $S_i <$ Threshold ในกรณีที่เว็บแคชเซิร์ฟเวอร์ทำการเก็บข้อมูลลงไปในแคชครั้งแรกจะทำให้ $0 < F_i/S_i \leq 1$ ซึ่งเมื่อนำมาเปรียบเทียบกับค่า F_i ในกรณีที่ $S_i \geq$ Threshold จะพบว่าค่าของ F_i จะมีค่ามากกว่าค่าของ F_i/S_i เสมอ ดังนั้นเมื่อนำมารวมกับค่า L จะทำให้ค่าของ K_i ในกรณี ที่ $S_i \geq$ Threshold มีค่ามากกว่า ค่า K_i ในกรณีที่ $S_i <$ Threshold เสมอ ทำให้โอกาสที่ข้อมูลที่มีขนาดมากกว่าค่า Threshold จะถูกนำออกไปจากแคชของเว็บแคชเซิร์ฟเวอร์มีค่าเป็นศูนย์

ดังนั้นเพื่อแก้ไขปัญหาที่เกิดขึ้น ในกรณี $S_i \geq$ Threshold จึงได้ทำการปรับค่าความถี่ของการนับข้อมูลที่ถูกร้องขอจากค่าเดิมที่จะเริ่มนับตั้งแต่ค่า 1 เปลี่ยนให้เริ่มนับจากค่า 0 แทน เพราะว่าเมื่อค่าความถี่ของการนับข้อมูลที่ถูกร้องขอมีค่าเริ่มตั้งแต่ 0 จะทำให้ค่า K_i ที่ได้เมื่อนำข้อมูลมาเก็บไว้ในแคชครั้งแรกจะมีค่าเป็นค่า L ทำให้ข้อมูลที่มีขนาดเกินมากกว่าหรือเท่ากับค่า Threshold มีโอกาสที่จะถูกนำออกไปจากแคชของเว็บแคชเซิร์ฟเวอร์ และในอัลกอริธึมการแทนที่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ข้อมูลแบบ GDSF ค่าของคีย์ความสำคัญจะเกี่ยวข้องกับค่าขนาดของข้อมูลดังนั้นหากให้ค่าความถี่ของการร้องขอเป็น 0 จะเป็นผลให้ค่าคีย์ความสำคัญเริ่มต้นมีค่าเป็น 0 จำนวนมากเป็นผลให้ค่าของฟังก์ชั่น L ไม่มีการเปลี่ยนค่า เพื่อแก้ปัญหาที่เกิดขึ้นจึงกำหนดให้ค่าความถี่ของการนับข้อมูลสำหรับกรณี S_i < Threshold เริ่มนับตั้งแต่ 1 และในการใช้งานอัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งจะทำการกำหนดให้ค่าของ C มีค่าเป็น 1 เพื่อให้ความถี่ของการร้องขอและอายุของข้อมูลที่ถูกเก็บไว้ในแคชของเว็บแคชเซิร์ฟเวอร์มีความสำคัญที่เท่าเทียมกันดังนั้นจากสมการที่ 4.1 จะสามารถเขียนได้ใหม่เป็นดังสมการที่ 4.2

$$K_i = \begin{cases} \left(\frac{F_i}{S_i}\right) + L & \text{เมื่อ } S_i < \text{Threshold} \\ F_i + L & \text{เมื่อ } S_i \geq \text{Threshold} \end{cases} \qquad (4.2)$$

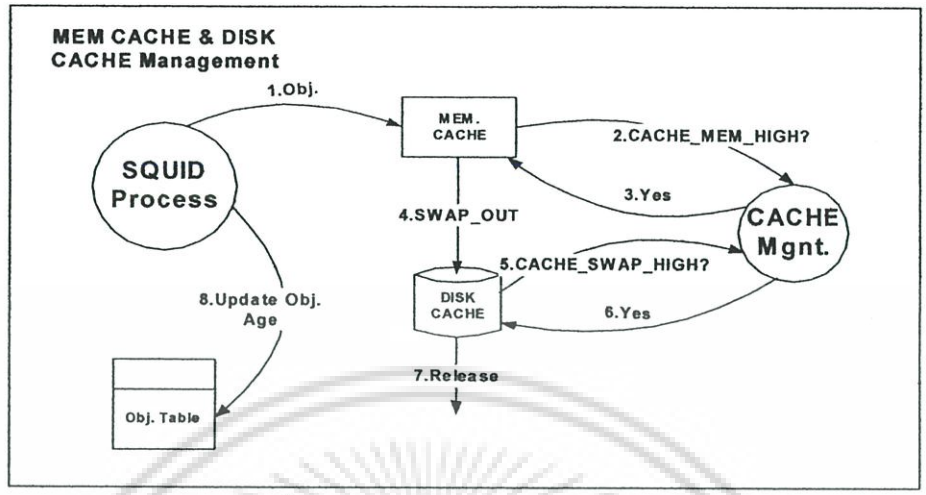
โดย K_i คือค่าคีย์สำหรับนำมาพิจารณาข้อมูลที่จะนำออกไปจากแคชของเว็บแคชเซิร์ฟเวอร์

F_i คือจำนวนครั้งหรือความถี่ของการร้องขอข้อมูลในแคชของเว็บแคชเซิร์ฟเวอร์ โดยจะมีค่าเริ่มตั้งแต่ 0 และเพิ่มขึ้นครั้งละ 1 เมื่อมีการร้องขอซ้ำ สำหรับกรณี $S_i \geq$ Threshold และมีค่าเริ่มตั้งแต่ 1 และเพิ่มขึ้นครั้งละ 1 เมื่อมีการร้องขอซ้ำ สำหรับกรณี $S_i <$ Threshold

L คือฟังก์ชั่น Running Age ซึ่ง จะมีค่าเริ่มต้นที่ 0 และจะมีการปรับเปลี่ยนค่าเมื่อมีการนำข้อมูล f ออกไปจากแคชของเว็บแคชเซิร์ฟเวอร์ และค่า L ที่มีการปรับเปลี่ยนจะถูกนำไปใส่ในค่าคีย์ (K) ของข้อมูลภายในแคช: $L = K_f$

จากรูปที่ 4.6 เราจะพิจารณาถึงขั้นตอนการทำงานของ Squid เว็บแคชเซิร์ฟเวอร์ในขั้นตอนการจัดเก็บข้อมูลลงในแคชของเว็บแคชเซิร์ฟเวอร์ โดยจะมีขั้นตอนการทำงานใหญ่ 2 ขั้นตอนคือ ขั้นตอนการจัดเก็บข้อมูลลงในแคชและขั้นตอนการเรียกใช้งานอัลกอริธึมการแทนที่ข้อมูลดังแสดงได้ในขั้นตอนที่ 1 จนถึงขั้นตอนที่ 8 ของรูปที่ 4.6

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.6 แสดงการจัดเก็บข้อมูลในแคชของ Squid

เมื่อพิจารณาเฉพาะในส่วนของการเรียกใช้อัลกอริธึมการแทนที่ข้อมูล (ฟังก์ชั่น CACHE Mgnt. ภายในรูปที่ 4.6) จะแสดงได้ในขั้นตอนที่ 2 จนถึงขั้นตอนที่ 7 โดยจะแบ่งขั้นตอนการทำงานออกเป็น 2 ส่วนคือส่วนของ Memory Cache โดย Memory Cache เป็นแคชของเว็บแคชเซิร์ฟเวอร์ที่สร้างจากแรมของเครื่องเซิร์ฟเวอร์และ ส่วนของDisk Cache โดย Disk Cache เป็นแคชของเว็บแคชเซิร์ฟเวอร์ที่สร้างจากฮาร์ดดิสก์ของเครื่องเซิร์ฟเวอร์

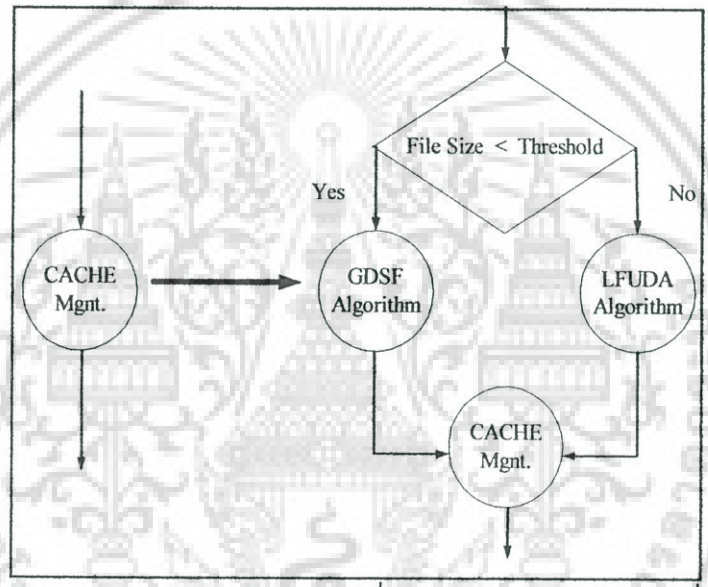
1. Memory Cache เป็นการเรียกใช้อัลกอริธึมการแทนที่ข้อมูลสำหรับการเลือกข้อมูลที่จะถูกย้ายจาก Memory Cache ลงไปเก็บไว้ใน Disk Cache ของเว็บแคชเซิร์ฟเวอร์โดยจะมีการทำงานตั้งแต่ขั้นตอนที่ 2 จนถึงขั้นตอนที่ 4 ของรูปที่ 4.6 ซึ่งในขั้นตอนที่ 2 เว็บแคชเซิร์ฟเวอร์จะตรวจสอบขนาดของข้อมูลที่อยู่ใน Memory Cache มีค่าเกินกว่าค่าที่กำหนดหรือไม่ หากมีค่าเกินจะเรียกใช้อัลกอริธึมการแทนที่ข้อมูล และนำข้อมูลที่ถูกเลือกออกไปเก็บไว้ใน Disk Cache ในขั้นตอนที่ 4 จากนั้นจะย้อนกลับไปทำงานในขั้นตอนที่ 2 ถึงขั้นตอนที่ 4 จนกระทั่งค่าของขนาดข้อมูลใน Memory Cache มีค่าน้อยกว่าค่าที่กำหนด

2. Disk Cache เป็นการเรียกใช้อัลกอริธึมการแทนที่ข้อมูลสำหรับการเลือกข้อมูลที่จะถูกนำออกไปจากDisk Cache ของเว็บแคชเซิร์ฟเวอร์โดยจะมีขั้นตอนการทำงานตั้งแต่ขั้นตอนที่ 5 จนถึงขั้นตอนที่ 7 ของรูปที่ 4.6 ซึ่งในขั้นตอนที่ 5 เว็บแคชเซิร์ฟเวอร์จะตรวจสอบขนาดของข้อมูลที่อยู่ใน Disk Cache มีค่าเกินกว่าค่าที่กำหนดหรือไม่ หากมีค่าเกิน จะเรียกใช้อัลกอริธึมการแทนที่ข้อมูล และนำข้อมูลที่ถูกเลือกออกไปจาก Disk Cache ในขั้นตอนที่ 7 จากนั้นจะย้อนกลับไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ทำงานในขั้นตอนที่ 5 ถึงขั้นตอนที่ 7 จนกระทั่งค่าของขนาดข้อมูลใน Disk Cache มีค่าน้อยกว่าค่าที่กำหนด

ในการปรับแต่งอัลกอริธึมการแทนที่ข้อมูลของการทดสอบจะเข้าไปปรับแต่งในส่วนฟังก์ชั่นที่ชื่อว่า CACHE Mgnt. ที่แสดงในรูปที่ 4.6 ซึ่งเป็นฟังก์ชั่นที่ทำหน้าที่เป็นอัลกอริธึมการแทนที่ข้อมูลของเว็บแคชเซิร์ฟเวอร์โดยในการทดสอบจะใช้โปรแกรม Squid เวอร์ชั่น 2.5 stable 3 และปรับแต่งฟังก์ชั่นการทำงานของอัลกอริธึมการแทนที่ข้อมูลที่ถูกเก็บไว้ภายในไฟล์ /src/repl/heap/store_heap_replacement.c ซึ่งเป็นไฟล์ที่มีหน้าที่ในการสร้างค่าสำหรับใช้ในการพิจารณาการนำข้อมูลออกของอัลกอริธึมการแทนที่ข้อมูลแต่ละชนิด การปรับแต่งแสดงได้ดังรูปที่ 4.7



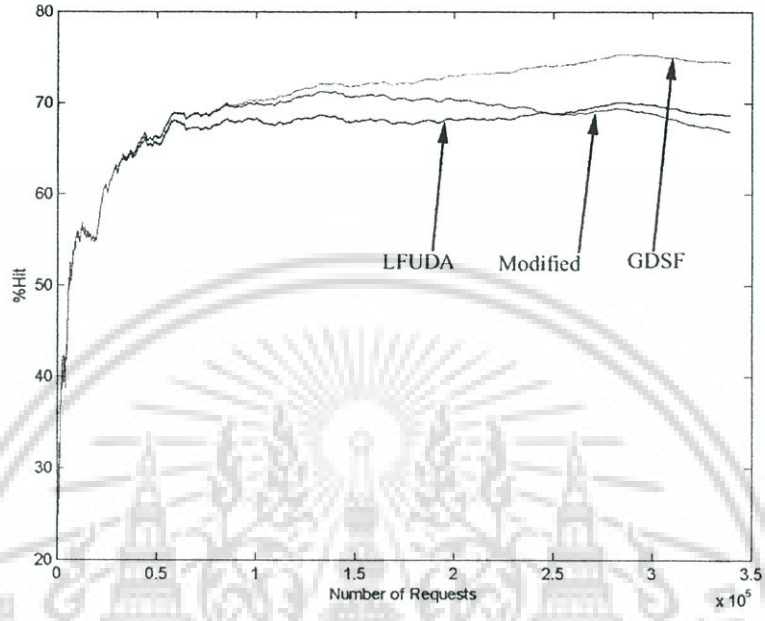
รูปที่ 4.7 การปรับแต่งอัลกอริธึมการแทนที่ข้อมูลของเว็บแคชเซิร์ฟเวอร์ที่นำเสนอ

4.5 ผลการทดสอบการทำงานของอัลกอริธึมการแทนที่ข้อมูลที่ถูกปรับแต่ง

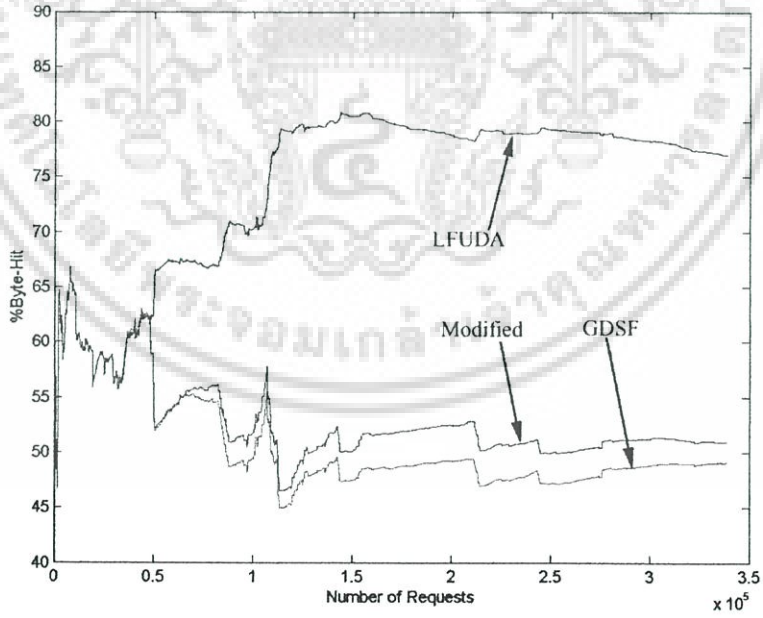
ในการทดสอบจะทำการทดสอบโดยทำการปรับเปลี่ยนค่าของ Threshold จำนวน 4 ค่า คือ 10Kbytes, 100 Kbytes, 1 Mbytes และ 10 Mbytes โดยใช้ข้อมูลชุดเดียวกันในการทดสอบจำนวน 2 ชุดคือข้อมูลที่ได้จากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง และข้อมูลที่ได้จาก Clarknet แล้วนำมาทำการวิเคราะห์ โดยทำการวิเคราะห์ในส่วนของ %Hit และ %Byte-Hit ที่ได้จากการวิเคราะห์ค่า Threshold ที่มีค่า 10Kbytes, 100 Kbytes, 1 Mbytes และ 10 Mbytes และได้ทำการปรับเปลี่ยนขนาดแคชของเว็บแคชเซิร์ฟเวอร์ให้มีขนาด 64 Mbytes, 128 Mbytes และ 256 Mbytes เพื่อพิจารณาผลกระทบที่เกิดจากขนาดของแคชที่เปลี่ยนไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.5.1 การเปรียบเทียบ อัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 10 Kbytes ซึ่งใช้ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง โดยตั้งแคชที่ขนาด 64 Mbytes



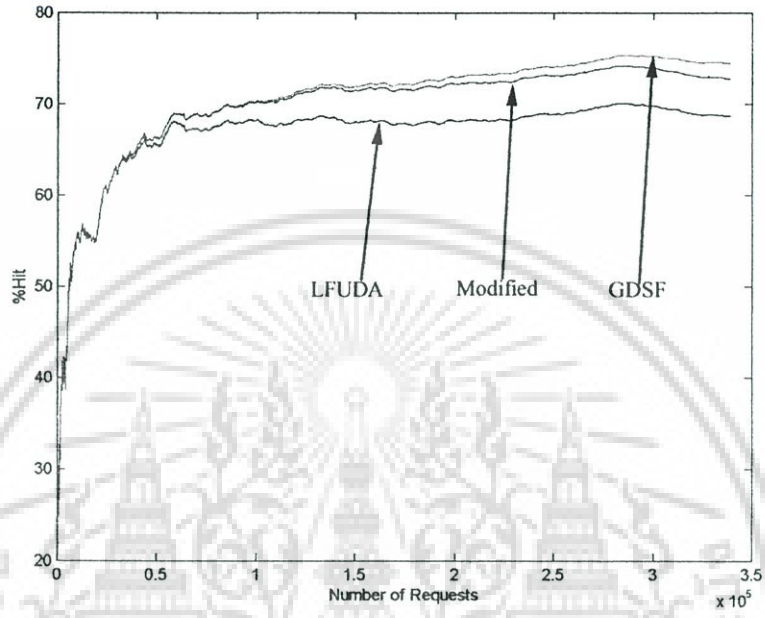
รูปที่ 4.8 กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 10 Kbytes



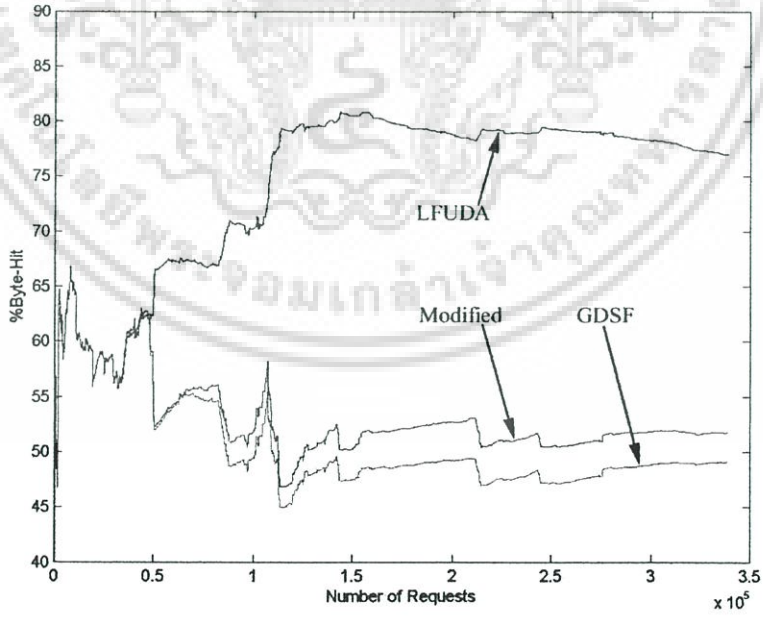
รูปที่ 4.9 กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 10 Kbytes

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.5.2 การเปรียบเทียบ อัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 100 Kbytes ซึ่งใช้ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง โดยตั้งแคชที่ขนาด 64 Mbytes



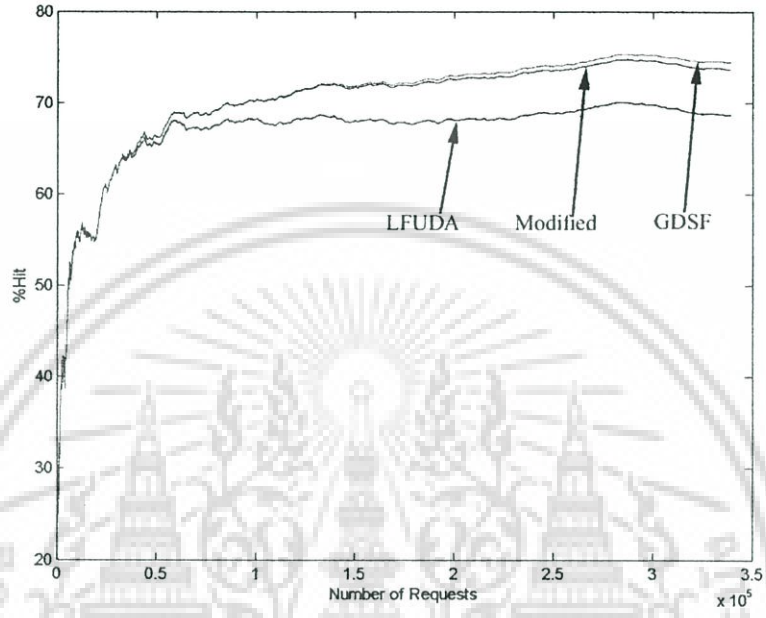
รูปที่ 4.10 กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 100 Kbytes



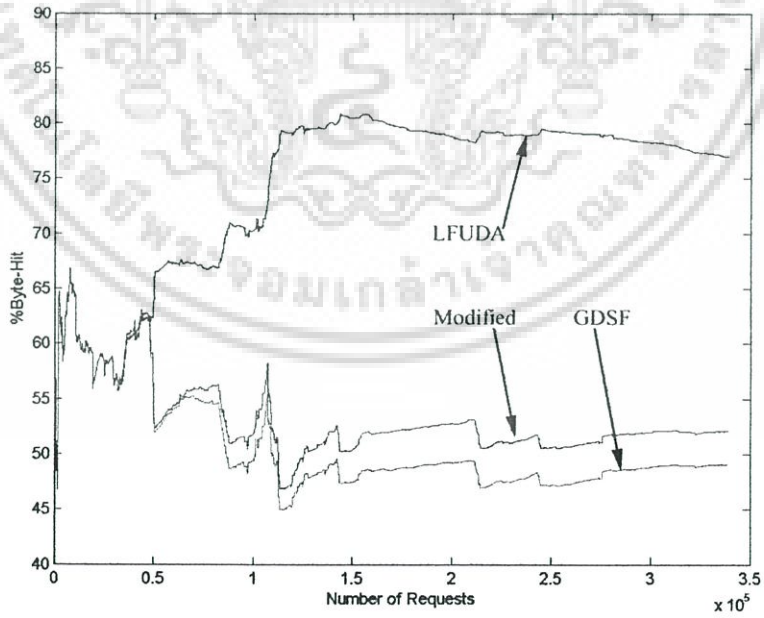
รูปที่ 4.11 กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 100 Kbytes

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.5.3 การเปรียบเทียบ อัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 1 Mbytes ซึ่งใช้ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง โดยตั้งแคชที่ขนาด 64 Mbytes



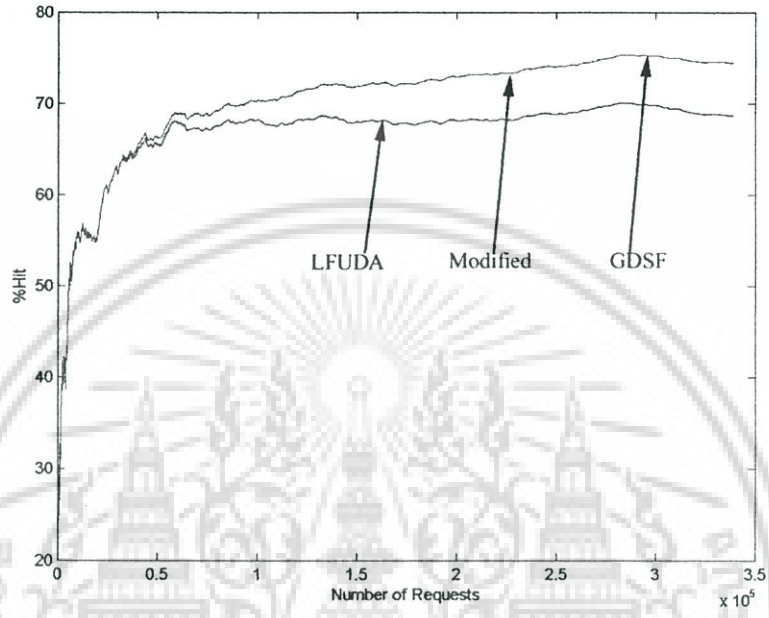
รูปที่ 4.12 กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 1 Mbytes



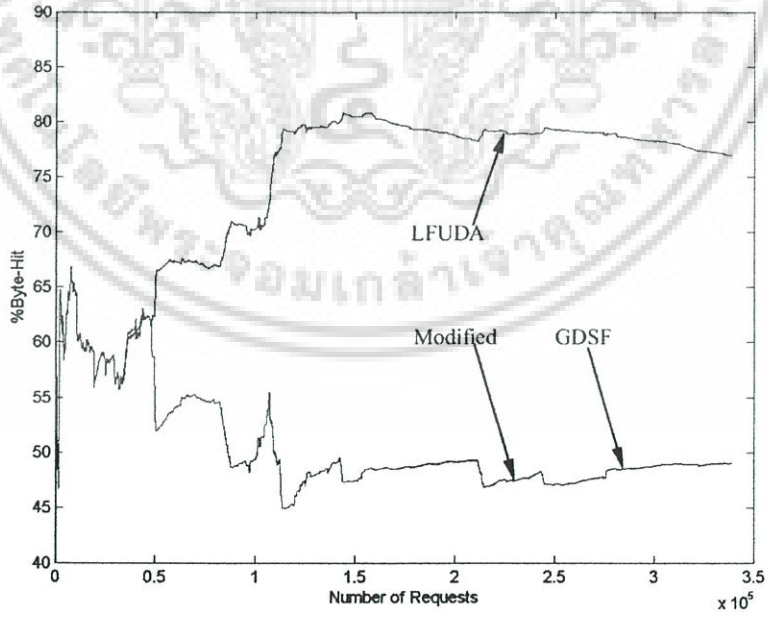
รูปที่ 4.13 กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 1 Mbytes

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.5.4 การเปรียบเทียบ อัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 10 Mbytes ซึ่งใช้ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง โดยตั้งแคชที่ขนาด 64 Mbytes



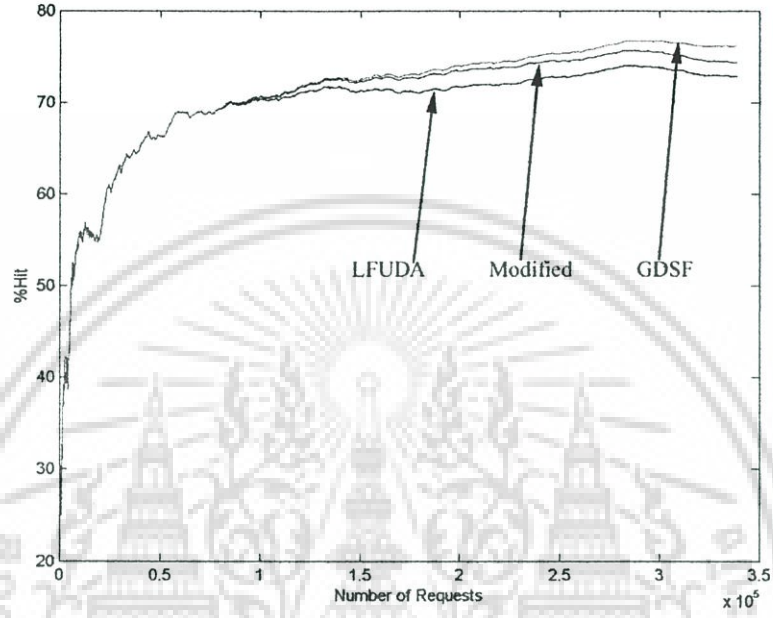
รูปที่ 4.14 กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 10 Mbytes



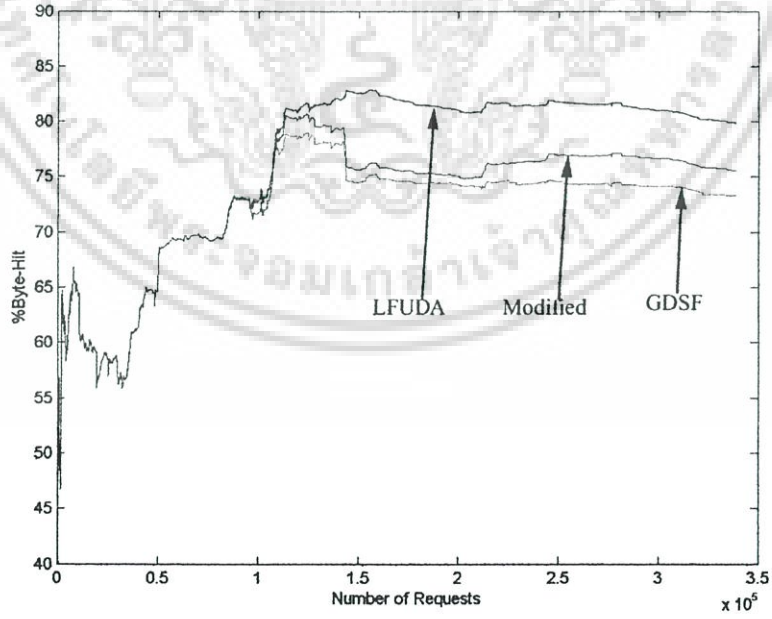
รูปที่ 4.15 กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 10 Mbytes

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.5.5 การเปรียบเทียบ อัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 10 Kbytes ซึ่งใช้ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง โดยตั้งแคชที่ขนาด 128 Mbytes



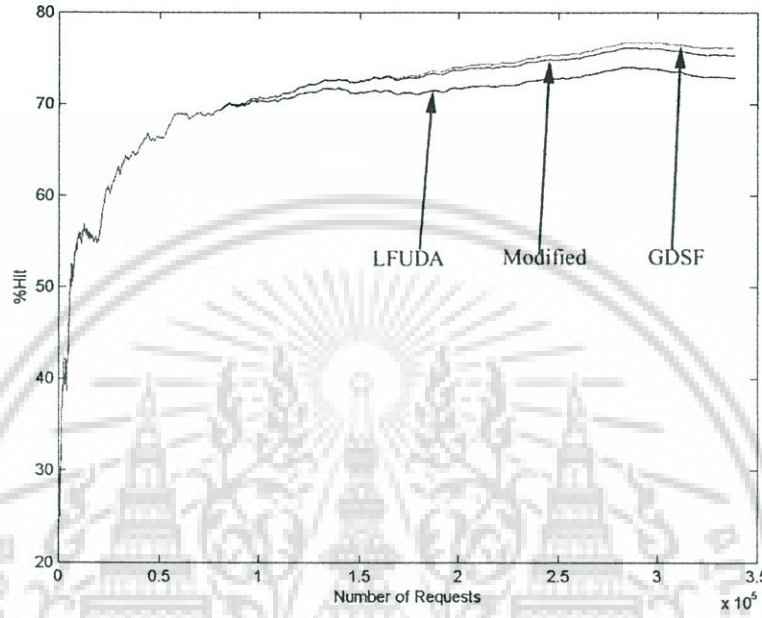
รูปที่ 4.16 กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 10 Kbytes



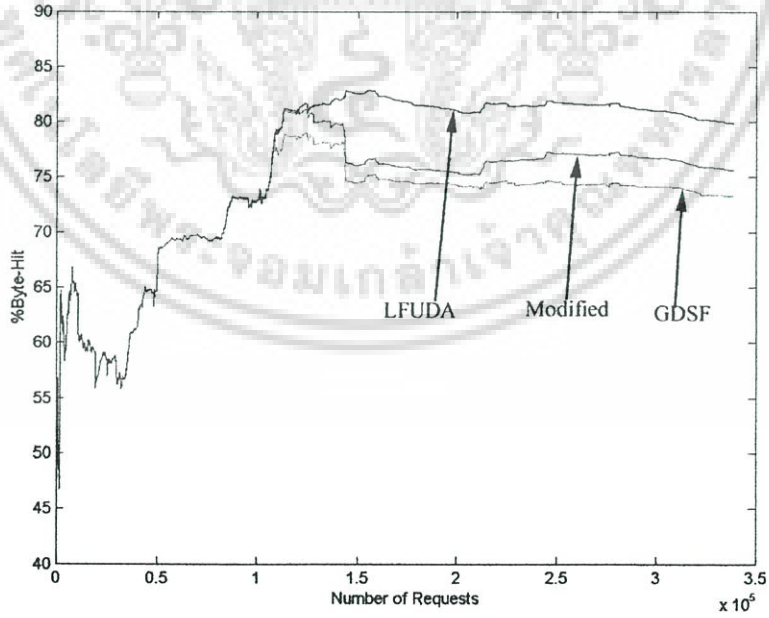
รูปที่ 4.17 กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 10 Kbytes

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.5.6 การเปรียบเทียบ อัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 100 Kbytes ซึ่งใช้ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง โดยตั้งแคชที่ขนาด 128 Mbytes



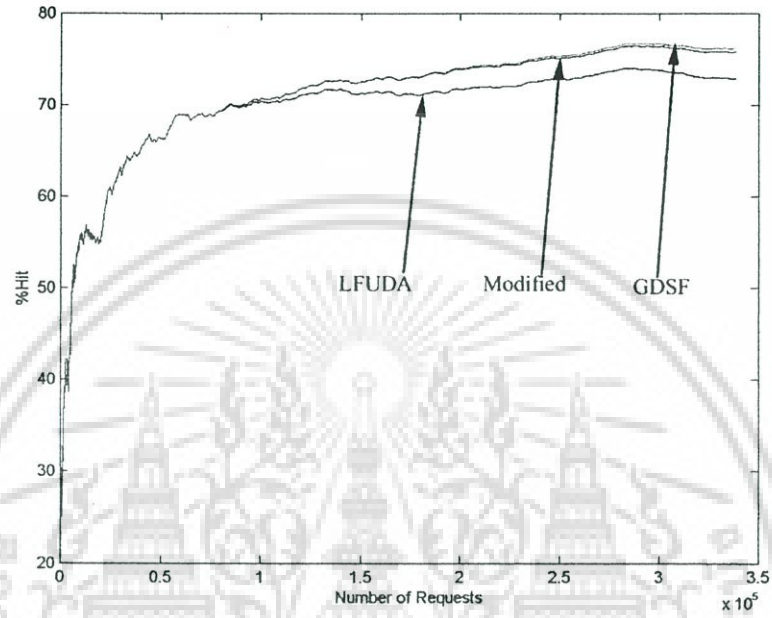
รูปที่ 4.18 กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 100 Kbytes



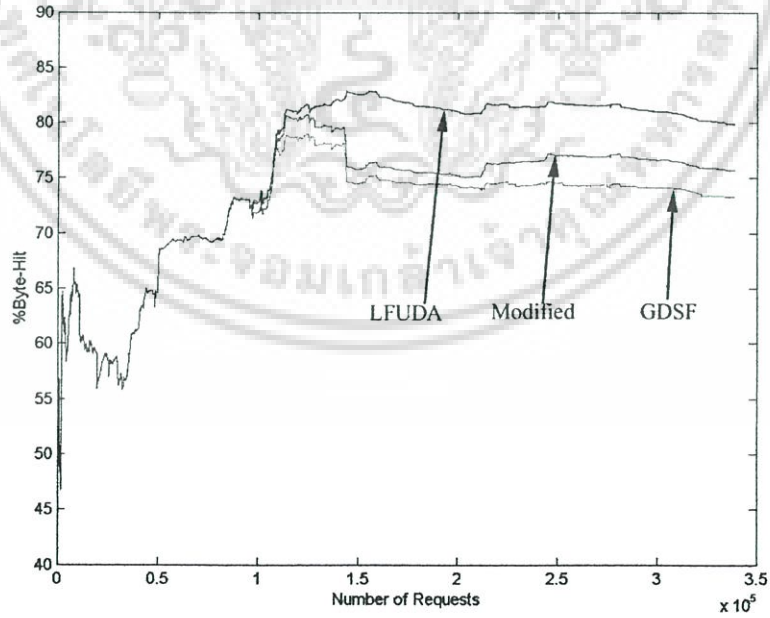
รูปที่ 4.19 กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 100 Kbytes

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.5.7 การเปรียบเทียบ อัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 1 Mbytes ซึ่งใช้ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง โดยตั้งแคชที่ขนาด 128 Mbytes



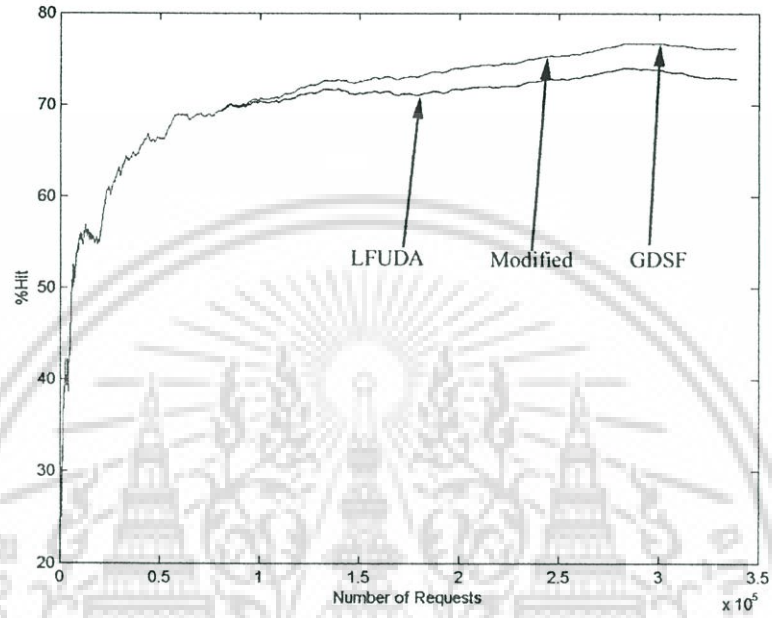
รูปที่ 4.20 กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 1 Mbytes



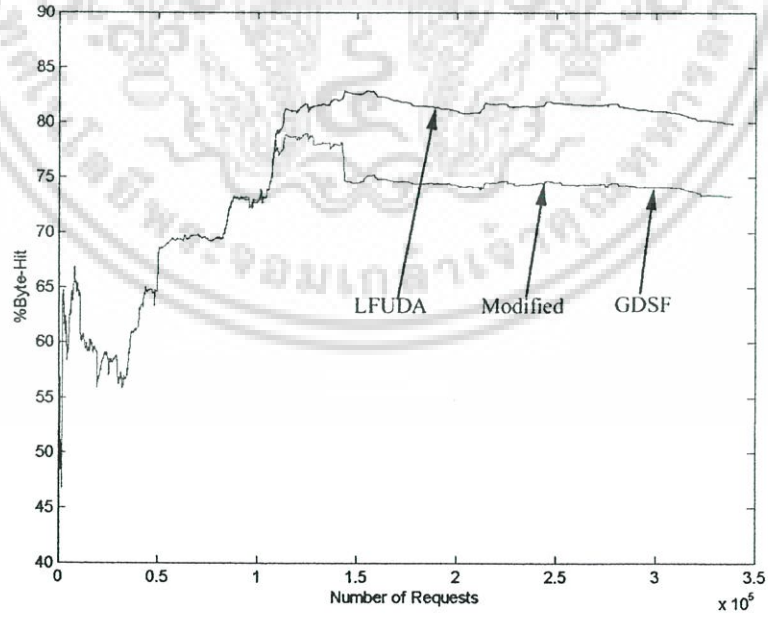
รูปที่ 4.21 กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 1 Mbytes

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.5.8 การเปรียบเทียบ อัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 10 Mbytes ซึ่งใช้ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง โดยตั้งแคชที่ขนาด 128 Mbytes



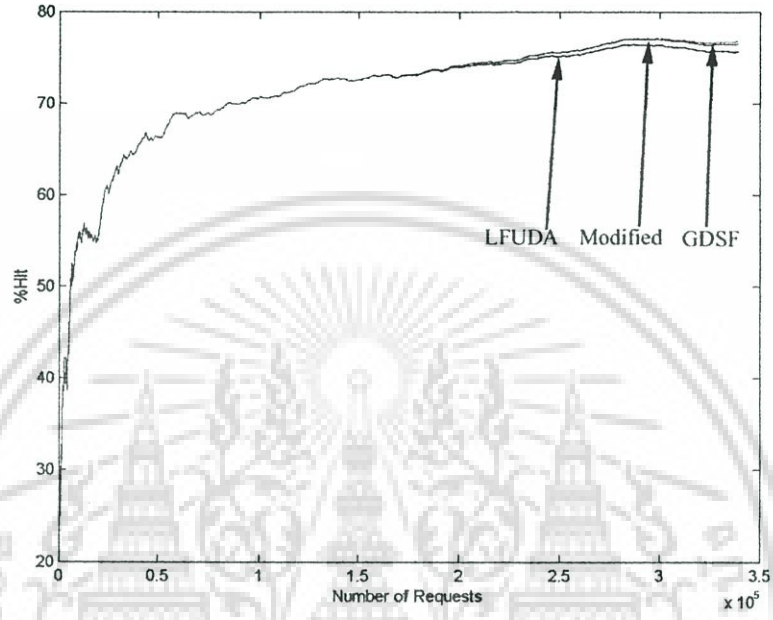
รูปที่ 4.22 กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า10 Mbytes



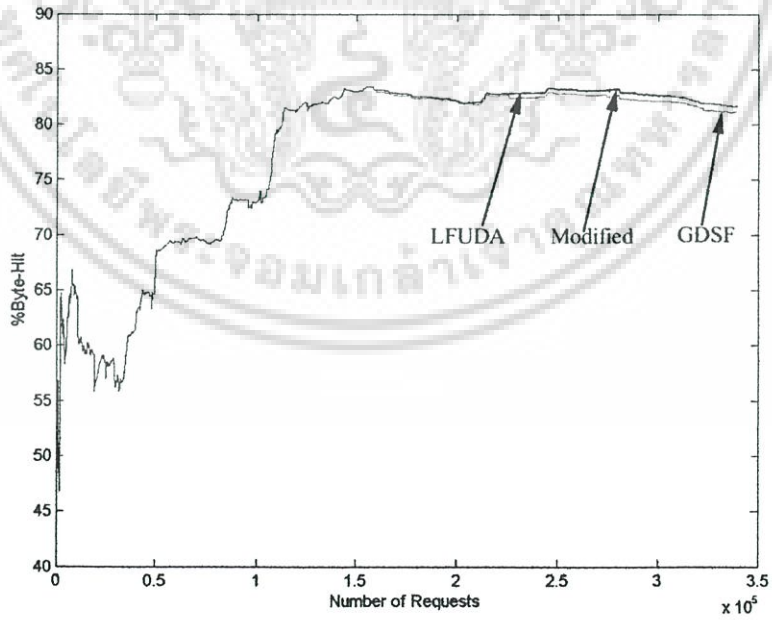
รูปที่ 4.23 กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 10 Mbytes

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.5.9 การเปรียบเทียบ อัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 10 Kbytes ซึ่งใช้ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง โดยตั้งแคชที่ขนาด 256 Mbytes



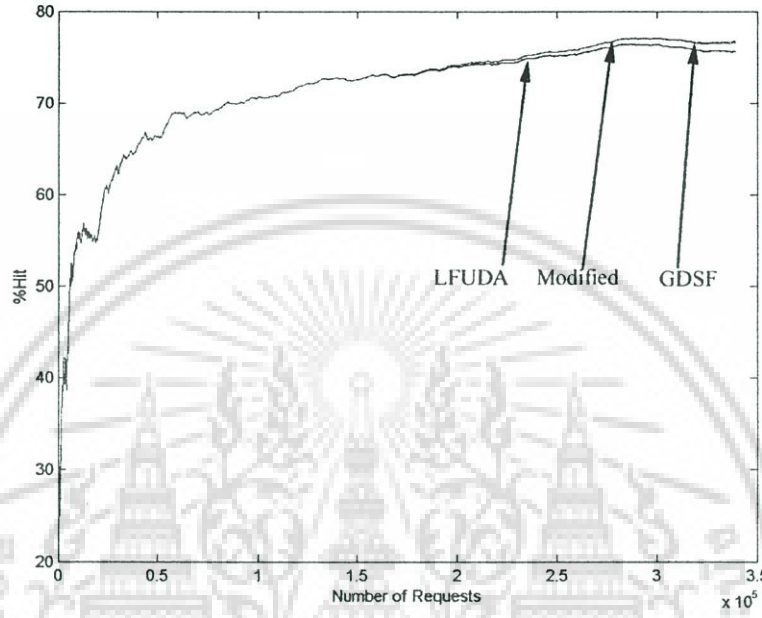
รูปที่ 4.24 กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 10 Kbytes



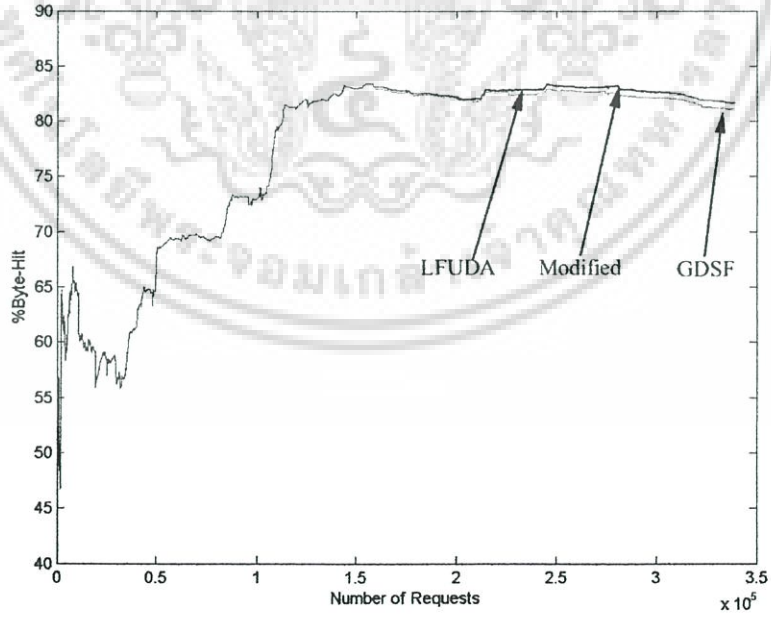
รูปที่ 4.25 กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 10 Kbytes

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.5.10 การเปรียบเทียบ อัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 100 Kbytes ซึ่งใช้ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง โดยตั้งแคชที่ขนาด 256 Mbytes



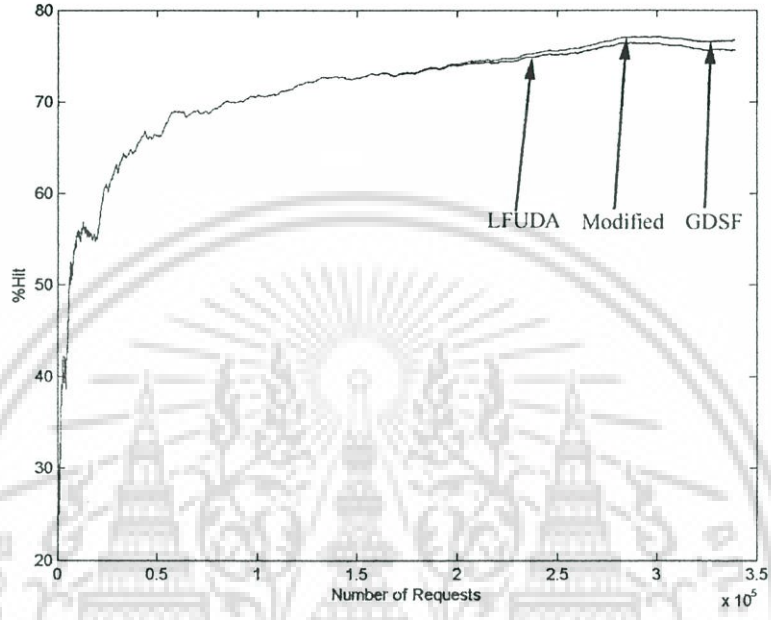
รูปที่ 4.26 กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 100 Kbytes



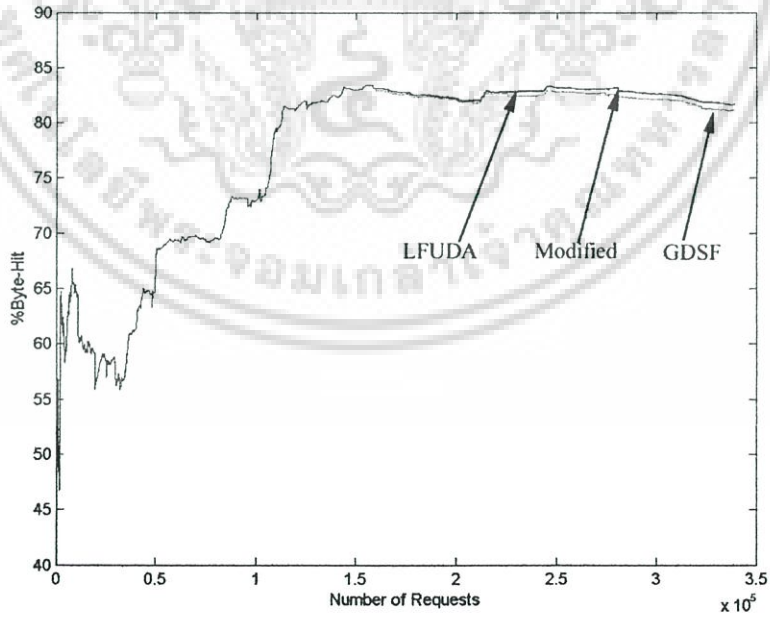
รูปที่ 4.27 กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 100 Kbytes

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.5.11 **การเปรียบเทียบ อัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 1 Mbytes ซึ่งใช้ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง โดยตั้งแคชที่ขนาด 256 Mbytes**



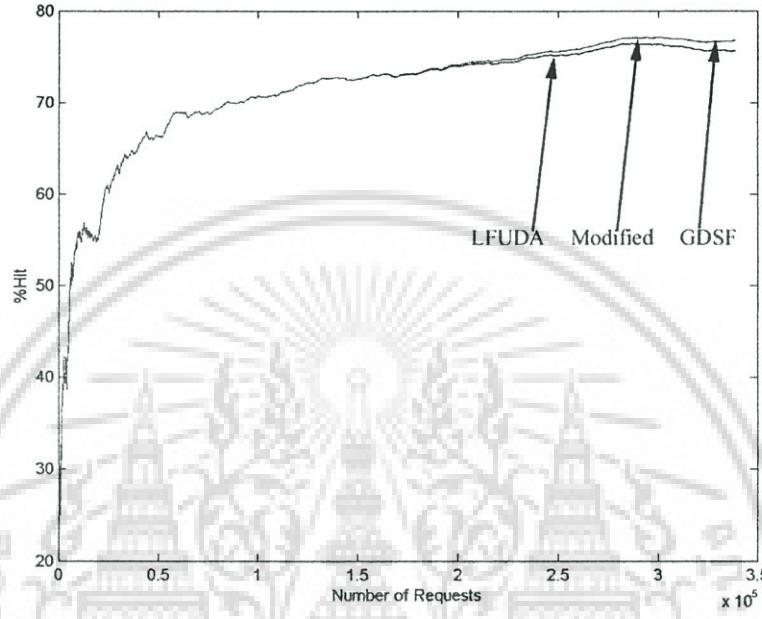
รูปที่ 4.28 กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 1 Mbytes



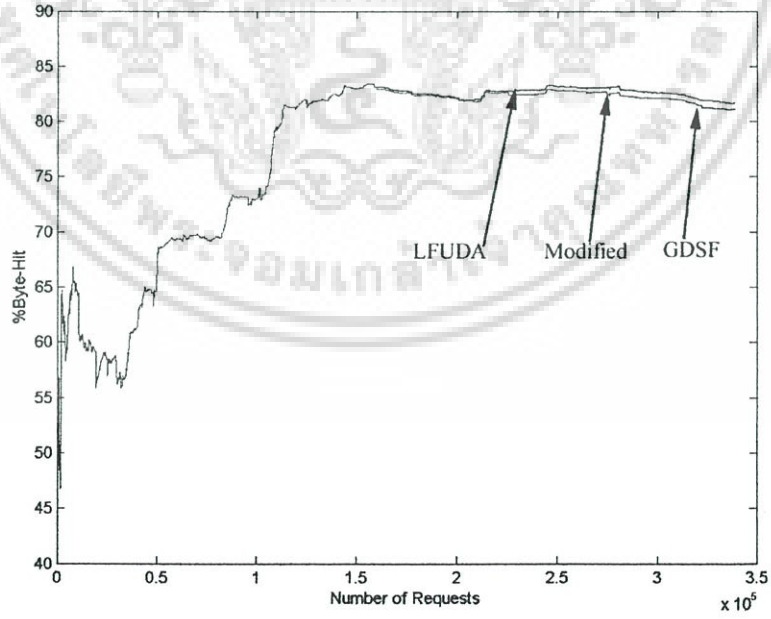
รูปที่ 4.29 กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 1 Mbytes

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.5.12 **การเปรียบเทียบ อัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 10 Mbytes ซึ่งใช้ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง โดยตั้งแคชที่ขนาด 256 Mbytes**



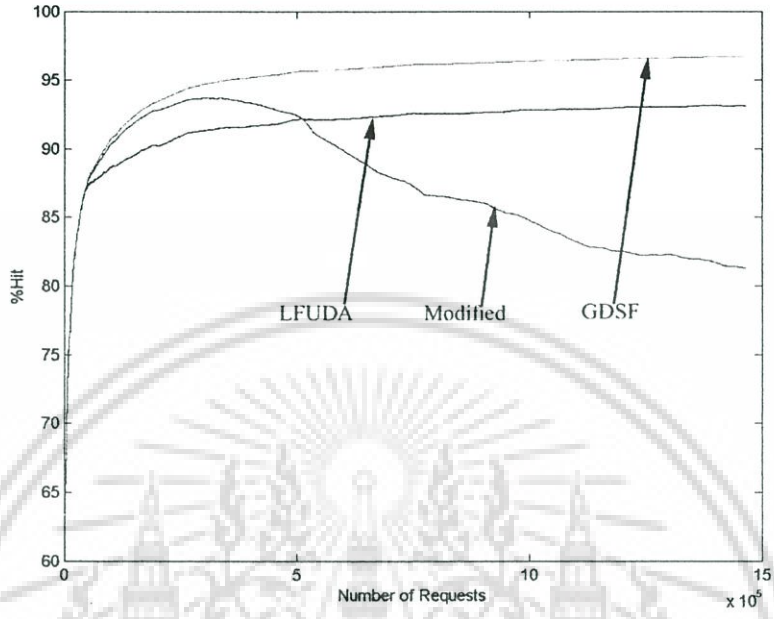
รูปที่ 4.30 กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 10 Mbytes



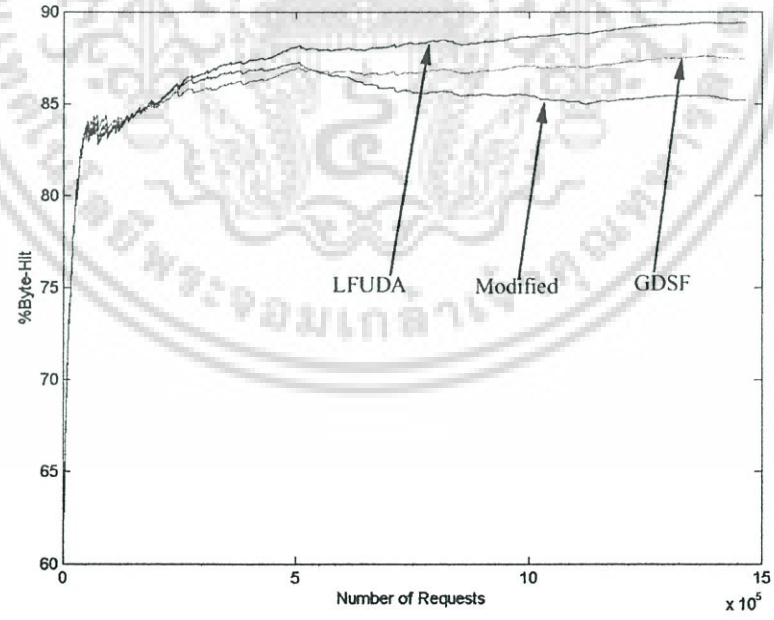
รูปที่ 4.31 กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 10 Mbytes

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.5.13 การเปรียบเทียบ อัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 10 Kbytes ซึ่งใช้ข้อมูลจาก Clarknet โดยตั้งแคชที่ขนาด 64 Mbytes



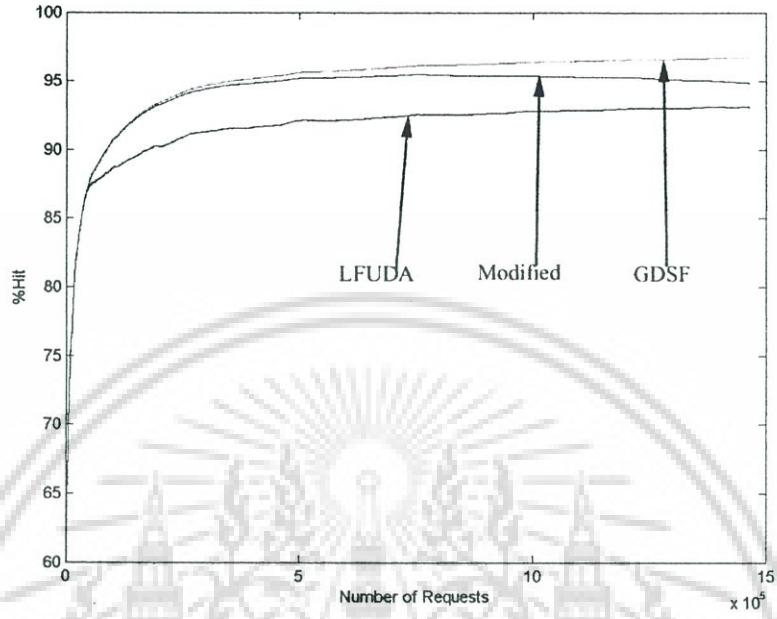
รูปที่ 4.32 กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 10 Kbytes



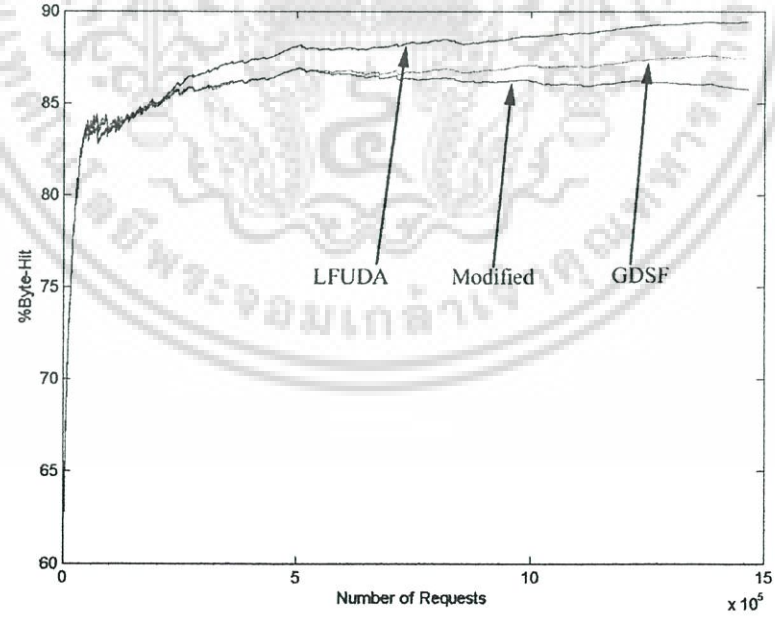
รูปที่ 4.33 กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 10 Kbytes

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.5.14 การเปรียบเทียบ อัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 100 Kbytes ซึ่งใช้ข้อมูลจาก Clarknet โดยตั้งแคชที่ขนาด 64 Mbytes



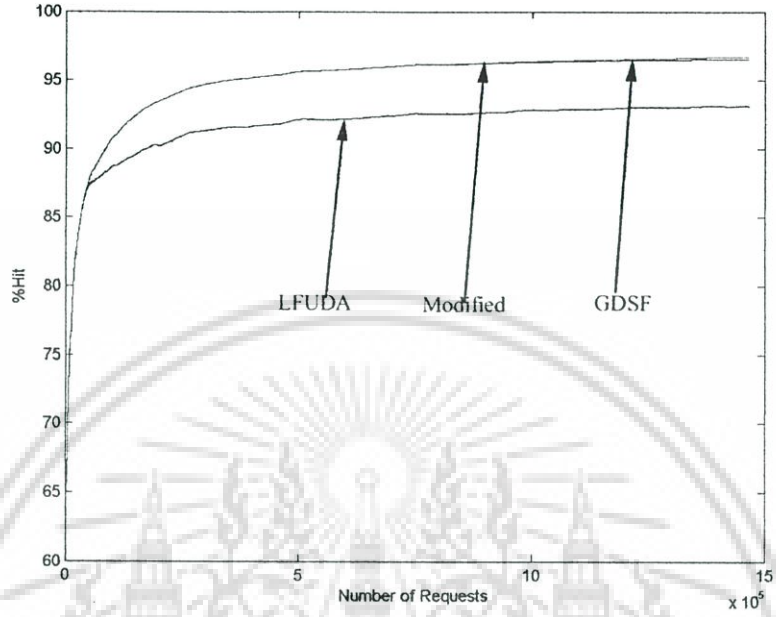
รูปที่ 4.34 กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 100 Kbytes



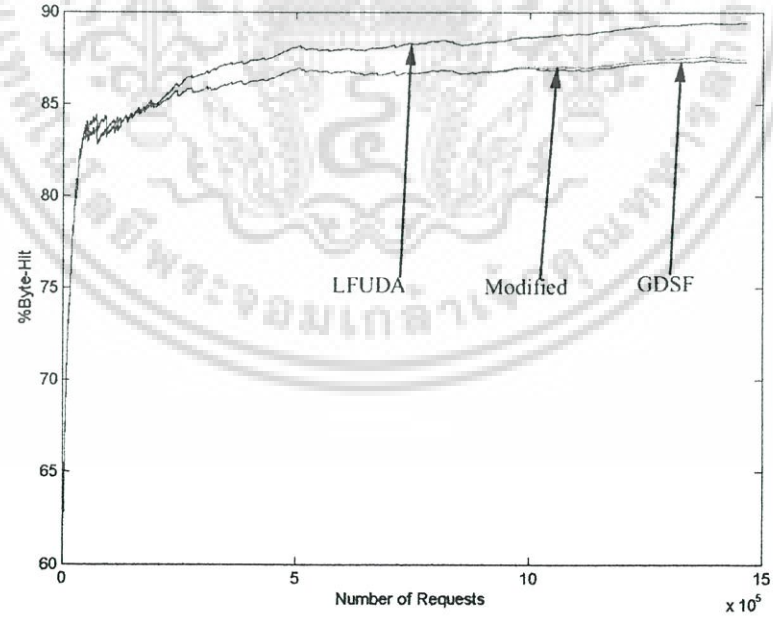
รูปที่ 4.35 กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 100 Kbytes

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.5.15 การเปรียบเทียบ อัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 1 Mbytes ซึ่งใช้ข้อมูลจาก Clarknet โดยตั้งแคชที่ขนาด 64 Mbytes



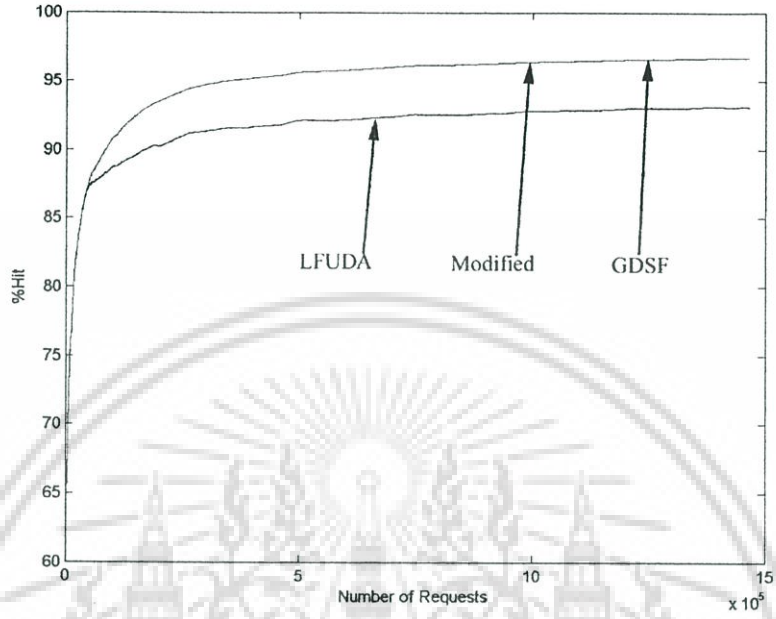
รูปที่ 4.36 กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 1 Mbytes



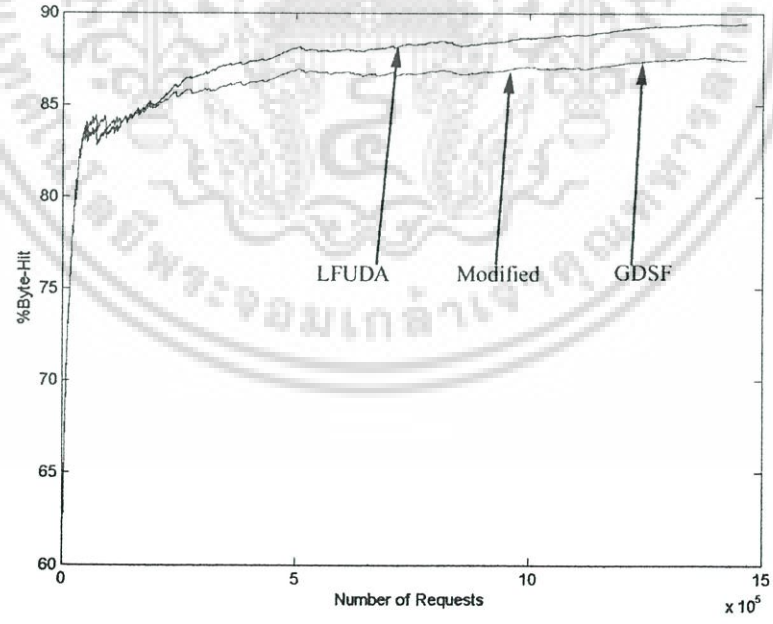
รูปที่ 4.37 กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 1 Mbytes

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.5.16 การเปรียบเทียบ อัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 10 Mbytes ซึ่งใช้ข้อมูลจาก Clarknet โดยตั้งแคชที่ขนาด 64 Mbytes



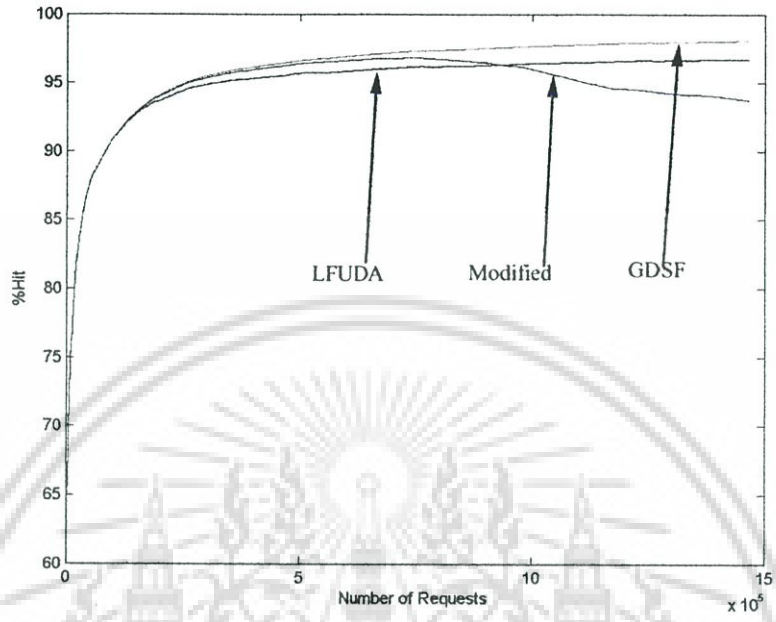
รูปที่ 4.38 กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 10 Mbytes



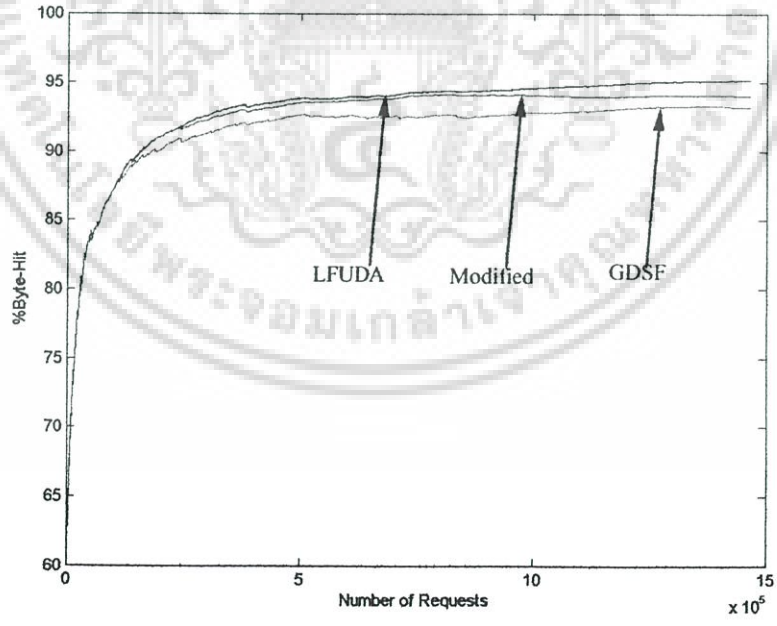
รูปที่ 4.39 กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 10 Mbytes

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.5.17 การเปรียบเทียบ อัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 10 Kbytes ซึ่งใช้ข้อมูลจาก Clarknet โดยตั้งแคชที่ขนาด 128 Mbytes



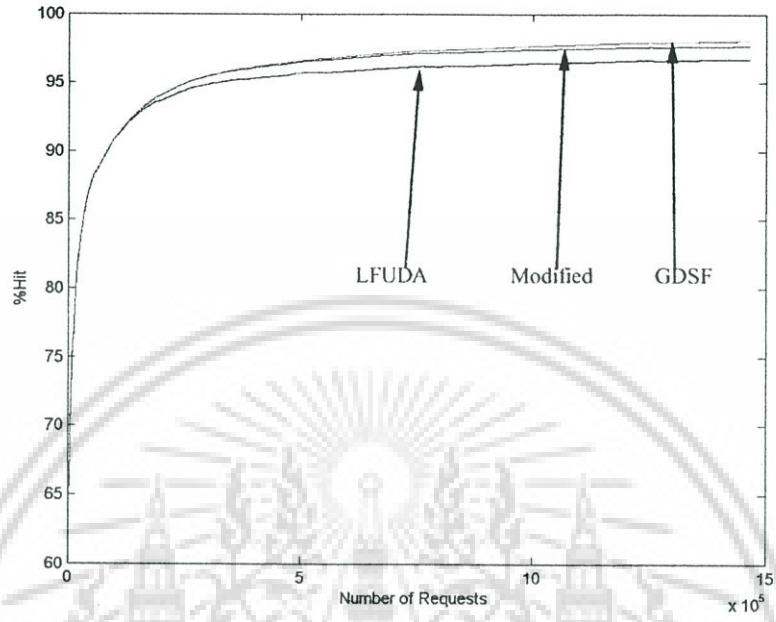
รูปที่ 4.40 กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 10 Kbytes



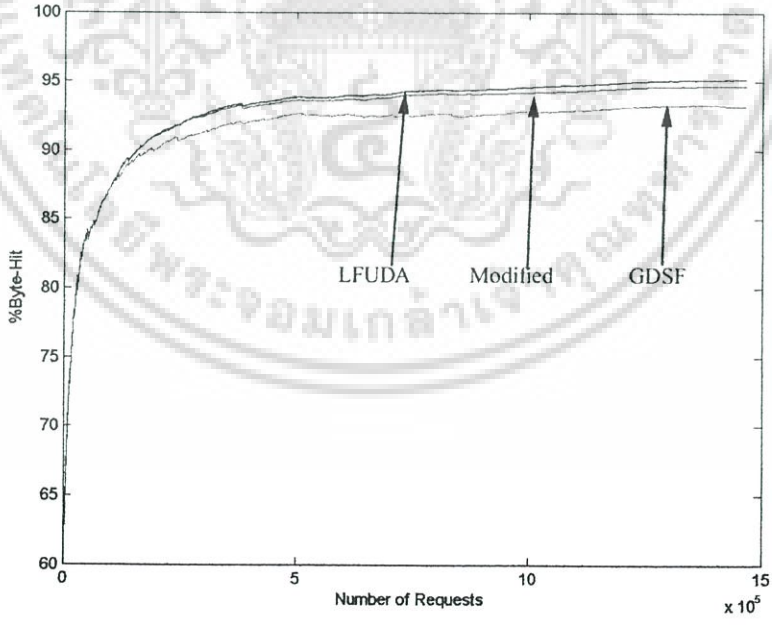
รูปที่ 4.41 กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 10 Kbytes

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.5.18 การเปรียบเทียบ อัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 100 Kbytes ซึ่งใช้ข้อมูลจาก Clarknet โดยตั้งแคชที่ขนาด 128 Mbytes



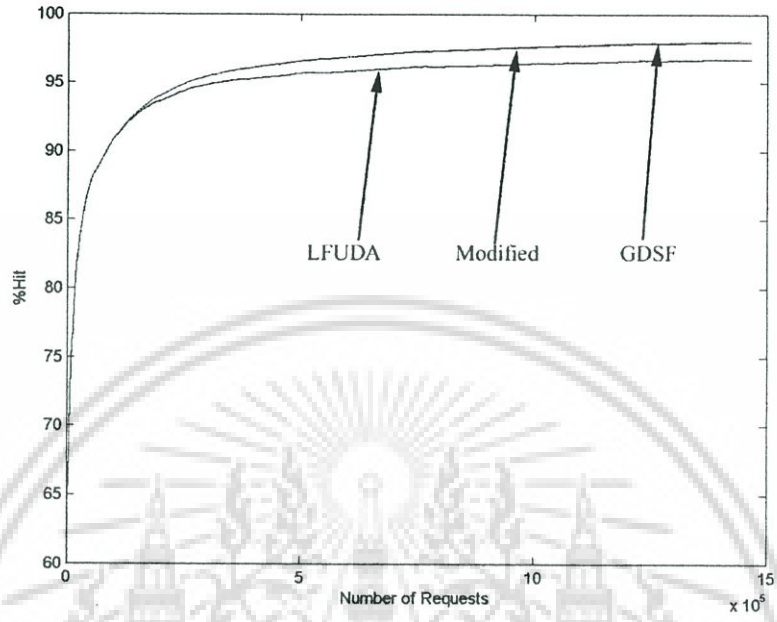
รูปที่ 4.42 กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 100 Kbytes



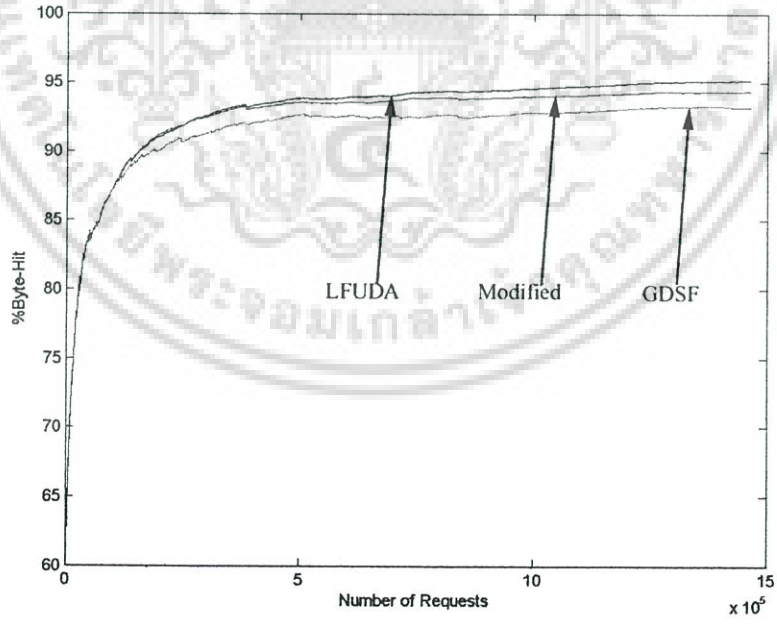
รูปที่ 4.43 กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 100 Kbytes

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.5.19 การเปรียบเทียบ อัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 1 Mbytes ซึ่งใช้ข้อมูลจาก Clarknet โดยตั้งแคชที่ขนาด 128 Mbytes



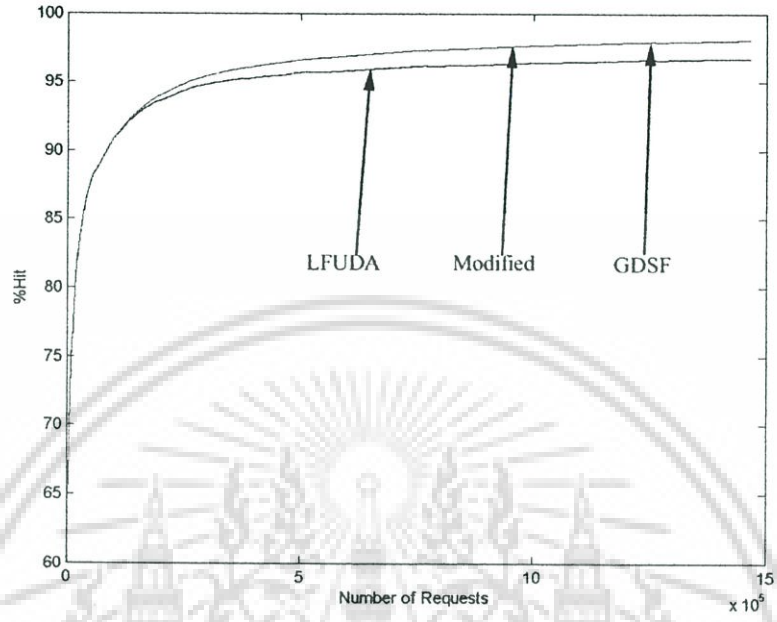
รูปที่ 4.44 กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 1 Mbytes



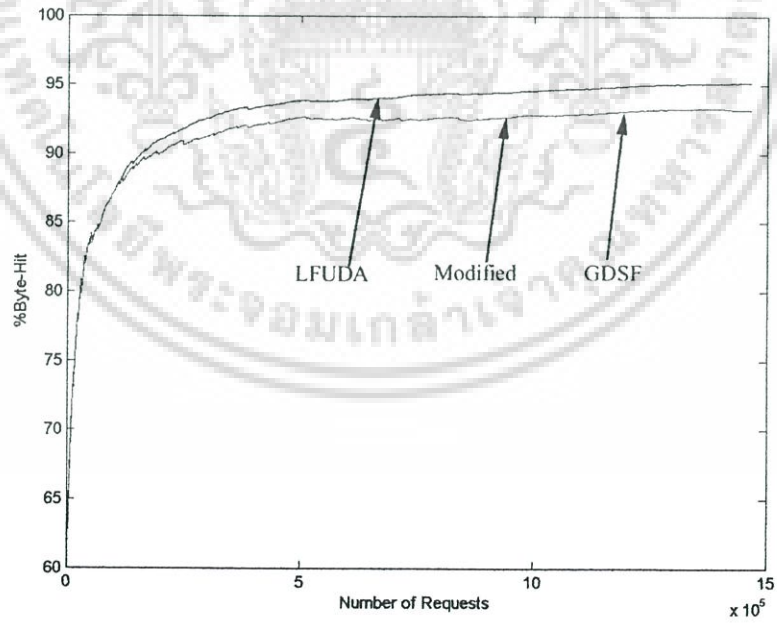
รูปที่ 4.45 กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 1 Mbytes

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.5.20 การเปรียบเทียบ อัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 10 Mbytes ซึ่งใช้ข้อมูลจาก Clarknet โดยตั้งแคชที่ขนาด 128 Mbytes



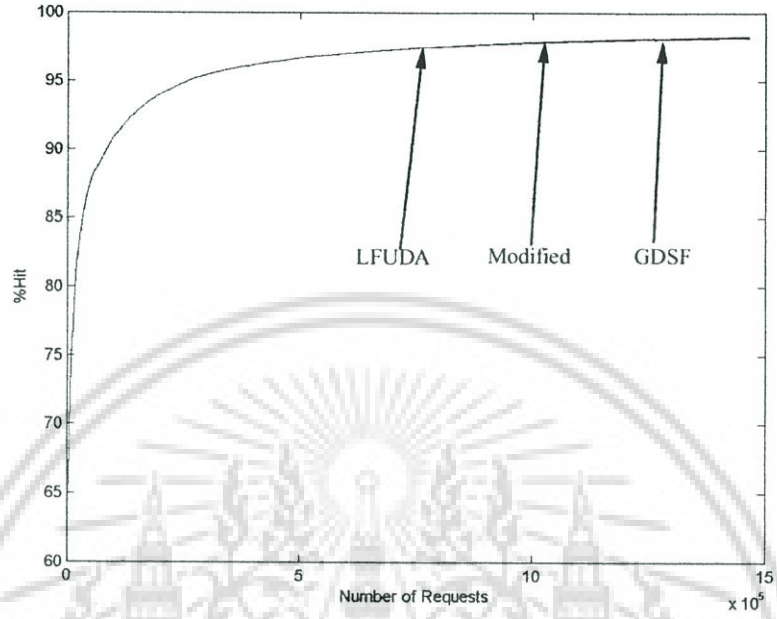
รูปที่ 4.46 กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 10 Mbytes



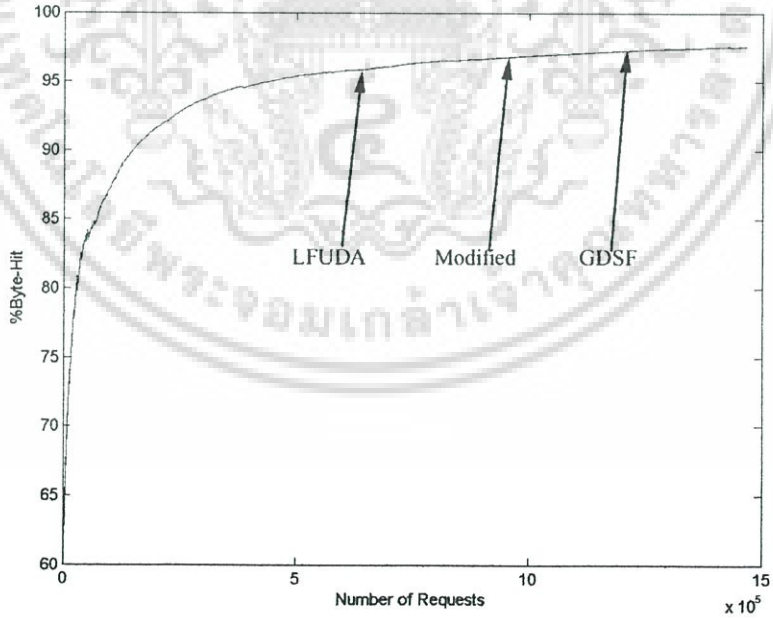
รูปที่ 4.47 กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 10 Mbytes

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.5.21 การเปรียบเทียบ อัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 10 Kbytes ซึ่งใช้ข้อมูลจาก Clarknet โดยตั้งแคชที่ขนาด 256 Mbytes



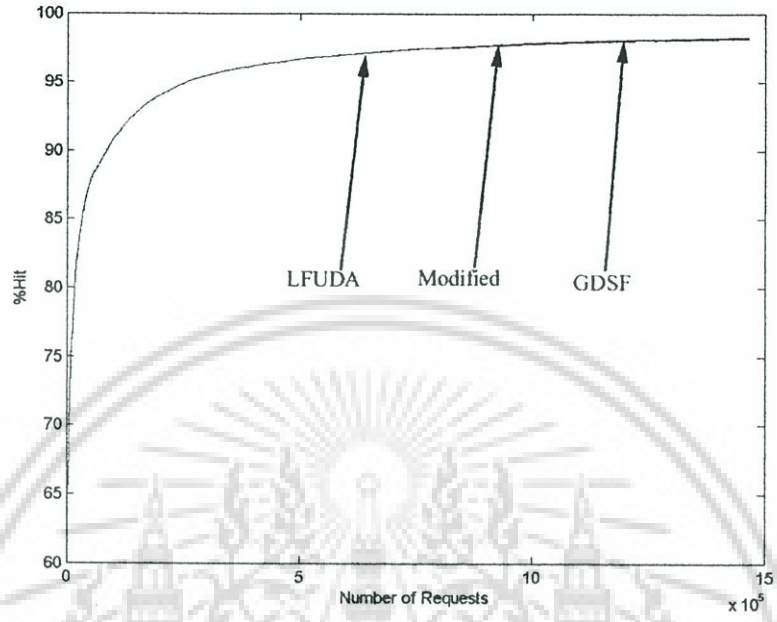
รูปที่ 4.48 กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 10 Kbytes



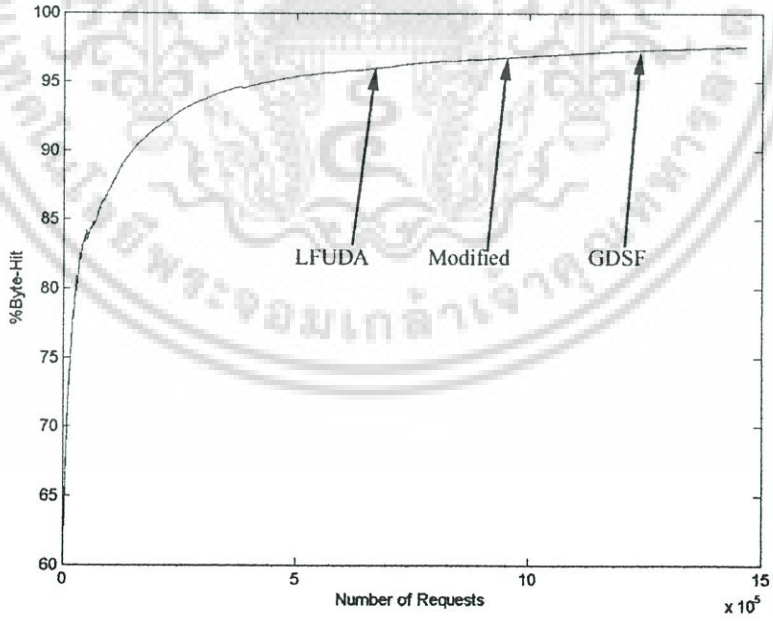
รูปที่ 4.49 กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 10 Kbytes

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.5.22 การเปรียบเทียบ อัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 100 Kbytes ซึ่งใช้ข้อมูลจาก Clarknet โดยตั้งแคชที่ขนาด 256 Mbytes



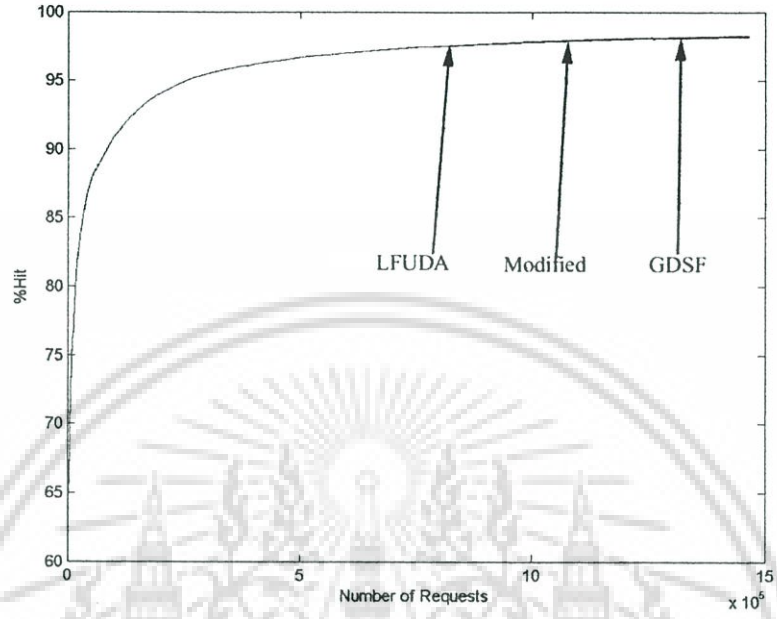
รูปที่ 4.50 กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 100 Kbytes



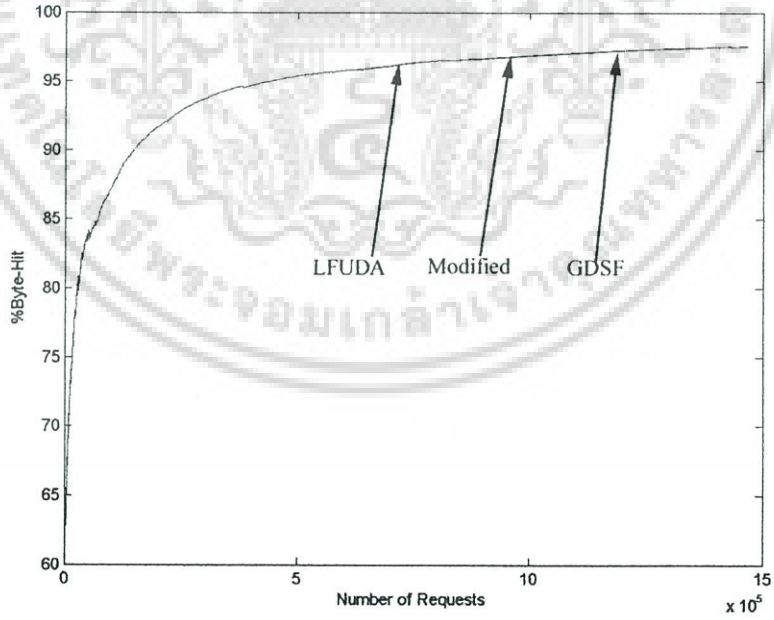
รูปที่ 4.51 กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 100 Kbytes

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.5.23 การเปรียบเทียบ อัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 1 Mbytes ซึ่งใช้ข้อมูลจาก Clarknet โดยตั้งแคชที่ขนาด 256 Mbytes



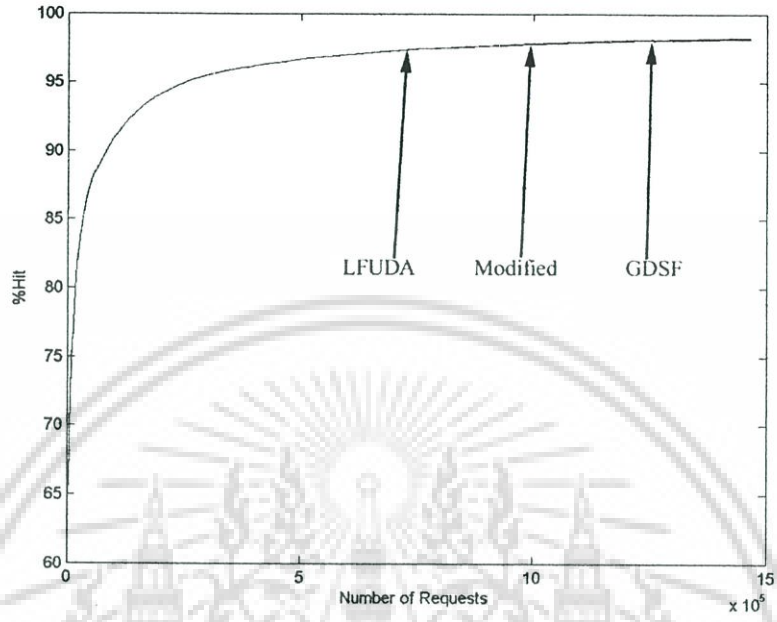
รูปที่ 4.52 กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 1 Mbytes



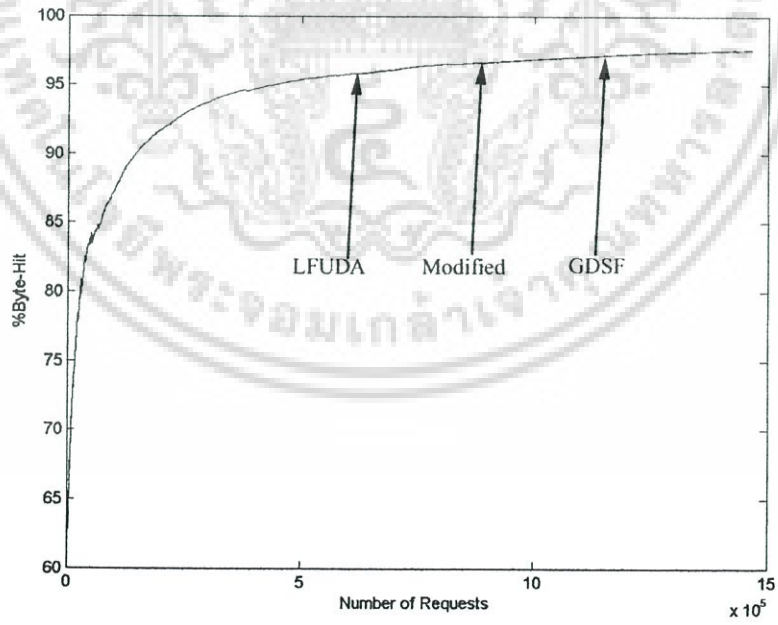
รูปที่ 4.53 กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 1 Mbytes

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.5.24 การเปรียบเทียบ อัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 10 Mbytes ซึ่งใช้ข้อมูลจาก Clarknet โดยตั้งแคชที่ขนาด 256 Mbytes



รูปที่ 4.54 กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 10 Mbytes



รูปที่ 4.55 กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 10 Mbytes

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.6 การวิเคราะห์ผลการทดสอบ

4.6.1 การวิเคราะห์ผลการปรับเปลี่ยนขนาดของแคช

จากการปรับเปลี่ยนขนาดแคชของเว็บแคชเซิร์ฟเวอร์โดยตั้งค่าขนาดของแคชเป็น 64 Mbytes, 128 Mbytes และ 256 Mbytes ใช้ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าลาดกระบัง และข้อมูลจาก Clarknet จะสามารถวิเคราะห์ค่าของ %Hit และ %Byte-Hit แบ่งตามขนาดของแคชได้ดังนี้

1. ขนาดของแคชที่ 64 Mbytes เมื่อใช้ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าลาดกระบังจะให้ค่า %Byte-Hit ของอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF ต่ำมากเนื่องจาก ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าลาดกระบัง เมื่อรวมขนาดของข้อมูลที่ตัดข้อมูลที่ซ้ำทิ้งไปแล้วจะมีขนาดประมาณ 463 Mbytes ซึ่งเมื่อนำมาเทียบกับขนาดของแคชที่ 64 Mbytes แล้วทำให้แคชของเว็บแคชเซิร์ฟเวอร์ถูกใช้งานจนเต็มภายในการร้องขอเพียงไม่นาน ทำให้ไม่มีพื้นที่สำหรับเก็บข้อมูลที่มีการร้องขอบ่อยๆ จึงเป็นผลทำให้ค่า %Byte-Hit ของอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF มีค่าต่ำ ในส่วนข้อมูลของ Clarknet จะพบปัญหาเช่นเดียวกัน แต่ค่าไม่ลดต่ำลงมากเท่ากับข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าลาดกระบัง ซึ่งเพื่อพิจารณาโดยรวมแล้วค่าขนาดแคชที่ขนาด 64 Mbytes จะให้ค่าของ %Hit และ %Byte-Hit ที่ต่ำเนื่องจากพื้นที่ของแคชที่ไว้สำหรับเก็บข้อมูลมีขนาดเล็กเกินไป

2. ขนาดของแคชที่ 128 Mbytes เมื่อใช้ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าลาดกระบังและข้อมูลจาก Clarknet พบว่าค่าของ %Hit และ %Byte-Hit ที่ได้ของอัลกอริธึมการแทนที่ข้อมูลที่นำเสนอ จะมีค่าน้อยกว่าอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF และมากกว่าอัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA สำหรับค่า %Hit และ จะมีค่าน้อยกว่าอัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA และมากกว่าอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF สำหรับค่า %Byte-Hit ซึ่งแสดงผลการทำงานของอัลกอริธึมการแทนที่ข้อมูลได้อย่างชัดเจนที่สุด

3. ขนาดของแคชที่ 256 Mbytes เมื่อใช้ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าลาดกระบังและข้อมูลจาก Clarknet พบว่ากราฟของอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF, LFUDA และที่นำเสนอ ทั้งในส่วนของ %Hit และ %Byte-Hit แทบจะเห็นความแตกต่างกันน้อยมากจนถึงไม่ต่างกันเลย เนื่องจากขนาดของแคชที่มีค่า 256 Mbytes จะสามารถเก็บข้อมูลไว้ได้มาก ซึ่งทำให้มีการเรียกใช้อัลกอริธึมการแทนที่ข้อมูลน้อยจึงทำให้ข้อมูลที่ถูกเก็บภายในแคชมีข้อมูลที่คล้ายหรือใกล้เคียงกัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.6.2 **การวิเคราะห์ผลการปรับเปลี่ยนค่า** Threshold

จากผลการทดสอบที่ขนาดของแคชมีค่า 128 Mbytes เนื่องจากเป็นขนาดของแคชที่แสดงให้เห็นการทำงานของอัลกอริธึมการแทนที่ข้อมูลชัดเจนที่สุด โดยการใช้ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าลาดกระบัง และข้อมูลจาก Clarknet พบว่าเมื่อทำการปรับแต่งอัลกอริธึมการแทนที่ข้อมูลตามค่า Threshold ที่ได้กำหนดขึ้นมาจำนวน 4 ค่าคือ 10 Kbytes, 100 Kbytes, 1 Mbytes และ 10 Mbytes ค่า %Hit ที่ได้เมื่อเทียบกับอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF จะมีค่าลดลงมา แต่จะมีผลทำให้ค่าของ %Byte-Hit มีค่าเพิ่มขึ้นและลดลงขึ้นกับค่า Threshold ที่กำหนด ซึ่งสามารถอธิบายในค่า Threshold แต่ละค่าได้ดังนี้

1. ค่า Threshold มีค่า 10 Kbytes พบว่าข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าลาดกระบัง จะให้ค่าของ %Hit และ %Byte-Hit อยู่ระหว่าง %Hit และ %Byte-Hit ของอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF และอัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA ซึ่งเป็นไปตามที่นำเสนอ แต่เมื่อพิจารณาข้อมูลที่ได้จาก Clarknet พบว่าค่าของ %Hit จะลดต่ำลงมาจาก %Hit ของอัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA และ %Byte-Hit จะมีแนวโน้มที่จะลดต่ำลงมาจาก %Byte-Hit ของอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF เนื่องจาก ข้อมูลของ Clarknet จะมีข้อมูลที่มีขนาดน้อยกว่า 10 Kbytes เป็นจำนวนมากเทียบกับข้อมูลขนาดอื่น ๆ ทำให้เนื้อที่ส่วนใหญ่ถูกนำไปเก็บข้อมูลที่มีขนาดเกินกว่าค่า Threshold ดังนั้นพื้นที่แคชของเว็บแคชเซิร์ฟเวอร์จึงเหลือน้อยสำหรับการเก็บข้อมูลที่มีขนาดน้อยกว่าค่า Threshold

2. ค่า Threshold มีค่า 100 Kbytes พบว่าข้อมูลที่ได้จากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังและข้อมูลจาก Clarknet จะมีค่า %Byte-Hit ของอัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งสูงกว่าอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF และมีค่าเข้าใกล้กับอัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA

3. ค่า Threshold มีค่า 1 Mbytes พบว่าข้อมูลทั้ง 2 ชุดมีค่า %Byte-Hit เพิ่มขึ้นเนื่องจากข้อมูลขนาดใหญ่กว่า 1 Mbytes ที่มีการร้องขอบ่อย ๆ เท่านั้นที่จะถูกเก็บไว้ในพื้นที่แคชของเว็บแคชเซิร์ฟเวอร์ ซึ่งการร้องขอข้อมูลที่มีขนาดเกิน 1 Mbytes ของข้อมูลทั้ง 2 ชุดจะมีเป็นจำนวนน้อยเมื่อเทียบกับการร้องขอทั้งหมด ดังนั้นเมื่อมีการร้องขอข้อมูลขนาดใหญ่จึงมีโอกาสที่จะพบข้อมูลภายในแคช

4. ค่า Threshold มีค่า 10 Mbytes พบว่าข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังจะให้ค่าของ %Hit และ %Byte-Hit มีค่าใกล้เคียงกับค่าที่ได้จากอัลกอริธึมการแทนที่แบบ GDSF มาก เนื่องจากข้อมูลมีข้อมูลที่ขนาดใหญ่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เกินกว่า 10 Mbytes เป็นจำนวนน้อยมาก ทำให้ข้อมูลที่มีขนาดใหญ่ถูกเก็บไว้ในแคชเพียงไม่นานแล้วก็จะถูกนำออกไปทำให้การทำงานของอัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งมีการทำงานใกล้เคียงกับการทำงานของอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF และสำหรับข้อมูลจาก Clarknet พบว่ากราฟที่ได้ของ %Hit และ %Byte-Hit ไม่มีความแตกต่างกันเลย เนื่องจากข้อมูลของ Clarknet จะไม่มีข้อมูลที่มีขนาดเกินกว่า 10 Mbytes เลยทำให้การทำงานของอัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งมีการทำงานเหมือนกับอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF

4.7 การวิเคราะห์หาค่า Threshold ที่เหมาะสม

จากผลการทดสอบของอัลกอริธึมการแทนที่ข้อมูลที่นำเสนอค่าของ %Hit และ %Byte-Hit ที่ได้จากการปรับค่า Threshold เป็นค่า 10 Kbytes, 100 Kbytes, 1 Mbytes และ 10 Mbytes จะพบว่าค่าของ %Hit และ %Byte-Hit ไม่สามารถที่จะแสดงออกมาในรูปของสมการทางคณิตศาสตร์ได้ จึงไม่สามารถที่จะหาค่า Threshold ที่เหมาะสมได้ด้วยสมการทางคณิตศาสตร์ ดังนั้นการวิเคราะห์หาค่า Threshold ที่เหมาะสมจะกระทำได้ด้วยการปรับเปลี่ยนค่า Threshold ไปเรื่อยๆ และทำการพิจารณาถึงค่าของ %Hit และ %Byte-Hit ที่ได้จากการปรับเปลี่ยนค่า Threshold ไปเรื่อยๆ โดยการพิจารณาจะถูกพิจารณาตามเงื่อนไข 4 ข้อนี้

1. ในกรณีของ %Hit ค่า %Hit ของอัลกอริธึมการแทนที่ข้อมูลที่นำเสนอจะต้องลดต่ำกว่า %Hit ของอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF น้อยที่สุด ($\overline{\Delta H_i}(gdsf)$ min)
2. ในกรณีของ %Hit ค่า %Hit ของอัลกอริธึมการแทนที่ข้อมูลที่นำเสนอจะต้องเพิ่มขึ้นกว่า %Hit ของอัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA มากที่สุด ($\overline{\Delta H_i}(lfuda)$ max)
3. ในกรณีของ %Byte-Hit ค่า %Byte-Hit ของอัลกอริธึมการแทนที่ข้อมูลที่นำเสนอจะต้องเพิ่มขึ้นกว่า %Byte-Hit ของอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF มากที่สุด ($\overline{\Delta H_b}(gdsf)$ max)
4. ในกรณีของ %Byte-Hit ค่า %Byte-Hit ของอัลกอริธึมการแทนที่ข้อมูลที่นำเสนอจะต้องลดลงกว่า %Byte-Hit ของอัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA น้อยที่สุด ($\overline{\Delta H_b}(lfuda)$ min)

โดยค่า $\overline{\Delta H_i}(gdsf)$, $\overline{\Delta H_i}(lfuda)$, $\overline{\Delta H_b}(gdsf)$ และ $\overline{\Delta H_b}(lfuda)$ สามารถคำนวณได้ดังสมการ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$\overline{\Delta H_i}(gdsf)$:

$$\Delta H_i(gdsf)_n = \frac{\%Hit_{gdsf} - \%Hit_{modified}}{\%Hit_{gdsf}} \tag{4.3}$$

$$\overline{\Delta H_i}(gdsf) = \frac{\sum_{n=1}^{N} \Delta H_i(gdsf)_n}{N} \tag{4.4}$$

$\overline{\Delta H_i}(lfuda)$:

$$\Delta H_i(lfuda)_n = \frac{\%Hit_{modified} - \%Hit_{lfuda}}{\%Hit_{lfuda}} \tag{4.5}$$

$$\overline{\Delta H_i}(lfuda) = \frac{\sum_{n=1}^{N} \Delta H_i(lfuda)_n}{N} \tag{4.6}$$

$\overline{\Delta H_b}(gdsf)$:

$$\Delta H_b(gdsf)_n = \frac{\%Byte_{modified} - \%Byte_{gdsf}}{\%Byte_{gdsf}} \tag{4.7}$$

$$\overline{\Delta H_b}(gdsf) = \frac{\sum_{n=1}^{N} \Delta H_b(gdsf)_n}{N} \tag{4.8}$$

$\overline{\Delta H_b}(lfuda)$:

$$\Delta H_b(lfuda)_n = \frac{\%Byte_{lfuda} - \%Byte_{modified}}{\%Byte_{lfuda}} \tag{4.9}$$

$$\overline{\Delta H_b}(lfuda) = \frac{\sum_{n=1}^{N} \Delta H_b(lfuda)_n}{N} \tag{4.10}$$

โดย

$\%Hit_{gdsf}$ คือ ค่า %Hit ของอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF

$\%Hit_{lfuda}$ คือ ค่า %Hit ของอัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA

$\%Byte_{gdsf}$ คือ ค่า %Byte-Hit ของอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$\%Byte_{lfuda}$ คือ ค่า %Byte-Hit ของอัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA

$\Delta H_i(gdsf)_n$ คือ ค่าผลต่างระหว่าง %Hit ของอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF กับ %Hit ของอัลกอริธึมการแทนที่ข้อมูลที่นำเสนอเทียบกับ %Hit ของอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF ในการร้องขอครั้งที่ n

$\Delta H_i(lfuda)_n$ คือ ค่าผลต่างระหว่าง %Hit ของอัลกอริธึมการแทนที่ข้อมูลที่นำเสนอกับ %Hit ของอัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA เทียบกับ %Hit ของอัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA ในการร้องขอครั้งที่ n

$\Delta H_b(gdsf)_n$ คือ ค่าผลต่างระหว่าง %Byte-Hit ของอัลกอริธึมการแทนที่ข้อมูลที่นำเสนอกับ %Byte-Hit ของอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF เทียบกับ %Byte-Hit ของอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF ในการร้องขอครั้งที่ n

$\Delta H_b(lfuda)_n$ คือ ค่าผลต่างระหว่าง %Byte-Hit ของอัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA กับ %Byte-Hit ของอัลกอริธึมการแทนที่ข้อมูลที่นำเสนอเทียบกับ %Byte-Hit ของอัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA ในการร้องขอครั้งที่ n

N คือ จำนวนการร้องขอทั้งหมด

ในการวิเคราะห์หาค่า Threshold ได้ทำการเปลี่ยนค่า Threshold ตั้งแต่ 10 Kbytes ไปจนถึง 5 Mbytes โดยการเพิ่มของค่า Threshold จะเพิ่มในอัตราส่วนแบบ Semi-log และหากค่าผลต่างเฉลี่ยที่ได้มีค่าติดลบจะถือว่าค่า Threshold นั้นให้ผลที่ต่ำกว่ามาตรฐาน จะไม่นำมาพิจารณาในการหาค่า Threshold และจากเงื่อนไขที่กำหนด 4 ข้อข้างบน เมื่อนำมาพิจารณากับค่าเฉลี่ยของอัลกอริธึมการแทนที่ข้อมูลแต่ละแบบ จะได้คุณลักษณะของค่า Threshold ที่เหมาะสมดังนี้

1. กรณีเมื่อเปรียบเทียบกับอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF เมื่อพิจารณาจากเงื่อนไขข้อที่ 1 และ 3 หากนำค่าเฉลี่ยผลต่างของ %Byte-Hit มาหักลบกับ ค่าเฉลี่ยผลต่างของ %Hit ผลของการหักลบที่ให้ค่ามากที่สุด ($\Delta gdsf$ (max)) จะแสดงถึงค่า Threshold ที่ให้ประสิทธิภาพมากที่สุด

$$\Delta gdsf = \overline{\Delta H_b(gdsf)} - \overline{\Delta H_i(gdsf)} \qquad (4.11)$$

2. กรณีเมื่อเปรียบเทียบกับอัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA เมื่อพิจารณาจากเงื่อนไขข้องที่ 2 และ 4 หากนำค่าเฉลี่ยผลต่างของ %Byte-Hit มาหักลบกับ ค่าเฉลี่ย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ผลต่างของ %Hit ผลของการหักลบที่ให้ค่าน้อยที่สุด ($\Delta lfuda$ (min)) จะแสดงถึงค่า Threshold ที่ให้ประสิทธิภาพมากที่สุด

$$\Delta lfuda = \overline{\Delta H_b}(lfuda) - \overline{\Delta H_i}(lfuda) \qquad (4.12)$$

ในการวิเคราะห์หาค่า Threshold ที่เหมาะสมจะใช้ข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าลาดกระบัง และข้อมูลจาก Clarknet โดยกำหนดขนาดของแคชไว้ที่ 128 Mbytes แสดงผลของ $\overline{\Delta H_i}(gdsf)$, $\overline{\Delta H_i}(lfuda)$, $\overline{\Delta H_b}(gdsf)$, $\overline{\Delta H_b}(lfuda)$, $\Delta gdsf$ และ $\Delta lfuda$ ได้ดังตารางที่ 4.1 และตารางที่ 4.2

ตารางที่ 4.1 แสดงค่า $\overline{\Delta H_i}(gdsf)$, $\overline{\Delta H_i}(lfuda)$, $\overline{\Delta H_b}(gdsf)$, $\overline{\Delta H_b}(lfuda)$, $\Delta gdsf$ และ $\Delta lfuda$ ของข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าลาดกระบัง

Threshold	$\overline{\Delta H_i}(gdsf)$	$\overline{\Delta H_b}(gdsf)$	$\Delta gdsf$	$\overline{\Delta H_i}(lfuda)$	$\overline{\Delta H_b}(lfuda)$	$\Delta lfuda$
10 Kbytes	0.0067	0.0155	0.0088	0.0139	0.0398	0.0259
20 Kbytes	0.0059	0.0179	0.0120	0.0147	0.0376	0.0229
30 Kbytes	0.0039	0.0163	0.0124	0.0168	0.0392	0.0224
40 Kbytes	0.0035	0.0168	0.0133	0.0172	0.0388	0.0216
50 Kbytes	0.0033	0.0168	0.0135	0.0174	0.0388	0.0214
60 Kbytes	0.0032	0.0166	0.0134	0.0175	0.0389	0.0214
70 Kbytes	0.0030	0.0168	0.0138	0.0177	0.0387	0.0210
80 Kbytes	0.0029	0.0169	0.0140	0.0179	0.0387	0.0208
90 Kbytes	0.0037	0.0183	0.0146	0.0170	0.0374	0.0204
10 0Kbytes	0.0036	0.0185	0.0149	0.0171	0.0372	0.0201
200 Kbytes	0.0025	0.0163	0.0138	0.0183	0.0392	0.0209
300 Kbytes	0.0024	0.0160	0.0136	0.0184	0.0395	0.0211
400 Kbytes	0.0021	0.0164	0.0143	0.0186	0.0391	0.0205
500 Kbytes	0.0018	0.0165	0.0147	0.0190	0.0390	0.0200
600 Kbytes	0.0017	0.0164	0.0147	0.0191	0.0390	0.0199
700 Kbytes	0.0016	0.0166	0.0150	0.0192	0.0389	0.0197
800 Kbytes	0.0016	0.0166	0.0150	0.0192	0.0389	0.0197
900 Kbytes	0.0016	0.0166	0.0150	0.0192	0.0389	0.0197
1 Mbytes	0.0016	0.0166	0.0150	0.0192	0.0389	0.0197

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

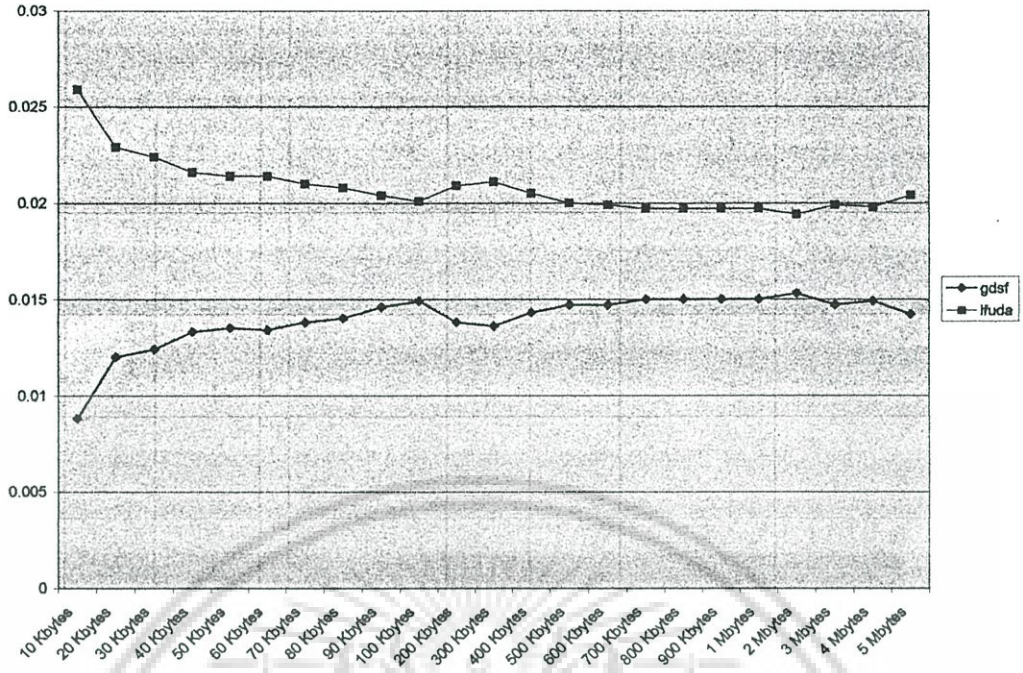
ตารางที่ 4.1 (ต่อ)

Threshold	$\overline{\Delta H_i}(gdsf)$	$\overline{\Delta H_b}(gdsf)$	$\Delta gdsf$	$\overline{\Delta H_i}(lfuda)$	$\overline{\Delta H_b}(lfuda)$	$\Delta lfuda$
2 Mbytes	0.0014	0.0167	0.0153	0.0194	0.0388	0.0194
3 Mbytes	0.0013	0.0160	0.0147	0.0195	0.0394	0.0199
4 Mbytes	0.0012	0.0161	0.0149	0.0196	0.0394	0.0198
5 Mbytes	0.0004	0.0146	0.0142	0.0205	0.0409	0.0204

จากตารางที่ 4.1 เมื่อเราพิจารณาเฉพาะค่า $\Delta gdsf$ และ $\Delta lfuda$ จะพบว่าเมื่อค่า $\Delta gdsf$ มีค่าเพิ่มขึ้นค่าของ $\Delta lfuda$ จะมีค่าลดลงด้วย เนื่องจากกราฟของอัลกอริธึมที่นำเสนอจะอยู่ระหว่างกราฟของ GDSF และ LFUDA ทั้งกรณีของ %Hit และ %Byte-Hit และเนื่องจากเงื่อนไขของค่า Threshold ที่เหมาะสมคือ $\Delta gdsf$ (max) และ $\Delta lfuda$ (min) ดังนั้นค่า Threshold ที่เหมาะสมจะอยู่ในช่วงที่ผลต่างระหว่าง $\Delta lfuda$ และ $\Delta gdsf$ มีค่าน้อยที่สุด ดังนั้นจากตารางที่ 4.1 เมื่อคำนวณหาค่า Threshold ที่ให้ค่าผลต่างระหว่าง $\Delta lfuda$ และ $\Delta gdsf$ น้อยที่สุดคือค่า Threshold ที่ 2 Mbytes

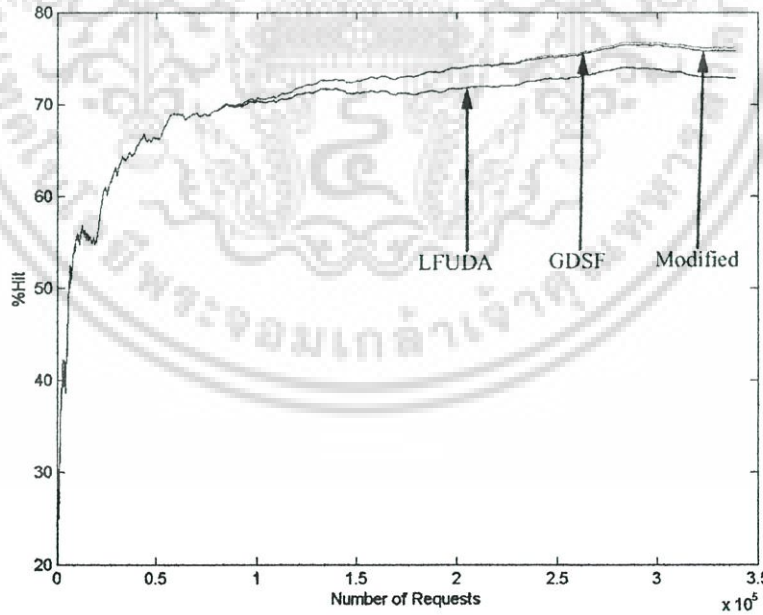
จากค่าของ $\Delta gdsf$ และ $\Delta lfuda$ ที่ได้เมื่อนำมาเขียนกราฟจะแสดงได้ดังรูปที่ 4.56 โดยในกรณีของข้อมูลที่ได้จากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าลาดกระบัง จะพบว่าชุดข้อมูลของ $\Delta gdsf$ และ $\Delta lfuda$ มีค่าเป็นบวกทั้งคู่และชุดข้อมูลของ $\Delta lfuda$ มีค่าสูงกว่าชุดข้อมูลของ $\Delta gdsf$ ดังนั้นจากเงื่อนไขของค่า Threshold ที่เหมาะสมจะพบว่าค่า Threshold ที่ให้ค่าผลต่างระหว่างชุดข้อมูลของ $\Delta gdsf$ และชุดข้อมูลของ $\Delta lfuda$ น้อยที่สุด จะเป็นค่า Threshold ที่เหมาะสม และเมื่อพิจารณาจากกราฟจะพบว่าค่า Threshold เท่ากับ 800 Kbytes, 900 Kbytes และ 1 Mbytes จะให้ค่าผลต่างระหว่างชุดข้อมูลของ $\Delta gdsf$ และชุดข้อมูลของ $\Delta lfuda$ ที่เท่ากันและใกล้เคียงกับค่าที่ Threshold เท่ากับ 2 Mbytes ทำให้ถือว่าค่า Threshold ที่จุดนั้นถือเป็นค่า Threshold ที่เหมาะสมเช่นกันและที่ค่า Threshold เท่ากับ 100 Kbytes จะเห็นว่าค่าผลต่างของกราฟจะมีค่าเข้าใกล้กับค่าผลต่างของกราฟที่ค่า Threshold เท่ากับ 800 Kbytes, 900 Kbytes และ 1 Mbytes ดังนั้นจึงเป็นค่า Threshold อีกค่าที่เหมาะสมในการใช้งาน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



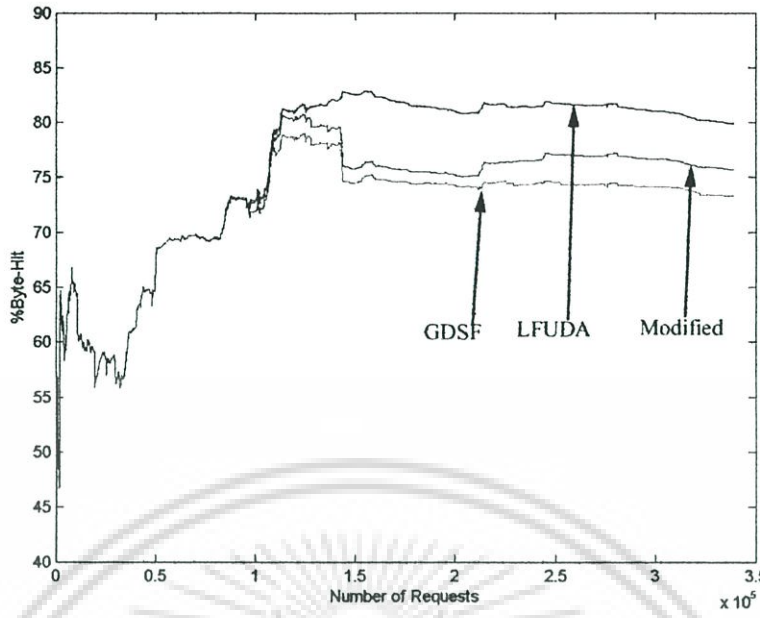
รูปที่ 4.56 กราฟแสดงผลต่างระหว่าง $\Delta gdsf$ และ $\Delta lfuda$

จากค่า Threshold ที่เหมาะสมที่ได้จากการวิเคราะห์คือค่า Threshold ที่ 2 Mbytes นำมาจำลองการทำงานหาค่า %Hit และ %Byte-Hit แสดงได้ดังรูปที่ 4.57 และ 4.58



รูปที่ 4.57 กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 2 Mbytes

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.58 กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 2 Mbytes

ตารางที่ 4.2 แสดงค่า $\overline{\Delta H_i(gdsf)}$, $\overline{\Delta H_i(lfuda)}$, $\overline{\Delta H_b(gdsf)}$, $\overline{\Delta H_b(lfuda)}$, $\Delta gdsf$ และ $\Delta lfuda$ ของข้อมูลจาก Clarknet

Threshold	$\overline{\Delta H_i(gdsf)}$	$\overline{\Delta H_b(gdsf)}$	$\Delta gdsf$	$\overline{\Delta H_i(lfuda)}$	$\overline{\Delta H_b(lfuda)}$	$\Delta lfuda$
10 Kbytes	-	-	-	-	-	-
20 Kbytes	0.0053	0.0131	0.0078	0.0046	0.0024	-0.0022
30 Kbytes	0.0036	0.0131	0.0095	0.0063	0.0023	-0.0040
40 Kbytes	0.0031	0.0128	0.0097	0.0069	0.0026	-0.0043
50 Kbytes	0.0026	0.0121	0.0095	0.0074	0.0034	-0.0040
60 Kbytes	0.0024	0.0127	0.0103	0.0076	0.0028	-0.0048
70 Kbytes	0.0022	0.0127	0.0105	0.0077	0.0028	-0.0049
80 Kbytes	0.0020	0.0129	0.0109	0.0079	0.0026	-0.0053
90 Kbytes	0.0018	0.0118	0.0100	0.0082	0.0036	-0.0046
10 0Kbytes	0.0018	0.0126	0.0108	0.0081	0.0028	-0.0053
200 Kbytes	0.0013	0.0125	0.0112	0.0087	0.0031	-0.0056
300 Kbytes	0.0009	0.0124	0.0115	0.0090	0.0031	-0.0059
400 Kbytes	0.0008	0.0121	0.0113	0.0092	0.0035	-0.0057
500 Kbytes	0.0007	0.0118	0.0111	0.0093	0.0038	-0.0055
600 Kbytes	0.0007	0.0110	0.0103	0.0093	0.0045	-0.0048

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

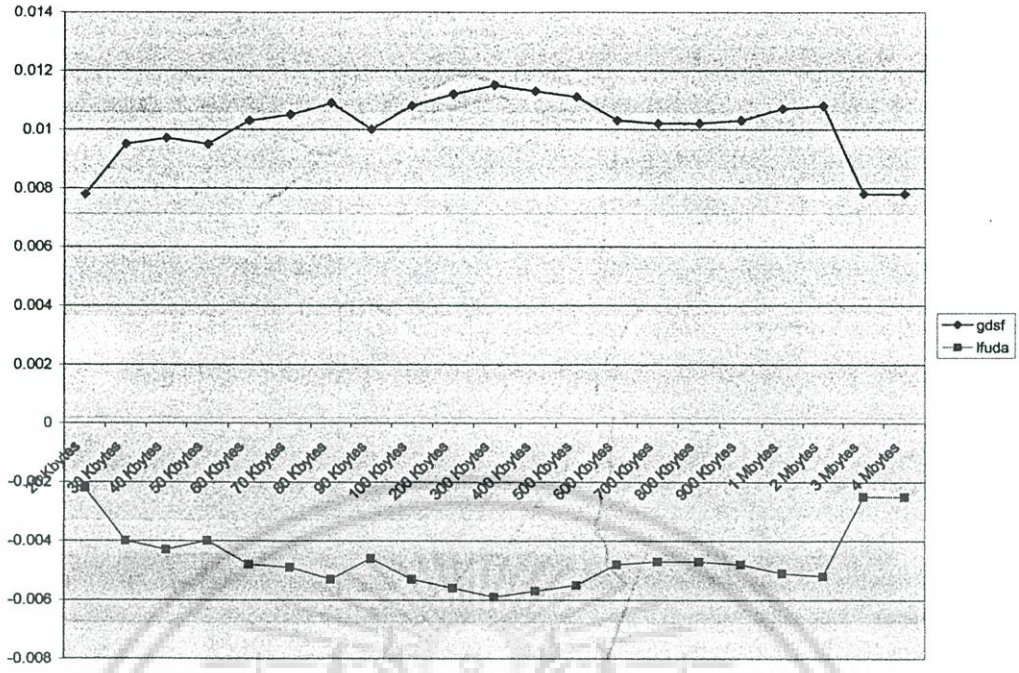
ตารางที่ 4.2 (ต่อ)

Threshold	$\overline{\Delta H_i(gdsf)}$	$\overline{\Delta H_b(gdsf)}$	$\Delta gdsf$	$\overline{\Delta H_i(lfuda)}$	$\overline{\Delta H_b(lfuda)}$	$\Delta lfuda$
700 Kbytes	0.0007	0.0109	0.0102	0.0093	0.0046	-0.0047
800 Kbytes	0.0007	0.0109	0.0102	0.0093	0.0046	-0.0047
900 Kbytes	0.0006	0.0109	0.0103	0.0094	0.0046	-0.0048
1 Mbytes	0.0005	0.0112	0.0107	0.0095	0.0044	-0.0051
2 Mbytes	0.0005	0.0113	0.0108	0.0095	0.0043	-0.0052
3 Mbytes	0.0002	0.0080	0.0078	0.0098	0.0073	-0.0025
4 Mbytes	0.0002	0.0080	0.0078	0.0098	0.00730.0073	-0.0025
5 Mbytes	-	-	-	-	-	-

จากตารางที่ 4.2 ค่าของ $\Delta gdsf$ และ $\Delta lfuda$ ใน Threshold ที่ค่า 10 Kbytes และ 5 Mbytes จะไม่นำวิเคราะห์เพราะที่ค่า Threshold เท่ากับ 10 Kbytes และ 5 Mbytes จะให้ค่า $\overline{\Delta H_i(lfuda)}$ และค่า $\overline{\Delta H_b(gdsf)}$ มีค่าเป็นลบ ซึ่งหมายความว่าค่าที่ได้จากการแทนค่า Threshold ที่ค่า 10 Kbytes และ 5 Mbytes ให้ผลต่ำกว่ามาตรฐานที่ต้องการ และเนื่องจากเงื่อนไขของค่า Threshold ที่เหมาะสมคือ $\Delta gdsf$ (max) และ $\Delta lfuda$ (min) ดังนั้นค่า Threshold ที่เหมาะสมจะอยู่ในช่วงที่ผลต่างระหว่าง $\Delta gdsf$ และ $\Delta lfuda$ มีค่ามากที่สุด ดังนั้นจากตารางที่ 4.2 เมื่อคำนวณหาค่า Threshold ที่ให้ค่าผลต่างระหว่าง $\Delta gdsf$ และ $\Delta lfuda$ มากที่สุดคือค่า Threshold ที่ 300 Kbytes

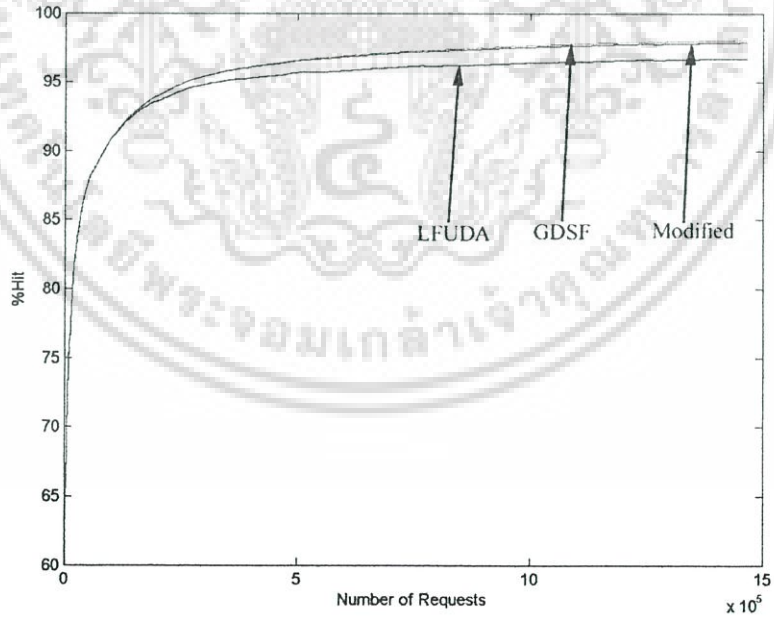
จากค่าของ $\Delta gdsf$ และ $\Delta lfuda$ ที่ได้เมื่อนำมาเขียนกราฟจะแสดงได้ดังรูปที่ 4.57 โดยในกรณีของข้อมูลที่ได้จาก Clarknet จะพบว่าชุดข้อมูลของ $\Delta gdsf$ จะมีค่าเป็นบวกและ $\Delta lfuda$ จะมีค่าเป็นลบ ดังนั้นชุดข้อมูลของ $\Delta lfuda$ จะมีค่าต่ำกว่าชุดข้อมูลของ $\Delta gdsf$ จากเงื่อนไขของค่า Threshold ที่เหมาะสม จะพบว่าค่า Threshold ที่ให้ค่าผลต่างระหว่างชุดข้อมูลของ $\Delta gdsf$ และชุดข้อมูลของ $\Delta lfuda$ มีค่ามากที่สุด จะเป็นค่า Threshold ที่เหมาะสมสำหรับข้อมูลของ Clarknet

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



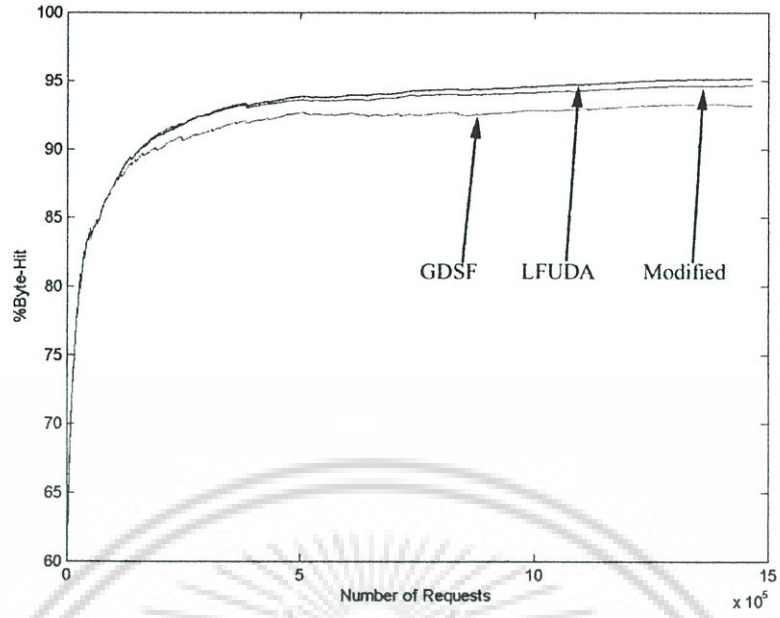
รูปที่ 4.59 กราฟแสดงผลต่างระหว่าง $\Delta gdsf$ และ $\Delta lfuda$

จากค่า Threshold ที่เหมาะสมที่ได้จากการวิเคราะห์คือค่า Threshold ที่ 300 Kbytes นำมาจำลองการทำงานหาค่า %Hit และ %Byte-Hit แสดงได้ดังรูปที่ 4.60 และ 4.61



รูปที่ 4.60 กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 300 Kbytes

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.61 กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 300 Kbytes

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 5
การประยุกต์ใช้งานอัลกอริธึมการแทนที่ข้อมูลที่มีการปรับแต่งกับระบบเว็บแคชเซิร์ฟเวอร์

จากการจำลองการทำงานของเว็บแคชเซิร์ฟเวอร์พบว่า ข้อมูลทดสอบของภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง และข้อมูลทดสอบของ Clarknet ค่า %Hit ของอัลกอริธึมการแทนที่ข้อมูลที่มีการปรับแต่งมีค่าใกล้เคียงกับค่า %Hit ของอัลกอริธึมการแทนที่ข้อมูลเดิมก่อนที่จะมีการปรับแต่งและค่า %Byte-Hit ของอัลกอริธึมการแทนที่ข้อมูลที่มีการปรับแต่งมีค่าเพิ่มขึ้นเมื่อเทียบกับค่า %Byte-Hit ของอัลกอริธึมการแทนที่ข้อมูลเดิม

ดังนั้นเพื่อเปรียบเทียบผลที่ได้จากการจำลองการทำงานของเว็บแคชเซิร์ฟเวอร์กับระบบเปิด จึงได้ทำการทดสอบภายในระบบเครือข่ายที่มีการติดต่อกับเครือข่ายอินเทอร์เน็ตโดยใช้ชุดการร้องขอข้อมูลที่มีการรวบรวมจากเว็บแคชเซิร์ฟเวอร์ของภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ภายในระยะเวลา 7 วัน โดยมีการร้องขอข้อมูลทั้งหมด 338822 ครั้ง ขั้นตอนการทดสอบภายในระบบเครือข่ายอินเทอร์เน็ตประกอบด้วย การเตรียมระบบเพื่อใช้ในการทดสอบ, การทดสอบ, ผลการทดสอบ และสรุปผลการทดสอบ

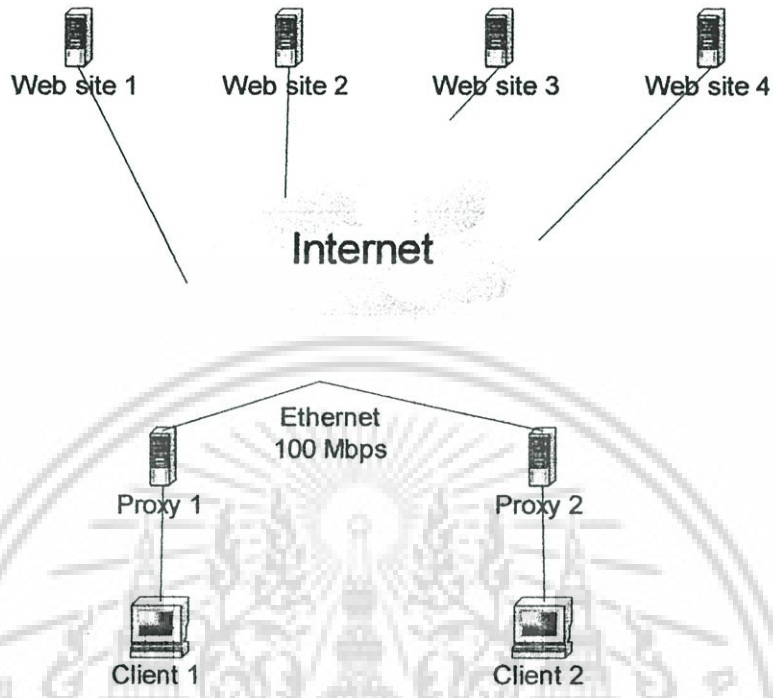
5.1 เครื่องมือที่ใช้ในการทดสอบ

เครื่องประมวลผลที่ทำหน้าที่เป็นเว็บแคชเซิร์ฟเวอร์หรือพรอกซี เป็นเครื่องไมโครคอมพิวเตอร์ของบริษัท HP รุ่น Kayak ใช้ซีพียู Pentium III 866 MHz, หน่วยความจำ 256 MB และ ฮาร์ดดิสก์ชนิด IDE ความจุ 20 กิกะไบต์ ระบบปฏิบัติการที่ใช้คือ Linux ของบริษัท Debian เวอร์ชั่น 3.0 [28] และใช้โปรแกรม Squid ทำหน้าที่เป็นเว็บแคชเซิร์ฟเวอร์ และได้กำหนดเนื้อที่แคชของเว็บแคชเซิร์ฟเวอร์ให้มีค่าเป็น 128 MB โดยเครื่องไมโครคอมพิวเตอร์ได้ต่อเข้ากับระบบเครือข่ายอินเตอร์เน็ต ผ่านทางระบบเครือข่ายอีเทอร์เนตความเร็ว 100 Mbps โดยระบบการเชื่อมต่อทั้งหมดแสดงได้ดังรูปที่ 5.1

โปรแกรม Squid เว็บแคชเซิร์ฟเวอร์ถูกตั้งค่าพารามิเตอร์ต่างตามค่าของไฟล์ squid.conf โดยค่าการตั้งค่าของ squid.conf แสดงไว้ในภาคผนวก และโปรแกรม Squid ถูกตั้งค่าให้รับการติดต่อมาจาก IP Address ที่กำหนดเท่านั้น ซึ่งหากมีการติดต่อมาจาก IP Address นอกเหนือจากที่กำหนดโปรแกรม Squid จะปฏิเสธการติดต่อจาก IP Address นั้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในการทดสอบจะใช้โปรแกรม Cfmc เป็นโปรแกรมสำหรับสร้างการร้องขอจากไคลเอนต์จำนวน 40 ไคลเอนต์ไปยัง Squid เว็บแคชเซิร์ฟเวอร์



รูปที่ 5.1 รูปแบบการเชื่อมต่อของระบบที่ใช้ในการทดสอบ

5.2 ข้อมูลที่ใช้ในการทดสอบ

ข้อมูลที่นำมาทดสอบการปรับแต่งอัลกอริธึมการแทนที่ข้อมูลของ Squid เว็บแคชเซิร์ฟเวอร์ เป็นข้อมูลที่ได้จากการเก็บไฟล์ Log จากเว็บแคชเซิร์ฟเวอร์ของภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังภายในระยะเวลา 1 สัปดาห์ โดยไฟล์ Log จะประกอบด้วยข้อมูลที่แบ่งตามชนิดของไฟล์ ซึ่งข้อมูลที่นำมาทดสอบจะเป็นข้อมูลของเว็บเพ็จ ต่าง ๆ ที่ประกอบไปด้วยไฟล์รูปภาพ ไฟล์ HTML และไฟล์ประเภทอื่น เมื่อพิจารณาจะพบว่าข้อมูลรูปภาพจะมีจำนวนมากที่สุด ดังนั้นจึงแบ่งข้อมูลแสดงตามชนิดของไฟล์ได้ดังตารางที่ 5.1 และ เมื่อพิจารณาถึงช่วงขนาดของไฟล์พบว่า ไฟล์รูปภาพที่เป็นชนิดไฟล์ที่มีมากที่สุด โดยส่วนใหญ่จะเป็นไฟล์ที่มีขนาดเล็กมีค่าไม่เกิน 10 Kbytes ดังแสดงในตารางที่ 5.2

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 5.1 แสดงเปอร์เซ็นต์ของข้อมูลที่ร้องขอแบ่งตามชนิดของไฟล์

ชนิดของไฟล์	ค่าเปอร์เซ็นต์
ไฟล์รูปภาพนามสกุล GIF	55.8 %
ไฟล์รูปภาพนามสกุล JPEG	18.1 %
ไฟล์นามสกุล HTML	7.5 %
อื่น ๆ	18.6 %

ตารางที่ 5.2 แสดงเปอร์เซ็นต์ของข้อมูลที่ร้องขอแบ่งตามช่วงขนาดของไฟล์

ช่วงขนาดของไฟล์	ค่าเปอร์เซ็นต์
100 – 999 Bytes	37.4 %
1000 – 9999 Bytes	51.4 %
10000 – 99999 Bytes	10.9 %
อื่น ๆ	0.3 %

5.3 เงื่อนไขการพิจารณาข้อมูลที่ได้จากไฟล์ Log ของ Squid เว็บแคชเซิร์ฟเวอร์

จากข้อมูลที่นำมาวิเคราะห์ในงานวิจัยนี้ เรากำหนดรูปแบบของข้อมูลดังต่อไปนี้

1. ข้อมูลเป็นข้อมูลที่ร้องขอผ่านทางโปรโตคอล HTTP เท่านั้น (ไม่รวมถึงการให้บริการผ่านโปรโตคอลตัวอื่น เช่น FTP, SSH และ SMTP)

2. ข้อมูลจะต้องผ่านการร้องขอใช้งานด้วยความถูกต้อง นอกจากนั้นจะไม่นับ URL ที่มีลักษณะแบบไดนามิกส์เช่น cgi, php, asp และ URL ที่มีสัญลักษณ์พิเศษประกอบอยู่เช่น ?, =, %, +, *, &, @, $, [ ], ( ) เนื่องจาก URL ที่มีลักษณะเช่นนี้เป็นURL ที่ไม่ได้ระบุถึงข้อมูลเว็บเพ็จที่สามารถนำมาใช้วิเคราะห์ได้

3. สำหรับการวิเคราะห์เราจะพิจารณาเปอร์เซ็นต์ของอัตราการพบข้อมูลและ อัตราการพบข้อมูลแบบไบต์ จะพิจารณาเฉพาะบรรทัดในไฟล์ Log ของเว็บแคชเซิร์ฟเวอร์ ที่มีคำว่า TCP_HIT, TCP_MEM_HIT, TCP_REFRESH_HIT, TCP_REF_FAIL_HIT และ TCP_IMS_HIT เนื่องจากบรรทัดในไฟล์ Log ที่มีคำที่กล่าวมา จะเป็นบรรทัดที่แสดงให้ทราบถึงการพบข้อมูลที่ไคลเอนต์มีการร้องขอภายในแคชของเว็บแคชเซิร์ฟเวอร์ และบรรทัดของไฟล์ Log ที่มีคำอยู่นอกเหนือจากที่กล่าวมาถือว่าไม่พบข้อมูลในแคชของเว็บแคชเซิร์ฟเวอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5.4 ผลการทดสอบอัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่ง

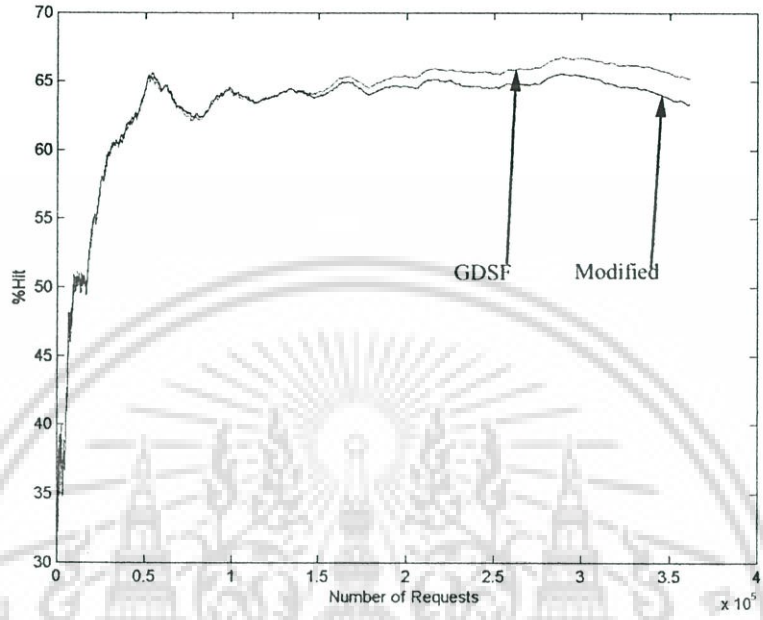
ในการทดสอบอัลกอริธึมการแทนที่ข้อมูลที่ทำการปรับแต่ง จะทำการทดสอบโดยใช้เครื่องคอมพิวเตอร์จำนวน 2 เครื่องโดยเครื่องคอมพิวเตอร์พรอกซีที่ 1 ตั้งอัลกอริธึมการแทนที่ข้อมูลเป็นอัลกอริธึมการแทนที่ข้อมูลที่ถูกปรับแต่งและเครื่องคอมพิวเตอร์พรอกซี 2 ตั้งอัลกอริธึมการแทนที่ข้อมูลเป็นแบบ GDSF เนื่องจากอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF เป็นอัลกอริธึมการแทนที่ข้อมูลที่ให้ค่า% Hit มีค่าสูงที่สุด แต่จะมีค่าของ %Byte-Hit มีค่าต่ำที่สุด ซึ่งสมมติฐานที่ตั้งไว้ในงานวิจัยนี้คือต้องการให้ค่าของ %Hit มีค่าใกล้เคียงกับค่าเดิมและมีค่าของ %Byte-Hit มีค่าเพิ่มขึ้นจากเดิม เมื่อเทียบกับค่า %Hit และค่า %Byte-Hit ของอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF

ในการทดสอบจะทำการทดสอบโดยทำการปรับเปลี่ยนค่าของ Threshold จำนวน 3 ค่า คือ 100 Kbytes, 1 Mbytes และ 10 Mbytes โดยใช้ข้อมูลชุดเดียวกันในการทดสอบ และเก็บค่ามาจากไฟล์ Log ซึ่งเป็นไฟล์ที่เก็บรายละเอียดการทำงานของโปรแกรม Squid แล้วนำมาทำการวิเคราะห์ โดยทำการวิเคราะห์ในส่วนของ %Hit และ %Byte-Hit โดย ค่า %Hit ที่ได้จากการวิเคราะห์ค่า Threshold ที่มีค่า 100 Kbytes, 1 Mbytes และ 10 Mbytes แสดงดังรูป 5.2, 5.5 และ 5.8 ส่วนค่าของ %Byte-Hit ที่ได้จากการวิเคราะห์ค่า Threshold ที่มีค่า 100 Kbytes, 1 Mbytes และ 10 Mbytes แสดงดังรูปที่ 5.3, 5.6 และ 5.9 และค่า %Hit เมื่อพิจารณาตามช่วงขนาดของข้อมูล ของค่า Threshold ที่มีค่า 100 Kbytes, 1 Mbytes และ 10 Mbytes แสดงดังรูปที่ 5.4, 5.7 และ 5.10

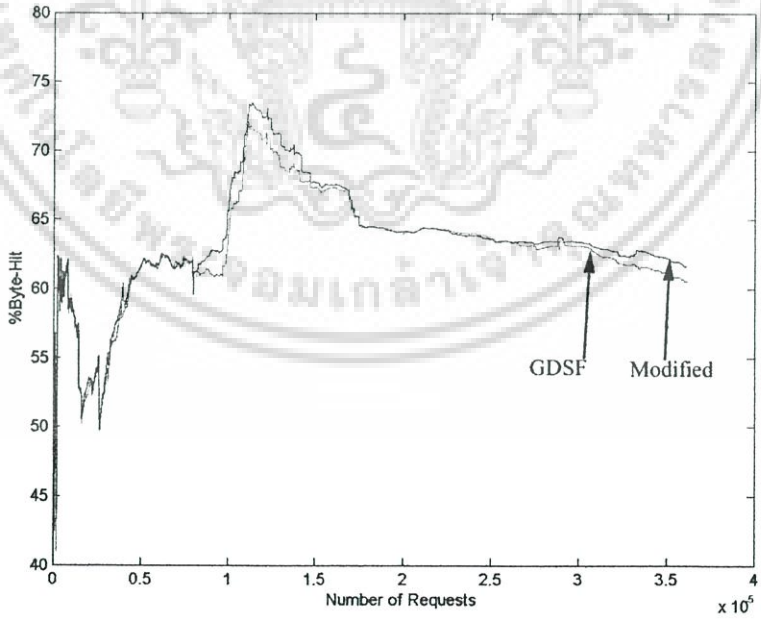
- ค่าในแกน X แสดงถึง จำนวนครั้งของการร้องขอที่เกิดขึ้น เริ่มจาก 0 ไปถึง 338,822 ครั้ง
- ค่าในแกน Y แสดงถึง %Hit และ %Byte-Hit ของค่า Threshold ตั้งแต่ 100 Kbytes, 1 Mbytes และ 10 Mbytes ตามลำดับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5.4.1 การเปรียบเทียบอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF กับ อัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 100 Kbytes

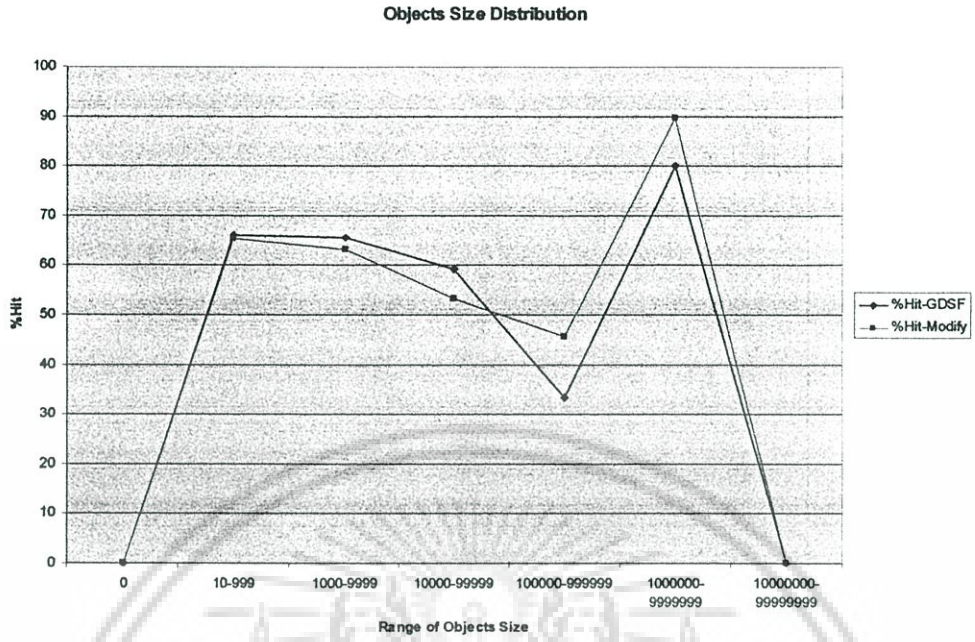


รูปที่ 5.2 กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 100 Kbytes



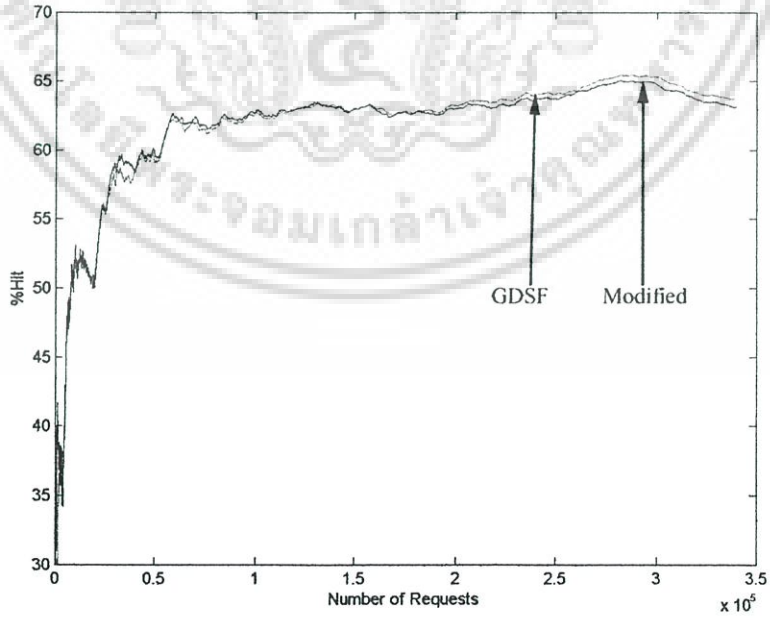
รูปที่ 5.3 กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 100 Kbytes

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



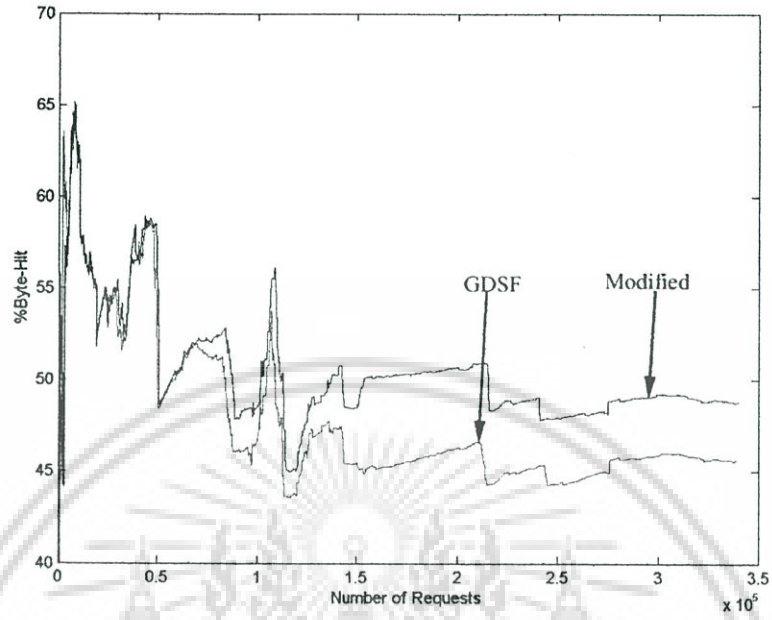
รูปที่ 5.4 กราฟเปรียบเทียบ %Hit ของการปรับค่า Threshold มีค่า 100 Kbytes แบ่งตามช่วงของขนาดข้อมูล

5.4.2 การเปรียบเทียบอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF กับ อัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 1 Mbytes

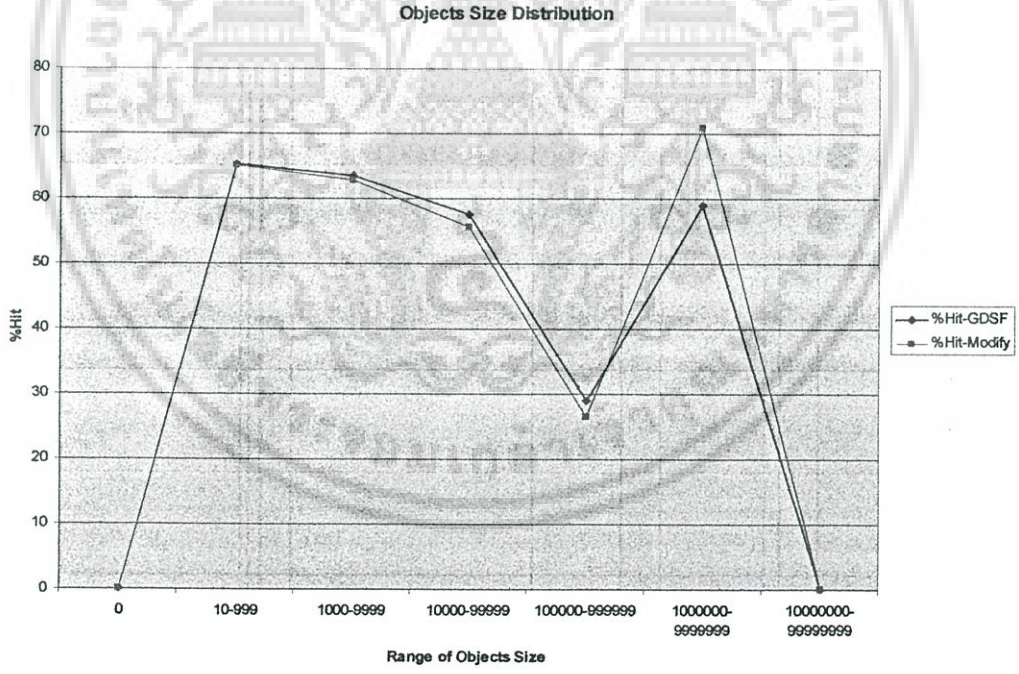


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 5.5 กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 1 Mbytes



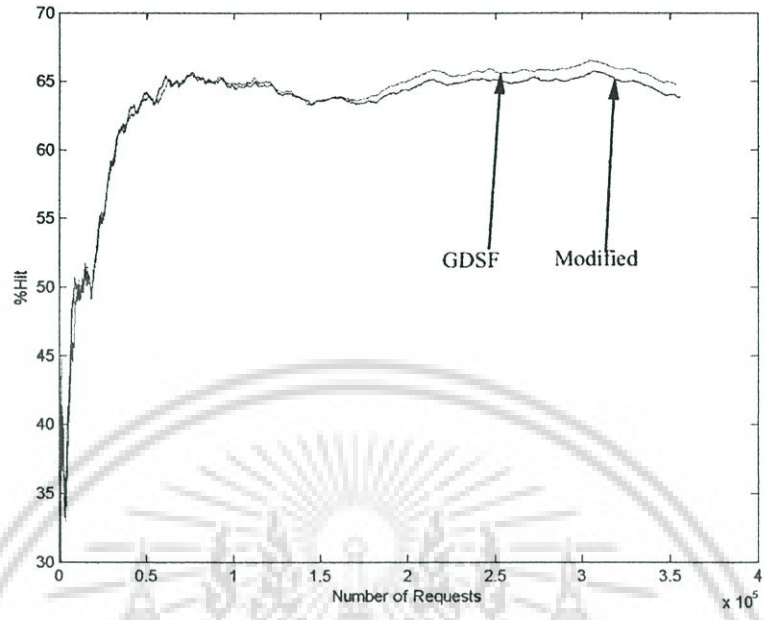
รูปที่ 5.6 กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 1 Mbytes



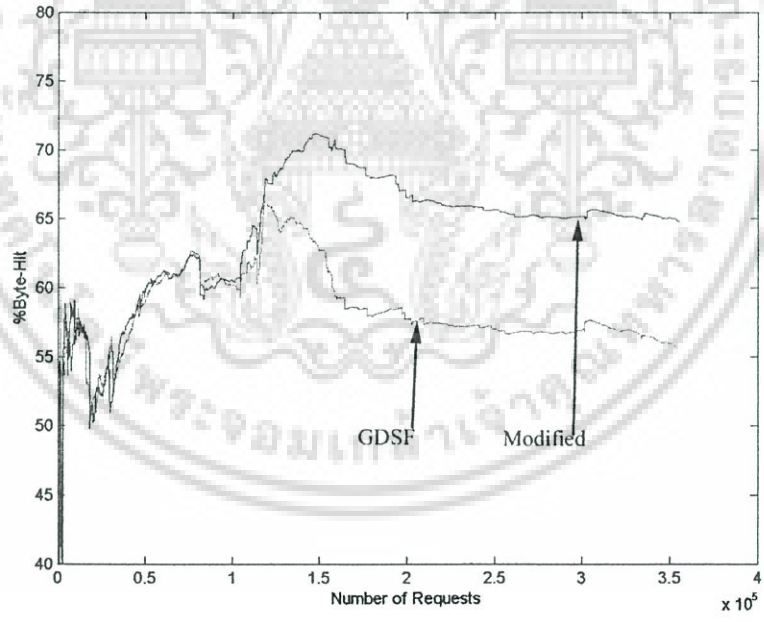
รูปที่ 5.7 กราฟเปรียบเทียบ %Hit ของการปรับค่า Threshold มีค่า 1 Mbytes แบ่งตามช่วงของขนาดข้อมูล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5.4.3 การเปรียบเทียบอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF กับ อัลกอริธึมการแทนที่ข้อมูลที่ปรับแต่งให้ค่า Threshold มีค่า 10 Mbytes

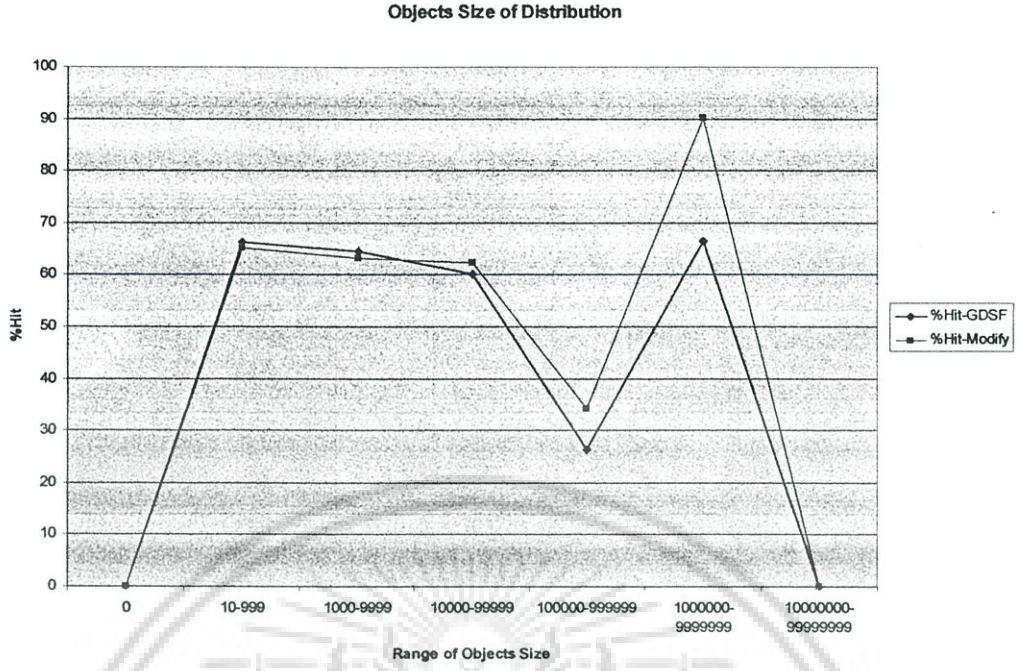


รูปที่ 5.8 กราฟแสดงค่า %Hit ของการปรับให้ค่า Threshold มีค่า 10 Mbytes



รูปที่ 5.9 กราฟแสดงค่า %Byte-Hit ของการปรับให้ค่า Threshold มีค่า 10 Mbytes

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 5.10 กราฟเปรียบเทียบ %Hit ของการปรับค่า Threshold มีค่า 10 Mbytes แบ่งตามช่วงของขนาดข้อมูล

5.5 การวิเคราะห์ผลการทดสอบ

จากกราฟในรูปที่ 5.2, 5.5 และ 5.8 ที่แสดงถึง %Hit ของค่า Threshold ที่ค่า 100 Kbytes, 1 Mbytes และ 10 Mbytes ตามลำดับ พบว่าค่า %Hit ก่อนทำการปรับแต่งอัลกอริธึมการแทนที่ข้อมูล (GDSF) และหลังทำการปรับแต่งอัลกอริธึมการแทนที่ข้อมูล (Modify) ของค่า Threshold แต่ละค่าจะมีรูปกราฟใกล้เคียงกันดังรูปที่ 5.2 และ 5.8 หรือเกือบจะเหมือนกันรูปที่ 5.5 เนื่องจากค่า %Hit จะเป็นค่าที่พิจารณาถึงจำนวนครั้งของการพบข้อมูลที่อยู่ในแคชของเว็บแคชเซิร์ฟเวอร์ ซึ่งตามสมมติฐานที่ตั้งไว้ค่า %Hit ของอัลกอริธึมที่ถูกปรับแต่งจะต้องมีค่าลดต่ำลงกว่าค่า %Hit ของอัลกอริธึมการแทนที่ข้อมูลก่อนทำการปรับแต่ง

เนื่องจากเมื่อทำการปรับแต่งให้ข้อมูลที่มีขนาดใหญ่บางส่วนยังคงถูกเก็บไว้ในแคชของเว็บแคชเซิร์ฟเวอร์ จากการทำงานของอัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA จะเป็นผลทำให้พื้นที่เก็บข้อมูลของแคชมีค่าลดน้อยลง ดังนั้นเว็บแคชเซิร์ฟเวอร์จึงสามารถเก็บข้อมูลที่มีขนาดเล็กได้น้อยลงซึ่งจะเป็นผลทำให้ค่าของ %Hit ที่ได้มีค่าลดลง แต่เนื่องจากค่า %Hit จะเป็นค่าที่พิจารณาถึงจำนวนครั้งในการพบข้อมูล ดังนั้น เมื่อจำนวนการพบข้อมูลมีค่าที่ลดลงไม่มากเมื่อเทียบกับการร้องขอข้อมูลที่มีค่ามากขึ้นเรื่อย ๆ จะทำให้ค่า %Hit ที่ออกมามีค่าไม่ต่างไปจากค่า %Hit ที่ได้ก่อนที่จะทำการปรับแต่งอัลกอริธึมการแทนที่ข้อมูล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อพิจารณาจากรูปที่ 5.3, 5.6 และ 5.9 ที่แสดงถึงค่า %Byte-Hit ที่ค่า Threshold มีค่า 100 Kbytes, 1 Mbytes และ 10 Mbytes พบว่าเมื่อทำการปรับค่า Threshold ให้มีค่ามากขึ้นเรื่อยๆ ค่า %Byte-Hit ซึ่งเป็นค่าที่พิจารณาถึงการพบข้อมูลโดยพิจารณาจากขนาดของข้อมูลที่พบในแคช จะมีค่าสูงขึ้นเมื่อเทียบกับค่า %Byte-Hit ก่อนทำการปรับแต่งอัลกอริธึมการแทนที่ข้อมูล เนื่องจากการตั้งค่า Threshold ให้มีขนาด 100 Kbytes ดังรูปที่ 5.3 จะทำให้ข้อมูลที่มีขนาดใหญ่ที่ควรถูกเก็บไว้ในแคชของเว็บแคชเซิร์ฟเวอร์ถูกนำออกไปจากแคช เพราะข้อมูลที่มีขนาดเล็กซึ่งเป็นข้อมูลประเภทรูปภาพจะถูกเรียกใช้งานออกไปจากแคชมากกว่าข้อมูลที่มีขนาดใหญ่ จากการทำงานของอัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA และในรูปที่ 5.6 และ 5.9 ที่มีการตั้งค่า Threshold เป็น 1 Mbytes และ 10 Mbytes จะพบว่าข้อมูลที่มีขนาดใหญ่จะถูกเก็บไว้ในแคชและถูกเรียกใช้งานทำให้ค่า %Byte-Hit ที่ได้ซึ่งพิจารณาถึงขนาดของไฟล์ที่มีการพบในแคชมีค่าสูง แต่เมื่อพิจารณาในรูปที่ 5.9 ที่มีค่า Threshold เป็น 10 Mbytes ซึ่งพบว่าความแตกต่างระหว่างค่า %Byte-Hit หลังทำการปรับแต่งอัลกอริธึมการแทนที่ข้อมูลกับค่า %Byte-Hit ก่อนทำการปรับแต่งอัลกอริธึมการแทนที่ข้อมูลมีค่ามากสุด จะมีค่าความแตกต่างระหว่างค่า %Hit หลังทำการปรับแต่งอัลกอริธึมการแทนที่ข้อมูลกับค่า %Hit ก่อนทำการปรับแต่งอัลกอริธึมการแทนที่ข้อมูลมากเช่นกัน เนื่องจากเว็บแคชเซิร์ฟเวอร์ใช้เนื้อที่ในการเก็บข้อมูลที่มีขนาดใหญ่หลายข้อมูลทำให้เหลือพื้นที่สำหรับนำมาเก็บข้อมูลที่มีขนาดเล็กน้อยลง ทำให้การพบข้อมูลที่ร้องขอจากในแคชของเว็บแคชเซิร์ฟเวอร์มีค่าลดลงตามไปด้วย

เมื่อพิจารณาถึง %Hit ตามช่วงขนาดของข้อมูลที่ค่า Threshold มีค่า 100 Kbytes, 1Mbytes และ 10 Mbytes ดังแสดงในรูปที่ 5.4, 5.7 และ 5.10 พบว่าการปรับแต่งอัลกอริธึมการแทนที่ข้อมูลโดยให้ค่า Threshold มีค่า 100 Kbytes, 1Mbytes และ 10 Mbytes จะมีผลทำให้ค่า %Hit ในช่วงข้อมูลที่มีใหญ่มีค่าเพิ่มมากขึ้น (ซึ่งแสดงถึงการพบข้อมูลที่มีขนาดใหญ่ภายในแคชของเว็บแคชเซิร์ฟเวอร์มากขึ้นกว่าเมื่อเทียบกับอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF) และจะมีค่า %Hit ในช่วงข้อมูลที่มีขนาดเล็กลดลง โดยในกราฟที่ 5.10 ที่มีค่า Threshold มีค่า 10 Mbytes จะพบว่าค่า %Hit ของข้อมูลที่มีช่วงขนาดเล็ก (10-999 Bytes) จะลดลงกว่าอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF อย่างชัดเจน แต่ %Hit ในช่วงข้อมูลขนาดใหญ่ (1000000-9999999 Bytes) จะมี %Hit เพิ่มมากกว่าอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF อย่างชัดเจนเช่นเดียวกัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 6
บทสรุป

6.1 สรุปงานวิจัยที่นำเสนอ

ในปัจจุบันการพัฒนาเพื่อเพิ่มประสิทธิภาพของเว็บแคชเซิร์ฟเวอร์ได้ถูกนำเสนอขึ้นมาในหลาย ๆ ด้าน ทั้งทางด้านการพัฒนาทางฮาร์ดแวร์และการพัฒนาทางซอฟแวร์ของเว็บแคชเซิร์ฟเวอร์ ซึ่งงานวิจัยนี้นำเสนอการพัฒนาทางซอฟแวร์ของเว็บแคชเซิร์ฟเวอร์โดยศึกษาฟังก์ชั่นที่ชื่อว่า อัลกอริธึมการแทนที่ข้อมูล ซึ่งเป็นฟังก์ชั่นที่ทำหน้าที่ในการดูแลและพิจารณาถึงการนำข้อมูลออกไปจากแคชของเว็บแคชเซิร์ฟเวอร์ โดยใช้ค่าที่นิยมนำมาใช้วัดประสิทธิภาพของเว็บแคชเซิร์ฟเวอร์คือเปอร์เซ็นต์การพบข้อมูลและเปอร์เซ็นต์การพบข้อมูลแบบไบต์ จากการศึกษาพบว่าอัลกอริธึมการแทนที่ข้อมูลจะเป็นตัวแปรตัวหนึ่งที่มีผลกระทบกับค่าทั้ง 2 นี้ และในแต่ละสภาพแวดล้อมที่ไม่เหมือนกัน พฤติกรรมของผู้ใช้งานเว็บแคชเซิร์ฟเวอร์ก็จะแตกต่างกันไป อัลกอริธึมการแทนที่ข้อมูลแต่ละแบบก็จะให้ผลลัพธ์ที่แต่งต่างกันไป ดังนั้นจึงได้ทำการวิเคราะห์เพื่อหาอัลกอริธึมการแทนที่ข้อมูลที่เหมาะสมกับระบบของเครือข่ายมหาวิทยาลัย

งานวิจัยนี้ได้พิจารณาถึงอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF และอัลกอริธึมการแทนที่แบบ LFUDA ซึ่งจากการทดลองพบว่าอัลกอริธึมการแทนที่ข้อมูลทั้ง 2 แบบ จะมีคุณลักษณะที่ต่างกันไป ดังนั้นงานวิจัยนี้จึงทำการนำคุณลักษณะของอัลกอริธึมแต่ละชนิดมาปรับใช้กับขนาดของไฟล์ โดยพิจารณาว่าหากไฟล์ที่มีขนาดใหญ่และถูกเรียกใช้งานบ่อยควรจะถูกเก็บไว้ในแคชของเว็บแคชเซิร์ฟเวอร์ จะทำให้ค่าของเปอร์เซ็นต์การพบข้อมูลแบบไบต์มีค่าสูงขึ้นเมื่อเทียบกับก่อนที่จะมีการปรับแต่ง แต่จะทำให้ค่าของเปอร์เซ็นต์การพบข้อมูลมีค่าลดลง เนื่องจากพื้นที่ส่วนหนึ่งของแคชจะต้องเสียไปสำหรับการเก็บไฟล์ที่มีขนาดใหญ่ ดังนั้นไฟล์ขนาดเล็กจะต้องถูกนำออกไปจากแคชมากกว่าก่อนที่จะมีการปรับแต่ง

6.2 สรุปผลการทดลอง

จากการทดลองแทนค่า Threshold ที่เหมาะสมของข้อมูลจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังและข้อมูลจาก Clarknet ได้ผลการทดลองออกมาดังนี้ จากการจำลองการทำงาน เมื่อเปรียบเทียบอัลกอริธึมการแทนที่ข้อมูลที่มีการปรับแต่งกับอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF พบว่าข้อมูลที่ได้จากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ที่ Threshold มีค่า 2 Mbytes มีค่า %Byte-Hit เพิ่มขึ้น 2.3949% มีค่า %Hit ลดลง 0.3775% ข้อมูลที่ได้จาก Clarknet ที่ Threshold มีค่า 300 Kbytes มีค่า % Byte-Hit

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เพิ่มขึ้น 1.4425% มีค่า %Hit ลดลง 0.1724% และจากจากการทดลองกับระบบเครือข่ายที่เชื่อมกับอินเทอร์เน็ตโดยใช้ข้อมูลของภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังมีค่า %Byte-Hit เพิ่มขึ้น 3.21198% มีค่า %Hit ลดลง 0.568511%

6.3 ปัญหาและอุปสรรค

ปัญหาที่พบในส่วนการจำลองการทดลองคือการประมวลผลที่ช้าของระบบ เนื่องจากในขั้นตอนการแปลงข้อมูลต้นฉบับเป็นไฟล์แบบตัวอักษร มาเป็นข้อมูลแบบตัวเลขสำหรับการจำลองการทดลอง ซึ่งใช้ภาษา Perl เป็นตัวจัดการแปลง และรูปแบบการจำลองการทดสอบจะเป็นการทำงานแบบลูปทำให้เกิดความล่าช้าในการทำงาน

ในส่วนของการจำลองการทำงานของเว็บแคชเซิร์ฟเวอร์ จำเป็นต้องใช้งานเครื่องคอมพิวเตอร์ที่มีประสิทธิภาพสูงเพื่อ ซึ่งในการจำลองแต่ละครั้งจะพบว่าต้องใช้เวลาประมาณ 5-15 ชม.สำหรับการจำลองการทำงานแต่ละครั้ง โดยจะขึ้นกับชนิดข้อมูลที่นำมาทดสอบและในวิเคราะห์หาค่า Threshold ที่เหมาะสมสำหรับข้อมูลแต่ละชุด จะต้องทำการปรับเปลี่ยนค่า Threshold ตั้งแต่ค่า 10 Kbytes ไป จนถึงค่า 5 Mbytes ซึ่งข้อมูลแต่ละชุดจะต้องทำการจำลองการทำงานอย่างน้อยที่สุดคือ 23 ครั้ง และหากต้องการหาค่า Threshold ที่เหมาะสมให้มีค่าละเอียดมากขึ้น ก็จะต้องเพิ่มค่า Threshold ที่นำมาทดสอบเพิ่ม ทำให้การจำลองการทำงานของข้อมูลแต่ละชุด ใช้เวลาสำหรับการจำลองการทำงานมาก

ในส่วนของการทดลองกับระบบเครือข่ายที่ต่อเชื่อมกับระบบเครือข่ายอินเทอร์เน็ต เนื่องจากต้องทำการทดลองบนเครื่อง 2 เครื่องพร้อมกันเพื่อทำการเปรียบเทียบผล พบว่าการทดลองจะมีตัวแปรที่เพิ่มเข้ามาเกี่ยวข้องหลายตัวเช่น Bandwidth ของระบบเครือข่ายขณะนั้น, การที่ระบบเครือข่ายมีการล่มจากหลาย ๆ สาเหตุ, ระบบไฟฟ้าขัดข้อง และ การถูกยกเลิกใช้งานเครือข่ายเพราะมีการใช้งานระบบเครือข่ายที่มากเกิน เนื่องจากการร้องขอข้อมูลในการทดสอบ ทำให้ถูกเข้าใจผิดว่าเป็นการทำงานของไวรัสคอมพิวเตอร์

6.4 แนวทางการปรับปรุงงานวิจัยในอนาคต

ในงานวิจัยนี้ได้นำเสนอเฉพาะในส่วนการปรับแต่งอัลกอริธึมการแทนที่ข้อมูลเดิมให้มีประสิทธิภาพที่ดีขึ้นโดยการนำคุณลักษณะของอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF และ LFUDA เข้ามารวมกัน โดยเพิ่มค่า Threshold สำหรับการตรวจสอบขนาดของไฟล์เพื่อเอาไว้เลือกชนิดของอัลกอริธึมการแทนที่ข้อมูล และหากทำให้เว็บแคชเซิร์ฟเวอร์สามารถปรับเปลี่ยนค่า Threshold ไปตามรูปแบบการร้องขอจะสามารถทำให้ได้ประสิทธิภาพของเว็บแคชเซิร์ฟเวอร์ได้ และในการทดลองได้กำหนดค่าของขนาดแคชของเว็บแคชเซิร์ฟเวอร์เป็นค่า 128 MB ซึ่งในความ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เป็นจริงค่าขนาดของแคชสามารถเพิ่มค่าได้ ซึ่งเมื่อขนาดแคชของเว็บแคชเซิร์ฟเวอร์มีค่าเพิ่มขึ้น จะมีผลทำให้ค่า %Hit และ ค่า %Byte-Hit มีค่าสูงขึ้น นอกจากนี้ ตัวแปรต่าง ๆ ในการพิจารณาการทำงานของอัลกอริธึมการแทนที่ข้อมูล นอกจากตัวแปรของขนาดไฟล์และเวลาที่ข้อมูลถูกร้องขอหรือเก็บลงไว้ในแคช ยังมีตัวแปรอีกหลายชนิด ที่มีผลต่อการทำงานของอัลกอริธึมการแทนที่ข้อมูล ซึ่งหากมีการพิจารณาและปรับแต่งอัลกอริธึมการแทนที่ข้อมูลด้วยตัวแปรหลาย ๆ ชนิดก็จะมีผลทำให้ค่าของ %Hit และ %Byte-Hit มีค่าเปลี่ยนแปลงเพิ่มขึ้นได้ด้วย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เอกสารอ้างอิง

[1] Dilley J., Arlitt M. and Perret S. "Enhancement and Validation of Squid's Cache Replacement Policy." Technical Report HPL-1999-69, Hewlett-Packard Laboratories, May 1999.

[2] Arlitt M. and Williamson C. "Internet Web Servers: Workload Characterization and Performance Implications." IEEE/ACM Transactions on Networking, vol. 5, no. 5, October 1997. pp. 631-645.

[3] Intraha W. "Proxy Performance Tuning for HTTP and FTP Protocol." Master Thesis of Information Science, King Mongkut's Institute of Technology Ladkrabang. 2001

[4] Squid Internet Object Cache Software. [Online]. Available : http://www.squid-cache.org/.

[5] CERN/W3C HTTP Daemon Software. [Online]. Available : http://www.w3.org/.

[6] Wessels D. and Claffy K. "Internet Cache Protocol (icp) version 2." Network Working Group RFC 2186, [Online]. Available : http://ds.internic.net/rfc/rfc2186.txt. September 1997.

[7] Netscape HTTP Daemon Software. [Online]. Available : http://www.netscape.com/.

[8] Delegate HTTP Daemon Software. [Online]. Available : http://www.delegate.org/.

[9] Microsoft Proxy Software. [Online]. Available : http://www.microsoft.com/.

[10] Wcol/Catalyst Proxy Software. [Online]. Available : http://shika.aist-nara.ac.jp/products/wcol/wcol.html/.

[11] Novell BorderManager Proxy Software. [Online]. Available : http://www.novell.com/.

[12] Apache HTTP Daemon Software. [Online]. Available : http://www.apache.org/.

[13] Cacheflow Proxy Software. [Online]. Available : http://www.cacheflow.com/.

[14] MOWS HTTP Daemon and Proxy Software. [Online]. Available : http://mows.rz.uni-mannheim.de/mows/.

[16] Williams S., Abrams M., Stanbridge C., Abdulla G. and Fox E. "Removal Policies in Network Caches for World-Wide Web Documents." Proc. ACM Sigcomm96, vol. 26, August 1996. pp. 293-305.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

[17] Abrams M., Stanbridge C., Abdulla G., Williams S. and Fox E. "Caching Proxies: Limitation and Potentials." WWW-4, December 1995.

[18] Wooster R. and Abrams M. "Proxy Caching the estimates Page Load Delays." Proc. 6th International World Wide Web Conference, 1997.

[19] Jin S. and Bestavros A. "Popularity-aware greedy dual-size Web proxy caching algorithms." Proc. 20th International Conference on Distributed Computing Systems, April 2000. pp. 254-261.

[20] Arlitt M., Cherkasova L., Dilley J., Friedrich R. and Jin T. "Evaluating Content Management Techniques for Web Proxy Caches." Technical Report HPL-98-173, Hewlett-Packard Laboratories, April 1999.

[21] Cherkasova L. "Improving WWW proxies Performance with Greedy-Dual-Size-Frequency Caching Policy." Technical Report HPL-1998-69R1, Hewlett-Packard Laboratories, November 1998.

[22] Podlipnig S. and Boszormenyi L. "A Survey of Web Cache Replacement Strategies." ACM Computing Surveys, vol. 35, no. 4, December 2003, pp. 374-398.

[23] Young N. "On-line caching as cache size varies." 2nd Annual ACM-SIAM Symposium on Discrete Algorithm, 1991. pp. 241-250.

[24] Cao P. and Irani S. "Cost Aware WWW Proxy Caching Algorithms." Proc. USENIX Symposium on Internet Technologies and Systems (USITS), December 1997, pp. 193-206.

[25] Redhat Linux Operating System Software. [Online]. Available : http://www.redhat.com/.

[26] Cfmc Software. [Online]. Available : http://www.cacheflow.com/.

[27] Internet Traffic Archive. [Online]. Available : http://ita.ee.lbl.gov/.

[28] Sopechoke P. and Piyatamrong B. "Improvement Web Cache Server with Hierarchical Replacement Algorithm." Proc. 7th National Computer Science and Engineering Conference 2003, October 2003. pp.140-144.

[29] Debian Linux Operating System Software. [Online]. Available : http://www.debian.org/.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ก.

ตัวอย่างไฟล์ Configuration ของ Squid ในส่วนที่มีการตั้งค่าสำหรับการเรียกใช้งานอัลกอริธึมการแทนที่ข้อมูล

```
# OPTIONS WHICH AFFECT THE CACHE SIZE
# -----------------------------------------------------------------------------

# TAG: cache_mem    (bytes)
#       NOTE: THIS PARAMETER DOES NOT SPECIFY THE MAXIMUM PROCESS SIZE.
#       IT ONLY PLACES A LIMIT ON HOW MUCH ADDITIONAL MEMORY SQUID WILL
#       USE AS A MEMORY CACHE OF OBJECTS. SQUID USES MEMORY FOR OTHER
#       THINGS AS WELL. SEE THE SQUID FAQ SECTION 8 FOR DETAILS.
#
#       'cache_mem' specifies the ideal amount of memory to be used
#       for:
#               * In-Transit objects
#               * Hot Objects
#               * Negative-Cached objects
#
#       Data for these objects are stored in 4 KB blocks.  This
#       parameter specifies the ideal upper limit on the total size of
#       4 KB blocks allocated.  In-Transit objects take the highest
#       priority.
#
#       In-transit objects have priority over the others.  When
#       additional space is needed for incoming data, negative-cached
#       and hot objects will be released.  In other words, the
#       negative-cached and hot objects will fill up any unused space
#       not needed for in-transit objects.
#
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
#       If circumstances require, this limit will be exceeded.
#       Specifically, if your incoming request rate requires more than
#       'cache_mem' of memory to hold in-transit objects, Squid will
#       exceed this limit to satisfy the new requests.  When the load
#       decreases, blocks will be freed until the high-water mark is
#       reached.  Thereafter, blocks will be used to store hot
#       objects.
#
#Default:
 cache_mem 16 MB


# TAG: cache_swap_low       (percent, 0-100)
# TAG: cache_swap_high      (percent, 0-100)
#
#       The low- and high-water marks for cache object replacement.
#       Replacement begins when the swap (disk) usage is above the
#       low-water mark and attempts to maintain utilization near the
#       low-water mark.  As swap utilization gets close to high-water
#       mark object eviction becomes more aggressive.  If utilization is
#       close to the low-water mark less replacement is done each time.
#
#       Defaults are 90% and 95%. If you have a large cache, 5% could be
#       hundreds of MB. If this is the case you may wish to set these
#       numbers closer together.
#
#Default:
 cache_swap_low 90
 cache_swap_high 95


# TAG: maximum_object_size (bytes)
#       Objects larger than this size will NOT be saved on disk.  The
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
#       value is specified in kilobytes, and the default is 4MB.  If
#       you wish to get a high BYTES hit ratio, you should probably
#       increase this (one 32 MB object hit counts for 3200 10KB
#       hits).  If you wish to increase speed more than your want to
#       save bandwidth you should leave this low.
#
#       NOTE: if using the LFUDA replacement policy you should increase
#       this value to maximize the byte hit rate improvement of LFUDA!
#       See replacement_policy below for a discussion of this policy.
#
#Default:
 maximum_object_size 4096 KB


# TAG: minimum_object_size (bytes)
#       Objects smaller than this size will NOT be saved on disk.  The
#       value is specified in kilobytes, and the default is 0 KB, which
#       means there is no minimum.
#
#Default:
 minimum_object_size 0 KB


# TAG: maximum_object_size_in_memory     (bytes)
#       Objects greater than this size will not be attempted to kept in
#       the memory cache. This should be set high enough to keep objects
#       accessed frequently in memory to improve performance whilst low
#       enough to keep larger objects from hoarding cache_mem .
#
#Default:
 maximum_object_size_in_memory 8 KB


# TAG: ipcache_size   (number of entries)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
# TAG: ipcache_low    (percent)
# TAG: ipcache_high   (percent)
#         The size, low-, and high-water marks for the IP cache.
#
#Default:
 ipcache_size 1024
 ipcache_low 90
 ipcache_high 95

# TAG: fqdncache_size         (number of entries)
#         Maximum number of FQDN cache entries.
#
#Default:
 fqdncache_size 1024

# TAG: cache_replacement_policy
#         The cache replacement policy parameter determines which
#         objects are evicted (replaced) when disk space is needed.
#
#           lru      : Squid's original list based LRU policy
#           heap GDSF : Greedy-Dual Size Frequency
#           heap LFUDA: Least Frequently Used with Dynamic Aging
#           heap LRU  : LRU policy implemented using a heap
#
#         Applies to any cache_dir lines listed below this.
#
#         The LRU policies keeps recently referenced objects.
#
#         The heap GDSF policy optimizes object hit rate by keeping smaller
#         popular objects in cache so it has a better chance of getting a
#         hit.  It achieves a lower byte hit rate than LFUDA though since
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
#       it evicts larger (possibly popular) objects.
#
#       The heap LFUDA policy keeps popular objects in cache regardless of
#       their size and thus optimizes byte hit rate at the expense of
#       hit rate since one large, popular object will prevent many
#       smaller, slightly less popular objects from being cached.
#
#       Both policies utilize a dynamic aging mechanism that prevents
#       cache pollution that can otherwise occur with frequency-based
#       replacement policies.
#
#       NOTE: if using the LFUDA replacement policy you should increase
#       the value of maximum_object_size above its default of 4096 KB to
#       to maximize the potential byte hit rate improvement of LFUDA.
#
#       For more information about the GDSF and LFUDA cache replacement
#       policies see http://www.hpl.hp.com/techreports/1999/HPL-1999-69.html
#       and http://fog.hpl.external.hp.com/techreports/98/HPL-98-173.html.
#
#Default:
 cache_replacement_policy heap GDSF

# TAG: memory_replacement_policy
#       The memory replacement policy parameter determines which
#       objects are purged from memory when memory space is needed.
#
#       See cache_replacement_policy for details.
#
#Default:
 memory_replacement_policy heap GDSF
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
# LOGFILE PATHNAMES AND CACHE DIRECTORIES
# ------------------------------------------------------------------------

# TAG: cache_dir
#       Usage:
#
#       cache_dir Type Directory-Name Fs-specific-data [options]
#
#       cache_dir diskd Maxobjsize Directory-Name MB L1 L2 Q1 Q2
#
#       You can specify multiple cache_dir lines to spread the
#       cache among different disk partitions.
#
#       Type specifies the kind of storage system to use. Only "ufs"
#       is built by default. To eanble any of the other storage systems
#       see the --enable-storeio configure option.
#
#       'Directory' is a top-level directory where cache swap
#       files will be stored.  If you want to use an entire disk
#       for caching, then this can be the mount-point directory.
#       The directory must exist and be writable by the Squid
#       process.  Squid will NOT create this directory for you.
#
#       The ufs store type:
#
#       "ufs" is the old well-known Squid storage format that has always
#       been there.
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
#
#       cache_dir ufs Directory-Name Mbytes L1 L2 [options]
#
#       'Mbytes' is the amount of disk space (MB) to use under this
#       directory.  The default is 100 MB.  Change this to suit your
#       configuration.  Do NOT put the size of your disk drive here.
#       Instead, if you want Squid to use the entire disk drive,
#       subtract 20% and use that value.
#
#       'Level-1' is the number of first-level subdirectories which
#       will be created under the 'Directory'.  The default is 16.
#
#       'Level-2' is the number of second-level subdirectories which
#       will be created under each first-level directory.  The default
#       is 256.
#
#       The aufs store type:
#
#       "aufs" uses the same storage format as "ufs", utilizing
#       POSIX-threads to avoid blocking the main Squid process on
#       disk-I/O. This was formerly known in Squid as async-io.
#
#       cache_dir aufs Directory-Name Mbytes L1 L2 [options]
#
#       see argument descriptions under ufs above
#
#       The diskd store type:
#
#       "diskd" uses the same storage format as "ufs", utilizing a
#       separate process to avoid blocking the main Squid process on
#       disk-I/O.
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
#
#       cache_dir diskd Directory-Name Mbytes L1 L2 [options] [Q1=n] [Q2=n]
#
#       see argument descriptions under ufs above
#
#       Q1 specifies the number of unacknowledged I/O requests when Squid
#       stops opening new files. If this many messages are in the queues,
#       Squid won't open new files. Default is 64
#
#       Q2 specifies the number of unacknowledged messages when Squid
#       starts blocking.  If this many messages are in the queues,
#       Squid blocks until it recevies some replies. Default is 72
#
#       Common options:
#
#       read-only, this cache_dir is read only.
#
#       max-size=n, refers to the max object size this storedir supports.
#       It is used to initially choose the storedir to dump the object.
#       Note: To make optimal use of the max-size limits you should order
#       the cache_dir lines with the smallest max-size value first and the
#       ones with no max-size specification last.
#
#Default:
 cache_dir ufs /squid/cache 128 16 256
```

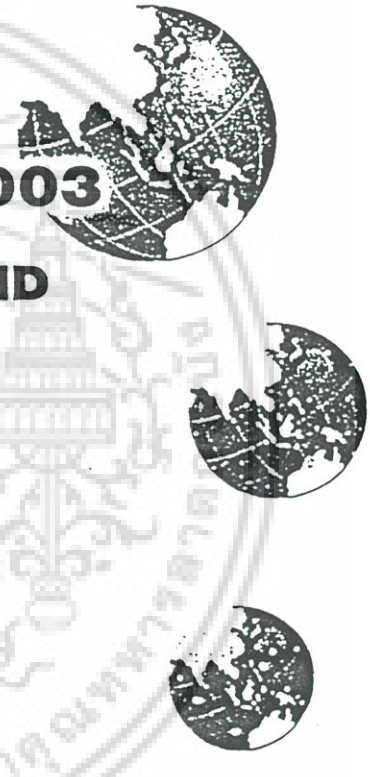
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ข.
ผลงานวิจัยในระหว่างการศึกษาที่ได้รับการตีพิมพ์เผยแพร่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

NCSEC2003

The 7th National Computer Science and Engineering Conference



October 28-30, 2003

Chonburi, THAILAND

Organized by :

Department of Computer Science,

Faculty of Science,

Burapha University

ISBN : 974-382-604-1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Presentation Session 2

Computer Security

A Comparative Analyzes of Over-Issuing Delta-CRLs with Distribution Points with Existing Certificate Revocation Methods

Aradee Rojanapasakorn, Chanboon Sathitwiriyawong 91

Privacy Control for Personal Information Databases

Yibing Kong, Janusz R. Getta, Ping Yu, Jennifer Seberry 97

An Enhanced Role-Based Access Control Model using Constraint Features

Burin Yenmunkong, Chanboon Sathitwiriyawong 103

A Simple Approach to Conflict Resolution for Access Control in XML

Tuangporn Tantichalermpan, Yongyuth Permpoontanalarp 109

Computer Networks

Multichannel MAC Protocol for Ad-Hoc Wireless Networks

Sudthida Wiwatthanasaranrom, Anan Phonphoem 115

Proportional Differentiation Service Based on Complete Buffer Partitioning

Borrirux Polyiam, Werasak Kurutach 121

การพัฒนาการหาเส้นทางในเครือข่ายสแคทเทอร์เนทของบลูทูธ

Thidarat Tawsook, Anan Phonphoem 128

Increasing Foreign Web Access Speed Using Grid Based Routing Proxy

Peerapon Vateekul, Arnon Rungsawang 134

การปรับแต่งเว็บแคชเซิร์ฟเวอร์ด้วยอัลกอริธึมการแทนที่ข้อมูล (Replacement Algorithm) แบบ Hierarchical

ภาวิน สพโชค บรรจง ปิยธำรง 140

Data Mining and Databases

Identifying Communities in Citation Graph

Panupong Wanjantuk 145

Text Summarization using Singular Value Decomposition for Thai

Ausdang Thangthai, Chuleerat Jaruskulchai 151

Towards a Unified Framework of Meta-Rules Guided Mining Based on Ontology

Jaremsri L. Mitrpanont, Preecha Tangworakitthaworn 157

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การปรับแต่งเว็บแคชเซิร์ฟเวอร์ด้วยอัลกอริธึมการแทนที่ข้อมูล(Replacement Algorithm) แบบ Hierarchical

ภาวิน สพโชค, บรรจง ปิยธำรง
ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
แขวงลำปำทิว เขตลาดกระบัง กรุงเทพฯ 10520
E-mail: s2061088, kpbanjon@kmitl.ac.th

บทคัดย่อ

บทความนี้นำเสนอการวิเคราะห์ลักษณะการเรียกใช้ข้อมูลเว็บผ่านเว็บแคชเซิร์ฟเวอร์ โดยพิจารณาจากการเปลี่ยนแปลงอัลกอริธึมการแทนที่ข้อมูล (Replacement Algorithm) [4]ของข้อมูลจำนวนหนึ่งที่ได้จากการเรียกใช้ของเครื่องลูกข่ายในระยะเวลาหนึ่ง ค่าที่นำมาวิเคราะห์คือปริมาณการร้องขอใช้งาน ขนาดข้อมูลต่อไอพี ประเภทของข้อมูลที่ร้องขอ เนื่องจากค่าเหล่านี้จะเปลี่ยนแปลงไปตามกลุ่มผู้ใช้งาน และงานวิจัยนี้พิจารณากลุ่มผู้ใช้งานในระบบเครือข่ายของสถาบัน (Campus Network) เป็นหลัก โดยหาความสัมพันธ์ของอัลกอริธึมการจัดวางเว็บแคชเซิร์ฟเวอร์ ที่มีการตั้งค่าอัลกอริธึมการแทนที่ข้อมูลในหลายรูปแบบ เนื่องจากถ้ามีการตั้งค่าของเว็บแคชเซิร์ฟเวอร์ไม่ตรงกับอัลกอริธึมการร้องขอข้อมูล จะทำให้ข้อมูลที่ได้ผิดรับพลาดไปจากข้อมูลที่ควรจะได้และ ทำให้สิ้นเปลืองแบนด์วิธของระบบเครือข่าย เนื่องจากเว็บแคชเซิร์ฟเวอร์ต้องไปนำข้อมูลมาให้ผู้ใช้หลายครั้ง ซึ่งข้อมูลที่ผู้ใช้ร้องขอควรมีเก็บเอาไว้อยู่ในแคชของเซิร์ฟเวอร์

Abstract

This paper describes about analysis of data from web cache server by scope in replacement algorithm of data that collected from client. Data that use for analysis are Request from all client, Size of object and Type of request which all of data will change depend on users. This paper scope in users on campus network for finding relation of several replacement algorithms in each cache server so if web cache server had not been configuration support the requests of users, it will make mistake of data that they need and loss bandwidth of campus network because web cache server should take same data from internet many time.

1. บทนำ

ในปัจจุบันอัตราการเติบโตของการใช้เว็บได้เพิ่มขึ้นมาอย่างรวดเร็ว ทั้งในส่วนของการร้องขอข้อมูลผ่านเว็บทั้งในส่วนของ รูปภาพ ตัวอักษร ไฟล์วิดีโอ ภาพเคลื่อนไหว ข้อมูลเหล่านี้จะมีลักษณะที่แตกต่างกันออกไป และจากการเติบโตของการใช้งานนี้ทำให้เกิดปัญหาการกับคั่งของข้อมูลบนเครือข่าย ผลที่เกิดขึ้นคือ การล่าช้าของการได้รับข้อมูลที่ต้องการ

เว็บแคชเซิร์ฟเวอร์ได้ถูกพัฒนาขึ้นเพื่อลดปัญหาที่เกิดขึ้นดังที่กล่าวมาแล้ว หลักการทำงานของเว็บแคชเซิร์ฟเวอร์คือรอรับการร้องขอข้อมูลจากไคลเอนต์ หากเว็บแคชเซิร์ฟเวอร์พบข้อมูลที่ร้องขอในแคชและยังไม่หมดอายุจะทำการส่งข้อมูลที่มีอยู่แล้วไปให้ไคลเอนต์ แต่ถ้าไม่พบข้อมูลหรือข้อมูลที่พบหมดอายุไปแล้ว จะทำการร้องขอไปยังต้นทางของข้อมูลเพื่อนำข้อมูลชุดใหม่มา และทำการส่งไปให้ไคลเอนต์ที่ร้องขอ พร้อมกับทำการสำเนาเก็บเอาไว้ในแคชด้วยเพื่อให้บริการสำหรับคำร้องขอข้อมูลนั้นในครั้งต่อไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2. งานวิจัยที่เกี่ยวข้อง

การพัฒนาความสามารถของแคชเซิร์ฟเวอร์จะสามารถพัฒนาได้หลายส่วนของระบบแคช โดยจุดมุ่งหมายหลักของการพัฒนาคือ เพิ่มสมรรถนะด้านความเร็ว ความเชื่อถือได้ของข้อมูลที่ส่งให้กับไคลเอนต์

ปกติประสิทธิภาพการทำงานของเว็บแคชเซิร์ฟเวอร์นิยมวัดจากอัตราการพบ (Hit Ratio), อัตราการพบแบบไบต์ (Byte-Hit Ratio) และค่าเวลาเฉลี่ยของข้อมูลตอบกลับ (Mean Response Time) ค่าทั้ง 3 ขึ้นกับหน่วยประมวลผลกลาง ความจุของหน่วยความจำ, ขนาดของดิสก์, การใช้งานของดิสก์และซอฟท์แวร์ที่ทำหน้าที่เป็นแคชที่แตกต่างกันไป นอกจากนี้องค์ประกอบอีกหนึ่งที่มีผลกระทบต่อสมรรถนะของเว็บแคชเซิร์ฟเวอร์คือ ลักษณะการใช้งานของผู้ใช้งานในระบบเครือข่าย หากนำข้อมูลที่ได้มาวิเคราะห์หาพารามิเตอร์ที่เป็นองค์ประกอบสำคัญของการทำงานของแคชแล้ว จะสามารถเพิ่มความสามารถของแคชเซิร์ฟเวอร์ได้โดยไม่ต้องหาโปรแกรมมาเพิ่มเติมให้กับตัวแคชเซิร์ฟเวอร์

สำหรับงานวิจัยนี้มุ่งเน้นการเพิ่มสมรรถนะเว็บแคชเซิร์ฟเวอร์โดยศึกษารูปแบบการจัดวางระบบแคชเซิร์ฟเวอร์และอัลกอริธึมการแทนที่ข้อมูล (Replacement Algorithm) ที่เหมาะสมกับรูปแบบของข้อมูลที่วิ่งอยู่ในระบบเครือข่าย โดยจะวิเคราะห์ตามประเภทของข้อมูลแต่ละชนิดเช่น ข้อมูลรูปภาพ ข้อมูลเท็กซ์ และข้อมูลอื่น ๆ

รูปแบบของอัลกอริธึมการแทนที่ข้อมูล ที่นำมาวิเคราะห์จะแบ่งได้ดังนี้

Least-Recently-Used (LRU)

Greedy-Dual-Size-Frequency (GDSF) [3, 5]

Least-Frequency-Used with Dynamic Aging (LFUDA)

หลักการทำงานของ อัลกอริธึมการแทนที่ข้อมูลแต่ละแบบ

LRU เป็นการแทนที่ข้อมูลโดยพิจารณาข้อมูลที่ไม่ได้รับการถูกร้องขอ เมื่อนับจากปัจจุบันเป็นเวลานานที่สุด นิยมใช้ทั่วไปในการทำงานของแคชเซิร์ฟเวอร์ อย่างไรก็ตามพบว่าอัลกอริธึมนี้ไม่เหมาะกับงานบางประเภทเมื่อเทียบกับอัลกอริธึมการแทนที่ข้อมูลแบบอื่น

GDSF ได้ถูกพัฒนามาจาก อัลกอริธึมการแทนที่ข้อมูลแบบ GD-Size โดยจะพิจารณาการเลือกข้อมูลออกโดยดูจากข้อมูลที่มีการใช้งานต่ำที่สุด ซึ่งจะมีตัวแปรที่ใช้สำหรับการเลือกข้อมูลออก (K_i) ดังสมการ

$$K_i = C_i / S_i + L$$

โดย C_i คือ ค่าที่นำมารวมเมื่อนำข้อมูลเข้ามาในแคช

S_i คือ ขนาดของข้อมูล

L คือ องค์ประกอบอายุของข้อมูลที่เริ่มตั้งแต่ 0 และเพิ่มขึ้นเรื่อย ๆ เมื่อมีการแทนที่ข้อมูลเข้าไปในแคช

ในกรณีของ อัลกอริธึมการแทนที่แบบ GDSF จะมีการเพิ่มในส่วนของความถี่จำนวนครั้ง (F_i) ในการเรียกใช้ข้อมูลตัวนั้นเข้าไปด้วยดังสมการ

$$K_i = F_i * C_i / S_i + L$$

ซึ่งในการทำงานจะให้ค่าของ C_i มีค่าเท่ากับ 1 เราจะเรียกรูปแบบนี้ว่า GDSF-Hits

LFUDA เป็นการแทนที่ข้อมูลที่ถูกร้องขอ โดยพิจารณาจากจำนวนครั้งของการเรียกใช้ข้อมูลนั้น แต่จะเพิ่มในส่วนของอายุของข้อมูลที่เก็บเอาไว้ในแคช (L) ดังสมการ

$$K_i = C_i * F_i + L$$

3. รูปแบบของข้อมูลที่นำมาวิเคราะห์

ข้อมูลที่ได้ มาจากการจำลองการทำงานของเว็บแคชเซิร์ฟเวอร์ที่อ้างอิงชุดข้อมูลจากเว็บแคชเซิร์ฟเวอร์ของภาควิชาวิศวกรรมคอมพิวเตอร์ภายใน 1 วัน โดยการจำลองนี้ใช้เวลาในการจำลองเป็นเวลาเฉลี่ยครั้งละประมาณ 30 ชั่วโมง

ข้อมูลที่นำมาวิเคราะห์จะต้องมีรูปแบบดังนี้ เป็นการทำงานที่ร้องขอผ่านทางโปรโตคอล เอชทีทีพี เท่านั้น (ไม่รวมถึงการให้บริการผ่านโปรโตคอลตัวอื่น เช่น เอฟทีพี) ข้อมูลจะต้องผ่านการร้องขอใช้งานด้วยความถูกต้อง นอกจากนั้นจะไม่นับรู

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

อาร์แอลที่มีลักษณะแบบไดนามิกส์เช่น cgi, php, asp และยูอาร์แอลที่มีสัญลักษณ์พิเศษประกอบอยู่เช่น ?, =, %, +, *, &, @, $, [ ], ( ) โดยยูอาร์แอลที่มีลักษณะเช่นนี้เป็นยูอาร์แอลที่ไม่ได้ระบุถึงข้อมูลเว็บเพจที่สามารถนำมาใช้วิเคราะห์สำหรับการวิเคราะห์ในส่วนเปอร์เซ็นต์ของอัตราการพบและ อัตราการพบแบบไบต์จะพิจารณาเฉพาะบรรทัดที่มีคำว่า TCP_HIT, TCP_MEM_HIT,TCP_REFRESH_HIT,TCP_REF_FAIL_HIT และ TCP_IMS_HIT นอกเหนือจากที่กล่าวมาถือว่าไม่พบข้อมูลในแคช

4. งานวิจัยและการวิเคราะห์ข้อมูล

4.1. ลักษณะของระบบที่ทำการทดสอบ

ข้อมูลที่นำมาวิเคราะห์มาจากการจัดเก็บข้อมูลจากการจำลองการร้องขอไปยังแคชเซิร์ฟเวอร์ของภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง โดยการจำลองจะนำรูปแบบการร้องของที่เกิดขึ้นใน 1 วัน และใช้โปรแกรม cinc [7] ทำการสร้างการร้องขอไปยังแคชเซิร์ฟเวอร์ที่กำหนด ทำการทดลองพร้อมกันทั้ง 3 รูปแบบของอัลกอริธึมการแทนที่ข้อมูล โดยเครื่องที่นำมาทดสอบ คือเครื่องของบริษัท HP รุ่น Kayak หน่วยประมวลผลของ Intel รุ่น Pentium III 866 เมกาเฮิรตซ์ หน่วยความจำ 256 เมกะไบต์ ฮาร์ดดิสก์ชนิด IDE ความจุ 20 กิกะไบต์ ระบบปฏิบัติการคือ Linux ของ debian เวอร์ชั่น 3.0 [8] และเครื่องคอมพิวเตอร์ทั้งหมดใช้โปรแกรมสควิดซ์ เวอร์ชั่น 2.4 Stable6 [6] เพื่อทำหน้าที่เป็นเว็บแคชเซิร์ฟเวอร์และกำหนดเนื้อที่สำหรับการเก็บข้อมูลของแคชไว้ขนาด 128 เมกกะไบต์ ระบบเครือข่ายต่อแบบอีเทอร์เน็ต และช่องทางการติดต่อภายนอกจำนวน 3 ช่องทางคือ ส่วนที่ออกต่างประเทศและภายในประเทศจะผ่านทางเครือข่ายของบริษัทเอเชียอินโฟเน็ต ส่วนที่ออกไปสู่มหาลัยอื่นจะผ่านทางเครือข่ายของยูนิเน็ต และส่วนที่ออกไปยังเครือข่ายของไทยสาร

จากข้อมูลที่ได้ทำการเก็บรวบรวมจากวิธีข้างต้น ถูกนำมาวิเคราะห์เพื่อหาพารามิเตอร์ที่เกี่ยวข้องกับการปรับแต่งแคชเซิร์ฟเวอร์ [1] โดยสามารถวิเคราะห์เป็นส่วน ๆ ดังนี้

%Hit คือค่าอัตราส่วนระหว่าง จำนวนการร้องขอที่ถือว่าเป็นการ Hit ต่อ จำนวนการร้องขอทั้งหมด แล้วนำมาคูณด้วย 100 เพื่อทำเป็นเปอร์เซ็น

%ByteHit คือค่าอัตราส่วนระหว่าง ขนาดข้อมูลของจำนวนการร้องขอที่ถือว่าเป็นการ Hit ต่อ ขนาดข้อมูลของจำนวนการร้องขอทั้งหมด แล้วนำมาคูณด้วย 100 เพื่อทำเป็นเปอร์เซ็นต์

ค่าที่ได้แสดงได้ดังกราฟข้างล่างนี้

รูปที่ 1 %Hit และ %ByteHit ของการแทนที่แบบ LRU

รูปที่ 2 %Hit และ %ByteHit ของการแทนที่แบบ GDSF

รูปที่ 3 %Hit และ %ByteHit ของการแทนที่แบบ LFUDA

- ค่าในแกน X แสดงถึง จำนวนครั้งของการร้องขอที่เกิดขึ้น เริ่มจาก 0 ไปถึง 400,000 (4×10^5) ครั้ง

- ค่าในแกน Y แสดงถึง %Hit สำหรับภาพด้านซ้ายและ %ByteHit สำหรับภาพด้านขวา (สำหรับภาพของ %ByteHit ค่าในแกน Y ของ LRU และ LFUDA จะสูงสุดที่ 70% แต่ GDSF จะสูงสุดที่ 60%)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โดยข้อมูลที่นำมาทดสอบประกอบด้วยชนิดของข้อมูลหลัก ๆ ดังนี้

ไฟล์รูปภาพนามสกุล GIF	55.8 %
ไฟล์รูปภาพนามสกุล JPEG	18.1 %
ไฟล์นามสกุล HTML	7.5 %
อื่น ๆ	18.6 %

และแบ่งตามช่วงขนาดของข้อมูลได้ดังนี้

100 – 999 Byte	37.4 %
1000 – 9999 Byte	51.4 %
10000 – 99999 Byte	10.9 %
อื่น ๆ	0.3 %

4.2. การทดสอบรูปแบบแคชเซิร์ฟเวอร์ที่มีอัลกอริธึมการแทนที่ข้อมูลแบบ Hierarchical

ในการทดสอบการเชื่อมต่อเว็บแคชเซิร์ฟเวอร์ให้มีรูปแบบการแทนที่เป็นแบบ Hierarchical [2] ระบบทั้ง 2 ติดต่อกันผ่านโปรโตคอลชื่อว่า Internet Cache Protocol (ICP) [9] (โปรโตคอล ICP มีหลักการทำงานดังนี้: เมื่อเซิร์ฟเวอร์ได้รับการร้องขอวัตถุ แต่ไม่มีวัตถุอยู่ในแคชเซิร์ฟเวอร์จะส่งสัญญาณไปยังเซิร์ฟเวอร์เครื่องอื่น เพื่อเช็คว่ามีข้อมูลหรือไม่ หากเซิร์ฟเวอร์เครื่องอื่นมี จะร้องขอข้อมูลมาจากเซิร์ฟเวอร์เครื่องอื่นแล้วส่งไปให้เครื่องไคลเอนต์ที่ร้องขอต่อไป พร้อมกับสำเนาข้อมูลเก็บไว้ในแคชของตัวเองด้วย) เพื่อหารูปแบบที่เหมาะสมของการแทนที่ข้อมูลที่จะใช้กับระบบเว็บแคชเซิร์ฟเวอร์แบบ Parent และ Child จัดรูปแบบได้ดังนี้

4.2.1. ระบบเว็บแคชเซิร์ฟเวอร์ที่ทำหน้าที่เป็น Child

ใช้อัลกอริธึมการแทนที่ข้อมูลแบบ GDSF เนื่องจากมี %Hit ของไฟล์ที่มีขนาดเล็ก โดยส่วนใหญ่เป็นไฟล์รูปภาพมากที่สุด และการแทนที่ข้อมูลแบบนี้จะเหมาะกับชนิดของข้อมูลในระบบเครือข่ายมากที่สุด

4.2.2. ระบบเว็บแคชเซิร์ฟเวอร์ที่ทำหน้าที่เป็น Parent

ทำการทดสอบโดยนำอัลกอริธึมการแทนที่ข้อมูลทั้ง 3 แบบมาทดสอบคือ LRU, LFUDA และ GDSF

การทดสอบได้ส่งการร้องขอด้วยข้อมูลชุดเดิมไปยังระบบที่ทำหน้าที่เป็น Child และเก็บข้อมูลของการทดสอบทั้ง 3 แบบ ได้ผลดังนี้ (ค่าเปอร์เซ็นต์ที่ได้ จะมาจากข้อมูลที่มีการ Hit บน Parent ต่อข้อมูลทั้งหมดที่นำมาจาก Parent)

ชนิดการแทนที่ข้อมูล	%Hit	%ByteHit
LRU	1.02	20.34
LFUDA	1.11	21.42
GDSF	0.45	0.86

และจากกราฟที่ได้แสดงไว้ข้างล่าง เป็นการเปรียบเทียบ % Hit และ % ByteHit ของการแทนที่แต่ละแบบ

รูปที่ 4 %Hit และ %ByteHit ของการแทนที่แบบ LRU

รูปที่ 5 %Hit และ %ByteHit ของการแทนที่แบบ GDSF

รูปที่ 6 %Hit และ %ByteHit ของการแทนที่แบบ LFUDA

กราฟที่ได้เป็น %Hit ที่พบบน Child ซึ่งการพบเป็นแบบ Refresh_Hit (หมายถึงออบเจกต์ที่ไคลเอนท์ร้องขอพบอยู่ในแคชแต่หมดอายุ ดังนั้น Squid ได้ส่งการร้องขอแบบ If-Modify-Since ไปยังเซิร์ฟเวอร์ต้นทางและได้รับคำตอบกลับมา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ทาง HTTP reply ว่า "304 Not modified" หมายถึงว่าออบเจกต์นั้นยังไม่มีการเปลี่ยนแปลง (ขนาดและ/หรือวันที่สร้าง) ดังนั้น squid จึงส่งออบเจกต์ที่เก็บอยู่ในแคชไปให้ไคลเอนท์ พร้อมทั้งเริ่มนับเวลาที่ออบเจกต์ถูกเก็บในแคชใหม่) ดังนั้นการดูว่ามีการนำข้อมูลที่อยู่ใน Parent มามากน้อยแค่ไหน ให้ดูจากค่าของ %Hit และ %ByteHit หากมีค่าน้อยแสดงว่ามีการดึงข้อมูลมาจาก Parent มาก

5. สรุปผลและวิเคราะห์

จากการทดลองสร้างการร้องขอไปยัง Proxy ที่ใช้อัลกอริธึมการแทนที่ (Replacement Algorithm) ทั้ง 3 แบบ พบว่า %Hit ของทั้ง 3 แบบ มีค่าใกล้เคียงกันมากแทบไม่แตกต่างกัน แต่เมื่อสังเกตในส่วนของ %ByteHit พบว่าค่า %ByteHit ของกระบวนการแทนที่แบบ GDSF มีค่าต่ำเมื่อเทียบกับอัลกอริธึมแบบ LRU และ LFUDA เนื่องจาก %Hit ส่วนใหญ่ของ GDSF เกิดจากการ Hit ของไฟล์ที่มีขนาดเล็ก ดังนั้น %ByteHit จึงมีค่าน้อยตามไปด้วย

จากการแยกชนิดของข้อมูลพบว่า ข้อมูลส่วนใหญ่ที่มีการดาวน์โหลดมาเป็นไฟล์รูปภาพที่มีขนาดเล็ก ซึ่งเหมาะสมกับการใช้อัลกอริธึมการแทนที่แบบ GDSF ที่สุด แต่จะเจอปัญหาในกรณีที่เมื่อมีการร้องขอไฟล์ที่มีขนาดใหญ่ (เป็นเปอร์เซ็นต์ส่วนน้อย) ทำให้เกิดการ Miss ของข้อมูลได้ ซึ่งการใช้งานระบบ proxy ควรที่จะพิจารณาถึงอัลกอริธึมการแทนที่ ที่เหมาะสมกับชนิดของข้อมูลด้วย

ในการทดสอบการใช้งานอัลกอริธึมการแทนที่ข้อมูลหลาย ๆ แบบพร้อมกันเพื่อให้สนับสนุนจุดด้อยของแต่ละรูปแบบ ได้ผลดังนี้

5.1. อัลกอริธึม GDSF+GDSF

จากการทดลองพบได้ว่า %Hit และ %ByteHit รวมบน Parent มีค่าต่ำสุดเมื่อเทียบกับอัลกอริธึมอื่น ถึงแม้ว่ากราฟของ %Hit และ %ByteHit มีค่าสูง แต่ค่าที่สูงส่วนใหญ่คือ ค่าที่เป็นการ Hit แบบ Refresh_Hit บน Child ด้วย ทำให้ไม่มีการดึงข้อมูลไปจาก Parent แต่เอาข้อมูลในแคชของเครื่อง Child ไปแทน และเนื่องจาก Child และ Parent ใช้อัลกอริธึมการแทนที่ข้อมูลเหมือนกัน ทำให้ข้อมูลที่เก็บอยู่ในตัวเครื่องคล้าย ๆ กัน

5.2. อัลกอริธึม GDSF+LRU และ GDSF+LFUDA

จากการทดลองเห็นได้ว่า %Hit และ %ByteHit รวมบน Parent ของอัลกอริธึมการแทนที่ทั้ง 2 แบบให้ผลใกล้เคียงกัน แต่กราฟของ LFUDA มีค่าต่ำกว่ากราฟของ LRU และเมื่อพิจารณาในส่วนของ %ByteHit ค่าที่ได้ต่ำมาก เนื่องจากอัลกอริธึมการแทนที่แบบ LFUDA ยังเก็บชนิดของข้อมูลที่มีขนาดใหญ่และได้ถูกร้องขอมาหลาย ๆ ครั้งในการร้องขอข้อมูลในครั้งก่อน ๆ เอาไว้แต่ LRU ดูเฉพาะอายุของข้อมูลหากไม่ได้มีการเรียกใช้นาน ก็ทำการลบข้อมูลนั้นทิ้งไป ดังนั้นหากเว็บแคชเซิร์ฟเวอร์มีแคชที่มีขนาดใหญ่อัลกอริธึมการแทนที่แบบ LFUDA จะให้ประสิทธิภาพที่ดีกว่าแบบ LRU เนื่องจากจะทำการเก็บข้อมูลที่มีการเรียกใช้บ่อย ๆ เอาไว้เป็นหลัก

6. เอกสารอ้างอิง

[1] Martin F. Arlit and Carey L. Williamson, "Internet Web Servers: Workload Characterization and Performance Implications", Proceeding of IEEE/ACM Transactions On Networking, Vol 5, No. 5, October 1997

[2] Yu, H.; Estrin, D.; Govindan, R., "A hierarchical proxy architecture for Internet-scale event services", Proceedings of Enabling Technologies: Infrastructure for Collaborative Enterprises, 1999. (WET ICE '99) IEEE 8th International Workshops on, 16-18 June 1999 Page(s): 78-83

[3] Shudong Jin; Bestavros, A., "Popularity-awar greedy dual-size Web proxy caching algorithms", Proceedings of Distributed Computering Systems, 2000, 20th International Conference on, 10-13 April 2000 Page(s):254-261

[4] John Dilley, Martin Arlitt, Stephane Perret, "Enhancement and Validation of Squid's Cache Replacement Policy", Technical Report HPL-1999-69, Hewlett-Packard Laboratories, May, 1999

[5] L. Cherkasova, "Improving WWW proxies Performance with Greedy-Dual-Size-Frequency Caching Policy", Technical Report HPL-1998-69R1, Hewlett-Packard Laboratories, Nov. 1998

[6] Squid Cache Software, Homepage, Accessible on the Internet at World Wide Web URL http://www.squid-cache.org

[7] Cfme Software, Homepage, Accessible on the Internet at World Wide Web URL http://www.cacheflow.com

[8] Debian Linux Software, Homepage, Accessible on the Internet at World Wide Web URL http://www.debian.org

[9] D. Wessels and K. Claffy, Internet Cache Protocol (icp), version 2, Network Working Group RFC 2186, September 1997. http://ds.internic.net/rfc/rfc2186.txt.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ค.
ผลงานวิจัยในระหว่างการศึกษาที่ได้รับการตอบรับและกำลังรอการตีพิมพ์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Acceptance Letter

2004 International Conference on Control, Automation and Systems (ICCAS 2004)

August 25-27, 2004
The Shangri-La Hotel, Bangkok, Thailand

Home page : http://www.kmitl.ac.th/iccas04
http://www.iccas.org

■ **General Chair**
Prakit Tangtisanon (KMITL, President, Thailand)

■ **General Co-Chair**
Keh Sik Min (ICASE, President, Korea)

■ **Advisory Council**
Taweesak Koranantakul (NECTEC, Thailand)
Hironobu Ono (SICE, Japan)
Han Fu Chen (CAA, China)
Youngil Youm (POSTECH, Korea)
Hidenori Kimura (RIKEN, Japan)
Toshio Wakabayashi (Tokai Univ., Japan)
Monai Krairiksh (KMITL, Thailand)

■ **Organizing Chair**
Tawil Paungma (KMITL, Thailand)

■ **Organizing Co-Chair**
Sangchul Won (POSTECH, Korea)

■ **Conference Program Chair**
Ouen Pinngern (KMITL, Thailand)

■ **Technical Co-Chairs**
Jun-ichi Takada (TIT, Japan)
Ruttikorn Varakulsiripunth (KMITL, Thailand)
Narong Yoothanom (SPU, Thailand)
David Banjerdpongchai (CU, Thailand)
Akachai Sang-In (CMU, Thailand)
Sujate Jantarang (MUT, Thailand)
Waree Kongpraweechon (TU, Thailand)
Sinchai Chinvorarat (KMITNB, Thailand)
Suthee Phoojaruenchanachai (NECTEC, Thailand)
Bhaisal Hoonkeo (SAU, Thailand)

■ **International Relations Co-Chairs**
Jongkol Ngarmwiwit (KMITL, Thailand)
Hyun Sik Ahn (Kookmin Univ., Korea)
Yutaka Yamamoto (Kyoto Univ., Japan)
Apinunt Thanachayanont (KMITL, Thailand)
Sunpasit Limnararat (KMITL, Thailand)

■ **Regional Program Co-Chairs**
Joonhong Lim (Hanyang Univ., Korea)
Noriyuki Komine (Tokai Univ., Japan)

■ **Exhibition Co-Chairs**
Werachet Khamngern (KMITL, Thailand)
Jin Bae Park (Yonsei Univ., Korea)
Mongkol Mongkolwongroj (KMITL, Thailand)

■ **Publication Co-Chairs**
Nopporn Chotikakamthorn (KMITL, Thailand)
Surapan Airphaiboon (KMITL, Thailand)
Young Il Lee (SNUT, Korea)

■ **Secretariats**
Taworn Benjanarasuth (KMITL, Thailand)
Hyun-Chang Yang (ICASE, Korea)
Vimolluck Thianjew (KMITL, Thailand)

May 21, 2004

Paper No : ABST 386

Title : Replacement Algorithm Selection Mechanism Considering File Size for Web Cache Server

Authors : Mr. Sontisiri Tanasun, Mr. Sopechoke Pawin, Dr. Thipchaksurat SakchaiDr. Thipchaksurat Sakchai Dr. Varakulsiripunth Ruttikorn

Dear : Mr. Sontisiri Tanasun, Mr. Sopechoke Pawin, Dr. Thipchaksurat SakchaiDr. Thipchaksurat Sakchai Dr. Varakulsiripunth Ruttikorn

I would like to express my gratitude for your continued attention and support for the ICCAS2004. On behalf of the ICCAS 2004 Program Committee, it is a great pleasure to inform you that your paper described above has been accepted for presentation. You are requested to submit the paper and abstract under **"Online Submission > Submit Full Paper"** in ICCAS04 web site (http://www.kmitl.ac.th/iccas04) by **June 21, 2004.** Please follow the guideline as can be found in the 'Author's Kit' section of the ICCAS04 web site, as well as those shown below when preparing the manuscript.

※ Guidelines

1. A manuscript must be submitted as a PDF file. Make sure that the manuscript uses only English characters.

2. A manuscript is limited to 6 single-spaced two-column pages including the abstract, references, figures and tables.

Thank you for your contribution to the ICCAS2004.
We are looking forward to seeing you at The Shangri-La Hotel, Bangkok, Thailand.

Yours Sincerely,

Tawil Paungma

Tawil Paungma
ICCAS 2004 Organizing Chair
http://www.kmitl.ac.th/iccas04

ICCAS2004 Organizing Chair Tawil Paungma
Research Center for Communications and Information Technology (ReCCIT)
King Mongkut's Institute of Technology Ladkrabang, Bangkok 10520, THAILAND
TEL: +66-27392427-8, FAX: +66-27392429
E-mail: iccas04@kmitl.ac.th

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ICCAS2004 August 25-27, The Shangri-La Hotel, Bangkok, THAILAND

Replacement Algorithm Selection Mechanism Considering File Size for Web Cache Server

Tanasun Sontisiri*, Pawin Sopechoke*, Sakchai Thipchaksurat*, and Ruttikorn Varakulsiripunth**

*Department of Computer Engineering, **Department of Electronics Engineering
Faculty of Engineering and Research Center for Communications and Information Technology (ReCCIT),
King Mongkut's Institute of Technology Ladkrabang,
Chalongkrung Road, Ladkrabang, Bangkok, Thailand 10520
E-mail: {s6063003, s2061088, ktsakcha and kvruttik}@kmitl.ac.th

Abstract: This paper describes the improvement of web cache server by scoping in replacement algorithm of data which are collected from the clients. We have found that each replacement algorithm is suitable for each type of data in the web pages. Therefore, we introduce the mechanism to select the replacement algorithm depending on the size of data called the Replacement Algorithm Selection Mechanism (RASM). RASM allows the web cache server to have the suitable replacement algorithm for each type of data. As the result, the byte hit ratio of web cache server can be increased and the congestion in the network can be alleviated.

Keywords: Replacement Algorithm, %Hit, %Byte-Hit ,and Threshold

1. INTRODUCTION

At present, the growth of World Wide Web (WWW) has been rapidly increased. The model of communication is client-server, which clients would contact website servers to request data for web pages. The growth rate mentioned above causes the congestion in the network. As a result, clients receive data quite slowly. To solve this problem web cache server has been developed. In the Internet, web cache server is located between a client and a web server. The client who needs the web page has to request to the web server. The web server then transfers the web page to the client. Therefore, all data will also be buffered in the web cache server. When the cache in the web cache server is nearly full, the replacement algorithm [1] will be called to select data which will be removed from the cache. Each of the replacement algorithm models has its own characteristics and will give the value of %Hit and %Byte-Hit differently.

According to the characteristics of the replacement algorithm, we introduced threshold variable into replacement algorithm of web cache server. It uses for selecting replacement algorithm between Greedy-Dual-Size with Frequency (GDSF) [2, 4] and Least Frequency Used with Dynamic Aging (LFUDA) [3, 4] to generate priority keys of data in the cache of web cache server. And web cache server will use priority keys to decide data out from cache of web cache sever.

The remainder of this paper is organized as follows. Section 2 refers to some related works. Section 3 describes Replacement Algorithm Selection Mechanism (RASM). Section 4 presents the performance of RASM. Section 5, we describe the suitable threshold for RASM. Finally the conclusion of this paper is given in Section 6.

2. RELATED WORKS

Until now, many types of replacement algorithm have been introduced in order to increase the performance of web cache server. Each type of replacement algorithm considers variables that deal with rules to choose data that will be removed from cache of web cache server to increase the value of %Hit and %Byte-Hit. It is found that there are 2 variables from replacement algorithm which are important to consider data kept in cache of web cache server - frequency of requests and age of data that is kept in cache. Replacement algorithms which consider both variables mentioned are the Greedy-Dual-Size with Frequency (GDSF) and the Least Frequency Used with Dynamic Aging (LFUDA).

2.1 Greedy-Dual-Size with Frequency (GDSF)

The GDSF algorithm was developed by Cao and Irani based on GD-Size algorithm [5]. The GD-Size algorithm performs well, but does have one significant shortcoming: it does not take into account how many times the data was accessed in the past. For example, consider how GD-Size handles two different data of the same size. If they are requested at the same time, they are inserted into a priority queue with the same priority key value (K_i). The data f_1, which was accessed n times in the past will get the same K_i value as the data f_2 accessed for the first time. In the worst case scenario, f_1 will be replaced instead of f_2.

The GD-Size algorithm can be improved to reflect data access patterns by incorporating a frequency count F_i in the computation of K_i as denoted in Eq. (1).

$$K_i = \frac{F_i * C_i}{S_i} + L \qquad (1)$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Where C_i is the cost associated with bringing data i into the cache. S_i is the data size and L is a running age factor that starts at 0 and is updated for each replaced data f to the priority key of this data in the priority queue.: i.e., $L=K_f$. This algorithm achieves the best hit rate when $C_i=1$.

2.2 Least Frequency Used with Dynamic Aging (LFUDA)

The LFUDA algorithm was developed from LFU and LFU-Aging algorithm [6] that can achieve the highest of %Byte-Hit. These algorithms accomplish by retaining the most popular data, regardless of data size. Unfortunately LFU-Aging requires parameterization in order to perform well. Parameter less algorithms are preferable as they are generally less complex and easier to manage. Thus, they then replace the tunable aging mechanism. They call this algorithm as Least Frequency Used with Dynamic Aging (LFUDA). LFUDA calculates the priority key value K_i for data i using Eq. (2).

$$K_i = (C_i * F_i) + L \tag{2}$$

By setting C_i to 1, this equation uses only the frequency count and the inflation factor to determine the priority key of a data. The LFUDA algorithm may be useful in other caching environments where frequency is an important characteristic but where LFU has not been utilized due to cache pollution concerns.

3. REPLACEMENT ALGORITHM SELECTION MECHANISM (RASM)

According to the characteristics of GDSF replacement algorithm and LFUDA replacement algorithm [7], We notice the following:

- Data that have small size and have more of frequency access should be kept in cache of web cache server.
- Data that have large size and have access more than 2 times in a short time should be kept in cache of web cache server.

Thus, we can introduce the selection method for replacement algorithm for web cache server called Replacement Algorithm Selection Mechanism (RASM) as the following steps:

1. Determine the threshold that is used to compare with the size of data which will be kept in cache of web cache server.

2. If web cache server detects data that have equal to size or over threshold, it will call LFUDA algorithm to make priority key of data. Since large sized data are accessed only one time, they must have low priority key. But data that have large size and are accessed more than 2 times in a short time must have high priority key.

3. If web cache server detects data that have size under threshold, web cache server will call GDSF algorithm to make priority key of data. Since small and medium sized data are accessed many times, they must have high priority key. As for data that have small or medium size and have a few times of access, they must have low priority key.

In this paper, the threshold is determined to compare with the size value of data. It is utilized as the value to select replacement algorithm. Furthermore, the values of the parameters in priority key (K_i) of each type of replacement algorithm is modified to be able to be considered together as show in Eq. (3).

$$K_i = \begin{cases} \left(\frac{F_i}{S_i}\right) + L & \text{If } S_i < \text{Threshold} \\ F_i + L & \text{If } S_i \geq \text{Threshold} \end{cases} \tag{3}$$

Where S_i is the data size. L is a running age factor that starts from 0 and is updated for each replaced data f to the priority key of this data in the priority queue: i.e., $L=K_f$ and F_i is frequency of each data i that start from 1 when $S_i <$ threshold. Otherwise, it will start from 0.

4. PERFORMANCE EVALUATION

We evaluate the performance of RASM to confirm its effect by computer simulation. We compare RASM with LFUDA and GDSF replacement algorithm.

4.1 Data Collection

The access log file which used for simulation RASM was collected from web cache server of the Department of Computer Engineering, King Mongkut's Institute of Technology Ladkrabang. These log file contain information of all client requests in one week. Each entry in an access log contains information on a single request received by the web cache server. The recorded information included the client IP address; the time of the request; the client's request such as method, URL, and HTTP version; the response status from the web cache server and the origin server; the amount of header and content data transfer; and the time required for the web cache server to complete its response. In this paper, we will not consider the following URLs:

- Dynamic URLs such as cgi, php, asp, etc.
- URL that contain the special symbol such as ?, =, %, +, *, &, @, $, [ ], ( ).

For considering hit rate and byte hit rate, we focus only on the lines of requests which have words such as TCP_HIT, TCP_MEM_HIT, TCP_REFRESH_HIT, TCP_REF_FAIL_HIT and TCP_IMS_HIT.

Table 1 Summary of access log file

Access Log Duration	One Week
Total Requests	338,822 Bytes
Unique Requests	78,580 Bytes
Total Content Bytes	2.61 GB

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.2 Simulation Model and Configuration

We evaluate the performance of RASM by modelling the simulation model as shown in Fig. 1

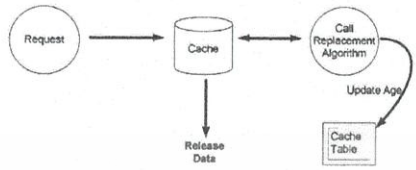


Fig. 1 Simulation Model

The simulation process can be explained as follows:

1. Define parameters of web cache server such as size of cache, type of replacement algorithm, high limit, low limit of cache, and threshold as follow:
   - The size of cache is set to 128 MB.
   - The low limit and high limit are equal to 90% and 95%, respectively.
   - Thresholds are calculated from the mode and mean of the data which have the value of 933 Bytes and 32 Kbytes, respectively.
2. Send requests of test data to web cache server one by one.
3. Check files in the cache. If a request file is a hit, its priority key will be updated and wait for a next request. Otherwise, the process will check the size of cache. If it is higher than a high limit, the replacement algorithm will start to remove files from a cache until the size of cache decreases equal a low limit. Then the replacement algorithm will stop and the new file is kept in the cache and waits for a next request.

In the real system, when we measure value of %Hit and %Byte-Hit, its value will increase more than simulation because the replacement algorithm can be divided into two processes. The first one is Memory Cache process and the second one is Disk Cache process. In the memory cache process, web cache server will check the using areas in memory cache. If they are higher than high limit of memory cache, a web cache server will use a replacement algorithm to move files from a memory cache to a disk cache until using areas come to a low limit of memory cache. In the disk cache process, web cache server also checks using areas like memory cache process. If the using areas are higher than the high limit of disk cache, a web cache server must remove files out until using areas come to a low limit of disk cache.

4.3 Simulation Results

This section provides the simulation results of the web cache server replacement algorithm. The threshold is equal to 933 Bytes and 32 Kbytes, respectively. We compare the performance of RASM with GDSF and LFUDA, respectively.

Fig.2 (a) and (b) show %Hit and %Byte-Hit for threshold value equal to 933 Bytes.. In Fig.2 (a) %Hit of RASM drop and close to %Hit of LFUDA algorithm. Fig.2 (b) %Byte-Hit of RASM get %Byte-Hit between %Byte-Hit of GDSF and LFUDA algorithm. We find that this threshold give us low performance because %Hit of RASM drops from that of GDSF algorithm a lot. The problem occurs when cache of web cache server keeps large data size which is larger than 933 Bytes. So, it has little space to keep data. Web cache server must call replacement algorithm to release data from cache many times. Consequently, %Hit will have lower value.

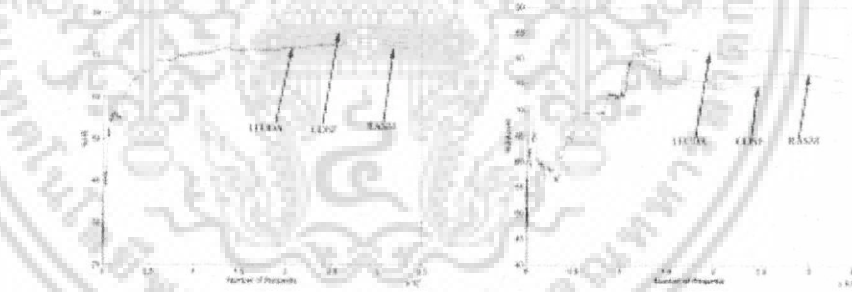


Fig.2 (a) %Hit versus number of requests for threshold 933 Bytes (b) %Byte-Hit versus number of requests for threshold 933 Bytes

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

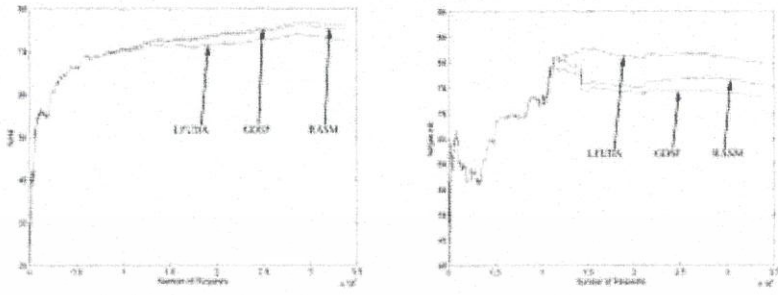


Fig.3 (a) %Hit versus number of requests for threshold 32 Kbytes (b) %Byte-Hit versus number of requests for threshold 32 Kbytes

Next, we change threshold to 32 Kbytes. The results show in Fig.3 (a) and (b), respectively. In Fig.3 (a) %Hit of RASM is close to %Hit of GDSF algorithm because when the threshold is increased, data that have larger size than the threshold are kept in cache of web cache server less than the threshold of 933 Bytes. Therefore, the cache of web cache server has space to keep data more than threshold of 933 Bytes. In Fig.3 (b) %Byte-Hit of RASM is shown between %Byte-Hit of GDSF and LFUDA algorithm. However, if we compare %Byte-Hit with %Byte-Hit of RASM for threshold equal to 933 Bytes, %Byte-Hit of RASM for threshold equal to 32 Kbytes is a little higher .so it means that threshold of 32 Kbytes has better performance than threshold of 933 Bytes.

5. DISCUSSION

Since %Hit and %Byte-Hit of RASM cannot be proven into mathematical equation, they depend on the environment of requests, the size of data, and the frequency of requests in each data. In order to consider the suitable threshold, we do a simulating by changing the threshold from 10 Kbytes to 5 Mbytes.

We can explained as follows:

1. %Hit of RASM must decrease from GDSF algorithm at the lowest degree denoted as $\overline{\Delta H_i(gdsf)}\min$ and must increase from LFUDA algorithm highest denoted as $\overline{\Delta H_i(lfuda)}\max$.

2. %Byte-Hit of RASM must decrease from LFUDA algorithm lowest denoted as $\overline{\Delta H_b(lfuda)}\min$ and must increase from GDSF algorithm highest denoted as $\overline{\Delta H_b(gdsf)}\max$.

Therefore, we derive the following equations:

$$\overline{\Delta H_i(gdsf)} = \frac{\sum_{n=1}^{N} \frac{\%Hit_{gdsf} - \%Hit_{modified}}{\%Hit_{gdsf}}}{N}, \quad (4)$$

$$\overline{\Delta H_i(lfuda)} = \frac{\sum_{n=1}^{N} \frac{\%Hit_{modified} - \%Hit_{lfuda}}{\%Hit_{lfuda}}}{N}, \quad (5)$$

$$\overline{\Delta H_b(gdsf)} = \frac{\sum_{n=1}^{N} \frac{\%Byte_{modified} - \%Byte_{gdsf}}{\%Byte_{gdsf}}}{N}, \quad (6)$$

$$\overline{\Delta H_b(lfuda)} = \frac{\sum_{n=1}^{N} \frac{\%Byte_{lfuda} - \%Byte_{modified}}{\%Byte_{lfuda}}}{N}. \quad (7)$$

Where $\%Hit_{gdsf}$ is the %Hit of GDSF algorithm. $\%Hit_{lfuda}$ is the %Hit of LFUDA algorithm. $\%Byte_{gdsf}$ is the %Byte-Hit of GDSF algorithm. $\%Byte_{lfuda}$ is the %Byte-Hit of LFUDA algorithm and N is the number of requests.

The suitable threshold can be obtained according to the following conditions.

1. For GDSF algorithm, we calculate the different between $\overline{\Delta H_b(gdsf)}$ and $\overline{\Delta H_i(gdsf)}$ denoted as H_{gdsf}. Therefore, we get.

$$H_{gdsf} = \overline{\Delta H_b(gdsf)} - \overline{\Delta H_i(gdsf)} \quad (8)$$

2. For LFUDA algorithm, we calculate the different between $\overline{\Delta H_b(lfuda)}$ and $\overline{\Delta H_i(lfuda)}$ denoted as H_{lfuda}. So, we get

$$H_{lfuda} = \overline{\Delta H_b(lfuda)} - \overline{\Delta H_i(lfuda)} \quad (9)$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

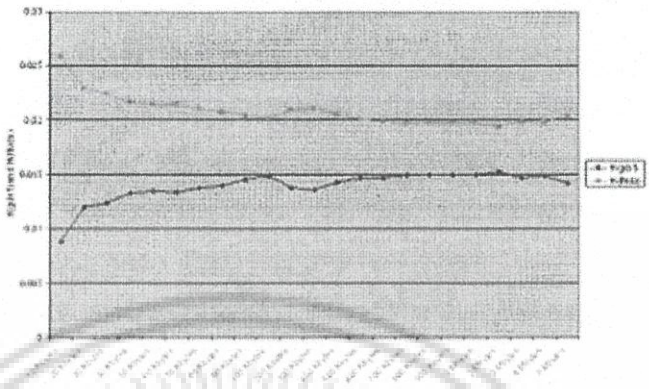


Fig. 4 H_{gdsf} and H_{lfuda} versus various value of threshold

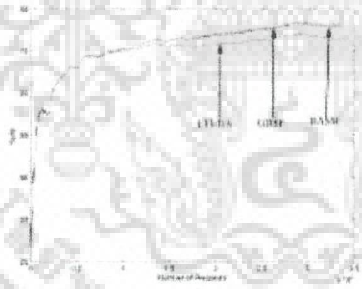
Hence we use both conditions to find suitable threshold value of data from web cache server of Department of Computer Engineering, King Mongkut's Institute of Technology Ladkrabang as shown in Fig. 4. and we get the suitable threshold equal to 2 Mbytes.

The simulation result for threshold of 2 Mbytes are shown in Fig. 5 (a) and Fig. 5 (b), respectively. In Fig. 5 (b), we can see that %Byte-Hit of RASM give the result nearly %Byte-Hit of RASM in Fig. 2 (b) and 3 (b). But in Fig. 5 (a), %Hit of our algorithm give the result better than graph %Hit of RASM in Fig. 2 (a) and Fig. 3 (a), respectively. Therefore, it may be suggested that our method for selecting may be applied for RASM.

6. CONCLUSION

In this paper, we have introduced the mechanism for selecting the suitable replacement algorithm based on the size of files contained in the web page called replacement algorithm selection mechanism (RASM). We have showed that RASM can provide the improvement over the GDSF replacement algorithm in the form of percentage of byte hit ratio and LFUDA replacement algorithm in the form of percentage of hit ratio.

For the future work, we will consider the dynamic threshold that can be changed according to the situation in the network in order to achieve higher byte hit and hit ratio of web cache server.



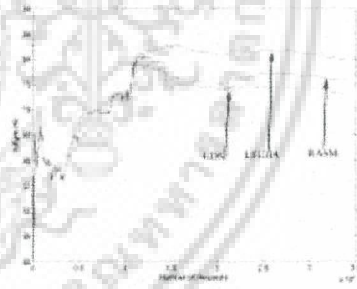


Fig.5 (a) %Hit versus number of requests for threshold 2 Mbytes (b) %Byte-Hit versus number of requests for threshold 2 Mbytes

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

REFERENCES

[1] J. Dilley, M. Arlitt and S. Perret, "Enhancement and Validation of Squid's Cache Replacement Policy." Technical Report HPL-1999-69, Hewlett-Packard Laboratories, May 1999.

[2] L. Cherkasova, "Improving WWW proxies Performance with Greedy-Dual-Size-Frequency Caching Policy." Technical Report HPL-1998-69R1, Hewlett-Packard Laboratories, November 1998.

[3] M. Arlitt, L. Cherkasova, J. Dilley, R. Friedrich and T. Jin, "Evaluating Content Management Techniques for Web Proxy Caches." Technical Report HPL-98-173, Hewlett-Packard Laboratories, April 1999.

[4] S. Jin and A. Bestavros, "Popularity-aware greedy dual-size Web proxy caching algorithms." Proc. 20th International Conference on Distributed Computing Systems, pp. 254-261, April 2000.

[5] P. Cao and S. Irani, "Cost Aware WWW Proxy Caching Algorithms." Proc. USENIX Symposium on Internet Technology and Systems (USITS), pp. 193-206, December 1997.

[6] S. Podlipnig and L. Boszormenyi, "A Survey of Web Cache Replacement Strategies." ACM Computing Surveys, vol. 35, no. 4, pp. 374-398, December 2003.

[7] P. Sopechoke and B. Piyatamrong, "Improvement Web Cache Server with Hierarchical Replacement Algorithm." Proc. 7th National Computer Science and Engineering Conference 2003, pp. 140-144, October 2003.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บันทึกข้อความ

ส่วนราชการ งานส่งเสริมการวิจัย กองบริการการศึกษา สำนักงานอธิการบดี ชั้น 4 และ ชั้น 8 โทร 3780

ที่ ศธ. 0524.01(5)/พิเศษ (175) **วันที่** 30 มิถุนายน 2547

เรื่อง ตอบรับตีพิมพ์ต้นฉบับบทความลงวารสารพระจอมเกล้าลาดกระบัง

เรียน คุณภาวิน สพโชค คุณศักดิ์ชัย ทิพย์จักษุรัตน์ และคุณบรรจง ปิยธำรง

เนื่องจากกองบรรณาธิการวารสารพระจอมเกล้าลาดกระบัง ได้รับต้นฉบับ

[✓] บทความวิจัย

[ ] บทความวิชาการ

เรื่อง อัลกอริธึมการแทนที่ข้อมูลสำหรับเว็บแคชเซิร์ฟเวอร์ (Replacement Algorithm for Web Cache Server)

ผลการพิจารณาจากผู้อ่านบทความเป็นดังนี้

[ ] ไม่ผ่านการพิจารณา

[ ] ผ่านการพิจารณา โดยไม่มีการแก้ไข

[✓] ผ่านการพิจารณา โดยให้แก้ไขเพิ่มเติม (ตามเอกสารแนบ)

ทั้งนี้กรุณาส่งต้นฉบับบทความที่แก้ไขพร้อมแผ่นบันทึกข้อมูล (Disk) หรือ ซีดีรอม (CD-ROM) เพื่อลงตีพิมพ์ในวารสารพระจอมเกล้าลาดกระบัง ฉบับที่ 2 ปีที่ 12 เดือนสิงหาคม พ.ศ. 2547 ภายใน วันที่ 16 เดือนกรกฎาคม พ.ศ. 2547

ในการนี้กองบรรณาธิการวารสารพระจอมเกล้าลาดกระบัง ขอขอบพระคุณท่านเป็นอย่างยิ่ง ที่ได้ให้ความสนใจและสนับสนุนวารสารพระจอมเกล้าลาดกระบัง และหวังว่าคงได้รับการสนับสนุนจากท่านอีกในโอกาสต่อไป

(รองศาสตราจารย์ ดร. ศักดา ไตรศักดิ์)

บรรณาธิการ

วารสารพระจอมเกล้าลาดกระบัง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

อัลกอริธึมการแทนที่ข้อมูลสำหรับเว็บแคชเซิร์ฟเวอร์

Replacement Algorithm for Web Cache Server

ภาวิน สพโชค นักศึกษาปริญญาโท
ศักดิ์ชัย ทีพอัจฉริยนันท์ อาจารย์
บรรจง ปิยธำรง รองศาสตราจารย์
ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

บทคัดย่อ

งานวิจัยนี้นำเสนอการปรับแต่งเว็บแคชเซิร์ฟเวอร์ โดยจะพิจารณาการพัฒนาปรับปรุงในส่วนของอัลกอริธึมการแทนที่ข้อมูล (Replacement Algorithm) [1] โดยค่าที่นำมาวิเคราะห์คือปริมาณการร้องขอข้อมูลจากไคลเอนต์ ขนาดข้อมูลต่อโอน ประเภทของข้อมูลที่ร้องขอ เนื่องจากค่าเหล่านี้จะเปลี่ยนแปลงไปตามกลุ่มผู้ใช้งาน ซึ่งพบว่าอัลกอริธึมการแทนที่ข้อมูลแต่ละชนิด จะเหมาะสมกับชนิดของไฟล์ที่มีขนาดแตกต่างกัน จากคุณสมบัติดังกล่าว ผู้วิจัยได้พัฒนาและปรับปรุงอัลกอริธึมการแทนที่ข้อมูลในเว็บแคชเซิร์ฟเวอร์ ให้เหมาะสมกับขนาดที่แตกต่างกันของข้อมูลที่มีการร้องขอ เพื่อปรับปรุงค่าของ Byte-Hit ratio ของเว็บแคชเซิร์ฟเวอร์ให้สูงขึ้น

Abstract

This research describes the improvement of web cache server by scoping in replacement algorithm of data which are collected from the clients. Data that we use for analysis are the request from all clients, size of object and type of request which all of data will be changed depending on users. However, we found that each replacement algorithm was suitable for each type of data. Therefore, we have developed web cache server to have the suitable replacement algorithm according to the size of data that clients have request. As the result, the value of Byte-Hit ratio of web cache server can be increased.

คำสำคัญ: เว็บแคชเซิร์ฟเวอร์, อัลกอริธึมการแทนที่ข้อมูล, เปอร์เซ็นการพบข้อมูล, เปอร์เซ็นการพบข้อมูลแบบไบต์
Keywords: web cache server, Replacement Algorithm, %Hit, %Byte-Hit

1. บทนำ

ในปัจจุบันอัตราการเติบโตของการใช้เวิลด์ไวด์เว็บได้เพิ่มขึ้นอย่างรวดเร็ว ซึ่งรูปแบบของการติดต่อจะเป็นแบบไคลเอนต์-เซิร์ฟเวอร์ โดยไคลเอนต์จะติดต่อไปยังเซิร์ฟเวอร์ต้นทางของเว็บไซต์เพื่อขอข้อมูลเว็บเพจ ข้อมูลเว็บเพจที่ได้รับ จะประกอบด้วยข้อมูลหลายชนิดทั้งส่วนของ รูปภาพ ตัวอักษร ไฟล์วิดีโอ ภาพเคลื่อนไหว ซึ่งข้อมูลเหล่านี้จะมีลักษณะที่แตกต่างกันออกไป และจากการเติบโตของการใช้งานนี้ทำให้เกิดปัญหาการคับคั่งของข้อมูลบนเครือข่าย ผลที่เกิดขึ้นคือ การล่าช้าของการได้รับข้อมูลที่ต้องการ

ดังนั้นเพื่อลดปัญหาดังกล่าวเว็บแคชเซิร์ฟเวอร์จึงมีการพัฒนาขึ้น หลักการทำงานของเว็บแคชเซิร์ฟเวอร์จะแสดงได้ดังรูปที่ 1 คือ เว็บแคชเซิร์ฟเวอร์จะรอรับการร้องขอข้อมูลจากไคลเอนต์ ถ้าเว็บแคชเซิร์ฟเวอร์ค้นพบข้อมูลที่ร้องขอในแคช (Cache) ซึ่งเป็นหน่วยความจำที่เว็บแคชเซิร์ฟเวอร์ใช้สำหรับเก็บข้อมูลที่ได้รับจากเซิร์ฟเวอร์ต้นทางของเว็บไซต์

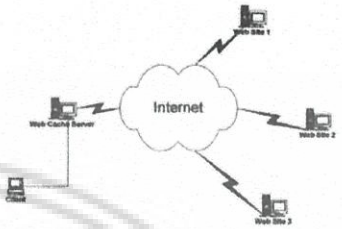
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

และยังไม่หมดอายุ (เวลาที่กำหนดไว้ เพื่อเก็บข้อมูลไว้ในแคชของเว็บแคชเซิร์ฟเวอร์) จะทำการส่งข้อมูลที่เก็บอยู่ในแคชไปให้ไคลเอนต์ แต่ถ้าไม่พบข้อมูลหรือข้อมูลที่พบหมดอายุไปแล้ว จะทำการร้องขอไปยังเซิร์ฟเวอร์ต้นทางของเว็บไซต์ เพื่อให้เซิร์ฟเวอร์ต้นทางของเว็บไซต์ส่งข้อมูลใหม่มาให้เว็บแคชเซิร์ฟเวอร์ และเว็บแคชเซิร์ฟเวอร์จะทำการส่งข้อมูลกลับไปให้ไคลเอนต์ที่ร้องขอ พร้อมกับทำการสำเนาเก็บเอาไว้ในแคชด้วยเพื่อให้บริการไคลเอนต์ ในกรณีที่มีการร้องขอข้อมูลชุดเดิมอีกในโอกาสต่อไป ในกรณีที่มีการค้นพบข้อมูลภายในแคชของเว็บแคชเซิร์ฟเวอร์จะเรียกว่า การพบข้อมูลหรือ Hit และ ในกรณีที่ไม่พบข้อมูลภายในแคชของเว็บแคชเซิร์ฟเวอร์หรือข้อมูลที่พบภายในแคชของเว็บแคชเซิร์ฟเวอร์หมดอายุไปแล้วจะเรียกว่า การไม่พบข้อมูล หรือ Miss

การพัฒนาความสามารถของเว็บแคชเซิร์ฟเวอร์จะสามารถพัฒนาได้หลายส่วนของระบบแคช โดยจุดมุ่งหมายหลักของการพัฒนาคือ เพื่อเพิ่มสมรรถนะด้านความเร็ว ความเชื่อถือได้ของข้อมูลที่ส่งให้กับไคลเอนต์และ จำนวนข้อมูลที่ถูกค้นพบภายในแคชของเว็บแคชเซิร์ฟเวอร์

สำหรับการวัดประสิทธิภาพการทำงานของเว็บแคชเซิร์ฟเวอร์มักจะนิยมวัดจากอัตราการพบ (Hit Ratio), อัตราการพบแบบไบต์ (Byte-Hit Ratio) และค่าเวลาเฉลี่ยของข้อมูลตอบกลับ (Mean Response Time) ซึ่งค่าทั้ง 3 จะขึ้นกับหน่วยประมวลผลกลาง ความจุของหน่วยความจำ, ขนาดของดิสก์, การใช้งานของดิสก์และ ซอฟท์แวร์ที่ทำหน้าที่เป็นเว็บแคชเซิร์ฟเวอร์ที่แตกต่างกันไป

จากหลักการทำงานของเว็บแคชเซิร์ฟเวอร์ที่กล่าวมาในข้างต้น เมื่อเว็บแคชเซิร์ฟเวอร์จะทำการเก็บข้อมูลที่ได้รับจากเซิร์ฟเวอร์ต้นทางของเว็บไซต์ภายในแคชของเว็บแคชเซิร์ฟเวอร์ เว็บแคชเซิร์ฟเวอร์จะทำการแบ่งข้อมูลในแต่ละเว็บเพื่อออกเป็นไฟล์ต่าง ๆ ที่เป็นส่วนประกอบของแต่ละเว็บเพื่อ เช่น ไฟล์ Html, ไฟล์ Text, ไฟล์รูปภาพ และ ไฟล์เสียง โดยแต่ละไฟล์จะถูกเก็บไว้ในโฟลเดอร์ที่ทำหน้าที่เป็นแคชของเว็บแคชเซิร์ฟเวอร์พร้อมกับค่าขนาดของไฟล์และเวลาที่ไฟล์ถูกจัดเก็บ ดังนั้นข้อมูลที่อยู่ภายในแคชของเว็บแคชเซิร์ฟเวอร์จะเป็นการระบุถึงชื่อไฟล์, ค่า URL (Uniform Resource Locator) ของไฟล์, ขนาดของไฟล์ และเวลาที่ไฟล์ถูกจัดเก็บลงไปในแคชของเว็บแคชเซิร์ฟเวอร์



รูปที่ 1 โมเดล ไคลเอนต์-เซิร์ฟเวอร์ ของเว็บแคชเซิร์ฟเวอร์และไคลเอนต์

เนื่องจากเว็บแคชเซิร์ฟเวอร์มีขนาดของแคชที่จำกัด และเนื้อที่แคชของเว็บแคชเซิร์ฟเวอร์จะมีค่าลดลง เมื่อมีการเก็บข้อมูลที่นำมาจากเซิร์ฟเวอร์ต้นทางของเว็บไซต์ทุกครั้งหลังจากที่ได้ส่งข้อมูลไปให้ไคลเอนต์ เมื่อข้อมูลที่เก็บอยู่ในแคชของเว็บแคชเซิร์ฟเวอร์มีค่ามากขึ้นจนถึงค่ากำหนดขนาดของแคช เว็บแคชเซิร์ฟเวอร์จะเรียกใช้ฟังก์ชั่นที่ทำหน้าที่เลือกข้อมูลที่เก็บอยู่ในแคชและนำข้อมูลที่เลือกไว้ออกจากแคชของเว็บแคชเซิร์ฟเวอร์จนกระทั่งขนาดของแคชมีค่าเท่ากับค่ากำหนดที่เว็บแคชเซิร์ฟเวอร์ตั้งไว้ ก็จะหยุดเรียกใช้งานฟังก์ชั่นและเริ่มทำการเก็บข้อมูลลงไปในแคชอีกครั้ง โดยฟังก์ชั่นที่เว็บแคชเซิร์ฟเวอร์เรียกใช้เรียกว่า อัลกอริธึมการแทนที่ข้อมูล (Replacement Algorithm)

2. งานวิจัยที่เกี่ยวข้อง

ในงานวิจัยนี้ ผู้วิจัยได้พิจารณาอัลกอริธึมการแทนที่ข้อมูล 3 ชนิด ดังต่อไปนี้

1) Least-Recently-Used (LRU)

2) Greedy-Dual-Size-Frequency (GDSF) [2, 3]

3) Least-Frequency-Used with Dynamic Aging (LFUDA) [2, 4]

2.1 Least-Recently-Used (LRU)

LRU เป็นอัลกอริธึมการแทนที่ข้อมูลที่พิจารณาและ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เลือกข้อมูลออกจากแคชของเว็บแคชเซิร์ฟเวอร์ โดยพิจารณาจากอายุของข้อมูลที่อยู่ในแคชของเว็บแคชเซิร์ฟเวอร์ ซึ่งอายุของข้อมูลจะเริ่มนับตั้งแต่วันที่ข้อมูลถูกเก็บลงในแคชของเว็บแคชเซิร์ฟเวอร์หรือเมื่อมีการเรียกใช้งานจากไคลเอนต์ครั้งสุดท้ายไปจนถึงวันที่ปัจจุบัน

LRU จะเลือกข้อมูลที่อายุมีค่ามากที่สุด ซึ่งหมายถึงข้อมูลที่ถูกเก็บไว้ในแคชของเว็บแคชเซิร์ฟเวอร์หรือมีการร้องขอข้อมูลจากไคลเอนต์เป็นระยะเวลานานเมื่อเปรียบเทียบกับวันที่ปัจจุบัน ออกไปจากแคชของเว็บแคชเซิร์ฟเวอร์ เนื่องจากข้อมูลที่มีค่าอายุค่ามากที่สุด

2.2 Greedy-Dual-Size-Frequency (GDSF)

GDSF ถูกพัฒนามาจาก อัลกอริธึมการแทนที่ข้อมูลแบบ GD-Size โดยพิจารณาการเลือกข้อมูลออกโดยดูจากข้อมูลที่มีการใช้งานต่ำที่สุด ซึ่งจะมีตัวแปรที่ใช้สำหรับการเลือกข้อมูลออก (K_i) ดังสมการที่ 1

$$K_i = \frac{C_i}{S_i} + L \qquad \text{.....(1)}$$

โดย

C_i คือ ค่าคงที่ซึ่งนำมารวมเมื่อนำข้อมูลเข้ามาในแคช

S_i คือ ขนาดของข้อมูล มีขนาดเป็นไบต์

L คือ องค์ประกอบอายุของข้อมูลที่เริ่มตั้งแต่ 0 และเพิ่มขึ้นเรื่อยๆ เมื่อมีการแทนที่ข้อมูลเข้าไปในแคช

ในกรณีของ อัลกอริธึมการแทนที่แบบ GDSF จะมีการเพิ่มในส่วนของความถี่จำนวนครั้ง (F_i) ในการเรียกใช้ข้อมูลตัวนั้นเข้าไปด้วยดังสมการที่ 2

$$K_i = F_i * \left(\frac{C_i}{S_i}\right) + L \qquad \text{.....(2)}$$

ซึ่งในการทำงานจะให้ค่าของ C_i มีค่าเท่ากับ 1 ผู้วิจัยจะเรียกรูปแบบนี้ว่า GDSF-Hits

2.3 Least-Frequency-Used with Dynamic Aging (LFUDA)

LFUDA เป็นอัลกอริธึมการแทนที่ข้อมูลที่ถูกร้องขอโดยพิจารณาจากจำนวนครั้งของการเรียกใช้ข้อมูลนั้น แต่จะเพิ่มในส่วนอายุของข้อมูลที่เก็บเอาไว้ในแคช (L) ดังสมการที่ 3

$$K_i = (C_i * F_i) + L \qquad \text{.....(3)}$$

2.4 Squid เว็บแคชเซิร์ฟเวอร์

Squid [5] คือ Internet Proxy Caching application ซึ่งมีจุดเริ่มต้นมาจากการพัฒนาโปรแกรมของ Harvest Project ซึ่งไม่ได้มีจุดมุ่งหมายหลักในการพัฒนาแคช โดยได้รับเงินทุนสนับสนุนงานวิจัยจาก National Laboratory of Network Research (NLANR) Squid เป็นแอพพลิเคชั่นที่เผยแพร่ Source Code โดยไม่คิดค่าใช้จ่าย จึงมีผู้นำ Squid ไปพัฒนาเพิ่มความสามารถอื่น ๆ และแก้ไขข้อผิดพลาดที่เกิดขึ้นอย่างแพร่หลายซึ่งปัจจุบันได้พัฒนาถึงเวอร์ชันที่ 2.5

3. รูปแบบของข้อมูลที่นำมาวิเคราะห์

ข้อมูลที่นำมาวิเคราะห์ในงานวิจัยนี้ ผู้วิจัยกำหนดรูปแบบของสมมติฐานดังต่อไปนี้

- ข้อมูลเป็นข้อมูลที่ร้องขอผ่านทางโปรโตคอล HTTP เท่านั้น จะไม่พิจารณาบริการผ่านโปรโตคอลอื่น เช่น FTP, SSH และ SMTP เป็นต้น
- ข้อมูลจะต้องผ่านการร้องขอใช้งานด้วยความถูกต้อง นอกจากนั้นจะไม่นับ URL ที่มีลักษณะแบบไดนามิกส์ เช่น cgi, php, asp และ URL ที่มีสัญลักษณ์พิเศษประกอบอยู่เช่น ?, =, %, +, *, &, @, $, [ ], ( ) เนื่องจาก URL ที่มีลักษณะเช่นนี้เป็นURL ที่ไม่ได้ระบุถึงข้อมูลเว็บเพื่อที่สามารถนำมาใช้วิเคราะห์ได้
- สำหรับการวิเคราะห์ ผู้วิจัยจะพิจารณาเปอร์เซ็นต์ของอัตราการพบข้อมูลและ อัตราการพบข้อมูลแบบไบต์ จะพิจารณาเฉพาะบรรทัดในไฟล์ Log ของเว็บแคชเซิร์ฟเวอร์ ที่มีคำว่า TCP_HIT, TCP_MEM_HIT, TCP_REFRESH_HIT, TCP_REF_FAIL_HIT และ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

TCP_IMS_HIT เนื่องจากบรรทัดในไฟล์ Log ที่มีคำที่กล่าวมา จะเป็นบรรทัดที่แสดงให้ทราบถึงการพบข้อมูลที่ไคลเอนต์มีการร้องขอภายในแคชของเว็บแคชเซิร์ฟเวอร์ และบรรทัดของไฟล์ Log ที่มีคำอยู่นอกเหนือจากที่กล่าวมาถือว่าไม่พบข้อมูลในแคชของเว็บแคชเซิร์ฟเวอร์

4. ขั้นตอนการปรับแต่งอัลกอริธึมการแทนที่ข้อมูลของ Squid เว็บแคชเซิร์ฟเวอร์

4.1 ลักษณะของระบบที่ทำการทดสอบ

ข้อมูลที่นำมาวิเคราะห์นำมาจากการจัดเก็บข้อมูลจากการจำลองการร้องขอไปยังเว็บแคชเซิร์ฟเวอร์ของภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ซึ่งผู้วิจัยได้ทำการสำรวจข้อมูลการร้องขอ ซึ่งถูกเก็บไว้ในไฟล์ Log เป็นเวลาหนึ่งสัปดาห์ จากการสำรวจผู้วิจัยพบว่า รูปแบบของการร้องขอข้อมูลที่เกิดขึ้นในแต่ละวันในหนึ่งสัปดาห์ มีรูปแบบที่เหมือนกัน ซึ่งเราสามารถใช้ข้อมูลใน 1 วัน มาทำการวิเคราะห์ได้ และผู้วิจัยได้ใช้โปรแกรม Cfmc [6] ทำการสร้างการร้องขอไปยังเว็บแคชเซิร์ฟเวอร์ที่กำหนด ระบบปฏิบัติการคือ Linux ของ Debian เวอร์ชัน 3.0 [7] และใช้โปรแกรม Squid เวอร์ชัน 2.5 Stable3 เพื่อทำหน้าที่เป็นเว็บแคชเซิร์ฟเวอร์และกำหนดเนื้อที่ขนาด 128 เมกกะไบต์สำหรับการเก็บข้อมูลของแคช ระบบเครือข่ายต่อแบบอีเทอร์เนต และมีช่องทางการติดต่อภายในประเทศและต่างประเทศจำนวน 3 ช่องทาง

จากข้อมูลที่ได้ทำการเก็บรวบรวมจากวิธีข้างต้น ถูกนำมาวิเคราะห์เพื่อหาพารามิเตอร์ที่เกี่ยวข้องกับการปรับแต่งแคชเซิร์ฟเวอร์ [8] โดยสามารถวิเคราะห์เป็นส่วน ๆ ดังนี้

- %Hit คืออัตราส่วนระหว่าง จำนวนการร้องขอที่ถือว่าเป็นการ Hit ต่อ จำนวนการร้องขอทั้งหมด คูณด้วย 100
- %Byte-Hit คืออัตราส่วนระหว่าง ขนาดข้อมูลของจำนวนการร้องขอที่ถือว่าเป็นการ Hit ต่อ ขนาดข้อมูลของจำนวนการร้องขอทั้งหมด คูณด้วย 100

ข้อมูลที่นำมาทดสอบการปรับแต่งอัลกอริธึมการแทนที่ข้อมูลของ Squid เว็บแคชเซิร์ฟเวอร์ ประกอบด้วยข้อมูลที่แบ่งตามชนิดของไฟล์ ซึ่งข้อมูลที่นำมาทดสอบจะเป็นข้อมูลของเว็บเพจ ต่าง ๆ ที่ประกอบไปด้วยไฟล์รูปภาพ ไฟล์ HTML และไฟล์ประเภทอื่น เมื่อพิจารณาจะพบว่าข้อมูลรูปภาพจะมีจำนวนมากที่สุด ดังนั้นจึงแบ่งข้อมูลการแสดงตามชนิดของไฟล์และได้ดังตารางที่ 1 และ เมื่อพิจารณาถึงช่วงขนาดของไฟล์พบว่า ไฟล์รูปภาพที่เป็นชนิดไฟล์ที่มีมากที่สุด โดยส่วนใหญ่จะเป็นไฟล์ที่มีขนาดเล็กมีค่าไม่เกิน 10 Kbytes ดังแสดงในตารางที่ 2

ตารางที่ 1 แสดงเปอร์เซ็นของข้อมูลที่ร้องขอแบ่งตามชนิดของไฟล์

ชนิดของไฟล์	ค่าเปอร์เซ็นต์
ไฟล์รูปภาพนามสกุล GIF	55.8 %
ไฟล์รูปภาพนามสกุล JPEG	18.1 %
ไฟล์นามสกุล HTML	7.5 %
อื่น ๆ	18.6 %

ตารางที่ 2 แสดงเปอร์เซ็นของข้อมูลที่ร้องขอแบ่งตามช่วงขนาดของไฟล์

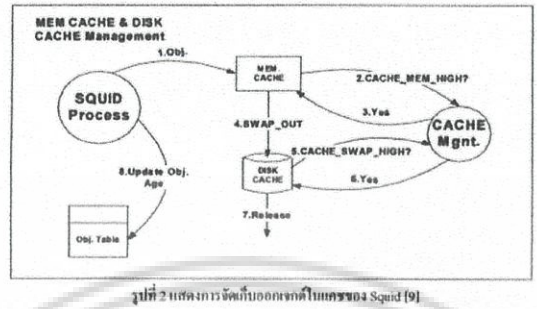
ช่วงขนาดของไฟล์	ค่าเปอร์เซ็นต์
100 – 999 Bytes	37.4 %
1000 – 9999 Bytes	51.4 %
10000 – 99999 Bytes	10.9 %
อื่น ๆ	0.3 %

4.2 การปรับแต่งอัลกอริธึมการแทนที่ข้อมูล

ในงานวิจัยนี้ มุ่งศึกษาถึงการเพิ่มค่าของ %Byte-Hit ของเว็บแคชเซิร์ฟเวอร์ ซึ่งค่าของ %Byte-Hit จะมีค่าสูงเมื่อมีการพบไฟล์ขนาดใหญ่ภายในแคชของเว็บแคชเซิร์ฟเวอร์เป็นจำนวนมาก เมื่อพิจารณาถึงคุณสมบัติของอัลกอริธึมการแทนที่ข้อมูลแต่ละชนิด [10] พบว่า

- อัลกอริธึมการแทนที่ข้อมูลแบบ GDSF จะมีตัวแปรที่มีผลกับค่าการเลือกข้อมูลออกประกอบด้วย ขนาดของไฟล์ จำนวนครั้งของการเรียกใช้ข้อมูลในแคช และเวลาของข้อมูลที่ถูกเก็บไว้ในแคชดังสมการที่ 2 ซึ่งพบว่าในกรณีของไฟล์ที่มีเวลาของข้อมูลที่เก็บไว้ในแคชเท่ากัน ขนาดของไฟล์จะเป็นตัวแปรสำคัญของค่าการเลือกข้อมูลออกไปจากแคช เพราะ เมื่อไฟล์มีขนาดใหญ่ค่าการเลือกข้อมูลออกจะมีค่าน้อยกว่าเมื่อเทียบกับไฟล์ที่มีขนาดเล็ก ดังนั้นไฟล์ที่มีขนาดใหญ่จึงถูกนำออกไปจากแคชของเว็บ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2 แสดงการจัดเก็บออกจากดิในแคชของ Squid [9]

แคชเซิร์ฟเวอร์เป็นลำดับแรก

- อัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA จะมีตัวแปรที่มีผลกับค่าการเลือกข้อมูลออกประกอบด้วย จำนวนครั้งของการเรียกใช้ข้อมูลในแคช และเวลาของข้อมูลที่ถูกเก็บไว้ในแคช ดังสมการที่ 3 ซึ่งพบว่าในกรณีของไฟล์ที่มีเวลาของข้อมูลที่เก็บไว้ในแคชเท่ากัน จำนวนครั้งของการเรียกใช้ข้อมูลในแคชจะเป็นตัวแปรสำคัญของการเลือกข้อมูลออกไปจากแคชเพราะ ข้อมูลที่ถูกเก็บไว้ในแคชและมีการเรียกใช้งานบ่อย จะมีค่าการเลือกข้อมูลออกสูงกว่าข้อมูลที่ถูกเก็บไว้ในแคชและมีการเรียกใช้งานน้อย ดังนั้นข้อมูลที่ถูกเรียกใช้งานน้อยจึงถูกนำออกไปจากแคชของเว็บแคชเซิร์ฟเวอร์เป็นลำดับแรก

จากคุณสมบัติของอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF และแบบ LFUDA จึงได้ทำการปรับปรุงอัลกอริธึมการแทนที่ข้อมูล ดังนี้

- กำหนดค่า Threshold เพื่อทำหน้าที่เป็นค่าสำหรับเปรียบเทียบกับขนาดของไฟล์ที่เว็บแคชเซิร์ฟเวอร์ได้รับมาจากเซิร์ฟเวอร์ต้นทางของเว็บไซต์ก่อนที่จะนำมาเก็บไว้ในแคชของเว็บแคชเซิร์ฟเวอร์
- ไฟล์ที่มีขนาดมากกว่าหรือเท่ากับค่า Threshold ที่กำหนดจะพิจารณาการนำข้อมูลออกจากแคชโดยใช้อัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA เนื่องจากไฟล์ที่มีขนาดใหญ่และมีการเรียกใช้งานบ่อย ๆ ควรจะถูกนำมาเก็บไว้ในแคชและไฟล์ขนาดใหญ่ที่ไม่มีการเรียกใช้งานบ่อย ๆ ก็ควรนำออกไปจากแคชของเว็บแคชเซิร์ฟเวอร์
- ไฟล์ที่มีขนาดน้อยกว่าค่า Threshold ที่กำหนดจะถูกพิจารณาการนำข้อมูลออกจากแคชโดยการใช้อัลกอริธึมการแทนที่ข้อมูลแบบ GDSF เนื่องจากการแทนที่ข้อมูลแบบ GDSF จะพิจารณาถึงไฟล์ที่มีขนาดเล็กและมีการเรียกใช้งานบ่อย ๆ เป็นหลัก จากตารางที่ 2 พบว่าข้อมูลส่วนใหญ่ของข้อมูลที่นำมาทดสอบจะมีขนาดอยู่ในช่วง 100 – 99999 Bytes

จากรูปที่ 2 ผู้วิจัยจะพิจารณาถึงขั้นตอนการทำงานของ Squid เว็บแคชเซิร์ฟเวอร์ ในขั้นตอนการจัดเก็บข้อมูลลงในแคชของเว็บแคชเซิร์ฟเวอร์ โดยสามารถแบ่งการทำงานได้ 2 ส่วนคือ ส่วนที่ 1 การจัดเก็บข้อมูลลงในแคชและส่วนที่ 2 การเรียกใช้งานอัลกอริธึมการแทนที่ข้อมูลดังแสดงได้ในขั้นตอนที่ 1 จนถึงขั้นตอนที่ 8 ของรูปที่ 2

เมื่อพิจารณาเฉพาะในส่วนของการเรียกใช้อัลกอริธึมการแทนที่ข้อมูล (ฟังก์ชัน CACHE Mgnt. ดังแสดงในรูปที่ 2) จะแสดงได้ในขั้นตอนที่ 2 ถึงขั้นตอนที่ 7 โดยจะแบ่งขั้นตอนการทำงานออกเป็น 2 ส่วนคือส่วนของ Memory Cache โดย Memory Cache เป็นแคชของเว็บแคชเซิร์ฟเวอร์ที่สร้างจากแรมของเครื่องเซิร์ฟเวอร์และ ส่วนของDisk Cache โดย Disk Cache เป็นแคชของเว็บแคชเซิร์ฟเวอร์ที่สร้างจากฮาร์ดดิสก์ของเครื่องเซิร์ฟเวอร์

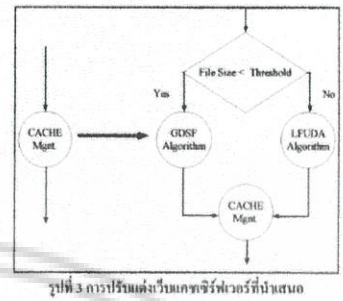
- Memory Cache เป็นการเรียกใช้อัลกอริธึมการแทนที่ข้อมูลสำหรับการเลือกข้อมูลที่จะถูกย้ายจาก Memory Cache ลงไปเก็บไว้ใน Disk Cache ของเว็บแคชเซิร์ฟเวอร์โดยจะมีการทำงานตั้งแต่ขั้นตอนที่ 2 จนถึงขั้นตอนที่ 4 ของรูปที่ 2

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ซึ่งในขั้นตอนที่ 2 เว็บแคชเซิร์ฟเวอร์จะตรวจสอบขนาดของข้อมูลที่อยู่ใน Memory Cache มีค่าเกินกว่าค่าที่กำหนดหรือไม่ หากมีค่าเกิน จะเรียกใช้อัลกอริธึมการแทนที่ข้อมูลและนำข้อมูลที่ถูกเลือกออกไปเก็บไว้ใน Disk Cache ในขั้นตอนที่ 4 จากนั้นจะย้อนกลับไปทำงานในขั้นตอนที่ 2 ถึงขั้นตอนที่ 4 จนกระทั่งค่าของขนาดข้อมูลใน Memory Cache มีค่าน้อยกว่าค่าที่กำหนด

- Disk Cache เป็นการเรียกใช้อัลกอริธึมการแทนที่ข้อมูลสำหรับการเลือกข้อมูลที่จะถูกนำออกไปจากDisk Cache ของเว็บแคชเซิร์ฟเวอร์ โดยจะมีขั้นตอนการทำงานตั้งแต่ขั้นตอนที่ 5 จนถึงขั้นตอนที่ 7 ของรูปที่ 2 ซึ่งในขั้นตอนที่ 5 เว็บแคชเซิร์ฟเวอร์จะตรวจสอบขนาดของข้อมูลที่อยู่ใน Disk Cache มีค่าเกินกว่าค่าที่กำหนดหรือไม่ หากมีค่าเกินจะเรียกใช้อัลกอริธึมการแทนที่ข้อมูล และนำข้อมูลที่ถูกเลือกออกไปจาก Disk Cache ในขั้นตอนที่ 7 จากนั้นจะย้อนกลับไปทำงานในขั้นตอนที่ 5 ถึงขั้นตอนที่ 7 จนกระทั่งค่าของขนาดข้อมูลใน Disk Cache มีค่าน้อยกว่าค่าที่กำหนด

ในการปรับแต่งอัลกอริธึมการแทนที่ข้อมูลของการทดสอบจะเข้าไปปรับแต่งในส่วนฟังก์ชั่นที่ชื่อว่า CACHE Mgnt. ที่แสดงในรูปที่ 2 ซึ่งเป็นฟังก์ชั่นที่ทำหน้าที่เป็นอัลกอริธึมการแทนที่ข้อมูลของเว็บแคชเซิร์ฟเวอร์ โดยเพิ่มฟังก์ชั่นการตรวจสอบขนาดของไฟล์เปรียบเทียบกับค่า Threshold ที่กำหนด หากมีค่ามากกว่าหรือทำทับค่า Threshold จะเลือกใช้อัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA และหากมีค่าน้อยกว่าค่า Threshold จะเรียกใช้อัลกอริธึมการแทนที่ข้อมูลแบบ GDSF ในการทดสอบจะใช้โปรแกรม Squid เวอร์ชัน 2.5 stable 3 และปรับแต่งค่าของอัลกอริธึมการแทนที่ข้อมูลที่ถูกเก็บไว้ภายในไฟล์ /src/repl/heap/store_heap_replacement.c ซึ่งเป็นไฟล์ที่มีหน้าที่ในการสร้างค่าสำหรับใช้ในการพิจารณาการนำข้อมูลออกของอัลกอริธึมการแทนที่ข้อมูลแต่ละชนิด การปรับแต่งที่ผู้วิจัยนำเสนอสามารถแสดงได้ดังรูปที่ 3



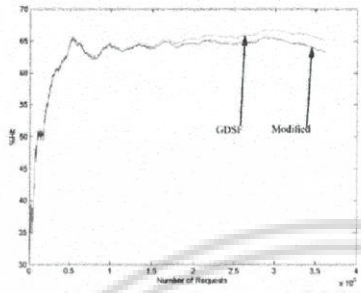
รูปที่ 3 การปรับแต่งเว็บแคชเซิร์ฟเวอร์ที่นำเสนอ

4.3 การทดสอบ

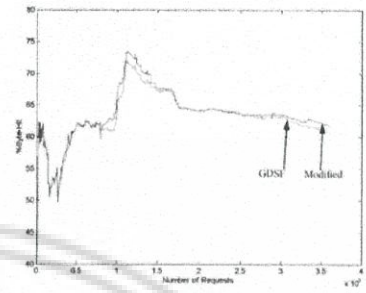
ในการทดสอบจะทำการทดสอบโดยทำการปรับเปลี่ยนค่าของ Threshold จำนวน 3 ค่าคือ 100 Kbytes, 1 Mbytes และ 10 Mbytes โดยใช้ข้อมูลชุดเดียวกันในการทดสอบ และเก็บค่ามาจากไฟล์ Log ซึ่งเป็นไฟล์ที่เก็บรายละเอียดการทำงานของซอฟท์แวร์ Squid แล้วนำมาทำการวิเคราะห์ โดยทำการวิเคราะห์ในส่วนของ %Hit และ %Byte-Hit ซึ่งกราฟที่ได้จากการปรับเปลี่ยนค่า Threshold จะนำมาเปรียบเทียบกับกราฟของอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF ที่ใช้ข้อมูลทดสอบชุดเดียวกับการทดสอบค่า Threshold เนื่องจากอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF จะให้ค่า %Hit ที่สูงที่สุดและค่า %Byte-Hit ที่ต่ำที่สุด จึงเหมาะสมสำหรับการนำมาเปรียบเทียบประสิทธิภาพของการปรับแต่งอัลกอริธึมที่ผู้วิจัยได้นำเสนอมา ค่า %Hit ที่ได้จากการวิเคราะห์ค่า Threshold ที่มีค่า 100 Kbytes, 1 Mbytes และ 10 Mbytes แสดงดังรูป 4, 5 และ 6 ส่วนค่าของ %Byte-Hit ที่ได้จากการวิเคราะห์ค่า Threshold ที่มีค่า 100 Kbytes, 1 Mbytes และ 10 Mbytes แสดงดังรูปที่ 7, 8 และ 9 ตามลำดับ

- ค่าในแกน X แสดงถึง จำนวนครั้งของการร้องขอที่เกิดขึ้น เริ่มจาก 0 ไปถึง 350,000 ครั้ง
- ค่าในแกน Y แสดงถึง %Hit และ %Byte-Hit ของค่า Threshold ตั้งแต่ 100 Kbytes, 1 Mbytes และ 10 Mbytes ตามลำดับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

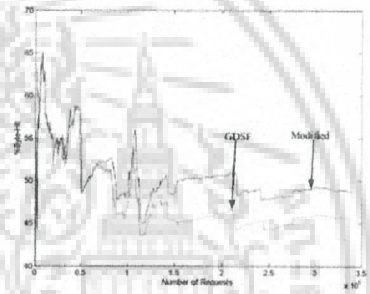


รูปที่ 4 ค่า %Hit ที่ค่า Threshold มีค่า 100 Kbytes

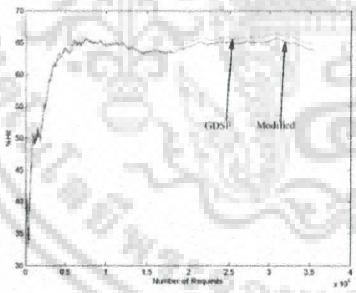


รูปที่ 7 ค่า %Byte-Hit ที่ค่า Threshold มีค่า 100 Kbytes

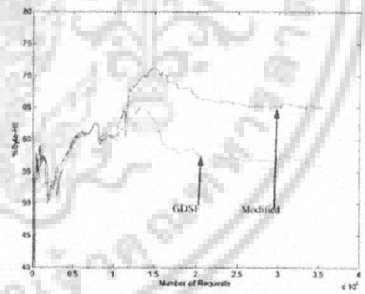
รูปที่ 5 ค่า %Hit ที่ค่า Threshold มีค่า 1 Mbytes



รูปที่ 8 ค่า %Byte-Hit ที่ค่า Threshold มีค่า 1 Mbytes



รูปที่ 6 ค่า %Hit ที่ค่า Threshold มีค่า 10 Mbytes



รูปที่ 9 ค่า %Byte-Hit ที่ค่า Threshold มีค่า 10 Mbytes

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.4 การวิเคราะห์ผลที่ได้

จากกราฟในรูปที่ 4,5 และ 6 ที่แสดงถึง %Hit ของค่า Threshold ที่ค่า 100 Kbytes, 1 Mbytes และ 10 Mbytes ตามลำดับ พบว่าค่าของ %Hit ก่อนทำการปรับแต่งอัลกอริธึมการแทนที่ข้อมูล (GDSF) และหลังทำการปรับแต่งอัลกอริธึมการแทนที่ข้อมูล (Modified) ของค่า Threshold แต่ละค่าจะมีรูปกราฟใกล้เคียงกันดังรูปที่ 4 และ 6 หรือเกือบจะเหมือนกันรูปที่ 5 เนื่องจากค่า %Hit จะเป็นค่าที่พิจารณาถึงจำนวนครั้งของการพบข้อมูลที่อยู่ในแคชของเว็บแคชเซิร์ฟเวอร์ ซึ่งตามสมติฐานที่ตั้งไว้ค่า %Hit ของอัลกอริธึมที่ถูกปรับแต่งจะต้องมีค่าลดต่ำลงกว่าค่า %Hit ของอัลกอริธึมการแทนที่ข้อมูลก่อนทำการปรับแต่ง

เนื่องจากเมื่อทำการปรับแต่งให้ข้อมูลที่มีขนาดใหญ่บางส่วนยังคงถูกเก็บไว้ในแคชของเว็บแคชเซิร์ฟเวอร์ จากการทำงานของอัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA จะเป็นผลทำให้พื้นที่เก็บข้อมูลของแคชมีค่าลดน้อยลง ดังนั้นเว็บแคชเซิร์ฟเวอร์จึงสามารถเก็บข้อมูลที่มีขนาดเล็กได้น้อยลงซึ่งจะเป็นผลทำให้ค่าของ %Hit ที่ได้มีค่าลดลง แต่เนื่องจากค่า %Hit จะเป็นค่าที่พิจารณาถึงจำนวนครั้งในการพบข้อมูล ดังนั้น เมื่อจำนวนการพบข้อมูลมีค่าที่ลดลงไม่มากเมื่อเทียบกับการร้องขอข้อมูลที่มีค่ามากขึ้นเรื่อยๆ จะทำให้ค่าของ %Hit ที่ออกมามีค่าไม่ต่างไปจากค่าของ %Hit ที่ได้ก่อนที่จะทำการปรับแต่งอัลกอริธึมการแทนที่ข้อมูล

เมื่อพิจารณาจากรูปที่ 7, 8 และ 9 ที่แสดงถึงค่าของ %Byte-Hit ที่ค่าของ Threshold มีค่า 100 Kbytes, 1 Mbytes และ 10 Mbytes พบว่าเมื่อทำการปรับค่าของ Threshold ให้มีค่ามากขึ้นเรื่อยๆ ค่าของ %Byte-Hit ซึ่งเป็นค่าที่พิจารณาถึงการพบข้อมูลโดยพิจารณาจากขนาดของข้อมูลที่พบในแคช จะมีค่าสูงขึ้นเมื่อเทียบกับค่าของ %Byte-Hit ก่อนทำการปรับแต่งอัลกอริธึมการแทนที่ข้อมูล

เนื่องจากการตั้งค่าของ Threshold ให้มีขนาด 100 Kbytes ดังรูปที่ 7 จะทำให้ข้อมูลที่มีขนาดใหญ่ที่ควรถูกเก็บไว้ในแคชของเว็บแคชเซิร์ฟเวอร์ถูกนำออกไปจากแคช เพราะข้อมูลที่มีขนาดเล็กซึ่งเป็นข้อมูลประเภทรูปภาพจะถูกเรียกใช้งานออกไปจากแคชมากกว่าข้อมูลที่มีขนาดใหญ่ จากการทำงานของอัลกอริธึมการแทนที่ข้อมูลแบบ LFUDA และในรูปที่ 8 และ 9 ที่มีการตั้งค่าของ Threshold เป็น 1 Mbytes และ 10 Mbytes จะพบว่าข้อมูลที่มีขนาดใหญ่จะถูกเก็บไว้ในแคชและถูกเรียกใช้งานทำให้ค่าของ %Byte-Hit ที่ได้ซึ่งพิจารณาถึงขนาดของไฟล์ที่มีการพบในแคชมีค่าสูง แต่เมื่อพิจารณาในรูปที่ 9 ที่มี Threshold เป็น 10 Mbytes ซึ่งพบว่าความแตกต่างระหว่างค่าของ %Byte-Hit หลังทำการปรับแต่งอัลกอริธึมการแทนที่ข้อมูลกับค่าของ %Byte-Hit ก่อนทำการปรับแต่งอัลกอริธึมการแทนที่ข้อมูลมีค่ามากสุด จะมีค่าความแตกต่างระหว่างค่า %Hit หลังทำการปรับแต่งอัลกอริธึมการแทนที่ข้อมูลกับค่าของ %Hit ก่อนทำการปรับแต่งอัลกอริธึมการแทนที่ข้อมูลมากเช่นกัน เนื่องจากเว็บแคชเซิร์ฟเวอร์ใช้เนื้อที่ในการเก็บข้อมูลที่มีขนาดใหญ่หลายข้อมูลทำให้เหลือพื้นที่สำหรับนำมาเก็บข้อมูลที่มีขนาดเล็กน้อยลง ทำให้การพบข้อมูลที่ร้องขอจากในแคชของเว็บแคชเซิร์ฟเวอร์มีค่าลดลงตามไปด้วย

5. สรุปผล

งานวิจัยนี้ นำเสนอการออกแบบและปรับแต่งอัลกอริธึมการแทนที่ข้อมูลของ Squid เว็บแคชเซิร์ฟเวอร์ให้มีความสามารถในการปรับเปลี่ยนชนิดของอัลกอริธึมการแทนที่ข้อมูลให้เหมาะสมได้ตามขนาดของข้อมูล เพื่อเพิ่มประสิทธิภาพการทำงานของระบบเว็บแคชเซิร์ฟเวอร์

ผลการวิจัยพบว่าค่า %Hit ของเว็บแคชเซิร์ฟเวอร์ที่ใช้อัลกอริธึมการแทนที่ข้อมูลที่ถูกปรับแต่งจะมีค่าใกล้เคียงกันกับค่า %Hit ของเว็บแคชเซิร์ฟเวอร์ที่ใช้อัลกอริธึมการแทนที่ข้อมูลก่อนทำการปรับแต่ง และค่า %Byte-Hit ของเว็บแคชเซิร์ฟเวอร์ที่ใช้อัลกอริธึมการแทนที่ข้อมูลที่ถูกปรับแต่งจะมีค่าเพิ่มขึ้นเมื่อเทียบกับค่า %Byte-Hit ของเว็บแคชเซิร์ฟเวอร์ที่ใช้อัลกอริธึมการแทนที่ข้อมูลก่อนทำการปรับแต่ง เนื่องจากเนื้อที่ของแคชบางส่วนจะต้องถูกนำไปใช้เก็บข้อมูลที่มีขนาดใหญ่ทำให้ข้อมูลที่มีขนาดเล็ก ถูกนำออกไปมากกว่าอัลกอริธึมการแทนที่ข้อมูลแบบ GDSF และการที่ค่าของ %Byte-Hit มีค่าเพิ่มขึ้น ทำให้การเรียกใช้ข้อมูลจากระบบ Internet มีปริมาณน้อยลงและความหนาแน่นของข้อมูลในระบบเครือข่ายก็จะมีค่าลดลง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากงานวิจัย ผู้วิจัยพบว่าหากมีการศึกษาเพื่อหาค่า Threshold ที่เหมาะสมสำหรับข้อมูลของเว็บแคชเซิร์ฟเวอร์ จะทำให้การปรับแต่งอัลกอริธึมการแทนที่ข้อมูลที่นำเสนอมาสามารถให้ประสิทธิภาพการทำงานใกล้เคียงกับค่า %Hit ของอัลกอริธึมการแทนที่ข้อมูลก่อนทำการปรับแต่ง และ ค่า %Byte-Hit ที่ได้จะมีค่าเพิ่มขึ้นจากอัลกอริธึมการแทนที่ข้อมูลก่อนทำการปรับแต่งมากที่สุด

เอกสารอ้างอิง

[1] J. Dilley, M. Arlitt and S. Perret, "Enhancement and Validation of Squid's Cache Replacement Policy", Technical Report HPL-1999-69, Hewlett-Packard Laboratories, May, 1999.

[2] S. Jin and A. Bestavros, "Popularity-aware greedy dual-size Web proxy caching algorithms", In: *Proceedings of 20th International Conference on Distributed Computing Systems*, pp. 254-261 April 2000.

[3] L. Cherkasova, "Improving WWW proxies Performance with Greedy-Dual-Size-Frequency Caching Policy", Technical Report HPL-1998-69R1, Hewlett-Packard Laboratories, November 1998.

[4] M. Arlitt, L. Cherkasova, J. Dilley, R. Friedrich and T. Jin, "Evaluating Content Management Techniques for Web Proxy Caches", Technical Report HPL-98-173, Hewlett-Packard Laboratories, April 1999.

[5] Squid Internet Object Cache Software, Available at http://www.squid-cache.org/.

[6] Cfmc Software, Available at http://www.cachefiow.com/.

[7] Debian Linux Operating System Software, Available at http://www.debian.org/.

[8] M. Arlitt and Carey L. Williamson, "Internet Web Servers: Workload Characterization and Performance Implications", In: *Proceeding of IEEE/ACM Transactions on Networking*, Vol 5, No. 5, October 1997.

[9] W. Intraha, "Proxy Performance Tuning for HTTP and FTP Protocol." Master Thesis of Information Science, King Mongkut's Institute of Technology Ladkrabang, 2001.

[10] P. Sopechoke and B. Piyatamrong, "Improvement Web Cache Server with Hierarchical Replacement Algorithm" In: *Proceedings of the seventh National Computer Science and Engineering Conference*, pp.140-144 October 2003.

ภาวิน สพโชค สำเร็จการศึกษาระดับปริญญาตรี สาขาวิศวกรรมอิเล็กทรอนิกส์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ปีการศึกษา 2540 และปริญญาโท สาขาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ปีการศึกษา 2546

ศักดิ์ชัย ทิพย์จักษุรัตน์ สำเร็จการศึกษาระดับปริญญาเอก สาขา Computer Science จาก Gunma University ประเทศญี่ปุ่น ปัจจุบันดำรงตำแหน่ง อาจารย์ ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

บรรจง ปียธำรง ดำรงตำแหน่งรองศาสตราจารย์ ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้