ระบบแทนช่วงแบบยืดหยุ่น

นายพิภพ เทียนประภาสิทธิ์

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิทยาศาสตร์คอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2550

ลิขสิทธิ์ของจุฬาลงกรณ์มหาวิทยาลัย

FLEXIBLE INTERVAL REPRESENTATION SYSTEM

Mr.Pipop Thienprapasith

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Science Program in Computer Science
Department of Computer Engineering
Faculty of Engineering
Chulalongkorn University
Academic Year 2007
Copyright of Chulalongkorn University

หัวข้อวิทยานิพนธ์	ระบบแทนช่วงแบบยืดหยุ่น
โดย	นายพิภพ เทียนประภาสิทธิ์
สาขาวิชา	วิทยาศาสตร์คอมพิวเตอร์
อาจารย์ที่ปรึกษา	ผู้ช่วยศาสตราจารย์ ดร.อรรถสิทธิ์ สุรฤกษ์

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

.... คณบดีคณะวิศวกรรมศาสตร์
(รองศาสตราจารย์ ดร.บุญสม เลิศหิรัญวงศ์)

คณะกรรมการสอบวิทยานิพนธ์

.... ประธานกรรมการ
(ศาสตราจารย์ ดร.ประภาส จงสถิตย์วัฒนา)

.... อาจารย์ที่ปรึกษา
(ผู้ช่วยศาสตราจารย์ ดร.อรรถสิทธิ์ สุรฤกษ์)

.... กรรมการ
(อาจารย์ ดร.เศรษฐา ปานงาม)

.... กรรมการ
(ผู้ช่วยศาสตราจารย์ ดร.อานนท์ รุ่งสว่าง)

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

พิภพ เทียนประภาสิทธิ์ : ระบบแทนช่วงแบบยืดหยุ่น. (FLEXIBLE INTERVAL REPRESENTATION SYSTEM) อ.ที่ปรึกษา : ผศ.ดร.อรรถสิทธิ์ สุรฤกษ์, 43 หน้า.

ในวงการงานวิจัยทางด้านเลขคณิตสำหรับคอมพิวเตอร์นั้น ปัญหาที่เราสนใจคือเรื่องความเร็วในคำนวณ โดยงานวิจัยเป็นจำนวนมากมุ่งเน้นไปที่ปัจจัยและเทคนิคที่สามารถทำให้การคำนวณมีความเร็วสูง แต่ในบางครั้งเราไม่สามารถคำนวณให้ได้คำตอบที่ถูกต้องเสมอไป สาเหตุสองประการของการคำนวณที่ให้คำตอบไม่ถูกต้อง ประการแรก ระบบคอมพิวเตอร์แบบดั้งเดิมนั้นประสบปัญหาในเรื่องของการปัดเศษทิ้ง สาเหตุเกิดจากรูปแบบที่จำกัดในการแทนจำนวนจริง ทั้งนี้ปัญหาการปัดเศษทิ้งสามารถเกิดขึ้นได้ในระหว่างกระบวนการคำนวณ ประการที่สอง ความคลาดเคลื่อนของข้อมูลนำเข้าที่จะนำมาใช้ในการคำนวณ จะส่งผลต่อความถูกต้องของข้อมูลนำออกด้วย ดังนั้นระบบแทนช่วงจึงได้ถูกเสนอขึ้นเพื่อแก้ปัญหาดังกล่าว เนื่องจากช่วงประกอบด้วยจำนวนสองจำนวน เราจึงสามารถรับประกันได้ว่าข้อมูลนำเข้าที่มีความคลาดเคลื่อนสามารถถูกเขียนให้อยู่ในรูปของช่วงได้ แต่อย่างไรก็ตามระบบแทนช่วงประสบปัญหาทางด้านความสิ้นเปลืองเนื้อที่และความล่าช้าในการคำนวณ

ในงานวิจัยนี้จึงได้เสนอระบบแทนช่วงแบบยืดหยุ่น โดยผลลัพธ์ทางทฤษฎีแสดงให้เห็นว่าระบบนี้สามารถลดเนื้อที่ที่ใช้ในการแทนช่วงให้น้อยลง 25 เปอร์เซ็นต์เมื่อเปรียบเทียบกับระบบแทนช่วงแบบมีเครื่องหมายดั้งเดิม อีกทั้งการดำเนินการพื้นฐานทางเลขคณิต ได้แก่ การบวก ลบ คูณ และหารได้ถูกเสนอในงานวิจัยนี้ด้วย

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

ภาควิชา วิศวกรรมคอมพิวเตอร์ ลายมือชื่อนิสิต พิภพ เทียนประภาสิทธิ์
สาขาวิชา วิทยาศาสตร์คอมพิวเตอร์ ลายมือชื่ออาจารย์ที่ปรึกษา Athasit Surarerks
ปีการศึกษา 2550

4770380721 : MAJOR COMPUTER SCIENCE
KEY WORD : INTERVAL ARITHMETIC / REDUNDANT NUMBER REPRESENTATION SYSTEM / SIGNED-DIGIT NUMBER REPRESENTATION SYSTEM

PIPOP THIENPRAPASITH : FLEXIBLE INTERVAL REPRESENTATION SYSTEM. THESIS ADVISOR : ATHASIT SURARERKS, Ph.D., 43 pp.

A major problem in a domain of computer arithmetic concerns how computational time can be speeded up. Many researches focused on introducing high speed computing techniques. However, the computation may not always produce the exact value. In detail, two types of inexact arithmetic are produced. First, the classical computer system has round-off error problem caused by the finite representation of real number. Round-off error is usually occurred during the computation process. Second, uncertainty in the input data affects the correct value of the output data. Therefore, interval representation system is established to handle the problem. Since an interval is a pair of numbers, it is guaranteed that uncertainty in the input data can be represented in this system. However, the space used and computational time for interval arithmetic is very high.

This thesis proposes a flexible interval representation system. Theoretical results show that the space used for representation an interval can be reduced up to twenty-five percents, compared to space used for the classical signed digit interval representation system. Fundamental arithmetic operations such as addition, subtraction multiplication and division are also introduced in this work.

Department Computer Engineering Student's signature พิภพ เทียนประภาสิทธิ์
Field of study Computer Science Advisor's signature Athasit Surarerks
Academic year 2007

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงไปได้ด้วยความอนุเคราะห์ และความช่วยเหลืออย่างดียิ่งจาก ผศ.ดร.อรรถสิทธิ์ สุรฤกษ์ อาจารย์ที่ปรึกษา ซึ่งเป็นผู้ให้คำปรึกษา ให้ข้อเสนอแนะที่เป็นประโยชน์อย่างยิ่งต่อการวิจัย และช่วยตรวจแก้ไขในส่วนที่บกพร่องต่างๆ ทำให้วิทยานิพนธ์ฉบับนี้สำเร็จลุล่วง

ขอขอบพระคุณ ศาสตราจารย์ ดร.ประภาส จงสถิตย์วัฒนา อาจารย์ ดร.เศรษฐา ปานงาม ผู้ช่วยศาสตราจารย์ ดร.อานนท์ รุ่งสว่าง ประธานกรรมการและกรรมการสอบวิทยานิพนธ์ ที่ได้กรุณาให้คำแนะนำในการแก้ไขวิทยานิพนธ์ให้มีคุณภาพยิ่งขึ้น และขอขอบพระคุณคณาจารย์ในภาควิชาวิศวกรรมคอมพิวเตอร์ จุฬาลงกรณ์มหาวิทยาลัยทุกท่านที่ประสิทธิประสาทความรู้อันมีค่ายิ่งแก่ผู้วิจัย

กราบขอบพระคุณ บิดา มารดา ที่เป็นกำลังใจสำคัญตลอดมา ขอขอบคุณ พี่ๆ เพื่อนๆ และน้องๆ ทุกคน โดยเฉพาะสมาชิกห้องปฏิบัติการวิจัย ELITE ที่ได้ให้ความช่วยเหลือ เสนอแนะ และแก้ไขเอกสาร จนกระทั่งวิทยานิพนธ์สำเร็จด้วยดี

สุดท้ายนี้ ผู้วิจัยหวังเป็นอย่างยิ่งว่างานวิจัยนี้จะเป็นประโยชน์ต่อผู้ที่สนใจหรือผู้ที่เกี่ยวข้องทั่วไป และหากมีข้อผิดพลาดประการใด ผู้วิจัยขออภัยมา ณ ที่นี้ด้วย

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

สารบัญ

บทที่ หน้า

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

สารบัญตาราง

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

สารบัญภาพ

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

คำอธิบายสัญลักษณ์และคำย่อ

สัญกรณ์ทางคณิตศาสตร์

X	รูปแบบแทนจำนวน (number representation)
β	ค่าฐาน (base)
min()	การเลือกค่าที่น้อยที่สุด (minimum)
max()	การเลือกค่าที่มากที่สุด (maximum)
lower	ค่าต่ำสุดของช่วงที่ได้จากการคำนวณ
upper	ค่าสูงสุดของช่วงที่ได้จากการคำนวณ
x_i	ตัวเลขตั้งต้นในระบบแทนช่วงแบบยืดหยุ่น
y_i	ตัวเลขที่นำมาคำนวณกับตัวเลขตั้งต้นในระบบแทนช่วงแบบยืดหยุ่น
z_i	ตัวเลขที่เป็นผลลัพธ์ในระบบแทนช่วงแบบยืดหยุ่น
λ	ฟังก์ชันการแปลงระหว่างชุดตัวเลข (conversion mapping function)
σ_d	ฟังก์ชันการส่งผ่านตัวทด (carry-transfer function)
ε_d	ฟังก์ชันจับคู่ตัวเลข (digit mapping function)

บทที่ 1

บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ในระบบการแทนจำนวนของจำนวนจริงด้วยรูปแบบที่จำกัด (finite representation) เป็นเรื่องที่ไม่สามารถทำได้ ซึ่งในระบบคอมพิวเตอร์จำเป็นต้องใช้รูปแบบแทนจำนวนที่จำกัดทำให้จำนวนจริงบางจำนวนไม่สามารถที่จะแสดงค่าที่ถูกต้องได้ ระบบแทนจำนวนสำหรับจำนวนจริงได้ถูกพัฒนาขึ้นมาหลายระบบเพื่อหลีกเลี่ยงปัญหาความคลาดเคลื่อนที่เกิดจากข้อจำกัดของรูปแบบแทนจำนวน อย่างไรก็ตามการคำนวณบนระบบที่สร้างขึ้นก็ไม่อาจรับประกันได้ว่าผลลัพธ์ของการคำนวณจะได้ค่าที่ถูกต้องเสมอเพราะอาจต้องทำการปัดเศษเลขทศนิยมในตำแหน่งที่หน่วยประมวลผลไม่สามารถแสดงได้ (round-off error) ผลลัพธ์ที่ได้จึงมีความคลาดเคลื่อน ถึงแม้ความคลาดเคลื่อนจะปรากฏเพียงเล็กน้อย แต่ความคลาดเคลื่อนนี้จะปรากฏเด่นชัดมากขึ้น เมื่อผลลัพธ์ถูกนำไปคำนวณต่อเรื่อยๆ [1, 2] ในกรณีเช่นนี้ทำให้เกิดค่าความคลาดเคลื่อนสะสม

นอกจากนี้ ความคลาดเคลื่อนของข้อมูลนำเข้าที่ไม่ทราบค่าที่ถูกต้อง (อาจเนื่องมาจากปัจจัยภายนอกเช่น เครื่องมือวัดมีความคลาดเคลื่อน เป็นต้น) จะส่งผลที่ก่อให้เกิดความผิดพลาดคลาดเคลื่อนไปจากคำตอบที่ถูกต้องมากยิ่งขึ้นไปด้วย แนวทางในการแก้ปัญหาจึงได้ถูกเสนอขึ้น ระบบแทนจำนวนแบบช่วง (interval number representation system) เป็นการแสดงจำนวนต่างๆ ให้อยู่ในรูปของช่วงที่ครอบคลุมค่าที่ถูกต้องของจำนวนจริงนั้นแทนรูปแบบแทนจำนวนที่มีข้อจำกัด อีกทั้งยังได้เสนอวิธีการของการคำนวณแบบช่วง (interval computation) เข้ามาใช้ เพื่อให้หน่วยประมวลผลสามารถจำกัดขอบเขตของความคลาดเคลื่อนได้ ทำให้ผลลัพธ์ที่ถูกต้องนั้นปรากฏอยู่ในช่วงที่ได้จากการคำนวณแน่นอน สำหรับแนวทางในการพัฒนาระบบแทนจำนวนแบบช่วงให้สามารถใช้ร่วมกับฮาร์ดแวร์สามารถศึกษาได้จาก [3]

ถึงแม้การคำนวณแบบช่วงจะสามารถแก้ปัญหาในเรื่องการคำนวณเลขทศนิยมได้และได้ถูกนำมาใช้ในการออกแบบหน่วยประมวลผลต่อมา แต่เนื่องจากจำนวนที่แสดงอยู่ในรูปของช่วงนั้นเกิดจากจำนวน 2 จำนวน ส่งผลให้ต้องใช้เนื้อที่ในการเก็บมากกว่าปกติ นอกจากนี้การคำนวณแบบช่วงก็มีข้อเสียในเรื่องความเร็วในการคำนวณที่ต้องใช้เวลามากกว่าการคำนวณแบบปกติ จึงได้มีความพยายามคิดค้นวิธีที่จะแก้ปัญหาดังกล่าวโดยนำระบบจำนวนซ้ำซ้อน (redundant number system) เข้ามาใช้ ซึ่งก็สามารถเพิ่มความเร็วในการคำนวณให้สูงขึ้น แต่เนื่องจากระบบจำนวนซ้ำซ้อนมีการขยายชุดตัวเลข (digit set) เพิ่มขึ้นเกือบสองเท่า ส่งผลให้เกิดข้อเสียในเรื่องขนาดของเนื้อที่ (space) ที่ต้องใช้ในการแทนค่าตัวเลข (digit) ที่ต้องเพิ่มขึ้นเป็นสองเท่าตามไปด้วย

งานวิจัยนี้ มุ่งเน้นที่จะแก้ปัญหาในเรื่องของการลดขนาดของเนื้อที่ที่ใช้ในการเก็บจำนวนซึ่งสามารถนำไปสู่การลดเวลาที่ใช้ในการคำนวณให้น้อยลงได้ ดังนั้นในงานนี้จึงได้มีการนำเสนอระบบแทนจำนวนแบบใหม่ที่เรียกว่า ระบบแทนช่วงแบบยืดหยุ่น (flexible interval representation system) พร้อมทั้งเสนออัลกอริทึมในการแปลงชุดตัวเลข (digit set conversion algorithm) ในระบบนี้ด้วย

1.2 วัตถุประสงค์ของงานวิจัย

ออกแบบระบบแทนช่วงแบบยืดหยุ่นที่สามารถลดขนาดของรูปแบบแทนช่วงลงได้ พร้อมทั้งออกแบบอัลกอริทึมสำหรับการคำนวณของตัวดำเนินการพื้นฐานทางเลขคณิต คือ การบวก ลบ คูณ และหาร ของระบบแทนช่วงแบบยืดหยุ่นนี้ด้วย

1.3 ขอบเขตของงานวิจัย

1) เสนอระบบแทนช่วงแบบยืดหยุ่น และอัลกอริทึมสำหรับการดำเนินการพื้นฐานทางเลขคณิตในรูปแบบของการแปลงชุดตัวเลข
2) เสนออัลกอริทึมในการแปลงระหว่างระบบแทนจำนวนแบบมีเครื่องหมายและระบบแทนช่วงแบบยืดหยุ่น

1.4 ขั้นตอนและวิธีดำเนินงานวิจัย

1) ศึกษางานวิจัยทางด้านการคำนวณแบบช่วง
2) วิเคราะห์ปัญหาของงานวิจัยที่มีความสอดคล้องกับงานวิจัยที่สนใจ
3) ออกแบบระบบแทนช่วงแบบยืดหยุ่น
4) ออกแบบอัลกอริทึมในการแปลงระบบแทนจำนวนและอัลกอริทึมสำหรับการดำเนินการพื้นฐานทางเลขคณิต
5) พิสูจน์อัลกอริทึมที่ได้ออกแบบไว้
6) จัดทำรายงานวิทยานิพนธ์

1.5 ประโยชน์ที่คาดว่าจะได้รับจากงานวิจัย

1) ได้ระบบแทนช่วงแบบใหม่ที่สามารถลดขนาดของรูปแบบแทนช่วงลงได้
2) ได้อัลกอริทึมการแปลงชุดตัวเลข สำหรับระบบแทนช่วงแบบใหม่

1.6 ผลงานที่ตีพิมพ์จากวิทยานิพนธ์

ส่วนหนึ่งของวิทยานิพนธ์นี้ได้รับการตีพิมพ์เป็นผลงานวิชาการในหัวข้อเรื่องดังต่อไปนี้

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

1) "Flexible interval representation system" โดย พิภพ เทียนประภาสิทธิ์ และ อรรถสิทธิ์ สุรฤกษ์ ในงานประชุมวิชาการ 10th Annual National Symposium on Computational Science and Engineering (ANSCSE10)
2) "Fundamental arithmetic operations for the flexible interval representation system" โดย พิภพ เทียนประภาสิทธิ์ และอรรถสิทธิ์ สุรฤกษ์ ในงานประชุมวิชาการ National Computer Science and Engineering Conference (NCSEC2006)
3) "Multiplication and Division for the flexible interval representation system" โดย พิภพ เทียนประภาสิทธิ์ และอรรถสิทธิ์ สุรฤกษ์ ในงานประชุมวิชาการ National Computer Science and Engineering Conference (NCSEC2007)
4) "Modified Arithmetic Algorithm for Flexible Interval Binary Representation System" โดย จักรพันธุ์ สุคนธราช พิภพ เทียนประภาสิทธิ์ และอรรถสิทธิ์ สุรฤกษ์ ในงานประชุมวิชาการ 12th Annual National Symposium on Computational Science and Engineering (ANSCSE12)
5) "A Flexible Interval Representation System and Its Fundamental Arithmetic Operations" โดย พิภพ เทียนประภาสิทธิ์ และอรรถสิทธิ์ สุรฤกษ์ ในงานประชุมวิชาการ 5th International Conference on Information Technology and Applications (ICITA 2008)

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

บทที่ 2

ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

2.1 ระบบจำนวน (number system)

ระบบจำนวน (β, D) ประกอบด้วยฐาน (base) β ซึ่ง β สามารถเป็นจำนวนจริงหรือจำนวนเชิงซ้อน โดยที่ $\|\beta\| > 1$ และชุดตัวเลขจำกัด (finite digit set) D โดยที่สมาชิกในชุดตัวเลขที่เรียกว่า ตัวเลข (digit) สามารถเป็นได้ทั้งจำนวนจริงและจำนวนเชิงซ้อน

รูปแบบแทนจำนวน X บนชุดตัวเลข D ภายใต้ฐาน β สามารถเขียนได้ดังนี้

$$X = (x_n x_{n-1} \ldots x_0 . x_{n-1} x_{n-2} \ldots)_\beta$$

โดยที่ $x_i \in D$ ซึ่ง $i \le n,\ n \in Z$

ค่าเชิงตัวเลข (numerical value) ของ X ในฐาน β สามารถเขียนได้ดังนี้

$$\|X\| = \sum_{i=-\infty}^{n} x_i \beta^n$$

ตัวอย่าง 2.1 *กำหนดให้ระบบแทนจำนวนประกอบด้วยเลขฐาน $\beta = 5$ และมีชุดตัวเลข $D = \{0,1,2,3,4\}$ จงเขียนค่าของ X =1234 ให้อยู่ในระบบจำนวนนี้*

วิธีทำ ค่าของ X =1234 สามารถเขียนให้อยู่ในระบบนี้ได้เป็น

$$1234 = (14414)_5$$ □

2.2 ระบบจำนวนแบบมีเครื่องหมาย (signed digit number system)

ระบบจำนวนแบบมีเครื่องหมายนี้ได้ถูกเสนอขึ้นครั้งแรกโดย อวีเซียนีส (Avizienis) [4] ปี ค.ศ. 1961 จุดประสงค์ของการเสนอระบบจำนวนใหม่นี้เพื่อจำกัดการแพร่ของตัวทดในระหว่างการคำนวณ โดยตัวเลขใหม่ที่ถูกสร้างขึ้นสามารถมีเครื่องหมายกำกับได้ นอกจากนี้จำนวนตัวเลขในชุดตัวเลขจะมีมากกว่าจำนวนตัวเลขในระบบดั้งเดิม

ระบบจำนวนแบบมีเครื่องหมายของอวีเซียนีสนั้น กำหนดให้มีค่าฐาน $\beta \ge 3$ โดยชุดตัวเลขกำหนดให้เป็น

$$D = \{-a, -a+1, \ldots, -1, 0, 1, \ldots, a-1, a\}$$

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

โดยที่ a เป็นจำนวนเต็มใด ๆที่อยู่ในช่วงของ $\frac{1}{2} < a < \beta$ ในกรณีที่ฐาน $\beta = 2$ จะมีชุดตัวเลขเป็น $D = \{\bar{1}, 0, 1\}$

หมายเหตุ *ในการเขียนตัวเลขในชุดตัวเลขแบบมีเครื่องหมาย นิยมใช้สัญลักษณ์* $\bar{d}$ *แทนตัวเลข* $-d$

ระบบจำนวนแบบมีเครื่องหมายนี้มีคุณสมบัติที่สำคัญประการหนึ่งคือ สมบัติซ้ำซ้อน (redundant property) ซึ่งหมายความว่า จำนวนบางจำนวนสามารถมีรูปแบบแทนจำนวน (number representation) แบบมีเครื่องหมายได้มากกว่าหนึ่งรูปแบบ

ตัวอย่าง 2.2 *กำหนดให้ระบบแทนจำนวนแบบมีเครื่องหมายประกอบด้วยเลขฐาน* $\beta = 5$ *และมีชุดตัวเลข* $D = \{\bar{3}, \bar{2}, \bar{1}, 0, 1, 2, 3\}$ *จงเขียนค่าของ X =1234 ให้อยู่ในระบบจำนวนนี้*

วิธีทำ ค่าของ X = 1234 สามารถเขียนให้อยู่ในระบบนี้ได้มากกว่าหนึ่งแบบคือ

$$1234 = \left(200\bar{3}\bar{1}\right)_5 = \left(1\bar{3}00\bar{3}\bar{1}\right)_5 \qquad \square$$

ต่อมาในปี ค.ศ.1990 พาร์ฮามี (Parhami) [5] ได้เสนอรูปแบบทั่วไปของระบบแทนจำนวนแบบมีเครื่องหมาย (generalized signed-digit: GSD) ซึ่งคุณสมบัติที่แตกต่างจากระบบเดิมคือ ชุดตัวเลขไม่จำเป็นต้องเป็นเซตที่สมมาตร โดยกำหนดเลขฐาน β เป็นจำนวนเต็มบวกที่ $\beta \geq 2$ ชุดตัวเลขสามารถกำหนดให้เป็น

$$D = \{-l, -l+1, -l+2, \ldots, \bar{1}, 0, 1, \ldots, m-1, m\}$$

โดยที่ $l \leq 0$ และ $m \geq 0$ โดยที่ $m - l + 1 > \beta$

ตัวอย่าง 2.3 *กำหนดให้ระบบแทนจำนวนแบบมีเครื่องหมายประกอบด้วยเลขฐาน* $\beta = 5$ *และมีชุดตัวเลข* $D = \{\bar{3}, \bar{2}, \bar{1}, 0, 1, 2\}$ *จงเขียนค่าของ X =1234 ให้อยู่ในระบบจำนวนนี้*

วิธีทำ ค่าของ X = 1234 สามารถเขียนให้อยู่ในระบบนี้ได้เป็น

$$1234 = \left(20\bar{1}2\bar{1}\right)_5 \qquad \square$$

ข้อดีของระบบแทนจำนวนซ้ำซ้อนคือมีความสามารถในการจำกัดการแพร่ของตัวทดในการคำนวณทางคณิตศาสตร์ไม่ให้แพร่ไปอย่างไม่จำกัดได้ ซึ่งต่างไปจากระบบแทนจำนวนไม่ซ้ำซ้อน เพราะฉะนั้นระบบแทนจำนวนแบบมีเครื่องหมายจึงสามารถทำการคำนวณ

แบบขนานได้ สำหรับบางตัวดำเนินการ (operator) ทำให้เวลาที่ใช้ในการคำนวณคงที่และไม่ขึ้นกับขนาดความยาวของตัวถูกดำเนินการ (operand)

2.3 ระบบแทนช่วง (interval representation system)

ในระบบการคำนวณสำหรับจำนวนจริงนั้น ความผิดพลาดอาจเกิดขึ้นได้ในระหว่างการคำนวณ ทั้งนี้เพราะรูปแบบจำกัดของการแทนจำนวน (finite representation of number) แนวคิดในการแทนจำนวนโดยใช้ช่วงและการคำนวณในรูปแบบของช่วงนั้นเปรียบเสมือนเป็นการคำนวณเพื่อหาขอบเขต (bound) ที่เป็นไปได้ของคำตอบในกรณีที่จำนวนที่ต้องการนำมาคำนวณนั้นมีความคลาดเคลื่อนไปได้ในช่วงที่กำหนด โดยมีการกำหนดค่าต่ำสุด (x_l) และค่าสูงสุด (x_u) ของช่วง ผลลัพธ์ที่ได้จากการคำนวณสามารถรับประกันได้ว่าผลลัพธ์ที่ถูกต้องต้องปรากฏอยู่ในช่วงที่ได้จากการคำนวณอย่างแน่นอน

กำหนดให้ X เป็นค่าเชิงตัวเลขใดๆ ในระบบแทนช่วง รูปแบบการแทนช่วงของ X ในระบบแทนช่วงเป็นดังนี้

$$X = [x_l, x_u]$$

2.4 การดำเนินการพื้นฐานทางเลขคณิตของระบบแทนช่วง (fundamental arithmetic operations for interval representation system)

กำหนดให้ $X = [x_l, x_u]$ และ $Y = [y_l, y_u]$ เป็นช่วงในระบบแทนช่วง โดยที่การดำเนินการพื้นฐานทางเลขคณิตของระบบแทนช่วงเป็นการคำนวณที่เกิดจากการบวก ลบ คูณ และหารกันของค่าต่ำสุดและค่าสูงสุดของช่วง โดยสามารถศึกษาได้จาก [6-8] ซึ่งสรุปได้ดังนี้

$$X + Y = [x_l + y_l, x_u + y_u]$$

$$X - Y = [x_l - y_u, x_u - y_l]$$

$$X \times Y = [\min(x_l \times y_l, x_l \times y_u, x_u \times y_l, x_u \times y_u), \max(x_l \times y_l, x_l \times y_u, x_u \times y_l, x_u \times y_u)]$$

$$X \div Y = [\min(x_l \div y_l, x_l \div y_u, x_u \div y_l, x_u \div y_u), \max(x_l \div y_l, x_l \div y_u, x_u \div y_l, x_u \div y_u)]$$

สำหรับการหารนั้น ช่วงที่เป็นตัวหารจะต้องไม่ครอบคลุมศูนย์

ตัวอย่างที่ 2.4 *กำหนดให้ $X = [5.25, 7.1]$, $Y = [4.65, 8.2]$ จงแสดงการบวก ลบ คูณ และหาร ของช่วง X และ Y*

วิธีทำ การคำนวณพื้นฐานของตัวดำเนินการบวก ลบ คูณ และหารของช่วง X และ Y สามารถแสดงได้ดังนี้

$$X + Y = [9.9, 15.3]$$

$$X - Y = [-2.95, 2.45]$$

$$\begin{aligned} X \times Y &= [\min(24.41, 43.05, 33.02, 58.22), \\ &\quad \max(24.41, 43.05, 33.02, 58.22)] \\ &= [24.41, 58.22] \end{aligned}$$

$$\begin{aligned} X \div Y &= [\min(1.129, 0.64, 1.527, 0.866), \\ &\quad \max(1.129, 0.64, 1.527, 0.866)] \\ &= [0.64, 1.527] \end{aligned}$$

□

2.5 การแปลงชุดตัวเลข (digit set conversion)

หลักการของการแปลงชุดตัวเลขคือการแปลงจากชุดตัวเลขหนึ่งไปยังอีกชุดตัวเลขหนึ่ง ซึ่งมีเลขฐาน β เดียวกัน สำหรับงานวิจัยในเรื่องนี้สามารถศึกษาเพิ่มเติมได้จาก [9, 10] โดยกำหนดให้ D และ E เป็นชุดตัวเลขจำกัดที่ต่างกัน และ β เป็นเลขฐานที่สามารถเป็นได้ทั้งจำนวนจริงและจำนวนเชิงซ้อน การแปลงชุดตัวเลขในระบบเลขฐาน β จากชุดตัวเลข D ไปยังชุดตัวเลข E สามารถเขียนได้ดังนี้

$$\lambda : D \rightarrow E$$

โดยที่ $X \in D$ และ $\|\lambda(X)\| = \|X\|$

สำหรับปัญหาของการแปลงชุดตัวเลขนั้นได้ถูกนำไปใช้ในการอธิบายการดำเนินการพื้นฐานทางเลขคณิต เช่น การบวกสามารถเทียบได้กับการแปลงชุดตัวเลขหนึ่งไปยังอีกชุดตัวเลขหนึ่งที่มีขนาดแตกต่างกัน แต่มีเลขฐานเดียวกัน ในรูปแบบของ $D = \{d \in \mathrm{Z} \mid 2a \le d \le 2b\}$ และ $E = \{e \in \mathrm{Z} \mid a \le e \le b\}$ เป็นต้น

ตัวอย่างที่ 2.5 *การบวกกันของเลขสองจำนวนในระบบเลขฐานสอง X = 0110110 และ Y = 0100101 โดยวิธีการแปลงชุดตัวเลข*

วิธีทำ *การบวกกันของเลขสองจำนวนบนระบบเลขฐานสอง X = 0110110 และ Y = 0100101 สามารถพิจารณาเป็นการแปลงชุดตัวเลขบนระบบเลขฐานสองจากชุดตัวเลข D = {0, 1, 2} ไปยังชุดตัวเลข E = {0, 1} ได้ โดยการแปลงจาก 0210211 ไปเป็น 1011011 ได้ดังนี้*

$$
\begin{array}{rl}
0\;\;1\;\;1\;\;0\;\;1\;\;1\;\;0\;\; & \quad X \\
\underline{0\;\;1\;\;0\;\;0\;\;1\;\;0\;\;1}+ & \quad Y \\
0\;\;2\;\;1\;\;0\;\;2\;\;1\;\;1\;\; & \quad X+Y \\
\underline{\underline{1\;\;0\;\;1\;\;1\;\;0\;\;1\;\;1}}\;\; &
\end{array}
$$

□

2.5.1 การแปลงแบบขนาน (parallel conversion)

การแปลงชุดตัวเลขแบบขนานนั้นมีข้อดีอย่างหนึ่งคือ ความเร็วในการคำนวณที่สูง เนื่องจากสามารถทำการคำนวณไปพร้อมกันในทุก ๆหลักของตัวเลขที่เป็นข้อมูลนำเข้า โดยเฉพาะในกรณีข้อมูลนำเข้าที่นำมาคำนวณกันอยู่ในรูปแบบที่ไม่มีตัวทด (carry-free) ส่งผลให้เราสามารถหาผลลัพธ์ได้พร้อมกันในทุก ๆหลัก โดยใช้เวลาในการคำนวณคือ O(1) แต่ในกรณีที่ข้อมูลนำเข้าที่นำมาคำนวณกันก่อให้เกิดตัวทดขึ้น ผลลัพธ์ในแต่ละหลักจะไม่สามารถผลิตออกมาได้พร้อมกัน เนื่องจากเกิดผลกระทบจากสายการแพร่ของตัวทด (carry propagation chain) ส่งให้ผลให้ตัวเลขทางซ้ายอาจเกิดความเปลี่ยนแปลงขึ้นได้ ดังนั้นการคำนวณแบบขนานอาจจำเป็นต้องทำการคำนวณมากกว่าหนึ่งครั้ง ทั้งนี้ขึ้นอยู่กับขนาดและรูปแบบของข้อมูลนำเข้า

2.6 สถาปัตยกรรมแบบทำควบคู่กัน (on-the-fly architecture)

สถาปัตยกรรมการแปลงแบบทำควบคู่กันเป็นกระบวนการผลิตตัวเลขที่เป็นข้อมูลนำออกแบบขนาน โดยอาศัยแนวคิดของฟังก์ชันประกอบ (composite function) เข้ามาประยุกต์ใช้ งานวิจัยของเออเสกโกแวค (Ercegovac) และแลง (Lang) [11] ถือเป็นงานวิจัยแรกที่ได้นำวิธีการแปลงชุดตัวเลขโดยใช้สถาปัตยกรรมแบบทำควบคู่กันมาใช้ โดยแปลงจำนวนจากจำนวนซ้ำซ้อนแบบมีเครื่องหมายไปเป็นรูปแบบสัญนิยม (conventional representation) ต่อมาคอร์เนอรับ (Kornerup) [9] ได้เสนอสถาปัตยกรรมแบบทำควบคู่กันเพื่อแปลงชุดตัวเลขซ้ำซ้อนให้เป็นชุดตัวเลขไม่ซ้ำซ้อนบนเลขฐานเดียวกัน โดยชุดตัวเลขซ้ำซ้อนจะต้องเป็นระบบจำนวนซ้ำซ้อนแบบมีเครื่องหมายเพื่อลดการเกิดสายการแพร่ของตัวทด ข้อดีของสถาปัตยกรรมแบบทำควบคู่กันคือการคำนวณแบบขนาน ทำให้การคำนวณมีความเร็วสูง เพราะทุก ๆตัวเลขของข้อมูลนำเข้าสามารถคำนวณไปพร้อม ๆกันได้ ทั้งนี้ตัวเลขที่มีนัยสำคัญสูงสุดของผลลัพธ์ขึ้นอยู่กับตัวเลขที่มีนัยสำคัญต่ำสุดของข้อมูลนำเข้า ดังนั้นเวลาที่ใช้ในการส่งผ่านค่าของตัวทดจากตัวเลขทางขวาสุดไปยังตัวเลขทางซ้ายสุดคือ O(log n) เมื่อ n หมายถึงจำนวนตัวเลขของข้อมูล

นำเข้า ส่วนข้อเสียของสถาปัตยกรรมแบบทำควบคู่กันคือ การสิ้นเปลืองทรัพยากรที่ใช้ในขั้นตอนการแปลงเป็นจำนวนมาก เช่น เรจิสเตอร์ เป็นต้น ทั้งนี้ขึ้นอยู่กับขนาดของข้อมูลนำเข้า ข้อเสียอีกประการหนึ่งคือ เราจะต้องทราบขนาดของข้อมูลนำเข้าก่อน จึงจะทำการแปลงได้

ในการแปลงชุดตัวเลข $D \to E$ โดยที่ $X \in P(\beta, D)$ และ $Y \in P(\beta, E)$ โดย $\|X\| = \|Y\|$ และ $D \neq E$ จะมีนัยสำคัญดังนี้

กำหนดให้ชุดตัวเลข E เป็นชุดตัวเลขไม่ซ้ำซ้อน และชุดตัวเลข C เป็นชุดตัวเลขของตัวทด c เราสามารถเขียนตัวเลข d ใดๆ ที่อยู่ในชุดตัวเลข D ซึ่งเป็นชุดตัวเลขซ้ำซ้อนให้อยู่ในรูปของสมการของการแปลงจาก $P(\beta, D)$ ไปยัง $P(\beta, E)$ คือ $d = c\beta + e$ โดยที่ $e \in E$ และ $c \in C$ แต่ในการคำนวณตามความเป็นจริงเราจะต้องรวมตัวทดที่เข้ามา (incoming carry) เข้ากับ d ก่อน หลังจากนั้นจึงผลิตตัวทดที่ส่งออก (outgoing carry) และ e ออกไป เพราะฉะนั้นจะได้ความสัมพันธ์ของการแปลงระหว่างชุดตัวเลข λ (conversion mapping λ) เป็นฟังก์ชันการแปลงดังต่อไปนี้

$$\lambda : C \times D \to C \times E$$

เมื่อ C เป็นชุดตัวเลขของตัวทด สำหรับบาง (c, d) ใน $C \times D$ มี c' อยู่ใน C และ e อยู่ใน E ซึ่ง

$$\lambda : (c', d) \to (c, e)$$

โดยเราสามารถเขียนเป็นสมการได้ดังนี้

$$c' + d = c\beta + e$$

เมื่อ β เป็นเลขฐาน โดยเราสามารถเรียก c' และ c คือ ตัวทดนำเข้าและตัวทดนำออกตามลำดับในแต่ละตัวเลข d ที่เป็นข้อมูลนำเข้า เช่น ในฐาน $\beta = 2$ และ $D = \{0, 1\}$ กำหนดให้ตัวทดนำเข้า $c' = 1$ และตัวเลข $d = 1$ เราสามารถคำนวณได้ว่าตัวเลข e และตัวทดนำออก c จะมีค่า $e = 0$ และ $c = 1$ เป็นต้น กำหนดให้ D = {0, 1, 2}, E = {0, 1} และ β = 2 ฟังก์ชันการแปลงสามารถแสดงได้ดังตารางที่ 2.1

ตารางที่ 2.1 แสดงฟังก์ชันการแปลงจาก D ไป E บนเลขฐานสอง

λ		D		
		0	1	2
C	0	00	01	10
	1	01	10	11

ผลลัพธ์ในแต่ละคู่ในตารางที่ 2.1 แทนค่า ce ซึ่ง (c, e) อยู่ใน $C \times E$ โดยการคำนวณในตารางจะเริ่มจากค่า c = 0 จะเห็นว่า ce ที่เป็นไปได้คือ 00, 01 และ 10 เมื่อ d คือ 0, 1 และ 2 ตามลำดับ ดังนั้นตัวทดนำออกที่เป็นไปได้คือ 0 และ 1 เราจึงต้องเพิ่มแถวที่ค่า c = 1 และผลลัพธ์ที่ได้คือ 01, 10 และ 11 จากค่า ce ทั้งหมดจะเห็นว่าตัวทดที่เกิดขึ้นคือ 0 และ 1 เท่านั้น ดังนั้นตารางนี้จึงสมบูรณ์แล้ว

จากแนวคิดฟังก์ชันการแปลงชุดตัวเลข เราสามารถเขียนฟังก์ชันของชุดตัวเลขของตัวทดได้เป็น $\{\sigma_d\}, \sigma_d : C \to C$ โดยที่ $d \in D$ เรียกว่า ฟังก์ชันการส่งผ่านตัวทด (carry-transfer function) ซึ่งสามารถเขียนเป็นสมการได้ดังนี้

$$\forall c \in C : \sigma_d(c') = c \text{ โดย } \lambda(c', d) = (c, e)$$

โดยที่ σ_d เป็นฟังก์ชันที่อธิบายเกี่ยวกับการจับคู่ (mapping) ค่าของตัวทดที่เข้ามา c' ไปยังค่าของตัวทดที่ส่งออกไป c โดยผ่านตัวเลข d หนึ่งๆ เท่านั้น ซึ่งเราสามารถเขียนฟังก์ชันการจับคู่นี้ผ่าน d ทุกตัวที่อยู่ใน D ได้ดังนี้

$$\sigma_{d_i d_{i-1} \ldots d_j}(c) = \sigma_{d_i}(\sigma_{d_{i-1}}(\ldots \sigma_{d_j}(c) \ldots))$$

เราเรียกฟังก์ชันนี้ว่าเป็นฟังก์ชันประกอบ และจากฟังก์ชันนี้เองทำให้เราสามารถหาค่า c ใดๆ ได้ โดยกำหนดให้ c_0 เริ่มต้นมีค่าเท่ากับ 0

$$c_i = \sigma_{d_i d_{i-1} \ldots d_j}(0)$$

เมื่อเราสามารถหาฟังก์ชันที่ทำการสร้างตัวทดใดๆ ได้แล้ว ต่อไปเราก็สามารถหาฟังก์ชันในการหา e ในลำดับใดๆได้ด้วย ฟังก์ชันจับคู่ตัวเลข (digit mapping function) $\{\varepsilon_d\}, \varepsilon_d : C \to E$ โดยที่ $d \in D$ ซึ่งเราสามารถเขียนเป็นสมการได้ดังนี้

$$e_i = \varepsilon_{d_i}(\sigma_{d_i d_{i-1} \ldots d_j}(0))$$

ตัวอย่างที่ 2.6 *กำหนดให้ $X = 3\bar{2}112\bar{2}11$ เป็นผลลัพธ์ที่ได้จากการนำจำนวน 3 จำนวนในฐาน β = 3 ซึ่งมีชุดตัวเลขคือ $\{\bar{1},0,1\}$ มาทำการบวกกันแบบขนาน จงทำการแปลงชุดตัวเลขด้วยสถาปัตยกรรมแบบควบคู่กันโดยที่ $D = \{\bar{3}, \bar{2}, \bar{1}, 0, 1, 2, 3\}$ และ $E = \{\bar{1}, 0, 1\}$*

วิธีทำ กำหนดให้ $X = x_8 x_6 \ldots x_1$ เป็นข้อมูลนำเข้า โดยที่ $x_i \in D$ เมื่อ i เป็นจำนวนเต็มบวก ฟังก์ชันการแปลงจะทำการผลิตคู่ผลลัพธ์ได้แก่ ตัวทดนำออกและตัวเลขที่เป็นผลลัพธ์ดังสมการต่อไปนี้

$$c' + d = c\beta + e$$

สามารถเขียนสมการใหม่ได้เป็น

$$c_{i-1} + x_i = c_i \beta + y_i$$

โดยที่มีตัวทดนำเข้าเริ่มต้น $c_0 =$ 0 และ $y_i \in E$ ผลลัพธ์ที่ได้คือ $Y = \bar{1}11111111$ เมื่อ $\|X\| = \|Y\|$ ทั้งนี้การแปลงนั้นจะแปลงจาก $D = \{\bar{3}, \bar{2}, \bar{1}, 0, 1, 2, 3\}$ ไปเป็น $E = \{\bar{1}, 0, 1\}$ โดยมี β = 3 และ $C = \{\bar{1}, 0, 1\}$ เราสามารถหาค่า c และ e ในแต่ละหลักได้ดังนี้

ค่า c จากฟังก์ชันประกอบสามารถแสดงได้คือ

$$c_0 = 0$$
$$c_1 = \sigma_1(c_0) = \sigma_1(0) = 0$$
$$c_2 = \sigma_2\sigma_1(0) = \sigma_2(0) = 0$$
$$c_3 = \sigma_3\sigma_2\sigma_1(0) = \sigma_3(0) = \bar{1}$$
$$c_4 = \sigma_4\sigma_3\sigma_2\sigma_1(0) = \sigma_4(\bar{1}) = 0$$
$$c_5 = \sigma_5\sigma_4\sigma_3\sigma_2\sigma_1(0) = \sigma_5(0) = 0$$
$$c_6 = \sigma_6\sigma_5\sigma_4\sigma_3\sigma_2\sigma_1(0) = \sigma_6(0) = 0$$
$$c_7 = \sigma_7\sigma_6\sigma_5\sigma_4\sigma_3\sigma_2\sigma_1(0) = \sigma_7(0) = \bar{1}$$
$$c_8 = \sigma_8\sigma_7\sigma_6\sigma_5\sigma_4\sigma_3\sigma_2\sigma_1(0) = \sigma_8(\bar{1}) = 1$$

ค่า e จากฟังก์ชันจับคู่ตัวเลข

$$e_1 = \varepsilon_1(c_0) = \varepsilon_1(0) = 1 = y_1$$
$$e_2 = \varepsilon_2(c_1) = \varepsilon_2(0) = 1 = y_2$$
$$e_3 = \varepsilon_3(c_2) = \varepsilon_3(0) = 1 = y_3$$
$$e_4 = \varepsilon_4(c_3) = \varepsilon_4(\bar{1}) = 1 = y_4$$
$$e_5 = \varepsilon_5(c_4) = \varepsilon_5(0) = 1 = y_5$$
$$e_6 = \varepsilon_6(c_5) = \varepsilon_6(0) = 1 = y_6$$
$$e_7 = \varepsilon_7(c_6) = \varepsilon_7(0) = 1 = y_7$$
$$e_8 = \varepsilon_8(c_7) = \varepsilon_8(\bar{1}) = \bar{1} = y_8$$
$$e_9 = \varepsilon_9(c_8) = \varepsilon_9(1) = 1 = y_9$$

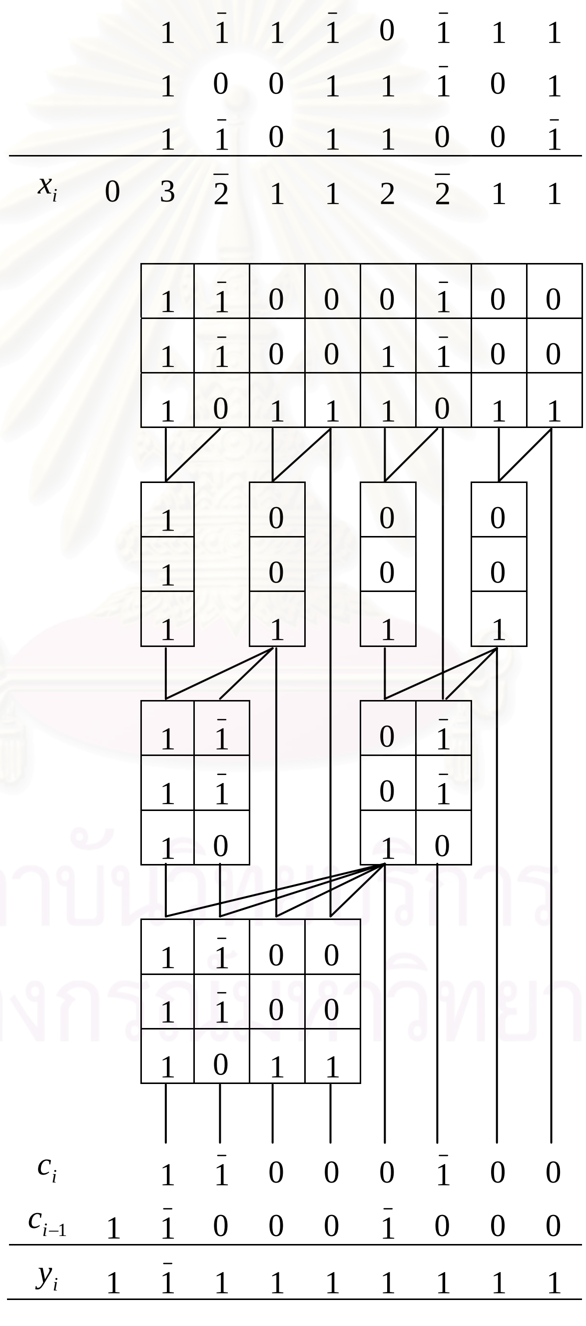
เพราะฉะนั้น $X = 3\bar{2}1122\bar{2}11$ โดยที่ $x_i \in D$ จะถูกแปลงได้เป็น $Y = \bar{1}11111111$ โดยที่ $y_i \in E$ สำหรับฟังก์ชันการแปลงชุดตัวเลขจาก $D = \{\bar{3}, \bar{2}, \bar{1}, 0, 1, 2, 3\}$ ไปยัง $E = \{\bar{1}, 0, 1\}$ ในฐาน $\beta = 3$ สามารถแสดงได้ดังตารางที่ 2.2

ตารางที่ 2.2 แสดงฟังก์ชันการแปลงจาก $D = \{\bar{3}, \bar{2}, \bar{1}, 0, 1, 2, 3\}$ ไปยัง $E = \{\bar{1}, 0, 1\}$ ในฐาน $\beta = 3$

λ		D						
		$\bar{3}$	$\bar{2}$	$\bar{1}$	0	1	2	3
C	$\bar{1}$	$\bar{1}\bar{1}$	$\bar{1}0$	$\bar{1}1$	$0\bar{1}$	00	01	$1\bar{1}$
	0	$\bar{1}0$	$\bar{1}1$	$0\bar{1}$	00	01	$1\bar{1}$	10
	1	$\bar{1}1$	$0\bar{1}$	00	01	$1\bar{1}$	10	11

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

กระบวนการของการแปลงชุดตัวเลขโดยใช้สถาปัตยกรรมแบบทำควบคู่กันนั้น จะเริ่มต้นโดยการคำนวณในรอบที่หนึ่งนั้น จะนำกรณีที่เป็นไปได้ของตัวทดนำเข้า ได้แก่ $\bar{1}$, 0 และ 1 มาทำการคำนวณกับค่าของ x_i ทั้ง 8 ตัว ซึ่งจะได้ผลลัพธ์คือตัวทดนำออกของแต่ละ x_i หลังจากนั้นทำการจับคู่ทั้ง 8 หลักมาคำนวณหาผลลัพธ์ในรอบที่สอง โดยอาศัยแนวคิดของฟังก์ชันประกอบ ทำการจับคู่ผลลัพธ์ในรอบที่สอง เพื่อคำนวณหาผลลัพธ์ในรอบที่สาม ทำซ้ำกระบวนการเดิมจนถึงรอบที่สี่ สำหรับกระบวนการของการแปลงชุดตัวเลขโดยใช้สถาปัตยกรรมแบบทำควบคู่กันนั้น สามารถแสดงได้ดังรูปที่ 2.1



รูปที่ 2.1 แสดงตัวอย่างการแปลงชุดตัวเลขโดยใช้สถาปัตยกรรมแบบทำควบคู่กัน

จากรูปที่ 2.1 จะเห็นได้ว่าผลลัพธ์ของ y_i ในหลักใดๆ จะได้จากการพิจารณาตัวทดนำเข้า c_{i-1} และ x_i โดยเริ่มจากหลักที่ $x_1 = 1$ จะกำหนดให้ตัวทดนำเข้า $c_0 = 0$ ซึ่งจะได้ผลลัพธ์ $y_1 = 1$ และ $c_1 = 0$ หลังจากนั้นในหลักที่ $x_2 = 1$ และตัวทดนำเข้า $c_1 = 0$ จะได้ผลลัพธ์ $y_2 = 1$ และ $c_2 = 0$ วนซ้ำกระบวนการนี้ไปจนถึง y_9 □

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

บทที่ 3

ระบบแทนช่วงแบบยืดหยุ่น

จุดประสงค์หลักของงานวิจัยนี้ คือ การนำเสนอรูปแบบการแทนจำนวนแบบช่วงแบบใหม่ ที่มีชื่อเรียกว่า ระบบแทนช่วงแบบยืดหยุ่น (flexible interval representation system, FIRS) โดยที่ระบบการแทนจำนวนแบบใหม่นี้สามารถลดเนื้อที่ที่ใช้ในการแสดงช่วงให้น้อยลงได้ เนื่องจากระบบใหม่นี้ใช้จำนวนเพียงจำนวนเดียวในการแสดงช่วง

สำหรับเนื้อหาในบทนี้จะกล่าวถึงแรงจูงใจในการเสนอรูปแบบการแทนจำนวนแบบใหม่ พร้อมทั้งเสนอระบบแทนช่วงแบบยืดหยุ่น ความสมบูรณ์ของระบบแทนช่วงแบบยืดหยุ่น การดำเนินการพื้นฐานทางเลขคณิต (บวก ลบ คูณ และหาร) สำหรับระบบแทนช่วงแบบยืดหยุ่น

3.1 บทกล่าวนำ

ระบบแทนช่วงแบบยืดหยุ่นเป็นระบบจำนวนที่ถูกพัฒนามาจากระบบแทนช่วง ระบบแทนช่วงเป็นระบบที่ถูกเสนอขึ้นมาเพื่อแก้ปัญหาที่เกิดขึ้นในระหว่างการคำนวณทางคณิตศาสตร์ ซึ่งในบางครั้งอาจมีความจำเป็นต้องปัดเศษทิ้งไปในบางตำแหน่งเพื่อลดเวลาที่ใช้ในการคำนวณ หรือในบางกรณีที่ระบบไม่สามารถรองรับการคำนวณให้ครบทุกตัวเลขได้ ทำให้มีความจำเป็นต้องปัดเศษทิ้งในบางตำแหน่งเพื่อให้ระบบสามารถทำงานต่อไปได้ อีกทั้งความผิดพลาดของข้อมูลอาจเกิดขึ้นได้จากความคลาดเคลื่อนของอุปกรณ์การวัด หรือจากข้อมูลนำเข้า ซึ่งปัญหาต่างๆเหล่านี้ ส่งผลให้คำตอบที่ได้จากการคำนวณมีความผิดพลาดคลาดเคลื่อนไปได้ ดังนั้นวิธีการแก้ปัญหาในเรื่องนี้คือการแทนข้อมูลที่ได้มาด้วยช่วง ทำให้สามารถรับประกันได้ว่าคำตอบที่ถูกต้องจะต้องปรากฏอยู่ในช่วงที่ได้จากการคำนวณอย่างแน่นอน แต่ระบบแทนช่วงประสบปัญหาทางด้านความสิ้นเปลืองเนื้อที่และความล่าช้าในการคำนวณ

จากที่ได้กล่าวมาข้างต้น ทำให้ผู้วิจัยให้ความสนใจกับการนำระบบแทนช่วงมาประยุกต์ใช้กับระบบจำนวนแบบมีเครื่องหมาย ทำให้ระบบแทนช่วงแบบใหม่นี้สามารถลดความสิ้นเปลืองของเนื้อที่ให้น้อยลง 25 เปอร์เซ็นต์เมื่อเปรียบเทียบกับระบบแทนช่วงแบบมีเครื่องหมายดั้งเดิม (classical signed digit interval representation system) ระบบแทนช่วงที่ได้นำเสนอขึ้นมาใหม่นี้มีชื่อเรียกว่า ระบบแทนช่วงแบบยืดหยุ่น โดยที่ระบบนี้สามารถแสดงช่วงได้โดยใช้จำนวนเพียงจำนวนเดียว พร้อมทั้งเสนอการดำเนินการพื้นฐานทางเลขคณิต ได้แก่ บวก ลบ คูณ และหาร สำหรับระบบแทนช่วงแบบยืดหยุ่นนี้ด้วย

3.2 ระบบแทนช่วงแบบยืดหยุ่น (Flexible interval representation system)

ลักษณะของระบบแทนช่วงแบบยืดหยุ่นและการแสดงค่าต่ำสุดหรือค่าสูงสุดของช่วงโดยใช้จำนวนเพียงจำนวนเดียวในระบบแทนช่วงแบบยืดหยุ่นเป็นดังนิยามต่อไปนี้

นิยามที่ 3.1 ระบบแทนช่วงแบบยืดหยุ่น*ประกอบด้วยฐาน* β *และชุดตัวเลข* D *โดยที่* $\beta = 2$ *และ* $D = \{\bar{1}, \gamma, 0, \alpha, 1\}$ *ซึ่ง* α *และ* γ *เรียกว่า* ตัวเลขยืดหยุ่น (flexible digit) *โดยที่* α *แทนตัวเลข 0 หรือ 1 และ* γ *แทนตัวเลข* $\bar{1}$ *หรือ 0*

นิยามที่ 3.2 *รูปแบบแทนช่วง* $X = x_1 x_2 x_3 \ldots x_n$ *ใน*ระบบแทนช่วงแบบยืดหยุ่น *สามารถแสดงช่วง [A, B] ได้โดย*

$$\text{A} = \sum_{i=0}^{n} \min(x_i) \times 2^i \quad \text{และ} \quad \text{B} = \sum_{i=0}^{n} \max(x_i) \times 2^i$$

สมบัติที่สำคัญของระบบแทนช่วงแบบยืดหยุ่นคือ ระบบนี้สามารถแสดงช่วงได้โดยใช้จำนวนเพียงจำนวนเดียว ดังตัวอย่างในตารางที่ 3.1

ตารางที่ 3.1 แสดงตัวอย่างของระบบจำนวนแบบยืดหยุ่น

ช่วง	รูปแบบแทนช่วงแบบยืดหยุ่น
[3, 8]	$1\gamma0\gamma$
[-10, 15]	$\alpha\gamma0\bar{1}\alpha$
[-1.25, 0.75]	$\alpha\bar{1}.0\bar{1}$
[123, 123]	1111011

เนื่องจากการนำเสนอระบบจำนวนแบบใหม่นั้นต้องคำนึงถึงความสมบูรณ์ (completeness) ของระบบ ซึ่งความสมบูรณ์ของระบบแทนช่วงแบบยืดหยุ่นสามารถแสดงได้ดังทฤษฎีบทที่ 3.1

ทฤษฎีบทที่ 3.1 *สำหรับจำนวนจริง x และ y ใดๆ ช่วง [x, y] สามารถมีรูปแบบแทนช่วง (interval representation) ในระบบแทนช่วงแบบยืดหยุ่นได้*

พิสูจน์

วิธีการพิสูจน์ทฤษฎีบทนี้ จะเสนออัลกอริทึมสำหรับแปลง (conversion) ช่วงใดๆที่อยู่ในรูปของรูปแบบแทนช่วงแบบมีเครื่องหมายดั้งเดิมให้อยู่ในรูปของรูปแบบแทนช่วงแบบยืดหยุ่น

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

อัลกอริทึม : การแปลง

input interval [*A*, *B*]

A = $a_0a_1...a_n$ where $a_i \in \{\bar{1},0,1\}$

B = $b_0b_1...b_n$ where $b_i \in \{\bar{1},0,1\}$

output *S* = $s_0s_1...s_n$ where $s_i \in \{\bar{1},\gamma,0,\alpha,1\}$

begin

C = *B* – *A* where *C* is binary number

$i \leftarrow 0$

while (not-end-of-data) **do**

if $c_i = 1$ **then** $c_i \leftarrow \alpha$

$i \leftarrow i+1$

endif

enddo

$S \leftarrow C + A$

end

พิสูจน์อัลกอริทึม

เนื่องจากขนาดของช่วงซึ่งคำนวณได้จากการนำค่าสูงสุดมาลบด้วยค่าต่ำสุดนั้นสามารถแสดงได้โดย *C* ที่เป็นจำนวนไม่เป็นลบ ดังนั้นช่วง [0, *C*] ใดๆ สามารถแสดงได้โดยใช้ลำดับของตัวเลข 0 และ α เพียงลำดับเดียวได้ เพราะฉะนั้นรูปแบบแทนช่วง [*A*, *B*] ใดๆ สามารถแสดงได้โดยการนำค่าต่ำสุดของช่วง (*A*) มาทำการบวกเข้ากับขนาดของช่วง (*C*) ■

ตัวอย่างที่ 3.1 *หารูปแบบแทนช่วง [1, 5] ในระบบแทนช่วงแบบยืดหยุ่น*

วิธีทำ ขั้นตอนการหารูปแบบแทนช่วงจะแสดงในตารางที่ 3.2

ตารางที่ 3.2 แสดงขั้นตอนการแปลงช่วง [1, 5] ให้มาอยู่ในระบบแทนช่วงแบบยืดหยุ่น

ขั้นตอนการแปลง	ผลลัพธ์
1. ช่วง [1, 5]	$[0001, 0101]$
2. รูปแบบแทนจำนวนของค่าต่ำสุดของช่วง	0001
3. รูปแบบแทนช่วงของขนาดของช่วง	$0\alpha00$
4. รูปแบบแทนช่วงของ [1, 5] แสดงได้โดยการนำผลลัพธ์ของขั้นตอนที่ 2 และ 3 มาบวกกัน	$0\alpha01$

□

3.3 การดำเนินการพื้นฐานทางเลขคณิตของระบบแทนช่วงแบบยืดหยุ่น

การคำนวณทางคณิตศาสตร์ของงานวิจัยนี้ จะมุ่งเน้นไปที่การดำเนินการพื้นฐานทางเลขคณิต ซึ่งประกอบไปด้วยการบวก ลบ คูณ และหาร เนื่องจากระบบแทนช่วงแบบยืดหยุ่นได้นำความรู้เรื่องเลขคณิตของระบบแทนช่วงมาประยุกต์ใช้ ดังนั้นจึงไม่สามารถทำการคำนวณแบบทั่วไปได้ แต่ต้องนำความรู้เรื่องเลขคณิตของระบบแทนช่วงมาประยุกต์ใช้กับระบบแทนช่วงแบบใหม่นี้ด้วย [12-15]

3.3.1 การบวกในระบบแทนช่วงแบบยืดหยุ่น

เราได้เสนอกฎการบวกของตัวเลข 2 ตัวใดๆ ในระบบแทนช่วงแบบยืดหยุ่น ซึ่งแสดงดังสมการต่อไปนี้

$$x + y = 2c + z$$

เมื่อ $x, y, z, c \in \{\bar{1}, \gamma, 0, \alpha, 1\}$ ซึ่ง x และ y คือตัวเลข 2 ตัวใดๆ ที่เป็นข้อมูลนำเข้า โดยที่ z คือตัวเลขที่เป็นผลลัพธ์ของการบวก และ c คือตัวทดที่เกิดขึ้น ซึ่งจะทดไปยังหลักที่ทางซ้าย สำหรับกฎการบวกของตัวเลข 2 ตัวใดๆ ในระบบแทนช่วงแบบยืดหยุ่น สามารถแสดงได้ในตารางที่ 3.3

ตารางที่ 3.3 แสดงกฎการบวกในระบบแทนช่วงแบบยืดหยุ่น

+		x				
		$\bar{1}$	γ	0	α	1
y	$\bar{1}$	$\bar{1}0$	$\bar{1}\alpha$	$0\bar{1}$	0γ	00
	γ	$\bar{1}\alpha$	$\gamma 0$	0γ	$\gamma 1, \alpha\bar{1}$	0α
	0	$0\bar{1}$	0γ	00	0α	01
	α	0γ	$\gamma 1, \alpha\bar{1}$	0α	$\alpha 0$	1γ
	1	00	0α	01	1γ	10

ผลลัพธ์ของแต่ละคู่ในตารางที่ 3.3 แทนค่า cz เช่น กำหนดให้ $x = 1$ และ $y = \alpha$ การบวกของ x และ y จะได้ตัวทด $c = 1$ และผลลัพธ์คือ $z = \gamma$ เป็นต้น แต่กฎการบวกที่ผ่านมายังไม่สมบูรณ์เนื่องจากยังไม่ได้พิจารณาตัวทดที่เข้ามาด้วย ดังนั้นเราสามารถเขียนสมการใหม่ได้ดังต่อไปนี้

$$c' + x + y = 2c + z$$

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

เมื่อ $c' \in \{\bar{1}, \gamma, 0, \alpha, 1\}$ โดยเรียก c' และ c คือตัวทดนำเข้าและตัวทดนำออก สำหรับกฎการบวกที่พิจารณาตัวทดนำเข้าด้วยนั้น หรือเรียกอีกอย่างหนึ่งว่า กฎการบวกสมบูรณ์ (complete addition rules) จะแสดงไว้ในตารางที่ 3.4

ตารางที่ 3.4 แสดงกฎการบวกสมบูรณ์ในระบบแทนช่วงแบบยืดหยุ่น

+		ตัวทด (c')				
		$\bar{1}$	γ	0	α	1
ผลบวก (x + y)	00	$0\bar{1}$	0γ	00	0α	01
	01	00	0α	01	1γ	10
	10	$1\bar{1}$	1γ	10	1α	11
	$0\bar{1}$	$\bar{1}0$	$\bar{1}\alpha$	$0\bar{1}$	0γ	00
	$\bar{1}0$	$\bar{1}\bar{1}$	$\bar{1}\gamma$	$\bar{1}0$	$\bar{1}\alpha$	$\bar{1}1$
	0α	0γ	$\gamma 1$	0α	$\alpha 0$	1γ
	$\alpha 0$	$\alpha\bar{1}$	$\alpha\gamma$	$\alpha 0$	$\alpha\alpha$	$\alpha 1$
	0γ	$\bar{1}\alpha$	$\gamma 0$	0γ	$\gamma 1$	0α
	$\gamma 0$	$\gamma\bar{1}$	$\gamma\gamma$	$\gamma 0$	$\gamma\alpha$	$\gamma 1$
	1γ	0α	$\alpha 0$	1γ	$\alpha 1$	1α
	$\gamma 1$	$\gamma 0$	$\gamma\alpha$	$\gamma 1$	$\alpha\gamma$	$\alpha 0$
	$\bar{1}\alpha$	$\bar{1}\gamma$	$\gamma\bar{1}$	$\bar{1}\alpha$	$\gamma 0$	0γ
	$\alpha\bar{1}$	$\gamma 0$	$\gamma\alpha$	$\alpha\bar{1}$	$\alpha\gamma$	$\alpha 0$

ผลลัพธ์ของแต่ละคู่ในตารางที่ 3.4 แทนค่า cz เช่น กำหนดให้ผลบวกของ x และ y คือ 1γ เมื่อนำ 1γ มาบวกกับตัวทดนำเข้า $c' = \alpha$ จะได้ตัวทดนำออก $c = \alpha$ และผลลัพธ์คือ $z = 1$ เป็นต้น จากค่า cz ทั้งหมดจะเห็นว่าตัวทดที่เป็นไปได้ทั้งหมด คือ $\bar{1}, \gamma, 0, \alpha$ และ 1 เท่านั้น ดังนั้นตารางนี้จึงสมบูรณ์แล้ว

สำหรับกระบวนการบวกในระบบแทนช่วงแบบยืดหยุ่น สามารถทำได้ด้วยการใช้อัลกอริทึมการบวกและกฎการบวกในระบบแทนช่วงแบบยืดหยุ่น ดังทฤษฎีบทที่ 3.2 ซึ่งวิธีการพิสูจน์ทฤษฎีบทนั้นสามารถทำได้โดยใช้บทตั้งที่ 3.1 และ 3.2

บทตั้งที่ 3.1 *รูปแบบแทนช่วงในระบบแทนช่วงแบบยืดหยุ่นที่ประกอบด้วยตัวเลขสองตัวสามารถแสดงช่วงได้โดยมีค่าอยู่ระหว่าง [-3, 3]*

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

พิสูจน์

กำหนดให้ รูปแบบแทนช่วงที่ประกอบด้วยตัวเลขสองตัวคือ $X = x_1x_2$ โดยที่ x_1 และ x_2 เป็นตัวเลขในระบบแทนช่วงแบบยืดหยุ่น

การพิสูจน์จะแบ่งออกเป็น 2 กรณี คือ หาค่าต่ำสุดและค่าสูงสุดที่เป็นไปได้ดังนี้
ค่าต่ำสุดที่รูปแบบแทนช่วง X สามารถแสดงค่าได้นั้น สามารถหาได้โดยที่ค่าของ x_1 และ x_2 ต้องมีค่าน้อยที่สุดนั่นคือ -1 ดังสมการที่ 3.1

$$\begin{aligned}(2\times\min(x_1))+x_2 &= 2(-1)+(-1)\\ &=-3\end{aligned} \qquad (3.1)$$

ค่าสูงสุดที่รูปแบบแทนช่วง X สามารถแสดงค่าได้นั้น สามารถหาได้โดยที่ค่าของ x_1 และ x_2 ต้องมีค่ามากที่สุดนั่นคือ 1 ดังสมการที่ 3.2

$$\begin{aligned}(2\times\max(x_1))+x_2 &= 2(1)+(1)\\ &=3\end{aligned} \qquad (3.2)$$

จากสมการที่ 3.1 และ 3.2 เราสามารถสรุปได้ว่า รูปแบบแทนช่วง X ใดๆ สามารถแสดงช่วงได้โดยมีค่าอยู่ระหว่าง [-3, 3] ■

บทตั้งที่ 3.2 *รูปแบบแทนช่วงในระบบแทนช่วงแบบยืดหยุ่นที่ประกอบด้วยตัวเลขสองตัว จะมีขนาดของช่วงไม่เกิน 3 ซึ่งขนาดของช่วงสามารถหาได้จากการนำค่าสูงสุดมาลบด้วยค่าต่ำสุด*

พิสูจน์

กำหนดให้ $|a|$ แทนขนาดของช่วง a
กำหนดให้ รูปแบบแทนช่วงที่ประกอบด้วยตัวเลขสองตัวคือ $X = x_1x_2$ โดยที่ x_1 และ x_2 เป็นตัวเลขในระบบแทนช่วงแบบยืดหยุ่น จะได้ว่า $|x_1| \le 1$ และ $|x_2| \le 1$

สมมติให้ $|X| > 3$ เพราะฉะนั้น $(2\times|x_1|)+|x_2| > 3$ เราสามารถสรุปได้ว่า $|x_1| > 1$ หรือ $|x_2| > 1$ ซึ่งเกิดข้อขัดแย้ง ดังนั้นขนาดของรูปแบบแทนช่วง X จะมีค่าไม่เกิน 3 ■

ทฤษฎีบทที่ 3.2 *การบวกในระบบแทนช่วงแบบยืดหยุ่นสามารถคำนวณได้*

พิสูจน์

วิธีการพิสูจน์ทฤษฎีบทนี้ จะเสนออัลกอริทึมการบวกและกฎการบวกสำหรับระบบแทนช่วงแบบยืดหยุ่น

อัลกอริทึม : การบวก

input FIR *X*, *Y*

$X = x_n x_{n-1} \ldots x_1$ where $x_i \in \{\bar{1},\ \gamma,\ 0,\ \alpha,\ 1\}$

$Y = y_n y_{n-1} \ldots y_1$ where $y_i \in \{\bar{1},\ \gamma,\ 0,\ \alpha,\ 1\}$

output FIR *Z*

$Z = z_n z_{n-1} \ldots z_1$ where $z_i \in \{\bar{1},\ \gamma,\ 0,\ \alpha,\ 1\}$

begin

$i \leftarrow 1$

$c_0 \leftarrow 0$

while (not-end-of-data) **do**

$2c_i + z_i \leftarrow x_i + y_i + c_{i-1}$ where $c_i, c_{i-1} \in \{\bar{1},\ \gamma,\ 0,\ \alpha,\ 1\}$

$i \leftarrow i+1$

enddo

end

พิสูจน์อัลกอริทึม

วิธีการพิสูจน์อัลกอริทึมการบวกนั้น เราจะพิจารณาว่าการบวกกันของตัวเลข x_i และ y_i ที่เป็นข้อมูลนำเข้าโดยมีการพิจารณาตัวทดนำเข้า c_{i-1} ด้วยนั้น จะให้ผลลัพธ์ $c_i z_i$ ซึ่งมีขนาดของช่วงมีค่าไม่เกิน 3 และ $c_i z_i$ มีค่าที่เป็นไปได้อยู่ในช่วงระหว่าง [-3, 3] โดยการพิสูจน์นั้นจะอิงตามสมการที่ 3.3

$$c_{i-1} + x_i + y_i = 2c_i + z_i \tag{3.3}$$

จากบทตั้งที่ 3.1 และ 3.2 เราสามารถพิสูจน์ได้ว่า การบวกกันของตัวเลข x_i และ y_i ที่เป็นข้อมูลนำเข้าโดยมีการพิจารณาตัวทดนำเข้า c_{i-1} ด้วยนั้น สามารถผลิตผลลัพธ์โดยใช้จำนวนตัวเลขเพียง 2 ตัวคือ $c_i z_i$ ที่สามารถเขียนให้อยู่ในระบบแทนช่วงแบบยืดหยุ่นได้เสมอ ดังนั้นอัลกอริทึมการบวกนี้ถูกต้อง ■

ตัวอย่างที่ 3.2 *การบวกของช่วง [1, 5] กับ [3, 6] ในระบบแทนช่วงแบบยืดหยุ่น*

วิธีทำ ขั้นตอนการบวกนั้น เริ่มจากการแปลงช่วง [1, 5] และ [3, 6] ให้อยู่ในระบบแทนช่วงแบบยืดหยุ่น นั่นคือ $0\alpha01$ และ $01\alpha\gamma$ ตามลำดับ การบวกของ $0\alpha01$ และ $01\alpha\gamma$ สามารถแสดงได้โดยใช้กฎการบวกสมบูรณ์ ดังตารางที่ 3.4

$$
\begin{array}{lcl}
0\ \alpha\ \ 0\ 1\ _{+} & \leftrightarrow & [1,5] \\
\underline{0\ 1\ \alpha\ \ \gamma} & \leftrightarrow & [3,6] \\
\underline{\underline{1\ \gamma\ \ \alpha\ \ \alpha}} & \leftrightarrow & [4,11]
\end{array}
$$

ผลลัพธ์ของการบวกในระบบแทนช่วงแบบยืดหยุ่น สามารถให้คำตอบที่ถูกต้อง □

3.3.2 การลบในระบบแทนช่วงแบบยืดหยุ่น

การลบในระบบแทนช่วงแบบยืดหยุ่นนั้นสามารถคำนวณได้โดยการนำช่วงที่เป็นตัวตั้งมาทำการบวกกับรูปแบบแทนลบ (negative representation) ของตัวที่นำมาลบ ซึ่งรูปแบบแทนลบสามารถหาได้จากการเปลี่ยนตัวเลข $\bar{1}$ เป็น 1, γ เป็น α, α เป็น γ, และ 1 เป็น $\bar{1}$

กระบวนการลบในระบบแทนช่วงแบบยืดหยุ่น สามารถทำได้ด้วยการใช้อัลกอริทึมการลบและกฎการบวกในระบบแทนช่วงแบบยืดหยุ่น ดังทฤษฎีบทที่ 3.3

ทฤษฎีบทที่ 3.3 *การลบในระบบแทนช่วงแบบยืดหยุ่นสามารถคำนวณได้*

พิสูจน์

วิธีการพิสูจน์ทฤษฎีบทนี้ จะเสนออัลกอริทึมการลบสำหรับระบบแทนช่วงแบบยืดหยุ่น

อัลกอริทึม : การลบ

input FIR *X*, *Y*

X = $x_n x_{n-1} \ldots x_1$ where $x_i \in \{\bar{1}, \gamma, 0, \alpha, 1\}$

Y = $y_n y_{n-1} \ldots y_1$ where $y_i \in \{\bar{1}, \gamma, 0, \alpha, 1\}$

output FIR *Z*

Z = $z_n z_{n-1} \ldots z_1$ where $z_i \in \{\bar{1}, \gamma, 0, \alpha, 1\}$

begin

$i \leftarrow 1$

while (not-end-of-data) **do**

if $y_i = \bar{1}$ **then** $y_i \leftarrow 1$

else if $y_i = 1$ **then** $y_i \leftarrow \bar{1}$

else if $y_i \leftarrow \alpha$ **then** $y_i \leftarrow \gamma$

else if $y_i \leftarrow \gamma$ **then** $y_i \leftarrow \alpha$

endif

$i \leftarrow i+1$

enddo

$i \leftarrow 1$

$c_0 \leftarrow 0$

while (not-end-of-data) **do**

  $2c_i + z_i \leftarrow x_i + y_i + c_{i-1}$ where $c_i, c_{i-1} \in \{\bar{1},\ \gamma,\ 0,\ \alpha,\ 1\}$

  $i \leftarrow i+1$

enddo

end

พิสูจน์อัลกอริทึม

วิธีการพิสูจน์อัลกอริทึมการลบนั้น จะทำการพิสูจน์เช่นเดียวกับอัลกอริทึมการบวก ■

ตัวอย่างที่ 3.3 *การลบของช่วง [2, 5] โดย [3, 7] ในระบบแทนช่วงแบบยืดหยุ่น*

วิธีทำ รูปแบบแทนช่วงของ [2, 5] และ [3, 7] ในระบบแทนช่วงแบบยืดหยุ่นคือ $01\gamma\alpha$ และ $0\alpha11$ ตามลำดับ รูปแบบแทนลบของตัวที่จะนำมาลบได้จากการแปลง $0\alpha11$ เป็น $0\gamma\bar{1}\bar{1}$ การบวกของ $01\gamma\alpha$ และ $0\gamma\bar{1}\bar{1}$ สามารถแสดงได้โดยใช้กฎการบวกสมบูรณ์ซึ่งแสดงในตารางที่ 3.4

$$
\begin{array}{lcc}
0\ 1\ \gamma\ \alpha & \leftrightarrow & [2,5] \\
\underline{0\ \gamma\ \bar{1}\ \bar{1}} \quad + & \leftrightarrow & [-7,-3] \\
\underline{\underline{0\ \gamma\ \alpha\ \gamma}} & \leftrightarrow & [-5,2]
\end{array}
$$

การลบในระบบแทนช่วงแบบยืดหยุ่นสามารถให้ผลลัพธ์ที่ถูกต้อง □

3.3.3 การคูณในระบบแทนช่วงแบบยืดหยุ่น

เราได้เสนออัลกอริทึมการคูณในระบบแทนช่วงแบบยืดหยุ่น ซึ่งได้มีการเพิ่มตัวเลขชั่วคราวที่ใช้ในระหว่างการคูณขึ้นมา 2 ตัว คือ δ และ ω โดยที่ δ สามารถแทนได้ทั้งตัวเลข 0 (ค่าต่ำสุด) หรือ $\bar{1}$ (ค่าสูงสุด) และ ω แทนตัวเลข 1 (ค่าต่ำสุด) หรือ 0 (ค่าสูงสุด)

ก่อนที่เราจะเสนออัลกอริทึมการคูณนั้น จะขอเสนออัลกอริทึมสำหรับการตรวจสอบเครื่องหมายของช่วงว่าเป็นช่วงบวกหรือลบ เพื่อกำหนดรูปแบบของตัวตั้งและตัวคูณที่เหมาะสมที่พร้อมจะนำไปเริ่มกระบวนการคูณในอัลกอริทึมการคูณ ซึ่งเราเรียกรูปแบบนี้ว่า รูปแบบพร้อมคูณ (multiplication-ready pattern) โดยอัลกอริทึมนี้จะถูกใช้ร่วมกับอัลกอริทึมการคูณและการหารด้วย

สถาบันวิทยบริการ จุฬาลงกรณ์มหาวิทยาลัย

อัลกอริทึม: การตรวจสอบเครื่องหมายของช่วง

input FIR *A*

$A = a_n a_{n-1} ... a_1$ where $a_i \in \{\bar{1},\ \gamma,\ 0,\ \alpha,\ 1\}$

output *min_A*, *max_A*

begin

min_A $\leftarrow$ null, *max_A* $\leftarrow$ null, $i \leftarrow n$

while *i* $\geq$ 0 **and** (*min_A* = null **or** *max_A* = null) **do**

if $a_i = 1$

if *min_A* = null **then** *min_A* $\leftarrow$ pos **endif**

if *max_A* = null **then** *max_A* $\leftarrow$ pos **endif**

else if $a_i = \bar{1}$

if *min_A* = null **then** *min_A* $\leftarrow$ neg **endif**

if *max_A* = null **then** *max_A* $\leftarrow$ neg **endif**

else if $a_i = \alpha$

if *max_A* = null **then** *max_A* $\leftarrow$ pos **endif**

else if $a_i = \gamma$

if *min_A* = null **then** min_A $\leftarrow$ neg **endif**

endif

$i \leftarrow i - 1$

enddo

if *min_A* = null **then** *min_A* $\leftarrow$ pos **endif**

if *max_A* = null **then** *max_A* $\leftarrow$ pos **endif**

end

หลังจากที่เราทราบเครื่องหมายของช่วงที่เป็นตัวตั้งและตัวตัวคูณแล้ว เราจะขอเสนอกฎการคูณเพื่อใช้ร่วมกับอัลกอรึมการคูณในระบบแทนช่วงแบบยืดหยุ่น ซึ่งกฎการคูณสามารถสร้างได้ดังนี้

เริ่มจากการนำตัวเลข x_i (ตัวตั้ง) และ y_i (ตัวคูณ) มาทำการคำนวณกัน ซึ่งแสดงได้ดังสมการที่ 3.4

$$\begin{aligned} lower &= \min(x_i) \times \min(y_i), \\ upper &= \max(x_i) \times \max(y_i) \end{aligned} \tag{3.4}$$

โดยที่ $x_i, y_i \in \{\bar{1}, \gamma, 0, 1, \alpha\}$ ซึ่ง *lower* และ *upper* ที่คำนวณได้ จะถูกนำไปใช้ในการรวมตัวเลขให้อยู่ในระบบแทนช่วงแบบยืดหยุ่นโดยใช้ตารางที่ 3.5 ดังนั้นกฎการคูณสามารถสร้างได้จากการนำ x_i และ y_i ใดๆมาทำการคำนวณกันดังสมการที่ 3.4 และได้ผลลัพธ์คือ combine(*lower*, *upper*) จากตารางที่ 3.5 โดยกฎการคูณสามารถแสดงได้ในตารางที่ 3.6

ตารางที่ 3.5 แสดงการรวมตัวเลขให้อยู่ในระบบแทนช่วงแบบยืดหยุ่น

lower	*upper*	combine(*lower*, *upper*)
$\bar{1}$	$\bar{1}$	$\bar{1}$
$\bar{1}$	0	γ
0	0	0
0	1	α
1	1	1
0	$\bar{1}$	δ
1	0	ω

ตารางที่ 3.6 แสดงกฎการคูณในระบบแทนช่วงแบบยืดหยุ่น

$\times$	$\bar{1}$	γ	0	α	1	ω	δ
$\bar{1}$	1	ω	0	δ	$\bar{1}$	γ	α
γ	ω	ω	0	0	γ	γ	0
0	0	0	0	0	0	0	0
α	δ	0	0	α	α	0	δ
1	$\bar{1}$	γ	0	α	1	ω	δ
ω	γ	γ	0	0	ω	ω	0
δ	α	0	0	δ	δ	0	α

เรายังได้เสนอกฎการบวกของ δ และ ω ในระบบแทนช่วงแบบยืดหยุ่น ซึ่งจะถูกใช้ในขั้นตอนสุดท้ายของการคูณ นั่นคือกระบวนการกำจัดตัวเลขชั่วคราวออกจากผลลัพธ์ที่ได้จากการคูณ โดยกฎการบวกของ δ และ ω ในระบบแทนช่วงแบบยืดหยุ่นแสดงไว้ในตารางที่ 3.7

ตารางที่ 3.7 แสดงกฎการบวกของ δ และ ω ในระบบแทนช่วงแบบยืดหยุ่น

+	$\bar{1}$	γ	0	α	1	δ	ω
0	$0\bar{1}$	0γ	00	0α	01	0δ	0ω
δ	$\delta\gamma$	$0\bar{1}$	0δ	00	0ω	$\delta 0$	$\omega\bar{1}, \delta 1$
ω	0δ	00	0ω	01	$\omega\alpha$	$\omega\bar{1}, \delta 1$	$\omega 0$

จากตารางกฎการบวกของ δ และ ω จะสังเกตได้ว่าถ้าเรานำ δ และ ω มาทำการบวกกับ $\bar{1},\gamma,0,\alpha,1,\delta$ และ ω ผลลัพธ์ทั้งหมดที่เป็นไปได้จะมีตัวทดนำออกได้แก่ $0,\delta$ และ ω เท่านั้น ดังนั้นตารางการบวกนี้จึงสมบูรณ์

กระบวนการคูณในระบบแทนช่วงแบบยืดหยุ่นภายหลังจากที่เราทราบเครื่องหมายของช่วงที่เป็นตัวตั้งและตัวคูณ รวมถึงกฎการคูณแล้ว สามารถทำได้โดยใช้อัลกอริทึมการคูณและกฎการคูณในระบบแทนช่วงแบบยืดหยุ่น ดังทฤษฎีบทที่ 3.4

ทฤษฎีบทที่ 3.4 *การคูณในระบบแทนช่วงแบบยืดหยุ่นสามารถคำนวณได้*

พิสูจน์

วิธีการพิสูจน์ทฤษฎีบทนี้ จะเสนออัลกอริทึมการคูณและกฎการคูณสำหรับระบบแทนช่วงแบบยืดหยุ่น

อัลกอริทึม: การคูณ

input FIR *X*, *Y*

X = $x_n x_{n-1} \ldots x_1$ where $x_i \in \{\bar{1}, \gamma, 0, \alpha, 1\}$

Y = $y_n y_{n-1} \ldots y_1$ where $y_i \in \{\bar{1}, \gamma, 0, \alpha, 1\}$

output FIR *Z*

Z = $z_n z_{n-1} \ldots z_1$ where $z_i \in \{\bar{1}, \gamma, 0, \alpha, 1\}$

begin

if *min_X* = pos {*X* > 0}

  if *min_Y* = pos {*Y* > 0}

    Lx ← min(), Ly ← min()

    Ux ← max(), Uy ← max()

  else if *max_Y* = neg {*Y* < 0}

    Lx ← max(), Ly ← min()

    Ux ← min(), Uy ← max()

  else {0 ∈ *Y*}

    Lx ← max(), Ly ← min()

    Ux ← max(), Uy ← max()

  endif

if *max_X* = neg {*X* < 0}

  if *min_Y* = pos {*Y* > 0}

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

```
                Lx ← min(),  Ly ← max()
                Ux ← max(), Uy ← min()
        else if  max_Y = neg   {Y < 0}
                Lx ← max(), Ly ← max()
                Ux ← min(), Uy ← min()
        else    {0 ∈ Y}
                Lx ← min(), Ly ← max()
                Ux ← min(), Uy ← min()
        endif
else    {0 ∈ X}
        if  min_Y = pos          {Y > 0}
                Lx ← min(),  Ly ← max()
                Ux ← max(), Uy ← max()
        else if  max_Y = neg   {Y < 0}
                Lx ← max(), Ly ← min()
                Ux ← min(), Uy ← min()
        else    {0 ∈ Y}
                a  ← Lower endpoint of X
                b  ← Upper endpoint of X
                c  ← Lower endpoint of Y
                d  ← Upper endpoint of Y
                min_Z  ← min{a×d, b×c}
                max_Z ← max{a×c, b×d}
                if  min_Z = b×c and max_Z = a×c
                        Lx ← max(), Ly ← min()
                        Ux ← min(), Uy ← min()
                else if min_Z = a×d and max_Z =  b×d
                        Lx ← min(),  Ly ← max()
                        Ux ← max(), Uy ← max()
                else if min_Z = a×d and max_Z = a×c
                        Lx ← min(),  Ly ← max()
                        Ux ← min(), Uy ← min()
                else
                        Lx ← max(), Ly ← min()
```

Ux $\leftarrow$ max(), Uy $\leftarrow$ max()

endif

endif

$i \leftarrow 1$

while (not-end-of-data) **do**

lower $\leftarrow$ Lx(x_i)

upper $\leftarrow$ Ux(x_i)

x_i $\leftarrow$ combine(*lower*, *upper*)

lower $\leftarrow$ Ly(y_i)

upper $\leftarrow$ Uy(y_i)

y_i $\leftarrow$ combine(*lower*, *upper*)

$i \leftarrow i+1$

enddo

Z $\leftarrow$ Multiplication by multiplication rules

$k \leftarrow 1$

while (not-end-of-data) **do**

if $z_k = \delta$ **then** $z_k \leftarrow \alpha$, $z_{k+1} \leftarrow z_{k+1} + \delta$ **endif**

if $z_k = \omega$ **then** $z_k \leftarrow \gamma$, $z_{k+1} \leftarrow z_{k+1} + \omega$ **endif**

$k \leftarrow k+1$

enddo

end

พิสูจน์อัลกอริทึม

เนื่องจากเราสามารถทราบเครื่องหมายของค่าต่ำสุดและค่าสูงสุดของช่วง *X* และ *Y* ว่าเป็นบวกหรือลบ ดังนั้นเราสามารถเขียนช่วง *X* และ *Y* ให้อยู่ในรูปแบบที่พร้อมที่จะทำการคูณได้ เรียกว่ารูปแบบพร้อมคูณ ซึ่งรูปแบบพร้อมคูณนี้สามารถสร้างได้จากตารางที่ 3.5 โดยผลลัพธ์ที่ได้จากการคูณสามารถคำนวณได้จากกฎการคูณในตารางที่ 3.6 แต่ในบางครั้งผลลัพธ์ที่ได้จากการคำนวณอาจจะมีตัวเลขชั่วคราว δ และ ω ปรากฏอยู่ได้ ซึ่งยังไม่ใช่ผลลัพธ์ที่ถูกต้อง เพราะฉะนั้นเราจึงจำเป็นต้องผ่านกระบวนการกำจัดตัวเลขชั่วคราวเหล่านั้นออกไปเพื่อที่จะให้ผลลัพธ์สุดท้ายเป็นคำตอบที่ถูกต้องและอยู่ในรูปแบบแทนช่วงแบบยืดหยุ่นด้วย

■

ตัวอย่างที่ 3.4 *การคูณของช่วง [3, 10] โดย [-8, -5] ในระบบแทนช่วงแบบยืดหยุ่น*

วิธีทำ จากอัลกอริทึมการคูณ ขั้นตอนการคูณของช่วง X = [3, 10] และ Y = [-8, -5] แสดงได้ใน ตารางที่ 3.8

ตารางที่ 3.8 แสดงขั้นตอนการคูณของช่วง [3, 10] และ [-8, -5]

ขั้นตอนการคูณ	ผลลัพธ์
1. รูปแบบแทนช่วง *X* = [3, 10] และ *Y* = [-8, -5]	$1\gamma\alpha\gamma, \bar{1}0\alpha\alpha$
2. เครื่องหมายของ *min_X* และ *max_X*	pos
3. เครื่องหมายของ *min_Y* และ *max_Y*	neg
4. รูปแบบพร้อมคูณของช่วง *X* และ *Y*	$1\delta\omega\delta, \bar{1}0\alpha\alpha$
5. การคูณโดยใช้กฎการคูณในตารางที่ 3.6	$\gamma 0\gamma\delta\delta\delta$
6. ผลลัพธ์สุดท้ายภายหลังกระบวนการกำจัดตัวเลขชั่วคราวออกไป	$\gamma 0\bar{1}000\alpha$

ผลลัพธ์ที่ได้จากอัลกอริทึมการคูณคือ $\gamma 0\bar{1}000\alpha$ โดยมีค่าเชิงตัวเลขเท่ากับ [-80, -15] ดังนั้น ผลลัพธ์ของการคูณในระบบแทนช่วงแบบยืดหยุ่นสามารถให้คำตอบที่ถูกต้อง □

3.3.4 การหารในระบบแทนช่วงแบบยืดหยุ่น

การหารในระบบแทนช่วงแบบยืดหยุ่นนั้น จะมีขั้นตอนในการหารแบ่งออกเป็น 2 ขั้นตอนเช่นเดียวกับการคูณ นั่นคือ ในขั้นตอนแรกจะต้องทำการตรวจสอบเครื่องหมายของช่วงที่จะนำมาหารกันโดยใช้อัลกอริทึมการตรวจสอบเครื่องหมายของช่วง เพื่อเป็นตัวกำหนดรูปแบบของตัวตั้งและตัวหารที่เหมาะสมที่จะนำไปเริ่มกระบวนการหาร ซึ่งเราเรียกรูปแบบนี้ว่า รูปแบบพร้อมหาร (division-ready pattern) และในขั้นตอนที่สองคือขั้นตอนของอัลกอริทึมการหาร

สำหรับกระบวนการการหารโดยทั่วไปนั้นจะต้องมีขั้นตอนของการลดค่าด้วย ดังนั้น เราจะขอเสนอกฎการลดค่า (reduction rules) สำหรับอัลริทึมการหาร โดยวิธีการสร้างตารางกฎการลดค่าสำหรับอัลกอริทึมการหารนั้นสามารถแสดงได้ดังนี้ เริ่มจากการนำตัวเลข x_i (ตัวตั้ง) และ y_i (ตัวลดค่า) มาทำการคำนวณกัน ซึ่งแสดงได้ดังสมการที่ 3.5

$$
\begin{aligned}
\textit{lower} &= \min(x_i) - \min(y_i), \\
\textit{upper} &= \max(x_i) - \max(y_i)
\end{aligned}
\tag{3.5}
$$

โดยที่ $x_i, y_i \in \{\bar{1}, \gamma, 0, 1, \alpha, \omega, \delta\}$ ซึ่งผลลัพธ์ในตารางกฎการลดค่าสำหรับอัลกอริทึมการหาร สามารถคำนวณได้จากสมการที่ 3.5 ทั้งนี้ *lower* และ *upper* คือค่าต่ำสุดและค่าสูงสุดของคำตอบที่ได้จากการคำนวณ โดยกฎการลดค่าสำหรับอัลกอริทึมการหารแสดงได้ดังตารางที่ 3.9

ตารางที่ 3.9 แสดงกฎการลดค่าสำหรับอัลกอริทึมการหาร

		ตัวตั้ง						
		$\bar{1}$	γ	0	α	1	ω	δ
ตัวลดค่า	$\bar{1}$	00	0α	01	1γ	10	1δ	0ω
	γ	0δ	00	0ω	01	1δ	$\omega 0$	$\delta 1$
	0	$0\bar{1}$	0γ	00	0α	01	0ω	0δ
	α	$\bar{1}\omega$	$0\bar{1}$	0δ	00	0ω	$\delta 1$	$\delta 0$
	1	$\bar{1}0$	$\bar{1}\alpha$	$0\bar{1}$	0γ	00	0δ	$\bar{1}\omega$
	ω	$\bar{1}\alpha$	$\gamma 0$	0γ	$\gamma 1$	0α	00	$0\bar{1}$
	δ	0γ	$\gamma 1$	0α	$\alpha 0$	1γ	01	00

กระบวนการหารในระบบแทนช่วงแบบยืดหยุ่นภายหลังจากที่เราทราบเครื่องหมายของช่วงที่เป็นตัวตั้งและตัวหารแล้ว สามารถทำได้ด้วยการใช้อัลกอริทึมการหารและกฎการคูณในระบบแทนช่วงแบบยืดหยุ่น รวมถึงกฎการลดค่าสำหรับอัลกอริทึมการหารด้วย ดังทฤษฎีบทที่ 3.5

ทฤษฎีบทที่ 3.5 *การหารในระบบแทนช่วงแบบยืดหยุ่นสามารถคำนวณได้*

พิสูจน์

วิธีการพิสูจน์ทฤษฎีบทนี้ จะเสนออัลกอริทึมการหารสำหรับระบบแทนช่วงแบบยืดหยุ่น

อัลกอริทึม: การหาร

input FIR *X*, *Y*

X = $x_n x_{n-1} \ldots x_1$ where $x_i \in \{\bar{1}, \gamma, 0, \alpha, 1\}$

Y = $y_n y_{n-1} \ldots y_1$ where $y_i \in \{\bar{1}, \gamma, 0, \alpha, 1\}$

output FIR *Z*

Z = $z_n z_{n-1} \ldots z_1$ where $z_i \in \{\bar{1}, \gamma, 0, \alpha, 1\}$

begin

if *min_X* = neg **and** *max_X* = pos {0 $\in$ *X*}

if *min_Y* = pos {*Y* > 0}

Lx $\leftarrow$ min(), Ly $\leftarrow$ min()

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

```
            Ux ← max(), Uy ← min()
      else  {Y < 0}
            Lx ← max(), Ly ← max()
            Ux ← min(), Uy ← max()
      endif
if  min_X = pos        {X > 0}
      if  min_Y = pos        {Y > 0}
            Lx ← min(), Ly ← max()
            Ux ← max(), Uy ← min()
      else  {Y < 0}
            Lx ← max(), Ly ← max()
            Ux ← min(), Uy ← min()
      endif
else  {X < 0}
      if  min_Y = pos        {Y > 0}
            Lx ← min(), Ly ← min()
            Ux ← max(), Uy ← max()
      else  {Y < 0}
            Lx ← max(), Ly ← min()
            Ux ← min(), Uy ← max()
      endif
endif
i ← 1
while (not-end-of-data) do
      lower ← Lx(x_i)
      upper ← Ux(x_i)
      x_i ← combine(lower, upper)
      lower ← Ly(y_i)
      upper ← Uy(y_i)
      y_i ← combine(lower, upper)
      i ← i + 1
enddo
Z ← Division by multiplication and reduction rules
k ← 1
```

while (not-end-of-data) **do**

if $z_k = \delta$ **then** $z_k \leftarrow \alpha$, $z_{k+1} \leftarrow z_{k+1} + \delta$ **endif**

if $z_k = \omega$ **then** $z_k \leftarrow \gamma$, $z_{k+1} \leftarrow z_{k+1} + \omega$ **endif**

$k \leftarrow k+1$

enddo

end

พิสูจน์อัลกอริทึม

เนื่องจากเราสามารถทราบเครื่องหมายของค่าต่ำสุดและค่าสูงสุดของช่วง *X* และ *Y* ว่าเป็นบวกหรือลบ ดังนั้นเราสามารถเขียนช่วง *X* และ *Y* ให้อยู่ในรูปแบบที่พร้อมที่จะทำการหารได้ เรียกว่า รูปแบบพร้อมหาร ซึ่งรูปแบบพร้อมหารนี้สามารถสร้างได้จากตารางที่ 3.5 โดยผลลัพธ์ที่ได้จากการหารสามารถคำนวณได้จากกฎการคูณในตารางที่ 3.6 สำหรับกฎการลดค่าในระหว่างกระบวนการหารนั้นถูกแสดงไว้ในตารางที่ 3.9 แต่ในบางครั้งผลลัพธ์ที่ได้จากการคำนวณอาจจะมีตัวเลขชั่วคราว δ และ ω ปรากฏอยู่ได้ซึ่งยังไม่ใช่ผลลัพธ์ที่ถูกต้อง เพราะฉะนั้นเราจึงจำเป็นต้องผ่านกระบวนการกำจัดตัวเลขชั่วคราวเหล่านั้นออกไปเพื่อที่จะให้ผลลัพธ์สุดท้ายเป็นคำตอบที่ถูกต้องและอยู่ในรูปแบบแทนช่วงแบบยืดหยุ่นด้วย

■

ตัวอย่างที่ 3.5 *การหารของช่วง [-30, 13] โดย [-26, -18] ในระบบแทนช่วงแบบยืดหยุ่น*

วิธีทำ จากอัลกอริทึมการหาร ขั้นตอนการหารของช่วง X = [-30, 13] และ Y = [-26, -18] แสดงได้ในตารางที่ 3.10

ตารางที่ 3.10 แสดงขั้นตอนการหารของช่วง [-30, 13] โดย [-26, -18]

ขั้นตอนการหาร	ผลลัพธ์
1. รูปแบบแทนช่วง *X* = [-30, 13] และ *Y* = [-26, -18]	$\gamma\bar{1}\bar{1}\bar{1}\alpha, \bar{1}\gamma 0\bar{1}0$
2. เครื่องหมายของ *min_X* และ *max_X*	neg, pos
3. เครื่องหมายของ *min_Y* และ *max_Y*	neg, neg
4. รูปแบบพร้อมหารของช่วง *X* และ *Y*	$\delta\bar{1}\bar{1}\bar{1}\omega, \bar{1}\gamma 0\bar{1}0$
5. การหารโดยใช้กฎการคูณในตารางที่ 3.6	$\alpha.1\alpha\delta\alpha\delta...$
6. ผลลัพธ์สุดท้ายภายหลังกระบวนการกำจัดตัวเลขชั่วคราวออกไป	$\alpha.10\alpha 0\alpha...$

ผลลัพธ์ที่ได้จากอัลกอริทึมการหารคือ $\alpha.10\alpha0\alpha$... โดยมีค่าเชิงตัวเลขเท่ากับ [0.5, 1.65…] ซึ่งมีค่าเท่ากับผลลัพธ์ที่ได้จากการหารกันในระบบเลขฐานสอง ดังนั้นผลลัพธ์ของการหารในระบบแทนช่วงแบบยืดหยุ่นสามารถให้คำตอบที่ถูกต้อง □

3.4 สรุป

ในบทนี้เราได้เสนอรูปแบบการแทนจำนวนแบบช่วงแบบใหม่ ที่มีชื่อเรียกว่า ระบบแทนช่วงแบบยืดหยุ่น ซึ่งระบบการแทนจำนวนแบบใหม่นี้สามารถลดเนื้อที่ที่ใช้ในการแสดงช่วงให้น้อยลงได้ เนื่องจากระบบใหม่นี้ใช้จำนวนเพียงจำนวนเดียวก็สามารถแสดงช่วงได้ อีกทั้งในบทนี้ยังได้เสนอความสมบูรณ์ของระบบแทนช่วงแบบยืดหยุ่นและอัลกอริทึมสำหรับการดำเนินการพื้นฐานทางเลขคณิต (บวก ลบ คูณ และหาร) สำหรับระบบแทนช่วงแบบยืดหยุ่นอีกด้วย

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

บทที่ 4

บทวิเคราะห์ระบบแทนช่วงแบบยืดหยุ่น

เนื้อหาในบทนี้กล่าวถึงบทวิเคราะห์เกี่ยวกับระบบแทนช่วงแบบยืดหยุ่นในการขยายขอบเขตของระบบแทนช่วงแบบยืดหยุ่นเดิมซึ่งอยู่ในฐานสอง ให้อยู่ในรูปทั่วไป (generic form) โดยที่มีเลขฐานมากกว่าหรือเท่ากับสอง และการนำสถาปัตยกรรมแบบทำควบคู่กันมาประยุกต์ใช้กับการคำนวณในระบบแทนช่วงแบบยืดหยุ่นเพื่อเพิ่มความเร็วในการคำนวณ โดยจะพิจารณาถึงผลกระทบต่าง ๆที่จะเกิดขึ้น

4.1 ระบบแทนช่วงแบบยืดหยุ่นในรูปทั่วไป

ลักษณะของระบบแทนช่วงแบบยืดหยุ่นในรูปทั่วไปนั้น จะมีการเพิ่มตัวเลขที่เป็นจำนวนเต็มและตัวเลขยืดหยุ่นในชุดตัวเลข D ให้มีจำนวนมากขึ้นเพื่อรองรับการแสดงช่วงต่าง ๆ ให้ครบถ้วนดังนิยามที่ 4.1 และ 4.2

นิยามที่ 4.1 *ระบบแทนช่วงแบบยืดหยุ่นในรูปทั่วไปประกอบด้วยฐาน β และชุดตัวเลข D โดยที่ β เป็นจำนวนเต็มใด ๆที่ $\beta \geq 2$ และชุดตัวเลข D ประกอบด้วยตัวเลขที่เป็นจำนวนเต็มและตัวเลขยืดหยุ่น ในส่วนของตัวเลขที่อยู่ในรูปแบบของจำนวนเต็ม e จะมีจำนวน $\beta+1$ ตัว โดย e จะเป็นจำนวนเต็มทั้งหมดที่อยู่ระหว่าง $\bar{1}$ ถึง $\beta-1$ ($\{e \in \mathrm{Z} \mid \bar{1} \leq e \leq \beta-1\}$) ในส่วนของตัวเลขยืดหยุ่น จะมีจำนวน $\beta \times (\beta-1)$ ตัว ขนาดของตัวเลขยืดหยุ่นจะมีค่าตั้งแต่ 1 ถึง $\beta-1$ ซึ่งขนาดของตัวเลขยืดหยุ่นสามารถคำนวณได้จากการนำค่าสูงสุดของตัวเลขยืดหยุ่นมาลบด้วยค่าต่ำสุดของตัวเลขยืดหยุ่น โดยในแต่ละขนาดของตัวเลขยืดหยุ่น จะมีจำนวนตัวเลขยืดหยุ่น β ตัว ทั้งนี้ตัวเลขยืดหยุ่นในแต่ละขนาดจะเริ่มสร้างจากตัวเลขที่เป็นจำนวนเต็มที่มีค่ามากที่สุด ($\beta-1$) แล้ววนซ้ำไปทางฝั่งของตัวเลขที่มีค่าลดลงจนครบจำนวนตัวเลขยืดหยุ่นของขนาดนั้น ๆ*

นิยามที่ 4.2 *รูปแบบแทนช่วง $X = x_1 x_2 x_3 \ldots x_n$ ในระบบแทนช่วงแบบยืดหยุ่นในรูปทั่วไปสามารถแสดงช่วง $[a\,,\,b]$ ได้โดย*

$$a = \sum_{i=0}^{n} \min(x_i) \times \beta^i \ \text{ และ } \ b = \sum_{i=0}^{n} \max(x_i) \times \beta^i$$

ตัวอย่างที่ 4.1 *จงหาระบบแทนช่วงแบบยืดหยุ่นในรูปทั่วไป สำหรับ β = 3*

วิธีทำ วิธีการสร้างระบบแทนช่วงแบบยืดหยุ่นในรูปทั่วไป สำหรับ β = 3 นั้น จะแบ่งการสร้างชุดตัวเลข D ออกเป็น 2 กรณี คือ ตัวเลขที่เป็นจำนวนเต็มและตัวเลขยืดหยุ่น

กรณีที่ 1

ตัวเลขที่เป็นจำนวนเต็มใน D $(e \in Z \mid \bar{1} \le e \le 2)$ คือ $\bar{1}, 0, 1, 2$

กรณีที่ 2

ตัวเลขยืดหยุ่นใน D จะมีขนาดของตัวเลขยืดหยุ่นไม่เกิน 2 นั่นคือ มีตัวเลขยืดหยุ่นที่มีขนาดเท่ากับ 1 และ 2 นอกจากนี้จำนวนตัวเลขยืดหยุ่นในแต่ละขนาดจะมีจำนวนเป็น 3 ตัว โดยวิธีการสร้างจะเริ่มจากฝั่งที่มีค่ามากที่สุด (ในตัวอย่างนี้คือ 2) และไล่ไปทางฝั่งที่มีค่าลดลงจนครบจำนวนตัวเลขยืดหยุ่น นั่นคือ ในช่วงขนาด 1 จะมีตัวเลขยืดหยุ่นจำนวน 3 ตัว คือ μ_1 = [1, 2], μ_2 = [0, 1] และ μ_3 = [$\bar{1}$, 0] และในช่วงขนาด 2 จะมีตัวเลขยืดหยุ่นจำนวน 3 ตัวคือ ψ_1 = [0, 2], ψ_2 = [$\bar{1}$, 1] และ ψ_3 = [$\bar{2}$, 0]

ดังนั้นในระบบแทนช่วงแบบยืดหยุ่นในรูปทั่วไปที่มี β = 3 นั้น จะมีชุดตัวเลข $D = \{\bar{1}, 0, 1, 2, \mu_1, \mu_2, \mu_3, \psi_1, \psi_2, \psi_3\}$ □

นิยามที่ 4.3 *ชุดตัวเลขซ้ำซ้อนน้อยที่สุด (minimally redundant digit set) หมายถึง ชุดตัวเลขที่มีจำนวนตัวเลขน้อยที่สุดที่ทำให้บางช่วงในระบบสามารถมีรูปแบบแทนช่วงได้มากกว่าหนึ่งรูปแบบ*

ลักษณะสำคัญประการหนึ่งของระบบแทนช่วงแบบยืดหยุ่นในรูปทั่วไปคือ ชุดตัวเลข D ของระบบแทนช่วงแบบยืดหยุ่นในรูปทั่วไป จะมีจำนวนตัวเลขเท่ากับ β^2+1 ซึ่งเราสามารถเรียกชุดตัวเลข D นี้ว่า ชุดตัวเลขซ้ำซ้อนน้อยที่สุด ดังทฤษฎีบทที่ 4.1

ทฤษฎีบทที่ 4.1 *ระบบแทนช่วงแบบยืดหยุ่นในรูปทั่วไปในฐาน $\beta \ge 2$ จะมีจำนวนตัวเลขในชุดตัวเลข D เท่ากับ β^2+1 ซึ่งในกรณีเช่นนี้จะเรียกชุดตัวเลข D นี้ว่า ชุดตัวเลขซ้ำซ้อนน้อยที่สุด*

พิสูจน์ ในระบบจำนวนในฐาน β ใดๆ ต้องมีจำนวนตัวเลขอย่างน้อย β ตัว เพื่อทำให้ระบบมีความสมบูรณ์ โดยสามารถแสดงจำนวนใดๆ ได้ทุกจำนวน แต่ในระบบแทนช่วงนั้นประกอบด้วยจำนวน 2 จำนวนคือ จำนวนที่แสดงค่าต่ำสุดและค่าสูงสุด ซึ่งในการแสดงค่าของแต่ละค่าจะต้องใช้จำนวนตัวเลขอย่างน้อย β ตัว ดังนั้นถ้าเรารวมจำนวน 2 จำนวนที่แสดงค่าต่ำสุดและค่าสูงสุดเข้าไว้ด้วยกันให้อยู่ในรูปของจำนวนเพียงจำนวนเดียว จะต้องใช้จำนวนตัวเลขทั้งหมดอย่างน้อย β^2 ตัว แต่เนื่องจากในระบบแทนช่วงแบบยืดหยุ่นในรูปทั่วไปเป็นระบบซ้ำซ้อน เพราะฉะนั้นเราจึงเพิ่มตัวเลขเข้าไปในชุดตัวเลขอีกหนึ่งตัว เพื่อทำให้ชุดตัวเลขดังกล่าวมีสมบัติซ้ำซ้อน ด้วยเหตุนี้จำนวนตัวเลขในชุดตัวเลขของระบบแทนช่วงแบบยืดหยุ่นในรูปทั่วไปจึงมีจำนวน β^2+1 ตัว ซึ่งเราสามารถเรียกชุดตัวเลขนี้ได้ว่า ชุดตัวเลขซ้ำซ้อนน้อยที่สุด

■

อีกสิ่งหนึ่งที่เราต้องคำนึงถึงคือเรื่องของความสมบูรณ์ (completeness) ของระบบซึ่งความสมบูรณ์ของระบบแทนช่วงแบบยืดหยุ่นในรูปทั่วไปสามารถแสดงได้ดังทฤษฎีบทที่ 4.2

ทฤษฎีบทที่ 4.2 *ทุกช่วง [x, y] ใด ๆ สามารถมีรูปแบบแทนช่วง ในระบบแทนช่วงแบบยืดหยุ่นในรูปทั่วไปได้*

พิสูจน์

วิธีการพิสูจน์ทฤษฎีบทนี้ จะเสนออัลกอริทึมสำหรับแปลง (conversion) ช่วงใด ๆ ให้อยู่ในรูปของรูปแบบแทนช่วงแบบยืดหยุ่นในรูปทั่วไป

อัลกอริทึม : การแปลง

input interval [A, B]

$A = a_0a_1\ldots a_n$ where $a_i \in \{\bar{1},0,\ldots,\beta-1\}$

$B = b_0b_1\ldots b_n$ where $b_i \in \{\bar{1},0,\ldots,\beta-1\}$

output $S = s_0s_1\ldots s_n$ where $s_i \in \{e_0,\ldots,e_n, f_0,\ldots,f_n\}$ with $e_i \in \mathrm{Z} \mid \bar{1} \le e \le \beta-1$ and f_i are flexible digits

begin

$C = B - A$ where $c_i \in C$ with $c_i \in \{0,\ldots,\beta-1\}$

$i \leftarrow 0$

k is any integer

while (not-end-of-data) **do**

if $c_i = k$ **then** $c_i \leftarrow$ flexible digit that represents [0, k]

endif

enddo

$S \leftarrow C + A$

end

พิสูจน์อัลกอริทึม

เนื่องจากขนาดของช่วงซึ่งคำนวณได้จากการนำค่าสูงสุดมาลบด้วยค่าต่ำสุดนั้นสามารถแสดงได้โดย C ซึ่งเป็นจำนวนไม่เป็นลบ ดังนั้นช่วง [0, C] ใด ๆ สามารถแสดงได้โดยใช้ลำดับของตัวเลข 0 และตัวเลขยืดหยุ่นในแต่ละขนาดเพียงลำดับเดียวได้ เพราะฉะนั้นรูปแบบแทนช่วง [A, B] ใด ๆ สามารถแสดงได้โดยการนำค่าต่ำสุดของช่วง (A) มาทำการบวกเข้ากับขนาดของช่วง (C) ■

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

ตัวอย่างที่ 4.2 *หารูปแบบแทนช่วง [2, 7] ในระบบแทนช่วงแบบยืดหยุ่นในรูปทั่วไปที่* $\beta = 3$

วิธีทำ ขั้นตอนการหารูปแบบแทนช่วงจะแสดงในตารางที่ 4.1 โดยใช้สัญลักษณ์ μ และ ψ แทนตัวเลขยืดหยุ่นจากตัวอย่างที่ 4.1

ตารางที่ 4.1 แสดงขั้นตอนการแปลงช่วง [2, 7] ให้อยู่ในระบบแทนช่วงแบบยืดหยุ่นในรูปทั่วไป

ขั้นตอนการแปลง	ผลลัพธ์
1. ช่วง [2, 7]	$[0002,0021]$
2. รูปแบบแทนจำนวนของค่าต่ำสุดของช่วง	0002
3. รูปแบบแทนช่วงของขนาดของช่วง	$00\mu_2\psi_1$
4. รูปแบบแทนช่วงของ [2, 7] สามารถแสดงได้โดยการนำผลลัพธ์ของขั้นตอนที่ 2 และ 3 มาบวกกัน	$00\mu_1\psi_2$

□

สำหรับการดำเนินการพื้นฐานทางเลขคณิต ได้แก่ การบวก ลบ คูณ และหารของระบบแทนช่วงแบบยืดหยุ่นในรูปทั่วไปนั้น อาจจะไม่สามารถคำนวณได้โดยใช้อัลกอริทึมของระบบแทนช่วงแบบยืดหยุ่น เพราะในระบบแทนช่วงแบบยืดหยุ่นในรูปทั่วไปนั้น ตัวเลขในชุดตัวเลขนั้นไม่สมมาตรกัน อาจจะก่อให้เกิดปัญหาในการคำนวณได้ เช่น การลบในระบบแทนช่วงแบบยืดหยุ่น จะทำการกลับเครื่องหมายของตัวเลขของช่วงที่เป็นตัวลบ ซึ่งเราไม่สามารถกระทำวิธีนี้ได้ในระบบแทนช่วงแบบยืดหยุ่นในรูปทั่วไป เป็นต้น ดังนั้นอัลกอริทึมสำหรับการดำเนินการพื้นฐานทางเลขคณิตของระบบแทนช่วงแบบยืดหยุ่นในรูปทั่วไป จำเป็นต้องได้รับการปรับปรุงเพื่อให้ระบบนี้สามารถคำนวณให้ได้ผลลัพธ์ที่ถูกต้อง

4.2 การบวกในระบบแทนช่วงแบบยืดหยุ่นโดยใช้สถาปัตยกรรมแบบควบคู่กัน

การบวกกันของช่วงสองช่วงใดๆ ในระบบแทนช่วงแบบยืดหยุ่นจะเป็นการบวกแบบอนุกรมซึ่งต้องใช้เวลาคือ O(n) เมื่อ n หมายถึงจำนวนตัวเลขที่เป็นข้อมูลนำเข้า สำหรับแนวทางในการเพิ่มความเร็วในการคำนวณนั้น ได้มีการนำสถาปัตยกรรมแบบควบคู่กันมาประยุกต์ใช้ ส่งผลให้เวลาที่ต้องใช้ในการคำนวณลดลงเหลือ O(log n) สำหรับกระบวนการทำงานนั้นจะเป็นการแปลงจากชุดตัวเลข $D = \{\bar{2},\bar{1},0,1,2,\alpha,\gamma,\tau_1,\tau_2,\tau_3,\tau_4,\tau_5\}$ ไปยังชุดตัวเลข $E = \{\bar{1},0,1,\alpha,\gamma\}$ โดยที่ $\tau_1 = [1,2]$, $\tau_2 = [0,2]$, $\tau_3 = [\bar{2},\bar{1}]$, $\tau_4 = [\bar{2},0]$ และ $\tau_5 = [\bar{1},1]$ ความสัมพันธ์ของการแปลงระหว่างชุดตัวเลข λ เป็นฟังก์ชันการแปลงสามารถแสดงได้ดังนี้

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

$$\lambda : C \times D \rightarrow C \times E$$

โดยที่ C คือชุดตัวเลขของตัวทดที่เป็นไปได้ทั้งหมดนั่นคือ $C = \{\bar{1}, 0, 1, \alpha, \gamma\}$ และสำหรับบาง c' ที่อยู่ใน C และ e ที่อยู่ใน E สามารถเขียนเป็นสมการได้ดังนี้

$$c' + d = c\beta + e$$

เมื่อ β = 2 โดยเราสามารถเรียก c' และ c คือตัวทดนำเข้าและตัวทดนำออกตามลำดับในแต่ละตัวเลข d ที่เป็นข้อมูลนำเข้า ซึ่งฟังก์ชันการแปลงชุดตัวเลขในระบบแทนช่วงแบบยืดหยุ่น สามารถแสดงได้ดังตารางที่ 4.2

ตารางที่ 4.2 แสดงฟังก์ชันการแปลงจาก D ไปยัง E ในระบบแทนช่วงแบบยืดหยุ่น

λ		D											
		$\bar{2}$	$\bar{1}$	0	1	2	α	γ	τ_1	τ_2	τ_3	τ_4	τ_5
C	$\bar{1}$	$\bar{1}\bar{1}$	$\bar{1}0$	$0\bar{1}$	00	01	0γ	$\bar{1}\alpha$	0α	$\alpha\bar{1}$	$\bar{1}\gamma$	$\gamma\bar{1}$	$\gamma 0$
	0	$\bar{1}0$	$0\bar{1}$	00	01	10	0α	0γ	1γ	$\alpha 0$	$\bar{1}\alpha$	$\gamma 0$	$\gamma 1$
	1	$0\bar{1}$	00	01	10	11	1γ	0α	1α	$\alpha 1$	0γ	$\alpha\bar{1}$	$\alpha 0$
	α	$\bar{1}\alpha$	0γ	0α	1γ	1α	$\alpha 0$	$\gamma 1$	$\alpha 1$	$\alpha\alpha$	$\gamma 0$	$\gamma\alpha$	$\alpha\gamma$
	γ	$\bar{1}\gamma$	$\bar{1}\alpha$	0γ	0α	1γ	$\gamma 1$	$\gamma 0$	$\alpha 0$	$\alpha\gamma$	$\gamma\bar{1}$	$\gamma\gamma$	$\gamma\alpha$

ผลลัพธ์ในแต่ละคู่ในตารางที่ 4.2 แทนค่า ce ซึ่ง (c,e) อยู่ใน $C \times E$ โดยการคำนวณในตารางจะเริ่มจากค่า $c = 0$ จะเห็นว่า ce ที่เป็นไปได้นั้น มีตัวทดนำออกที่เป็นไปได้คือ 0, 1, $\bar{1}$, α และ γ เราจึงต้องเพิ่มแถวที่ค่า $c = 1$, $\bar{1}$, α และ γ ซึ่งจาก ce ทั้งหมดจะเห็นว่าตัวทดที่เกิดขึ้นคือ 0, 1, $\bar{1}$, α และ γ เท่านั้น ดังนั้นตารางนี้จึงสมบูรณ์

ตัวอย่างที่ 4.3 *กำหนดให้* $X = 1\alpha\tau_5\tau_1$ *คือผลลัพธ์ที่ได้จากการบวกกันของช่วง* $1\gamma\alpha 1$ *และ* $01\gamma\alpha$ *โดยการบวกแบบขนาน จงทำการแปลงชุดตัวเลขด้วยสถาปัตยกรรมแบบควบคู่กันโดยที่* $D = \{\bar{2}, \bar{1}, 0, 1, 2, \alpha, \gamma, \tau_1, \tau_2, \tau_3, \tau_4, \tau_5\}$ *และ* $E = \{\bar{1}, 0, 1, \alpha, \gamma\}$

วิธีทำ กำหนดให้ $X = x_4 x_3 x_2 x_1$ เป็นข้อมูลนำเข้า โดยที่ $x_i \in D$ เมื่อ i เป็นจำนวนเต็มบวก ฟังก์ชันการแปลงจะทำการผลิตคู่ผลลัพธ์ได้แก่ ตัวทดนำออกและตัวเลขที่เป็นผลลัพธ์ดังสมการต่อไปนี้

$$c' + d = c\beta + e$$

สามารถเขียนสมการใหม่ได้เป็น

$$c_{i-1} + x_i = c_i \beta + y_i$$

โดยที่มีตัวทดนำเข้าเริ่มต้น $c_0 = 0$ และ $y_i \in E$ ผลลัพธ์ที่ได้คือ $Y = 1\gamma 00\gamma$ เมื่อ $\|X\| = \|Y\|$ ทั้งนี้การแปลงนั้นจะแปลงจาก $D = \{\bar{2}, \bar{1}, 0, 1, 2, \alpha, \gamma, \tau_1, \tau_2, \tau_3, \tau_4, \tau_5\}$ ไปเป็น $E = \{\bar{1}, 0, 1, \alpha, \gamma\}$ โดยมี $\beta = 2$ และ $C = \{\bar{1}, 0, 1, \alpha, \gamma\}$ เราสามารถหาค่า c และ e ในแต่ละหลักได้ดังนี้

ค่า c จากฟังก์ชันประกอบสามารถแสดงได้คือ

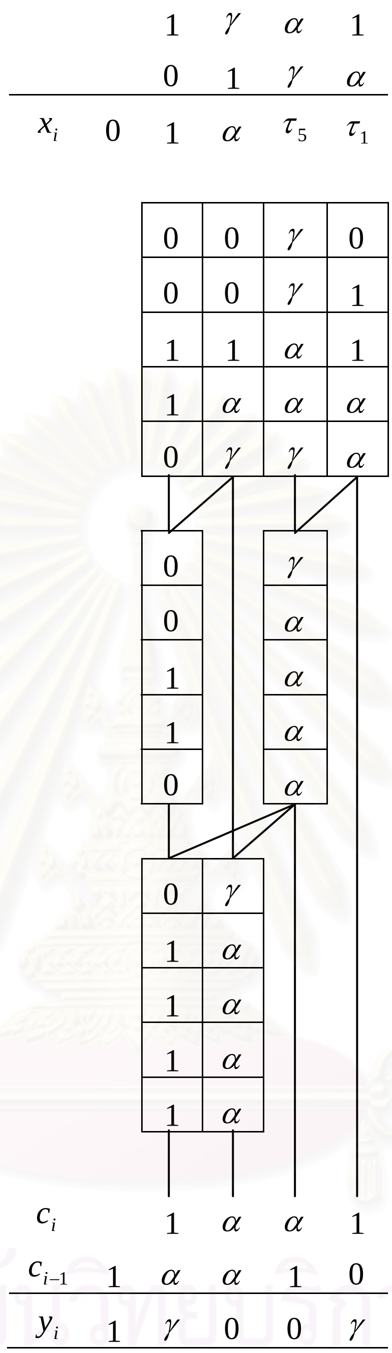
$$
\begin{aligned}
c_0 &= 0 \\
c_1 &= \sigma_1(c_0) = \sigma_1(0) = 1 \\
c_2 &= \sigma_2\sigma_1(0) = \sigma_2(1) = \alpha \\
c_3 &= \sigma_3\sigma_2\sigma_1(0) = \sigma_3(\alpha) = \alpha \\
c_4 &= \sigma_4\sigma_3\sigma_2\sigma_1(0) = \sigma_4(\alpha) = 1
\end{aligned}
$$

ค่า e จากฟังก์ชันจับคู่ตัวเลข

$$
\begin{aligned}
e_1 &= \varepsilon_1(c_0) = \varepsilon_1(0) = \gamma = y_1 \\
e_2 &= \varepsilon_2(c_1) = \varepsilon_2(1) = 0 = y_2 \\
e_3 &= \varepsilon_3(c_2) = \varepsilon_3(\alpha) = 0 = y_3 \\
e_4 &= \varepsilon_4(c_3) = \varepsilon_4(\alpha) = \gamma = y_4 \\
e_5 &= \varepsilon_5(c_4) = \varepsilon_5(1) = 1 = y_5
\end{aligned}
$$

เพราะฉะนั้น $X = 1\alpha\tau_5\tau_1$ โดยที่ $x_i \in D$ จะถูกแปลงได้เป็น $Y = 1\gamma 00\gamma$ โดยที่ $y_i \in E$ สำหรับกระบวนการแปลงชุดตัวเลขโดยใช้สถาปัตยกรรมแบบทำควบคู่กันจะแสดงในรูปที่ 4.1

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย



รูปที่ 4.1 แสดงตัวอย่างการแปลงชุดตัวเลขในระบบแทนช่วงแบบยืดหยุ่นโดยใช้สถาปัตยกรรมแบบทำควบคู่กัน

จากรูปที่ 4.1 จะเห็นได้ว่าผลลัพธ์ของ y_i ในหลักใดๆ จะได้จากการพิจารณาตัวทดนำเข้า c_{i-1} และ x_i โดยเริ่มจากหลักที่ $x_1 = \tau_1$ จะกำหนดให้ตัวทดนำเข้า $c_0 = 0$ ซึ่งจะได้ผลลัพธ์ $y_1 = \gamma$ และ $c_1 = 1$ หลังจากนั้นในหลักที่ $x_2 = \tau_5$ และตัวทดนำเข้า $c_1 = 1$ จะได้ผลลัพธ์ $y_2 = 0$ และ $c_2 = \alpha$ วนซ้ำกระบวนการนี้ไปจนถึง y_5

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

บทที่ 5

สรุปผลงานวิจัยและข้อเสนอแนะ

5.1 สรุปผลงานวิจัย

งานวิจัยนี้นำเสนอรูปแบบการแทนจำนวนแบบช่วงแบบใหม่ ที่มีชื่อว่า ระบบแทนช่วงแบบยืดหยุ่น ซึ่งเป็นระบบที่สามารถลดเนื้อที่ที่ใช้ในการแสดงช่วงให้น้อยลงได้ เนื่องจากในระบบใหม่นี้ใช้จำนวนเพียงจำนวนเดียวก็สามารถที่จะแสดงช่วงได้แล้ว โดยเนื้อที่ที่ใช้ในการแสดงช่วงนั้นใช้เนื้อที่ลดลง 25 เปอร์เซ็นต์เมื่อเปรียบเทียบกับระบบแทนช่วงแบบมีเครื่องหมายดั้งเดิม เพราะในระบบแทนช่วงแบบยืดหยุ่นใช้ 3 บิตในการแสดงช่วง ต่างจากระบบแทนช่วงแบบมีเครื่องหมายดั้งเดิมที่ใช้ถึง 4 บิตในการแสดงช่วง โดยใช้อย่างละ 2 บิตในการแสดงค่าต่ำสุดและค่าสูงสุดของช่วง ซึ่งตรงกับวัตถุประสงค์ของงานวิจัยนี้

นอกจากนี้ ยังได้ออกแบบอัลกอริทึมในการแปลงระหว่างระบบแทนจำนวนแบบมีเครื่องหมายและระบบแทนช่วงแบบยืดหยุ่น รวมถึงอัลกอริทึมสำหรับการดำเนินการพื้นฐานทางเลขคณิต ได้แก่ บวก ลบ คูณ และหารสำหรับระบบแทนช่วงแบบยืดหยุ่นอีกด้วย

5.2 ข้อเสนอแนะ

สำหรับในงานวิจัยนี้สิ่งที่จะเสนอแนะคือ การดำเนินการพื้นฐานทางเลขคณิตในระบบใหม่นี้เป็นการคำนวณแบบอนุกรม ถ้าเราพัฒนาให้ระบบนี้สามารถทำการคำนวณแบบขนานได้ จะส่งผลให้ความเร็วในการคำนวณสูงยิ่งขึ้น สาเหตุที่ระบบนี้ไม่สามารถทำการคำนวณแบบขนานได้ เนื่องจากช่วงบางช่วงในระบบนี้ไม่มีสมบัติซ้ำซ้อน เช่น [1, 3] สามารถมีรูปแบบแทนช่วงในระบบนี้คือ $\alpha 1$ เพียงรูปแบบเดียวเท่านั้น ดังนั้นถ้าเราสามารถพัฒนาให้ระบบนี้มีสมบัติซ้ำซ้อนสำหรับทุกช่วงที่ไม่ใช่ศูนย์ การคำนวณแบบขนานย่อมสามารถกระทำได้ ซึ่งปัญหาที่กล่าวมายังคงเป็นปัญหาที่น่าสนใจอยู่

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

รายการอ้างอิง

[1] Skeel, Robert. Roundoff Error and the Patriot Missile. *SIAM News* 25, 11 (July 1992).

[2] Brian Hayes. A Lucid Interval. *American Scientist* 91, 6 (2003): 484-488.

[3] Shettar R., Banakar R.M, and Nataraj, P.S.V.. Design and Implementation of Interval Arithmetic Algorithms. *in. Proc. International Conference on Industrial and Information Systems*, (August 2006): 328-331.

[4] A. Avizienis. Signed-digit number representation for fast parallel arithmetic. *IRE Trans. Comput*. 10 (1961): 389-400.

[5] B. Parhami. Generalized Signed-Digit Number Systems: A Unifying Framework for Redundant Number Representations. *IEEE Trans. Comput.* 39, 1 (January 1990): 89-98.

[6] M. Daumas, G. Melquiond, and C. Muñoz. Guaranteed Proofs Using Interval Arithmetic. *in Proc. International Symposium on Computer Aritmetic*, (June 2005): 188-195.

[7] Moore, Ramon E.. Methods and Applications of Interval Analysis. *Society for Industrial and Applied Mathematics*, Philadelphia, (1979).

[8] R. B. Keafott. Interval computations: Introduction, uses, and resources. *Euromath Bulletin* 2(1) (1996): 95-112.

[9] P. Kornerup. Digit-Set Conversions: Generalizations and Applications. *IEEE Transactions on Computers* 43 (1994): 622-629.

[10] B. Phillips and N. Burgess. Minimal Weight Digit Set Conversions. *IEEE Transactions on Computers* 53 (2004): 666-677.

[11] M. D. Ercegovac and T. Lang. On-the-fly Conversion of Redundant into Conventional Representations. *IEEE Transactions on Computers,* (1985): 895-897.

[12] P. Thienprapasith and A. Surarerks. Flexible interval representation system. *Proceedings of the 10th Annual National Symposium on Computational Science and Engineering (ANSCSE10)*, (2006).

[13] P. Thienprapasith and A. Surarerks. Fundamental arithmetic operations for the flexible interval representation system. *Proceedings of the 10th National Computer Science and Engineering Conference (NCSEC2006)*, (2006).

[14] P. Thienprapasith and A. Surarerks. Multiplication and Division for the flexible interval representation system. *Proceedings of the 11th National Computer Science and Engineering Conference (NCSEC2007)*, (2007).

[15] J. Sukontarach, P. Thienprapasith and A. Surarerks. Modified Arithmetic Algorithm for Flexible Interval Binary Representation System. *Proceedings of the* 12^{th} *Annual National Symposium on Computational Science and Engineering (ANSCSE12)*, (2008).

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

ประวัติผู้เขียนวิทยานิพนธ์

นายพิภพ เทียนประภาสิทธิ์ เกิดเมื่อวันที่ 12 กันยายน พ.ศ. 2525 เรียนจบการศึกษาระดับมัธยมศึกษาตอนต้นและตอนปลายจากโรงเรียนปทุมคงคา เขตคลองเตย จังหวัดกรุงเทพมหานคร เข้ารับการศึกษาต่อในระดับปริญญาบัณฑิต สาขาคณิตศาสตร์ คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย จนสำเร็จการศึกษาในปี พ.ศ. 2547

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย