JSSE
JOURNAL OF SCIENCE & SCIENCE EDUCATION
Faculty of Science, Ubon Ratchathani University
Vol. 7 No. 2 (Jul. – Dec. 2024): 260 - 271
http://doi.org/10.14456/jsse.2024.20

Research Article in Science

A modified Greedy Algorithm for Multidimensional Knapsack Problem

Sujaree Srisaard[1] and Ratee Bojaras[1,*]

[1]Department of Mathematics, Statistics and Computer, Faculty of Science, Ubon Ratchathani University, Ubon Ratchathani

**Email: ratee.b@ubu.ac.th*

Received <28 February 2024>; Revised <11 May 2024>; Accepted <11 May 2024>

Abstract

The Knapsack Problem (KP) is a combinatorial optimization problem that involves selecting items to be placed in a backpack in order to maximize the total value without exceeding the specified capacity. This problem can be formulated as an integer linear programming problem (ILP). In this study, we focused on the multidimensional knapsack problem (MDKP), which has multiple constraints and is more complex to solve. We experimented with different methods for sorting the benefits and evaluated the results using the solver function in Microsoft Excel. We considered 49 examples of 0-1 binary and pure integer knapsack problems. The experimental results showed that selecting items based on modified benefit values (profit/weight ratio) yielded better results than using the difference in benefits, as measured by the mean absolute percentage error (MAPE) across all samples.

Keywords: Multidimensional knapsack problem, Combinatorial optimization, Integer linear programming, Greedy algorithm, The mean absolute percentage error

บทความวิจัยทางวิทยาศาสตร์

การแก้ปัญหาถุงเป้หลายมิติด้วยขั้นตอนวิธีละโมบแบบปรับปรุง

สุจารี ศรีสะอาด[1] และรตี โบจรัส[1,*]

[1]ภาควิชาคณิตศาสตร์ สถิติและคอมพิวเตอร์ คณะวิทยาศาสตร์ มหาวิทยาลัยอุบลราชธานี จังหวัดอุบลราชธานี

**Email: ratee.b@ubu.ac.th*

รับบทความ: 28 กุมภาพันธ์ 2567 แก้ไขบทความ: 11 พฤษภาคม 2567 ยอมรับตีพิมพ์: 11 พฤษภาคม 2567

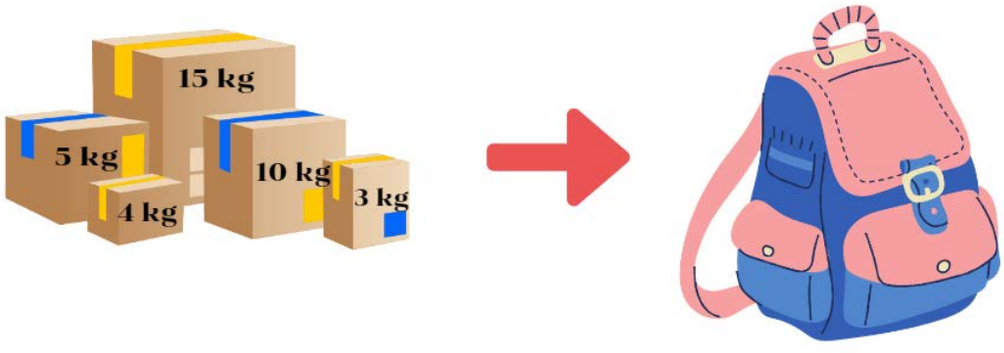
บทคัดย่อ

ปัญหาถุงเป้เป็นปัญหาการหาค่าเหมาะที่สุดเชิงการจัด กล่าวคือ ต้องการเลือกของใส่ถุงเป้โดยเลือกของใส่ถุงให้มีมูลค่ารวมสูงสุดแต่ไม่เกินความจุที่กำหนด ซึ่งเป็นรูปแบบหนึ่งของปัญหากำหนดการเชิงจำนวนเต็ม งานวิจัยนี้ศึกษาปัญหาถุงเป้หลายมิติที่มีหลายเงื่อนไขและมีวิธีแก้ปัญหาซับซ้อนมากขึ้น โดยปรับปรุงการเรียงลำดับการเลือกของใส่ถุงเป้จากวิธีละโมบและนำผลเฉลยที่ได้ตรวจสอบกับผลเฉลยจากการใช้ฟังก์ชัน solver ในโปรแกรมสำเร็จรูปไมโครซอฟท์เอกซ์เซล จากการศึกษาปัญหาแบบ 0-1 และปัญหาแบบจำนวนเต็มแท้ทั้งหมด 49 ตัวอย่าง พบว่าการเรียงลำดับการเลือกของใส่ถุงเป้โดยปรับปรุงค่า benefit (อัตราส่วนกำไรต่อน้ำหนักสิ่งของ) ให้ผลเฉลยดีกว่าการใช้ค่า benefit แบบเดิม ทั้งนี้โดยพิจารณาจากค่าเฉลี่ยของร้อยละความผิดพลาดสัมบูรณ์จากจำนวนตัวอย่างทั้งหมด

คำสำคัญ: ปัญหาถุงเป้หลายมิติ การหาค่าเหมาะที่สุดเชิงการจัด กำหนดการเชิงจำนวนเต็ม วิธีละโมบ ค่าเฉลี่ยของร้อยละความผิดพลาดสัมบูรณ์

บทนำ

ปัญหาถุงเป้ (knapsack problem) เป็นปัญหาการหาค่าเหมาะที่สุดเชิงการจัดที่เป็นปัญหา NP hard ซึ่งมีการศึกษาวิจัยมากกว่า 40 ปี (Akçay, Li and Xu, 2007; Varnamkhasti, 2012; Sajjan *et al.*, 2014; Irmeilyana, Bangun and Izzah, 2017) โดยสามารถอธิบายในรูปแบบคณิตศาสตร์ได้ดังนี้ มีสิ่งของทั้งหมด n ชิ้นโดยแต่ละชิ้นมีมูลค่าและน้ำหนักเป็นจำนวนเต็ม v_i และ w_i ตามลำดับ เป้าหมายของการแก้ปัญหานี้คือ ต้องการพิจารณาว่าจะเลือกสิ่งของชิ้นใดบรรจุในถุงเป้เพื่อให้มูลค่ารวมของสิ่งของมีค่าสูงสุด โดยที่ความจุรวมของสิ่งของต้องไม่เกินความจุรวมที่ถุงเป้บรรจุได้ (W) เราจะเรียกปัญหาถุงเป้แบบที่พิจารณาการเลือกหรือไม่เลือกสิ่งของมาบรรจุในถุงนี้ว่า ปัญหาถุงเป้แบบทวิภาค (binary knapsack problem) เพราะตัวแปรที่พิจารณาจะมีค่าเป็น 0 (หมายถึงไม่เลือก) หรือ 1 (หมายถึงเลือก) เท่านั้น ไม่สามารถกำหนดให้เป็นเศษส่วนได้ (Koohathongsumrit and Luangpaiboon, 2021; Buayen and Werapun, 2018). ปัญหาถุงเป้ได้ถูกนำไปใช้เพื่อจัดการทรัพยากรที่มีอยู่อย่างจำกัด เช่น การจัดตารางงาน การจัดงบประมาณให้แต่ละหน่วยงาน การตัดสินใจลงทุน การเลือกทำโครงการ การเลือกบรรจุภัณฑ์สินค้า เป็นต้น



ภาพที่ 1 ปัญหาถุงเป้

ปัญหาถุงเป้หลายมิติเป็นปัญหาถุงเป้ที่มีเงื่อนไขบังคับมากกว่าหนึ่งเงื่อนไข ซึ่งเป็นปัญหาจริงที่พบได้ในชีวิตประจำวัน เช่น ปัญหาการโหลดสินค้า (cargo loading) เป็นปัญหาการเคลื่อนย้ายสินค้าจากสถานที่หนึ่ง เช่น คลังสินค้า หรือโรงงาน มาบรรจุใส่ตู้คอนเทนเนอร์บนรถบรรทุก เพื่อกระจายสินค้าไปตามจุดหมายต่าง ๆ เช่น ศูนย์กระจายสินค้า หรือจุดจำหน่ายสินค้า โดยก่อนโหลดขึ้นรถบรรทุก สินค้าจะต้องได้รับการบรรจุมาเป็นอย่างดี เพื่อป้องกันการเกิดความเสียหายระหว่างเคลื่อนย้ายและระหว่างขนส่ง ต้องจัดเรียงสินค้าในรถบรรทุกให้เหมาะสม ไม่แน่นเกินเพราะอาจทำให้สินค้าชำรุดได้ จึงต้องพิจารณาเลือกขนาดของสินค้า (ปริมาตร) น้ำหนักหรือเวลาในการจัดเรียงสินค้าให้มีน้ำหนักรวม พื้นที่และเวลาที่จำกัด แต่ให้มีกำไรจากการขนส่งมากที่สุด

เนื่องจากปัญหาถุงเป้ คือ ปัญหาการเลือกของใส่ถุงโดยต้องเลือกของใส่ถุงให้มีมูลค่ารวมสูงสุด จึงเป็นรูปแบบหนึ่งของปัญหากำหนดการเชิงจำนวนเต็ม (Integer linear programming : ILP) การหาผลเฉลยของปัญหาถุงเป้ส่วนมากใช้ขั้นตอนวิธีแบบเดิม (classic algorithm) เช่น วิธีการปัดเศษ (rounding method) วิธีกราฟ (graphical method) วิธีระนาบตัด (cutting-plane method) วิธีขยายและจำกัดเขต (branch-bound algorithm) และขั้นตอนวิธีฮิวริสติก (heuristic algorithm) เช่น ขั้นตอนวิธีแบบละโมบ (greedy algorithm) วิธีเชิงพันธุกรรม (genetic algorithm) วิธีการเรียงลำดับข้อมูล (sorting algorithm) เป็นต้น โดยขั้นตอนวิธีแบบเดิมจะเหมาะกับปัญหาที่ไม่มีความซับซ้อนและฟังก์ชันจุดประสงค์ (objective function) อยู่ในรูปแบบง่าย ส่วนขั้นตอนวิธีฮิวริสติกสามารถแก้ปัญหาบางอย่างที่ขั้นตอนวิธีแบบเดิมไม่สามารถแก้ไขได้ เช่น ปัญหาที่มีความซับซ้อนและใช้เวลานาน อย่างไรก็ตาม ขั้นตอนวิธีฮิวริสติกไม่ได้ให้ผลเฉลยเหมาะที่สุดแต่ให้คำตอบที่ใกล้เคียงกับผลเฉลยเหมาะที่สุดเท่านั้น เพื่อให้เข้าใจมากขึ้นจึงขอเปรียบเทียบวิธีขยายและจำกัดเขตกับวิธีแบบละโมบโดยสังเขป ดังตารางที่ 1

แม้ว่าทั้งสองวิธีจะสามารถแก้ปัญหาถุงเป้ได้ แต่ยังมีความแตกต่างของผลเฉลยเหมาะที่สุด ความซับซ้อนของแต่ละขั้นตอนและการประยุกต์ใช้กับปัญหาที่มีขนาดต่างกัน ทำให้ผู้วิจัยสนใจปรับปรุงขั้นตอนวิธรแบบละโมบเนื่องจากง่ายต่อการพัฒนาแก้ไข นอกจากนี้ได้นำเสนองานวิจัยที่เกี่ยวข้องกับวิธีแก้ปัญหาถุงเป้หลายมิติพอสังเขป ดังตารางที่ 2 โดยผู้อ่านสามารถศึกษารายละเอียดเพิ่มเติมจาก Varnamkhasti (2012)

ตารางที่ 1 เปรียบเทียบวิธีแบบละโมบกับวิธีขยายและจำกัดเขต

รูปแบบขั้นตอน	ข้อดี	ข้อเสีย
วิธีแบบละโมบ	1. เป็นขั้นตอนวิธีที่ตัดสินใจภายใต้คำตอบ (ตัวเลือก) ที่ดีที่สุดในแต่ละขั้นตอน โดยไม่คำนึงถึงผลลัพธ์ในขั้นตอนถัดไป	1. ไม่รับประกันว่าจะพบผลเฉลยเหมาะที่สุด หรือไม่ อาจนำไปสู่การแก้ปัญหาไม่ได้
	2. มีขั้นตอนเรียบง่ายไม่ยุ่งยากซับซ้อน ง่ายต่อการพัฒนาแก้ไข	
	3. ใช้ได้ดีกับปัญหาขนาดเล็ก	
วิธีขยายและจำกัดเขต	1. สามารถหาผลเฉลยเหมาะที่สุดได้	1. มีขั้นตอนซับซ้อนกว่า
	2. ใช้กับปัญหาขนาดใหญ่ได้	2. ต้องใช้ทรัพยากร (หน่วยความจำ) มากกว่า

จากที่กล่าวมาข้างต้นจะเห็นได้ว่า ขั้นตอนวิธีฮิวริสติกสามารถหาผลเฉลยได้ดีเช่นเดียวกันและจากการศึกษางานวิจัยของ Durmuş, Güneri and Incekırık, (2019) ที่ใช้ค่าความแตกต่างของค่า benefit (อัตราส่วนกำไรต่อน้ำหนักสิ่งของ) มาเรียงลำดับการเลือกสิ่งของใส่ถุงเป้ ทำให้ผลเฉลยที่ได้มีความแตกต่างจากผลเฉลยเหมาะที่สุด ผู้วิจัยจึงสนใจแก้ปัญหาถุงเป้หลายมิติโดยใช้ค่าเฉลี่ยของค่า benefit และค่า efficiency (Puchinger, Raidl and Pferschy, 2010) มาเรียงลำดับการเลือกสิ่งของใส่ถุงเป้ใหม่ เพื่อหาผลเฉลย พร้อมทั้งนำผลเฉลยที่ได้มาเปรียบเทียบกับผลเฉลยจากฟังก์ชัน solver ในโปรแกรมสำเร็จรูปไมโครซอฟท์เอกซ์เซล ทั้งนี้ผู้อ่านสามารถศึกษารายละเอียดการใช้ฟังก์ชัน solver จาก Minsan (2021).

ตารางที่ 2 สรุปงานวิจัยที่เกี่ยวข้องกับการแก้ปัญหาถุงเป้

งานวิจัยที่เกี่ยวข้อง	วิธีแก้ปัญหา
Glover (1965)	Exact algorithm
Senju and Toyada (1968)	Heuristic algorithm
Pisinger (1999)	Greedy algorithm
Hassan *et al*. (2005) และ Murat *et al*. (2013)	Genetic algorithm
Mohammad, Saleh and Abdeen (2006)	Brute force algorithm
Akçay, Li and Xu (2007)	Greedy-like heuristic method
Irmeilyana, Bangun and Izzah (2017)	Branch and bound/ Greedy algorithm
Mingo López, Gómez Blas and Arteta Albert (2018)	Genetic operation
Al Etawi and Aburomman (2020)	Dynamic Programming

วัตถุประสงค์การวิจัย

1) แก้ปัญหาถุงเป้หลายมิติที่มีตัวแปรตัดสินใจแบบทวิภาคและตัวแปรตัดสินใจเป็นจำนวนเต็มบวก ด้วยขั้นตอนวิธีละโมบโดยปรับปรุงค่าอัตราส่วนกำไรต่อน้ำหนักสิ่งของ (เรียกว่า ค่า benefit)
2) เปรียบเทียบผลเฉลยที่ได้จากข้อ 1) กับผลเฉลยจากการใช้ฟังก์ชัน solver ในโปรแกรมสำเร็จรูปไมโครซอฟท์เอกซ์เซล

ประเภทของปัญหากำหนดการเชิงจำนวนเต็ม

เนื่องจากปัญหาถุงเป้ คือ ปัญหาการเลือกของใส่ถุงโดยต้องเลือกของใส่ถุงให้มีมูลค่ารวมสูงสุด จึงเป็นรูปแบบหนึ่งของปัญหากำหนดการเชิงจำนวนเต็ม ในหัวข้อนี้จะแสดงประเภทปัญหากำหนดการเชิงจำนวนเต็ม 3 รูปแบบ ดังนี้

1) ปัญหากำหนดการเชิงจำนวนเต็มแท้ (pure integer linear programming problems) คือ ปัญหากำหนดการเชิงจำนวนเต็มที่มีตัวแปรตัดสินใจทุกตัวมีค่าเป็นจำนวนเต็ม ดังสมการที่ (1.1) - (1.3)

ฟังก์ชันจุดประสงค์ $Max \ \ z = \sum_{j=1}^{n} c_j x_j$ (1.1)

ภายใต้เงื่อนไข $\sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij} x_j \le b_i$ (1.2)

$x_j \ge 0$ และเป็นจำนวนเต็ม $j = 1, 2, ..., n$ (1.3)

เมื่อ m คือ จำนวนตัวแปรและ n คือจำนวนเงื่อนไข

2) ปัญหากำหนดการเชิงจำนวนเต็มแบบผสม (mixed integer linear programming problems) คือ ปัญหากำหนดการเชิงจำนวนเต็มที่มีตัวแปรตัดสินใจบางตัวมีค่าเป็นจำนวนเต็มและตัวแปรตัดสินใจอื่นๆ จะมีค่าเป็นจำนวนจริง ดังสมการที่ (2.1) – (2.3)

ฟังก์ชันจุดประสงค์ $Max \ \ z = \sum_{j=1}^{n} c_j x_j$ (2.1)

ภายใต้เงื่อนไข $\sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij} x_j \le b_i$ (2.2)

$x_j \ge 0$, $j = 1, 2, ..., n$ และ x_j เป็นจำนวนเต็ม (สำหรับบาง j) (2.3)

3) ปัญหากำหนดการเชิงจำนวนเต็มแบบทวิภาค หรือปัญหากำหนดการเชิงจำนวนเต็มแบบ 0-1 (binary or 0-1 integer linear programming problems) คือ ปัญหากำหนดการเชิงจำนวนเต็มที่มีตัวแปรตัดสินใจทุกตัวมีค่าเป็น 0 หรือ 1 ดังสมการที่ (3.1) – (3.3)

ฟังก์ชันจุดประสงค์ $Max \ \ z = \sum_{j=1}^{n} c_j x_j$ (3.1)

ภายใต้เงื่อนไข $\sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij} x_j \le b_i$ (3.2)

$x_j \in \{0, 1\}$, $j = 1, 2, ..., n$ (3.3)

ในงานวิจัยนี้เราสนใจศึกษาปัญหาถุงเป้ที่มีค่าตัวแปรแบบทวิภาค (binary integer linear programming problem) และปัญหาถุงเป้ที่มีค่าตัวแปรเป็นจำนวนเต็มแท้ (pure integer linear programming problems)

วิธีดำเนินการวิจัย

ในหัวข้อนี้จะแสดงขั้นตอนวิธีละโมบแบบเดิม ขั้นตอนวิธีละโมบแบบปรับปรุงโดยใช้ค่าเฉลี่ยของค่า benefit และขั้นตอนวิธีละโมบโดยใช้ค่า efficiency จากงานวิจัยของ Puchinger, Raidl and Pferschy (2010) และแสดงการแก้ปัญหาด้วยขั้นตอนวิธีละโมบแบบปรับปรุงเพื่อให้การเรียงลำดับการเลือกของใส่ถุงเป้มีความแตกต่างกัน จากนั้นนำผลเฉลยที่ได้มาตรวจสอบกับผลเฉลยจากการใช้ฟังก์ชัน solver ในโปรแกรมสำเร็จรูปไมโครซอฟท์เอกซ์เซล โดยนำเสนอการแก้ปัญหาถุงเป้หลายมิติที่มีค่าตัวแปรเป็น 0 หรือ 1 และปัญหาที่มีค่าตัวแปรเป็นจำนวนเต็มแท้ จากตัวอย่างมาตรฐานของปัญหาถุงเป้หลายมิติทั่วไป (Multiple Knapsack Problems, 2023) ที่มีตัวแปรไม่เกิน 105 ตัวและเงื่อนไขไม่เกิน 30 เงื่อนไข

ขั้นตอนวิธีละโมบแบบเดิม

ขั้นตอนที่ 1: คำนวณค่า benefit $r_{ij} = c_j / a_{ij}$ ทุกค่า $j = 1, 2, ..., n$, $i = 1, 2, ..., m$

ขั้นตอนที่ 2: จัดลำดับค่า r_{ij} จากมากไปน้อย

ขั้นตอนที่ 3: สิ่งของชิ้นที่ให้ค่า r_{ij} มากสุดจะถูกใส่เข้าไปในเป้ก่อน แล้วตรวจสอบค่าความจุใหม่ ถ้ายังไม่เกินความจุรวมจะใส่ของชิ้นที่ให้ค่า r_{ij} รองลงมาเข้าไปในเป้ แต่ถ้าความจุเกินให้เลือกของชิ้นที่ให้ค่า r_{ij} ลำดับถัดไป

ขั้นตอนที่ 4: ตรวจสอบค่าความจุรวม ถ้าของในเป้มีความจุมากกว่าหรือเท่ากับค่าความจุรวม ให้ออกจากการดำเนินการ ถ้าเป็นกรณีอื่นๆ ให้กลับไปทำในขั้นตอนที่ 3

ขั้นตอนวิธีละโมบแบบปรับปรุงโดยใช้ค่าเฉลี่ยของค่า benefit มีขั้นตอนวิธีการแก้ปัญหาดังต่อไปนี้

ขั้นตอนที่ 1: คำนวณค่า benefit $r_{ij} = c_j/a_{ij}$ ทุกค่า $j = 1, 2, ..., n$, $i = 1, 2, ..., m$

ขั้นตอนที่ 2: คำนวณค่า r_{ij}^* จาก $(\sum_{j=1}^{n} r_{ij})/m$ ทุกค่า $j = 1, 2, ..., n$

ขั้นตอนที่ 3: จัดลำดับค่า r_{ij}^* จากมากไปน้อย ถ้าลำดับเท่ากันให้พิจารณาค่า c_j ที่มากกว่าให้จัดลำดับก่อน

ขั้นตอนที่ 4: สิ่งของชิ้นที่ให้ค่า r_{ij}^* มากสุดจะถูกใส่เข้าไปในเป้ก่อน แล้วตรวจสอบค่าความจุใหม่ ถ้ายังไม่เกินความจุรวมจะใส่ของชิ้นที่ให้ค่า r_{ij}^* รองลงมาเข้าไปในเป้ แต่ถ้าความจุเกินให้เลือกของชิ้นที่ให้ค่า r_{ij}^*ลำดับถัดไป

ขั้นตอนที่ 5: ตรวจสอบค่าความจุรวม ถ้าของในเป้มีความจุมากกว่าหรือเท่ากับค่าความจุรวม ให้ออกจากการดำเนินการ ถ้าเป็นกรณีอื่นๆ ให้กลับไปทำในขั้นตอนที่ 4

ขั้นตอนวิธีละโมบแบบปรับปรุงโดยใช้ค่า efficiency มีขั้นตอนวิธีการแก้ปัญหาดังต่อไปนี้

ขั้นตอนที่ 1: คำนวณค่า efficiency $e_{ij} = a_{ij}/b_i$ (ปรับปรุงค่า benefit) ทุกค่า $j = 1, 2, ..., n$, $i = 1, 2, ..., m$

ขั้นตอนที่ 2: คำนวณค่า e_{ij}^* จาก $c_j/\sum_{i=1}^{m}(a_{ij}/b_i)$ ทุกค่า $j = 1, 2, ..., n$

ขั้นตอนที่ 3: จัดลำดับค่า e_{ij}^* จากมากไปน้อย ถ้าลำดับเท่ากันให้พิจารณาค่า c_j ที่มากกว่าให้จัดลำดับก่อน

ขั้นตอนที่ 4: สิ่งของชิ้นที่ให้ค่า e_{ij}^* มากสุดจะถูกใส่เข้าไปในเป้ก่อน แล้วตรวจสอบค่าความจุใหม่ ถ้ายังไม่เกินความจุรวมจะใส่ของชิ้นที่ให้ค่า e_{ij}^*รองลงมาเข้าไปในเป้ แต่ถ้าความจุเกินให้เลือกของชิ้นที่ให้ค่า e_{ij}^*ลำดับถัดไป

ขั้นตอนที่ 5: ตรวจสอบค่าความจุรวม ถ้าของในเป้มีความจุมากกว่าหรือเท่ากับค่าความจุรวม ให้ออกจากการดำเนินการ ถ้าเป็นกรณีอื่นๆ ให้กลับไปทำในขั้นตอนที่ 4

จากขั้นตอนวิธีละโมบแบบปรับปรุง จะคำนวณค่า r_{ij}^* , e_{ij} และ e_{ij}^* ซึ่งงานวิจัยของ Durmuş, Güneri and Incekırık (2019) ใช้ค่าผลต่างของ benefit แนวแถว ($r_{ij}^d = r_{1j}-r_{2j}$) เพื่อเรียงลำดับการเลือกสิ่งของ ดังตัวอย่างที่ 3.1

<u>ตัวอย่าง 3.1</u> ปัญหาถุงเป้แบบ 0-1 ที่มีจำนวน 2 เงื่อนไข (Two-dimensional 0-1 KP) กำหนดให้เลือกสิ่งของจากทั้งหมด 7 ชิ้นเพื่อให้มีมูลค่ารวมสูงสุด (หน่วยเป็นบาท) ซึ่งมีฟังก์ชันจุดประสงค์

$$\text{Max z} = 20x_1 + 10x_2 + 40x_3 + 20x_4 + 58x_5 + 44x_6 + 26x_7 = \sum_{j=1}^{7} c_j x_j$$

ภายใต้เงื่อนไข

$$15x_1 + 23x_2 + 7x_3 + 9x_4 + 10x_5 + 13x_6 + 5x_7 \leq 50 \quad \text{(เวลา)}$$

$$12x_1 + 15x_2 + 12x_3 + 11x_4 + 25x_5 + 21x_6 + 18x_7 \leq 70 \quad \text{(ค่าน้ำหนักหรือความจุ)}$$

$$14x_1 + 13x_2 + 15x_3 + 14x_4 + 16x_5 + 15x_6 + 7x_7 \leq 100 \quad \text{(ปริมาตร)}$$

$$x_j = \{0, 1\} \text{ เมื่อ } j = 1, 2, \ldots, 7$$

ใช้ขั้นตอนวิธีละโมบแบบปรับปรุงโดยค่าเฉลี่ยของค่า benefit และค่า efficiency แสดงค่าในตารางที่ 3-4

ตารางที่ 3 ค่า benefit ทุกค่า $j = 1, 2, \ldots, 7$ และ $i = 1, 2, 3$

x_j	c_j	a_{1j}	a_{2j}	a_{3j}	r_{1j}	r_{2j}	r_{3j}
x_1	20	15	12	14	1.33	1.67	1.43
x_2	10	23	15	13	0.43	0.67	0.77
x_3	40	7	12	15	5.71	3.33	2.67
x_4	20	9	11	14	2.22	1.82	1.43
x_5	58	10	25	16	5.80	2.32	3.63
x_6	44	13	21	15	3.38	2.10	2.93
x_7	26	5	18	7	5.20	1.44	3.71

หมายเหตุ r_{1j}=c_j/a_{1j} เมื่อ c_j คือ สัมประสิทธิ์ของตัวแปร x_j ในฟังก์ชันจุดประสงค์

a_{1j} คือ สัมประสิทธิ์ของตัวแปร x_j ในเงื่อนไขที่ 1 เมื่อ $j = 1, 2, \ldots, 7$

เช่นเดียวกันกับค่า r_{2j} และ r_{3j}

ตารางที่ 4 ค่า efficiency ทุกค่า $\mathrm{j} = 1, 2, \ldots, 7$ และ $i = 1, 2, 3$

x_j	c_j	a_{1j}	a_{2j}	a_{3j}	e_{1j}	e_{2j}	e_{3j}
x_1	20	15	12	14	0.30	0.17	0.14
x_2	10	23	15	13	0.46	0.21	0.13
x_3	40	7	12	15	0.14	0.17	0.15
x_4	20	9	11	14	0.18	0.16	0.14
x_5	58	10	25	16	0.20	0.36	0.16
x_6	44	13	21	15	0.26	0.30	0.15
x_7	26	5	18	7	0.10	0.26	0.07

ตารางที่ 5 ลำดับ (rank) จากค่า ${r_{ij}}^*$, ${e_{ij}}^*$ และ ${r_{ij}}^d = r_{1j} - r_{2j} - r_{3j}$ (ค่าผลต่างของ benefit แนวแถว)

x_j	${r_{ij}}^*$	rank	${e_{ij}}^*$	rank	${r_{ij}}^d$	rank
x_1	1.48	6	42.42	6	1.76	7
x_2	0.62	7	14.83	7	1.00	4
x_3	3.90	2	128.44	1	0.29	3
x_4	1.82	5	59.32	5	1.02	5
x_5	3.92	1	104.10	2	0.15	2
x_6	2.80	4	78.57	3	1.64	6
x_7	3.45	3	72.80	4	0.04	1

จากตัวอย่าง 3.1 ผลเฉลยที่ได้จากลำดับการเลือกสิ่งของด้วยค่า ${e_{ij}}^*$ มีค่าเท่ากับผลเฉลยจากฟังก์ชัน solver และให้ค่าฟังก์ชันจุดประสงค์เท่ากันคือ 162 ซึ่งมากกว่าการใช้ลำดับการเลือกสิ่งของด้วยค่า ${r_{ij}}^*$ และ ${r_{ij}}^d$ ตามลำดับ ดังตารางที่ 6

ตารางที่ 6 ผลเฉลยของปัญหาถุงเป้หลายมิติและค่าฟังก์ชันจุดประสงค์ จากลำดับการเลือกสิ่งของโดยใช้ค่า ${r_{ij}}^*$, ${e_{ij}}^*$ ${r_{ij}}^d$ และจากฟังก์ชัน solver (คำตอบ 0 หมายถึง ไม่เลือก / 1 หมายถึงเลือก)

x_j	ผลเฉลยจาก ${r_{ij}}^*$	ผลเฉลยจาก ${e_{ij}}^*$	ผลเฉลยจาก ${r_{ij}}^d$	ผลเฉลยจาก solver
x_1	0	0	0	0
x_2	0	0	1	0
x_3	1	1	1	1
x_4	1	1	0	1
x_5	1	1	1	1
x_6	0	1	0	1
x_7	1	0	1	0
Z= มูลค่ารวม (บาท)	144	162	134	162

ตารางที่ 7 และ 8 แสดงผลเฉลยของปัญหาถุงเป้หลายมิติที่มีค่าตัวแปรเป็น 0 หรือ 1 และมีค่าตัวแปรเป็นจำนวนเต็ม จากตัวอย่างมาตรฐานของปัญหาถุงเป้หลายมิติทั่วไปที่มีตัวแปร n ตัว จำนวนเงื่อนไข m เงื่อนไข ผลเฉลยจากขั้นตอนวิธีละโมบแบบปรับปรุง ที่ใช้การเรียงลำดับการเลือกของที่แตกต่างกัน คือ ค่าเฉลี่ย benefit (${r_{ij}}^*$) ผลต่างของค่า benefit (${r_{ij}}^d$) และค่า efficiency (${e_{ij}}^*$) โดยเปรียบเทียบกับผลเฉลยจากฟังก์ชัน solver รูปที่ 2 แสดงค่าเฉลี่ยของร้อยละความผิดพลาดสัมบูรณ์จากลำดับการเลือกของที่แตกต่างกันทั้งสามแบบ เทียบกับผลเฉลยจาก Solver ของปัญหาถุงเป้หลายมิติที่มีค่าตัวแปรเป็น 0 หรือ 1 และค่าตัวแปรเป็นจำนวนเต็ม ตามลำดับ

ตารางที่ 7 ผลเฉลยของปัญหาถุงเป้หลายมิติที่มีค่าตัวแปรเป็น 0 หรือ 1 (ปัญหา BIN) จากตัวอย่างมาตรฐาน ORLib Weing Weish และ Sento

ปัญหา	n	m	ผลเฉลยจาก r_{ij}^*	ผลเฉลยจาก r_{ij}^d	ผลเฉลยจาก e_{ij}^*	ผลเฉลยจาก Solver
1	28	2	123309	104799	139278	141278
2	28	2	102932	87371	128322	130773
3	28	2	62620	62380	93278	95677
4	28	2	111291	104799	115321	119337
5	28	2	62620	62380	93278	98796
6	28	2	102932	87136	128212	130623
7	29	2	82087	75013	93962	98251
8	105	2	1081247	727159	1091507	1092911
9	105	2	459738	354475	620060	623952
10	27	4	2792	2706	2792	3073
11	28	4	1644	375	2840	2852
12	34	4	2528	2913	2825	3164
13	35	4	2528	2913	2824	3164
14	30	5	4090	1571	4534	4554
15	30	5	3398	1330	4483	4531
16	30	5	3634	1430	4021	4506
17	30	5	3121	1277	4475	4531
18	30	5	2451	1151	4514	4514
19	40	5	5169	1519	5461	5557
20	40	5	5091	2164	5478	5542
21	40	5	5201	1519	5478	5567
22	40	5	2415	1110	5246	5246
23	50	5	4953	1351	6265	6339
24	50	5	3160	1263	5500	5643
25	50	5	4955	1111	6008	6339
26	50	5	3649	1082	6025	6159
27	60	5	4207	1062	6845	6954
28	60	5	4174	887	7213	7458
29	60	5	3390	1142	6828	7289
30	60	5	7668	1989	8293	8633
31	70	5	7928	1062	8175	9580
32	70	5	4243	1242	7051	7698
33	70	5	6619	2911	9380	9450
34	70	5	5199	1341	8714	9074
35	80	5	6620	2466	8836	8729
36	80	5	6219	2635	7493	8079
37	80	5	8778	3737	9818	9979
38	80	5	7643	2815	9787	9787
39	90	5	6934	2380	9278	9584
40	90	5	7190	2514	9764	9819
41	90	5	6147	2705	8671	9492
42	90	5	5563	2585	8638	9410

ปัญหา	n	m	ผลเฉลยจาก r_{ij}^*	ผลเฉลยจาก r_{ij}^d	ผลเฉลยจาก e_{ij}^*	ผลเฉลยจาก Solver
43	90	5	9278	3128	10819	11187
44	20	10	1525	1629	1925	2139
45	100	15	3641	3382	3653	3741
46	37	30	959	294	1022	1035
47	40	30	433	136	504	762
48	60	30	7429	1311	7606	7772
49	60	30	8478	1046	8709	8659

ตารางที่ 8 ผลเฉลยของปัญหาถุงเป้หลายมิติที่มีค่าตัวแปรเป็นจำนวนเต็ม (ปัญหา GIN) จากตัวอย่างมาตรฐาน ORLib Weing Weish และ Sento

ปัญหา	n	m	ผลเฉลยจาก r_{ij}^*	ผลเฉลยจาก r_{ij}^d	ผลเฉลยจาก e_{ij}^*	ผลเฉลยจาก Solver
1	28	2	260810	259650	712800	712800
2	28	2	258170	258170	594000	594000
3	28	2	124920	124020	356400	356400
4	28	2	259490	259100	712800	712800
5	28	2	126240	125340	356400	356400
6	28	2	244530	244530	584830	584830
7	29	2	105444	68652	123200	168670
8	105	2	6432000	6432000	14871000	17691816
9	105	2	1072000	1072000	2478500	2921634
10	27	4	3773	1078	5152	5157
11	28	4	3463	800	3456	3482
12	34	4	3371	1705	3363	5675
13	35	4	3371	3233	3363	5675
14	30	5	6180	6180	6767	12810
15	30	5	9094	6608	8896	10356
16	30	5	8490	8542	8542	10356
17	30	5	4726	2511	6390	7350
18	30	5	4787	2456	6030	7083
19	40	5	11146	7358	10954	13206
20	40	5	11454	5544	10816	11599
21	40	5	12028	7358	8301	13972
22	40	5	10496	10329	9120	12590
23	50	5	16428	13311	17283	20506
24	50	5	15017	12824	15482	17218
25	50	5	16428	13400	17283	20144
26	50	5	16239	13276	16852	20008
27	60	5	20051	17864	20541	20506
28	60	5	12252	4000	11122	17312
29	60	5	16924	13080	15674	20713
30	60	5	27295	27360	27362	37024
31	70	5	23340	18370	23303	32384
32	70	5	22406	19936	22969	28022
33	70	5	15224	12370	11098	26262

ปัญหา	n	m	ผลเฉลยจาก r_{ij}^*	ผลเฉลยจาก r_{ij}^d	ผลเฉลยจาก e_{ij}^*	ผลเฉลยจาก Solver
34	70	5	14010	15026	11098	25340
35	80	5	26083	23832	27334	33956
36	80	5	25842	22400	26072	32294
37	80	5	27424	21600	27252	37182
38	80	5	17412	9170	15276	30174
39	90	5	29540	26482	30622	37546
40	90	5	29540	26912	30622	38164
41	90	5	29540	26472	30622	37114
42	90	5	29540	26446	30622	37444
43	90	5	28140	11200	19932	48024
44	20	10	1068	1627	1926	2207
45	100	15	2760	2540	3096	3932
46	37	30	834	210	1016	1244
47	40	30	307	129	363	825
48	60	30	12454	1691	12348	15579
49	60	30	21858	2782	22500	26286

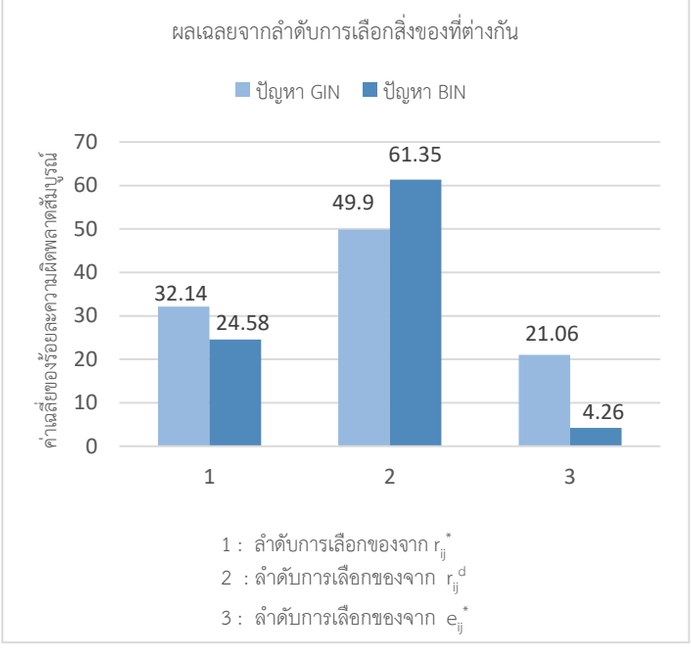
จากตารางที่ 7 และ 8 นำผลเฉลยจากการเรียงลำดับทั้งสามแบบมาหาค่าความผิดพลาดสัมบูรณ์โดยเทียบกับผลเฉลยจาก Solver แล้วนำมาหาค่าเฉลี่ยของร้อยละความผิดพลาดสัมบูรณ์ (the mean absolute percentage error : MAPE)

$$\text{MAPE} = \sum_{i=1}^{N} \left| \frac{x_i - A_i}{A_i} \right| \times \frac{100}{N} \qquad (4)$$

เมื่อ x_i คือ ผลเฉลยที่ได้จากขั้นตอนวิธีละโมบแบบปรับปรุง

A_i คือ ผลเฉลยจาก Solver

N คือ จำนวนตัวอย่าง (ในงานวิจัยนี้ $N = 49$)



ภาพที่ 2 ค่าเฉลี่ยของร้อยละความผิดพลาดสัมบูรณ์ที่ใช้การเรียงลำดับการเลือกของที่แตกต่างกัน

สรุปผลการวิจัยและข้อเสนอแนะจากการวิจัย

ขั้นตอนวิธีละโมบแบบปรับปรุงที่นำมาใช้แก้ปัญหาถุงเป้หลายมิติเป็นขั้นตอนวิธีการแก้ปัญหาที่ไม่ยุ่งยากซับซ้อน ด้วยการปรับค่า benefit และใช้ค่า efficiency สามารถหาผลเฉลยได้ใกล้เคียงกับผลเฉลยจากฟังก์ชัน solver และให้ผลเฉลยดีกว่าการใช้ค่า benefit แบบเดิม จากภาพที่ 2 พบว่าผลเฉลยจากลำดับการเลือกสิ่งของด้วยค่า e_{ij}^* ในปัญหาถุงเป้หลายมิติชนิดที่ตัวแปรเป็น 0 หรือ 1 มีค่าเฉลี่ยของร้อยละความผิดพลาดสัมบูรณ์น้อยกว่า 5 ซึ่งให้ผลเฉลยเหมาะที่สุดใกล้เคียงกับผลเฉลยจากฟังก์ชัน solver ส่วนผลเฉลยจากลำดับการเลือกสิ่งของด้วยค่า r_{ij}^* และ r_{ij}^d มีค่าเฉลี่ยของร้อยละความผิดพลาดสัมบูรณ์เป็น 24.58 และ 61.35 ตามลำดับ สำหรับปัญหาถุงเป้หลายมิติที่มีค่าตัวแปรเป็นจำนวนเต็มพบว่าผลเฉลยจากลำดับการเลือกสิ่งของด้วยค่า e_{ij}^* มีค่าเฉลี่ยของร้อยละความผิดพลาดสัมบูรณ์น้อยที่สุดเช่นกัน เมื่อเปรียบเทียบผลเฉลยชนิดที่ตัวแปรเป็น 0 หรือ 1 กับตัวแปรที่มีค่าเป็นจำนวนเต็ม พบว่า ลำดับการเลือกสิ่งของด้วยค่า e_{ij}^* ในปัญหาที่ตัวแปรเป็น 0 หรือ 1 ให้ผลเฉลยใกล้เคียงกับผลเฉลยจากฟังก์ชัน solver มากที่สุด

ถึงแม้ว่าค่าเฉลี่ยของร้อยละความผิดพลาดสัมบูรณ์จากลำดับการเลือกสิ่งของด้วยค่า e_{ij}^* ในปัญหาถุงเป้หลายมิติที่มีค่าตัวแปรเป็นจำนวนเต็มจะดีกว่าผลเฉลยจากลำดับการเลือกสิ่งของด้วยค่า r_{ij}^* และ r_{ij}^d แต่ยังพบว่าค่าเฉลี่ยของร้อยละความผิดพลาดสัมบูรณ์มากกว่า 10% ผู้วิจัยต้องปรับปรุงค่า e_{ij}^* หรือศึกษาวิธีอื่นเพื่อให้ผลเฉลยใกล้เคียงกับผลเฉลยจากฟังก์ชัน solver มากขึ้น

References

Al Etawi, N. A., and Aburomman, F. T. (2020). 0/1 knapsack problem: greedy vs. Dynamic-programming. **International Journal of Advanced Engineering and Management Research**, 5(2), 1-10.

Akçay, Y., Li, H. and Xu, S. H. (2007). Greedy algorithm for the general multidimensional knapsack problem. **Annals of operations research**, 150, 17–29.

Berberler, M. E., Guler, A., and Nurıyev, U. G. (2013). A genetic algorithm to solve the multidimensional knapsack problem. **Mathematical and Computational Applications**, 18(3), 486-494.

Buayen, P. and Werapun, J. (2018). Parallel time–space reduction by unbiased filtering for solving the 0/1-Knapsack problem. **Journal of Parallel and Distributed Computing**, 122, 195 - 208.

Durmuş, B., Güneri, O. I. and Incekırık, A. (2019). Comparison of Classic and Greedy Heuristic Algorithm Results in Integer Problems. **Mugla Journal of Science and Technology**, 5(1), 34-42.

Glover, F. (1965). A multiphase-dual algorithm for the zero-one integer programming problem. **Operation Research**, 13, 879–919.

Hassan, R., Cohanim, B., De Weck, O., and Venter, G. (2005, April). A comparison of particle swarm optimization and the genetic algorithm. In **46th AIAA/ASME/ASCE/AHS/ASC structures, structural dynamics and materials conference** (pp. 2005-1897). Texas.

Irmeilyana, I., Bangun, P. B. J. and Izzah, H. (2017). Solution of Multiple Constraints Knapsack Problem (MCKP) by using Branch and Bound and Greedy Algorithm. **Journal of Modeling and Optimization**, 9(2), 112-119.

Koohathongsumrit, N. and Luangpaiboon, P. (2021), An Integrated FAHP–ZODP Approach for Strategic Marketing Information System Project Selection. **Managerial and Decision Economics**, 43(6), 1792-1809.

Mingo López, L. F., Gómez Blas, N., and Arteta Albert, A. (2018). Multidimensional knapsack problem optimization using a binary particle swarm model with genetic operations. **Soft Computing**, 22, 2567-2582.

Minsan, P. (2021). Comparing Methods of Optimization in Solver of Microsoft Excel 2007 and 2019: A Case Study of Statistical Models, **The journal of KMUTNB**, 31(3), 496-511.

Mohammad, A., Saleh, O., and Abdeen, R. A. (2006). Occurrences algorithm for string searching based on brute-force algorithm. **Journal of Computer Science**, 2(1), 82-85.

Multiple Knapsack Problems (2023). Retrieved 5 September 2023, from OR-library: **http://people.brunel.ac.uk/**

Pisinger, D. (1999). Linear Time Algorithms for Knapsack Problems with Bounded Weights. **Journal of Algorithms**. 33(1), 1-14.

Puchinger, J., Raidl, G. R., and Pferschy, U. (2010). The multidimensional knapsack problem: Structure and algorithms. **INFORMS Journal on Computing**, 22(2), 250-265.

Sajjan, S. P., Roogi, R., Badiger, V. and Amaragatti, S. (2014). A New Approach to Solve Knapsack Problem. **Oriental Journal of Computer Science & Technology**, 7(2), 219.

Senju, S. and Toyada, Y. (1968). An approach to linear programming problems with 0–1 variables, **Management Science**, 15(4), 196–207.

Varnamkhasti, M. J. (2012). Overview of the Algorithms for Solving the Multidimensional Knapsack Problems. **Advanced Studies in Biology**, 4(1), 37-47.