รายงาน

เรื่อง: อัลกอริทึม Bellman-Ford และการประยุกต์ใช้งานในระบบเครือข่าย

บทนำ

ในการคำนวณหาวิธีเดินทางที่ประหยัดทรัพยากรมากที่สุด เช่น เวลา ต้นทุน หรือจำนวนฮอป (hop) ในระบบเครือข่าย อัลกอริทึมในการค้นหาเส้นทางที่สั้นที่สุด (Shortest Path Algorithm) ถือเป็นหัวใจสำคัญ โดยเฉพาะอย่างยิ่งเมื่อพิจารณาถึงเครือข่ายที่มีค่าน้ำหนักของเส้นทางที่สามารถติดลบได้ ซึ่งอัลกอริทึม Bellman-Ford เป็นหนึ่งในอัลกอริทึมที่รองรับกรณีดังกล่าวได้อย่างมีประสิทธิภาพ

แนวคิดเชิงทฤษฎีของอัลกอริทึม Bellman-Ford

นิยามเบื้องต้น

Bellman-Ford เป็นอัลกอริทึมที่ใช้แก้ปัญหา **Single-Source Shortest Path** ซึ่งหมายถึง การหาเส้นทางที่สั้นที่สุดจาก **จุดเริ่มต้นเดียว** ไปยัง **โหนดอื่น ๆ ทั้งหมดในกราฟ**

สิ่งที่ทำให้อัลกอริทึมนี้โดดเด่นคือ **สามารถจัดการกับค่าน้ำหนักติดลบได้** ต่างจากอัลกอริทึมอย่าง Dijkstra ที่ไม่สามารถรองรับค่าน้ำหนักลบได้ เนื่องจากมีโอกาสทำให้การประเมินระยะทางผิดพลาด

แนวคิดหลักของ Bellman-Ford

หลักสำคัญคือการ “Relax” เส้นทาง นั่นคือ การพิจารณาว่าเส้นทางที่เคยประเมินไว้สามารถปรับให้ดีขึ้นได้หรือไม่แต่ละรอบการทำงานจะทำการตรวจสอบขอบ (edge) ทั้งหมดของกราฟ และอัปเดตค่าระยะทางสั้นที่สุดที่ทราบอยู่ทำซ้ำทั้งหมด V - 1 ครั้ง (V คือจำนวนโหนดในกราฟ) เพื่อให้มั่นใจว่าเส้นทางทั้งหมดถูกพิจารณาแล้วในรอบสุดท้าย ตรวจสอบว่ามีการเปลี่ยนแปลงระยะทางอีกหรือไม่ หากยังมี แสดงว่ามี “วงจรลูปติดลบ” (Negative Cycle) ซึ่งเป็นสิ่งที่ต้องระวัง

ตัวอย่างการทำงาน

พิจารณากราฟที่ประกอบด้วย 4 โหนด และมีเส้นทางดังนี้:

- A → B (4)
- A → C (5)
- B → C (-3)
- C → D (2)

เริ่มจากโหนด A

กำหนดค่าเริ่มต้น:

distance[A] = 0

distance[B, C, D] = ∞

รอบที่ 1:

A → B → distance[B] = min(∞, 0+4) = 4

A → C → distance[C] = min(∞, 0+5) = 5

B → C → distance[C] = min(5, 4-3) = 1

C → D → distance[D] = 1+2 = 3

รอบที่ 2 และ 3:

ไม่มีการเปลี่ยนแปลงใด ๆ เพิ่มเติม

ผลลัพธ์ระยะทางจาก A:

A = 0

B = 4

C = 1

D = 3

การประยุกต์ใช้ในระบบเครือข่าย

อัลกอริทึม Bellman-Ford ถูกนำมาใช้อย่างกว้างขวางใน **ระบบเครือข่ายคอมพิวเตอร์** โดยเฉพาะในการจัดเส้นทางการส่งข้อมูล (Routing) ให้มีประสิทธิภาพมากที่สุด

โปรโตคอล RIP (Routing Information Protocol)

RIP คือหนึ่งใน **Distance Vector Routing Protocol** ที่อิงหลักการของ Bellman-Ford

หลักการของ RIP:

แต่ละเราเตอร์จะเก็บ “Routing Table” ที่ระบุว่า จะไปยังแต่ละโหนดในเครือข่ายได้อย่างไร และใช้ค่า hop เท่าใด

เราเตอร์จะแลกเปลี่ยน Routing Table กับเราเตอร์ที่เป็นเพื่อนบ้านทุก ๆ 30 วินาที

เมื่อได้รับข้อมูลใหม่ เราเตอร์จะใช้การคำนวณแบบ Bellman-Ford เพื่อตัดสินใจว่าเส้นทางใหม่ดีกว่าหรือไม่

ตัวอย่างสถานการณ์:

เรามีเราเตอร์ A, B, และ C ที่เชื่อมต่อกันดังนี้:

A ↔️ B (hop = 1)

B ↔️ C (hop = 1)

A ↔️ C (hop = 5)

ตอนเริ่มต้น A จะส่งข้อมูลไป C โดยตรง (hop = 5)
เมื่อ B แจ้งกับ A ว่าสามารถไปยัง C ได้ภายใน 1 hop
A จะคำนวณว่า หากไป B (1 hop) แล้วต่อไป C (1 hop) ⟶ รวม = 2 hops
ดังนั้น A จะเลือกเส้นทาง A ⟶ B ⟶ C แทน ซึ่งประหยัดกว่า

ปัญหา “Count to Infinity” และวิธีแก้ไข

Bellman-Ford มีปัญหาในการใช้งานจริงหากมีการตัดการเชื่อมต่อ เช่น:

หากลิงก์จาก A ไปยัง C ขาด แล้ว B ยังแจ้งว่า “ฉันยังไป C ได้อยู่นะ”

A จะเข้าใจผิดว่าตนยังสามารถไปยัง C ผ่าน B ได้ ⟶ เกิด “Count to Infinity”

วิธีแก้ไข:

Split Horizon: เราเตอร์จะไม่แจ้งเส้นทางไปยังโหนดที่ตนเองเรียนรู้มาจากเราเตอร์นั้น

Poison Reverse: แจ้งค่า hop เป็น ∞ เมื่อเส้นทางนั้นไม่สามารถใช้ได้อีก

ข้อดีและข้อจำกัด

ข้อดี	ข้อจำกัด
รองรับค่าน้ำหนักลบ	ทำงานช้าเมื่อกราฟมีจำนวนโหนดมาก
ตรวจจับวงจรลูปติดลบได้	ไม่เหมาะสำหรับกราฟขนาดใหญ่แบบ real-time
ง่ายต่อการติดตั้งแบบ Distributed	มีปัญหา Count to Infinity ในการใช้งานจริง

สรุป

อัลกอริทึม Bellman-Ford เป็นเครื่องมือที่มีประโยชน์มาก โดยเฉพาะในการประยุกต์ใช้กับระบบเครือข่าย เช่น โปรโตคอล RIP ด้วยความสามารถในการรองรับค่าน้ำหนักลบ และโครงสร้างการทำงานแบบกระจาย (Distributed) อย่างไรก็ตาม การใช้งานจริงจำเป็นต้องมีการออกแบบที่ระมัดระวัง เพื่อหลีกเลี่ยงปัญหาที่อาจเกิดขึ้น เช่น การนับระยะทางแบบไม่มีที่สิ้นสุด