Algorithm Design and Analysis

วิชาบังคับก่อน: 204251 และ 206281
ผู้สอน: ตอน 1 ผศ. เบญจมาศ ปัญญางาม เรียน ห้อง 207
ตอน 2 ผศ. ดร. จักริน ชวชาติ + เรียน ห้อง 209

วันสอบปลายภาค : วันศุกร์ที่ 13 พ.ย. 2563
เวลา 8:00 -11:00 น. (ตามประกาศมหาวิทยาลัย)

บทที่ 11
NP-Completeness Part I

Classes of Problems

ปัญหาโดยทั่วไปส่วนใหญ่สามารถแก้ได้ใน Polynomial time

ปัญหา	เวลาการทำงานของอัลกอริทึม
Search	**O(log n)**
Sorting	**O(n log n)**
Minimum Spanning Tree	**O(E log E)**
O/1 knapsack	**$O(2^n)$**

Classes of Problems

อัลกอริทึมแบ่งเป็น 2 กลุ่ม

- Polynomial-time algorithm : อัลกอริทึมที่มีเวลาในการทำงานเป็น $O(n^k)$
  - On inputs of size, their worst-case running time is $O(n^k)$ for some constant k
  - ตัวอย่าง Polynomial time: O(1) , O(log n), O(n), O(n log n), $O(n^2)$, $O(n^k)$
- Exponential-time algorithm อัลกอริทึมที่มีเวลาในการทำงานโตเร็วเกินฟังก์ชันพหุนาม
  - $O(2^n)$, $O(n^n)$, $O(n!)$

Tractable or Intractable Problems

- ปัญหาง่ายหรือเรียกว่า Tractable Problem
  - มี Polynomial-time algorithm แก้ปัญหานั้นได้
- ปัญหายากหรือเรียกว่า Intractable Problem
  - พิสูจน์ได้ว่าต้องใช้ Exponential-time algorithm ในการหาคำตอบของปัญหานั้น เช่น ปัญหาการหาวิธีเดินหมากรุกให้ชนะ ปัญหาการหยุดทำงานของโปรแกรม (Halting Problem) ซึ่งเป็นปัญหาที่ไม่มีอัลกอริทึมใดๆ หาคำตอบได้ (noncomputable Problem)

Tractable or Intractable Problems

- คำถาม : ปัญหาหนึ่งพบว่ามีอัลกอริทึมที่ใช้เวลาในการแก้ปัญหาเป็น 2^n เราจะสรุปได้หรือไม่ว่าปัญหานั้นเป็นปัญหายาก
- คำตอบ : ไม่ได้ เพราะ
  - แม้ว่าปัจจุบันเรายังไม่สามารถหา Polynomial-time algorithm มาแก้ปัญหานั้นได้ก็ตาม แต่ก็ยังไม่สามารถพิสูจน์ได้ว่าไม่มีอัลกอริทึมแบบ Polynomial-time ในการหาคำตอบของปัญหานี้

Decision Problem

- เพื่อให้จัดกลุ่มปัญหาให้ง่ายขึ้น จะสนใจเฉพาะกลุ่มปัญหาการตัดสินใจ (Decision Problem) เท่านั้น
- Decision Problem : ปัญหาที่ให้คำตอบว่า yes หรือ no
  - ปัญหาที่มีสองคำตอบเท่านั้น เช่น ใช่หรือไม่ใช่ มีหรือไม่มี ได้หรือไม่ได้ จริงหรือเท็จ เช่น

“Is x a multiple of y?”

ถ้า input คือ x=45 และ y=15 คำตอบคือ yes

ถ้า input คือ x= 45 และ y=13 คำตอบคือ no

Class P

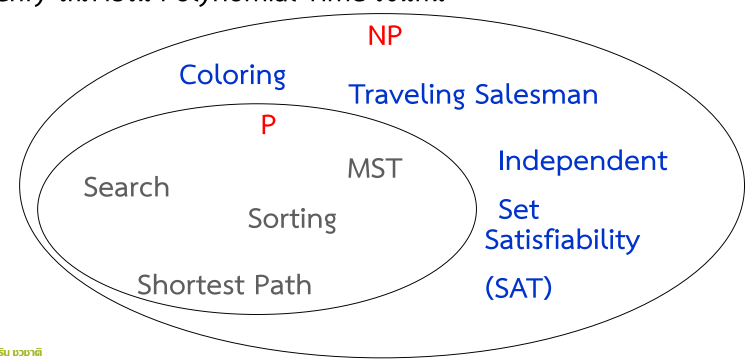
- กลุ่ม Decision problem ที่สามารถ<u>หาคำตอบ</u>ได้ด้วย Polynomial-time algorithm

ตัวอย่าง Decision problem

- ปัญหา Minimum Spanning Tree (หา MST จากกราฟ G)
  - กราฟ G มี Spanning Tree ที่มีค่าผลรวม weight ไม่เกิน K หรือไม่
- ปัญหา Single Shortest Path
- กราฟ G มีเส้นทางที่เริ่มต้นจากโนด s ไปยังโนด t แล้วมีผลรวมระยะทางไม่เกิน k หรือไม่

Class NP

- กลุ่ม Decision problem ที่สามารถ<u>ทวนสอบ</u> (Verify) ได้ในเวลา Polynomial-time
- หาก Q เป็นปัญหาในกลุ่ม P ปัญหา Q จะอยู่ในกลุ่ม NP ด้วยเพราะปัญหาในกลุ่ม P เป็นปัญหาที่แก้ได้ภายใน Polynomial Time ดังนั้นจะสามารถ verify ได้ภายใน Polynomial Time เช่นกัน



Verify Algorithm

- ปัญหา Single Shortest Path (SSP) จาก S ไป t
  - กราฟ G มีเส้นทางที่เริ่มต้นจากโนด s ไปยังโหนด t แล้วมีผลรวมระยะทางไม่เกิน k หรือไม่
- ต้องมีชุดคำตอบ(solution) ที่ต้องการ verity เช่น เส้นทาง $<s, v_1, v_2, \ldots, t>$
- Verify_SSP(*G=(V,E),s,t, k, solution*) {

  - ตรวจสอบโหนดแรกของเส้นทางคือ s โหนดสุดท้ายของเส้นทางคือ t
  - ตรวจสอบว่าทุกเส้นในชุด solution เป็นเส้นเชื่อมที่เป็นสมาชิกของเส้นเชื่อม E ในกราฟ G ทั้งหมดหรือไม่
  - ตรวจสอบผลรวมระยะทางต้อง $\leq k$

}
- หา Time Complexity เป็น polynomial time หรือไม่?

Verify Algorithm

เช่น Decision Problem ของปัญหา Search

Prolem1: “x อยู่ในแถวลำดับ A หรือไม่”

ตัวอย่าง verify algorithm

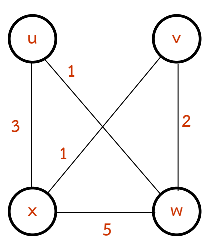
```
Search_VerifyProblem1 (A[], x) {
    for i=1 to size(A)
        if A[i]= x return true
    return false
}
```

Time = O(n)

ดังนั้น Problem1 เป็น NP problem

Practice :Traveling Salesman Problem

- กำหนด G=(V,E) เป็นกราฟแบบมีน้ำหนักและไม่ระบุทิศทาง
- มีวงจรในกราฟ G ซึ่งผ่านโนดแต่ละโนดเพียง 1 ครั้ง โดยผลรวมน้ำหนักของวงจรนี้มีค่าไม่เกิน k หรือไม่



คำตอบคือ ⟨u, w, v, x, u⟩
โดยผลรวมน้ำหนักคือ 1 + 2 + 1 + 3 =7

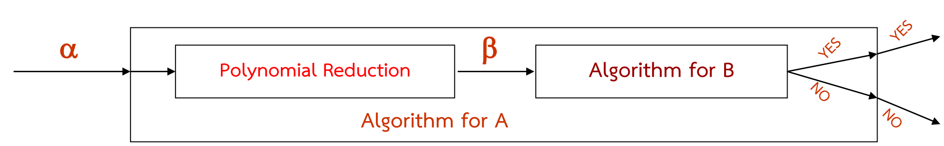
คำถาม:
ปัญหานี้เป็น NP หรือไม่ ?

Reduction

- ก่อนกล่าวถึงปัญหา NP complete จะอธิบายการลดรูปปัญหาก่อน
- กำหนด Decision Problem A และ B
  - หากสามารถลดรูปปัญหา A ไปเป็นปัญหา B ก็แสดงว่าเรามีวิธี<u>เปลี่ยน</u> Instances (α) ของปัญหา A ไปเป็น Instances (β) ของปัญหา B ได้ภายใน polynomial time

$$A \leq_p B$$

  - จากนั้นใช้อัลกอริทึมที่แก้ปัญหา B เพื่อหาคำตอบของ B
  - คำตอบที่ได้จะเป็นคำตอบของ A ด้วย (Yes/No)



Reduction

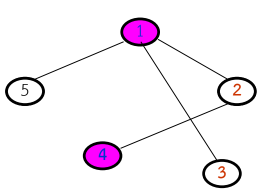
- หากสามารถ reduce ปัญหา A ไปยัง ปัญหา B ภายใน polynomial time ($A \le_p B$) สรุปได้ว่า
  - ปัญหา A ไม่ยากกว่าปัญหา B
  - หรือปัญหา B ไม่ง่ายกว่าปัญหา A
- นั่นคือทั้งสองปัญหายากง่าย<u>เท่ากัน</u>
  - หากปัญหา A เป็นปัญหาง่าย ปัญหา B ก็เป็นปัญหาง่ายด้วย
  - หากปัญหา A เป็นปัญหายาก ปัญหา B ก็เป็นปัญหายากด้วย

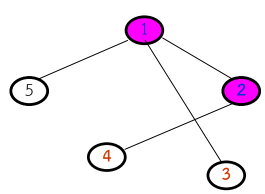
Reduction

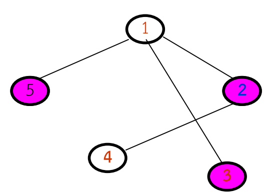
- Decision Problem
  - Vertex Cover : กำหนด G (V,E) จะมีเซตย่อย $V_{vc} \subseteq V$ ที่ cover ทุกเส้นเชื่อม โดยที่ $|V_{vc}|$ ไม่เกิน k หรือไม่
  - Independent Set : กำหนด G (V,E) จะมีเซตย่อย $V_I \subseteq V$ โดยจะไม่มีเส้นเชื่อมใดๆ ใน E ที่เชื่อมระหว่างโหนดใน V_I โดยที่ $|V_I|$ ไม่เกิน k หรือไม่

Vertex Cover

- Vertex Cover : กำหนด G (V,E) จะมีเซตย่อย $V_{vc} \subseteq V$ ที่ cover ทุกเส้นเชื่อม

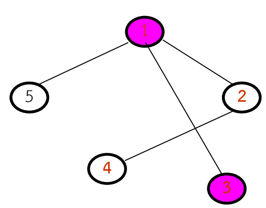






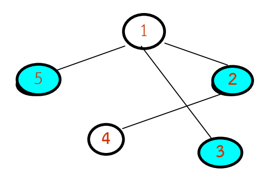
จะได้ว่า {1,4} {1,2} หรือ {2,3,5} ต่างก็เป็นเซตย่อย (V_{vc})ได้ ที่ cover ทุกเส้นเชื่อม

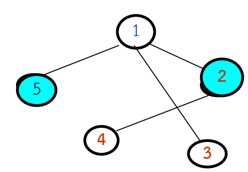
แต่ {1,3} ไม่ใช่เซตย่อย V_{vc} ของปัญหานี้

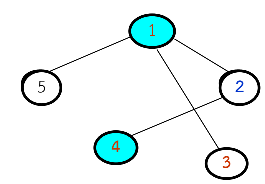


Independent Set

- Independent Set: กำหนด G (V,E) จะมีเซตย่อย $V_I \subseteq V$ โดยจะไม่มีเส้นเชื่อมใดๆ ใน E ที่เชื่อมระหว่างโหนดใน V_I

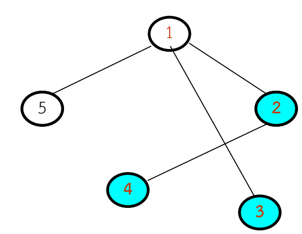




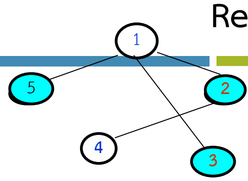


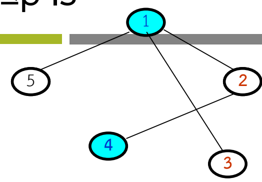
จะได้ว่า {2,3,5} {2,5} หรือ {1,4} ต่างก็เป็นเซตย่อย V_I ได้ ที่ ไม่มีเส้นเชื่อมระหว่างโหนดในเซตย่อย

แต่ {2,3,4} ไม่ใช่เซตย่อย V_I



Reduction : VC ≤p IS





V_I={2,3,5} และ $|V_I|$=3 V_{vc}={1,4}, $|V_{vc}|$=2

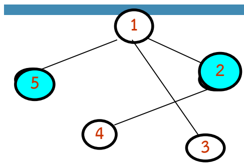
- จะเห็นว่าV_{vc}= V – V_I เนื่องจากไม่มีเส้นเชื่อมใดๆ ในกราฟที่เชื่อมระหว่างโหนดใน V_I เลย แสดงว่าทุกเส้นเชื่อม (E) ในกราฟ G จะอยู่ที่เซตของโหนดใน V – V_I ซึ่งแสดงว่าสามารถหาคำตอบของ Vertex Cover ได้จากเซตคำตอบของ Independent Set
- จะเห็นว่ากราฟ G(V,E) ที่มี | V_I |= k กราฟนี้ จะมี | V_{vc} |= |V| - k

Reduction : VC ≤p IS

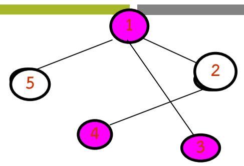
- ดังนั้นคำตอบอัลกอริทึมของปัญหา VC หาได้จากคำตอบอัลกอริทึมของปัญหา IS สรุปได้ว่า VC ≤p IS นั้นคือ IS ไม่ง่ายกว่า VC

```
VC (G=(V,E) ,  k) {
   //Reduce VC To IS by Changing parameters
     return IndepentSet(G,|V|-k)
}
```


Reduction : VC ≤p IS

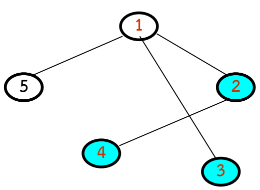


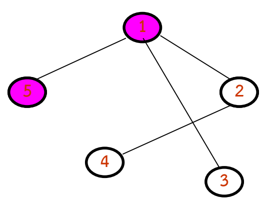
V_I={2,5}
และ $|V_I|$=2



V_{vc}={1, 3, 4}
และ $|V_{vc}|$=3

- พบว่า V_I={2,3,4} และ $|V_I|$=3 โดย V_I <u>ไม่ใช่คำตอบ</u>ของปัญหา IS
- V_{vc}=V-V_I={1,5} และ $|V_{vc}|$=2 โดย V_c <u>ไม่ได้เป็นคำตอบ</u>ของปัญหา VC เช่นกัน

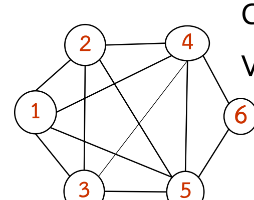




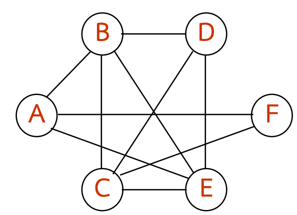
- พบว่า V_I={2,5} และ $|V_I|$=2 เป็นคำตอบของ IS จะได้ V_c =V-V_I ที่มีขนาด |V|-2 <u>เป็นคำตอบ</u>ของปัญหา VC ด้วย

Assignment#7 : Clique Problem

- กำหนด G=(V,E) เป็นกราฟแบบไม่ระบุทิศทาง และ k คือจำนวนเต็ม
- Clique : G มีคลิก (clique) ขนาดอย่างน้อย k หรือไม่
- คลิกคือกราฟย่อยที่เป็น complete graph (ทุกโหนดมีเส้นเชื่อมถึงกัน)
- ขนาดของคลิก (Vc) คือจำนวนโนดในกราฟย่อย



Clique ขนาด 5
Vc={1,2,3,4,5}



Clique ขนาด 4
Vc={B,C,D,E}

คำถาม:
ปัญหานี้เป็น NP หรือไม่ ?
สามารถ reduce ปัญหานี้ไปเป็นปัญหาใดได้ใน polynomial time