

FACULTY OF ENGINEERING  
CHULALONGKORN UNIVERSITY  
2110327 Algorithm Design

YEAR III, First Semester, Final Examination, September 30, 2010, Time 8:30 – 11:30

ชื่อ-นามสกุล \_\_\_\_\_ เลขประจำตัว 

										2	1
--	--	--	--	--	--	--	--	--	--	---	---

 CR58

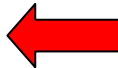
หมายเหตุ

1. ข้อสอบมีทั้งหมด 8 ข้อในกระดาษคำถามคำตอบจำนวน 2 แผ่น 3 หน้า คะแนนเต็ม 60 คะแนน
2. ไม่นอนุญาติให้นำตำราและเครื่องคำนวณต่างๆ ใดๆ เข้าห้องสอบ
3. ควรเขียนตอบด้วยลายมือที่อ่านง่ายและชัดเจน สามารถใช้ดินสอได้
4. ห้ามการหยิบยื่นสิ่งใดๆ ทั้งสิ้น จากผู้สอบอื่นๆ เว้นแต่ผู้คุมสอบจะหยิบยื่นให้
5. ห้ามนำส่วนใดส่วนหนึ่งของข้อสอบออกจากห้องสอบ ข้อสอบเป็นทรัพย์สินของราชการซึ่งผู้ลักพาอาจมีโทษทางคดีอาญา
6. ผู้ที่ประสงค์จะออกจากห้องสอบก่อนหมดเวลาสอบ แต่ต้องไม่น้อยกว่า 45 นาที
7. เมื่อหมดเวลาสอบ ผู้เข้าสอบต้องหยุดการเขียนใดๆ ทั้งสิ้น
8. ผู้ที่ปฏิบัติเข้าข่ายทุจริตในการสอบ ตามประกาศคณะวิศวกรรมศาสตร์ มีโทษ คือ ได้รับ สัญลักษณ์ F ในรายวิชาที่ทุจริต และพักการศึกษาอย่างน้อย 1 ภาคการศึกษา

รับทราบ

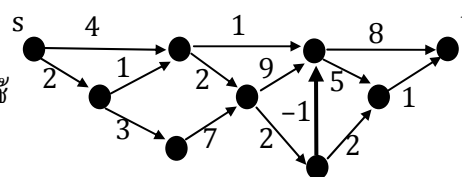
ลงชื่อนิติ (.....)

หมายเหตุ (เพิ่มเติม)

1. ข้อใดที่ให้ออกแบบอัลกอริทึมนั้น นิสิตสามารถตอบโดยเขียนบรรยายแนวคิดที่ implement ได้ในทางปฏิบัติ หรือจะเขียนเป็นรหัสเทียมประกอบแนวคิดที่นำเสนอด้วยก็ได้ และต้องวิเคราะห์ประสิทธิภาพเชิงเวลาของอัลกอริทึมที่นำเสนอด้วย นอกจากนี้ คะแนนที่ได้จะแปรตามประสิทธิภาพการทำงานของอัลกอริทึม
2. ต้องแสดงวิธีทำทุกข้อ การเขียนคำตอบเพียงอย่างเดียวจะไม่มีคะแนนให้ (ยกเว้นว่าจะเขียนในคำสั่ง)
3. ให้นิสิตเขียนรหัสประจำตัวและเลขที่ใน CR58 ในทุกหน้าของกระดาษคำถามด้วย
4. ให้เขียนตอบข้อที่ k ไว้ที่หน้าที่ k ในสมุดคำตอบ ( $k = 1, 2, \dots, 8$ ) เขียนไม่พอ ไปต่อที่หน้า 9, ... 

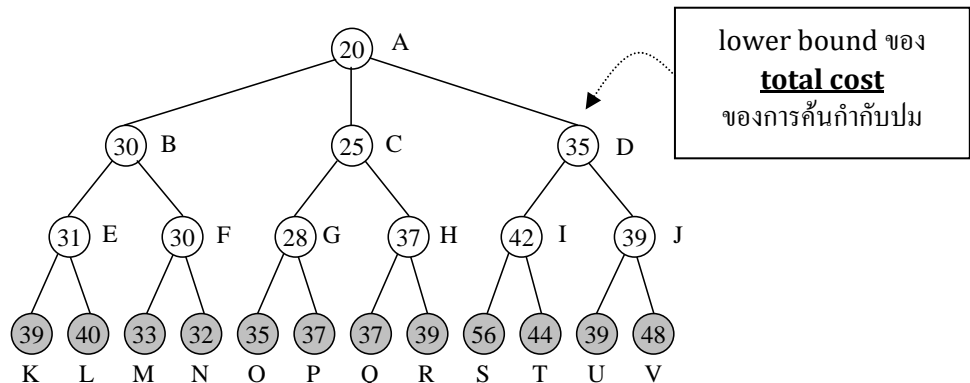
**ส่วนที่ 1: ข้อละ 5 คะแนน**

1. จงพิจารณาข้อความในแต่ละข้อต่อไปนี้ ว่าเป็นข้อความที่ถูกต้องหรือไม่ (ไม่ต้องอธิบาย) ถ้าคำตอบที่ถูกต้องได้ 0.5 คะแนน แต่ถ้าเป็นคำตอบที่ผิดเสีย 0.5 คะแนน คะแนนติดลบจะไม่ส่งผลกระทบต่อข้ออื่น
- ก. ให้  $V$  คือ จำนวนปมของกราฟ เราสามารถหาจำนวน connected components ของกราฟในเวลา  $\Theta(V)$
- ข. ให้  $G$  เป็น undirected connected graph การทำ depth-first search ในกราฟ  $G$  ไม่ว่าจะเริ่มที่ปมใด ก็ย่อมผ่านทุกปมใน  $G$  อย่างแน่นอน
- ค. แอปพบว่ากราฟ  $G$  ที่ได้รับมีเพียงปมเดียวเท่านั้นที่ไม่มีเส้นเชื่อมพุ่งเข้าหาเลย (in-degree เป็น 0) แอปมั่นใจว่ากราฟนี้มี topological sort เพียงรูปแบบเดียว
- ง. อัมได้กราฟ  $G$  ที่แสดงด้านล่างนี้มาหาวิถีสั้นที่สุดจากปม  $s$  ถึง  $t$  อัมเลือกใช้ Dijkstra's algorithm แต่ก็ไม่ได้หวัง อัมได้วิถีสั้นที่สุดจริงๆ
- จ. พลอยมีปัญหา ปัญหาของพลอยถามว่า “กราฟ  $G$  มีวิถีสั้นที่สุดจากปม  $s$  ถึง ปม  $t$  ที่มีระยะทางรวมไม่เกิน  $k$  หรือไม่” คิดสักครู่ พลอยก็สรุปว่า ปัญหาที่ง่ายมาก ปัญหาของพลอยจึงไม่จัดอยู่ในกลุ่มปัญหา NP



- จ. มินไม่มีปัญหา เพราะมินรู้ว่า ปัญหาที่ถามว่า “จากงาน events ต่าง ๆ ที่ติดต่อมาในเดือนธันวาคม (ซึ่งไม่สามารถรับงานที่ซ้อนเหลื่อมกันได้) จะมีวิธีเลือกรับงาน เพื่อให้มีรายได้ไม่ต่ำกว่า  $k$  หรือไม่?” เป็นปัญหา NP อย่างแน่นอน
- ข. ให้  $G$  เป็น directed graph ปัญหาที่ถามว่า “มีวิถีจากปม  $s$  ไปยังทุกปมใน  $G$  หรือไม่?” นั้นง่ายกว่า ปัญหาที่ถามว่า “มีวิถีจากทุกปมใน  $G$  มายังปม  $s$  หรือไม่?”
- ค. หากเรามีวิธี reduce (ในเวลาแบบ polynomial) ปัญหา  $S$  ไปเป็นปัญหา  $Q$  เราสามารถสรุปได้ว่า ปัญหา  $S$  ไม่ง่ายกว่า ปัญหา  $Q$
- ง. เมื่อต้นเดือนสิงหาคมที่ผ่านมา คุณวินัยแห่ง HP ได้เสนอผลงานการพิสูจน์ว่า  $P \neq NP$  ให้นักวิชาการตรวจสอบว่าถูกต้องหรือไม่ ถ้าผลการพิสูจน์ของวินัยถูกต้อง จะทำให้สารพัดปัญหาที่เคยคิดว่า น่าจะยาก กลับกลายเป็นปัญหาที่สามารถหาคำตอบได้อย่างง่ายดายทันที
- ฉ. ปัญหา NP-complete เป็นปัญหาที่ต้องใช้เวลาแบบฟังก์ชัน exponential ของปริมาณข้อมูลขาเข้า ในการหาคำตอบ

2. ต้นไม้ข้างล่างนี้คือ state space tree ของปัญหาหนึ่ง ซึ่งต้องการ minimize cost function บางอย่างของตัวปัญหา ปมสีเทาแทน solution state เราต้องการใช้ least cost search พร้อมด้วย branch and bound มาค้นคำตอบที่มี cost น้อยสุดในต้นไม้ ซึ่งได้มีการออกแบบ bounding function ไว้แล้ว โดยค่าของ lower bound ของ **total cost** ที่กำกับปมแสดงไว้ภายในปมของต้นไม้ (ตัวอักษรข้าง ๆ ปมคือชื่อปม) จงบอกลำดับของชื่อปมต่าง ๆ ที่ branch and bound พิจารณา โดยพิจารณาจากลำดับของปมที่นำออกจาก priority queue จนได้ปมคำตอบ (answer state) ที่มี cost น้อยที่สุด (ให้สังเกตว่าบางปมอาจจะไม่โดนพิจารณาโดย branch and bound ก็เป็นไปได้)



- 3. กำหนดให้มีนิพจน์บูลีนในรูปแบบ conjunctive normal form ดังต่อไปนี้  $(\bar{a} \vee b \vee c \vee d \vee \bar{e}) \wedge (\bar{f} \vee g \vee h)$  จงแปลงนิพจน์ดังกล่าวให้เป็นนิพจน์บูลีนแบบ conjunctive normal form ซึ่งแต่ละ clause มี literal ไม่เกิน 3 ตัว
- 4. กำหนดให้ `bool E[][][]` เป็นอาร์เรย์สองมิติที่ระบุถึง adjacency matrix ของ undirected graph ที่มี vertex จำนวน  $N$  vertex และกำหนดให้ vertex ของ graph นี้คือหมายเลข 0 ถึง  $N - 1$  จงเขียนฟังก์ชันภาษา C (พร้อมทั้งวิเคราะห์เวลาในการทำงาน) ที่รับ input เป็น `int sol[]` ซึ่งเป็นอาร์เรย์หนึ่งมิติจำนวน  $K$  ช่องที่เก็บ subset ของ vertex ของ graph นี้ เพื่อตรวจสอบว่า subset ดังกล่าวเป็น vertex cover ของ graph นี้หรือไม่ กำหนดให้ใช้โครงฟังก์ชันดังนี้

```
bool isVertexCover(bool E[][][], int N, int sol[], int K) {
    // E คืออาร์เรย์ขนาด N x N ช่องที่ประกอบด้วยเลข 0,1
    // ฟังก์ชันนี้ต้องคืนค่า true เมื่อ sol เป็น subset ที่เป็น vertex cover
}
```

**ส่วนที่ 2: ข้อละ 10 คะแนน**

5. ปัญหา Knapsack นั้นประกอบด้วยของจำนวน  $N$  ชิ้น โดยมีอาร์เรย์ float  $v[]$  และ float  $w[]$  เก็บมูลค่าและน้ำหนักของของแต่ละชิ้น และกำหนดให้น้ำหนักมากที่สุดของถุงคือ  $K$  เราต้องการทราบว่าการเลือกของเพื่อให้ได้มูลค่ารวมสูงสุดโดยที่น้ำหนักรวมของของที่เลือกไม่เกิน  $K$  นั้นมีมูลค่าเท่าไร โดยกำหนดให้มีเงื่อนไขเพิ่มเติมคือ ของแต่ละชิ้นมีหมายเลขกำกับอยู่ตั้งแต่ 0 ถึง  $N - 1$  และกำหนดให้คำตอบที่ได้นั้น ต้องไม่มีการเลือกของสองชิ้นที่มีหมายเลขติดกัน (เช่น ถ้าเลือกของหมายเลข 3 แล้ว จะต้องไม่เลือกของหมายเลข 4 และ 2 และกำหนดให้ชิ้นที่ 0 ไม่ติดกับชิ้นที่  $N - 1$ )
- ก. กำหนดให้มีของ 4 ชิ้น จงเขียน state space tree ของปัญหาดังกล่าวโดยคำนึงถึงเงื่อนไขเพิ่มเติมด้วย
  - ข. จงเขียนฟังก์ชันภาษา C เพื่อแก้ปัญหาดังกล่าวโดยใช้อัลกอริทึมแบบ DFS ซึ่งต้องใช้วิธีการ backtracking ใด ๆ ก็ได้ (ประสิทธิภาพของการ backtracking ไม่มีผลต่อคะแนน แต่ backtracking นั้นต้องใช้งานได้จริง คือสามารถตัดปมที่ไม่จำเป็นทิ้งไปได้) พร้อมทั้งวิเคราะห์ประสิทธิภาพในการทำงาน กำหนดให้ใช้โครงฟังก์ชันดังนี้

```
float knapsack(int N, float K, float v[], float w[]) {
    // ฟังก์ชันนี้ต้องคืนมูลค่าสูงสุดที่เป็นไปได้ จากการเลือกของที่ไม่ขัดกับเงื่อนไขในโจทย์
}
```

6. ปัญหาค่าผ่านทางเป็นดังต่อไปนี้ กำหนดให้มีเมืองอยู่  $N$  เมือง และมีถนนเชื่อมเมืองบางเมือง (ถนนไม่ได้เชื่อมทุกคู่เมือง ถนนสามารถวิ่งได้ทั้งไปและกลับ) กำหนดให้มี boolean matrix  $A$  โดยที่  $A_{ij}$  จะมีค่าเป็น 1 ก็ต่อเมื่อมีถนนจากเมือง  $i$  ไปยังเมือง  $j$  การเดินทางผ่านเมือง  $k$  นั้นจะต้องเสียค่าผ่านทางเป็น  $T_k$  สมมติให้ปัจจุบันเราอยู่ที่เมือง 0 และต้องการไปที่เมือง  $N - 1$  (เราต้องเสียค่าผ่านทางที่เมือง 0 และเมือง  $N - 1$  ด้วย) โดยต้องการที่จะเสียค่าผ่านทางน้อยที่สุด เราต้องการที่จะทราบค่าผ่านทางที่น้อยที่สุด จงออกแบบอัลกอริทึมสำหรับปัญหานี้ พร้อมทั้งวิเคราะห์ประสิทธิภาพในการทำงาน และคะแนนที่ได้จะแปรตามประสิทธิภาพของอัลกอริทึมที่ทำการออกแบบ
7. ปัญหาวิ่งผลัดประหลาดเป็นดังต่อไปนี้ การวิ่งผลัดเป็นการวิ่งจับเวลา โดยที่จะแข่งขันครั้งละ 1 ทีม สมมติให้ทีมทีมหนึ่งมีนักวิ่งทั้งหมด  $N$  คน ซึ่งจะต้องวิ่งในลู่วิ่งวงกลม นักวิ่งแต่ละคนจะต้องวิ่งรอบวงกลมนี้หนึ่งรอบทุกคน นักวิ่งแต่ละคนจะเริ่มต้นที่จุดเดียวกันทุกคน ก่อนที่การวิ่งจะเริ่มขึ้น นักวิ่งแต่ละคนจะได้รับแจกรายการที่ระบุถึง “นักวิ่งในทีมเดียวกันที่ต้องวิ่งครบรอบก่อน” ซึ่งนักวิ่งแต่ละคนจะเริ่มวิ่งได้ก็ต่อเมื่อนักวิ่งคนที่อยู่ในรายการ “นักวิ่งในทีมเดียวกันที่ต้องวิ่งครบรอบก่อน” นั้นได้วิ่งครบรอบแล้วเท่านั้น กำหนดให้  $B_i$  เป็นรายการ “นักวิ่งในทีมเดียวกันที่ต้องวิ่งครบรอบก่อน” ของนักวิ่งหมายเลข  $i$  และกำหนดให้นักวิ่งทุกคนวิ่งเร็วเท่ากัน โดยแต่ละคนใช้เวลาวิ่ง 1 นาทีในการวิ่ง 1 รอบลู่วิ่ง จงออกแบบอัลกอริทึมที่ใช้ในการคำนวณหาเวลาที่นักวิ่งคนสุดท้ายวิ่งครบรอบ พร้อมทั้งวิเคราะห์ประสิทธิภาพในการทำงาน และคะแนนที่ได้จะแปรตามประสิทธิภาพของอัลกอริทึมที่ทำการออกแบบ (หมายเหตุ: กำหนดให้  $B_0$  นั้นเป็นรายการว่าง หมายความว่านักวิ่งหมายเลข 0 นั้นจะเริ่มวิ่งทันที และให้สังเกตว่าเป็นไปได้ที่จะมีนักวิ่งมากกว่า 1 คนวิ่งพร้อมกันในสนามได้ และรับประกันว่ารายการ  $B$  ของนักวิ่งทุกคนนั้น จะทำให้นักวิ่งทุกคนได้วิ่งแน่นอน)
8. กำหนดให้  $G$  เป็น undirected connected weighted graph และ กำหนดขอบเขตเป็นตัวเลขจำนวนเต็มบวก  $D$  ให้ จงออกแบบอัลกอริทึมเพื่อหาต้นไม้ทอดข้ามที่มีน้ำหนักรวมต่ำสุด (minimum spanning tree) โดยมีข้อกำหนดเพิ่มเติมคือ MST ที่ได้นั้นจะต้องไม่มีเส้นทาง (path) ระหว่างสอง vertex ใด ๆ ที่มีจำนวนเส้นเชื่อมใน path นั้นเกิน  $D$  เส้น โดยให้วิเคราะห์ประสิทธิภาพในการทำงานด้วย และคะแนนที่ได้จะแปรตามประสิทธิภาพของอัลกอริทึมที่ทำการออกแบบ