דף סיכום בחינה

מזהה קורס: 20585 שם קורס: מבוא לתורת החישוביות והסיבוכיות

מספר שאלה	ציון מירבי	ציון שאלה סופי
1	20.00	20.00
2	20.00	20.00
3	20.00	20.00
4	20.00	19.00
5	20.00	
6	20.00	20.00

ציון בחינה סופי : 99.00

הבחינה הבדוקה בעמודים הבאים

מבחן בחישוביות וסיבוכיות

השתדלתי להקליד את הרוב, עניתי על שאלות 1,2,3,4,6

שאלה 1:

כן, אם L מזוהה טיורינג אז בהכרח קיים לה מונה לפי אין סוף.

המונה יעשה 3 דברים:

1. המונה יחשב וישמור רשימה של כל המילים המזוהות
2. המונה ידפיס אין סוף פעמים את המילים המזוהות
3. המונה ידפיס פעם אחת את כל המילים בסדר לקסיקוגרפי.

כמובן שאי אפשר להגיד פשוט שהמונה יתחיל עם 3, יעבור ל1 ואז יעבור ל2, כי הוא אף פעם לא יסיים.

נתחיל עם קצת אינטואיציה – אם היינו רוצים רק להדפיס את כל המילים בE^* כמות אינסופית של פעמים, היינו פועלים באופן הבא, עבור על המילים באופן לקסיקוגרפי והדפס את המילה הראשונה, אחר כך את המילה הראשונה והשנייה, ואז את המילה הראשונה השנייה והשלישית, וכו'.

אם L מזוהה טיורינג אז קיימת מ"ט M שמזהה אותה.

המונה יפעל באופן הבא:

התחל את *count* ל1

בצע אין סוף פעמים:

1. הרץ על ה *count* מילים הראשונות *count* צעדים על M
   אם M זיהתה מילה, בדוק אם המילה כבר קיימת במאגר המילים המזוהות, אם לא הוסף את המילה למאגר
2. הדפס את כל מאגר המילים המזוהות
3. הדפס את המילה ה*count* בE^*
4. הגדל את *count* ב1

נכונות:

שלב 1 תמיד ירוץ זמן סופי של זמן – *count* בריבוע צעדים על M, ובדיקה של כמות סופית של מילים במאגר סופי. גם 2 ירוץ כל פעם בזמן סופי, כי מאגר המילים תמיד סופי. גם 3 ו4 כמובן.

כל מילה ששייכת לL תצטרף למאגר המילים המזוהות בשלב ש*count* שווה למיקום של המילה בסדר הלקסיקוגרפי או לכמות הצעדים שלוקח לM לזהות את המילה, הגבוה מבניהם.
לאחר מכן, המילה תודפס אינסוף פעמים, כי שלב 2 ירוץ אינסוף פעמים לאחר שהמילה התווספה למאגר המילים המזוהות.

כל מילה שלא שייכת לL תודפס פעם אחת בדיוק בשלב 3

מש"ל

20
(1)

שאלה 2:

נראה רדוקצית מיפוי מהמשלימה של *ATM* ל *MINIMAL-WORD*:

עבור הקלט הבא *<M,W>*: (*M*- מ"ט, *W* – מילה)

בנה מכונה *F* שתפעל באופן הבא:

המכונה *F* תקבל את המילה *Wa*, כאשר *a* תו קלט כל שהוא (נבחר תו שרירותי, כמובן שונה מרווח)

F תידחה כל מילה *V* ששונה מ*W* וקצרה או שווה ל*W*

עבור הקלט *W*, *F* תריץ את *M* עם קלט *W*, עם *M* מקבלת את *W*, אז *F* גם מקבלת את *W*, אחרת *F* דוחה את *W*

F תקבל כל מילה אחרת

החזר*<F,Wa>*

<u>הוכחת נכונות בשני כיוונים:</u>

אם *<M,W>* שייך למשלימה של *Atm* אז *M* תדחה את *W*, ולכן *F* תדחה כל מילה שקצרה מ*WA* ולכן

<F,Wa> שייכת ל *MINIMAL-WORD*

בכיוון השני:

אם *<F,Wa>* שייך ל*WORD-MINIMAL* אז *F* קיבלה את *WA* ו*F* לא קיבלה אף מילה שקצרה מ*WA*, בפרט, *F* לא קיבלה את *W*, וזה קורה אם"ם *M* לא קיבלה את *W* ולכן *<M,W>* שייך למשלימה של *ATM*

משל

20

(2)

שאלה 3.

UHAMCIRCUIT שייך ל NP. כי יש לו מאמת
בזמן פולינומיאלי (7.19). המאמת הוא רשימת הקודקודים
וצמתים המהווים מעגל המילטוני.

מהנתון ש $P=NP$ מתקבל כי יש מ"ט M המכריעה
את UHC (UHAMCIRCUIT) בזמן פולינומיאלי.
נציג אלגוריתם שמשתמש ב M כדי להחזיר מעגל
המילטוני בגרף G:

I בדוק האם קיים מעגל המילטוני ב G (בעזרת M)
- אם לא, החזר לא.

II עבור על כל הקשתות בגרף G בסדר לקסיקוגרפי:
עבור כל קשת: בדוק האם לגרף G (ללא הקשת)
יש מעגל המילטוני. אם כן הוצא את הקשת
מהגרף, והמשך. אם לא, החזר את הקשת והמשך.

III כל הקשתות שנשארו הן קשתות המעגל
ההמילטוני. נשאר רק לסדר ולהחזיר את הצמתים
- נבחר צומת באופן שרירותי, ונבצע DFS
החזר את רשימת הצמתים לפי הגילוי.

שאלה 3 המשך:

הוכחת נכונות:

1. עם בG לא קיים מעגל – האלגוריתם יחזיר לא.
2. עם ב G קיים לפחות מעגל המילטוני אחד, אז אם נסדר את המעגלים לפי סדר לקסיקוגרפי, האלגוריתם יחזיר את האחד, שהקשת הראשונה שלה (מבחינה לקסיקוגרפית) היא האחרונה לקסיקוגרפית מבין כל המעגלים.
(לדוגמא, אם יש מעגל A ומעגל B, נסדר את A ונסדר את B ונשווה בין הקשתות הראשונות של מעגל A ומעגל B, זאת עם הקשת ה"מאוחרת" יותר תהיה המעגל שיוחזר)
הסבר: כל הקשתות הקודמות לקשת הראשונה של המעגל המוחזר יוסרו, כי ניתן להסירם ולשמור על מעגל המילטוני בגרף, לאחר מכן, יוותר מעגל אחד בגרף, וכל שאר הקשתות יוסרו.

3 כאשר נשארו רק הקשתות שבמעגל, ריצת DFS תרוץ על המעגל ותחזיר את הצמתים בסדר הנכון.

זמן ריצה: אנחנו משתמשים במ"ט על גרף G (שלם או חלקי) פעם אחת בהתחלה ופעם אחת לכל קשת בG

זמן הריצה כפי שהראנו של M הוא פולינומיאלי, וכמובן שמספר הקשתות פולינומיאלי בקלט, לכן סך זמן הריצה הוא פולינומיאלי.

20
(3)

שאלה 4

כדי להראות ש*COVC* (*COVER-VERTEX-OR-CLIQUE*) היא *NP* שלמה נראה שהיא ב*NP* וקיים רדוקציה מ *CLIQUE* אליה.

COVC היא ב*NP* כי קיים לה מאמת: המאמת יהיה קבוצת צמתים שמהווים קליקה או כיסוי
(המאמת ירוץ בזמן פולינומיאלי, קודם יבדוק אם הקודקודים מהווים קליקה – זמן פולינומיאלי, ואם לא, יבדוק אם הם מהווים כיסוי – זמן פולינומיאלי)

כדי להראות ש *COVC* שלמה נראה רדוקציה מ*CLIQUE* ל*COVC*. כלומר, אם נוכל לפתור את *COVC* בזמן פולינומיאלי, נוכל לפתור גם את *CLIQUE* בזמן פולינומיאלי.

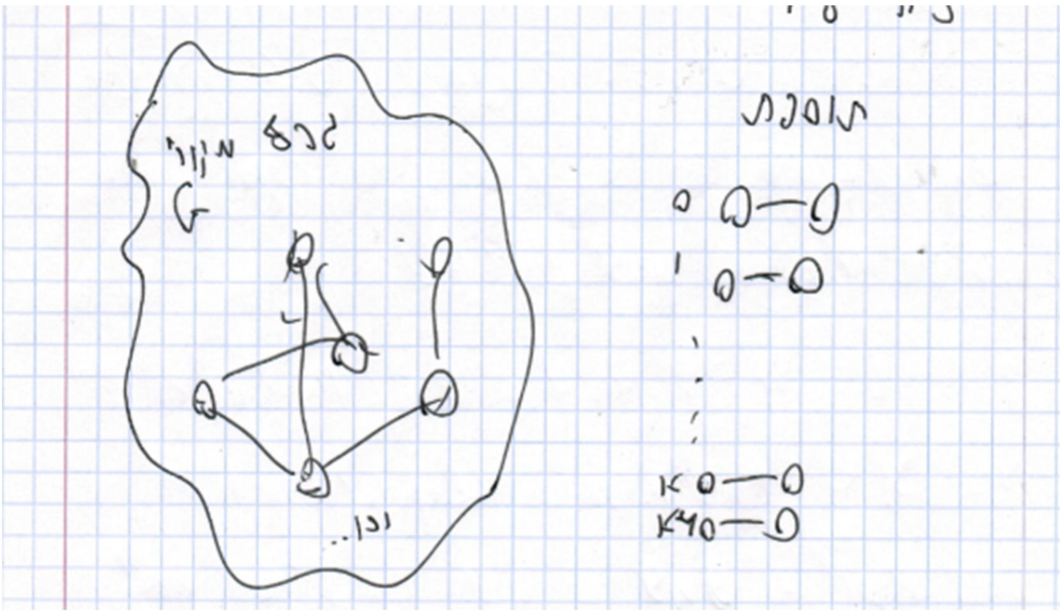
הרדוקציה:

(אני מניח ש *K* גדול מ2, אחרת אפשר לבדוק את *CLIQUE* בזמן פולינומיאלי ולהחזיר גרף בהתאם)

עבור גרף *G* ומספר *K* הוסף לגרף *K* + 1 זוגות צמתים, כשלכל זוג קיים קשת שמחברת ביניהם.

החזר את *G*' ואת *K*

ציור ל4 שמתאר את התוספת לגרף:



הרדוקציה תקינה – נראה שני כיוונים:

כיוון ראשון, אם קיים קליקה בגודל *K* אז גם בגרף החדש יהיה קליקה בגודל *K* ו*COVC* תחזיר תשובה חיובית

כיוון שני – אם *COVC* מקבל, אז או שקיים בגרף החדש קליקה בגודל *K* או שקיים כיסוי בגודל *K*.

אבל לא יכול להיות שקיים כיסוי בגודל *K*, כי קיימים *K* + 1 קשתות שזרות זו לזו מבחינת צמתים, ולכן קיים בגרף החדש קליקה בגודל *K*.

19
(4)

לא חישבת את זמן הריצה של הרדוקציה

K הצמתים שמהווים קליקה בהכרח שייכים גם לגרף המקורי, כי הוספנו רק צמתים שמחוברים לצומת אחד בלבד, ולכן אם <G',K> שייך ל *COVC* אז בהכרח <G,K> שייך ל*CLIQUE*. משל

שאלה 6

לא, אי אפשר להסיק מהנתונים שלבעיית *MAXIS* יש אלגוריתם קירוב פולינומיאלי בעל יחס קירוב קבוע.

נראה זאת תחילה בצורה אלגברית, ואז בצורה יותר אינטואיטיבית.

נסמן MVC_{opt} קבוצת הקודקודים שמהווה כיסוי מינימלי

ו- MVC לקבוצת הקודקודים שמתקבלת מאלגוריתם הקירוב.

בצורה דומה MI_{opt} - פתרון אופטימלי ל $MAX\text{-}IS$

MI פתרון שמתקבל אם [illegible] משתמשים באלגוריתם הקירוב של MVC

מהנתונים $|MVC| \le 2\cdot|MVC_{opt}|$ ⊛

מהנתון ⊖ W מהווה כיסוי מינימלי אזי $V-W$ מהווה פתרון ל $MAXIS$ מקסימלי

~~[illegible]~~ $|MVC_{opt}| + |IS_{opt}| = |V|$

$|MVC_{opt}| = |V| - |IS_{opt}|$ ולכן $\Leftarrow$ $|MVC| + |IS| = |V|$

$|MVC| = |V| - |IS|$

נציב ב ⊛ ונקבל

$$|V| - |IS| \le 2\cdot(|V| - |IS_{opt}|)$$

$$|V| - |IS| \le 2\cdot|V| - 2\cdot|IS_{opt}|$$

$$-|IS| \le |V| - 2|IS_{opt}| \quad /\cdot -1$$

$$|IS| \ge 2|IS_{opt}| - V$$

לכן, אי אפשר להגיע לקבוע קירוב

נסביר בצורה יותר אינטואיטיבית:

הפתרון של *MVC* קטן מלפחות פי 2 מהפתרון האופטימאלי, אך הפתרון האופטימאלי יכול גם להכיל את חצי מהקודקודים, (לדוגמא גרף שמורכב רק מזוגות של קודקודים שמחוברים אחד לשני) ולכן, במקרה כזה, הפתרון המקורב ל*MVC* יכול להכיל את כל הקודקודים (כי רק מובטח לנו שהוא לא יותר מפי שניים מהאופטימאלי) – ואז לא "יישארו" לנו קודקודים ל*MAXIS*.

צורה שונה להגיד את זה:

אנחנו יכולים להגדיל את אחוז הקודקודים ששייכים לפתרון מקורב כרצוננו, ובהתאמה, להקטין את אחוז הקודקודים באלגוריתם מקורב ב*MAXIS* (כי אלגוריתם שמתקבל מהנתונים ישייך את כל הקודקודים שלא ב*MVC* ל*MAXIS*)

לכן, אי אפשר לבנות או להוכיח שקיים אלגוריתם קירוב ל*MAXIS* רק על סמך הנתונים.

20
(6)