**ממ"ן 11**

**שאלה 1**

1. נניח בשלילה שהטענה לא נכונה כלומר בגרף פשוט לא מכוון עם יותר מקודקוד אחד לא קיימים שני קודקודים בעלי אותה דרגה. לכן בהכרח לכל קודקוד דרגה שונה.

נניח כי בגרף n קודקודים ולכן קיימות n דרגות אפשריות שהן: 0, 1, 2, … , n-1.

אם לכל הקודקודים דרגות שונות אז קיים קודקוד אחד בכל דרגה.

הקודקוד בדרגה 0 אינו מחובר לאף קודקוד ואילו הקודקוד בדרגה n-1 מחובר לכולם. קיבלנו סתירה ומכאן שהנחתנו שגויה כלומר הטענה נכונה – בכל גרף לא מכוון עם יותר מקודקוד אחד קיימים שני קודקודים בעלי אותה דרגה.

1. כיוון ראשון – נניח כיG גרף לא מכוון קשיר.

יהי S קבוצת קודקודים לא ריקה חלקית ל-V.

G קשיר, לכן מכל קודקוד בהכרח קיים מסלול לכל קודקוד .

יהי w הקודקוד האחרון ב-S בו מבקר המסלול. לאחר הביקור ב-w המסלול עובר לקבוצת הקודקודים שאינה ב-S . לכן בהכרח קיימת קשת שקצה אחד שלה ב-w, כלומר בקבוצהS וקצה שני שלה אינו ב-S, ולכן הטענה מתקיימת.

כיוון שני– נניח כי בכל קבוצת קודקודים S ישנה קשת שקצה אחד שלה ב-S והשני איננו ב-S.

נניח בשלילה ש-G אינו קשיר, לכן קיימים לפחות שני רכיבי קשירות. נגדיר את S כקבוצת קודקודי אחד מרכיבי הקשירות ואת המשלים  כרכיב הקשירות השני. ע"פ הנתון קיימת קשת שקצה אחד שלה ב-S והשני אינו ב-S כלומר קיימת קשת בין שני רכיבי הקשירות. קיבלנו סתירה, ולכן הטענה לא נכונה כלומר G בהכרח גרף קשיר.

**שאלה 2**

תיאור האלגוריתם: נפתור את הבעיה ע"י רדוקציה לחיפוש BFS. מהגרף הנתון G ניצור גרף חדש G' בו כל קודקודי הקבוצה S יוחלפו בקודקוד בודד s. קשתות הקודקוד s יהיו כל הקשתות היוצאות מקודקודי הקבוצה S לקודקודים מחוץ לקבוצה. לאחר מכן נבצע חיפושBFS על G' ונקבל את רשימת השכבות עבור כל מרחק מהקודקוד s. רשימת שכבות זו זהה לרשימת השכבות עבור כל מרחק מהקבוצה S.

האלגוריתם:

1 create new graph G' from G: s node replace all S nodes

2 Ls = BFS(G', s)

3 foreach n node in S

3.1 add n to Ls(0)

4 endfor

5 return Ls

נכונות:

טענה: מרחק כל קודקוד  הנמצא בשכבה Ls(i) מ-S שווה ל- i.

נכונות הטענה נובעת ישירות מנכונות חיפוש BFS שכן האלגוריתם בונה גרף חדש מהגרף הנתון כך שבמקום כל קודקודי S מופיע קודקוד בודד s. כלומר מרחק שאר הקודקודים מהקבוצה S נשמר גם לאחר השינוי. על הגרף G' ביצענו חיפוש BFS שתוצאתו היא רשימת שכבות הקודקודים לפי מרחקם מקודקוד s, ומכאן שרשימה זאת שווה לרשימת השכבות לפי מרחקם מקבוצה S. לכן ע"פ נכונות אלגוריתם BFS מרחק כל קודקוד  הנמצא בשכבה Ls(i) מקבוצהS הוא בהכרח i.

סיבוכיות: סיבוכיות בניית הגרף החדש G' מהגרף הנתוןG מצריכה מעבר על כל הקודקודים והקשתות של הגרףG ולכן פעולה זאת תתבצע בזמן O(m+n).

לאחר מכן האלגוריתם מבצע חיפוש BFS רגיל שתוצאתו היא רשימת השכבות לפי מרחקים, סיבוכיותו גם היא O(m+n).

לבסוף האלגוריתם מאתחל את רשימת השכבה Ls(0) לקודקודי S שהם לכל היותר n ולכן סיבוכיות הפעולה היא O(n).

מכאן שסיבוכיות האלגוריתם כולו היא O(m+n).

**שאלה 3**

תיאור האלגוריתם: גרף G ניתן לכיוון כך שבגרף המכוון לכל קודקוד דרגת כניסה גדולה מאפס אם ורק אם כל רכיבי הקשירות בגרף מכילים מעגל. האלגוריתם ימצא תחילה את רכיבי הקשירות בגרף ועבור כל רכיב יבדוק האם ניתן לכוונו (ע"י בדיקה אם קיים מעגל). במידה וניתן, נבחר זוג קודקודים סמוכים במעגל נכוון את הקשת בכיוון כלשהו ונבצע חיפוש BFS החל מהקודקוד אליו נכנסת הקשת. כיווני קשתות העץ המתקבל יתאימו לכיוון הנדרש לגרף. במקרה זה נבצע רדוקציה לחיפוש BFS כך שיחזיר את רשימת הקשתות המווכנות.

האלגוריתם:

1 initialize an empty list L

2 find all connected components of G

3 foreach connected component C in G

3.1 if  then

3.1.1 

3.1.2 

3.1.3 add edge  to L

3.2 else

3.2.1 return "graph G cannot be directed"

4 endfor

5 direct randomly all indirect edges

6 return L

נכונות:

טענה: ניתן לכוון קשתות גרף קשיר לא מכוון G כך שדרגת כניסה לכל קודקוד גדולה מאפס אם ורק אם יש בו מעגל.

הוכחה:

כיוון ראשון – נניח כי ניתן לכוון קשתות גרף קשיר לא מכוון G באופן הנדרש.

נניח בשלילה שהטענה לא נכונה, כלומר ניתן לכוון את G אך אין בו מעגל. יהי G' הגרף המכוון ללא מעגלים בסוף הריצה. ע"פ טענה 3.19 ע'מ 110 הגרף G' מכיל קודקוד v שלא נכנסת אליו אף קשת כלומר דרגתו היא אפס. קיבלנו סתירה, לכן הנחתנו שגויה כלומר G בהכרח מכיל מעגל.

כיוון שני – נניח כי הגרף G מכיל מעגל.

יהיה u,v זוג קודקודים סמוכים במעגל. בה"כ נכוון את הקשת בכיוון . כעת נבצע חיפוש BFS החל מקודקוד u. העץ T המתקבל בנוי כאשר קודקוד u הוא השורש וכל יתר קודקודי הגרף צאצאים שלו. לכל צאצא דרגת הכניסה גדולה מאפס שכן הקשת מאביו נכנסת אליו. כמו כן לקודקוד השורש u נכנסת קשת מקודקוד v ולכן גם הוא מקיים את הטענה. כל הקודקודים עונים על התנאי ולכן אם קיימים קשתות לא מכוונות נוספות ניתן לכוון אותן באופן שרירותי ולקבל גרף G' העונה על התנאי המבוקש.

נכונות האלגוריתם נובעת מנכונות טענה 3.2 לפיה בגרף יש מעגל אם מספר הקשתות גדול או שווה למספר הקודקודים בו, וכן מנכונות הטענה שהוכחנו. הטענה שהוכחנו מסתמכת על נכונות אלגוריתם BFS הבונה עץ מכוון המכיל את כל הקודקודים באותו רכיב קשירות. מנכונות כל הטענות תוצאת האלגוריתם בהכרח תהיה גרף מכוון שבו דרגת הכניסה לכל קודקוד גדולה מאפס.

סיבוכיות: סיבוכיות אלגוריתם מציאת רכיבי הקשירות בגרף היא O(m+n) (סעיף 3.3 עמ' 102(. גרף G מכיל לכל היותר n רכיבי קשירות ולכן לולאת ה-for תתבצע לכל היותר n פעמים. בלולאת ה-for מתבצע חיפוש זוג קודקודים במעגל וחיפוש BFS. סיבוכיות כל אחד מהם היא O(m+n). אולם בכל איטרציה מתבצעת כמות קבועה של עבודה על קשת או קודקוד נתון של רכיב קשירות ולכן סיבוכיות האלגוריתם עדיין O(m+n). מכאן שסיבוכיות האלגוריתם כולו היא O(m+n).

**שאלה 4**

1. תיאור האלגוריתם: גרף G מכיל מעגל אם באחד מרכיבי הקשירות שלו מספר הקשתות גדול או שווה למספר הקודקודים שבו (טענה 3.2). לכן נמצא תחילה את רכיבי הקשירות של הגרף G ולאחר מכן נבדוק האם קיים מעגל באחד מהם. במידה וכן נבצע חיפוש BFS המחזיר את הקשת הראשונה המעבירה קודקוד לקודקוד בו ביקר. לאחר מכן נמצא את האב הקדמון המשותף של הקודקודים אותם מחברת הקשת. מכל קודקוד נוסיף את הקשתות המחברות את הקודקוד לאב המשותף. לבסוף נקבל כי המסלול יתחיל ויסתיים באותו קודקוד (האב המשותף) כלומר נקבל מעגל.

האלגוריתם:

1 initialize an empty list L

2 find all connected components of G

3 foreach connected component C in G

3.1 if  then

3.1.1 set s a random node of C

\\ e is the first edge that does not appear in T 

3.1.2 

3.1.3 

3.1.4 add all nodes from w to v to L

3.1.5 add all nodes from u to w to L

3.2 endif

4 endfor

5 return "circle was not found"

נכונות:

טענה: אם בחיפוש BFS מתגלה קשת שלא מופיעה בעץ אז קיים מעגל בגרף.

הוכחה: יהיו u,v שני קודקודים בגרף המחוברים בקשת שאינה מופיעה בעץ BFS. לשני הקודקודים בהכרח קיים אב קדמון משותף כלשהו שכן שניהם צאצאים של השורש. נבנה מסלול באופן הבא: נמצא את האב הקדמון המשותף הנמוך ביותר, נסמן w. נוסיף את רשימת הקודקודים המובילה מ-w לצאצא v. נוסיף את רשימת הקודקודים מצאצא u לאב w. מסלול זה בהכרח מעגל שכן התחלנו מכך שקיימת קשת המחברת את v ו-u ומצאנו שקודקוד ההתחלה והסיום של המסלול הוא w.

נכונות האלגוריתם נובעת מנכונות אלגוריתם מציאת רכיבי הקשירות בגרף (סעיף 3.3), נכונות טענה 3.2 ונכונות אלגוריתם BFS. לאחר ביצוע שלוש פעולות אלה במידה ונמצא רכיב קשירות המכיל מעגל, נמצא את הקשת המחברת בין שני קודקודים ונבנה מעגל כמו שתואר בטענה.

סיבוכיות: סיבוכיות אלגוריתם מציאת רכיבי הקשירות בגרף היא O(m+n) (סעיף 3.3 עמ' 102). לכל היותר יש n רכיבי קשירות ולכן לולאת ה-for תתבצע לכל היותר n פעמים בזמן קבוע ולכן הסיבוכיות היא O(n).

סיבוכיות ביצוע חיפושBFS למציאת הקשת שלא מופיעה בגרף ובניית העץ היא O(m+n). יצירת רשימת קודקודי המעגל עשויה לקחת כמספר הקודקודים בגרף כלומר O(n) לכל היותר. מכאן שסיבוכיות האלגוריתם כולו היא O(m+n).

1. הטענה לא נכונה. דוגמא נגדית:

**V1**

**V2**

**V4**

**V3**

יהי הגרף G הבא:

**V1**

**V4**

**V2**

**V3**

ויהי העץ הפורש T הבא:

כל עץ DFS של G הוא בהכרח מהצורה:

שכן כל קודקוד מחובר לכל האחרים ולכן ניתן לעבור על

כולם מבלי לחזור אפילו פעם אחת לאחור.

בצורה זו אין קודקוד שדרגתו 3 ולכן לא ניתן לקבל את

עץ DFS שגרף התשתית שלו הוא העץ הפורש T המוצג.

1. הטענה לא נכונה. דוגמא נגדית:

יהי הגרף G הבא:

**V1**

**V4**

**V2**

**V3**

**V1**

**V4**

**V2**

**V3**

עץ DFS ראשון:

**V1**

**V3**

**V2**

**V4**

עץ DFS שני:

שני העצים נבנו מגרף G הנתון.

העץ הראשון מכיל עלה אחד והעץ השני

מכיל שני עלים.

**שאלה 5**

תיאור האלגוריתם: הגרף הנתון G הוא עץ, שכן הוא גרף קשיר ללא מעגלים. לכן לכל קודקוד יש מסלול יחיד המגיע אליו. קוטר עץ הוא המרחק המקסימלי בין שני עלים. על מנת למצוא אותו נבצע שתי הרצות BFS. האחת מקודקוד אקראי s והשנייה מקודקוד u שהוא הקודקוד הרחוק ביותר מקודקוד s. קוטר הגרף הוא המרחק בין הקודקודu לקודקוד הרחוק ביותר ממנו v.

האלגוריתם:

FIND\_DIAMETER(G)

1 select random node s in G

2 

3 set u the deeper node of T1 tree

4 

5  calc to distance between u and the deeper node of T2 tree

6 return diam

נכונות:

טענה 1: קוטר העץ מחבר בין שני עלים.

הוכחה: נניח בשלילה שהטענה אינה נכונה. יהי  קוטר העץ. בה"כ נניח כי u1 אינו עלה. לכן קיים ל-u1 צאצא v ומתקיים שכן הוספנו קשתות לאחר הביקור ב-u1 ולכן המסלול שהתקבל בהכרח ארוך יותר. סתירה.

טענה 2: אחד מקצוות הקוטר הוא הקודקוד המרוחק ביותר מהשורש.

הוכחה: נניח בשלילה שהטענה אינה נכונה כלומר הקוטר הוא מסלול בין  כאשר שניהם אינם הקודקודים המרוחקים ביותר מהשורש.

יהי v הקודקוד המרוחק ביותר מהשורש. נפריד למקרים:

1. הקוטר עובר דרך השורש –

**r**

**u2**

**z**

**u1**

**v**

מתקיים: 





קיבלנו סתירה שכן הנתחתנו היא כי הוא הקוטר.

1. הקוטר לא עובר דרך השורש –

**r**

**u2**

**z**

**u1**

**v**

מתקיים:









כלומר  שוב סתירה.

נכונות האלגוריתם נובעת מטענות 1 ו-2 לפיהם הקוטר הוא המסלול המחבר בין שני עלים כאשר אחד מהם הוא העלה הרחוק ביותר מהשורש. וכן מנכונות אלגוריתם BFS המוצא את המרחק הקצר ביותר בין שני קודקודים. לכן כשנמצא העלה המרוחק ביותר משורש כלשהו, הקוטר יהיה המרחק לקודקוד שנמצא אחרון בחיפושBFS כלומר הקודקוד הרחוק ביותר מאותו עלה.

סיבוכיות: סיבוכיות בחירת קודקוד אקראי היא O(1). סיבוכיות כל אחת מהרצות חיפוש BFS ומציאת הקודקוד הרחוק ביותר היא O(m+n). מכאן שסיבוכיות האלגוריתם כולו היא O(m+n) .