**אלגוריתמים**

**ממ"ן 11 - תשובות**

מספר קורס: 20417

מנחה: אורן רות

שם הסטודנטית: ברית בן-דוד

ת"ז: 204879365

**שאלה 1:**

בעיית השידוך היציב. פתרו את שאלה 6.1 בספר הקורס.

**פתרון:**

**נתון:**

* n ספינות - S
* n נמלים - P
* m>n ימים בחודש
* כל ספינה נמצאת בכל נמל בדיוק יום אחד במהלך החודש, לפי לוח זמנים קיים.
* אסור לשתי ספינות להימצא באותו נמל באותו יום.

**הבעיה:** בניית לוח זמנים מקוצץ (מלוח הזמנים הקיים) עבור כל אחת מהספינות.

על-מנת לענות על הבעיה נתבסס על האלגוריתם של גייל ושייפלי לבעיית הזיווג היציב.

נקביל את האלגוריתם שלנו לאלגוריתם החיזור בתצורה כזאת שהנמלים יקבלו את תפקיד הגברים ואילו הספינות יקבלו את תפקיד הנשים.

1. **תיאור האלגוריתם:**

**בתחילת האלגוריתם** כל הנמלים נמצאים במקומם ואילו כל הספינות אינן עוגנות בשום נמל (לשם ההפשטה – נמצאות בנמל אחר ממנו הן יוצאות / נמצאות בים).

**במהלך ריצת** האלגוריתם לקיצוץ לוחות הזמנים, יתקיימו התנאים הבאים:

* כל **נמל** יעדיף שהספינה הראשונה שביקרה אצלו תישאר לעגון בו עד לסוף החודש. כמובן שיש לשים לב ששום ספינה לא עוגנת בו קודם לכן.
* כל **ספינה** תעדיף להישאר לעגון בנמל האחרון בו ביקרה. גם כאן יש לשים לב ששום ספינה אחרת לא עוגנת בנמל זה לפני כן.

ריצת האלגוריתם **תסתיים** כאשר כל הנמלים מאוישים (בכל נמל יש ספינה אחת שעוגנת בו).

**הגדרות לשימוש האלגוריתם:**

* "פנוי/ה" – נמל או ספינה שאין לה שידוך (נמל שלא עוגנת בו אף ספינה או ספינה שלא עוגנת באף נמל.
* "שידוך סופי" – ציוות בין נמל עגינה לבין הספינה שתישאר לעגון בו עד לסוף החודש.
* "שידוך זמני" – ציוות בין נמל עגינה לבין ספינה, עד שמתקבלת הסכמה כי זהו השידוך הסופי.
* "תפוסה" – כמו שידוך זמני, ספינה שעברה בנמל אחד לפחות.

**האלגוריתם:**

1. כל הנמלים והספינות "פנויים".
2. **כל עוד** קיים נמל (p) פנוי יתקיימו השלבים הבאים:
   1. נבחר נמל p מסוים.
   2. נשדך בין נמל p לבין הספינה אותה הוא מעדיף (הספינה הראשונה שביקרה אצלו).
   3. אם הספינה אינה "תפוסה" (לא עוגנת בשום נמל אחר):
      1. אז – נשדך זמנית בין הנמל לבין הספינה (p,s).
      2. אחרת (משודכת זמנית ל-p') – נבדוק מה סדר העדיפויות של אותה הספינה (לפי הנמל האחרון בו ביקרה בלו"ז הקיים):
         1. אם הספינה מעדיפה את נמל p', אז – נמל p יישאר פנוי.
         2. אחרת, אם הספינה מעדיפה את נמל p – נמל p' יהפוך לפנוי.
3. כשנסיים את לולאת הwhile כל הנמלים כבר לא יהיו פנויים, וכן גם הספינות ונקבע את השידוכים הזמניים כשידוכים סופיים.

**נראה דוגמה קטנה להמחשת ריצת האלגוריתם:**

יהיו 4 נמלים ויהיו 4 ספינות , נביט בלו"ז החודשי של הנמלים והספינות, שעונה על תנאי הבעיה, כאשר בחודש נניח כי קיימים m=5 ימים (m>4).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **נמלים** | **יום א'** | **יום ב'** | **יום ג'** | **יום ד'** | **יום ה'** |
|  |  | לב ים |  |  |  |
|  | לב ים |  |  |  |  |
|  |  |  |  | לב ים |  |
|  |  |  |  | לב ים |  |

לפי האלגוריתם שבנינו נבנה טבלאות ליחסי העדפות של הנמלים ושל הספינות:

* כל **נמל** יעדיף שהספינה הראשונה שביקרה בו תישאר לעגון בו עד לסוף החודש.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **נמלים** | **עדיפות 1** | **עדיפות 2** | **עדיפות 3** | **עדיפות 4** |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

* כל **ספינה** תעדיף להישאר לעגון בנמל האחרון בו ביקרה.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ספינות** | **עדיפות 1** | **עדיפות 2** | **עדיפות 3** | **עדיפות 4** |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

כעת, נפעיל את האלגוריתם:

בשלב הראשון יוצרו השידוכים הזמניים הבאים (לפי טבלת העדיפויות של הנמלים): ולאחר-מכן . בשלב זה ניתן לראות כי הספינה "תפוסה" (2.3.2) ולכן נבדוק מה העדיפויות שלה (מעדיפה להישאר לעגון בנמל האחרון בו ביקרה). מעדיפה לעגון ב- מאשר ב-, לכן השידוך הזמני שיישאר יהיה ואילו נמל יהפוך לפנוי.

כעת, נמל יבדוק מי הספינה הבאה שתעגון אצלו ויעדיף שהיא תישאר אצלו, על-פני הספינות הבאות, ניתן לראות כי מדובר בספינה . לא עוגנת עדין בשום נמל ולכן השידוך ישמר כשידוך זמני. נמשיך לעבור על נמלים ו- עד שכל הנמלים לא יישארו פנויים ונסיים.

לסיום נקבל את השידוכים הסופיים הבאים:

וכך יראה הלו"ז המקוצץ שלנו:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **נמלים** | **יום א'** | **יום ב'** | **יום ג'** | **יום ד'** | **יום ה'** |
|  |  | לב ים |  |  |  |
|  | לב ים |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

ניתן לראות שלו"ז זה עונה על תנאי הבעיה.

1. **הוכחת נכונות:**

האלגוריתם מסתיים כאשר השידוך הוא שידוך סופי (יציב), נוכיח שבמקרה כזה לא ייתכן מצב שבו 1 משני המצבים יקרה:

1. יישאר נמל פנוי.
2. תישאר ספינה שלא תעגון בשום נמל.

(כאשר בכל נמל עוגנת ספינה אחת בדיוק).

ההוכחה בחלוקה ל-2 מצבים:

1. **נרצה להוכיח שאף נמל לא יישאר פנוי:** קיימות n ספינות, כל ספינה מסרבת לעגון בלכל היותר n-1 נמלים. לכן, אחרי n(n-1) שלבים, אף ספינה לא תישאר פנויה.

[**נניח בשלילה**](https://he.wikipedia.org/wiki/%D7%94%D7%95%D7%9B%D7%97%D7%94_%D7%91%D7%93%D7%A8%D7%9A_%D7%94%D7%A9%D7%9C%D7%99%D7%9C%D7%94) שנמל נשאר פנוי לאחר שכל הספינות סרבו לעגון אצלו, זאת אומרת שכל הספינות כבר תפוסות (עוגנות בנמלים אחרים). **קיבלנו סתירה**, מכיוון שיש מספר זהה של נמלים ושל ספינות (n) ולכן לא ייתכן שכל הספינות תפוסות. לכן האלגוריתם מסתיים.

1. **נרצה להוכיח שאף ספינה לא תישאר פנויה:** קיימים n נמלים, כל נמל בוחר לכל היותר ב- n-1 ספינות שיעגנו אצלו. לכן, אחרי n(n-1) שלבים, אף נמל לא יישאר פנוי.

[**נניח בשלילה**](https://he.wikipedia.org/wiki/%D7%94%D7%95%D7%9B%D7%97%D7%94_%D7%91%D7%93%D7%A8%D7%9A_%D7%94%D7%A9%D7%9C%D7%99%D7%9C%D7%94) שספינה תישאר פנויה לאחר שכל הנמלים לא בחרו בה לעגון אצלם, זאת אומרת שכל הנמלים בחרו בספינות אחרות ולכן כל הנמלים אינם פנויים. **קיבלנו סתירה**, מכיוון שיש מספר זהה של נמלים ושל ספינות (n) ולכן לא ייתכן שכל הנמלים אינם פנויים אם נשארה ספינה אחת פנויה. לכן האלגוריתם מסתיים.

הראנו שאף-אחד מהשותפים לשידוך לא יישאר פנוי, ז"א לכל זוג תהיה התאמה ובדרך זו נוכל לקצץ את הלו"ז הקיים, ולכן האלגוריתם נכון ומתקיים.

1. **ניתוח סיבוכיות זמן ריצה:**

לולאת ה-while (שורות 2-2.3.2.2) תמיד תעבור על n הנמלים הקיימים.

בנוסף, המעבר על סדר העדיפויות של הספינה (שורות 2.3.2-2.3.2.2), במידה ולא נמצאת התאמה ישר, יכול להגיע עד ל-n-1 נמלים.

לכן, סה"כ סיבוכיות זמן הריצה היא:

**שאלה 2:**

**הכוונת צלעות**. הציגו אלגוריתם שמכריע, בהינתן גרף לא מכוון , האם ניתן לכוון כל אחת מהצלעות, כך שבגרף המכוון שמתקבל, **דרגת הכניסה** של כל קדקוד תהיה **גדולה מאפס**. (לכל צלע ניתן לבחור כיוון יחיד או לחלופין ). כשהתשובה חיובית, על האלגוריתם להחזיר הכוונה של הצלעות - המקיימת את הנדרש. נדרשת תשובה קצרה ומדויקת של עד 8 משפטים.

**פתרון:**

1. **תיאור האלגוריתם:**

לפי נתוני השאלה, אם האלגוריתם מחזיר תשובה חיובית, זה אומר שניתן לכוון כל אחת מהצלעות בתצורה כזאת שבגרף המכוון כל דרגת כניסה של כל קודקוד תהיה גדולה מאפס. אם כל דרגת כניסה גדולה מאפס, זה אומר שכל רכיב קשירות השייך לגרף מכיל מעגל (אנו יודעים זאת מכיוון שבכל גמ"ל יש צומת שלא נכנסת אליו אף קשת). אם כל רכיב קשירות מכיל מעגל ניתן לכוון את צלעות הגרף המקורי בתצורה הנדרשת, והתשובה של האלגוריתם תהיה חיובית.

**האלגוריתם**:

1. נתחיל את הפעלת האלגוריתם מהקודקוד בגרף שיש לו הכי פחות צלעות שמחוברות אליו (נניח בה"כ קודקוד v), בנוסף נאתחל כל קודקוד להיות בצבע לבן.
2. כל עוד קיימים קודקודים לבנים, נמשיך לשכן שהוא קודקוד לבן (לכל היותר v קודקודים):
   1. נבצע סריקה לרוחב מהקודקוד הנ"ל (נפעיל את BFS) ובדרך זו נמצא את כל רכיבי הקשירות של הקודקוד. עבור כל קודקוד (u) עליו נעבור:
      1. נצבע אותו בשחור.
      2. נכווין את הקשת מהקודקוד הקודם אליו (v,u).
   2. נבדוק אם קיים בגרף הקשירות הנוכחי קשת לקודקוד אחר שכבר צבוע בשחור (בתצורה כזאת שיוצר תת מעגל):
      1. אם מצאנו קשת כזו – נמשיך.
      2. אחרת – נחזיר FALSE (גרף הקשירות אינו מכיל מעגל ולכן לא ייתכן שכל דרגת כניסה תהיה גדולה מאפס).
3. נחזיר TRUE (סרקנו את כל הקודקודים וכל רכיבי הקשירות מכילים מעגלים).
4. **הוכחת נכונות:**

נוכיח שהאלוגריתם נכון בחלוקה ל-2 מצבים:

1. נניח שניתן לכוון כל אחת מהצלעות בתצורה כזאת שבגרף המכוון כל דרגת כניסה של כל קודקוד תהיה גדולה מאפס ונוכיח שהאלוגריתם מחזיר, בהתאם, תוצאה חיובית.

**נניח בשלילה שהאלגוריתם מחזיר תוצאה שלילית**, ז"א שהגענו לרכיב קשירות מסוים בו לא מצאנו מעגל. ז"א שקיים קודקוד ברכיב קשירות זה שאין קשת שמוכוונת אליו, ז"א שדרגת הכניסה שלו היא אפס. זאת, בסתירה להנחה שלנו. ולכן, האלגוריתם **מחזיר תוצאה חיובית**.

1. נניח שלא ניתן לכוון כל אחת מהצלעות בתצורה כזאת שבגרף המכוון כל דרגת כניסה של כל קודקוד תהיה גדולה מאפס ונוכיח שהאלוגריתם מחזיר, בהתאם, תוצאה שלילית.

**נניח בשלילה שהאלגוריתם מחזיר תוצאה חיובית**, ז"א שסיימנו לעבור על כל האלגוריתם לפני שהוא יעצר באמצע ויחזיר FALSE. ז"א שמצאנו בהכרח תת מעגל בכל רכיב קשירות שהוא חלק מהגרף. במקרה כזה מתקיים בהכרח שכל דרגת כניסה של כל קודקוד גדולה מאפס, זאת בסתירה להנחה שלנו. לכן, האלגוריתם **מחזיר תוצאה שלילית.**

1. **ניתוח סיבוכיות זמן ריצה:**

נעבור על כל הקודקודים בגרף (V) וגם, לכל היותר נעבור על כל הקשתות (E).

לכן, הסיבוכיות תהיה:

**שאלה 3:**

**בעיית הספיקות (2-SAT).**

הגדרות: נוסחת k-CNF היא נוסחה מהצורה כשלכל פסוקית הצורה , וכל הינו אחד מהליטרלים .

למשל: הינה נוסחת 2-CNF עם k=2 ליטרלים בכל פסוקית, n=3 משתנים, ו-m=4 פסוקיות.

לעומתה, הינה נוסחת 3-CNF עם k=3 ליטרלים בכל פסוקית, n=5 משתנים, ו-m=2 פסוקיות.

השמה הינה פונקציה שמתאימה לכל משתנה ערך "אמת" T או "שקר" F. בהינתן השמה מסוימת, אזי הליטרל מסופק אם ההשמה מקיימת , והליטרל מסופק אם .

הפסוקית מסופקת, אם לפחות אחד מהליטרלים שבה מסופק. הנוסחא כולה מסופקת אם כל הפסוקיות מסופקות.

הנוסחא נקראת ספיקה, אם לפחות אחד מבין ההשמות האפשריות מספקת אותה.

הציגו אלגוריתם יעיל שבהינתן נוסחה בצורה 2-CNF מוצא לה השמה מספקת, ואם אין השמה כזו – מדווח שהנוסחה איננה ספיקה.

הדרכה: העזרו בגרף מכוון G שמותאם לנוסחה .

**פתרון:**

1. **תיאור האלגוריתם:**

על-מנת למצוא האם קיימת עבור נוסחת 2-CNF - השמה מספקת נבצע מספר שלבים שבאמצעותם נוכל לבנות גרף ולבדוק האם רכיבי הקשירות שלו מאפשרים השמה מספקת לליטרלים הנתונים המרכיבים את הפסוקיות.

1. כל עוד קיימות פסוקיות בנוסחה:
   1. עבור כל פסוקית () נבנה שני ביטויים (באמצעותם נבנה בהמשך גרף):
      1. ביטוי ראשון:
      2. ביטוי שני:
2. נבנה מכל הביטויים גרף בתצורה כזאת שכל הליטרלים ה"חיוביים" () יהיו משמאל בטור ואילו כל הליטרלים ה"שליליים" () יהיו מימין בטור (לצורך הנוחות שלנו בלבד).
3. עבור כל ליטרל (עד שנגיע ל- i=1):
   1. נריץ אלוגריתם BFS עבור .
   2. נבדוק האם הוא חלק מהעץ שקיבלנו עבור :
      1. אם לא – נציב עבור כל הליטרלים בעץ T ועבור כל הליטרלים הנגדיים להם בהתאם F. נחזיר TRUEונסיים את האלגוריתם.
      2. אחרת (אם כן) – ההשמה לא אפשרית ונעבור לבדוק את במקום:
         1. נריץ אלוגריתם BFS עבור .
         2. נבדוק האם הוא חלק מהעץ שקיבלנו עבור :
            1. אם לא – נציב עבור כל הליטרלים בעץ T ועבור כל הליטרלים הנגדיים להם בהתאם F. נחזיר TRUEונסיים את האלגוריתם.
            2. אחרת (אם כן) – ההשמה לא אפשרית ומכיוון שבדקנו את שני ה"צדדים" של לא קיימת השמה אפשרית. נחזיר FALSEונסיים את האלגוריתם.
4. **הוכחת נכונות:**

נוכיח את נכונות האלגוריתם בחלוקה ל-2 מקרים:

1. נניח שקיימת השמה מספקת, ולכן האלגוריתם יחזיר TRUE.

**נניח בשלילה שלא קיימת השמה מספקת** והאלגוריתם החזיר TRUE. אם האלגוריתם החזיר TRUE זה אומר שהתקיימה אחת מבין 2 אפשרויות:

* 1. אינו חלק מהעץ שקיבלנו עבור .
  2. אינו חלק מהעץ שקיבלנו עבור .

אם התקיימה אחת מהאופציות האלו, ז"א שמצאנו דרך למצוא השמה לעץ של כך שהליטרלים בו יהיו T או F בהתאם, ואילו הליטרלים בעץ הנגדי יהיו F או T בהתאם. אם נסתכל על הנוסחה, בדרך זו נגיע למצב שכל פסוקית תחזיר T (כיוון שנקבל ליטרל שהוא F או ליטרל שהוא T) ולכן כשנבצע "וגם" בין כל הפסוקיות, ז"א, במילים אחרות, שמצאנו השמה מספקת – בסתירה להנחה. לכן, קיימת השמה מספקת שגורמת לכך שהאלגוריתם יחזיר TRUE.

1. נניח שלא קיימת השמה מספקת, ולכן האלגוריתם יחזיר FALSE.

**נניח בשלילה שקיימת השמה מספקת** שגרמה לאלגוריתם להחזיר FALSE. אם האלגוריתם החזיר FALSE זה אומר ש- הוא חלק מהעץ שקיבלנו עבור וגם הוא חלק מהעץ שקיבלנו עבור . מכיוון שהליטרלים הנגדיים של נמצאים בשני רכיבי קשירות ואינם מופרדים, לא נוכל לקבוע איזה מהם ( או ) הוא T או F בהתאמה. ז"א לא נוכל לקבוע השמה מספקת לליטרלים, ולכן לא נוכל לקבוע השמה מספקת לפסוקיות ולנוסחה כולה. זאת, בסתירה להנחה שקיימת השמה מספקת. ולכן ברור שבמקרה ולא קיימת הנחה מספקת האלגוריתם יחזיר FALSE.

הראנו ששני המקרים מתקיימים, מכאן שהאלגוריתם נכון.

1. **ניתוח סיבוכיות זמן ריצה:**

נגדיר כ-n את מספר הפסוקיות בנוסחה.

באלגוריתם אנחנו נעבור על כל הפסוקיות - .

בנוסף עבור כל פסוקית נבנה 2 ביטויים - .

לאחר-מכן ניצור מכל הביטויים גרף - .

נבצע BFS עבור כל קודקודים מסוימים - , במקרה הגרוע עבור כל קודקוד נבדוק BFS עבור הליטרל הנגדי שלו - .

סה"כ זמן הריצה של האלגוריתם יהיה: .

**שאלה 4:**

**מסלולים מזעריים דרך קדקודים מועדפים**. נתון גרף מכוון , צמד קדקודים , ותת קבוצה של קודקודים מועדפים , המקיימת וכן .

לכל מסלול P בגרף נסמן ב- את אורך המסלול (=מספר הצלעות במסלול), וב- את מספרם של קודקודי U במסלול.

הציגו אלגוריתם למציאת מסלול מ-s ל-t, שמבקר ב-U בדיוק פעמיים, ושאורכו מזערי מבין כל המסלולים הללו. כלומר, האלגוריתם נדרש להחזיר מסלול מ-s ל-t כך ש-, וכך שאם ישנם מסלולים נוספים P' מ-s ל-t המקיימים -, אז .

בשאלה זו מותרים מסלולים לא פשוטים, כלומר מותר למסלול לעבור דרך קדקוד מסוים, ואפילו דרך צלע מסוימת יותר מפעם אחת. למשל, נניח שהגרף הינו משולש מכוון ללא שום קדקוד או צלע נוספת, ושהקבוצה המועדפת הינה . אזי המסלול היחיד, ולכן גם המזערי שמקיים - הינו המסלול לא פשוט .

הדרכה: העזרו ברדוקציה לגרף אחר .

**פתרון:**

1. **תיאור האלגוריתם:**

בהתאם לנתוני הבעיה ועל-מנת למצוא פתרון אולטימטיבי, נבנה גרף חדש G' שבו בחלק העליון יופיע הקודקוד s, בחלק התחתון יופיע הקודקוד t, במידה וקיימים קודקודים נוספים ב-V שאינם שייכים ל-U הם יופיעו גם כאן כחלק מטור אמצעי זה (מתחת ל-s או מעל t), ביניהם יופיעו שני טורים לאורך:

* בטור שמאל יופיעו כל הקודקודים () ששייכים ל-U.
* בטור ימין יופיעו שוב כל הקודקודים ששייכים ל-U בתצורה שונה, במקום יופיעו .

נשרטט את גרף G ב-G' בצורה כזו שכל קשת שמחברת בין קודקוד ששייך ל-U תופיע פעמיים, גם בטור השמאלי וגם בטור הימני.

**האלגוריתם**:

* + - 1. אתחול – הגרף G' שבנינו.
      2. משתנה שנאתחל אותו להיות אפס.
      3. משתנה שנאתחל אותו להיות אינסוף.
      4. נריץ BFS על צומת s, נחפש את המסלול הקצר ביותר ל-t, כאשר נוסיף באלג' BFS את התנאים הבאים:
  1. נריץ BFS רק במידה והמסלול אינו מוגדר FALSE (הסבר למטה מה הכוונה).
* נציין שכשמריצים BFS על קודקודים ששייכים ל-U המסלול יעבור פעם אחת בצד שמאל, פעם אחרת בימין וכן הלאה (לכן יעבור ב-U לפחות פעמיים גם אם קיימים פחות מ-2 קודקודים ב-U).
  1. כאשר עוברים בקודקוד ששייך ל-U (כמובן שכולל גם את הקודקודים בתצורתם המשוכתבת - ) נגדיל את ב-1.
  2. אם סיימנו את ריצת ה-BFS, מצאנו את המסלול הקצר ביותר מ-s ל-t והגענו למצב שבו :
     1. אז – נחזיר את המסלול P שמצאנו.
     2. אחרת – נגדיר מסלול זה כ-FALSE.

1. **הוכחת נכונות:**

נוכיח באינדוקציה.

* **בסיס האינדוקציה**: יהי גרף G ובו קודקוד יחיד u ששייך ל-U ובהתאם לכך 3 קוקודים ששייכים ל-V (s,t,u). נבנה את גרף G' בהתאם לפירוט מעלה ונקבל את הקודקודים (s,t,u,u') השייכים לו.

נפעיל את האלגוריתם ונראה כי במידה ו-s בעל קשת מכוונת ל-t המסלול שהתקבל יהיה מסלול FALSE (משום ש- ) ולכן נצטרך להריץ שוב BFS ונעבור פעם אחת ב-u ופעם שניה ב-u' עד שנמצא מסלול מ-s ל-t שעובר פעמיים ב-U (כמו המשולש שבדוגמה של השאלה). במצב כזה נחזיר את המסלול:

.

* **צעד האינדוקציה:**
  + - הנחת האינדוקציה: נניח כי האלג' מחזיר את המסלול הקצר ביותר שעובר פעמיים בדיוק ב-U עבור גרף G שבו v קודקודים, מתוכם u קודקודים שייכים ל-U.
    - הוכחת האינדוקציה: נוכיח כי האלג' מחזיר את המסלול הקצר ביותר שעובר פעמיים בדיוק ב-U עבור גרף G שבו v+1 קודקודים, מתוכם u+1 קודקודים שייכים ל-U. (גרף זה זהה לגרף מהנחת האינדוקציה, נוספו לו קודקוד וקשת).

נבנה את גרף G' בהתאם לנדרש. נריץ BFS על s ונקבל מסלול P אותו קיבלנו בהנחת האינדוקציה, ידוע בוודאות שקיים מסלול כזה שעובר פעמיים ב-U כיוון שהנחנו שהוא קיים. בהתאם לכך, נחזיר את המסלול שמצאנו.

הוכחנו שגם עבור גרף G שבו v+1 קודקודים ניתן להחזיר את המסלול הקצר ביותר שעובר פעמיים בדיוק ב-U.

ולכן האינדוקציה מתקיימת לכל v, וברור כי האלג' מתקיים ונכון.

1. **ניתוח סיבוכיות זמן ריצה:**

בניית הגרף G' דורשת - .

הרצת BFS גם הוא דורש , במקרה הפחות טוב נריץ BFS שוב.

סה"כ זמן הריצה יהיה .