**ממן 14**

1)

**הרעיון המרכזי של האלגוריתם**:

*נגדיר את בתור מחיר המסלול המינימלי שמתחיל בשכבה השמאלית ומסתיים באיבר .*

*לפיכך, נגדיר את להיות מטריצה מסדר , שכן אנו זקוקים לשורות במהלך החישוב של נוסחת הנסיגה (אם כי ערכים אלה אינם תורמים לפתרון, שכן אנו מאתחלים בהם ).*

*מחיר המסלול המינימלי מהשכבה השמאלית לשכבה הימנית הוא .*

*כעת, נוכל ממחיר המסלול המינימלי לשחזר את קודקודי המסלול כך:*

*השחזור יתבצע מהסוף להתחלה. נמצא ערך ה- עבורו שווה לערך מחיר המסלול המינימלי. נסמן אותו בתור .*

*כעת, נוכל לחשב בעזרת בצורה הבאה:*

* *מועמד להיות או או .*
* ערך ה- המתאים *יקיים . אם המטריצה ומחיר המסלול המינימלי תקינים, יהיה קיים ערך מתאים.*

*נחשב כך אחורנית את כל הים מ ועד .*

*בסיום נקבל סדרת אינדקסים אשר מסמלות את ערכי ה- של הקודקודים המתאימים במסלול בעל המחיר המינימלי המבוקש. הקודקוד ה-i במסלול זה הוא הקודקוד .*

**האלגוריתם**:

נגדיר מטריצה מגודל *, ונמלא את ערכיה:*

1. *לכל , נגדיר .*
2. *לכל , נגדיר .*
3. *לכל נגדיר:*

*נחשב את min\_cost – עלות המסלול המינימלית מהשכבה השמאלית אל השכבה הימנית:*

1. *נחשב את כאשר ואת התשובה נאחסן במשתנה min\_cost*

*נשחזר את המסלול מתוך min\_cost:*

1. *נמצא ערך ה- מ1 עד n עבורו שווה לערך מחיר המסלול המינימלי. נסמן אותו בתור .*
2. *לכל :*
   1. *נחשב את לפי האלגוריתם שהצגנו למעלה.*
3. *נחזיר את התאים .*

**ננמק את נכונות האלגוריתם לפתרון הבעיה:**

*בשורות 1-3 אנו ממלאים את ערכי המטריצה . נבחין שהנוסחה הרקורסיבית מחשבת כהלכה את מחיר המסלול המינימלי, שכן עבור כל תא במטריצה, אם כבר ידועים לנו ערכי כל העמודה שקדמו לתא, אפשר לבחון מי מהאופציות זולה יותר (שמאלה, שמאלה-למעלה או שמאלה-למטה), ונבחר אותו בתוספת עלות התא שאנו נמצאים בו כרגע. בנוסף ברור שהמחיר התקבל הוא המינימלי מכל שלושת האופציות האפשריות.*

*משום שאנו מבצעים את החישוב משמאל לימין (מ עד ), בכל חישוב של תא בהכרח יתקיים שהעמודה הסמוכה לתא משמאל תהיה מלאה בערכים נכונים.*

*בשורה 4 אנו בוחנים את המחיר המינימלי של כל המסלולים שמתחילים בשכבה השמאלית ומסתיימים בתא ספציפי בשכבה הימנית. מינימום זה הוא אכן מחיר המסלול המינימלי משכבה שמאלית לשכבה ימנית.*

*כעת, בשורה 5-7 אנו משחזרים את המסלול עצמו מתוך המחיר המינימלי שמצאנו, באופן הפוך לאופן שבו חישבנו את המחיר בשורות 1-4.*

*לפיכך, האלגוריתם פותר את הבעיה.*

**ננתח את סיבוכיות זמן הריצה:**

*במהלך שורות 1-3 אנו מכניסים ערכים לכל אחד מתאי המטריצה . במטריצה יש תאים.*

*חישוב כל תא בטבלה אורך זמן קבוע: שורה 1 ו-2 ברורות, ובשורה 3 נבחין שאנו ניגשים לתא במטריצה, בעלות קבועה גם כן.*

*לפיכך, שורות 1-3 רצות בזמן של .*

*שורה 4 מבצעת מינימום מתוך איברים ולכן רצה בזמן לינארי.*

*שורה 5 רצה על איברים, ולכן רצה בזמן לינארי.*

*שורות 6-7 רצות על עמודות, ובכל פעם מבצעות עבודה קבועה. לפיכך, זמן הריצה גם כאן הוא לינארי.*

*לפיכך, שורות 4-7 רצות בזמן לינארי.*

***סה"כ זמן ריצה של , כנדרש.***

2)

**הרעיון המרכזי של האלגוריתם**:

*ראשית נמיין את התיבות לפי הרוחב (ערך הw) שלהן בסדר יורד. ברור שלכל תיבות מתקיים*

*כעת נגדיר את להיות גובה המגדל היציב המקסימלי שבו התיבה ה-i היא בראש המגדל. נבחין שבמגדל אופטימלי זה, כל תיבה מקיימת , שכן אם תיבה בראש המגדל, כל שאר התיבות צריכות להיות גדולות באורך וגדולות ברוחב ממנה.*

*לפיכך, נוכל להגדיר את בעזרת הנוסחה הרקורסיבית:*

*ז"א, בעת חישוב גובה המגדל היציב המקסימלי עם התיבה ה-i בראש המגדל, נבחן מספר מקרים:*

* *מקרה א' – מגדל שבו התיבה i נמצאת לבד.*
* *מקרה ב' – מגדלים קיימים אליהם אנו מוסיפים את התיבה i לראש המגדל. נבחן רק את המגדלים המקיימים ו-, ולכן אנו יודעים שחוקי לשים את התיבה i בראש המגדל.*

*נבחר את גובה המגדל היציב המקסימלי שנמצא.*

*אם נחשב את ערכי הOPT מ ועד בסדר זה, נמלא את מערך הOPT הדרוש.*

*בנוסף, נגדיר מערך חדש בגודל n בשם previous\_box שיכיל במקום ה-i את התיבה j שנמצאת בראש המגדל שעליו שמנו את התיבה i (במילים אחרות, את המקסימום שבחרנו במקרה ב'). אם מדובר במקרה א', נציב NIL, שכן אין שום תיבה ששמנו עליה את התיבה i (המגדל הוא מגדל של תיבה אחת – i).*

*גובה המגדל היציב המקסימלי הדרוש יהיה .*

*כעת, נוכל מגובה המגדל היציב המקסימלי לשחזר את סדר התיבות שנבחרו, כך:*

*השחזור יתבצע מהסוף להתחלה. נמצא את ערך ה-i עבורו שווה לגובה המגדל היציב המקסימלי. נסמן אותו בתור .*

*כעת, אנו יודעים שתיבה נמצאת בראש המגדל. נפנה למערך previous\_box במקום ונקבל את התיבה שבאה אחריה, ונסמן אותה בתור , וחוזר חלילה עד שנתקל בNIL (סוף המגדל). בסיום נקבל סדרת אינדקסים אשר מסמלות את האינדקסים של התיבות ברשימה הממוינת שלהן, לפי הסדר שבו הן מופיעות במגדל היציב המקסימלי.*

**האלגוריתם**:

1. *נמיין את התיבות לפי הרוחב (ערך הw) שלהן בסדר יורד.*

*נגדיר מערך OPT בגודל n וערך previous\_box בגודל n, ונמלא את ערכיהן:*

1. *לכל :*
   1. *נחשב את גובה המגדל המקסימלי לפי נוסחאת הנסיכה ונכניס את התשובה אל .*
   2. *אם הגובה המקסימלי נבחר לפי מקרה א', נבצע .*
   3. *אחרת, אם הגובה המקסימלי נבחר לפי מקרה ב', ו j הוא האינדקס שנבחר במקסימום, נבצע .*

*נחשב את max\_length – גובה המגדל היציב המקסימלי:*

1. *נחשב את כאשר ואת התשובה נאחסן במשתנה max\_length.*

*נשחזר את המסלול מתוך max\_length ו previous\_box:*

1. *נמצא את ערך ה-i מ1 עד n עבורו . נסמן אותו בתור .*
2. *נסמן , .*
3. *כל עוד :*
4. *נחזיר את הרשימה .*

**ננמק את נכונות האלגוריתם לפתרון הבעיה:**

*בשורה 2 אנו מחשבים את ערכי OPT.*

*ראשית, נבחין שהצעד הרקורסיבי של חישוב תיבה משתמש בתוצאות החישוב של תיבות אחרות המקיימות בלבד. ולכן, כאשר אנו מחשבים את ערכי OPT מ ועד , לא נתקל באף שלב בניסיון קריאה לערך בOPT שעוד לא איתחלנו אותו.*

*בכל צעד רקורסיבי אנו בוחנים איזה מקרה יניב את גובה המגדל המקסימלי: מקרה א', שבו אנו שמים את התיבה i לבדה במגדל, ומקרה ב', בו אנו שמים את התיבה i מעל למגדל קיים אחר.*

*נבחין שהמגדלים היחידים שאנחנו יכולים לשים עליהם את התיבה i הם מגדלים שבראשם יש תיבה j המקיימת ו- , שכן אז משום שהתיבות ממוינות, יתקיים שאורך ורוחב התיבה בראש המגדל גדולים מאורך ורוחב התיבה i, ורק אז נוכל להניח את תיבה i על j ועדיין לקבל מגדל יציב.*

*בנוסף, נדרוש שהמגדלים הללו יהיו מקסימליים, ואז נבחן את כל המקרים לעיל ונבחר את זה שייתן לנו גובה מגדל מקסימלי.*

*בשורה 3 אנו בוחרים את הגובה המקסימלי של מגדל יציב מבין כל n האפשרויות שחישבנו עד כה, שבהם תיבה i נמצאת בראש המגדל.*

*בשורות 4-6 אנו משתמשים במערך previous\_box שבנינו, ומצביע עבור כל תיבה i על התיבה שנמצאת לפניו במגדל המקסימלי שבראשו נמצאת התיבה i (מה שגובהו נמצא ב). לכן, ריצה אחורנית על כל התיבות בצורה זו מאפשרת לנו לסרוק את כל התיבות במגדל היציב המקסימלי (בסדר הפוך – מראש המגדל לבסיסו).*

*לפיכך, האלגוריתם פותר את הבעיה.*

**ננתח את סיבוכיות זמן הריצה:**

*שורה 1 רצה בזמן של .*

*במהלך שורה 2 אנו מכניסים ערכים לכל אחד מתאי OPT, סה"כ n תאים.*

*בחישוב כל תא אנו מבצעים במקרה הגרוע קריאות למערך OPT, שזמן הקריאה ממנו קבוע, ואז אנו מחפשים את המקסימום מכל הקריאות הללו. לכן, חישוב כל תא לוקח זמן ריצה של .*

*לפיכך, שורה 2 רצה בזמן של .*

*שורה 3 רצה בסיבוכיות לינארית. שורות 4-5 ו-7 רצות בסיבוכיות קבועה.*

*שורה 6 רצה על כל התיבות במגדל המקסימלי, שבמקרה הגרוע מדובר בn תיבות, ולכן רצה בזמן לינארי. בכל איטרציה של הלולאה מתבצעת עבודה קבועה.*

*סה"כ זמן ריצה של , כנדרש.*

3)

א)

*ראשית נגדיר את הפולינומים באופן כללי:*

*בכל הפיתרון נתעלם מפתרונות טריוויאליים כמו מקדמים שכולם 0, שכן אלה פתרונות טריוויאליים שאינם יקדמו אותנו בדרך לנוסחת נסיגה שתעזור לנו בהמשך.*

*נדרוש:*

*שכן האינטרפולציות של נקודות ו- צריכות לקבל אותו ערך בנקודה , והאינטרפולציות של נקודות ו- צריכות לקבל אותו ערך בנקודה*  (אחרת, האינטרפולציות ודאי אינן תקינות).

נצטרך להציג דרישות נוספות ע"מ שהנוסחה בשאלה תניב את נוסחה (1) ואת נוסחה (2).

ע"מ שיתקיים (1) נדרוש . מ נקבל ש .

ע"מ שיתקיים (2) נדרוש . מ נקבל ש .

מ ו- נקבל:

*מכאן ש או . הדרישה איננה הגיונית, שכן בחירת הנקודות שרירותית ולא תלויה, וכמובן שאנו יכולים לבחור 2 נקודות שלא מקיימות משוואה זו. לכן הדרישה הכרחית. מכאן ש.*

*כעת נציב בדרישה ונקבל:*

ולכן .

לסיכום נקבל:

*ולפי כל הדרישות הגדרנו, הפולינומים הללו יקיימו שוויון במשוואה שבתרגיל:*

ב)

**הרעיון המרכזי של האלגוריתם**:

*נניח ונתונות לנו n נקודות ונרצה לחשב את האינטרפולציה . נבחין שבעזרת נוסחת הנסיגה שמצאנו בסעיף א', נוכל לחשב את האינטרפולציה בעזרת האינטרפולציות ו .*

לכן, אלגוריתם תכנון דינמי יפתור את הבעיה בקלות. נגדיר את להיות פולינום האינטרפולציה של הנקודות ., כאשר OPT תהיה מטריצה מגודל .

בתור אתחול נגדיר לכל i, , שזהו פולינום האינטרפולציה של הנקודה .

כעת נמלא את שאר הטבלה OPT עבור כל ערכי i,j המקיימים , לפי נוסחת הנסיגה שמצאנו בסעיף א' (במקום ניגש ל ).

*התשובה שמעניינת אותנו היא .*

**האלגוריתם**:

*נגדיר מערך בגודל*  ונאתחל אותו כך:

1. לכל , .
2. לכל :
   1. נגדיר
   2. לכל :
      1. חשב את לפי הנוסחה הרקורסיבית שמצאנו בסעיף א' (במקום ניגש ל ).
      2. נגדיר
3. נחזיר את .

**ננמק את נכונות האלגוריתם לפתרון הבעיה:**

*הפיתרון נכון שכן אנו מתבססים על הנוסחה שחישבנו בסעיף א' ועל נכונותה.*

*את OPT אנו מחשבים בסדר לפי אלכסונים: בשלב 1 אנו מחשבים את הערכים עבור האלכסון הראשי, ובשלב 2 אנו מחשבים לפי הסדר את האלכסונים בתור מימין (ז"א, בשלב 2 בראשית, כאשר , נחשב את האלכסון שמתחיל בתא ומסתיים ב, ולאחר מכן בשלב בלולאה, נחשב את האלכסון שמתחיל בתא ) ומסתיים ב .*

*בנוסף,* נבחין שבעת חישוב התא כאשר , אנו זקוקים *רק לערכים ו , וכאשר אנו לא זקוקים כלל לערכי OPT.*

*משום שאנו מתעניינים אך ורק בערכי OPT המקיימים , ומשום שחישבנו את ערכי OPT בסדר נכון, לא נתקל באף שלב בניסיון קריאה לערך בOPT שטרם איתחלנו.*

*לפיכך, האלגוריתם פותר את הבעיה.*

**ננתח את סיבוכיות זמן הריצה:**

*שורה 1 רצה בזמן לינארי.*

*בשורה 2 יש לנו לולאה שרצה פעמים, ובתוכה לולאה מקוננת שרצה במקרה הגרוע פעמים. הלולאה המקוננת מבצעת בכל פעם עבודה קבועה (משום שלפי הנחת השאלה, ניתן להניח שפעולות אריתמטיות יחשבו כפעולות אלמנטריות שמחשבים בסיבוכיות קבועה).*

*לכן, שורה 2 רצה ב.*

*שורה 3 רצה בזמן קבוע.*

*סה"כ זמן ריצה של , כנדרש.*

ג)

*נבדוק את האלגוריתם מסעיף ב'. לפי הפולינום והנקודות הנתונות, מצאנו את הנקודות הבאות:*

*לפי הנחיית השאלה, מנקודות אלה נוכל לשחזר את הפולינום הנתון בשאלה, שכן יש לנו 5 נקודות בפולינום ממעלה 4. בסעיף א' מצאנו את התנאי הרקורסיבי שיעזור לנו בחישוב הפולינום, ובסעיף ב' מימשנו אותו. נבדוק כעת את מימוש זה.*

*נציג טבלת מעקב:*

*במקרה זה . נגדיר בראשית מטריצה OPT בגודל ונמלא את ערכיה:*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Step* | *k* | *i* | *j* |  |
| *1* |  | *1* |  |  |
| *1* |  | *2* |  |  |
| *1* |  | *3* |  |  |
| *1* |  | *4* |  |  |
| *1* |  | *5* |  |  |
| *2* | *2* | *1* | *2* |  |
| *2* | *2* | *2* | *3* |  |
| *2* | *2* | *3* | *4* |  |
| *2* | *2* | *4* | *5* |  |
| *2* | *3* | *1* | *3* |  |
| *2* | *3* | *2* | *4* |  |
| *2* | *3* | *3* | *5* |  |
| *2* | *4* | *1* | *4* |  |
| *2* | *4* | *2* | *5* |  |
| *2* | *5* | *1* | *5* |  |

*ואכן מתקיים , כנדרש בתרגיל.*

4)

א)

*האלגוריתם מחשב את המסלול בעל העלות המינימלית מהקודקוד הנתון אל כל קודקוד . את התשובות הוא מאחסן במערך החד-ממדי .*

*נוכיח זאת באינדוקציה על האיטרציה ה-i של הלולאה החיצונית, החל מ.*

*נגדיר: מסלול הוא מסלול n-נגיש אם אורכו הוא בין 0 ל-n (כולל קצוות).*

*טענת האינדוקציה: בסוף האיטרציה ה-i של הלולאה החיצונית, לכל :*

* *כאשר אז אין מסלול i-נגיש מr ל-v.*
* *כאשר (כאשר k סופי ואי-שלילי), אז קיים מסלול i-נגיש מr ל-v שעלותו k, וk קטן או שווה מהעלות המינימלית של כל המסלולים ה i-נגישים מr ל-v.*

*בסיס האינדוקציה: בסוף האיטרציה ה-0 של הלולאה (למעשה, לפני האיטרציה הראשונה – איטרציה 1), הצומת היחיד שיש אליו מסלול באורך 0 הוא r עצמו, ו הוא אכן לא אינסופי (הוא מכיל 0).*

*כל שאר הצמתים אינם 0-נגישים, והם אכן מקיימים .*

*צעד האינדוקציה: נניח שהטענה נכונה עבור ונוכיח אותה עבור :*

*ננסה להראות בשלילה שהתנאים לא מתקיימים. כאשר נכשל בכל ההפרכות, נוכיח שטענת האינדוקציה נכונה עבור .*

*נניח בשלילה שקיים צומת עבורו הערך סופי, אך אינו מייצג עלות מסלול n+1-נגיש מr לv. לפי הנחת האינדוקציה, כל הערכים המייצגים מסלולים n-נגישים מr לצומת כלשהי הינם תקינים ומייצגים מסלולים תקינים. ז"א, אם קיים ערך שגוי, הוא נוצר במהלך האיטרציה הנוכחית. נסמן ב את הקשת שבסריקתה עודכן הערך השגוי. לפי האלגוריתם, אנו מעדכנים . לפי הנחת האינדוקציה הוא משקל מסלול n-נגיש קיים מr לu. אנו מוסיפים לו את עלות הקשת , לכן ברור ש הוא מסלול n+1-נגיש קיים מr לv, בסתירה להנחת השלילה.*

*נניח בשלילה שקיים צומת עבורו קיים מסלול -נגיש מr ל-v שעלות המסלול שלו k קטנה מ. הקשת האחרונה במסלול כזה תהיה . לפי הנחת האינדוקציה, הוא עלות המסלול הn נגיש המינימלי מr לu. לפי האלגוריתם, אנו עוברים על כל הקשתות מu לv, ומשתמשים בקשת שתביא את עלות המסלול מr לv אל המינימום. לכן, לא יתכן שתבחר העלות כמינימלית, שכן אם נשתמש בקשת במקום נגיע לעלות נמוכה יותר, בסתירה להנחת השלילה.*

*כעת, נניח בשלילה שקיים עבורו אך קיים מסלול n+1-נגיש מr לv. אם קיים מסלול כזה, ודאי שגם ניתן לחשב את עלותו – k. כמו כן, הקשת האחרונה בו היא . אם נמחק את הקשת מהמסלול, נקבל מסלול n-נגיש מr לu. לפי הנחת האינדוקציה, מוגדר כראוי, עם עלות המסלול מr לu, שהיא כמובן סופית. ז"א, במהלך איטרציה זו, בעת המעבר על הקשת v בשלב מסוים נגיע אל הקשת , שכן היא קשת המגיעה אל הצומת v. בשלב זה, יבחן הסכום . שתי עלויות אלה סופיות, ולכן גם הסכום סופי. סכום זה יבחן אל מול הערך שקיים ב והמינימום ישמר בחזרה אל . אם , אז כעת יוצב הסכום שחושב. אחרת, יתכן שהסכום שחושב יכנס או לא, אך ודאי , בסתירה להנחת השלילה.*

*לאחר שהראנו שהתנאים אכן מתקיימים, הראנו שטענת האינדוקציה נכונה עבור .*

ב)

*לפי הכלל שהוכחנו באינדוקציה בסעיף הקודם, האיטרציה ה-i בלולאה דואגת שבכל הצמתים אנו מציבים במערך עלות מסלול i-נגיש מינימלית. לגבי מסלולים j-נגישים המקיימים לא ניתן להגיד דבר, וזאת משום שעדיין לא "הספקנו" להגיע אליהם ע"מ לעדכן את ערכם (אם צריך).*

*אנו אכן רואים שבמהלך האיטרציות אנו רצים על כל הצמתים ומחפשים לראות האם יש להם שכן שמצביע אליהם, ולו יש עלות סופית כלשהי. אם כן, אנו מעדכנים עלות זו, ורצים שוב על כל הצמתים, עד שלא מתבצע שום עדכון.*

*קל לראות שאם קיים מסלול באורך מצומת r אל צומת v כלשהי, והצמתים מסודרים בסדר לקסיקוגרפי* ***הפוך*** *לסדר שלהם במסלול, אז בכל איטרציה חיצונית יעודכן ערכו של צומת אחת בלבד במסלול, ז"א, יהיו n איטרציות חיצוניות, כאשר הראשונות יעדכנו ערכים במערך A, והאחרונה תרוץ על כל הצמתים, תוודא שאכן אין עוד שינויים, ותצא.*

*אם נמחק את כל שאר הקשתות והצמתים שלא קשורים אל מסלול זה, נקבל שיש n צמתים במסלול לעיל, וגם יש n צמתים בגרף כולו, ולכן נרוץ על הגרף n איטרציות.*

*נבחין גם שאם לא הינו מוחקים את הצמתים והקשתות, לפי הטענה שהוכחנו בסעיף הקודם, לא יתכן שקיים מסלול n-נגיש שלא יתגלה באיטרציה ה-n.*

*לפיכך – מספר המרבי של האיטרציות שמתבצעות בלולאה החיצונית על גרפים בעלי n קודקודים היא n. ().*

*נציג סדרת גרפים כזאת בצורה פורמלית:*

*נגדיר גרף הבא בגודל n שמשקל כל קשתותיו בו הוא 1:*

לפי ההסבר לעיל, מספר האיטרציות על גרף זה יהיה .

ג)

*אם הינו בונים את אותו גרף מהסעיף הקודם באופן בו כל הצמתים מסודרים בסדר לקסיקוגרפי* ***התואם*** *לסדר שלהם במסלול, אז באיטרציה הראשונה החיצונית יחושבו כל ערכי A, ובאיטרציה השנייה נגלה שאין עוד ערכים שלא חושבו ונצא מן הלולאה: סה"כ 2 איטרציות חיצוניות.*

*נציג סדרת גרפים כזאת בצורה פורמלית:*

*נגדיר גרף הבא בגודל n שמשקל כל קשתותיו בו הוא 1:*

לפי ההסבר לעיל, מספר האיטרציות על גרף זה יהיה 2.

*ברור שמספר הצלעות ב שווה לשל – לשניהם יש בדיוק n צלעות.*