**ממ"ן 14 אלגוריתמים**

**שאלה 1**

נגדיר OPT(i,j) כמחיר מזערי של מסלול מהנקודה (i,j) לשכבה הימנית ביותר (שבה i=n).

אם i=n אז המסלול מכיל רק נקודה אחת והמחיר שלו זה מחיר הנקודה לכן במקרה זה OPT(i,j)=C(i,j). במקרה הכללי כלומר עבור נחפש את המסלול המזערי מבין שלושת הצעדים החוקיים ונוסיף את מחיר הנקודה הנוכחית לכן נקבל OPT(i,j)=MIN{OPT(i+1,j+1), OPT(i+1,j), OPT(i+1,j-1)} + C(i,j). נשים לב שיכולים להיות מצבים שבהם נחרוג מגודל השריג (למשל עבור נקודות שבהן j=n ואז הצעד ל(i+1,j+1) חורג מהשריג) לכן נגדיר שאם או אז וכך *בוודאות נקודות שלא נמצאות בשריג לא ייבחרו בחיפוש המסלול המינימלי.*

*באלגוריתם תחילה ניצור לפי הגדרת* OPT *מטריצה* M בגודל  *שבה* M[i,j] *הוא* מחיר מזערי של מסלול מהנקודה (i,j) לשכבה הימנית ביותר *לאחר מכן נמצא את האיבר המינימלי בשכבה השמאלית ביותר של* M, *הערך של איבר זה הוא המחיר המזערי של מסלול מהשכבה השמאלית ביותר לשכבה הימנית ביותר והאינדקסים של איבר זה מייצגים את הנקודה הראשונה במסלול זה ולסיום נשחזר את המסלול במחיר מזערי לפי* M *(כל פעם נתקדם לערך המינימלי של* M *מבין הצעדים החוקיים).*

***האלגוריתם***

*יהי* C *מטריצת המשקלים של הנקודות ו* M *מטריצה בגודל .*

*שגרת העזר* OPT ש*מקבלת אינדקסים* i,j:

*(1)* אם או *החזר*

*(2) אחרת אם* M[i,j] *כבר מכיל ערך החזר* M[i,j]

*(3) אחרת אם* i=n החזר C[i,j]

(4) אחרת החזר MIN{OPT[i+1,j+1], OPT[i+1,j], OPT[i+1,j-1]} + C[i,j].

*השגרה הראשית:*

*(1) לכל* i *מ*n- *עד 1*

*(1.1) לכל* j *מ*n- *עד 1*

*(1.1.1)*

*(2) ונסמן את ה*j *שעבורו מתקבל הערך המינמילי ב*

*(3) עבור* i *מ1 עד* n-1

*(3.1) ,*

*(3.2) אם וגם אז ,*

*(3.3) אם וגם אז ,*

*(4) החזר את סדרת הנקודות שהם המסלול בעל המחיר המזערי*

*נכונות*

*נוסחת הנסיגה* OPT *נכונה מכיוון שהיא בודקת את כל שלושת הצעדים האפשריים ומתייחסת למקרי הקצה של* *חריגה מהגבולות, הנוסחה תמיד תעצור כי כל צעד מגדיל את* i *ב1 לכן בהכרח מתישהו נגיע ל*i=n *(שהנוסחה מחזירה עבורו את* C[i,j]*).*

*האלגוריתם משתמש בנוסחת נסיגה זו ליצירת* M, *ידוע לנו שהמסלול מתחיל מהעמודה* i=1 *אבל לא ידוע לנו איזה שורה לכן* *אנחנו מחפשים את האיבר המינימלי בעמודה זו ואז ממנו משחזרים את המסלול שוב על ידי בדיקת כל הצעדים האפשריים לפי הערכים ב*M*.*

*זמן ריצה*

*הלולאה ליצירת M רצה עם* i *ו*j *מ*n *ל1 מה שמבטיח שתמיד יהיו ערכים בעמודה* הi+1 *ולכן שגרת העזר* OPT *לא תקרא לעצמה רקורסיבית יותר מפעם אחת מה שאומר שהיא רצה בזמן קבוע ובסך הכל זמן הריצה של יצירת* M *הוא .*

*מציאת האיבר המינימלי בשכבה השמאלית דורשת בדיקה של* n *איברים לכן לוקחת וגם השיחזור מתבצע בלולאה מ1 עד* n-1 *שבגוף הלולאה יש רק פעולות אלמנטאריות לכן גם לו יש זמן ריצה* וקיבלנו שזמן הריצה הכולל הוא *.*

**שאלה 2**

*נגדיר* OPT(i) *כגובה המקסימלי של בניין יציב ש-*i *הוא בסיס המגדל (התיבה התחתונה ביותר).*

*אם אין שום תיבה שאפשר להניח על* i *אז ברור כי* OPT(i)=h(i) *אחרת* OPT(i)=h(i)+MAX{OPT(j)} *כאשר* j *היא תיבה שניתן להניח על* i *(כלומר* w(j)<w(i) וגם l(j)<l(i)*)*.

*בשביל למצוא מגדל יציב בגובה מרבי נייצר מערך* OPT *שיכיל עבור כל תיבה* i *את* OPT(i) *כאשר נעזר במערך נוסף* next\_box *שיכיל עבור כל תיבה את התיבה הבאה שתונח עליה (בבניית בניין יציב בגובה מקסימלי) או* NULL *אם אין כזאת ובאמצעותו נוכל לשחזר את הבניין.*

*נמיין את המערך לפי רוחב ונרוץ מ1 עד* n *ואז כל פעם שנבדוק את תיבה* i *נדע בוודאות שה*OPT *של התיבה הבאה שמונחת במגדל שנסמנה ב*j *כבר מוגדר* *כי בהכרח* w(j)<w(i) לכן j<i.

*האלגוריתם*

*(1) ניצור מערך OPT בגודל n ומערך* next *\_box בגודל n*

*(2) נמיין את התיבות לפי הרוחב שלהן*

*(3) לכל* i *מ 1 עד* n

*(3.1) ,*

*(3.2) לכל j מ1 עד* i*-*1

*(3.2.1) אם* l(j)<l(i) *וגם*  *אז ,*

*(3.3)*

*(4) ונסמן את ה*i *שעבורו מתקבל הערך המקסימלי ב*

*(5)*

*(6) כל עוד*

*(6.1)*

*(6.2)*

*(7) החזרת את הרשימה*

*נכונות*

*חישוב ערכי* OPT *- הצעד הרקורסיבי של חישוב תיבה משתמש בתוצאות החישוב של תיבות אחרות המקיימות בלבד (הרי מיינו את המערך וחייב להתקיים* w(j)<w(i)*). לכן, כאשר אנו מחשבים את ערכי* OPT *מ ועד , לא נתקל באף שלב בניסיון קריאה לערך ב*OPT *שעוד לא איתחלנו אותו.*

*בגלל שאי אפשר לדעת מה היא התיבה שמהווה את הבסיס בבניין היציב המקסימלי אנחנו בודקים את כל האפשרויות כלומר כל איברי המערך* OPT.

*בסוף אנחנו משחזרים בעזרת המערך* next\_box*, מתחילים מהתיבה שעבורה יש את הערך המקסימלי ב*OPT *וממשיכים עד שמקבלים NULL.*

*זמן ריצה*

*המיון לוקח זמן ריצה של* , יצירת המערך OPT מתבצע עם לולאה שרצה פעמי ובתוכה עוד לולאה שבמקרה הגרוע רצה וסך הכל , מציאת האיבר המקסימלי בOPT רץ על מערך בגודל n לכן זמן הריצה הוא והשיחזור במקרה הגרוע עובר על כל התיבות לכן לוקח וסך הכל זמן הריצה של האלגוריתם הוא כנדרש.

**שאלה 3**

א. נבחר את הפולינומים , , .

נראה שעבור פולינומים אלה מתקיים:

נציב את הנקודות  *ונקבל בהתאמה את מה שייראה שהפולינומים שווים.*

*נציב באגף הימני של הפולינום ונקבל:*

*נציב באגף הימני של הפולינום ונקבל:*

לכל k כך שמתקיים נראה שמתקיים השיוויון לעיל: נציב באגף הימני של הפולינום:

על פי ההגדרה מתקיים: , ולכן מתקיים:

הוכחנו שעבור שלושת הפולינומים q s r מתקיים השיוויון כנדרש.

ב. האלגוריתם ייצר מטריצה M בוגדל כך ש יהיה פולינום האינטרפולציה של הנקודות  *ובסוף יחזיר את .*

תחילה נאתחל את M כך שלכל , (מתקיים מכיוון שזהו פולינום ממעלה 0 עובר בנקודה *) ולאח מכן לכל נמצא את לפי נוסחת הנסיגה שמצאנו בסעיף א'.*

*האלגוריתם*

*(1) צור מטריצה* M *בגודל*

*(2) לכל* i *מ1 עד* n

*(3) לכל* k *מ1 עד* n-1

*(3.1) לכל* i *מ 1 עד* n-k

*(3.1.1)* חשב את לפי הנוסחה שמצאנו בסעיף א'

*(4) החזר את*

*נכונות*

*בסיס נכונות האלגוריתם נובע מנכונות נוסחת הנסיגה, בשביל שתמיד ערכי המטריצה M שאנחנו ניגשים אליהם יהיו מוגדרים היא מחושבת בסדר לפי אלכסונים: בהתחלה אנחנו מחשבים את הערכים עבור האלכסון הראשי ולאחר מכן אנחנו עוברים על שאר האלכסונים בעזרת k, תחילה את האלכסון שמתחיל בונגמר ב ואז בכל פעם מחשבים את האלכסון שמתחיל בתא ומסתיים בתא עד שמגיעים לאלכסון שהוא פשוט התא .*

*זמן ריצה*

*הכנסת האיברים לאלכסון הראשי לוקחת זמן ריצה (יש בדיוק* n *איברים באלכסון זה).*

*המעבר על האלכסונים המשניים מתבצע עם שני לולאות שכל אחת רצה במקרה הגרוע פעמים לכן סך הכל זמן הריצה הוא*  וזהו זמן הריצה הכולל של האלגוריתם.

ג. נתון הפולינום . נציב בו את הערכים הנתונים ונקבל:

*נריץ את האלגוריתם מסעיף ב ונציג את ערכי M בטבלה ונראה כיצד הפולינום* p(x) *מתקבל ב-M[1,n]:*

*במקרה זה לכן גודל המטריצה הוא .*

הכנסת ערכים באלכסון הראשי:

|  |  |
| --- | --- |
| *i* |  |
| *1* |  |
| *2* |  |
| *3* |  |
| *4* |  |
| *5* |  |

שאר האלכסונים:

|  |  |  |  |
| --- | --- | --- | --- |
| *k* | *i* | *j=i+k* |  |
| 1 | *1* | *2* |  |
| *1* | *2* | *3* |  |
| *1* | *3* | *4* |  |
| *1* | *4* | *5* |  |
| 2 | *1* | *3* |  |
| 2 | *2* | *4* |  |
| 2 | *3* | *5* |  |
| 3 | *1* | *4* |  |
| 3 | *2* | *5* |  |
| 4 | *1* | *5* |  |

ואכן קיבלנו את הפולינום הנתון כנדרש.

**שאלה 4**

א. *האלגוריתם מחשב את המסלול בעל המשקל המינימלי מהקודקוד הנתון אל כל קודקוד*  כאשר התשובות מאחסונות במערך A[v], אם לא קיים מסלול אז הערך יהיה .

טענה: בסוף האיטרציה ה- i של הלולאה החיצונית *המערך A יכיל את המשקל המינימלי מבין כל המסלולים מ-r ל-v* שאורכם הוא לכל היותר הi, נוכיח טענה זו באינדוקציה.

עבור i=0: הקודוק היחיד שאפשר להגיע אליו מr עם מסלול שאורכו לכל היותר 0 הוא r עצמו לכן כל האיברים בA הם חוץ מA[r] שערכו 0.

נניח שהטענה נכונה עבור i<n, ונראה שהיא נכונה עבור i=n:

יהי צומת ב-G,

1. אם לא קיים מסלול שארוכו לכל היותר i מr לv אז לפי טענת האינדוקציה לפני האיטרציה A[k]= לכל צומת k שאין מסלול באורך קטן מi מr אליה לכן באיטרציה הi לא נמצא קשת (k,v) שמקיימת A[v]>A[k]+c(e)
2. אם המשקל המינימלי שייך למסלול שערכו קטן מi לפי הנחת האינדוקציה לפני האיטרציה ה-i A[v] יכיל את משקל מסלול זה ובאיטרציה ה-i בהכרח הוא לא ידרס כי אחרת זה אומר שקיים מסלול באורך i-1 לצומת שנסמנה u כך שמסלול זה בתוספת הקשת (u,v) הוא בעל משקל קטן יותר מהמסלול המינימלי שאורכו קטן מi וזו סתירה.
3. אם המשקל המינימלי שייך למסלול שערכו בדיוק i שנסמנו כך אז המסלול הוא מסלול מינימלי בארוך i-1 לכן לפי הנחת האינדוקציה A[w] מכיל את משקלו ובהכרח יתקיים התנאי A[v]>A[w]+c(w,v)

מטענת האינדוקציה אפשר להבין שאם בהרצה מסויימת לא בוצעו עדכונים זה אומר שכל המסלולים בעלי המשקל המינימלי כבר נמצאו כי רישא של מסלול בעל משקל מינימלי היא גם בעלת משקל מינימלי ולכן בסוף האלגוריתם A יכיל את המסלולים בעלי המשקל המינימלי לכל הקודקודים.

ב. המספר המירבי של איטרציות בלולאה החיצונית הוא B(n)=n וזה קורה כאשר קיים קודקוד f שהמסלול מr אליו עובר בכל שאר הקודקודים ואז יש n-1 איטרציות שבהן מתעדכנת צומת אחת בדיוק ובאיטרציה הn (האחרונה) לא יתבצע שינויים והאלגוריתם יסתיים.

דוגמא לסדרת גרפים שעליהם מתבצעות בדיוק B(n) איטרציות היא סדרה שבה אל כל צומת (חוץ מ-r) מגיעה קשת אחת בדיוק ומכל צומת (חוץ מהצומת האחרונה) יוצאת קשת אחת בדיוק:

………..

ג. בסדרת גרפים  *שבה מקודקוד* r *יוצאת קשת לכל קודקוד אחר (סך הכל* n-1 *קשתות) ואין עוד קשתות לא קיימים מסלולים שאורכם גדול מ1 לכן הלולאה החיצונית תרוץ פעמיים בדיוק, באיטרציה הראשונה יחושבו כלהמסלולים שאורכם 1 ובאיטרציה השנייה לא יבוצעו עדכונים והלולאה תעצר.*

מתקיים .

………..