**בעיית תת-מטריצה עם סכום מקסימלי**

**Max-Sum Submatrix of a Matrix**

* חיפוש שלם,
* בעזרת מערך עזר,
* בעזרת מטריצת עזר,
* אלגורתם יעיל

תחום השימוש: עיבוד תמונה, זיהוי תבניות, ניתוח רצף ביולוגי, כריית מידע או כריית נתונים (data mining)

דוגמה: עבור מטריצה נתונה:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 4 | 3 | 2 | 1 |  |
| 0 | -7 | -2 | 0 | 1 |
| 2 | -6 | 2 | 9 | 2 |
| 1 | -4 | 1 | -4 | 3 |
| -2 | 0 | 8 | -1 | 4 |

תת-מטריצה בעלת סכום גדול ביותר (15) היא

|  |  |  |
| --- | --- | --- |
| 2 | 1 |  |
| 2 | 9 | 2 |
| 1 | -4 | 3 |
| 8 | -1 | 4 |

**חיפוש שלם**

נתונה מטריצה עם n שורות ו-m עמודות. צריך לעבור על כל תתי מטריצות מלבניות. ובכל תת-מטריצה לסכום את התאים שלה ולבדוק האם זה סכום מקסימלי, אם כן, לעדכן את המשתנה שמשקף את הסכום המקסימלי ובהתאם לעדכן את המשתנים שמהווים את האינדקס ההתחלתי, את האינדקס הסופי וכ"ד לפי הנדרש.

איך לייצר את כל תתי מטריצות מלבניות?

מלבן (תת-מטריצה מלבנית) מגדירים ע"י שתי נקודות נגדיות כאשר קואורדינטות של פינה שמאלית עליונה הן(i,j) וקואורדינטות של פינה ימנית תחתונה הן (p,q).

(i,j)

שקול ל-

(p,q)

כי

max= mat[0][0]

for i ←1 to n // rows n

for j←1 to m// columns m

for p ←i to n

for q←j to m

for x ←i to p

for y←j to q

sum = sum + mat[x][y]

max = Math.max(max, sum);

….

**חיפוש שלם בעזרת מערך עזר**

נתבונן בתת-מטריצה מלבנית בפס (פס שנוצר ע"י כמה שורות או ע"י כמה עמודות). בעצם, מבצעים דחיסת הנתונים. נראה מקרה, כאשר נייצר תתי-מטריצה מלבניות בפס לפי כל האפשרויות של תתי השורות ועבור כל האפשרות של הפס, נסכום איברים בכל עמודה, נכניס את הסכום לתוך המערך ועל המערך נפעיל best וכ"ד:

לדוגמה, נגזור במטריצה הנתונה את הפס הבא:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 4 | 3 | 2 | 1 |  |
| 2 | -6 | 2 | 9 | 2 |
| 1 | -4 | 1 | -4 | 3 |

תוצאת הסיכום של עמודות ושימוש ב-Best יהי מערך עם סכום הגדול ביותר של 8:

|  |  |  |  |
| --- | --- | --- | --- |
| 4 | 3 | 2 | 1 |
| 3 | -10 | 3 | 5 |

אז תת-מטריצה מלבנית עם סכום 8 מוגדרת ע"י אינדקסים בשורות 2,3 ואינדקסים בעמודות 1,2:

|  |  |  |
| --- | --- | --- |
| 2 | 1 |  |
| 2 | 9 | 2 |
| 1 | -4 | 3 |

שקול ל-

כי

יצירת כל הפסים (לפי שורות) היא , לסכום איברי העמודות בפס ולהפעיל best

כתיבה מדויקת:

מטרה שלנו לקבל כמות פסים מינימלית כי מציאת כל תת-מטריצה מלבנית בפס לוקחת זמן משמעותי. כלומר, אם , אז כמות הפסים נקבע לפי לכן, מציאת כל תתי-מטריצות מלבניות בכל פס ולסכום תאי תת המטריצה ייקח זמן ובכל תת-מטריצה הפעלת ה- Bestיהיה לפי כך, הסוביכיות הכללית תהיה .

max= mat[0][0]

for i ←1 to n // rows n

for j←i to n // rows n

arr[]={0}

for p ←i to j

for q←1 to m

arr[q] = arr[q]+mat[p,q]

maxBest = Best(arr)

max = Math.max(max, maxBest);

….

**חיפוש שלם בעזרת מטריצת עזר**

קודם כל נראה דוגמה של שימוש במערך עזר במערך רגיל.

נתון מערך

a[]={a[0],a[1],…,a[n-1]}

נגדיר מערך עזר

h[]={h[0],h[1],…,h[n-1]}

כך ש-

h[0]=a[0],

h[1]=h[0]+a[1],

… ,

h[i]=h[i-1]+a[i],

… ,

h[n-1]=h[n-2]+a[n-1]

או במילים אחרות

h[i]=a[0]+a[1]+ … +a[i]

לכן סכום של איברי קטע [p,q] במערך a שווה להפרש של שני איברים במערך h **:**

a[p]+a[p+1]+ … +a[q-1]+a[q] = h[q] – h[p]

**תכנון דינאמי:** באופן דומה, עבור מטריצה נתונה A

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 4 | 3 | 2 | 1 |  |
| 0 | -7 | -2 | 0 | 1 |
| 2 | -6 | 2 | 9 | 2 |
| 1 | -4 | 1 | -4 | 3 |
| -2 | 0 | 8 | -1 | 4 |

נגדיר מטריצת עזר. מטריצת העזר היא סכום של המטריצה שנגמרת בתא ספציפי ומתחילה בתא (1,1)

1. בניית מטריצת העזר H:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 4 | 3 | 2 | 1 |  |
| 9- | 9- | -2 | 0 | 1 |
| 2- | -4 | 9 | 9 | 2 |
| 8- | 11- | 6 | 5 | 3 |
| -4 | -4 | 13 | 4 | 4 |

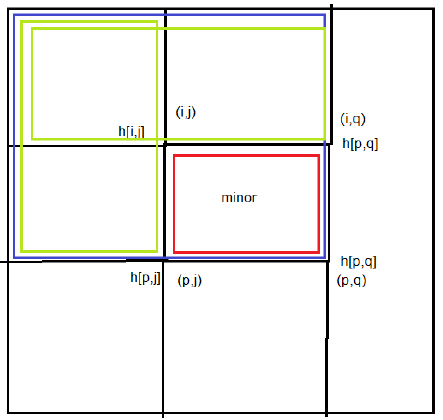
H[n,m]

H[0,0] = mat[0][0]

H[0,j]=H[0,j-1]+A[0,j]

H[i,0]=H[i-1,0]+A[i,0]

H[i,j]=H[i,j-1]+ H[i-1,j]- H[i-1,j-1]+A[i,j]



**int[][] getHelpMatrix(mat[][])**

h[m,n]

h[0,0] = m[0,0]

for i=1 to m-1

h[i][0] = h[i-1][0] + m[i][0] // i=0,1,…,m-1

for j=1 to n-1

h[0][j] = h[0][j-1] + m[0][j] // j=0,1,…,n-1

for i=1 to m-1

for j=1 to n-1

h[i,j] = m[i,j] + h[i-1,j] + h[i,j-1] - h[i-1,j-1]

return h

**end-getHelpMatrix**

1. נחשב סכומי תתי מטריצה:

**int sum\_ij\_pq(help[][], i, j, p, q)**

if (i==0 and j==0) sum = help [p,q]

else if (i==0 and j>0) sum = help [p,q] - help [p,j-1]

else if (i>0 and j==0) sum = help [p,q] - help [i-1,q]

else sum = help[p,q] - help[p,j-1] - help[i-1,q] + help[i-1,j-1]

return sum

**end\_sum\_ij\_pq**

שקול ל-

כי

max= mat[0][0]

for i ←1 to n

for j←i to n

for p ←i to j

for q←1 to m

sum=H[p,q]- H[p,j-1]- H[i-1,q]+ H[i-1,j-1]

max = Math.max(max, sum);

….

**אלגוריתם יעיל super best -**

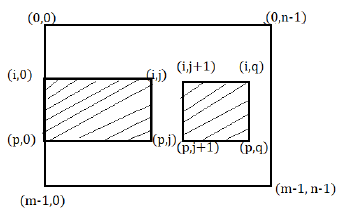
שילוב שני האלגוריתמים הקודמים:

נשתמש באותה מטריצת עזר כמו בסעיף הקודם לחישוב סכום איברי של תת-מטריצות.

נגדיר פס השורות בין שורה i לשורה p ונחשב את סכומים של תתי-מטריצות הנמצאות באותו פס

השורות בין עמודות 0 ל .j כמו ב **, best** ברגע שנקבל סכום שלילי מאפסים את הסכום ועוברים

לסכום מחדש מעמודה j+1 . בכול חישוב הסכום מעדכנים את הסכום המקסימאלי.



פסדו-קוד של אלגוריתם:

int **matrixSuperBestON3**(int[][] mat){ //O(N^3)

help[][] = getHelpMatrix(mat)

max = help[0][0], t=0

**for** i=0 to m-1

**for** p=i to m - 1

j = 0

**for** q=0 to n-1

t = *sum\_ij\_pq*(help, i, j, p, q) //O(1)

**if** (t < 0) j = q + 1

**else if** (t>max) max = t

**end-for**

**end-for**

**end-for**

**return** max;

**end-matrixSuperBestON3**

max= mat[0][0]

for i ←1 to m // columns m

arr[]={0}

for j←i to m // columns m

for q ←1 to n // rows n

arr[q]=arr[q] + mat[q,j]

maxBest = Best(arr)

max = Math.max(max, maxBest);

….

שקול ל-

כי

כתיבה מדויקת:

מטרה שלנו לקבל כמות פסים מינימלית כי מציאת כל תת-מטריצה מלבנית בפס לוקחת זמן משמעותי. כלומר, אם , אז כמות הפסים נקבע לפי לכן, מציאת כל תתי-מטריצות מלבניות בכל פס יקח זמן ובכל תת-מטריצה הפעלת ה- Bestיהיה לפי כך, הסיבוכיות הכללית תהיה .

**דוגמה:**

עבור מטריצה נתונה A

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 4 | 3 | 2 | 1 |  |
| 0 | -7 | -2 | 0 | 1 |
| 2 | -6 | 2 | 9 | 2 |
| 1 | -4 | 1 | -4 | 3 |
| -2 | 0 | 8 | -1 | 4 |

נראה את האלגוריתם שרץ על עמודות וסוכם שורות:

i=1, j=1 (עמודות)

|  |
| --- |
| 0 |
| 9 |
| -4 |
| -1 |

startR=1, startC=1,

After Best: endR=2, endC=1

Sum=9

i=1, j=2(עמודות)

|  |
| --- |
| -2 |
| 11 |
| -3 |
| 7 |

startR=2, startC=1,

After Best: endR=2, endC=2

Sum=11

i=1, j=3 (עמודות)

|  |
| --- |
| -9 |
| 5 |
| -7 |
| 7 |

startR=2, startC=1, endR=2, endC=2 – לא משנים כי אין שיפור בסכום

Sum=5

i=1, j=4 (עמודות)

|  |
| --- |
| -9 |
| 7 |
| -6 |
| 5 |

startR=2, startC=1, endR=2, endC=2 – לא משנים כי אין שיפור בסכום

Sum=7

i=2, j=1 (עמודות)

|  |
| --- |
| 9 |
| -4 |
| -1 |

startR=2, startC=1, endR=2, endC=1

Sum=4

i=2, j=2 (עמודות)

|  |
| --- |
| 11 |
| -3 |
| 7 |

startR=2, startC=1, endR=4, endC=2

Sum=15

.....

…..

**משימות:**

ליישם את האלגוריתם של בעית תת-מטריצה עם סכום מקסימלי לפי 4 שיטות:

1. חיפוש שלם,
2. בעזרת מערך עזר,
3. בעזרת מטריצת עזר,
4. אלגורתם יעיל