المحاضرة الثانية

حساب تعقيدات الخزن

يتم حساب (تخمين) تعقيدات الخزن من خلال حساب عدد خلايا الذاكرة التي نحتاجها لانجاز الخطوات الاحتسابية المطلوبة لحل مثال مسألة مع استبعاد الخزن المخصص للمدخلات. وبصورة عامة يتم الاشارة الى عدد الخلايا بدون تعريف دقيق لها حيث يمكن تصور الخلية بانها مقدار من الذاكرة كافي لخزن عدد واحد فقط.

حساب تعقيدات الوقت

توجد طريقتان لحساب (تخمين) الوقت:-

١) عد العمليات (Operation counts) :- ان احدى الطرق لتخمين الوقت هو باختيار عملية واحدة او اكثر (مثلا الجمع، الضرب، المقارنة، والاحلال) وحساب الوقت نسبة الى هذه العمليات من خلال عدها، حيث يكون هذا العدد دالة في خصائص المثال. ان نجاح هذه الطريقة يعتمد على القدرة لتعريف العمليات التي تساهم اكثر في تعقيدات الوقت.

٢) عد الخطوات (Step counts) :- في طريقة عد العمليات يتم التركيز على خطوات معينة في الخوارزمية وتجاهل بقية الخطوات. اما في طريقة عد الخطوات فيتم تحديد العدد الكلي من الخطوات المنفذة من خلال الخوارزمية حيث يكون هذا العدد دالة في خصائص المثال. وبالرغم من امتلاك مثال معين العديد من الخصائص(مثلا عدد المدخلات، عدد المخرجات، قيمة المدخلات والمخرجات) الا ان عدد الخطوات يتم احتسابها كدالة لمجموعة جزئية منها. حيث يتم في العادة اختيار الخصائص التي نرغب بها.

تعريف/ خطوة برنامج هي اي خطوة احتسابية وقتها مستقل عن خصائص المثال. فمثلا قد تعتبر ١٠ عمليات جمع كخطوة واحدة و ١٠٠ عملية جمع كخطوة واحدة ايضا في حين n من عمليات الجمع (n هي خصائص المثال) لا تعد كذلك.

امثلة على حساب تعقيدات الوقت والخزن

مثال١/ ايجاد ناتج العلاقة $\sum_{i=1}^{n} a_i$

```
Algorithm sum(A,n):
Input: a positive integer n and an array A indexed from 1 to n.
Output: S, the sum of the numbers in A.
1. S ←—— 0
2. for j ←—— 1  to  n
3.    S ←—— S + A[j]
4. end for
5. return S
```

يتصف مثال المسألة بـ n

تعقيدات الخزن:- تتطلب هذه الخوارزمية خليتان للخزن هما S و j وهو خزن ثابت لا يعتمد على خصائص المثال.

$$S_{sum}(n)= 0$$

تعقيدات الوقت باستخدام عد العمليات:- يتم اختيار الجمع بين عناصر المصفوفة A كعملية يتم قياس الوقت نسبة لها.

$$T_{sum}(n)= n$$

<u>تعقيدات الوقت باستخدام عد الخطوات:-</u>

$$\left.\begin{array}{l} 1.\ \ldots\ldots\ 1 \\ 2.\ \ldots\ldots\ n+1 \\ 3.\ \ldots\ldots\ n \\ 4.\ \ldots\ldots\ 0 \\ 5.\ \ldots\ldots\ 1 \end{array}\right\} \Longrightarrow 1+n+1+n+0+1=2n+3$$

$$T_{sum}(n)=2n+3$$

<u>مثال٢/</u> ايجاد مجموع عناصر مصفوفتين $C(m\times n) = A(m\times n) + B(m\times n)$

```
Algorithm add (A, B, m, n):
Input: a positive integers m, n and two-dimensional arrays of numbers
       A and B each of which has its rows indexed from 1 to m and
       columns from 1 to n.
Output: a two-dimensional array of numbers C, containing addition of
        A and B.
1.   for i ←—— 1  to  m
2.     for j ←—— 1  to  n
3.        C[i, j] ←—— A[i, j]+B[i, j]
4.     end for
5.   end for
6.   return array  C
```

يتصف مثال المسألة بـ m و n

<u>تعقيدات الخزن:-</u>

$$S_{add}(m,n)=mn+2$$

<u>تعقيدات الوقت باستخدام عد العمليات:-</u> يتم اختيار الجمع بين عناصر المصفوفتين A و B كعملية يتم قياس الوقت نسبة لها.

$$T_{add}(m,n)=mn$$

<u>تعقيدات الوقت باستخدام عد الخطوات:-</u>

$$\begin{array}{l} 1.\ \ldots\ldots\ m+1 \\ \left.\begin{array}{l} 2.\ \ldots\ldots \\ 3.\ \ldots\ldots \end{array}\right\} ms \Longrightarrow m(n+1+n) \Longrightarrow m(2n+1) \\ 6.\ \ldots\ldots\ mn \end{array}$$

$$T_{add}(m,n)=3mn+2m+1$$

<u>ملاحظة/</u> هذه العلاقة تكون مقبولة اذاكان $m\leq n$ اما عندما يكون $m>n$ فانه يفضل مبادلة تعليمتي for لان ذلك يخفض تعقيدات الوقت لتصبح

$$T_{add}(m,n)=3mn+2n+1$$

<u>مثال٣/</u> اعداد فيبوناتشي (Fibonacci numbers).

متتابعة فيبوناتشي للاعداد تبدأ كالاتي:

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, ….

حيث ان كل حد جديد يتم الحصول عليه من خلال جمع الحدين السابقين، فاذا كان F_0 يمثل الحد الاول في المتتابعة فان $F_0=0$ و $F_1=1$ و بصورة عامة:

$$F_n = F_{n-1} + F_{n-2} \qquad , \ n\geq 2$$

```
Algorithm fibonacci (n):
Input: a nonnegative integer n.
Output: fib, the nth term of the fibonacci sequence.
1.  if n ≤ 1 then
2.    fib ←—— n
3.  else
4.      fnm1 ←—— 0
5.      fnm2 ←—— 1
6.      for i ←—— 2  to  n
7.         fib ←—— fnm1 + fnm2
8.         fnm1 ←—— fnm2
9.         fnm2 ←—— fib
10.     end for
11. end if
12. return fib
```

يتصف مثال المسألة بـ n

<u>تعقيدات الخزن:-</u> تتطلب هذه الخوارزمية اربع خلايا خزنية لخزن قيم fib و fnm1 و fnm2 و i وهو خزن ثابت لا يعتمد على خصائص المثال.

$$S_{fibonacci}(n)= 0$$

<u>تعقيدات الوقت باستخدام عد العمليات:-</u> يتم اختيار الاحلال بين قيم المتتابعة كعملية يتم قياس الوقت نسبة لها.

7. …… n-1
8. …… n-1
9. …… n-1

$$T_{fibonacci}(n)= 3(n-1) = 3n-3$$

<u>تعقيدات الوقت باستخدام عد الخطوات:-</u> يجب اعتبار حالتين في هذه الطريقة:

الحالة الاولى :- عندما n=0 او n=1 فان عدد الخطوات يساوي 3.
الحالة الثانية :- عندما n>1 فان عدد الخطوات يساوي 4n+1.

$$T_{fibonacci}(n) = \begin{cases} 3 & , \ if \quad n \in \{0,1\} \\ 4n+1 & , \ if \quad n > 1 \end{cases}$$

<u>ملاحظة/</u> في العديد من المسائل توجد موازنة ما بين كمية الوقت والخزن المخصص للخوارزمية. فالخزن الاكبر المخصص يؤدي بالنتيجة الى زيادة في سرعة التنفيذ و العكس بالعكس.

<u>مثال٤ /</u> اعادة صياغة الخوارزمية Fibonacci.

```
comment: f[0..n] is an auxiliary array.
f[0]←—— 0;
if n > 0 then
  f[1]←—— 1
  for i←—— 2  to  n
    f[i]←——  f[i-1] + f[i-2];
  end for
end if
return f[n];
```

<u>تعقيدات الخزن:-</u>

$$S_{fibonacci}(n) = n+2$$

<u>تعقيدات الوقت باستخدام عد الخطوات:-</u>

$$T_{fibonacci}(n) = \begin{cases} 3 & , \ if \quad n = 0 \\ 2n+3 & , \ if \quad n \geq 1 \end{cases}$$