بسم الله الرحمن الرحيم

جامعة السودان المفتوحة

برنامج علوم الحاسوب و تقانة المعلومات

تحليل وتصميم الخوارزميات

رمز المقرر: حسب 1028

تأليـــــف:

د. معاويـــة الفكـــي يحيى

مراجعـــــة: أ.د. عزالدين محمد عثمان
التصميم التعليمي: أ.منال محمد بشير التنقاري
التدقيق اللغــوي: أ. الخاتم عبد الرحمن أبو الحسن
التنضيد الطباعي: آسيا عبد القادر
التصميــم الفنــي: نـــــــادر محمّد علي الأمين

مقدمـــة المقـــرر

عزيزي الدارس

مرحباً بك إلى هذا المقرر الذي يعتبر من المقررات الأساسية للدارس في مجـــال الحاسوب. لعلك عزيزي الدارس يرد في ذهنك سؤال لم تجد له إجابة بعـــد فـــي مجـــال الحاسوب وهو كيف يقوم الحاسوب بحل المسائل والمشكلات المعقدة التي تدخل إليـــه فـــي شكل مدخلات ليعطينا النتائج في شكل مخرجات بعض إجراء تنفيذ المطلوب منـــه بدقـــة كاملة.

إذا أردت حل مسألة رياضية ما، هل تفضل أن تعطي النتائج مباشرة أم تضع الحل في شكل خطوات لتسهل الوصول إلى الحل الصحيح وبصورة دقيقة، وهذا بالضبط ما نفعله حين نكتب خوارزمية للمسألة وهو أننا نقسم المسألة إلى مسائل أصغر وسهلة الحل ليسهل إيجاد الإجابة الصحيحة و بصورة أسرع.

إن الخوارزمية هي الخطوات اللازمة لحل مسألة ما، وسوف يتم في هذا المقـــرر كتابة الخوارزميات بلغة شبيهة بلغة باسكال، فمن المهم أن تكـــون قـــادراً علـــى تحليـــل الخوارزميات مع بعضها البعض لاختيار الأفضل والأنجع، هـــذا وعنـــد المقارنـــة بـــين الخوارزميات فإن المقاييس التي يتم استخدامها في المقارنة تشتمل على:

1. مقدار الوقت الذي يأخذه الحاسوب لتنفيذها.
2. مساحة ذاكرة الحاسوب التي تحتاج إليها الخوارزمية.
3. وضوح الخوارزمية وبساطتها.

وقد بين كثير من العلماء أن أهم مقياس من المقاييس الثلاثة هو عامل الوقت. فتحويل الخوارزمية إلي برنامج وتنفيذ هذا البرنامج لحساب الوقت الحقيقي اللازم لتنفيذ الخوارزمية لا يعطي فكرة جيدة عن جودة الخوارزمية أو عدمها، وذلك لأن وقت التنفيذ يعتمـــد علـــى عوامل مختلفة وهي:

1. سرعة الحاسوب إذ أن الوقت الحقيقي لتنفيذ أي خوارزمية يعتمـــد علـــى سرعة معالج الحاسوب المستخدم لتنفيذ الخوارزمية.

2. كمية البيانات، إذ أن الوقت الحقيقي يعتمد أيضاً عل كمية البيانات المراد معالجتها.
3. عوامل أخرى لها علاقة بطريقة التنفيذ، ومنها لغة البرمجة المختارة والمبرمج الذي كتب البرنامج.

ونسبة لهذه العوامل كان لابد من تطوير طريقة رياضية أكثر تجرداً لتحليل وقت تنفيذ الخوارزمية، وأيضاً هنالك سببٌ مقنعٌ لدراسة طرق التصميم المختلفة للتعرف على كيفية اختيار أفضلها عند حل مسألة معينة.

يحتوي هذا المقرر على سبعة وحدات هي

الوحدة الأولى وهي الخلفية الرياضية التي تشتمل على الخلفية الرياضية المطلوبة في تحليل وتصميم الخوارزميات وتتألف من اللوغريثمات، المتسلسلات وخوارزميات الاستدعاء الذاتي.

والوحدة الثانية وهي مبادئ تحليل الخوارزميات التي

تشتمل على الخلفية الأساسية في تحليل الخوارزميات والتي تجد فيها المعنى العام لتحليل الخوارزميات والتعرف على معدلات النمو المختلفة للخوارزميات.

ثم الوحدة الثالثة وهي التحليل الزمني للخوارزميات وتشتمل على كيفية تحليل الخوارزميات متضمنة التعرف على نموذج التحليل الزمني للخوارزميات حيث يتم دراسة الخوارزميات المختلفة والتعرف على مدى كفاءتها من حيث وقت التنفيذ.

وكذلك في الوحدة الرابعة نجد تحليل خوارزميات الترتيب وفيها وفيها الخلفية الأساسية عن خوارزميات الترتيب المختلفة والتعرف على كيفية المقارنة بينها.

أما الوحدة الخامسة فسوف تجد فيها تحليل خوارزميات البحث وتشتمل على الخلفية الأساسية عن خوارزميات البحث المختلفة والتعرف على كيفية المقارنة بينها.

وفي الوحدة السادسة نقدم تصميم الخوارزميات والذي يشتمل على الخلفية الأساسية عن طرق تصميم الخوارزميات والتي تتضمن طريقة الخوارزمية الجشعة وخوارزمية فرق تسد وخوارزمية البرمجة الفعالة.

وأخيراً في الوحدة السابعة سوف تجد شيئاً عن محدودية قوة الخوارزميات حيث نقوم بدراسة نظرية الحد الأدنى للخوارزميات والتعرف على شجرة القرارات والتي تستخدم في قياس أداء الخوارزميات وخاصة خوارزميات الترتيب والبحث، كما سوف نتعرف على كيفية تصنيف المسائل حسب درجة تعقيدها إلى مسائل محددة الحل بواسطة خوارزميات محددة بالزمن ومسائل غير محددة الحل ولا يتم حلها إلا بواسطة خوارزميات غير محددة الزمن.

الساعات المعتمدة:

3 ساعة معتمدة (2 نظري) + (2 عملي).

المتطلبات السابقة:

1) رياضيات متقطعة
2) أساسيات برمجة
3) هياكل بيانات

الأهداف العامة للمقرر

عزيزي الدارس عندك انتهاءك من دراسة هذا المقرر يتوقع منك أن تكون قادراً علي أن:

- توضح مبادئ تحليل الخوارزميات.
- تحلل الخوارزميات زمنيا.
- تقارن بين الخوارزميات المختلفة.
- تحلل خوارزميات البحث والترتيب.
- تعرف الطرق المختلفة في تصميم الخوارزميات.
- تصمم وتحلل الخوارزميات الجشعةو الخوارزميات الفعالة و خوارزميات فرق تسد.
- تصنف المسائل المختلفة إلى مسائل محددة الحل أو غير محددة الحل بصورة تامة.

محتويات المقرر


الوحدة الأولى
الخلفية الرياضية
Mathematical Background

محتويـــات الوحـــدة

المقدمة

تمهيد

تهدف هذه الوحدة إلي إعطاء لمحة سريعة عن الخلفية الرياضية المطلوبة في هذا المقرر، وتمثل هذه الخلفية أساساً هاماً لفهم مبادئ تحليل وتصميم الخوارزميات.

تتكون هذه الوحدة من ثلاثة أقسام رئيسة. القسم الأول منها يبحث في اللوغريثمات؛ أنواعها وكيفية التعامل معها، أما القسم الثاني فيشرح المتسلسلات، ثم يأتي القسم الثالث ليوضح كيفية التعامل مع خوارزميات الاستدعاء الذاتي.

هذا وقد زودناك خلال هذه الوحدة بمجموعة من الأنشطة والتدريبات وأسئلة التقويم الذاتي نأمل أن تمرن نفسك على حل التدريبات الواردة في هذه الوحدة لتصبح لديك المهارة الكافية لإتقانها. كما نأمل أن تجدها وحدة مفيدة، وأن تساهم معنا في نقدها وتطويرها، وفقك الله.

أهداف الوحدة

عزيزي الدارس، بعد فراغك من دراسة هذه الوحدة أتوقع أن تكون قادراً على أن:

- ✓ **تُعرّف** اللوغريثمات.
- ✓ **تحل** مسائل اللوغريثمات باستخدام النظريات الموضوعة.
- ✓ **تعرف** المتسلسلة.
- ✓ **تحل** مسائل المتسلسلات.
- ✓ **تعرف** الاستدعاء الذاتي.
- ✓ **تنفذ** خوارزميات الاستدعاء الذاتي.

1. اللوغريثمات Algorithms

بما أن اللوغريثمات سوف تلعب دوراً هاماً في تحليل الخوارزميات فإننا سوف نتناول بعض التعريفات والنظريات حولها كما سوف نتعرف علي خواصها الأساسية.

تعريف (1)

إذا كان $x^a = b$ فإننا نعبر عن القوى a بواسطة اللوغريثم بالصورة $\log_x b = a$.

مثال (1)

$\text{Log}_2 8 = 3$ لأن $2^3 = 8$

سوف نستخدم الصيغة المختصرة log للتعبير عن $\log_2$ في هذا المقرر.

1.1 نظريات على اللوغريثمات

نظرية (1)

$$\text{Log}_a b = \log_b / \log_a$$

نظرية (2)

$$\text{Loga } b = \text{loga} + \text{logb}$$

مثال (2)

لاحظ أن $\log_4 16 = 2$

سوف نجد من النظرية أن:

$$\text{Log}_4 16 = \log 16 / \log 4 = 4/2 = 2$$

2.1 خواص اللوغريثمات

هذه بعض الخواص الأخرى للوغريثمات

1. $\text{Log}_b 1 = 0$
2. $\text{Log}_b b = 1$
3. $\text{Log}_b x^a = a \text{ Log}_b x$
4. $\text{Log } a/b = \text{Log } a - \text{Log } b$
5. $\text{Log } x < x$ لكل $x > 0$

أسئلة تقويم ذاتي

1/ أوجد

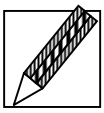
1/ $\log_2 64$		2/ $\log_2 1/64$	
3/ $\log_{64} ½$		4/ $\log_{100} 10^{-4/5}$	
5/ $\log_{32} 8$		6/ $\log_9 \sqrt{3}$	

2/ أثبت أن:

1/ $\log_p xy = \log_p x + \log_p y$

2/ $\log_p 1/x = -\log x$

تدريب (1)



1. أوجد قيمة اللوغريثمات أدناه

(i) $\text{Log}_3 27$ (ii) $\text{Log}_5 75$

2. أثبت الآتي:

(i) $\text{Log}_b a = a \log_b$ (ii) $\text{Log } x < x$ لكـل $x > 0$

2. المتسلسلات Series

عزيزي الدارس، نجد أن المتسلسلات تمثل جزءاً هاماً في تحليل الخوارزميات، حيث إننا نحتاج إلي جمع مجموعة قيم عند عملية تحليل الخوارزميات.

تعريف(2)

المتسلسلة: لتكن لدينا المتتالية $a_1,a_2,a_3,\ldots a_n$ فإن :

$$\sum_{r=1}^{n} ar = a_1 + a_2 + ... + a_n$$

تسمى المتسلسلة المرتبطة بهذه المتتالية. فإذا كانت المتتالية منتهية فإن المتسلسلة المرتبطة بها منتهية، وأما إذا كانت غير منتهية، تسمى المتسلسلة المرتبطة بها غير منتهية.

سوف نتناول بعض صيغ المتسلسلات المعروفة

1.2 المجموعة الأولى

$$\sum_{i=1}^{N} C*i = C* \sum_{i=1}^{N} i \qquad (i)$$

$$\sum_{i=c}^{N} i = \sum_{i=0}^{N-c} (i+c) \qquad (ii)$$

$$\sum_{i=c}^{N} i = \sum_{i=0}^{N} i - \sum_{i=0}^{c-1} i \qquad (iii)$$

$$\sum_{i=1}^{N} (A+B) = \sum_{i=1}^{N} A + \sum_{i=1}^{N} B \qquad (iv)$$

$$\sum_{i=0}^{N} (N-i) = \sum_{i=0}^{N} i \qquad (v)$$

2.2 المجموعة الثانية

$$\sum_{i=1}^{N} 1 = N \qquad (i)$$

$$\sum_{i=1}^{N} C = c * N \qquad (ii)$$

$$\sum_{i=1}^{N} i = N(N+1)/2 \approx N^2/2 \qquad (iii)$$

3.2 المجموعة الثالثة

$$\sum_{i=1}^{N} i^2 = N(N+1)\,(2N+1)/6 \approx N^3/3 \qquad (i)$$

$$\sum_{i=1}^{N} i^k \cong N^{k+1}/\left|k+1\right| \quad , \quad k \neq -1 \qquad (ii)$$

$$\sum_{i=0}^{N} 2^i = 2^{N+1} - 1 \qquad (iii)$$

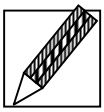
4.2 المجموعة الرابعة

$$\sum_{i=1}^{N} a^i = (a^{N+1} - 1)/(a-1) \leq 1/(1-a) \qquad (i)$$

$$\sum_{i=1} i2^i = (N-1)2^{N+1} + 2 \qquad (ii)$$

$$\sum_{i=1}^{N} 1/i = \log_e N \qquad (iii)$$

تدريب (2)



(1) من خلال تعرفك على المجموعات السابقة حل المتسلسلات أدناه:

$$\sum_{i=1}^{N} (i^2 - 2i) \qquad (i)$$

$$\sum_{i=7}^{N} i \qquad (ii)$$

$$\sum_{i=1}^{N} (3i + 7) \qquad (iii)$$

$$\sum_{i=5}^{N} (2i^2 + 1) \qquad (iv)$$

(2) أثبت الآتي:

$$\sum_{i=1}^{N} (2i - 1) = N^2 \qquad (i)$$

$$\sum_{i=1}^{N} i^3 = \left[\sum_{i=1}^{N} i\right]^2 \qquad (ii)$$

3. الاستدعاء الذاتي Recurrence

من الملاحظ أن هنالك العديد من المسائل التي يتطلب حلها استدعاءً ذاتياً وغالباً ما يكون تعريف هذه المسائل على نحو يتطلب تعريفاً ذاتياً.

مثال (3)

يعرف مضروب العدد الصحيح n تعريفاً ذاتياً على النحو الآتي:

$$n! = \begin{cases} 1, & if\ n = 0 \\ n*(n-1)! & if\ n > 0 \end{cases}$$

لاحظ أن هذا التعريف هو تعريف ذاتي أي بمعنى أن دالة المضروب تـم تعريفهـا باستخدام دالة المضروب نفسها، ويمكن صياغة التعريف السابق على النحو الآتي:

$$F_n = n* F_{n-1,}$$
$$F_0 = 1$$

مثال (4)

يمكن كتابة خوارزمية لإيجاد المضروب باستخدام الإستدعاء الذاتي على النحو الآتي:

```
Function fact (N = Integer): Integer
    Begin
        if (N = 0) then
                fact = 1
        else    fact = fact (N-1)*N
        end if
End
```

أي أن مضروب الصفر هو الرقم 1 ومضروب أي رقم صحيح أكبر من الصفر يساوي حاصل ضرب ذلك الرقم في مضروب الرقم الذي يقل عنه بواحد.

لاحظ مدى توافق الدالة fact (N) مع تعريف المضروب، مما يجعله سهل الكتابة والفهم، لاحظ أن الجملة: fact = fact (N - 1)* N

هي الجملة التي تستدعي الدالة استدعاءاً ذاتياً لحساب مضروب N-1 .

1.3 كيف يتم تنفيذ الاستدعاء الذاتي؟

بالرجوع للمثال السابق والذي تم فيه توضيح دالة المضروب fact (N) يمكننا فهم ما يجري لتنفيذ دالة الاستدعاء الذاتي، علينا أن نتخيل أن الحاسوب يعمل نسخة من هذه الدالة عند كل مرة يستدعي فيها fact (N) وينفذ تلك النسخة ثم يعود الحاسوب لإكمال تنفيذ النسخة المستدعية كأي برنامج فرعي آخر إذ تعود السيطرة عند انتهاء تنفيذ البرنامج الفرعي إلي البرنامج المستدعى له؛ فمثلاً للحصول على مضروب الرقم 2 فإن الحاسوب سوف يقوم باستدعاء نسخة من دالة المضروب لحساب مضروب الرقم 1، ولحساب الرقم ولحساب مضروب الرقم 1 فإن الحاسوب سوف يقوم باستدعاء نسخة أخرى من دالة المضروب لحساب مضروب الصفر.

لاحظ أن مسألة المضروب تتميز بالميزتين التاليتين:

أولاً: هنالك حالة لا يتطلب حساب المضروب لها استدعاءاً ذاتياً وهي عند N =0 وتسمي هذه الحالة بحالة الأساس.

ثانياً: في كل مرة تستدعي فيها دالة المضروب استدعاءاً ذاتياً يتم استدعاؤها على أبسط من الحالة السابقة لها، بمعنى أنها أقرب حالة إلي حالة الأساس. أي أننا في كل مرة نستدعي فيها دالة المضروب fact (N) نستدعيه على رقم أقل بواحد من الرقم الأصلي وذلك ضروري

لضمان توقف العملية والوصول إلي حالة الأساس. ولولا ذلك فقد تستمر عملية الاستدعاء الذاتي إلى ما لا نهاية.

لهذا من الضروري توفر هاتين الميزتين في أي مسألة كي يكون حلها بالاستدعاء الذاتي ممكناً.

نشاط

أدرس المتوالية الرقمية التالية

0,1,1,2,3,5,8,13,21,…

تدعى هذه المتوالية متوالية فايبونسي، حيث أن كل رقم بها باستثناء أول رقمين يساوي حاصل جمع الرقمين السابقين أي أن:

$F_0 = 0$, $F_1 = 1$

$F_n = F_{n-1} + f_{n-2}$

اكتب برنامجاً فرعياً Fib(n) يقوم بحساب قيمة الحد رقم n في متوالية فايبونسي مستخدماً الاستدعاء الذاتي.

أسئلة تقويم ذاتي

1) عرّف الاستدعاء الذاتي ومثّل له.

2) بالخطوات وضح كيف يتم الاستدعاء الذاتي لدالة معينة

الخلاصة

عزيزي الدارس في نهاية هذه الوحدة الاستهلالية أرجو أن تكون قد حققت الأهداف المرجوة منها، وللتأكد من ذلك أرجو إعادة استعراض الأهداف مرة أخرى ثم قراءة المفاهيم التالية التي تخص الوحدة.

- عرفنا اللوغريثمات بالصورة $\log_x b=a$ ثم طبقنا النظريات الموضوعة على اللوغريثمات لحلها وتبسيطها.
- عرفنا المتسلسلة وقلنا إنها تمثل جزءً هاماً في تحليل الخوارزميات ثم درسنا صيغ المتسلسلات المعروفة وطبقناها.
- عرفنا الاستدعاء الذاتي وقلنا إن التعريف الذاتي هو تعريف الدالة باستخدامها هي نفسها ومثلنا لخوارزمية كتبت باستخدام الاستدعاء الذاتي، واستخدمنا المثال في مناقشة كيف يتم تنفيذ الاستدعاء الذاتي.

وأخيراً فإننا نرى هذه الوحدة تمثل مدخلاً موفقاً لما بعدها من وحدات، أرجو أن تناقش مشرفك الأكاديمي في أي أمور تجدها غير واضحة في هذه الوحدة –وفقك الله في دراستها.

لمحة مسبقة عن الوحدة التالية

عزيزي الدارس،

في وحدتنا التالية نقوم بشرح المفاهيم الأساسية في تحليل الخوارزميات مثل تعقيد المساحة ومعدلات النمو. فلنبدأ معاً الخطوة الأولى في تحليل الخوارزميات وتصميمها.

إجابات التدريبات

تدريب (2)

(1)

(i)

$$\sum_{i=1}^{N}\left(i^2-2i\right)=\sum_{i=1}^{N}i^2-2\sum_{i=1}^{N}i=\frac{N(N+1)(2N+1)}{6}-\frac{2(N)(N+1)}{2}$$

$$=\frac{N(N+1)(2N-5)}{6}$$

$$N\to\infty \quad =$$

$$N^3\!\!\diagup_3$$

(ii)

$$\sum_{i=7}^{N}i=\sum_{i=1}^{N}i-\sum_{i=1}^{6}i=N(N+1)/2-\frac{6(6+1)}{2}=\frac{N(N+1)}{2}-21$$

$$=\left(\frac{N(N+1)-42}{2}\right)$$

$$N\to\infty \qquad \simeq N^2\!\!\diagup_2$$

(iii)

$$\sum_{i=1}^{N}(3i+7)=3\sum_{i=1}^{N}i+7\sum_{i=1}^{N}1=\frac{3N(N+1)}{2}+7N=\frac{N}{2}[3(N+1)+14]$$

$$N\to\infty \qquad 3N^2\!\!\diagup_2$$

(iv)

$$\sum_{i=5}^{N}\left(2i^2+1\right)=2\sum_{i=5}^{N}i^2+\sum_{i=5}^{N}1$$

$$=2\sum_{i=1}^{N}i^2-2\sum_{i=1}^{5}i^2+\sum_{i=1}^{N}1+\sum_{i=1}^{5}1=\frac{2N(N+1)(2N+1)}{6}-\frac{2(5)(6)(11)}{6}$$

$$+N-5$$

$$=\frac{N(N+1)(2N+1)}{3}-110+N-5=\frac{N(N+1)(2N+1)}{3}+N-15$$

$N\to\infty$ $\quad 2N^3/3$

(v)

$$\sum_{i=1}^{N}\ (2i-1)=N^2\sum_{i=1}^{N}i-\sum_{i=1}^{N}i=\frac{2N(N+1)}{2}-N=N^2$$

(2)

(i) مكررة

(ii)

$$\sum_{i=1}^{N}i^3=\left[\sum_{i=1}^{N}i\right]^2\left[\frac{N(N+1)}{2}\right]^2=\frac{N^2(N+1)^2}{4}$$

$N\to\infty$

$$\simeq\frac{N^4}{4}$$

مسرد المصطلحات

- **اللوغريثمات Logarithms**

طريقة رياضية لحل مسألة باستخدام أسلوب حسابي أبسط، بشكل متكرر حيث إن الفكرة الأساسية للوغريثمات هو تحويل عمليتي الضرب والقسمة المعقدتين إلى عمليتي جمع وطرح.

- **متسلسلات Series**

عملية جمع مجموعة من القيم المتتالية.

- **استدعاء ذاتي Recursive**

بمعنى أن الدالة تم تعريفها باستخدام الدالة نفسها.

- **دالة Function**
- **مضروبFacto**

المراجع

المراجع العربية

1) مجموعة مؤلفين، تركيب البيانات وتصميم الخوارزميات، منشورات جامعة القدس المفتوحة، 1998.

2) السماني عبد المطلب، هياكل البيانات، منشورات جامعة السودان المفتوحة، 2005.

المراجع الأجنبية

1) Weiss, M.A., Data Structures and Alogrithm Analysis, Benjamin pub., 1992.
2) McConnell, J.J., Analysis of Alogrithms: An Active Approach, Jones pub., 2001.
3) Lavitin, A., Introduction to the Design and Analysis of Alogrithms, Addisions ues by, 2003.
4) Ulman, J.D (etal), The Design and Analysis of Compurte Alogrithms, Addison Wesley, 1974.
5) Basese, S., Compurte Alogrithms : Introduction to Analysis and Design, Addison Wesley, 2000.
6) Sedgewick, R., An Introduction to the Analysis of Alogrithms, Addison Wesley 1996.


الوحدة الثانية
مبادئ تحليل الخوارزميات
Analysis Basics

محتويات الوحدة

المقدمة

تمهيد

عزيزي الدارس،

في هذه الوحدة سوف نتعرف على المبادئ الأساسية في تحليل الخوارزميات. والمقصود بتحليل الخوارزميات هو دراسة الخوارزميات المختلفة بغرض التعرف على مدى كفاءتها من حيث التنفيذ اللازم وذاكرة الحاسوب اللازمة، وقد حاولنا في هذه الوحدة توضيح المفاهيم الرئيسة مستخدمين العديد من الأمثلة والتدريبات المناسبة.

تتكون هذه الوحدة من قسمين؛ في القسم الأول سوف نتعرف على المعنى العام لتحليل الخوارزميات والهدف منه، أما في القسم الثاني فسوف ندرس معدلات النمو المختلفة للدوال والاصطلاحات الرياضية حولها هذا وقد زودنا الوحدة بمجموعة من الأنشطة والتدريبات وأسئلة التقويم الذاتي نرجو أن تستفيد منها الفائدة الكاملة كما نرجو أن تشاركنا نقد هذه الوحدة وتقويمها.

أهداف الوحدة

عزيزي الدارس،

بعد دراسة هذه الوحدة وتنفيذ تدريباتها يتوقع أن تكون قادراً على أن:

- ✓ **توضح** المعنى العام لتحليل الخوارزميات.
- ✓ **تصف** معدلات النمو والترتيب الزمني للخوارزميات.
- ✓ **تميز** بين المقاييس الزمنية المختلفة لمعدلات النمو والمقارنة بينها.

1. المعنى العام لتحليل الخوارزميات

يعطينا تحليل الخوارزمية معلومات هامة عن الزمن الذي تحتاجه الخوارزمية لإنجاز مسألة معينة، ولكل خوارزمية، نستطيع أن نصل لتقدير لمقدار الزمن الذي تستغرقه الخوارزمية لحل مسألة ما لها مجموعة من N قيمة كمدخل. لذا مثلاً لربما نحتاج لتحديد عدد المقارنات المطلوبة لترتيب قائمة مكونة من N قيمة، أو لربما نحتاج لتحديد عدد العمليات الحسابية المطلوبة لضرب مصفوفتين في حجم N x N .

تعرف الخوارزمية بأنها مجموعة خطوات بسيطة لحل مسألة ما، ولذا يمكننا أن نجد مجموعة من الخوارزميات المختلفة لحل نفس المسألة. وعليه فإن من الضروري القيام بتحليل الخوارزميات لاختيار الأنسب والأفضل من بينها لحل أي مسألة بين أيدينا. وكمثالٍ سوف نقوم بكتابة خوارزميتين لحل مسألة إيجاد العنصر الأكبر بين ثلاثة عناصر مختلفة على النحو الآتي:

الخوارزمية الأولى

```
largest = a
if    b > largest then
          largest = b
end if
if c > largest  then
        largest = c
end if
return largest
```

الخوارزمية الثانية

```
if a>b  then
        if a>c   then
                largest = a
        else
                largest = c
```

```
        end if
else
        if  b>c   then
                largest = b
        else
                largest = c
        end if
end if
return largest
```

إذا فحصت الخوارزميتين السابقتين، يمكنك أن تحصي أن كل خوارزمية تحتاج لمقارنتين للوصول للحل الصحيح، لكن مع أن الخوارزمية الأولية هي الأوضح والأسهل في الفهم والاستيعاب، فإن الخوارزميتين لهما نفس المستوى من ناحية التعقيد عند تنفيذهما بالحاسوب.

بالنسبة **للزمن** فإن الخوارزميتين تتضمان لنفس المجموعة الزمنية، أما بالنسبة **للمساحة** المطلوبة داخل الحاسوب فإننا نجد أن الخوارزمية الثانية هي الأفضل نتيجة للمتغير largest والذي تحتاج الخوارزمية الثانية له لمرة واحدة فقط للوصول للحل الصحيح، بينما تحتاج الخوارزمية الأولي لتكراره لمرتين لنفس الغرض.

إن مقياس المساحة لا يكون ذا أهميَّة تذكر في حالة مقارنة أرقام أو حروف بسيطة. لكن في حالة مقارنة عناصر بيانية معقدة مثل الصور مثلاً فإن المساحة المطلوبة داخل الحاسوب تكون ذات أهمية بالغة، عندما نكون مهتمين فقط بكفاءة الخوارزمية لحل مسألة ما، نقوم بالتركيز على مقياس الزمن فقط.

إن تحليل الخوارزميات لا يعني أن نقوم بتقديم صيغة لحساب الزمن الفعلي بالثواني أو دورات الحاسوب التي تحتاجها الخوارزمية المعنية لحل مسألة ما، هذه المعلومات ليست ذات أهمية حيث إننا لا نقوم بالتركيز على نوع حاسوب محدد؛ أي هل يخدم مستخدم واحد أم عدة مستخدمين وما هو نوع المعالج الذي يستخدمه وكيف يقوم المترجم بتمثيل شفرة البرنامج. إذا كنا نهتم بهذه المعلومات السابقة فهذا يعني أن

الخوارزمية سيئة التصميم إذا تمت برمجتها ووضعت داخل حاسوب سريع فإنها سوف تكون أفضل لأنها سوف تنجز مهمتها بصورة أسرع، وهذا غير صحيح، ولذا فإننا نقوم بتحليل الخوارزميات دون اعتبار لأي نوع من الحواسيب.

1.1 حجم المدخلات

أن المدخلات تلعب دوراً هاماً في تحليل الخوارزميات لأنها تحدد المسار الذي تسلكه الخوارزمية عند التنفيذ. مثلاً إذا أعدنا كتابة الخوارزمية الأولى لأيجاد العنصر الأكبر ضمن N عنصر مختلفاً على النحو الآتي:

```
largest = list [1]
for i = 2     to N  do
    if (list [i] > largest)     then
            largest = list [i]
    end if
end for
return largest
```

إذا نظرنا للخوارزمية السابقة نجد أن ترتيب المدخلات يحدد سلوك الخوارزميات لإيجاد العنصر الأكبر، فمثلاً إذا تم ترتيب المدخلات ترتيباً تنازلياً أي أن العنصر الأكبر في المركز الأول فإن الخوارزمية لا تحتاج إلا للتعيين الذي تم قبل الحلقة للوصول للحل الصحيح، بينما إذا تم ترتيب العناصر للخوارزمية ترتيباً تصاعدياً، فإن الخوارزمية سوف تحتاج إلى التعيين الأساسي قبل الحلقة وإلي عدد (N-1) تعين داخل الحلقة. لذا إن تحليل الخوارزميات لابد أن يأخذ في اعتباره جميع الحالات المحتملة للمدخلات، وبالرجوع للخوارزمية نجد أنه إذا كان المطلوب تحديد الأكبر ضمن 10 عناصر مثلاً، فإن هنالك 101 أو 3.628.800 طريقة لتنظيم المدخلات للخوارزمية. ولذا نحتاج لتقسيم هذه الحالات المختلفة من المدخلات إلى 10 أصناف مختلفة بحيث إن **أفضلها** يحتاج فقط إلي تعيين واحد **وأسوؤها** يحتاج إلي 10 تعيينات لنفس الغرض.

أسئلة تقويم ذاتي

1. ما المقاييس التي تستخدم في المقارنة بين الخوارزميات.
2. هل تحليل الخوارزميات يعني تقديم صيغة لحساب الزمن الفعلي بالثواني أو دورات الحاسوب التي تحتاجها الخوارزمية المعينة لحل مسألة ما.

2.1 تعقيد المساحة Space Complexity

إن بعض صيغ التحليل يمكن إجراؤها اعتماداً على المساحة المطلوبة لإنجاز الخوارزمية لمهمتها. كان تحليل تعقيد المساحة داخل الحاسوب مهماً في السابق عندما كانت السعة التخزينية للحواسيب محدودة. وعندما نهتم بتعقيد المساحة فإن الخوارزميات يتم تقسيمها إلي قسمين:

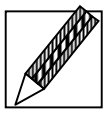
1) خوارزميات تحتاج إلي مساحة إضافية لإنجاز مهامها.
2) خوارزميات تستطيع إنجاز مهامها بالمساحة الموجودة.

و لا يكون معقولاً أن يقوم المبرمج باختيار الخوارزمية الأبطأ فقط لأنها تقوم بإنجاز مهمتها في المساحة المتوفرة ولأنه لا توجد مساحة إضافية للخوارزميات الأسرع.

إن ذاكرة الحاسوب هي مقياس أساسي عند تحليل الخوارزميات، لذا نجد أن تحليل المساحة هو مبدأ هام يهتم بفحص البيانات المطلوب تخزينها داخل الحاسوب وذلك للبحث عن أفضل الطرق لتخزينها من المساحة المتوفرة، بالنظر للبرمجيات المتوفرة حالياً في الأسواق، نجد أن تحليل المساحة لا يتم استخدامه. وأقل البرامج تعقيداً يحتاج إلى مساحة داخل الحاسوب تقدر بالميقابيت. وذلك لأن شركات البرمجيات لا تهتم كثيراً بالمساحة وتقديرها بسبب أن المستخدم يستطيع شراء ذاكرة إضافية لحاسوبه

الشخصي ببساطة ولكن بالنسبة للحاسوب صغير الحجم أي المساعد الرقمي الشخصي (PDA) ، فالأمر يختلف حيث إن المطلوب تطوير برمجيات بسيطة تؤدي أدواراً هامة وتحتفظ بالمساحة المتوفرة والتي لا تتعدى 2 أي 8 ميقابيت.

تدريب (1)



1. اكتب خوارزمية بدون استخدام التعبيرات الشرطية المركبة لتحديد ما إذا كانت ثلاثة أعداد صحيحة جميعها مختلفة (مفرده). في المتوسط كم هو عدد المقارنات التي تحتاجها خوارزميتك؟ خذ في اعتبارك جميع المجالات المحتملة للمدخلات.
2. اكتب خوارزمية لأيجاد العنصر الأكبر المثالي ضمن قائمة من N عنصر. ما هو عدد المقارنات التي تحتاجها خوارزميتك في أسوأ حالاتها؟

2. معدلات النمو Rates of Growth

في تحليل الخوارزميات ليس مهماً التعرف على العدد الفعلي للعمليات التي تحتاجها الخوارزمية لأداء مهمتها. ولكن المهم هو التعرف على معدل الازدياد في العمليات التي تحتاجها الخوارزمية كلما زاد حجم المسألة المطلوب حلها بواسطة الخوارزمية. وهذا المبدأ يطلق عليه معدل نمو الخوارزمية. ولهذا تم وضع بعض الاصطلاحات الرياضية التي تحدد الأصناف التي يمكن أن تتبع لها معدلات نمو الخوارزمية المختلفة.

1.2 تصنيف معدلات النمو

Classification of Growth

لأن معدل نمو الخوارزميات مهمٌ في تحليل الخوارزميات المختلفة، ظهر ما يعرف بالترتيب النسبي للدوال أو الخوارزميات وهذا يعني أننا يمكن أن نصنف الخوارزميات مع بعضها البعض حسب ترتيبها النسبي وفقاً لثلاثة أصناف رئيسة.

الصنف الأول: يشتمل على الخوارزميات التي تنمو بمعدلات أقل أو تعادل دالة ما.

الصنف الثاني: يعبر عن الخوارزميات التي تنمو بمعدلات متكافئة.

الصنف الثالث: يمثل الخوارزميات التي تنمو بمعدل أعلى أو تعادل دالة ما.

وعليه سوف نقوم بشرح الأصناف الثلاثة في الفقرات القادمة:

1.1.2 اصطلاح O الكبيرة Big O Notation

سوف نقوم باستخدام اصطلاحاً ويكتب O(f) ويطلق عليه O الكبيرة، ويمثل الصنف الأول وهو صنف الدوال التي تنمو بمعدلات أقل وتكافئ الدالة f . وهذا يعني بأن الدالة f هي أقل حد أعلي لهذا الصنف من الدوال. ويعبر عنه رياضياً على النحو الآتي:

لتكن T(n) و f(n) دالتين في العدد الصحيح n و $(n\geq 0)$ يكتب اصطلاحاً:

$$T(n) = O(f(n))$$

أو $T(n) \in O(f(n))$

عندما يكون هنالك ثابتان n_0 و c بحيث

$T(n) \leq c\, f(n)$ عند $n \geq n_0$

وهذا الصنف ذو أهمية بالغة في تحليل الخوارزميات، حيث بواسطته يمكننا مقارنة الخوارزميات المختلفة.

فمثلاً إذا وجدنا خوارزمية تقع ضمن صنف دالة O لخوارزمية أخرى أو هي ضمن صنف دالة O أقل من دالة O لخوارزمية أخري، فإننا يمكن أن نصل إلي الخوارزمية الأولى لهما بكفاءة من ناحية المعدل الزمني أفضل من الخوارزمية الثانية.

مثال (1)

أثبت أن:

$100n + 5 = O(n^2)$

لاحظ أن :

$100n + 5 \leq 100n + n = 101n \leq 101n^2$

لكل $n \geq 5$

وهكذا نكون أخذنا $c = 101$ و $n = 5$

لإثبات أن $100n + 5 = O(n^2)$

مثال (2)

أثبت أن:

$3n(n+1) = O(n^2)$

الحل: لاحظ أن:

$3n^2 + n \leq 4n^2$

لكل $(n \geq 1)$

مثلاً خذ $n = 2$ نجد أن:

$3X2^2 + 2 = 14 < 4 X 2^2 = 16$

إذاً فإننا نأخذ $c = 4$ و $n_0 = 1$ لإثبات صحة العلاقة المطلوبة.

2.1.2 إصطلاح Ω الكبير Big Omega Notaion

نستخدم مصطلح Ω(f) ونطلق عليه Ω الكبيرة لتمثيل صنف الدوال التي تنمو بمعدلات أعلى أو تكافئ للدالة f وهذا يعني أن الدالة f تمثل أكبر حد أدنى لهذا الصنف من الدوال. ونعبر عن هذا المصطلح رياضياً كالآتي:

لتكن T(n) و f(n) دالتين في العدد الصحيح n و $n \geq 0$) يكتب:

$T(n) = \Omega((f(n))$ أو $T(n) \in \Omega(f(n))$

إذا وجد ثابتان c و n_0 بحيث

$T(n) \geq c\ f(n)$ عند $n \geq n_0$

وحيث إننا نهتم بكفاءة الخوارزميات فإن هذا المقياس لا يكون هاماً لدينا مثل المقياس السابق O(f) .

مثال (3)

أثبت أن:

$n^3 = \Omega(n^2)$

الحل: خذ $c=1$, و $n_0=1$

لاحظ $n^3 \geq n^2$ لكل $n \geq 0$

إذاً العلاقة المطلوبة صحيحة

3.1.2 اصطلاح θ الكبيرة Big Theta Notation

يعني مصطلح θ (f) ويطلق عليه **θ** الكبيرة عن صنف الدوال التي تكافئ معدلاتها في النمو معدل الدالة f . وهذا يعني أنَّ الدالة f تمثل أقل حد أعلى وأكبر حد أعلى في نفس الوقت لهذا الصنف من الدوال. ويكتب رياضياً كالآتي:

لتكن T(n) و f(n) دالتين في العدد الصحيح n و $(n \geq 0)$. يكتب

$T(n) = \boldsymbol{\theta}((f(n))$ أو $T(n) \in \boldsymbol{\theta}(f(n))$

إذا وجد ثلاثة ثوابت c_2 و c_1 و n_0 بحيث

$T(n) \leq c_1\ f(n)$ عند $n \geq n_0$

$T(n) \geq c_2\ f(n)$ عند $n \geq n_0$

أو بمعنى آخر إذا وفقط إذا كان:

$T(n) = \mathbf{O}\ ((f(n))$ أو $T(n) \in \mathbf{O}\ (f(n))$

$T(n) = \Omega\ ((f(n))$ أو $T(n) \in \Omega\ ((f(n))$

ملاحظة

بالنسبة لهذا المقياس (Big$\boldsymbol{\theta}$) لا يمثل أهمية لدينا حيث أننا نبحث عن الخوارزمية الأفضل في الكفاءة، ولهذا لا يتم الرجوع إليه كثيراً.

مثال (4)

أثبت أن:

$$½\ n\ (n-1) = \theta(n^2)$$

الحل:المطلوب هو إيجاد ثلاثة ثوابت c_1, c_2, n_0 بحيث :

$c_1\ n^2 \leq ½\ n(n-1) \leq c_2\ n^2$ عند $n \geq n_0$

أولاً: إثبات الطرف الأيمن:

$$½n\ (n-1) = ½\ n^2 - ½\ n \leq ½\ n^2$$

لكل: $n \geq 0$

ثانياً: إثبات الطرف الأيسر:

$$½n\ (n-1) = ½\ n^2 - ½\ n \geq ½\ n^2 - ½n - ½n = ¼n^2$$

لكل $n \geq 2$

إذاً سوف يتم اختيار $n_0 = 2,\ c_2 = ¼,\ c_1 = ½$

لإثبات العلاقة المطلوبة.

أسئلة تقويم ذاتي

1) علل:

الصنف O الكبيرة يعتبر ذا أهمية كبيرة وبواسطته يمكن مقارنة الخوارزميات المختلفة.

2.2 خواص معدلات النمو

عند استخدام التعريفات السابقة لمعدلات النمو والاصطلاحات حولهـا، يمكننـا إثبات بعض الخواص حولها. الخواص أدناه هي خواص هامة في تحليل الخوارزميات وخاصة في تحليل التتابع والتكرار في أي خوارزمية.

نظرية (1)

إذا كان $T_1(n) = O(f(n))$ و $T_2(n) = O(g(n))$ فإن

$$T_1(n) + T_2(n) = O(\max(f(n)), (g(n)))$$

الإثبات:

بما أن $T_1(n) = O(f(n))$

فإننا نجد أن هنالك ثابتين c_1, n_1

حيث $T_1(n) \le c_1 f(n)$ عند $n \ge n_1$

أيضاً بما أن $T_2(n) = O(g(n))$

فإننا سوف نجد ثابتين c_2, n_2 حيث

$$T_2(n) \le c_2 g(n) \text{ عند } n \ge n_2$$

ضع $c_3 = \max\{c_1, c_2\}$ وضع $n \ge \max\{n_1, n_2\}$

لذا يمكننا استخدام المتباينتين السابقتين على نحو:

$$T_1(n)+T_2(n) \le c_1 f(n)+c_2 g(n)$$
$$\le c_3 f(n)+c_3 g(n)$$
$$= c_3 \big(f(n)+g(n)\big)$$
$$\le 2\, c_3 \max\{ f(n)+ g(n \}$$

إذا نصل إلى أن المجموع $T_1(n) + T_2(n)$ هو يتبع لصنف دالة Oالخاص ب $\max\{f(n),g(n)\}$من التعريف حيث ان الثابتان المطلوبان هما:

$$C=2C_3=2 \ \max\{ C_1, C_2\}$$
$$n_0= \max\{n_1, n_2\}$$

ملاحظة

هذه الخاصية هامة جداً عند تحليل أي خوارزمية تحتوي على جزأين متتابعين حيث إننا يمكن حساب كفاءة الخوارزمية من الجزء الذي يحتاج إلي معدل النمو الأكبر.

نظرية (2)

$$T_1(n) = O\ (f(n))$$
$$T_2(n) = O(g(n))$$

فان

$$T_1(n)*\ T_2(n)=O\big(f(n)*\ g(n)\big)$$

أيضاً يمكنك إثبات هذه النظرية بسهولة بنفس الكيفية التي تم بها إثبات النظرية السابقة.وهذه الخاصية الثانية هامة أيضاً في تحليل الخوارزميات التى تحتوي على أجزاء متكررة أي حلقات (loops) ، حيث يمكننا حساب كفاءة الخوارزمية عند هذه الأجزاء بواسطة الحصول على حاصل ضرب حجم الحلقة أو الحلقات في معدل النمو للجزء الداخلي للحلقة أو الحلقات.

3.2 كيفية الحصول على معدلات النمو

بالإضافة للتعريفات السابقة والاصطلاحات **O** و**Ω**و**Θ** ، فإننا يمكن الاستفادة من حساب نهاية نسبة دالتين في مقارنة الدوال المختلفة، وهنالك ثلاث حالات على النحو الآتي:

$$\lim_{n \to \infty} T(n)/f(n) = \begin{cases} 0 \ then \ T(n) = O\left(g(n)\right) \\ C \ then \ T(n) = \theta\left(g(n)\right) \\ \infty \ then \ T(n) = \Omega\left(g(n)\right) \end{cases}$$

أسئلة تقويم ذاتي

حلل العلاقة السابقة منطقياً موضحاً العلاقة التي تربط الدالتين f(n) و T(n) في الثلاث حالات.

مثال (5)

قارن معدلات نمو الدالتين

$g(n) = n^2$ و $f(n) = ½\ n(n-1)$

الحل: المطلوب هو الحصول على النهاية:

$$\lim_{n \to \infty} f(n)/g(n) = \lim_{n \to \infty} \frac{1}{2} n(n-1)/n^2$$

$$= \frac{1}{2} \lim_{n \to \infty} (n^2 - n)/n^2$$

$$= \frac{1}{2} \lim_{n \to \infty} (1 - 1/n)$$

$$= \frac{1}{2} \lim_{n \to \infty} 1 - \frac{1}{2} \lim_{n \to \infty} 1/n$$

$$= \frac{1}{2}(1-0) = \frac{1}{2}$$

ولهذا يمكننا أن نكتب العلاقة التالية:

$$f(n) = \Theta\left(g(n)\right)$$

وهذا يعنى أن الدالتين لهما نفس معدلات النمو.

مثال (6)

قارن معدلات نمو الدالتين:

$$g(n) = \sqrt{n} \quad , \quad f(n) = \log_2 n$$

الحل: المطلوب حساب النهاية التالية:

$$\lim_{n \to \infty} f(n)/g(n) = \lim_{n \to \infty} \log_2 n / \sqrt{n}$$

يمكننا استخدام قاعدة هبتل كالآتى:

$$\lim_{n \to \infty} \log_2 n / \sqrt{n} = \lim_{n \to \infty} (\log_2 n)/(\sqrt{n})$$

$$= \lim_{n \to \infty} \left((\log_n e)1/n\right)/\frac{1}{2}\sqrt{n}$$

$$= 2\log_2 e \lim_{n \to \infty} \sqrt{n}/n$$

$$= 2\log_2 e \lim_{n \to \infty} 1/\sqrt{n} = 0$$

$$= \lim_{n \to \infty} \left((\log_n e)1/n\right)$$

$$= 2\log_2 e \lim_{n \to \infty} \sqrt{n}/n$$

$$= 2\log_2 e \lim_{n \to \infty} 1/\sqrt{n} = 0$$

ولهذا يمكننا أن نصل للعلاقة التالية:

$$f(n) = O(g(n))$$

ملاحظة

في بعض الأحيان العلاقة السابقة يتم التعبير عنها بواسطة اصطلاح جديد يطلق عليه o الصغيرة ويكتب كالأتي:

$$f(n) = O(g(n))$$

(ولكن هذا الاصطلاح الأخير يستخدم نادراً)

عموماً هنالك بعض المقاييس الزمنية الشائعة والتي تستخدم في مقارنة المعدلات الزمنية للخوارزميات المختلفة. ويمكن ترتيب هذه المقاييس حسب درجة قوتها من أعلى إلي أسفل كما هو موضح في الجدول أدناه:

	المقياس	الاسم العلمي
1	1	ثابت (constant)
2	$\mathbf{Log_n}$	لوغريثمى (logarithmic)
3	**n**	خطي (linear)
4	$\mathbf{n \log_n}$	خطي لوغريثمى (linear logarithmic)
5	$\mathbf{n^2}$	تربيعي (quadratic)
6	$\mathbf{n^3}$	تكعيبي (cubic)
7	$\mathbf{2^n}$	أسي (Exponential)
8	**n!**	مضروب(factorial)

أ) المعدل الثابت Constant Rate

المعدل الزمني الثابت يشار إليه بـ O(1) وهو الوقت المطلوب لإنجاز أي عملية بسيطة كجملة التعيين أو استبدال قيمة بقيمة في عملية الترتيب أو جملة قراءة البيانات من الشاشة. وهذا المعدل دائماً يكون ملازماً للحالة الأفضل التي قد تمر بالخوارزميات المختلفة.

ب) المعدل اللوغريثمي Logarithmic Rate O($\log_n$)

المعدل الزمني اللوغريثمي هو الوقت المطلوب لإنجاز عمليات تلازم مسألة معينة بواسطة تقليص حجمها بنسبة ثابتة في كل مرة يتم فيها تكرار تلك العمليات؛ فمثلاً البحث الثنائي يحتاج إلي معدل لوغريثمي O($\log_n$) ، والجملة الدورانية التالية أيضاً تحتاج إلي معدل لوغريثمي.

```
k=1
While Begin (k < = N) do
        k=k div 2
    end while
```

ج) المعدل الخطي Linear Rate

أن الخوارزمية التي يشار إليها بمعدل زمني o(n) هي خوارزمية ذات وقت تنفيذ يتناسب طردياً مع حجم المسألة أي مع أي زيادة تحدث في حجم البيانات. من أمثلة الخوارزميات التي تحتاج إلي معدل خطي خوارزمية الحصول على مجموع البيانات لأننا نحتاج أن نزور جميع البيانات.

د) المعدل الخطي اللوغريثمي Linear Logarithmic Rate

معظم الخوارزميات التي تحتاج إلي معدل خطي لوغريثمي o(nlogn) تكون خوارزميات فرق تسد. فمثلاً خوارزمية الترتيب السريع تحتاج إلي معدل خطي لوغريثمي، والجملة الدورانية التالية أيضاً تحتاج إلي معدل O(nlogn) :

```
for  i = 1   to   n    do
     Begin
```

```
        j=i
    while  (j ≠ 0)  do
        Begin
            j = j div  2
        end while
end for
```

هـ) المعدلات التربيعية والتكعيبية Quadratic & Cubic Rate

معظم الخوارزميات التي تحتاج معدلات تربيعية($O(n^2)$ أو تكعيبية($O(n^3)$ تحتوي على أجزاء دورانية متداخلة، مثلاً خوارزمية الترتيب بالإدخال تحتاج إلي معدل تربيعي وخوارزمية ضرب مصفوفتين تحتاج إلي معدل تكعيبي.

و) المعدل الأسي Exponential Rate

أي خوارزمية تحتاج إلي وقت تنفيذ بمعدل أسي ($O(2^n)$ هي خوارزمية غير عملية ولا ينصح باختيارها، والخوارزمية ذات المعدل الأسي هي الخوارزمية التي تتطلب توحيد جميع المجموعات الجزئية لمجموعة عناصر ذات حجم n . وأحد أمثلتها المشهورة هو خوارزمية توليد أرقام فايبونسي.

ي) معدل المضروب Factorial Rate

أي خوارزمية تقوم بتوليد جميع تباديل لمجموعة عناصر ذات حجم n تكون خوارزمية ذات معدل زمني $O(n!)$ ومن أمثلتها الخوارزمية التي تقوم بالبحث عن أفضل جدولة للمهام المطلوب تنفيذها في معالج الحاسوب وأيضاً الخوارزميات التي تحتاج إلي معدلات $O(n!)$ لا تعتبر خوارزميات عملية ولا ينصح بها، وكقاعدة عامة ينبغي عدم اختيار خوارزمية يزيد معدل كفاءتها الزمنية عن المعدل التكعيبي $O(n^3)$

تدريب (2)

(1) اثبت العلاقات التالية:

$\frac{1}{2}n(n+1)= O(n^3)$ (i)

$\frac{1}{2}n(n+1)= \Theta(n^2)$ (ii)

$\frac{1}{2}n(n+1)= \Omega(n)$ (iii)

(2) عين الصنف $\Theta(f(n))$ التي تتبع له كل الدوال أدناه.

$(n^2+1)^{10}$ (i)

$\sqrt{10n^2+7n+3}$ (ii)

$2n \log(n+1)^2$ (iii)

$2^{n+1}+3^{n-1}$ (iv)

(3) رتب الدوال التالية حسب معدلات نموها من اسفل الى أعلى.

$n \log n$ $n+n^2+n^3$ $n^3+\log n$ 2^n

$n^2 \sqrt{n}$ $(\frac{1}{2})^n$ $(\log n)^2$

$\log n$ n^3 n $n!$

الخلاصة

عزيزي الدارس،

في هذه الوحدة عرَّفنا الخوارزمية بأنها مجموعة خطوات لحل مسألة ما، وقلنا إن تحليل الخوارزمية يعطينا معلومات هامة عن الزمن الذي تحتاجه الخوارزمية لإنجاز مسألة معينة.

كما علمنا أن المدخلات (Input) تلعب دوراً هاماً في تحليل الخوارزمية لأنها تحدد المسار الذي تسلكه الخوارزمية.

ثم بيَّنا أن بعض صيغ التحليل يمكن إجراؤها اعتماداً على المساحة المطلوبة لإنجاز الخوارزمية لمهمتها، وقد قسمنا الخوارزميات إلى قسمين الأول خوارزميات تحتاج إلى مساحة إضافية لإنجاز مهمتها والثاني خوارزميات تستطيع إنجاز مهامها بالمساحة الموجودة.

ثم تعرفنا على معدل نمو العمليات التي تحتاجها الخوارزمية كلما زاد حجم المسألة المطلوب حلها بواسطة الخوارزمية. وقمنا بتصنيفها إلى ثلاثة أصناف وهي:

أولاً: اصطلاح O الكبيرة، ويمثل صنف الدوال التي تنمو بمعدلات أقل أو تكافئ الدالة f، وهذا الصنف أهم الأصناف الثلاثة.

ثانياً: اصطلاح Θ الكبيرة ويمثل صنف الدوال التي تنمو بمعدلات أعلى أو تكافئ الدالة f .

ثالثاً: اصطلاح Ω الكبيرة ويمثل صنف الدوال التي تكافئ معدلاتها في النمو معدل الدالة f وهذا الصنف لا يتم الرجوع إليه كثيراً.

كما عرفنا خواص معدلات النمو هي:

إذا كان $T_1(n) = O(f(n))$ و $T_2(n) = O(\log(n))$ **فإن:**

(1) $T_1(n)+ T_2(n) = O(\max (f(n), g(n))$

(2) $T_1(n)* T_2(n) = O((f(n)* g(n))$

بالإضافة لذلك فإننا يمكننا الحصول على معدلات النمو بحساب نهاية نسبة دالتين على النحو التالي:

$$\lim_{n \to \infty} T(n)/f(n) = \begin{cases} O & then \quad T(n) = O\big(g(n)\big) \\ C & then \quad T(n) = \theta\big(g(n)\big) \\ \infty & then \quad T(n) = \Omega\big(g(n)\big) \end{cases}$$

وأخيراً ذكرنا أن هنالك بعض المقاييس الزمنية الشائعة والتي تستخدم في مقارنة المعدلات الزمنية للخوارزميات المختلفة، وهي الثابت واللوغريثمي والخطي والتربيعي والتكعيبي والأسي والمضروب.

لمحة مسبقة عن الوحدة التالية

عزيزي الدارس، بعد أن أصبحت تملك المبادئ الأساسية لتحليل الخوارزميات سنتعرف في وحدتنا التالية على دراسة الخوارزميات المختلفة بغرض التعرف على مدى كفاءتها من حيث وقت التنفيذ اللازم.

إجابات التدريبات

تدريب (1)

1. أحسن حالة مقارنة واحد فقط وأسوأ حالة مقارنة ثلاثة

```
if (a=b) then
return (0);
end if
if (a=c) then
return (0);
end if
if (b=c) then
return (0)
else return (1)
```

2.

```
largest= list [1]
for   i=2 to N  do
if (list[i] > largest) then
largest = list [i]
end
end
return largest
```

في أسوأ حالة تحتاج إلى (N-1) مقارنة إذا كانت العناصر مرتبة ترتيب تصاعدي.

تدريب (2)

(1)

(i)

$N \geq n_0, T(n) \leq cf(n)$

$c = 1$

$n_0 = 1$

$\frac{1}{2} n(n+1) = \frac{1}{2} n^2 + \frac{1}{2} n \leq \frac{1}{2} n(n+n)$

$n \geq 1$ **لكل**

$\Rightarrow \frac{1}{2} n(n+1) \leq n^2 \leq n^3 \Rightarrow \frac{1}{2} n(n+1) = O(n^3)$

تكعيبي (Cubic)

(ii)

$n \geq n_0, \quad c_1 n^2 \leq \frac{1}{2} n(n+1) \leq c_2 n^2$

$L.H.S \quad choose \quad c_1 = \frac{1}{2} \quad n \geq 0$

$R.H.S \quad c_1 = \frac{1}{2} n^2 + \frac{1}{2} n \leq c_2 n^2$

$n \geq 0 \quad chose \ c_2 = 1 \quad \Rightarrow \frac{1}{2} n(n+1) = \theta(n^2)$

تربيعي (quadratic)

(iii)

$T(n) \geq cf(n)$

$n \geq n_0$

$\frac{1}{2} n(n+1) = \Omega(n)$

$choose \ \ c_1 = \frac{1}{2}$

$\frac{1}{2} n^2 + \frac{1}{2} n \geq \frac{1}{2} n$

$n \geq 0 \quad \Rightarrow \frac{1}{2} n(n+1) = \Omega(n)$

خطي (Linear)

(2)

(i)

$$T(n)=(n^2+1)^{10},\ f(n)=n^{20}$$

$$\lim_{n\to\infty}\frac{T(n)}{f(n)}=\lim_{n\to\infty}\frac{(n^2+1)^{10}}{n^{20}}=\frac{\lim_{n\to\infty}(n^{20}+10n^{19}+\ldots+1)}{n^{20}}=1=c$$

$$=f(n)=n^{20}\Rightarrow(n^2+1)^{10}=\theta(n^{20})$$

(ii)

$$T(n)=(10n^2+7n+3)^{1/2}=\sqrt{10n^2+7n+3}$$

$$f(n)=n$$

$$\Rightarrow\lim_{n\to\infty}\frac{T(n)}{f(n)}=\lim_{n\to\infty}\sqrt{\frac{10n^2+7n+3}{n}}=\lim_{n\to\infty}\sqrt{\frac{10n^2+7n+3}{n^2}}$$

$$=\lim_{n\to\infty}\sqrt{10n^2+7/n+\frac{3}{n^2}}=\sqrt{10}=c$$

$$\Rightarrow\sqrt{10n^2+7n+3}=\theta(n)$$

(iii)

$$2n\log(n+1)^2=4n\log(n+1)$$

$$let\ \ T_1(n)=4n, T_2(n)=\log(n+1)$$

$$f_1(n)=n\Rightarrow\lim_{n\to\infty}\frac{T_1(n)}{f_1(n)}=\lim_{n\to\infty}\frac{4n}{n}=4$$

$$f_2(n)=\log(n+1)\Rightarrow\lim_{n\to\infty}\frac{T_2(n)}{f_2(n)}=\lim_{n\to\infty}\frac{\log(n+1)}{\log(n+1)}=1$$

$$\Rightarrow 2u\quad\log(n+1)^2=\theta(n\log n)$$

(خطي لوغرثيمي)

(iv)

$$2^{n+1}+3^{n+1}$$

$$f_1(n)=2^{n+1}$$

$$f_2(n)=2^{n+1}$$ **exponential أسي**

$$\max(\theta(2^{n+1}),\theta(2)^{n-1})=\theta(2^{n+1})$$ **expanential أسي**

1) $\left(1/2\right)^n$

2) $\log^n$

3) $\sqrt{n}$

4) $(\log)^2$

5) $\log n^n$

6) $n \log n$

7) n^2

8) $n + n^2 + n^3$

9) $n^3 + \log n$

10) n^3

11) 2^n

12) $n!$

$n \log n$	~~$n+n^2+n^3$~~	$n^3 + \log n$	2^n	$\left(1/2\right)^n$	$(\log)^2$	$\sqrt{n}$	n^2	$\log^n$	n^3	n	$n!$	
2	14	9	4	$1/4$	1	$\sqrt{2}$	4	1	8	2	2	2
8	84	66	16	$1/16$	4	2	16	2	64	4	24	4
24	584	215	256	$1/256$	9	$2\sqrt{2}$	64	3	584	8	40330	8
64	4368	4100	256×256	$1/256 \times 1/25$	16	4	256	4	256×16	16	16!	16

مسرد المصطلحات

- **الخوارزميةAlogrithm**

مجموعة خطوات بسيطة لحل مسألة ما.

- **التحليلAnalysis**

إعطاء معلومات هامة عن زمن ومساحة الخوارزمية

- **مصطلح O Notaion O**

يمثل صنف الدوال التي تنمو بمعدلات اقل وتكافئ الدالة.

- **مصطلح θ Notaion θ**

يمثل صنف الدوال التي تكافئ معدلاتها في النمو معدل الدالة .

- **مصطلح Ω Notaion Ω**

تمثل صنف الدوال التي تنمو بمعدلات اعلي او تكافئ الدالة.

- **معدل النموRate of growth**

هو معدل الازدياد في العمليات التي تحتاجها الخوارزمية كلما زاد حجم المسألة المطلوب حلها بواسطة الخوارزمية.

- **ترتيب نسبيRelative Order**
- **تعقيد المساحةSpace Complexity**

تحليل المساحة المطلوبة لإنجاز الخوارزمية لمهمتها.

- **تعقيد الزمنTime Complexity**

تحليل الزمن المطلوب لإنجاز الخوارزمية لمهمتها.

- **معدل خطيLinear Rate**

تعني أن وقت تنفيذ الخوارزمية يتناسب طرديا مع أي زيادة تحدث في حجم البيانات.

- **معدلات تربيعية و تكعيبية Cubic & Quadratic Rate**

هي المعدلات الزمنية للخوارزميات التي تحتوي علي أجزاء دورانية متداخلة.

- **معدل أسيExponential Rate**

هو معدل الخوارزمية التي تتطلب توحيد جميع المجموعات الجزئية لمجموعة عناصر ذات حجم n.

- **معدل لوغريثميLogarithmic Rate**

هو الوقت اللازم لإنجاز عمليات تلازم مسألة معينة بواسطة تقليص حجمها بنسبة ثابتة في كل مرة يتم فيها تكرار تلك العمليات.

- **معدل ثابتConstant Rate**

الوقت المطلوب لإنجاز أي عملية بسيطة .

- **زمن التنفيذRuning Time**

الزمن الازم لتنفيذ الخوارزمية.

المراجع

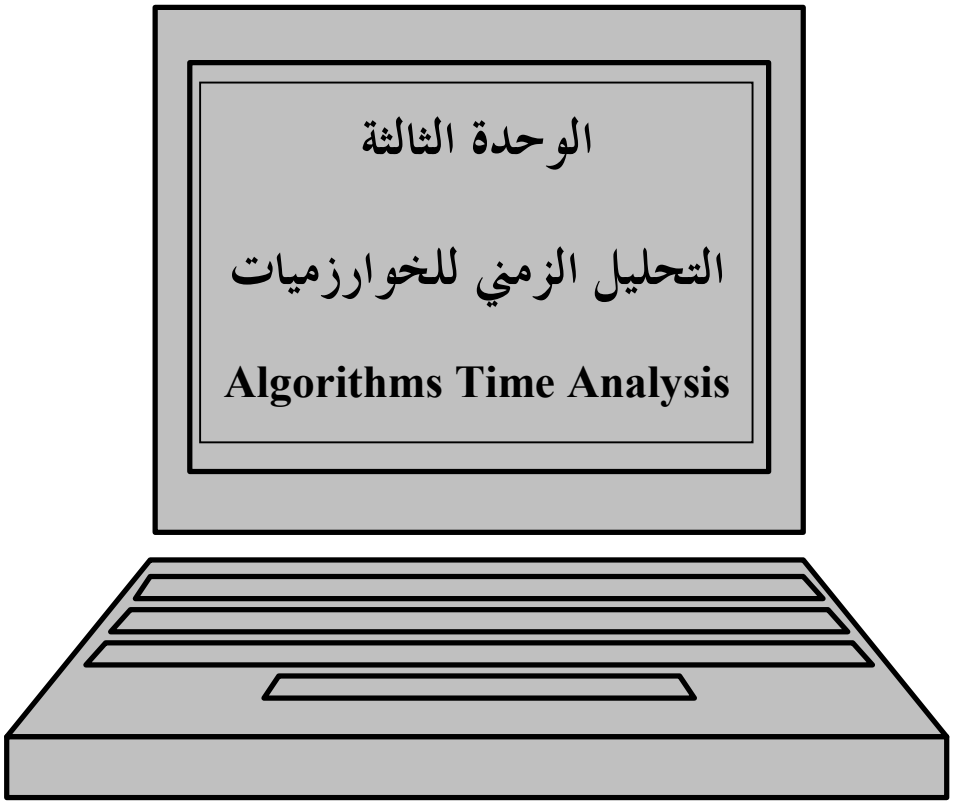
المراجع العربية

1) مجموعة مؤلفين، تركيب البيانات وتصميم الخوارزميات، منشورات جامعة القدس المفتوحة، 1998.

2) السماني عبد المطلب، هياكل البيانات، منشورات جامعة السودان المفتوحة، 2005.

المراجع الأجنبية

1) Weiss, M.A., Data Structures and Alogrithm Analysis, Benjamin pub., 1992.
2) McConnell, J.J., Analysis of Alogrithms: An Active Approach, Jones pub., 2001.
3) Lavitin, A., Introduction to the Design and Analysis of Alogrithms, Addisions ues by, 2003.
4) Ulman, J.D (etal), The Design and Analysis of Compurte Alogrithms, Addison Wesley, 1974.
5) Basese, S., Compurte Alogrithms : Introduction to Analysis and Design, Addison Wesley, 2000.
6) Sedgewick, R., An Introduction to the Analysis of Alogrithms, Addison Wesley 1996.


الوحدة الثالثة
التحليل الزمني للخوارزميات
Algorithms Time Analysis

محتويات الوحدة

المقدمة

تمهيد

في هذه الوحدة سوف نتعرف على كيفية التحليل الزمني للخوارزميات؛ ونقصد بالتحليل الزمني للخوارزميات دراسة الخوارزميات المختلفة بغرض التعرف على مدى كفاءتها من حيث وقت التنفيذ اللازم، وقد حاولنا في هذه الوحدة توضيح المفاهيم الرئيسة مستخدمين العديد من الأمثلة والتدريبات المناسبة.

تتكون هذه الوحدة من أربعة أقسام؛ في القسم الأول سوف نتعرف على النموذج العام للتحليل الزمني للخوارزميات، أما في القسم الثاني فسوف ندرس حالات المدخلات المختلفة وكيفية التعامل معها. وفي القسم الثالث فسوف نتعرَّف على كيفية حساب المعدلات الزمنية للخوارزميات التكرارية والقواعد المصاحبة لها. وأخيراً في القسم الرابع سوف نتعرف على كيفية تحليل خوارزميات الاستدعاء الذاتي.

أهداف الوحدة

عزيزي الدارس،

بعد دراسة هذه الوحدة وتنفيذ تدريباتها يتوقع منك أن تكون قادراً على أن:

- ✓ **تصف** كيفية التحليل الزمني للخوارزميات.
- ✓ **توضح** كيفية تنفيذ تحليل الحالات السيئة والحالات الوسطى والحالات الأفضل للخوارزميات.
- ✓ **تميز** بين التحليل الزمني للخوارزميات التكرارية و التحليل الزمني لخوارزميات الاستدعاء الذاتي.

1. كيفية التحليل الزمني للخوارزميات

عموماً يتم تقسيم الخوارزميات إلي نوعين رئيسين

1) **خوارزمية تكرارية** (Expetitive Alogrithm)

هي خوارزمية تحتوي على جمل دورانية (Loops) وجمل شرطية ولذا تحليلها يتطلب تحديد الجهد المبذول في الجمل الدورانية وعدد المرات التي تحتاجها الجمل الدورانية لإنجاز مهامها.

2) **خوارزمية استدعاء ذاتي** (Excursive Alogrithm)

خوارزميات الاستدعاء الذاتي هي خوارزمية تقسم المسألة إلي عدد من الأقسام الفرعية وتنفيذ الخوارزمية عليها ويتطلب تحليل خوارزمية الاستدعاء الذاتي تحديد الجهد المبذول لتقسيم المسألة بالإضافة للجهد المبذول لتكرار الخوارزمية لكل فرع وذلك للحصول على حل المسألة الكلي، ولذا يتم دائماً تجميع هذه المعلومات بالنسبة للمسائل الفرعية وحجمها الذي يوصلنا لما يعرف بالمتتابعة أو علامة الاستدعاء الذاتي (Recurrence Relation) ، و التي غالباً ما يتم تحديد الصيغة المغلفة لها لكي يسهل تحديد معدل نموها.

ما هي الأشياء التي يتم تحديدها عند تحليل الخوارزميات؟

هنالك خطوتان لعمل ذلك:

- **الخطوة الأولى:** يتم تحديد العملية أو العمليات الهامة .
- **الخطوة الثانية:** يتم تعيين العمليات المتكاملة مع الخوارزمية والعمليات المساعدة غير المعنوية بالنسبة للخوارزمية.

هنالك نوعان من العلميات الأساسية هما عمليات المقارنة وعمليات الحساب، فعمليات المقارنة جميعها يتم التعامل معها على أنها متكافئة، وهي هامة جداً في خوارزميات البحث والترتيب، حيث يتم مقارنة قيمتين لتحديد – في حالة البحث – هل

القيمة هي القيمة المطلوبة – وفي حالة الترتيب – هل القيمة هي داخل أم خارج الترتيب، أما عمليات الحساب فيتم تقسيمها إلي نوعين: إضافات وتضريبات. فعمليات الإضافات تشمل على عمليتي الجمع والطرح وعمليات التضريبات تشمل على الضرب والقسمة وخارج القسمة (modules)، ويتم إحصاء المجموعتين بطريقتين مختلفتين حيث إن عمليات التضريبات تحتاج إلي زمن أكثر من عمليات الإضافات.

أسئلة تقويم ذاتي

1) اذكر النوعين الرئيسيين للخوارزميات وما الفرق بينهما؟
2) ما هي الأشياء التي يتم تحديدها عند تحليل الخوارزمية؟

2. الحالات المختلفة للمدخلات Input Cases

عند تحليل الخوارزميات يجب تحديد طبيعة المدخلات والتي قد تكون هامة في لتحديد سلوك الخوارزمية. فالمدخلات المرتبة لربما تمثل حالة أفضل لبعض خوارزميات الترتيب وتمثل حالة سيئة للبعض الآخر، ولذا يجب عدم التعامل مع حالة واحدة فقط للمدخلات، حيث يجب تحديد الحالات التي تجعل الخوارزمية تؤدي مهامها بسرعة أو ببطء، وأيضاً يمكننا تمديد الحالات التي تعبر عن الأداء المتوسط للخوارزمية، وعليه هنالك ثلاث حالات مختلفة على النحو الآتي:

1.2 الحالة الأفضل Best Case

الحالة الأفضل للخوارزمية هي الحالة التي تكون فيها المدخلات علي صيغة تتطلب الخوارزمية أن تأخذ أقل زمن تنفيذ، ويتم التعبير عنها بواسطة $T_{best}(N)$ لخوارزمية تحتاج مدخلات ذات حجم N، فمثلاً في خوارزمية البحث، نجد أن الحالة الأفضل هي حالة المدخلات التي يكون فيها المفتاح المطلوب البحث عنه موجود في

أول المواقع التي سوف تختبرها الخوارزمية، وطبعاً هذا يتطلب فقط مقارنة واحدة لتحديد نتيجة البحث، وبالتالي يكون معدل الحالة الأفضل على النحو:

$T_{best}(N)=O(1)$

وبما إن الحالة الأفضل تعطي معدلات صغيرة فإنه لا يتم التعامل معها كثيراً في تحليل الخوارزميات.

2.2 الحالة الأسوء Worst Case

تعتبر الحالة الأسوء هامة جداً في تحليل الخوارزميات لأنها تعطينا فكرة عن الزمن الكلي الذي قد تحتاجه الخوارزمية لإنجاز مهمتها. إن الحالة الأسوء هي الحالة التي تتطلب منا تحديد صيغة المدخلات التي تتسبب في أن الخوارزمية تقوم بجميع الأعمال وفي أطول زمن ممكن، ويتم التعبير عن الحالة الأسوء لمدخلات ذات حجم N بواسطة $T_{worst}(N)$.

مثلاً في خوارزمية البحث فإن الحالة الأسوء هي الحالة التي يكون فيها المفتاح المطلوب في الموقع الأخير أو الحالة التي لا تتضمن المفتاح المطلوب أصلاً. ولذا هذه الحالة تتطلب مقارنة جميع المدخلات ذات الحجم N وذلك للوصول إلي القرار النهائي عن عملية البحث، وتعطي معدل زمن قدره:

$$T_{worst}(N) = O(N)$$

إن الحالة الأسوء توضح لنا الحد الأعلى وكيف يمكن أن تعمل الخوارزمية ببطء حسب اختيارنا، ولهذا ففي هذا المقرر سوف نتعامل دوماً مع الحالة الأسوء للخوارزميات بالمعدل $T(N)$.

3.2 الحالة الوسطى Average Case

أن الحالة الوسطى التي قد تمر بها الخوارزمية هي حالة صعب النيل منها لأنها تتطلب تفاصيل كثيرة، ولهذا نبدأ العملية الأساسية لتحديد المعدل الزمني عند الحالة الوسطى بحساب عدد المجموعات المختلفة للمدخلات المحتمل ورودها للخوارزمية. وفي المرحلة الثانية يتم تحديد الاحتمال لورود المدخلات من كل مجموعة من

مجموعات المدخلات المعينة مسبقاً. وفي المرحلة الثالثة يتم حساب معدل الزمن للخوارزمية، لكل مجموعات المدخلات. ولذا فإن المعدل الزمني للحالة الوسطى يُعطى بالصيغة التالية:

$$T_{avg}(N) = 1/m \sum_{i=1}^{m} P_i * T_i$$

حيثN هي حجم المدخلات

m هي عدد المجموعات

P_i هي احتمال المدخلات في المجموعة رقم i

T_i هي المعدل ألزمني عند المجموعة رقم i

في بعض الأحيان يكون الاحتمال متساوي لكل مجموعات ولذا يمكن كتابة الصيغة السابقة على النحو:

$$T_{avg}(N) = 1/m \sum_{i=1}^{m} T_i$$

أسئلة تقويم ذاتي

1) لماذا تعتبر الحالة الأسوء حالة هامة جداً في تحليل الخوارزميات؟
2) لماذا لا يتم التعامل كثيراً مع الحالة الأفضل في تحليل الخوارزميات؟

3. تحليل الخوارزميات التكرارية

Analysis of Expetitive Algorithms

والآن سوف نقوم أولاً بكتابة خطة عامة لتحليل الخوارزميات التكرارية ومن ثم شرح بعض الأمثلة لتطبيق الخطة.

1.3 نموذج لتحليل الخوارزميات التكرارية

1. حدد معلم (او معالم) تدل على حجم المدخلات.
2. عين العمليات الأساسية والجمل الأساسية في الخوارزمية.
3. أفحص عدد مرات تنفيذ العمليات الأساسية لمعرفة ما إذا كان يعتمد على حجم المدخلات فقط أم يعتمد على عامل إضافي آخر. في حالة اعتماده على خاصية أخري يجب التعامل مع الحالات السيئة والوسطى والأفضل بصورة منفصلة.
4. عين المجموع الذي يعبر عن عدد مرات استخدام العمليات الأساسية داخل الجمل الأساسية لخوارزمية مستخدماً القواعد الأساسية لذلك.

استخدم القواعد القياسية للمتسلسلات لإيجاد الصيغة المنطقية للمجموع ومن ثم عيّن معدل النمو المماثل له.

وهنالك بعض القواعد الهامة الملازمة لهذه الخطة والتي يجب التعامل معها بحرص وهي:

1.1.3 قاعدة الحلقات loops

يحسب المعدل عند الحلقات على النحو الآتي:

المعدل الكلي = حجم الحلقة x المعدل داخل الحلقة.

مثلاً الحلقة أدناه المعدل الزمني المطلوب لأنجازها هو O(N) :

```
for  i = 1   to  N   do
     Begin
          sum = sum +i
       end for
```

2.1.3 قاعدة الحلقات المتداخلة Nested Loops

يحسب المعدل عند الحلقات المتداخلة كالآتي:

المعدل الكلي = مضروب حجم الحلقات * المعدل داخل الحلقات

مثلاً الحلقتان المتداخلتان أدناه لهما معدل $O(N^2)$:

```
for i = 1   to  N  do
   for j = 1  to  N  do
          Sum = sum + i
  end for j
end for i
```

3.1.3 قاعدة المتتابعات Sequences

يحسب المعدل للمتتابعات على النحو الآتي:

المعدل الكلي = أعلى معدل زمني في المتتابعات

(راجع نظرية (1) في الوحدة الثانية)

مثلاً: خذ المتتابعتين أدناه، سوف تجد أن المعدل الكلي هو الأعلى وهو $O(N^2)$:

```
for i = 1  to  N   do
              C(i, j) = 0
end for i

for i = 1  to  N   do
    for j = 1  to  N  do
          C(i,j) = C(i,j) + A(i,j)
     end for i
end fo j
```

4.1.3 قاعدة الاختيار Selection

يحسب المعدل الزمني عند الجمل الشرطية والتي تحقق غرض الاختيار والتـــي يتكون على الصيغة:

```
IF (C)   THEN
        S1
ELSE
         S2
```

بالعلاقة التالية:

المعدل الكلي ≥ المعدل الزمني للشرط (C) + أعلى معدل من الجملتين S1 و S2 .

(في الحالة السيئة يصل هذا المعدل إلي أعلى حد ممكن).

مثال (1) معدل خطي

خذ مسالة إيجاد القيمة الكبرى في قائمة مـــن N عنصـــر. الخوارزميـــة أدنـــاه هـــي خوارزمية قياسية لحل هذه المسألة:

خوارزمية (1)

```
max {A(1..N)}
    // Determines the maximum value
    // Input:  Average A{1..N} of integer
    // Output:  The maximum value
    Max = A [1]
    for i = 2  to  N   do
        if A[i] > max  then
                max = A[i]
        end if
    end for
return max
```

من هذا المثال:

من الواضح أن مقياس حجم المدخلات هو عددها N، ومن الواضح أيضاً أن هنالك عمليتين يتم استخدامها داخل حلقة (for loop) . والعمليتان هما : عملية المقارنة A [i] > max وعملية التعيين max = A[i] . ما هي العملية الأساسية الهامة في هذه الخوارزمية ؟

من الواضح أن عملية المقارنة هي العملية الأساسية لأنها تستخدم في كل مرة يتم تنفيذ الحلقة بينما نجد عملية التعين يتم استخدامها وأحياناً. أيضاً هنالك ملاحظة هامة وهي ان عملية المقارنة هي ضرورية لكل عناصر القائمة ، أي أن حجم المدخلات هو العامل الأساسي، ولذا فإن جميع الحالات السيئة والوسطى والأفضل يتم التعامل معها بصورة واحدة.

ضع المقياس T(N) بحيث يمثل عدد المرات التي تستخدم فيها تلك المقارنات داخل الخوارزمية. سوف نقوم بمحاولة وضع صيغة عامة لحساب هذا المقياس بدلالة N. من الملاحظ أن الخوارزمية تقوم بمقارنة قيمة واحدة لكل دوره من دورات الحلقة (for) بحيث تحتاج ذلك من (1) وحتى (N-1) . إذا يمكن كتابة صيغة للتعبير عن T(N) على النحو :

$$T(n) = \sum_{i=1}^{N-1} 1$$
$$= N - 1$$

ومن الواضح أن الوصول للمعدل الزمني لا يحتاج إلى تحويل هذا المجموع إلي صيغة مغلفة ، حيث يمكن إيجاد المعدل الزمني بسهولة كالآتي:

$$T(N) = N-1 = \theta(N)$$

وبالطبع فإن هذا المعدل هو معدل خطي يعتمد أساساً على حجم المدخلات N.

مثال (2) معدل تربيعي

خذ مسألة الفردية وهى فحص ما إذا كان جميع العناصر الموجودة في قائمة من N هي عناصر فريدة أي أنَّ قيمتها لا تتشابه. هنالك خوارزمية سريعة على النحو الآتي:

خوارزمية (2)

```
Unique (A [1..N])_
// Checks the uniqueness of N elements
// Input: an error A [1..N].
// Output: returns "true" if all elements
          Are distinct and "false"
          Other wise.
for i = 1  to  N-1   do
    for j =  i+1   to  N   do
          if A[I] = A[j]  then
                return false
          end if
     end for j
end for i
return true
```

أيضاً يتضح هنا أن عدد عناصر القائمة هو المقياس الأساسي لحجم المدخلات، كما أن العملية الأساسية هي عملية المقارنة A[i] = A[j] والتي تتم داخل الحلقة الداخلية . ملاحظة هامة أخرى وهي أن عدد المقارنات المطلوب لا يعتمد فقط على حجم المدخلات N ولكن يعتمد أيضاً على وجود عنصرين متشابهين في القائمة، حيث يتم عندها التوقف عن المقارنة. لذا سوف نحاول إيجاد المعدل الزمني للحالة السيئة فقط ويمكن التعبير عنه بواسطة T(N) . من المعروف أن الحالة السيئة هي الحالة التي تعطي فيها القائمة أكبر عدد مقارنات T(N) من جميع الحالات الأخرى والتي حجم

المدخلات فيها N، ولذا يمكننا الوصول إلي نوعين فقط للحالات السيئة للمدخلات: الحالة الأولى والتي لا يوجد فيها عنصران متشابهان والحالة الثانية وهي الحالة التي يكون فيها العنصران الآخران فقط هما العنصرين المتشابهين. ففي الحالتين يتم استخدام مقارنة واحدة داخل الحلقة الداخلية أي لكل قيمة للمتغير زمن وحتى N، كما يتم أيضاً تكرار هذا بالنسبة للحلقة الخارجية أي للمتغير i من 1 وحتى N-1 . لذا يمكن صياغة المجموع كالآتي:

$$T(N) = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} 1$$

$$= \sum_{i=1}^{N-1} [N-(i+1)+1]$$

$$= \sum_{i=1}^{N-1} [N-i] = \sum_{i=1}^{N-1} N - \sum_{i=1}^{N-1} i$$

$$= N\sum_{i=1}^{N-1} i - \frac{(N-1)N}{2}$$

$$= N(N-1) - ½ \text{ N} \quad \text{(N-1)}$$

$$½\, N(N-1) = \theta\ (\text{N}^2)$$

وهذا المعدل التربيعي متوقع حيث إن الخوارزمية تحتاج في أسوأ حالاتها لمقارنة ½N(N-1) زوج ضمن N عنصر.

نظرية (1) كيفية حساب المعدل اللوغريثمي

أي خوارزمية تأخذ معدل لوغريثمي $O(\log_2 N)$

إذا كانت تحتاج معدلاً زمنياً ثابتاً $O(1)$ لتقسيم حجم المسألة إلي نسبة معينة في كل مره، وغالباً ما تكون نسبة تقارب النصف.

ومن الآن فصاعداً فسوف نكتب $\log N$ للدلالة على $\log_2 N$ في جميع الأمثلة القادمة.

مثال (3) معدل لوغريثمي

خذ خوارزمية أقليدس (Eculide) لحساب القاسم المشترك الأعظم (Greatest Common Disisor) ، والتي تعتمد أساساً على تكرار العلاقة التالية:

$$GCD(M,N) = GCD(N, M \bmod N)$$

حيث $M \bmod N$ هو خارج قسمة M على N ، ويتم تكرار هذه العملية حتى يكون خارج القسمة مساوياً الصفر، حيث يكون $GCD(r,o)=r$ وعندها يكون r هو نفس القاسم المشترك الاعظم لـ M و N . مثلاً خذ $GCD(60,24)$ نجد أن:

$$GCD(60,25)=GCD(25,10)=GCD(10,5) = GCD(5,0)=5$$

ويمكن صياغة خوارزمية أقليدس على النحو:

خوارزمية (3)

```
GCD(M,N)
// computes GCD(M,N) by Euclidi's Alg.
// Input:  Two non negative integers M and N.
// Output: Greatest common division of M and
N.

While  (N≠0)   d0
      Begin
            R=M  mod  N
            M = N
            N = R
       end While
return M
```

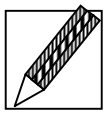
فمثلاً خذ الحالة السابقة لحساب GCD(60,25) كالاتى:

1) M=60, N=25 Then R = 10
2) M=25, N=10 Then R = 5
3) M=10, N=5 Then R = 0
4) M=5, N=0 Then (Stop) GCD = 5

من الواضح في هذه الخوارزمية أن العملية الأساسية والهامة ليست داخل الدارة(While) ولكنها هي عملية المقارنة (N≠0) والتي تحدد نهاية دوران الدارة (While) عندما تصبح N معدومة أي مساوية للصفر، وهنالك ملاحظة أخرى وهي المعدل الزمني داخل الدارة هو معدل ثابت لثلاث جمل أي O(1)، كما أن الخوارزمية بعد دورتين يكون لها خاصية تقسيم حجم المسألة بنسبة ثابته حيث إن (m mod n) هو أصغر من n ½ . وعليه وحسب النظرية 4-1، نجد أن المعدل الزمني لهذه الخوارزمية يصبح كالآتي:

$$T(N) = O(\log N)$$

تدريب (1)



(1) اكتب خوارزمية لإيجاد الوسيط بين ثلاثة أعداد صحيحة مختلفة (فريدة). تقع مدخلات هذه الخوارزميَّة في ثلاثة مجموعات مختلفة ماهى؟ وما هي الحالة السيئة؟ وما هي الحالة الوسطى ثم ما هي الحالة الأفضل؟

(2) الخوارزميتان B,A تم تحليلهما وأعطتا معدلاً زمنياً للحالة السيئة لا يزيد عن $15n \log_2 n$ و n^2 على التوالي، أجب عن الأسئلة التالية:

أ- ماهى الخوارزمية الأضمن لزمن التنفيذ لقيم n الكبرى ($n > 10000$) ولماذا؟

ب- ما هي الخوارزمية الأضمن لزمن التنفيذ عند قيم n الصغرى ($n < 1000$)S؟

ج- ما هي الخوارزمية الأضمن عند متوسط زمني للتنفيذ $n=1000$ ؟ ولماذا؟

(3) خذ الخوارزمية التالية:

```
Mystery (N)
S=0
for i = 1 to N do
    s = s +i *i
end for i
    return s
```

(أ) ما هو الشئ الذي تقوم بحسابه الخوارزمية؟

(ب)ما هي العمليات الأساسية؟

(ج) كم هو عدد المرات التي يتم فيها استخدام العمليات الأساسية؟

(د) ما هو المعدل الزمني لهذه الخوارزمية؟

(4) خذ الخوارزمية التالية:

```
Secret (A [0.. N-1])
min = A [0]   ,    Max  = A [0]
for i = 1   to   n-1    do
    if  A [i]  <  min       then
                min  =  A [i]
    if  A [i]  >  max     then
end for
return (max –min)
```

أجب عن الأسئلة من (أ) وحتى (د) كما في السؤال (3)

(5) خذ الخوارزمية التالية:

```
Matrix (A [0..N-1,  0..N-1])
for  i = 0  to  N-2  do
     for j = i+ 1   to    N-1     do
                    if  A [i , j]   ≠ A [s , i]
                        return   false
     end for  j
end for   i
return  true
```

أجب عن الأسئلة من (أ) وحتى (د) كما في السؤال (3).

نشاط

لكل خوارزمية جزئية أدناه :

(i) عين المعدل للزمن O الكبيرة.

(ii) نفذ الشفرة بأي لغة برمجة وأحسب المعدل من البرنامج لعـــدة قيم لـــ n .

(iii) قارن نتيجة تحليلك مع المعدل الفعلي بعد التنفيذ.

1)

```
Sum =0
 for i =1  to n do
        Sum = sum +1
              end for
```

2)

```
sum =0
 for i =1 to n do
      for j =1 to n**2 do
       sum = sum+1
        end for j
end for I
```

3)

```
sum =0
for i = 1 to n do
    for j = 1 to i**2 do
        for k=1 to j do
              sum =
sum+1
       end fo k
     end fo j
end for i
```

4. تحليل خوارزميات الاستدعاء الذاتي
Analysis of Excursive Algorithms

الآن سوف نبدأ الانتقال لخوارزميات الاستدعاء الذاتي ومعرفة كيفية التحليل الزمني لها. ولذا سوف نقوم بتوضيح خطة عامة لتحليل هذه الخوارزميات ومن ثم نشرح بعض الأمثلة في تطبيقها.

1.4 نموذج لتحليل خوارزميات الاستدعاء الذاتي

1. حدد معلم (أو معالم) تدل على حجم المدخلات.
2. عين العمليات الأساسية للخوارزمية.
3. افحص عدد مرات استخدام العمليات الأساسية – هل يختلف عند المدخلات المختلفة ذات الحجم N ، إذا كان ذلك كذلك فإن المسألة تمر بحالات مختلفة من الأفضل وإلي الأسوأ.
4. عين علاقة الاستدعاء الذاتي وحدد الحالة الابتدائية لها لعدد مرات استخدام العمليات الأساسية.
5. حل علاقة الاستدعاء الذاتي وحول العلاقة إلي صيغتها المغلقة ومن ثم عين المعدل الزمني المقابل لها.

مثال (4) معدل خطي

خذ خوارزمية إيجاد مضروب العدد N (N!) ، حيث :

$$N! = N (N-1) !, \quad o! = 1$$

خوارزمية(4)

```
F(N)
// Computes:  N!  recursively
// input:  A non negative N
// output:  The value of N!
        if (N=0)   return  1
          else  return  F(N-1)*N
```

من المعلوم أن العدد N هو الدليل عن حجم المدخلات وأن العملية الأساسية هي عملية الضرب F(N-1)*N ، وعدد مرات استخدامها هو T(N)، ولذا من تعريف المضروب بالصيغة.

F(N) = F(N-1)*N, N >0 يمكننا صياغة علاقة الاستدعاء الذاتي على النحو:

T(N) = T(N-1) + 1 , T(0) = 0

حيث T(N-1) هو عدد المرات للحصول على F(N-1) والواحد هو تكلفة الضرب F(N-1)*N و T(0)=0 هو الحالة الابتدائية عند N=0 حيث لا يكون هنالك داعي لعملية الضرب وليس هناك حاجة للحصول على F(N-1) والآن يمكننا استنتاج الإجابة بسهولة عن الصيغة المغلقة للمقياس T(N) . وهنالك طرق مختلفة للوصول لهذه الصيغة ومنها طريقة التعويض الخلفي على النحو الآتي:

$$T(N) = T(N-1) + 1$$

وبتعويض $T(N-1) = T(N-2) + 1$

ينتج الآتي:

$$T(N) = T(N-1) + 1 = (T(N-2) + 1) + 1$$
$$= T(N-2) + 2$$

وهكذا يمكننا الوصول للصيغة التالية:

$$T(N) = T(N-1) +1 = \ldots = T(N-r) +r$$
$$= \ldots = T(N-N) + N = N$$

ولهذا فإن المعدل المطلوب هو $\theta(N)$

مثال (5) معدل لوغريثمي

خذ مسألة الحصول على عدد الخانات الثنائية في التمثيل الثنائي للعدد الصحيح الموجب N . هنالك خوارزمية استدعاء ذاتي للحصول على الإجابة كالآتي:

خوارزمية (5)

```
Bin (N)
// Input : Positive integer N
// output: Thr number of binary
          digits in N's  bibary representation.

if n = 1  return 1
else return Bin (N/2) + 1
```

لتعيين علاقة الاستدعاء الذاتي والحالة الابتدائية، ضع T(N) للتعبير عن عـــدد المرات والحالة الابتدائية لاستخدام عملية الجمع، ولذا يمكننا كتابة T(N) على النحو:

$$T(N) = T(N/2) + 1 \ , \ N >1$$

ويمكننا الوصول للحالة الابتدائية $T(N) = 0$

حيث إن الخوارزمية عندها لا تحتاج لعملية إضافية ولا تحتـــاج لتكـــرار Bin (N/2) لمرة أخرى.

في هذه الخوارزمية سوف نقوم بحل علاقة الاستدعاء الذاتي باستخدام قاعدة هامـــة في تحليل الخوارزميات والتي تنص بأن معدلات النمو عـــن $N=2^k$ تعطـــي معـــدلات صحيحة للقيم N . لذا عند تطبيق هذه القاعدة يمكننا صياغة العلاقة على النحو:

$$T(2^k) = T(2^{k-1}) + 1, \quad k > 0$$
$$T(2^0) = 0$$

ولذا يمكننا استخدام التعويض الخلفي بدون مشاكل:

$$T(2^k) = T(2^{k-1}) + 1$$
$$= (T(2^{k-2}) + 1) + 1 = T(2^{k-2}) + 2$$
$$= (T(2^{k-3}) + 1) + 2 = T(2^{k-3}) + 3$$
$$= \ldots = T(2^{k-r}) + r$$
$$= \ldots = T(2^{k-k}) + k$$
$$= T(1) + k = K$$

وللرجوع للأصل نجد أن $N = 2k$

أي أن $k = \log_2 N$ ولذا يمكننا تحديد المعدل الزمني لهذه الخوارزمية كالآتي:

$$T(N) = O(k) = O(\log_2 N)$$

مثال (6) معدل أسي

خذ خوارزمية أرقام فايبونسي والتي تعطي أرقام على النحوالآتي:

0,1,1,2,3,5,8,13,21,...

والتي يمكننا التعبير عنها بالعلاقة:

$$F(N) = F(N-1) + F(N-2) \quad \text{for } N > 1, \text{ and}$$
$$F(0) = 0 \, , \, F(1) = 1$$

خوارزمية (6)

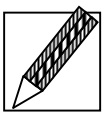
```
F(N)
// Compute the Nth Fibonacci number.
// input: A non negative integer N.
// output : The Nth Fibonacci number.
if  N ≤ 1      return   N
else   return  F(N-1) + F(N-2)
```

هنا العملية الأساسية هي عملية الإضافة ، ضع T(N) هو المعـــدل الزمنـــي أي عدد مرات استخدام هذه العملية للحصول على الرقم المطلوب لعـــدد الصـــحيح F(N) . لذا يمكننا الوصول لعلاقة الاستدعاء الذاتي على النحو:

$$T(N) = T(N-1) + T(N-2) + 1 \quad \text{for } N>1$$
$$T(0) = 0 \, , \, T(1) = 0$$

لاحظ أن $T(N) > F(N)$ ، ولذا يمكننا استخدام خاصيتين لأرقام فايبونسي هما $F(N) < (5/3)^N$ و $F(N) \geq (2/3)^N$ دالتان يمكننا إثباتهـــا عـــن طريـــق الإثبـــات بالاستقراء (حاول ذلك بنفسك)، ولذا يمكننا أن نصل إلي معدل أسي لهـــذه الخوارزميـــة وهو $T(N) = O(2^N)$ ولذا تصبح الخوارزمية ذات كفاءة سيئة ويمكننا تحسين كفاءتها بتصميم خوارزمية برمجة فعالة كما سنعرض لاحقاً في هذا المقرر.

تدريب (2)



(1) خذ الخوارزمية أدناه والتي تقوم بحساب المجموع

$sum = 1^3 + 2^3 + 3^3 + \ldots + N^3$

عن طريق الاستدعاء الذاتي في كل مرة:

```
Sum (N)
    if  N = 1    return  1
    else  return  sum (N-1)  + N*N*N
```

(أ) عين علاقة الاستدعاء الذاتي لعدد مرات استخدام العمليات الأساسية في الخوارزمية وقم بحلها وحدد المعدل الزمني للخوارزمية.

(ب) كيف يمكنك أن تقارن هذه الخوارزمية مع الخوارزمية التكرارية لنفس الغرض؟

(2) خذ خوارزمية الاستدعاء الذاتي التالية:

```
Q (N)
if  N = 1   return  1
else  return   Q (N-1) + 2*N-1
```

(أ) عين علاقة الاستدعاء الذاتي وقم بحلها وحدد المعدل الزمني المقابل لها.

(ب) حاول كتابة خوارزمية تكرارية تقوم بنفس الغرض الذي تقوم به هذه الخوارزمية.

(3) خذ خوارزمية الاستدعاء الذاتي التالية:

```
Min Arrag  (A [0.. N-1])
 If   N = 1    return    A [0]
else  tmp= Min Arrag  (A [0..N-2])
        if tom ≤  A [N-1]  then
                    return  tmp
else return A[N-1]
```

(أ) ما هو الغرض من الخوارزمية؟

(ب) عين علاقة الاستدعاء الذاتي وحدد المعدل الزمني للخوارزمية.

نشاط

اكتب خوارزمية استدعاء ذاتي لحساب 2^n للعدد الصحيح الموجب n والتي تعتمد على الصيغة:

$$2^n = 2^{n-1} + 2^{n-1}$$

(أ) عين علاقة الاستدعاء الذاتي لعدد مرات الإضافة وقم بحلها وحدد المعدل الزمني للخوارزمية.

(ب) هل هذه الخوارزمية هي خوارزمية جيدة لحل هذه المسألة ؟ لماذا ؟

الخلاصة

عزيزي الدراس،

- في هذه الوحدة تعرفنا على تحليل الخوارزميات المختلفة ومدى كفاءتها من حيث زمن التنفيذ اللازم، وذكرنا في قسمنا الأول أن الخوارزميات تنقسم إلى قسمين؛ خوارزميات تكرارية وخوارزميات الإستدعاء الذاتي.

ثم وضحنا خطوتين للأشياء التي يجب تحديدها عند التحليل الأولى وهي تحديد العمليات الهامة والثانية هي تعيين العمليات المتكاملة والمساعدة للخوارزمية.

كما شرحنا في قسمنا الثاني أنه يجب تحديد طبيعة المدخلات عند التحليل وقسمنا دالات المدخلات إلى ثلاث حالات :

1) الحالة الآفضل Best Case تكون المدخلات على صيغة تتطلب أقل زمن في تنفيذ الخوارزمية.
2) الحالة السيئة Worst Case تتطلب تحديد صيغة للمدخلات التي تتسبب في أن الخوارزمية تقوم بجميع الأعمال وفي أطول زمن ممكن.
3) الحالة الوسطى Average Case وتتطلب حساب عدد المجموعات واحتمال ورودها ثم حساب المعدل الزمني للخوارزمية.

ثم بدأنا في قسمنا الثالث بالنوع الأول من الخوارزميات وهو الخوارزميات التكرارية ووضعنا نموذج لتحليلها وبعض القواعد الهامة والأمثلة والتدريبات والنشاطات .

وأخيراً في قسمنا الرابع وضعنا نموذجاً لتحليل النوع الثاني من الخوارزميات وهي خوارزميات الاستدعاء الذاتي ومثلنا ببعض الخوارزميات ثم وضعنا التدريبات المناسبة والنشاطات.

لمحة مسبقة عن الوحدة التالية

وحدتنا التالية عزيزي الدراس، تعالج خوارزميات الترتيب وهي الخوارزميات التي تستخدم في ترتيب العناصر حسب قيمة المفتاح المعين، أرجو أن تجدها وحدة مليئة بالموضوعات والأمثلة والتدريبات المفيدة – وفقك الله.

مسرد المصطلحات

- **نموذج تحليليAnalysis Model**
- **خوارزمية تكراريةRepetitive Alogrithm**

هي خوارزمية تحتوي على جمل دورانية (Loops) وجمل شرطية.

- **خوارزمية استدعاء ذاتيRecursive Alogrithm**

هي خوارزمية تقسم المسألة إلي عدد من الأقسام الفرعية.

- **تحليل الخوارزميةAlogrithm Anlaysis**

يعني اعطاء المعلومات الخاصة بالزمن الذي تستغرقه الخوارزمية و المساحة المطلوبة لإنجازها.

- **الحالة السيئةWorse Case**

هي الحالة التي تتطلب منا تحديد صيغة المدخلات التي تتسبب في أن الخوارزمية تقوم بجميع الأعمال في أطول وقت ممكن.

- **الحالة الوسطىAverage Case**

تقوم بحساب المعدل الزمني للخوارزمية لكل مجموعات المدخلات.

- **الحالة الأفضلBest Case**

هي الحالة التي تتطلب أن تاخذ الخوارزمية أقل زمن ممكن.

- **المعدل الزمنيTime Rate**
- **التحليل الزمني للخوارزميةAlgorithm Time Analysis**

المراجع

المراجع العربية

1) مجموعة مؤلفين، تركيب البيانات وتصميم الخوارزميات، منشورات جامعة القدس المفتوحة، 1998.

2) السماني عبد المطلب، هياكل البيانات، منشورات جامعة السودان المفتوحة، 2005.

المراجع الأجنبية

1) Weiss, M.A., Data Structures and Alogrithm Analysis, Benjamin pub., 1992.
2) McConnell, J.J., Analysis of Alogrithms: An Active Approach, Jones pub., 2001.
3) Lavitin, A., Introduction to the Design and Analysis of Alogrithms, Addisions ues by, 2003.
4) Ulman, J.D (etal), The Design and Analysis of Compurte Alogrithms, Addison Wesley, 1974.
5) Basese, S., Compurte Alogrithms : Introduction to Analysis and Design, Addison Wesley, 2000.
6) Sedgewick, R., An Introduction to the Analysis of Alogrithms, Addison Wesley 1996.


الوحدة الرابعة
خوارزميات الترتيب
Sorting Algorithms

محتويات الوحدة

المقدمة

تمهيد

عزيزي الداري: مرحباً بك في الوحدة الرابعة من هذا الكتاب والتي تأتيك بعنوان خوارزميات الترتيب. ونعني بخوارزميات الترتيب الخوارزميات التي تستخدم في ترتيب العناصر حسب قيمة مفتاح معين، الترتيب إما يكتب تصاعدياً أو تنازلياً لقيمة المفتاح. وسوف نتناول في هذه الوحدة ثلاثة خوارزميات هامة، وهي خوارزمية الترتيب بالإدخال وخوارزمية الترتيب بالتجزئة وخوارزمية الترتيب السريع. وهذه الخوارزميات تختلف عن بعضها البعض من حيث الكفاءة، فبعضها أكثر ملاءمة من الآخر لعدد قليل من البيانات بينما البعض الآخر أكثر ملاءمة عندما يكون حجم البيانات المراد ترتيبها كبيراً. وفي هذه الوحدة قمنا بتوضيح الأمثلة والتدريبات المناسبة لموضوع الوحدة. ولتدعيم المادة والمفاهيم الواردة في هذه الوحدة يمكنك تطبيق الأمثلة والتدريبات على الحاسوب بكتابة برامج لخوارزميات الترتيب وتنفيذها على أجهزة الحاسوب المتوفرة لديك.

تتكون هذه الوحدة من خمسة أقسام رئيسة. في القسم الأول نقدم لك تعريفاً لترتيب البيانات في الحاسوب ونشرح لك أهميته. وفي القسم الثاني سوف نقوم بشرح كيفية تحليل خوارزميات الترتيب المختلفة. أما في القسم الثالث فسنتناول خوارزمية الترتيب الأولى في هذه الوحدة وهي خوارزمية الترتيب بالإدخال. وفي القسم الرابع سوف نقوم بشرح الخوارزمية الثانية وهي خوارزمية الترتيب بالتجزئة، ويأتي شرح الخوارزمية الثالثة وهي خوارزمية الترتيب السريع في القسم الخامس والأخير من هذه الوحدة.

أهداف الوحدة

عزيزي الدارس،

ينتظر منك بعد الانتهاء من دراسة هذه الوحدة أن تكـــون قـــادراً على أن:

- ✓ **توضح** أهمية الترتيب عند معالجة البيانات.
- ✓ **تميز** طرق الترتيب المختلفة عن بعضها البعض.
- ✓ **تقوم** ببناء الخوارزميات والبرامج اللازمة بترتيب البيانات.
- ✓ **تحلل** خوارزميات الترتيب المختلفة وتقارن فيما بينها.

1. مقدمة عن الترتيب البياني

لا شك أننا نواجه في حياتنا اليومية كثيراً من الأمور التي تحتاج لحفظ المعلومات واسترجاعها، وهذا يقتضي حفظ المعلومات أو البيانات وفق ترتيب معين كي تسهل علينا عملية البحث عنها، وتعتبر عمليات الترتيب والبحث من العمليات الأساسية في علم الحاسوب، وقد تم تطوير عديد من الطرق الخاصة بترتيب البيانات داخل الحاسوب، ونقصد بالترتيب البياني عملية ترتيب البيانات قيد البحث، حسب ترتيب معين: إما تصاعدياً أو تنازلياً، وفقاً لقيم عددية أو رمزية، كترتيب قائمة طلاب حسب أرقامهم، أو معدلاتهم أو أسمائهم.

تبرز أهمية الترتيب عند معالجة مجموعة كبيرة من السجلات، ونستطيع أن نستخدم خوارزميات بحث فعالة وسريعة إذا كانت قوائم السجلات مرتبة، ولا نستطيع ذلك فيما إذا كانت قوائم السجلات غير مرتبة. ولهذا فإننا نحتاج لعملية الترتيب في جميع الأعمال التي تتطلب الاحتفاظ بالبيانات وسرعة الوصول إليها. وتتطلب عملية الترتيب إنشاء خوارزميات ملائمة تؤثر على أدائها عدة عوامل منها:

أ. الوقت الذي يحتاجه المبرمج لكتابة برنامج ينفذ الخوارزمية بلغة برمجة معينة.
ب. المدة التي يستغرقها تنفيذ البرنامج.
ج. مساحة الذاكرة التي تحتاجها الخوارزمية لتنفيذ الطريقة التي تتبعها في ترتيب البيانات.
د. خواص البيانات المطلوب ترتيبها مثل عددها ونوعها.

ولذا فإن عدد العناصر المراد ترتيبها يؤثر بشكل مباشر على عملية اختيار الخوارزمية المناسبة في ترتيب البيانات. فإذا كان عدد العناصر قليل 50-100 على سبيل المثال) فلا داعي للبحث عن خوارزمية ذات كفاءة عالية تحتاج إلى جهد أكبر من المبرمج لكتابتها و تنفيذها على الحاسوب، أما إذا كان عدد العناصر المراد ترتيبها كبيراً

جداً، فإنه من المنطقي البحث عن خوارزمية ترتيب تستغرق وقتاً أقل عند تنفيذها على الحاسوب.

إذن يمكن القول بأن هنالك فائدة كبيرة ومهمة للترتيب البياني في علم الحاسوب، وخاصة في المساعدة في إنجاز عمليات البحث بفعالية عالية، بالإضافة إلى أن البيانات المرتبة أفضل للعرض على المستخدم، فتصور أن برنامجاً لحساب درجات الطلاب في مؤسسة أكاديمية يعطينا قائمة أسماء الطلاب ودرجاتهم بدون ترتيب، ولا شك أننا سوف نفضل برنامجاً يقوم بعرض قائمة الطلاب مرتبة حسب الترتيب الأبجدي لأسمائهم أو حسب ترتيب درجاتهم التي تحصلوا عليها.

2. التحليل الزمني لخوارزميات الترتيب

في هذه الوحدة سوف نقوم بتوضيح ثلاثة أنواع مختلفة من خوارزميات الترتيب؛ الخوارزمية الأولى هي الترتيب بالإدخال والتي تتلخص طريقتها في إدخال أي عنصر جديد في المكان المناسب له ضمن القائمة التي سبق ترتيبها. أما الترتيب بالتجزئة ويطلق عليه أحياناً ترتيب شل نسبة للعالم الذي اكتشفه (D.Shell) فتتلخص طريقته بتنفيذ ترتيب القائمة في عدة مراحل في كل مرحلة يتم تجزئة القائمة لقوائم جزئية بحيث يتم ترتيبها بالإدخال، وفي كل مرحلة قادمة يتم تقليص عدد المجموعات الجزئية وزيادة حجم العناصر فيها، بينما الترتيب السريع هو ترتيب استدعاء ذاتي يتم فيه اختيار عنصر ارتكاز من عناصر القائمة ثم يتم تقسيم عناصر القائمة إلى قسمين أدنى وأعلى من عنصر الارتكاز، وعند توضيح كل خوارزمية سوف نقوم بوصف الطريقة المتبعة أو الفكرة الرئيسية بها ثم نوضح الخوارزمية وإعطاء مثال توضيحي ونختم بتحليل الخوارزيمة ومعرفة المجموعات التي تتبع لها وفقاً للحالات السيئة والوسطى.

1.2 الحد الأدنى لخوارزمية الترتيب البسيطة

لحساب الحد الأدنى لمعدلات نمو خوارزميات الترتيب، نحتاج إلى أن نضع التعريف التالي:

تعريف (1)

المعكوس في قائمة من الأعداد هو الزوج (i , j) والذي يحمل خاصية:

$$a[i] > a[j] \quad \text{عند} \quad i<j$$

مثال (1)

بين المعكوسات في القائمة التالية:

21 32 51 64 8 34

في هذه القائمة نجد هنالك 9 معكوسات مختلفة وهي (1,2) ، (1,5) ، (1,6) ، (3,4) ، (3,5) ، (3,6) ، (4,5) ، (4,6) ، (5,6) .

ويمكننا أن نلاحظ أن هذه المعكوسات هي نفسها عدد التبديلات المطلوبة عند ترتيب هذه القائمة بأي خوارزمية ترتيب. ولذا يمكننا أن نثبت النظريات التالية والتي تفيدنا كثيراً في تقدير الحد الأدنى:

نظرية (1)

متوسط عدد المعكوسات في أي قائمة من n عدد مفرد هو

$$¼ \; n(n-1)$$

وهذه النظرية تقتضي أن أبسط خوارزمية ترتيب تتبع للمعدلات التربيعية في المتوسط. وأيضاً بواسطتها يمكننا حساب الحد الأدنى لأي خوارزمية تقوم بمقارنة العناصر المتجاورة للوصول لترتيب القائمة.

نظرية (2)

أي خوارزمية ترتيب تقوم بعملية الترتيب بتبديل العناصر المتجاورة في القائمة تحتاج في بالمتوسط إلى معدل زمني $\Omega(n^2)$

هذه النظرية توضح الحد الأدني المطلوب لأي خوارزمية تتبع لصنف الخوارزميات التي تتطلب مقارنة العناصر المتجاورة، ولذا يمكننا أن نصل إلى النتيجة التالية:

نتيجة (1)

لأي خوارزمية ترتيب وللحصول على معدل أقل من المعدل التربيعي أو $O(n^2)$ في المتوسط يجب أن نبتدع طريقة لمقارنة وتبديل العناصر البعيدة عن بعضها البعض أي تقليل عدد المعكوسات أثناء عملية الترتيب.

أسئلة تقويم ذاتي

1. ما هي أهمية الترتيب البياني في علم الحاسوب؟
2. كيف تحصلنا على النتيجة (1) أعلاه؟
3. متي تبرز أهمية الترتيب؟

3. الترتيب بالإدخال (Insertion Sort)

1.3 الطريقة

طريقة الترتيب بالإدخال من أبسط طرق الترتيب وتتلخص في الآتي:

خلال (n-1) مرحلة تضمن خوارزمية الترتيب وللمؤشر p حيث p يبدأ من العنصر الثاني (p =2) وينتهي بالعنصر الأخير في القائمة (p=n)تضمن بأن عناصر القائمة المراد ترتيبها من المواقع 1 وحتى p في وضع مرتب. تقوم خوارزمية الترتيب على

حقيقة أن العناصر من الموقع 1 وحتى الموقع (p-1) هي عناصر مرتبة مسبقاً أي بمعنى تم ترتيبها خلال المرحلة السابقة من عملية الترتيب. وهذه الطريقة تقوم على أستراتيجية أن العنصر رقم p المتبقي وفي أثناء المرحلة ، يمكن نقله إلى الموقع الصحيح له ضمن أول p عنصر في القائمة.

2.3 الخوارزمية

من الطريقة السابقة للترتيب بالإدخال يمكن صياغة الخوارزمية التالية:

خوارزمية (1)

```
Insertion Sort (a,n)
{1}        a[0] =    Min – Data
{2}        for p=2   to    n    do
               begin
{3}                j = p  ,  tmp = a[p]
{4}                while   tmp  <  a[j-1]  do
                        begin
{5}                          a[j] = a [j-1]
{6}                          j = j-1
                        end while
{7}            a [j] = tmp
               end for
```

في هذه الخوارزمية العنصر a(0) هو قيمة صغرى نحتاجها لإيقاف حلقة (while) . الخطوات من {4} وحتى {7} تقوم بتنفيذ تبديل العناصر مع بعضها البعض. العنصر في الموقع p يتم حفظه في المتغير tmp ، وجميع العناصر التي تفوقه في القيمة يتم تحويلها إلى موقع إلى اليمين. ثم يتم وضع العنصر tmp في موقعه الصحيح ضمن القائمة الأولى من العنصر الأول وحتى العنصر رقم p .

مثال (2) قم بترتيب القائمة التالية بواسطة الترتيب بالإدخال :

34 8 64 51 32 21

لحل هذه المسألة سوف نقوم باتباع طريقة معينة موضحة في الجدول (1) أدنـــاه، حيث أنها تشتمل على توضيح المرحلة (p) والعنصر tmp بمربع وحساب عدد المقارنات المطلوبة والتبديلات اللازمة لنقل العنصر tmp لموقعه الصحيح.

المرحلة	القائمة المراد ترتيبها						المقارنات	التبديلات
P	21	32	51	64	8	32	C	S
2:	–	–	–	–	8	34	1	0
	–	–	–	–	34	8	0	1
3:	–	–	–	64	34	8	1	0
4:	–	–	51	64	34	8	1	0
	–	–	64	51	34	8	1	1
5:	–	32	64	51	34	8	1	0
	–	64	32	51	34	8	1	1
	–	64	51	32	34	8	1	1
	–	64	51	34	32	8	1	1
6:	21	64	51	34	32	8	1	0
	64	21	51	34	32	8	1	1
	64	51	21	34	32	8	1	1
	64	51	34	21	32	8	1	1
	64	51	34	32	21	8	1	1
المجموع							13	9

جدول (1)

3.3 التحليل الزمني لخوارزمية الترتيب بالإدخال

1.3.3 تحليل الحالات السيئة Worst-case Analysis

إذا نظرنا للحلقة الداخلية (while) سوف نجد أن معظم الجهد يمكن أن يظهر عند وجود عنصر أصغر من جميع العناصر الموجودة في القائمة المرتبة. وأيضاً معظم الجهد الذي سوف تقوم ببذله الخوارزمية يظهر عندما تكون القائمة في حالة معكوسة، وهذه الحالة هي الحالة السيئة التي يمكن أن تلازم هذه الخوارزمية. فإذا نظرنا إلى هذه الحالة فسوف نجد أن أول عنصر يراد إدخاله هو العنصر الثاني والذي سوف نقوم بترتيبه مع العنصر الأول، ثم العنصر التالي سوف يكون هو العنصر الثالث والذي سوف يتطلب ترتيبه مع العنصرين السابقين وهكذا حتي الوصول إلى العنصر الأخير في القائمة والذي يتطلب ترتيبه مع جميع عناصر القائمة المرتبة أي (n-1) عنصر. لهذا يمكن صياغة معدل الحالات السيئة على النحو الآتي:

$$T_{wrs}(n) = \sum_{i=1}^{n-1} i = \frac{1}{2}n(n-1) = O(n^2)$$

2.3.3 تحليل الحالات الوسطي Average-case Analysis

لحساب معدل الحالات الوسطى نحتاج لمرحلتين، في المرحلة الأولى نحتاج لتحديد متوسط عدد المقارنات المطلوبة لنقل عنصر واحد لمكانه الصحيح. في المرحلة الثانية نقوم بحساب متوسط عدد العمليات المطلوبة لنقل جميع العناصر لأماكنها الصحيحة. لنحاول الآن تحديد متوسط عدد المقارنات المطلوبة لنقل العنصر رقم i لموقعه الصحيح. وإضافة العنصر رقم i يحتاج على الأكثر i مقارنة ومن الواضح أن أي عنصر يحتاج لمقارنة واحدة حتى لو ظل في مكانه. أيضاً نلاحظ أنَّ العنصر رقم i له عدد (i+1) موقع محتمل يمكن نقله إليها داخل القائمة. وأيضاً يمكننا ملاحظة أن العنصر رقم i يحتاج لعدد مقارنات 1 ، 2 ، 3 ، ...، i مقارنة للوصول إلى المواقع i+1 ، i ، i – 1 ، ...، 2 . كما يحتاج إلى i مقارنة للوصول إلى الموقع الأول في القائمة ..... لذا يمكن حساب متوسط عدد المقارنات للعنصر رقم i كالآتي:

$$a_i = \frac{1}{i+1}\left[\sum_{p=1}^{i} p + i\right]$$

ولذا يمكن حساب المعدل الزمني للحالات الوسطى على النحو:

$$T_{avg}(n) = \sum_{i=1}^{n-1} a_i = \sum_{i=1}^{n-1}\left[\frac{i}{2} + 1 - \frac{1}{i+1}\right]$$

$$= \sum_{i=1}^{n-1}\frac{i}{2} + \sum_{i=1}^{n-1} 1 + \sum_{i=1}^{n-1}\frac{1}{i+1}$$

لاحظ الآتي:

$$\sum_{i=1}^{n-1}\frac{1}{i+1} = \sum_{i=2}^{n}\frac{1}{i} = \left(\sum_{i=1}^{n}\frac{1}{i}\right) - 1$$

ولذا يمكن أن نصل إلى الصيغة التالية:

$$T_{avg}(n) \approx \frac{1}{2}\left[\frac{(n-1)n}{2}\right] + (n-1) - (\ln n - 1)$$

والتي يمكن أن نختصرها على النحو الآتي:

$$T_{avg}(n) \approx \frac{n^2 + 3^n + 4}{4} - (\ln n - 1)$$

$$\approx \frac{n^2}{4}$$

$$= O(n^{2)}$$

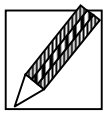
ملحوظة :

إذا رجعنا للمثال السابق فسوف نستطيع الوصول إلى معدل تقديري وفقاً لعدد المقارنات وعدد التبديلات على النحو الآتي:

$$T(n) = 13+3x9= 13+27 = 40$$

حيث عدد العمليات المطلوب وهو 40 يقع ضمن مجموعة $6^2= 36$ أي $O(n^2)$

تدريب (1)



(1) وضح نتيجة الترتيب لكل مرحلة من مراحل الترتيب بالإدخال للقائمة (من الشمال إلى اليمين): 8 1 6 5 2 4 9 3 7

(2) وضح نتيجة الترتيب لكل مرحلة من مراحل الترتيب بالإدخال للقائمة (من الشمال إلى اليمين): 7 4 6 1 8 9 2 5 3

(3) ما هو المعدل الزمني لخوارزمية الترتيب بالإدخال إذا تم استخدامه لترتيب قائمة بها n عنصر جميعها متساوية؟

4. الترتيب بالتجزئة Shell Sort

1.4 الطريقة

يطلق على هذا النوع من الخوارزميات بترتيب شـل نسـبة إلـى مصـممه (D. Shell). ويعتبر هذا النوع ترتيباً بالإدخال معدلاً، وتشتمل عملية الترتيب بهذا النـوع على عدة مراحل، في كل مرحلة يتم تقسيم القائمة إلى مجموعة من القوائم الجزئية حيث يتم ترتيبها باستخدام الترتيب بالإدخال، حيث في كل مرحلة يتم تقليص عدد المجموعات الجزئية وزيادة حجم كل مجموعة، وحتى الوصول في المرحلة الأخيرة إلـى مجموعـة واحدة هي المجموعة الكلية.

ولذا فإن الترتيب بالتجزئة يستخدم متتابعة h_1 ، h_2، h_3، ... h_t تسمى بمتتابعة الزيادة (Increment Sequence) .

وفي كل مرحلة من الترتيب يتم اختيار زيادة معينة، مثلاً h_k ، وبحيث يتم توليد h_k مجموعة جزئية من العناصر. ولكل مجموعة يجب التأكد من أن $a[i] \le a[i + h_k]$ لكل i في مدى المحصلة ويطلق على ذلك ترتيب h_k – وهو ترتيب بالإدخال جزئي.

ويمكن تلخيص طريقة شل كالآتي : (لكل موقع في القائمة h_k+1، h_k +2، ...، n ضع أي عنصر في الموقع الصحيح له من المواقـع : i ، i - h_k ، i – 2 h_k، ...الخ. حيث يمثل الترتيب ترتيب h_k – ترتيب بالإدخال جزئي للقائمـة الجزئيـة . ويتم تكراره حتى الوصول للترتيب الكلي للقائمة).

2.4 الخوارزمية

لصياغة خوارزمية تعمل وفقاً للطريقة السابقة للترتيب بالتجزئة، نحتاج لتعريف طريقة بسيطة للحصول على متتابعة الزيادة. يمكننا أن نختار الطريقة التالية:

للحصول على أكبر زيادة خذ $h_t = n/2$. وللحصول على الزيـادة التاليـة خذ القاعدة:

$$h_t - 1 = h_t / 2$$

فمثلاً إذا كان لدينا قائمة بها 10 عناصر أي (n=10) فإننـــا يمكننـــا اختيـــار الزيادات على النحو الآتي:

$$h_t = N/2 = 10/2 = 5$$

$$h_{t-1} = 5/2 \cong 2$$

$$h_{t-2} = 2/2 = 1$$

ولذا تكون لدينا متتابعة الزيادة التالية : 1 ، 2 ، 4

خوارزمية (2)

```
Shell Sort (a, n)
{1}   H = n  div  2
{2}   while  (h <  0)    do
          begin
{3}           for i = h+1   to   n   do
                  Begin
{4}                   tmp = a [i]  , j = i  ,  r  =  j-h
{5}                   while r  >  0   do
                          Begin
{6}                           if   tmp  <  a [j-h] then
                                  Begin
{7}                                       a [j] = a [j-h]
{8}                                       j = j-h ,  r = j
                                  Else
{9}                                       R = 0
                              end if
                      end while
{10}                  a [j]  = tmp
              end for
{11}      h = h  div  2
      end while
```

هذه الخوارزمية تقوم بتحديد عدد المجموعات الجزئية في كل مرحلة وهو h . ثم تبدأ بتنفيذ ترتيب بالإدخال على كل مجموعة من الخطوة {3} وحتى الخطوة {10} . ثم تقوم بتحديد عدد جديد للمجموعات حسب الخطوة {11} . تستمر حتى الوصول إلى المرحلة الأخيرة والتي فيها

(h = 1) أي سوف يكون هنالك مجموعة واحدة هي القائمة الأصلية.

ولك أن تسأل ما هي الحكمة من هذه الخوارزممية إذا كانت ستنفذ خوارزمية الترتيب بالإدخال على كل مجموعة جزئية، على كل حال هنالك فائدتان لهذا العمل:

الفائدة الأولى:

أن ترتيب قائمة تبتعد عناصرها عن بعضها البعض بمقدار h قد يزيد من سرعة وصول أي عنصر لموقعه الصحيح في القائمة، وخذ في بالك النتيجة التي توصلت لها في السابق والتي بواسطتها يمكننا أن نضمن معدل نمو لا يزيد عن $O(n^2)$.

الفائدة الثانية:

تذكر أن خوارزمية الترتيب بالإدخال فعالة جداً إذا كانت القائمة شبه مرتبة مسبقاً، وذلك لأن العنصر الموجود في مكانه الصحيح لا يحتاج لمقارنته مع بقية عناصر القائمة، وعليه ولكون كل مرحلة تنتج عنها قائمة مرتبة جزئياً، فهذا يعني أن تطبيق خوارزمية الترتيب بالإدخال لن يكون مكلفاً حيث يتم تطبيقها على قائمة أكثر ترتيباً من ذي قبل.

مثال (3)

رتب القائمة بترتيب التجزئة:

34 8 64 51 32 21

سنحاول أيضاً حل هذه المسألة بتوضيح جدول المتابعة كما في خوارزمية الترتيب بالإدخال، حيث إن المرحلة هنا يتم تحديدها وفقاً لمتتابعة الزيادات، وفي هذه المسألة يمكننا حساب الزيادات (h_t) على النحو الآتي:

$h_{t-1} = h_t/2 = 3/2 = 1 \longleftarrow h_t = n/2 = 6/2 = 3$

أي هنالك مرحلتان : المرحلة الأولى هي ترتيب –3 والمرحلة الثانية هي ترتيب –1، حيث في المرحلة الأولى سوف يتم تقسيم القائمة إلى ثلاث مجموعات جزئية وهي:

المجموعة الأولى	51	34
المجموعة الثانية	32	8
المجموعة الثالثة	21	64

وعليه سوف ننفذ ترتيباً بالإدخال على كل مجموعة، أما في المرحلة الثانية فسوف يكون هنالك مجموعة واحدة فقط هي القائمة الأصلية مرتبة ترتيباً جزئياً. يمكنك متابعة الحل في الجدول أدناه وحساب عدد المقارنات وعدد التبديلات المطلوبة لكل مرحلة.

المرحلة h	المجموعة P	القائمة						المقارنات C	التبديلات S
ترتيب –3	:4	34	–	–	[51]	–	–	1	0
	:5	–	8	–	–	[32]	[21]	1	0
	:6	–	–	64	–	–		1	0
		–	–	21	–	–	[64]	0	1
ترتيب –1	:2	34	[8]	–	–	–	–	1	0
		8		[21]				0	1
	:3	8	34	[34]	–	–	–	1	1
		8	21	[34]	–	–	–	0	1
	:4	8	21	34	[51]	–	–	0	1
	:5	8	21	34	51	[32]	–	1	1
		8	21	34	32	[51]	–	1	1
	:6	8	21	32	34	51	[64]	0	1
	المجموع	8	21	32	34	51	64	11	5

3.4 التحليل الزمني لخوارزمية الترتيب بالتجزئة

1.3.4 تحليل الحالات السيئة

على الرغم من أن خوارزمية الترتيب بالتجزئة بسيطة من ناحية التنفيذ، لكنها صعبة من ناحية تحليلها تحليلاً زمنياً، فالتحليل الزمني لهذه الخوارزمية يعتمد أساساً على طريقة اختيار متتابعة الزيادة بالطريقة الاعتيادية لشل في اختيار متتابعة الزيادة h_t $h_t = n/2$, $h_{t-1} = h_{t/2}$

نجد أن الخوارزمية تتبع لمجموعة $\theta(n^2)$. إذا تم اختيار n كقوى أساسية لعدد 2 بحيث تعدل كل الزيادات زوجية إلا الزيادة الأخيرة وهي فردية وهي 1 ، يمكننا إثبات ذلك بواسطة إثبات الحد الأدنى، خذ قائمة بحيث تصبح n/2 من الأعداد الكبيرة في القائمة موجودة في مواقع زوجية و n/2 من الأعداد الصغيرة موجودة في مواقع فردية، عند المرحلة الأخيرة، ولهذا فإن العنصر رقم I وقبل بداية المرحلة الأخيرة يكون عند الموقع 2i – 1، حيث يحتاج إلى تحريكة لعدد (i-1) موقع ليصل لموقعه الصحيح بالقائمة. على هذا فإننا لوضع n/2 عدد في مكانها الصحيح نحتاج على الأقل لجهد قدره $\sum_{i=1}^{n/2}(i-1)$ حركة ولذا فإن الحد الأدنى هو $\Omega(n^2)$. ولإثبات الحد الأعلى ، نجد أن أي مرحلة بزيادة قدرها h_k تحتوي على عدد h_k ترتيب بالإدخال وتضمن عدد n/h_k عنصر. وبما أن الترتيب بالإدخال هو ذو معدل تكعيبي، فإن المعدل الكلي للمرحلة يصبح هو $O\left(h_k\ (\ n/h_k)^2\right)$ أي $O\ (n^2/h_k)$. ولذا يصبح المعدل الكلي لكل المراحل كالآتي:

$$n^2 / h_i = n^2 \sum_{i=1}^{t} 1/h_i = n^2 \sum_{i=1}^{t}$$

لأن متتابعة الزيادات هي متوالية هندسية في نسبة ثابتة هي 2 وأكبر حد هو عند $h_i = 1$. ولذا نجد أن الحد الأعلى هو $O(n^2)$.

2.3.4 تأثير الزيادات على التحليل الزمني

لأن اختيار متتابعة الزيادات له أثر واضح في الترتيب النسبي لخوارزمية الترتيب بالتجزئة، فإن البحث عن المتتابعة المثالية لا يكون ذا جدوى. ولذا كان هناك عدد من الخيارات التي قام بوضعها عدد من العلماء في محاولة تحسين الترتيب النسبي لخوارزمية الترتيب بالتجزئة.

وضع العالم هيبارد خيار ثابت لمتتابعة على النحو 1، 3، 7، ...، 2^{k-1} للوصول إلى معدل نسبي قدره $\theta(n^{3/2})$. أيضاً هنالك خيار آخر وفيه يتم حساب كل القيم المحتملة $(2^i 3^j)$ (حيث $i \geq 0$, $j \geq 0$) والتي تكون أقل من حجم القائمة واستخدامها في متتابعة الزيادة ، مثلاً إذا كان (n=40) فيكون لدينا متتابعة على النحو:

$1(2^0 3^0)$ ، $2(2^1 3^1)$ ، $3(2^0 3^1)$ ، $4(2^2 3^0)$ ، $6(2^1 3^1)$ ، $8(2^3 3^0)$ ، $9(2^0 3^2)$ ، $12(2^2 3^1)$ ، $16(2^4 3^0)$ ، $18(2^1 3^2)$ ، $24(2^3 3^1)$ ، $27(2^0 3^3)$ ، $32(2^5 3^0)$ ، $36(2^2 3^2)$.

ولذا عند استخدام متتابعة على ذلك النحو يتم تقليل الترتيب النسبي إلى :

$$O\left(n(\log n)^2\right)$$

إذن خوارزمية الترتيب بالتجزئة هي مثال لخوارزمية تؤثر معالمها على الترتيب النسبي لها.

تدريب (2)

(1) وضح مراحل الترتيب المطلوبة لترتيب القائمة باستخدام متتابعة زيادة : 7 ، 5 ، 3 ،1 .

القائمة (من الشمال إلى اليمين):

16 15 14 13 12 11 10 9 8 7 6 5 3 2 1

ما هو عدد المقارنات المطلوب؟

(2) خذ متتابعة الزيادة : 8 ، 4 ، 2 ، 1 لترتيب نفس القائمة السابقة، وضح مراحل الترتيب المختلفة. ما هو عدد المقارنات المطلوب؟

(3) افترض في الترتيب بالتجزئة أننا نحتاج دوماً لتعديل العنصرين a[i] و a[i+k] . أثبت أنه على الأقل 1 معكوس وعلى الأكثر (2k – 1) معكوس يمكننا حذفهم.

(4) أحسب المعدل الزمني المطلوب عند استخدام الترتيب بالتجزئة بواسطة متتابعة زيادة {1,2}.

5. الترتيب السريع Quick Sort

يعتبر الترتيب السريع من أسرع طرق الترتيب المعروفة، وبواسطته يمكن الحصول على معدل للحالات الوسطى وقدره $O(n \log n)$ ، بينما معدل الحالات السيئة لا يتجاوز $O(n^2)$.

وخوارزمية الترتيب السريع هي خوارزمية استدعاء ذاتي وتتبع لتصميم فرق تسد والذي سوف نقوم بتوضيحه في الوحدة السادسة من هذا الكتاب، وتتلخص طريقة الترتيب السريع بتقسيم القائمة المراد ترتيبها لقسمين وفقاً لعنصر يطلق عليه عنصر الارتكاز (Pivot) بحيث القائمة الأولى تحتوي على عناصر أقل من عنصر الارتكاز، القائمة الثانية تحتوي على عناصر أكبر من عنصر الارتكاز. وثم نبدأ عملية الاستدعاء الذاتي لخوارزمية الترتيب السريع على القائمتين الجزئيتين وتستمر العملية وحتى الانتهاء من ترتيب القائمة.

1.5 طريقة الترتيب السريع

سوف نقوم بتوضيح خطوات بسيطة لشرح طريقة الترتيب السريع ثم نقوم بمناقشة بعض الجوانب الهامة في الترتيب السريع مثل كيفية اختيار عنصر الارتكاز وكيفية تقسيم القائمة وخطوات طريقة الترتيب السريع لترتيب القائمة أي Quick Sort (S) كالآتي:

1) إذا كانت القائمة S تحتوي على 0 أو 1 عنصر، توقف.
2) اختار عنصر V من S يطلق عليه عنصر الارتكاز (Pivot) .
3) قسم القائمة S- {V} بواسطة العنصر V إلى قسمين على النحو:

$$\{x \in s - \{v\} : x \le v\}, \text{ and } s_1 =$$
$$\{x \in s - \{v\} : x \ge v \; s_2 =$$

4) قم بترتيب القائمة على النحو:

Quick Sort (s_1) + v + Quick Sort (s_1)

هذه الخطوات لا تحتوي على التفاصيل المتعلقة بالخطوتين الثانية والثالثة، ولذا سوف نقوم بتوضيحهما ببعض الطرق المختلفة وسوف نقوم باتباع طرق محددة في هذا الكتاب للتعامل معهما.

1.1.5 اختيار عنصر الارتكاز

هنالك عدد من الطرق المختلفة لاختيار عنصر الارتكاز. وتختلف هذه الطرق عن بعضها البعض باختلاف الكيفية التي تتبعها في تحديد العنصر الذي على ضوءه يتم تقسيم القائمة، وقد تؤثر الطريقة المتبعة في اختيار عنصر الارتكاز على سرعة خوارزمية الترتيب السريع، ولذا كانت هنالك طرق جيدة وأخرى سيئة، وسوف نحاول توضيح بعض الطرق المختلفة في ذلك:

1) اختيار العنصر الأول:

من أشهر الطرق المتبعة في اختيار عنصر الارتكاز هي اختيار العنصر الأول في القائمة، وهذه الطريقة مقبولة إذا كانت القائمة المدخلة عشوائية، ولكن إذا كانت القائمة غير ذلك أي مرتبة مسبقاً أو معكوسة الترتيب، فإن هذه الطريقة سوف تكون طريقة سيئة التصميم لأن كل العناصر سوف تذهب إلى القائمة s_1 أو القائمة s_2.

والأثر العملي لذلك هو أن اختيار العنصر الأول قد يؤثر على الترتيب النسبي للخوارزمية عند حالة المدخلات والتي تتمثل في إدخال قائمة مرتبة مسبقاً ليصل إلى معدل تربيعي $O(n^2)$ هنالك خيار بتحديد العنصر الأكبر من العنصرين ألأولين في القائمة، ولكن يظل أيضاً هما أول عناصر القائمة.

2) اختيار عشوائي

هنالك طريقة باختيار عنصر الارتكاز عشوائياً من بين عناصر القائمة المراد ترتيبها. وهذه الطريقة أكثر أمناً من الطريقة الثانية وقد تؤدي إلي تحسين المعدل وسرعة الخوارزمية، ولكنها أيضاً تعاني من صعوبة تحديد عنصر الارتكاز عشوائياً لأن هذا قد يتطلب جهداً إضافياً لخوارزمية الترتيب السريع.

3) اختيار الوسيط

إن اختيار الوسيط بالقائمة قد يكون حلاً مجدياً، ولكن أيضاً يواجه بصعوبة حسابه وقد يبطئ من الخوارزيمة. لذا نجد أن هنالك طريقة عملية تتمثل في اختيار ثلاثة عناصر هي العنصر الأول والعنصر الثاني وسط القائمة والعنصر الأخير وثم حساب الوسيط من بينهم، واختياره عنصراً للارتكاز، وهذه الطريقة تتجاوز الحالات السيئة السابقة وتقلل معدل خوارزمية الترتيب.

2.1.5 استراتيجية تقسيم القائمة

هنالك عدد من الطرق المختلفة والتي تستخدم في تقسيم القائمة وفقاً لعنصر الارتكاز، وسوف نقوم هنا باتباع طريقة سهلة في التنفيذ والفهم تعتمد على مسح القائمة مرتين: من الشمال إلي اليمين ومن اليمين إلى الشمال. المسح من الشمال إلى يبدأ المسح من العنصر الثاني في القائمة، وحيث إننا نحتاج للعناصر الأقل في الجزء الأول من القائمة، فإننا نتجاوز العناصر الأقل من عنصر الارتكاز أمَّا ونتوقف عند أول عنصر أكبر أو يكافئ عنصر الأرتكاز. المسح من اليمين إلى الشمال فيبدأ عند آخر عنصر بالقائمة، وحيث إننا نحتاج العناصر الأكبر في الجزء الثاني من القائمة، فإننا نتجاوز العناصر الأكبر من عنصر الارتكاز ونتوقف عند أول عنصر أقل أو يكافئ عنصر الارتكاز، ولذا نجد أن هنالك ثلاثة حالات تعتمد على مدى تقاطع مؤشري المسح.

<u>الحالة الأولي:</u> إذا لم يتقاطع مؤشري المسح i (من على الشمال) و j (من على اليمين)، فيتم استبدال العنصر a[i] بالعنصر a[j] واستئناف عملية المسح بزيادة i وتقليل j .

v	all are ≤ v	a [j] ≥ v	…	a [j] ≤ v	all are ≥ v
		⟶ i		⟵ j	

<u>الحالة الثانية:</u> إذا تقاطع المؤشران أي (i > j) فإننا نكون أنجزنا التقسيم المطلوب ولذا نقوم باستبدال عنصر الارتكاز v بالعنصر a[j] .

v	all are ≤ v	a [j] ≤ v	a [j] ≥ v	all are ≥ v
		j ⟵	⟶ i	

<u>الحالة الثالثة:</u> إذا توقف المؤشرين عند نفس العنصر أي i = j فإن العنصر هو عنصر يكافئ عنصر الارتكاز ولذا نكون أنجزنا التقسيم ونقوم باستبدال عنصر الارتكاز v بالعنصر a[j] أيضاً.

v	all are ≤ v	a[j] = v و a [i]	all are ≥ v
⟵ i = j ⟶			

2.5 خوارزمية الترتيب السريع

تقوم خوارزمية الترتيب السريع على مستويين؛ المستوى الأول هو المستوى الرئيس وفيه الخوارزمية الرئيسة للترتيب السريع والتي تستدعي خوارزمية التقسيم حيث يبدأ المستوى الثاني، ومن ثم يتم الاستدعاء الذاتي للخوارزمية لترتيب القائمتين الجزئيتين بعد عملية التقسيم.

خوارزمية (3)

```
Quick Sort (a, L, R)
    if  L <  R    then
        begin
            S = partition (a, L, R),
            quick sort (a, L, S-1),
            quick sort (a, S+1, R)
        end
```

في المستوى الثاني تقوم خوارزمية التقسيم بتقسيم القائمة وفقاً للطريقة المتبعة والتي تم شرحها مسبقاً، حيث يتم تحديد عنصر الارتكاز v في مكان العنصر الأول، وتتم عملية المسح وفقاً للمؤشرين j وi ، بحيث يتم التحكم في المسح وفقاً للحالات الثلاثة السابقة وفي هذه الخوارزمية سوف نقوم باستخدام خوارزمية بسيطة هي خوارزمية التبديل (Swapping) لتبديل أي عنصرين وهي خوارزمية مفهومة ويمكنك كتابتها بنفسك.

خوارزمية (4)

```
Partition (a, L, R)
{1}   v = a [L],
{2}    i  =  L ,   j = R + 1,
        repeat
{3}           repeat  i  =  i+1  until  a [i]  ≥  v,
{4}           repeat  j  =  j-1   until a [j]  ≤  v,
{5}           swap  (a [i]    ,   a [j]),
        until   i  >   j,
{6}     swap (a [i]    ,   a [j]),
{7}     swap (a [L]    ,   a [j]),
{8}     return   j
```

في هذه الخوارزمية وفي الخطوتين {1} و {2} يتم تحديد موقع عنصر الارتكاز في بداية القائمة أي في الموقع L وتحديد المؤشر: i من البداية والمؤشر: j في النهاية. تبدأ عمليتا المسح وفقاً للخطوتين {3} و {4} . عند حدوث الحالة الأولى تبدأ عملية التبديل وفقاً للخطة {5} . وعند حدوث تقاطع المؤشرين أو تكافئهما تبدأ عملية تكرار تبديل آخر عنصرين و عملية تبديل عنصر الارتكاز بالعنصر a[j] .

مثال(4)

رتب القائمة التالية مستخدماً الترتيب السريع

34 8 64 51 32 21

سوف نقوم بعملية الترتيب مستخدمين الخوارزمية السابقة موضحين المؤشرين i و j والعناصر التي سوف تقوم الخوارزمية باستبدال مواقعها بعلامة (*) والعناصر التي تجاوزتها بعلامة (-) ، كما سوف نوضح موقع عنصر الارتكاز بمربع وسنقوم بتوضيح خطوات التدرتيب على جدول المتابعة الثاني:

التبديلات	المقارنات	القـــــائمـــة						المؤشرين	
S	C	34	8	64	51	32	21	i 's	j's
0	3	**34**	-	*	.	.	*	3	6
1	0	**34**	_	21	.	.	46	3	6
0	2	**34**	_	_	*	*	_	4	5
1	0	**34**	_	_	32	51	_	4	5
0	2	**34**	_	_	* _	_ *	_	5	4
1	0	**34**	_	_	51	32	_	5	4
1	0	**34**	_	_	32	51	_	5	4
1	0	32	_	_	**34**	51	_	5	4
0	4	**32**	-	*	.	**51**	*	3,6	3,5
2	0	**32**	_	21	.	**64**	51	3,6	3,5
1	1	21	_	**32**	.	**51**	64	3,6	3,5
0	2	**21**	*	.	.	.	.	2	1
1	0	8	**21**	.	.	.	.	2	1
1	0	**21**	8	.	.	.	.	2	1
1	0	8	**21**	.	.	.	.	2	1
0	0	.	.	.	.	.	.	2	1
11	**14**	**8**	**21**	**32**	**34**	**51**	**64**		**المجموع**

في هذا المثال تحتاج خوارزمية الترتيب السريع إلى 11 حركة تبديل مختلفة لتنجز مهمتها. وهذا المعدل هو أعلى من المعدل الذي تحتاجه خوارزمية الترتيب بالإدخال لإنجاز نفس المهمة في مثال (2) . ويتضح من هذه المقارنة بأن خوارزمية الترتيب السريع تعمل بفعالية فقط في القوائم ذات الأحجام الكبيرة، ولا تكون فعالة في ترتيب القوائم صغيرة الحجم.

3.5 التحليل الزمني لخوارزمية الترتيب السريع

1.3.5 تحليل الحالات السيئة

لأن خوارزمية الترتيب السريع هي خوارزمية فرق تسد، لا يكون هنالك حالة أفضل إلا إذا تم تقسيم القائمة إلى قائمتين شبه متكافئتين في حجمهما، ولذا فإن الحالة السيئة هي الحالة المختلفة والتي تكون فيها القائمتان الجزئيتان مختلفتي الحجم. وأكبر اختلاف في حجمهما يحدث عندما يكون عنصر الارتكاز إما أكبر أو أقل من جميع عناصر القائمة وعند هذه الحالة يكون لدينا جزء به (n-1) عنصر وجزء آخر لا يوجد به أي عنصر، ولذا إذا حدث هذا كل مرة لا نستطيع إلا سحب عنصر واحد من القائمة كل مرة وهو عنصر الارتكاز، ولهذا فإن المعدل الزمني لهذه الحالة السيئة يمكن التعبير عنه على النحو الآتي:

$$\sum_{i=2}^{n} (i-1)\ T_{wrs}(n) =$$
$$= n(n-1)/2 = O(n^2)$$

2.3.5 تحليل الحالات الوسطى

تذكر عندما قمنا بتحليل الترتيب بالتجزئة لاحظنا أن هذه الطريقة أفضل من طريقة الترتيب بالإدخال لإنها تقوم بتقليل عدد المعكوسات في كل مرة تقوم بمقارنة العناصر البعيدة عن بعضها البعض.

ولنا الحق أن نضع سؤالاً هنا وهو كيف تقوم خوارزمية الترتيب السريع بتقليل عدد المعكوسات؟

خذ قائمة بها n عنصر نود تقسيمها وفقاً لعنصر الارتكاز، فإذا أخذنا الحالة التي يكون فيها عنصر الارتكاز أكبر من جميع عناصر القائمة ، فإن خوارزمية التقسيم سوف تقوم بإرجاع n ، ولهذا فإن عنصر الارتكاز سوف يتم استبدال موقعه في أول موقع في القائمة مع آخر موقع فيها، وهنالك احتمال بأن يكون العنصر الموجود في

الموقع الأخير هو أصغر عناصر القائمة، وفي هذه الحالة سوف تكتسب الخوارزمية ميزة تقليل عدد المعكوسات لأن تحول العنصر الأكبر إلى الموقع الأخير يقلل عدد (n-1) معكوس، أما تحول العنصر الأصغر إلى أول موقع يقلل عدد (n-1) معكوس. ولهذا فإن حركة تبديل واحد تكفل لتقليل 2n-2 معكوس، وهذا يدل على أن خوارزمية الترتيب السريع لها معدل زمني للحالات الوسطى يختلف تماماً عن معدلها الزمني للحالات السيئة.

إن الأهمية الكبيرة لخوارزمية التقسيم في تحديد معدل الحالات الوسطى، جعلتنا نقوم بدراستها لجميع الحالات المحتملة، وعندما نقوم بتنفيذ هذه الخوارزمية على قائمة بها n عنصر، فأي موقع من المواقع والتي عددها n له فرصة أن يكون هو الموقع الصحيح لعنصر الارتكاز ليتم التحول إليه، ولهذا عندما نود أن ندرس الحالة الوسطى لابد أن نأخذ في اعتبارنا كل هذه الحالات ونحسب المتوسط لها. لاحظ أنه عندما تقوم خوارزمية التقسيم بإرجاع الموقع الصحيح لعنصر الارتكاز، مثلاً p، فإننا نقوم باستدعاء ذاتي لخوارزمية الترتيب السريع مع قائمة بها (p-1) عنصر، وقائمة أخرى بها (n-p) عنصر. ولهذا فإن الحالة الوسطى لابد أن تنظر لجميع الحالات المختلفة أي n حالة مختلفة لقيمة p . ولذا نستطيع أن نعبر عن معدل الحالات الوسطى على النحو الآتي:

$$T_{avg}(n) = (n-1) + 1/n \sum_{i=1}^{n} [T_{avg}(i-1) + T_{avg}(n-i)]$$

$$T_{avg}(0) = T_{avg}(1) = 0 \quad , \quad n \geq 2 \qquad \text{عند}$$

ولتبسيط هذه العلاقة نستخدم الملاحظة التالية:

للحد الأول يستخدم قيم من صفر وحتى (n-1) ، بينما الحد الثاني يستخدم قيم من (n-1) وإلى صفر، وهذا يدل على أن المجموع يستخدم قيم من 0 وحتى (n-1) مرتين. لذلك نصل للتبسيط التالي:

$$T_{avg}(n) = (n-1) + 1/n \left[2 \sum_{i=0}^{n-1} T_{avg}[i] \right]$$

$$T_{avg}(0) = T_{avg}(1) = 0 \quad , \quad n \geq 0 \qquad \text{عند}$$

لحل هذه العلاقة وللتخلص من المقام قم بحساب:

$n^* T_{avg}(n)$, $(n-1)^* T_{avg}(n-1)$ كالآتي:

$$n^* T_{avg}(n) = n(n-1) + 2\sum_{i=0}^{n-1} T_{avg}[i]$$

$$= n(n-1) + 2\,T_{avg}(n-1) + 2\sum_{i=0}^{n-2} T_{avg}[i]$$

$$(n-1)^* T_{avg}(n-1) = (n-1)(n-2) + 2\sum_{i=0}^{n-2} T_{avg}[i]$$

والآن قم بحساب الفرق بين المعادلتين:

$$n * T_{avg}(n) - (n-1) * T_{avg}(n-1)$$
$$= 2\,T_{avg}(n-1) + n(n-1) - (n-1)(n-2)$$
$$= 2\,T_{avg}(n-1) + 2^2 - n - (n^2 - 3n + 2)$$
$$= 2\,T_{avg}(n-1) + 2(n-1)$$

والآن قم بإضافة $(n-1) * T_{avg}(n-1\_$ للطرفين:

$$n^* T_{avg}(n) = 2\,T_{avg}(n-1) + (n-1) * T_{avg}(n-1) + 2n - 2$$
$$= (2 + n-1) * T_{avg}(n-1) + 2n-2$$

ولهذا نصل لعلاقة الاستدعاء الذاتي التالية:

$$T_{avg}(n) = \frac{(n-1)^* T_{avg}(n-1) + 2n-2}{n}$$

$$T_{avg}(0) = T_{avg}(1) = 0$$

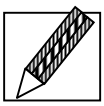
وحل هذه العلاقة سوف يوصلنا للنتيجة التالية:

$$T_{avg}(n) \approx 1.4\,(n+1)\log n$$

ولذا يمكننا أن نصل للمعدل التالي:

$$T_{avg}(n) = O(n \log n)$$

تدريب (3)



(1) حاول أن تستخدم الترتيب السريع في ترتيب القائمة :

4 , 35 , 7 , 2 , 1 , 13 , 9 , 42 , 3 , 21 , 17 , 23

حدد بداية ونهاية القائمة وعنصر الارتكاز في كل استدعاء ذاتي. ثم قم بحساب عدد المقارنات وعدد التبديلات.

(2) حاول أن تقوم بنفس المطلوب في السؤال الأول مع القائمة:

1 , 20 , 18 , 6 , 8 , 15 , 17 , 2 , 12 , 14 , 9 , 3

(3) حاول أن تقوم بحل السؤالين الأول والثاني مستخدماً طريقة الوسيط في اختيار عنصر الارتكاز من بين العناصر الأول والأخير وعنصر الوسط أي العنصر الموجود في الموقع(L+R) ½ .

(4) كم عدد المقارنات التي تحتاجها خوارزمية الترتيب السريع لترتيب قائمة من N عنصر متشابهاً؟

(5) مستخدماً نفس الفكرة الموجودة في خوارزمية التقسيم قم بكتابة خوارزمية لنقل الأعداد الزوجية لمواقع زوجية ونقل الأعداد الفردية لمواقع فردية في قائمة من N عدد مفرد.

أسئلة تقويم ذاتي

- من خلال فهمك لعملية الترتيب بالإدخال قم بوصف مراحل الترتيب في المثال (2) كتابة.

الخلاصة

عزيزي الدارس،

نلخص معاً الوحدة التالية لنستفيد من النقاط الهامة الواردة فيها، تحدثنا في بداية الوحدة عن أهمية الترتيب عند معالجة مجموعة كبيرة من السجلات باستخدام خوارزميات سريعة وفعالة. وقد ضحنا ثلاثة أنواع مختلفة من خوارزميات الترتيب وهي كالآتي:

1) الترتيب بالإدخال، وهي أبسط طرق الترتيب وتتم خلال (n-1) مرحلة ويبدأ المؤشر في العنصر الثاني حتى العنصر رقم n ، وكتبنا صيغة الخوارزمية وبينا كيف تتعلم هذه الخوارزمية.
2) الترتيب بالتجزئة أو ترتيب شل ويعتبر هذا النوع ترتيباً بالإدخال معدلاً حيث يتم تقسيم القائمة إلى مجموعة من القوائم الجزئية ويتم ترتيبها باستخدام الترتيب بالإدخال، وقلنا إن خوارزمية الترتيب بالتجزئة بسيطة من ناحية التنفيذ ولكنها صعبة من ناحية تحليلها تحليلاً زمنياً.
3) التريب السريع ويعتبر من أ سرع طرق الترتيب المعروفة وبواسطته يمكن الحصول على معدل للحالات الوسطى وقدره $O(n \lg o\ n)$ ، بينما معدل الحالات السيئة لا يتجاوز $O(n^2)$ وخوارزميته هي خوارزمية استدعاء ذاتي وتتبع لتصميم فرق تسد، وتقسم الخوارزمية العناصر قسمين وفقاً لعنصر يطلق عليه عنصر الارتكاز وقد شرحنا ثلاثة طرق لاختيار هذا العنصر.

أتمني عزيزي الدارس أن تكون قد وفقت في دراسة هذه الوحدة وأكملت حل التدريبات وأسئلة التقويم الواردة فيها – أعانك الله.

لمحة مسبقة عن الوحدة التالية

تعالج الوحدة التالية موضوع خوارزميات البحث، وتتم عملية البحث عن عنصر معين ضمن مجموعة من العناصر، ارجوعزيزي الدارس ان تكون قد استوعبت ما سبق حتي تواصل معنا لخوارزميات البحث، هذه الخوارزميات الهامة جداً في مجال علوم وتقانة الحاسوب ، وفقك الله.

مسرد المصطلحات

- **ترتيب Sorting**

عملية ترتيب البيانات قيد البحث، حسب ترتيب معين: إما تصاعدياً أو تنازلياً، وفقاً لقيم عددية أو رمزية.

- **ترتيب الإدخال Insertion Sort**

تتلخص طريقتها في إدخال أي عنصر جديد في المكان المناسب له ضمن القائمة التي سبق ترتيبها.

- **ترتيب شل (التجزئة) Shell Sort**

تتلخص طريقته بتنفيذ ترتيب القائمة في عدة مراحل في كل مرحلة يتم تجزئة القائمة لقوائم جزئية بحيث يتم ترتيبها بالإدخال

- **الترتيب السريع Quick Sort**

وتتلخص طريقته بتقسيم القائمة المراد ترتيبها لقسمين وفقاً لعنصر يطلق عليه عنصر الارتكاز (Pivot) حيث أن القائمة الأولى تحتوي على عناصر أقل من عنصر الارتكاز والقائمة الثانية تحتوي على عناصر أكبر من عنصر الارتكاز.

- **متتابعة الزيادة Increment Sequence**

هي متتابعة تتزايد وفقاً لمعدل ثابت، يستخدم الترتيب بالتجزئة متتابعة h_1 ، h_2، h_3، ... h_t ، وفي كل مرحلة من الترتيب يتم اختيار زيادة معينة، مثلاً h_k، بحيث يتم توليد h_k مجموعة جزئية من العناصر. ولكل مجموعة يجب التأكد من أن $a[i] \leq a[i + h_k]$ لكل i في مدى المحصلة ويطلق على ذلك ترتيب h_k – وهو ترتيب بالأدخال جزئي.

- **عنصر ارتكاز Pivot Element**

عنصر يستخدم في تقسيم القائمة المراد ترتيبها لقسمين وفقاً حيث أن القائمة الأولى تحتوي على عناصر أقل من عنصر الارتكاز والقائمة الثانية تحتوي على عناصر أكبر من عنصر الارتكاز

المراجع

المراجع العربية

1) مجموعة مؤلفين، تركيب البيانات وتصميم الخوارزميات، منشورات جامعة القدس المفتوحة، 1998.

2) السماني عبد المطلب، هياكل البيانات، منشورات جامعة السودان المفتوحة، 2005.

المراجع الأجنبية

1) Weiss, M.A., Data Structures and Alogrithm Analysis, Benjamin pub., 1992.
2) McConnell, J.J., Analysis of Alogrithms: An Active Approach, Jones pub., 2001.
3) Lavitin, A., Introduction to the Design and Analysis of Alogrithms, Addisions ues by, 2003.
4) Ulman, J.D (etal), The Design and Analysis of Compurte Alogrithms, Addison Wesley, 1974.
5) Basese, S., Compurte Alogrithms : Introduction to Analysis and Design, Addison Wesley, 2000.
6) Sedgewick, R., An Introduction to the Analysis of Alogrithms, Addison Wesley 1996.


الوحدة الخامسة
تحليل خوارزميات البحث
Seaching Algorithms Analsis

محتويات الوحدة

المقدمة

تمهيد

عزيزي الدراس مرحباً بك إلى الوحدة الخامسة من هذا المقرر وهي بعنوان تحليل خوارزميات البحث.

تعالج هذه الوحدة موضوع خوارزميات البحث، وتتم عملية البحث عن عنصر معين ضمن مجموعة من العناصر، ففي حالة وجود هذا العنصر ضمن مجموعة العناصر يتم تحديد موقع العنصر ضمن هذه العناصر، هذا وقد تكون البيانات مرتبة حسب مفتاح معين أو غير مرتبة، لكن في أغلب الأحيان تتم عملية البحث ضمن بيانات مرتبة مما يساعد في إتمام العملية بكفاءة أكثر، ومثال لذلك أنك تجد فرقاً شاسعاً في كفاءة عملية البحث في دليل الهاتف المرتب وبين عملية البحث عندما تكون الأسماء غير مرتبة، وسوف نقوم بمعالجة طريقتين؛ هما البحث التتابعي والبحث الثنائي، ويجدر الذكر أن الطريقتين تختلفان من حيث الكفاءة. فإحداهما أكثر ملاءمة من الأخرى لعدد قليل من البيانات وهي طريقة البحث التتابعي، والأخرى تحتاج لترتيب البيانات قبل عملية الترتيب وهي طريقة البحث الثنائي. في هذه الوحدة قمنا بتوضيح الأمثلة والتدريبات المناسبة لموضوع الوحدة، ولتدعيم المفاهيم المتضمنة في هذه الوحدة، يمكنك تطبيق الأمثلة والتدريبات على الحاسوب بكتابة برامج وتنفيذها على أجهزة الحاسوب المتوفرة لديك.

تتكون هذه الوحدة من أربعة أقسام رئيسة. في القسم الأول سوف نقوم بتوضيح مفهوم عملية البحث وأهميتها في دراسات الحاسوب. وفي القسم الثاني سوف نتناول الكيفية العامة لعملية البحث عن البيانات. أما في القسم الثالث فسوف نقوم بشرح طريقة البحث التتابعي والتعرف على كيفية تحليلها. وفي القسم الرابع والأخير سوف نقوم بتوضيح طريقة البحث الثنائي وكيفية تحليلها.

هذا وقد زودناك في هذه الوحدة بمجوعة من الأنشطة والتدريبات وأسئلة التقويم الـــذاتي نتمنى أن تستفيد منها كما نتتمنى أن تشاركنا في في نقدها وتقويمها.

أهداف الوحدة

عزيزي الدارس بعد الانتهاء من دراسة هذه الوحدة أتوقع أن تكون قادراً على أن:

- ✓ **توضح** أهمية عملية البحث عند معالجة البيانات.
- ✓ **تميز** بين طرق البحث المختلفة.
- ✓ **تقوم** بكتابة البرامج اللازمة للبحث عن البيانات.
- ✓ **تحلل** خوارزميات البحث المختلفة تقارنها مع بعضها.

1. مقدمة إلى البحث عن البيانات

كما وضحنا لك في الوحدة الرابعة أهمية عملية الترتيب في دراسات الحاسوب، فإننا سوف نقوم بتوضيح الأهمية للبحث عن البيانات، إن عمليتي الترتيب والبحث هما عمليتان متلازمتان عند معالجة البيانات. وسوف عندما تود حفظ البيانات واسترجاعها، إنك تحتاج إلى ترتيبها وفق ترتيب معين لكي يسهل عليك عملية البحث عن قيمة معينة ضمنها لغرض معالجتها بالتعديل أو الحذف أو الإضافة. ولهذا تعتبر عمليتا الترتيب والبحث من العمليات الأساسية في مجال دراسات الحاسوب، ونقصد هنا بالبحث عملية إيجاد موقع عنصر معين ضمن مجموعة من القيم، فإن لم يكن ذلك العنصر موجوداً يمكن طباعة إشعار بذلك، كما يمكن البحث عن قيم عددية أو قيم رمزية كذلك، هذا وكما وضحنا في الوحدة الرابعة فإن أهمية الترتيب والبحث تبرز عند معالجة مجموعة كبيرة من البيانات التي تتطلب الاحتفاظ بها وسرعة الوصول إليها، ومن أمثلة ذلك النظم المختلفة على الحاسوب مثل نظام المرتبات، نظام المكتبة، نظام التسجيل والقبول، وما إلى ذلك وتتطلب عملية البحث أن نصمم خوارزميات ملائمة تؤثر عليها عوامل عدة منها:

أ. الوقت الذي يحتاجه المبرمج لكتابة البرنامج وفقاً للخوارزمية المصممة.
ب. المدة التي يستغرقها تنفيذ البرنامج.
ج. مساحة الذاكرة التي يحتاجها تنفيذ البرنامج.
د. خواص البيانات المراد البحث عنها مثل عددها ونوعها.

فإذا كان عدد العناصر المراد البحث عنها قليلاً فلا داعي للبحث عن خوارزمية ذات كفاءة عالية قد تحتاج كتابتها إلى جهد أكبر من المبرمج، أما إذا كان عدد العناصر المراد البحث عنها كبيراً جداً وخصوصاً عندما يكون نوعها معقداً مثل البحث عن الأسماء مثلاً فإننا نحاول أن نجد خوارزمية للبحث تستغرق أقل وقت ممكن عند تنفيذها على جهاز الحاسوب.

إذاً هنالك فائدة كبيرة ومهمة لعملية البحث عن البيانات وهي المساعدة في إنجاز معالجة البيانات بغرض التعديل أو الحذف أو الإضافة، والحقيقة إنه بسبب توفر عملية البحث نستطيع إيجاد أي قيمة مراد تعديلها بسهولة ضمن مجموعة من القيم المرتبة أو غير المرتبة.

2. هل تتسأل كيف نبحث عن البيانات؟

إن عملية البحث عن المعلومات ضمن قائمة هي عملية أساسية في مجال دراسات الحاسوب، ولتوضيح عملية البحث، نفترض وجود قائمة تحتوي على سجلات من المعلومات المختلفة والتي يمكن حفظها بواسطة المصفوفات في برنامج الحاسوب مثلاً، وهذه السجلات يفترض أن تكون موجودة بشكل متتابع داخل القائمة، حيث إن كل سجل له موقع محدد داخل القائمة، وهذه المواقع يمكن فهرستها من الموقع الأول إلى الموقع رقم N ، حيث N يمثل عدد السجلات بالقائمة، وأي سجل يمكن أن يحتوي على مجال واحد أو عدد من المجالات المختلفة، ولكننا نكون ولغرض البحث مهتمين بمجال واحد فقط، نطلق عليه المفتاح وبواسطته تتم عملية البحث عن أي قيمة تتبع لهذا المجال، وهذه السجلات تكون مرتبة أو غير مرتبة بناءً على قيم المفتاح.

عندما تكون السجلات غير مرتبة، فلا يكون لدينا خيار غير استخدام طريقة البحث التتابعي والتي بالطبع هي طريقة سهلة جداً وتتلخص في البحث عن القيمة المراد إيجادها ضمن القائمة بالتتابع أو التوالي من أول قيمة وإلى آخر قيمة بالقائمة وفقاً لقيم المفتاح طبعاً، أما إذا كانت السجلات مرتبة، فيمكننا استخدام طريقة البحث الثنائي والتي تتلخص في تقليص حجم القائمة عند كل مقارنة بواسطة تقسيمها إلى قائمتين فرعيتين أحدهما لا يوجد بها العنصر المراد البحث عنه ولذا نتجاهلها، والأخرى نضمن وجود العنصر المراد البحث عنه ولذا تستمر بها في المرحلة القادمة من عملية البحث.

مثال (1)

يمكننا عمل ملف خاص بالطلاب المسجلين في جامعة السودان المفتوحة، حيث يحتوي سجل الطالب على بعض المعلومات مثل رقم الطالب، اسم الطالب، عنوانه، تخصصه، بالإضافة إلى معلومات أخرى. كما يمكننا اعتبار رقم الطالب المفتاح الذي يميز السجلات عن بعضها البعض. ولذا عند البحث عن معلومات طالب معين نحتاج لمعرفة رقمه لكي نستطيع استرجاع سجله، ومن الممكن في بعض الأحوال أن تكون لدينا رغبة في معرفة معلومات عن طالب معين لا نعرف رقمه، لكننا نعرف إسمه، ففي هذه الحالة نستعمل إسم الطالب كمفتاح بحيث يتم البحث عن معلومات الطالب بواسطة الإسم المذكور.

3. البحث التتابعي

1.3 ما هي الطريقة المتبعة في هذا البحث؟

تعتبر هذه الطريقة من طرق البحث الأقل كفاءة والأكثر سهولة، وتستخدم هذه الطريقة عندما تكون السجلات مخزنة دون اعتبار للترتيب. أو عندما تكون وسيلة التخزين لا توفر أي نوع من طرق الاسترجاع المباشر أو المفهرس. وتتلخص هذه الطريقة في أخذ كل سجل من سجلات الملف على حده بدءاً بالسجل الأول في القائمة. وتتم مقارنة مفتاح البحث مع المفتاح الموجود في السجل الأول، فإذا كشفت النتيجة عن وجود المفتاح في السجل تتوقف عملية البحث ويتم إرجاع موقع هذا السجل بالمف، أما إذا لم يوافق مفتاح البحث مقابله في السجل فإن عملية المقارنة تنتقل إلى السجل التالي، وهكذا حتى نهاية القائمة أو نجاح عملية المقارنة.

2.3 الخوارزمية

الآن عزيزي الدارس، يمكننا صياغة الخوارزمية أدناه وفقاً للطريقة التي تم شرحها سابقاً عن البحث التتابعي:

خوارزمية (1)

```
Sequential Search (list, target, N)
        Begin
{1}         for  i=1   to    N    do
                Begin
{2}                 if (target=list [i])    then
{3}                             return  i
                    end if
              end for
{4}         return 0
          end
```

في هذه الخوارزمية البسيطة يتم البحث من المواقع 1 وحتى الموقع N، في الخطوة الثانية {2} تتم مقارنة مفتاح البحث المطلوب (target) مع مفتاح السجل ([i] list). فإذا وجد مفتاح البحث وتطابق مع مفتاح السجل فإن الخوارزمية تقوم بإرجاع رقم السجل (i) ، أما إذا لم يتم إيجاد السجل فتكتمل عملية البحث وتقوم الخوارزمية بإرجاع الرقم (صفر) أي يعني ذلك بأن مفتاح البحث يقع في موقع خارج نطاق القائمة.

مثال (2)

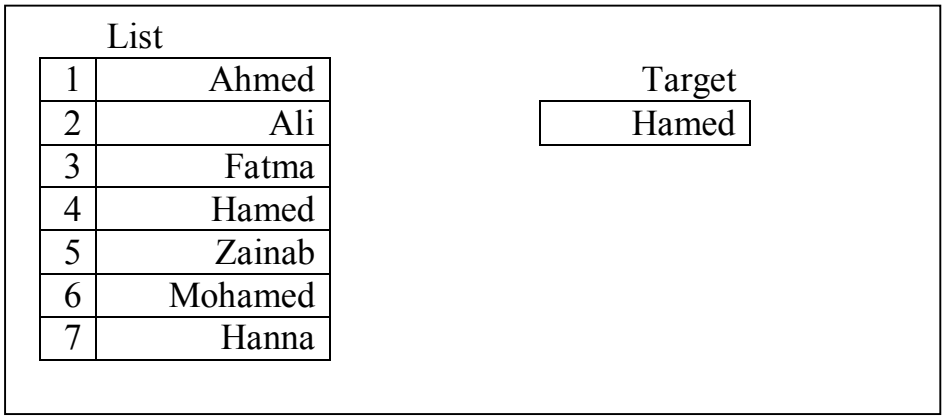
إذا فرض أن لدينا قائمة بأسماء 7 طلاب :

Ahmed, Ali, Fatma, Hamed, Zainab, Mohamed, Hanan.

المطلوب هو البحث عن اسم Hamed بالقائمة بواسطة البحث التتابعي.

الحل

حسب خوارزمية البحث التتابعي فإن قيمة المتغير i سوف تبدأ من الموقع الأول أي (i=1) ، وذلك لمقارنة قيمة مفتاح البحث (target) وهي هنا (Hamed) مع مفتاح السجل الأول بالقائمة وهو هنا (Ahmed)، وحيث أنه لا يساويه، فنقوم بالانتقال إلى العنصر الثاني في القائمة، وحيث إن العنصر الثاني أيضاً لا يساوي قيمة مفتاح بالبحث، فإن الخوارزمية تتابع المقارنة مع بقية العناصر بالقائمة حتى تصل إلى العنصر الرابع أي (Hamed)، عندها يكون مفتاح البحث مساوياً لمحتوى مفتاح السجل، ولذا تقوم الخوارزمية بالتوقف وإرجاع قيمة المتغير i عندها أي الموقع الرابع.



شكل رقم (1)

3.3 التحليل الزمني لخوارزمية البحث التتابعي

1.3.3 تحليل الحالات السيئة

هنالك حالتان تمر بهما خوارزمية البحث التتابعي يمكن اعتبارهما حالتين سيئتين؛ الحالة الأولى هي الحالة التي يكون فيها العنصر المطلوب موجوداً بالموقع الأخير بالقائمة. والحالة الثانية هي الحالة التي لا يكون فيها العنصر المطلوب موجوداً بالقائمة، فإذا فرضنا أن جميع العناصر الموجودة بالقائمة هي عناصر فريدة، فإننا سوف نقوم بمقارنة العنصر المطلوب مع جميع العناصر بالقائمة، وحتى الوصول لمقارنته بالعنصر الأخير بالقائمة. ولذا نحتاج إلى N مقارنة، حيث N هو حجم القائمة. وأيضاً فإننا نحتاج إلى N مقارنة لمعرفة أن العنصر المطلوب بالقائمة لو كان غير موجود بها فإننا نحتاج إلى N مقارنة لمعرفة ذلك، لهذا فإننا نستطيع أن نستنتج بإن الحد الأعلى لمعدل هذه الخوارزمية بناءً على عملية المقارنة هو N أي أن معدل الحالات السيئة هو:

$$T_{wrs}(N) = O(N)$$

2.3.3 تحليل الحالات الوسطى

هنالك حالتان يمكن دراستهما لتحليل الحالة الوسطى لخوارزمية البحث التتابعي. نفترض الحالة الأولى بأن عملية البحث ناجحة في إيجاد العنصر ضمن عناصر القائمة، بينما الحالة الثانية نفترض أن العنصر المطلوب البحث عنه في بعض الأحيان لا يتم إيجاده ضمن عناصر القائمة.

إذا كان العنصر المطلوب موجوداً بالقائمة، فإن هنالك N موقعاً محتملاً. دعنا نفترض أن احتمال وجوده في أي موقع من المواقع التي عددها N هو احتمالاً متساوياً، أي أنه يكافئ 1/N لكل موقع بالقائمة.

الآن دعنا نجيب عن بعض الأسئلة: ما هو عدد المقارنات المطلوب لإيجاد العنصر في الموقع الأول ؟ في الموقع الثاني؟ في الموقع الثالث؟ في الموقع رقم N ؟. الإجابة عن هذه الأسئلة ببساطة يمكنك معرفتها إذا رجعت لخوارزمية البحث التتابعي. إن الإجابة هي على التوالي: N,3,2,1 . ولهذا يمكننا صياغة المعدل الزمني للحالات الوسطى على النحو الآتي:

$$T_{\arg}(N) = \frac{1}{N}\sum_{i=1}^{N} i$$

$$= \frac{1}{N} * \quad \frac{N(N+1)}{2}$$

$$= \frac{N+1}{2} \quad = O(N)$$

والآن إذا ذهبنا للحالة الثانية حيث إن العنصر لا يتم إيجاده بالقائمة، نجد أنه لدينا (N+1) احتمالاً. وبنفس الطريقة السابقة إذا افترضنا أن جميع تلك الاحتمالات متكافئة، فإنه يمكننا صياغة معدل الحالات السيئة في هذه الحالة على النحو:

$$T_{\arg}(N) = \frac{1}{N+1} * \left(\sum_{i=1}^{N} i + N \right)$$

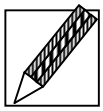
$$= \frac{1}{N+1} \sum_{i=1}^{N} i + \frac{N}{N+1}$$

$$= \frac{1}{N+1} * \frac{N(N+1)}{2} + \frac{N}{N+1}$$

$$= \frac{N}{2} + \frac{N}{N+1} = \frac{N}{2} + 1 - \frac{1}{N+1}$$

$$\cong \frac{N+2}{2} = O(N)$$

تدريب (1)



1) نفذ خوارزمية البحث التتابعي للبحث عن الرقم (44) في القائمة التالية:

64, 35, 21, 100, 85, 19, 44, 17, 153, 28

2) خذ خوارزمية شبيهة بخوارزمية البحث التتابعي تقوم بمسح قائمة لإرجاع عدد مرات وجود عنصر ما بالقائمة، هل كفاءة هذه الخوارزمية تختلف من كفاءة خوارزمية البحث التتابعي؟

نشاط

يمكنك استخدام طريقة البحث التتابعي مع قائمة مرتبة، أكتب خوازمية تقوم بإرجاع نفس نتيجة خوارزمية البحث التتابعي الأصلية، ولكنها يمكن تنفيذها بشكل أفضل وأسرع لأنها يمكن أن تتوقف عند اكتشاق أن العنصر المطلوب أكبر من العنصر الحالي في عملية البحث وقبل الوصول إلى العنصر الأخير بالقائمة، وعند كتابتك لهذه الخوارزمية استخدم دالة مقارنة compare (x,y) تعرف كالآتي:

$$compare(x,y)=\begin{cases}-1 & if \quad x < y\\ 0 & if \quad x = y\\ 1 & if \quad x > y\end{cases}$$

هذه الدالة يمكن استدعاؤها عند كل مقارنة، كما يمكن استخدامها مع جملة case أو switch . قم بإجراء تحليل الحالات السيئة والحالات الوسطى عند حالة وجود العنصر ضمن القائمة وأيضاً عند عدم وجوده ضمن القائمة.

4. البحث الثنائي

1.4 الطريقة

تتطلب هذه الطريقة أن تكون القائمة مرتبة. وتتلخص الطريقة علي النحـــو الآتـــي: تبدأ عملية البحث عند منتصف القائمة، فإذا كان عنصـــر البحـــث أصـــغر مـــن عنصـــر المنتصف، فإنك في هذه الحالة تقوم بتجاهل العناصر من المنتصف وحتى العنصر الأخيـــر لأنه من غير الممكن أن يساوي أي منها عنصر البحث، حيث إن جميعها أكبر منـــه، ولـــذا تقوم بالبحث في منتصف القائمة المحصورة بين أول عنصر والعنصر في منتصف القائمـــة الكلية. ونقوم بتكرار هذه العملية حتى نجد العنصر المطلوب البحث عنه، أو حتـــى ينتهـــي البحث دون إيجاد مثل هذه العنصر، ونلاحظ هنا أن البحث الثنائي يقسم القائمة الرئيسية إلى قوائم فرعية حيث إنه في كل مرة نحاول حصر العنصر بين طرفـــي القائمـــة الأصـــلية أو الجزئية.

2.4 الخوارزمية

حسب الطريقة أعلاه تستطيع أن تضع خوارزمية كما هو موضح أدناه:

خوارزمية (2)

```
Binary Search (list, target, N)
      Begin
{1}          first =1,
{2}          last =N,
{3}        while (first ≤ last)   do
                Begin
{4}                   mid =(first +last) /2
{5}                   if target < list [mid]     then
{6}                          last=mid-1
{7}                   else if target > list  [mid]   then
{8}                            first = mid+1
                             else
{9}                             return     mid
                              end if
                         end if
             end while
{10}        return 0
       end
```

بالرجوع للخوارزمية نجد أنها تبدأ بالقائمة الرئيسة من العنصر 1 وحتى العنصر N ، ثم تقوم بتحديد عنصر الوسط mid حسب الخطوة رقم {4} . في حالة وجود العنصر المطلوب البحث عنه (target) في القائمة الصغرى أي عندما يكون أقل من عنصر الوسط حسب الخطوة {5}، يتم التحول إلى تلك القائمة الفرعية من العنصر الأول وحتى العنصر رقم mid . أما في الحالة الأخرى والتي يكون فيها عنصر البحث أكبر من عنصر الوسط، فيتم التحول نحو القائمة الكبرى والتي تبدأ عند الموقع (mid+1) وحتى الموقع الأخير. وبالطبع عندما يتساوى عنصر البحث وعنصر الوسط، يتم إرجاع موقع عنصر الوسط و (mid) على أنه الموقع المطلوب. في هذه الخوارزمية عندما لا يتم إيجاد العنصر المطلوب ضمن القائمة فيتم إرجاع قيمة (صفر).

يمكنك أن تلاحظ أن الخوارزمية تتوقف في حالة إيجاد العنصر وعندها يتم إرجاع موقع الوسط (mid) أو عندما تصبح قيمة البداية (first) أكبر من قيمة النهاية (last) ، أي عندما لا تكون هنالك قائمة متوفرة لكي يتم تقسيمها للحصول على العنصر المطلوب.

مثال (3)

أفرض أنك أعطيت القائمة التالية:

9	11	16	18	25	29	32	35

First Last

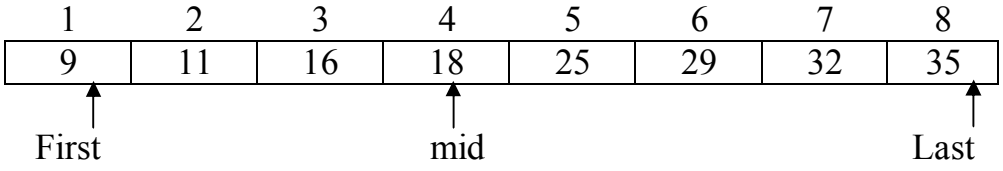
والمطلوب منك استخدام طريقة البحث الثنائي للبحث عن القيمة (25) ضمن القائمة.

الحل

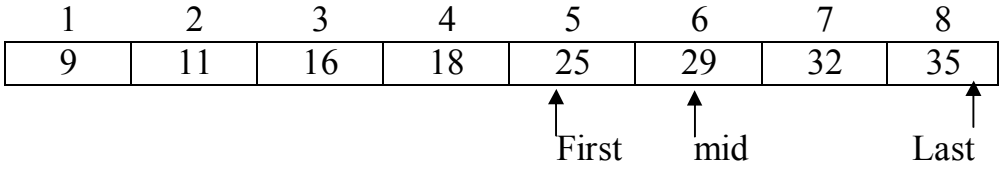
تحتوي هذه القائمة على 8 عناصر فريدة ومرتبة ترتيباً تصاعدياً، وعند البداية يتم تحديد مؤشر البداية (first=1) ومؤشر النهاية (first=8) . كما يجب تحديد عنصر الوسط بالمعادلة:

$$Mid = (first+last) / 2 = (1+8) /2 \cong 4$$

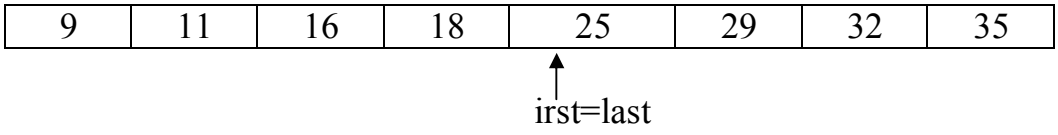
وسوف يكون mid هو الموقع الرابع بعد تقريب حاصل القسم السابقة



وبما أن عنصر البحث (25) هو أكبر من عنصر الوسط (18) ، فإن المواقع من 1 وحتى 4 لا تحتوي علي هذا العنصر، ولهذا يتم التحول إلى القائمة الجزئية من الموقع 5 وحتى الموقع الأخير 8 . مرة أخري نستطيع تحديد موقع الوسط (mid=6) وتصبح القائمة كالتالي:



وفي هذه الخطوة يتم مقارنة عنصر البحث (25) مع عنصر الوسط الجديد وهو (29) ، ونلاحظ أن عنصر البحث أصغر من عنصر الوسط، فهذا يتم التحول نحوالقائمة الفرعية الجديدة التي تبدأ من العنصر الخامس (first=5) وحتى العنصر قبل الوسط (mid=mid-1)=5) أي القائمة سوف يكون بها عنصرٌ واحدٌ لأن مؤشري البداية والنهاية متساويان.



وهنا يتم تحديد موقع عنصر الوسط بالمعادلة:

$$Mid = (first+last)\ /2 = (5+5)\ /2 = 5$$

وبهذا تصبح قيم مواقع البداية والنهاية والوسط متساوية، ومن جديد نقارن القيمة المطلوبة (25) مع الرقم الموجود عند الوسط (25) ونجدهما متساويتين فيكون الموقع الذي وجد فيه العنصر هو موقع الوسط (mid=5). اما إذا كانت القيمتان غير متساويتين فإننا نتوقف لأن قيمة البداية سوف تصبح أكبر من قيمة النهاية.

3.4 التحليل الزمني لخوارزمية البحث الثنائي

بسبب عملية التقسيم التي تتطلبها خوارزمية البحث الثنائي، سوف نفترض أن $N=2^k-1$ للعدد k . ولذا نحتاج للإجابة عن الأسئلة التالية:

ما هو عدد العناصر المتبقية للمرحلة الثانية؟ للمرحلة الثالثة ؟ عموماً إذا كان لدينا 2^i-1 عنصراً في مرحلة ما ، فيكون لدينا $2^{i-1}-1$ في النصف الأول وعنصراً واحداً للوسط و $2^{i-1}-1$ عنصراً في النصف الثاني، ولذا يتبقى للمرحلة التالية $2^{i-1}-1$ عنصراً لكل $(1\leq i \leq k)$، وهذا الافتراض سوف يسهل علينا تحليل الحالات السيئة والحالات الوسطى كما سنعرف لاحقاً.

1.3.4 تحليل الحالات السيئة

كما لاحظنا سابقاً فإن القوى الخاصة بالعدد 2 تقل بواحد لكل مرحلة في الخوارزمية، لنفترض ونفرض أن آخر مرحلة يكون فيها حجم القائمة الفرعية هو 1 فقط، أي يحدث عندما تكون $(i=1)$ (أي $2^1-1=1$)، وهذا يعني أن هنالك على الأكثر k مرحلة عند $N=2^k-1$ وحل هذه المعادلة يعطينا معدل الحالات السيئة على النحو الآتي:

$$T_{wrs}(N) = O(k) = O(\log (n-1))$$

2.3.4 تحليل الحالات الوسطى

كما في عملية البحث التتابعي هنا لدينا حالتان؛ الحالة الأولى عندما يكون العنصر موجوداً بالقائمة، بينما الحالة الثانية عندما لا يكون العنصر موجوداً ضمن القائمة.

في الحالة الأولى لدينا N موقع محتمل باحتمالات متساوية تعادل 1/N. إذا أخذنا في الحسبان شجرة ثنائية لتمثيل عملية البحث، نجد أنه لدينا مقارنة واحدة فقط لإيجاد العنصر عند الجزر، مقارنتان لإيجاد العنصر في الرأس بالمستوى الثاني وعموماً نحتاج لعدد i مقارنة، وكما هو معلوم فإن هنالك 2^{i-1} رأس بالمستوى i في الشجرة الثنائية أي عندما يكون $N=2^k-1$ فإنه يعني أنَّ الشجرة بها k مستوى، وهذا يعني أننا نستطيع حساب عدد المقارنات الكلي بجمع، حاصل ضرب عدد الرؤوس بالمستوى في عدد المقارنات بالمستوى في كل مستوى. ولذا نستطيع أن نصل للصيغة التالية لمعدل الحالات الوسطى:

$$T_{avg}(N) = \frac{1}{N}\sum_{i=1}^{k} i2^{i-1}, \qquad N = 2^k - 1$$

$$= \frac{1}{N} * \frac{1}{2}\sum_{i-1}^{k} i2^i$$

وباستخدام خواص المتسلسلات كما في الوحدة الأولى، نستطيع أن نصل إلى الصيغة التالية:

$$T_{avg}(N) = \frac{1}{N} * \frac{1}{2}\left((k-1)2^{k+1} + 2\right)$$

$$= \frac{1}{N}\left((k-1)2^k + 1)\right)$$

$$= \frac{1}{N}(k2^k - 2^k + 1$$

$$= \frac{\left(k2^k - (2^k - 1)\right)}{N}$$

$$= \frac{k2^k - N}{N}$$

$$= \frac{k2^k}{N} - 1$$

ولأن $N=2^k-1$ أي $2^k=N+1$ ، نصل إلى :

$$T_{avg}(N) = k\frac{(N+1)}{N} - 1$$
$$= \frac{k*N+k}{N} - 1$$

ولذا عندما تكون N كبيرة جداً، نصل إلى :

$$T_{avg}(N) = k-1$$

أي أن المعدل يصبح كالآتي:

$$T_{avg}(N) \cong O(\log (N+1)-1)$$

أما في الحالة الثانية والتي لا يكون العنصر موجوداً بالقائمة، يكون لدينا دائماً N حالة محتملة بأن العنصر موجوداً بالقائمة مضافاً إليها عدد N+1 حالة محتملة بأن العنصر غير موجود بالقائمة وهنالك N+1 حالة إضافية لأن عنصر البحث يمكن أن يكون أصغر من العنصر الأول أو أكبر من العنصر الأول ولكن أقل من العنصر الثاني، أو أكبر من العنصر الثاني وأقل من العنصر الثالث وهكذا حتى ...أن يكون أكبر من العنصر الأخير في الموقع N . وفي كل حالة نحتاج إلى K مقارنة لنعلم بأن العنصر لا يوجد بالقائمة. والآن هنالك 2*N+1 حالة مختلفة لها نفس الاحتمال للوقوع، ولذا نستطيع صياغة المعدل على النحو الآتي:

$$T_{avg}(N) = \frac{1}{2N+1}\left((\sum_{i=1}^{k} i2^{i-1}) + (N+1)k\right)$$

وبنفس الطريقة السابقة يمكننا تبسيط هذه الصيغة على النحو الآتي:

$$T_{avg}(N) = \frac{\left((k-1)2^k+1\right)+(N+1)k}{2N+1}$$

ونستطيع أن نصل إلي الصيغة:

$$T_{avg}(N) \cong \frac{k2^{k+1} - 2^k + 1}{2^{k+1}}$$

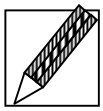
$$\cong k - 1/2$$

والآن يصبح المعدل المطلوب كالآتي:

$$T_{avg}(N) = O\left(\log(N+1) - 1/2\right)$$

ونستطيع أن نلاحظ أنَّ هذا المعدل أكبر من المعدل السابق بنحو ½ فقط، ولهذا يمكننا تجاهل هذا الاختلاف بالنسبة للأعداد الكبيرة للحجم N.

تدريب (2)



1) نفذ عملية بحث ثنائي على القائمة التالية:

9	21	29	47	65	71	98	107	137	150

وذلك لإيجاد العنصر 98.

2) أعد كتابة خوارزمية البحث الثنائي مستخدماً الاستدعاء الذاتي. وهل كفاءة الخوارزمية الجديدة تختلف عن كفاءة الخوارزمية الأساسية؟

نشاط

أكتب دالة وفقاً لخوارزمية البحث الثنائي وثم قم باستخدامها في برنامج حاسوب لاختبارها، ثم قم بتوليد أعداد عشوائية من المدى 1 وحتى N لاستخدامها في البرنامج . خذ N كعدد من المدى 100 وحتى 100 . أضف للبرنامج متغيراً عاماً لحساب عدد المقارنات، حيث يجب تجهيزه بالقائمة الصفرية منذ البداية وتحديثه متي تمت عملية مقارنة لغرض البحث الثنائي. بعد عمل ذلك قم باستدعاء دالة البحث من داخل البرنامج وقم بتنفيذه للبحث عن جميع الاعداد العشوائية السابقة من الموقع 1 وحتى الموقع N . قم بحساب العدد الكلي للمقارنات مقدماً على N والذي يعطي تقديراً لمعدل الحالات الوسطى.

الخلاصة

عزيزي الدراس،

أرجو أن تكون قد حققت ما هدفنا إليه في هذه الوحدة وتعرفت على عملية البحث عن عنصر معين ضمن مجموعة من العناصر. فقد قلنا إن البحث ضمن البيانات المرتبة يساعد على إتمام عملية البحث بكفاءة أكبر، وكذلك فإن البحث عن البيانات عملية ملازمة لعملية معالجة البيانات بحفظها واسترجاعها بسهولة وسرعة.

ثم بيّنا من أهم العوامل التي تؤثر على خوارزمية البحث، الوقت، ومدة التنفيذ والمساحة وخواص البيانات المراد البحث عنها.

ثم ناقشنا كيفية البحث عن البيانات وافترضنا أن البيانات عبارة عن سجلات محفوظة في شكل مصفوفات وكل سجل له موقع محدد داخل القائمة وفهرست هذه المواقع، وعرفنا أن كل مجال له مفتاح بواسطته تتم عملية البحث عن أي قيمة تتبع لهذا المجال.

ذكرنا أن السجلات غير المرتبة تستخدم طريقة البحث التتابعي للبحث عن سجل فيها، وهي طريقة سهلة جداً، قمنا سوياً بشرح هذه الطريقة ثم وضعنا صياغة للخوارزمية المستخدمة ومثلنا لهذه الطريقة لندعم شرحنا وقد قدمنا نوعين من التحليل الزمني وهما تحليل الحالات السيئة وتحليل الحالات الوسطى.

أما البيانات المرتبة فتنتج معها طريقة البحث الثنائي وفيها يبدأ البحث عند منتصف القائمة، وهذه الطريقة تقسم القائمة الرئيسية إلى قوائم فرعية حتى يتم حصر العنصر المراد البحث عنه بين طرفي القائمة الجزئية.

أرجو عزيزي الدراس أن تكون قد أستفدت من هذه الوحدة الهامة خاصة أن البحث عملية لا يستغني عنها أي برنامج حاسوبي.

ناقش ما لم تفهمه مع زملائك أو مشرفك الأكاديمي أثناء اللقاءات حتى تكتمل الاستفادة.

وفقك الله لما فيه خيرك وخير أمتك.

لمحة مسبقة عن الوحدة التالية

عزيزي الدارس، التصميم هو أساس كل شئ نريد بناءه قبل مرحلة تنفيذه وهذا ما سنناقشه في وحدتنا التالية، **فهيا إلى الوحدة السادسة وهي تصميم الخوارزميات.**

مسرد المصطلحات

- **بحث Searching**

تحديد موقع عنصر معين ضمن مجموعة من العناصر. قد تكون البيانات مرتبة حسب مفتاح معين أو غير مرتبة.

- **البحث التتابعي Sequential Search**

وتستخدم هذه الطريقة عندما تكون السجلات مخزنة دون اعتبار للترتيب. أو عندما تكون وسيلة التخزين لا توفر أي نوع من طرق الاسترجاع المباشر أو المفهرس، وتتلخص هذه الطريقة في أخذ كل سجل من سجلات الملف على حده بدءاً بالسجل الأول في القائمة.

- **البحث الثنائيBinary Search**

في هذه الطريقة تكون القائمة مرتبة، وتتلخص علي النحو الآتي: تبدأ عملية البحث عند منتصف القائمة، فإذا كان عنصر البحث أصغر من عنصر المنتصف، فإنك في هذه الحالة تقوم بتجاهل العناصر من المنتصف وحتى العنصر الأخير.

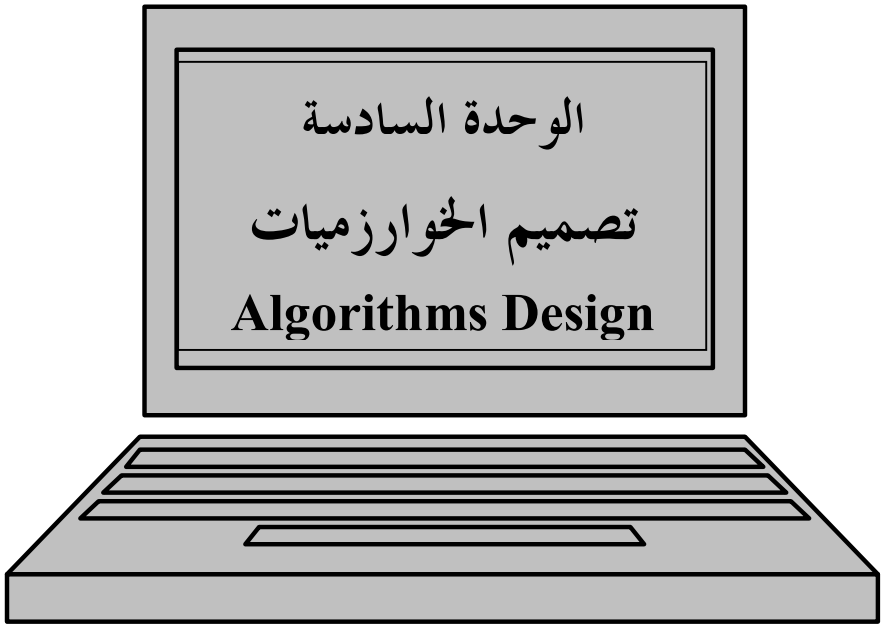
المراجع

المراجع العربية

1) مجموعة مؤلفين، تركيب البيانات وتصميم الخوارزميات، منشورات جامعة القدس المفتوحة، 1998.
2) السماني عبد المطلب، هياكل البيانات، منشورات جامعة السودان المفتوحة، 2005.

المراجع الأجنبية

1) Weiss, M.A., Data Structures and Alogrithm Analysis, Benjamin pub., 1992.
2) McConnell, J.J., Analysis of Alogrithms: An Active Approach, Jones pub., 2001.
3) Lavitin, A., Introduction to the Design and Analysis of Alogrithms, Addisions ues by, 2003.
4) Ulman, J.D (etal), The Design and Analysis of Compurte Alogrithms, Addison Wesley, 1974.
5) Basese, S., Compurte Alogrithms : Introduction to Analysis and Design, Addison Wesley, 2000.
6) Sedgewick, R., An Introduction to the Analysis of Alogrithms, Addison Wesley 1996.


الوحدة السادسة
تصميم الخوارزميات
Algorithms Design

محتويات الوحدة

المقدمة

تمهيد

عزيزي الدارس! مرحباً بك إلى الوحدة السادسة من هذا المقرروهـــي بعنـــوان: تصميم الخوارزميات.

تتكون هذه الوحدة من أربعة أقسام رئيسية؛ في القسم الأول منها ســـوف نقـــوم بتوضيح المعنى العام لتصميم الخوارزميات والتعرف على الأسباب التي تدعونا لدراسة طرق التصميم المختلفة. أما القسم الثاني فيركز علـــى طريقـــة الخوارزميـــة الجشـــعة والتعرف على المنهج الذي تتبعه. وفي القسم الثالث سوف نقوم بالتعرف علـــى مـــنهج الخوارزمية الفعالة ومعرفة بعض الأمثلة حوله، أما في القسم الرابع والأخيـــر، فســـوف نقوم بدراسة طريقة فرق تسد في تصميم الخوارزميات والتعرف علـــى كيفيـــة تحليـــل الخوارزميات المختلفة والتي تتبع هذا الأسلوب.

وقد زودناك بباقة من الأنشطة والتدريبات وأسئلة التقويم الذاتي والتي نتمنى أن تستفيد منها كما نتمنى أن تشاركنا في نقد هذه الوحدة وتقويمها.

أهداف الوحدة

عزيزي الدارس،

بعد انتهائك من هذه الوحدة أتوقع أن تكون قادراً على أن:

- ✓ **توضح** المعنى العام من تصميم الخوارزميات.
- ✓ **تصميم** وتحليل الخوارزميات الجشعة.
- ✓ **تصميم** وتحليل الخوارزميات الفعالة.
- ✓ **تصميم** وتحليل خوارزميات فرق تسد.

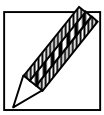
1. ما المعنى العام من تصميم الخوارزميات؟

إذا وضعنا سؤال على النحو الآتي: مع معرفتنا جميع مكونات حل مسألة معنية، كيف نقوم بتصميم خوارزمية لحل هذه المسألة؟

وهذا سؤال هام سوف نحاول الإجابة عليه بدراسة بعض طرق التصميم المختلفة للخوارزميات، ولكن دعنا نسأل: ما هي طريقة تصميم الخوارزميات؟

طريقة تصميم الخوارزميات هي منهج أو أسلوب عام لحل المسائل والذي يمكن أن يكون فعالاً مع عدد كبير من المسائل والتي تقع ضمن مجالات مختلفة في علوم الحاسوب، وهنالك عدد لا بأس به من الطرق المختلفة في تصميم الخوارزميات، ولكل طريقة أسلوبها المتميز في حل المسائل. وسوف نقوم بالتعرض لثلاثة أنواع هامة في هذا الكتاب؛ النوع الأول يعتمد على التكرار الممرحل لحل المسائل ويطلق عليه الخوارزمية الجشعة، أما النوع الثاني فيعتمد على تسجيل حالات المسألة والاستمرار في الحل دون الاستعانة بالاستدعاء الذاتي للحل ويطلق عليه البرمجة الفعالة، أما في النوع الثالث فالمنهج يعتمد على تقسيم الحل إلى عدد من الحلول الفرعية وبتجميع تلك الحلول يفترض الوصول إلى الحل النهائي للمسألة ويطلق عليه خوارزمية فرق تسد، ودراسة هذه الطرق المختلفة له سببان هامان: السبب الأول من دراسة هذه الطرق هو مدنا بكيفية توجهنا لتصميم الخوارزميات لحل مسألة جيدة، أو بصورة أوضح مسألة لا نجد لها خوارزمية حل واضحة عندئذ، ودعنا نعرف بأنه ليس صحيحاً أن تكون هذه الطرق صالحة لحل جميع المسائل التي نتعرض لها، ولكن دراستها تمكننا من جمع أدوات هامة لاستخدامها في حل المسائل العلمية المختلفة. أما السبب الثاني فيلعب دوراً هاماً في تصنيف علم الحاسوب والذي تمثل الخوارزميات أساسه الذي يقوم عليه إن دراسة الطرق المختلفة لتصميم الخوارزميات تمكننا من تصنيف الخوارزميات وفقاً للمنهج التصميمي المتبع، ولذلك تصبح أسلوباً طبيعياً في تنوع ودراسة الخوارزميات المختلفة.

تدريب (1)



(1) ما هي الصيغة الصحيحة من الصيغ أدناه والتي تعبر عن خوارزمية إيجاد مساحة المثلث الذي له ثلاثة أضلاع. c, b, a ؟:

أ- الصيغة الاولى:

$$S = \sqrt{p(p-a)(p-b)(p-c)}$$

حيث:

$$p = (a+b+c)/2$$

ب- الصيغة الثانية:

$S = \frac{1}{2} b\ c \sin A$

حيث A هي الزاوية بين الضلعين c و b .

ج- الصيغة الثالثة:

$S = \frac{1}{2} a\ h_a$

حيث h_a هو الارتفاع على القاعدة a .

(2) قم بتصميم خوارزمية لحساب جزري المعادلة $ax^2 + bx + c$.
يمكنك الاعتماد على دالة معرفة مسبقاً لحساب الجزر وهي sqrt(x).

(3) قم بوصف خوارزمية للحصول على التمثيل البياني للعدد الصحيح الموجب العشري.

2. الخوارزمية الجشعة (Greedy Algorithm)

1.2 الطريقة

هذه الطريقة يمكن استخدامها في حل المسائل التي تبحث عن الحل المثالي. تقوم هذه الطريقة على بناء حل مكون من عدة مراحل متتابعة، كل مرحلة تقوم بتقريب الحل إلى الحل الأمثل النهائي وفي كل مرحلة يجب أن يكون الحل الذي تم اختياره:

أ) **معقول** : أي بمعنى أنه يتوافق مع قيود المسألة المراد حلها.

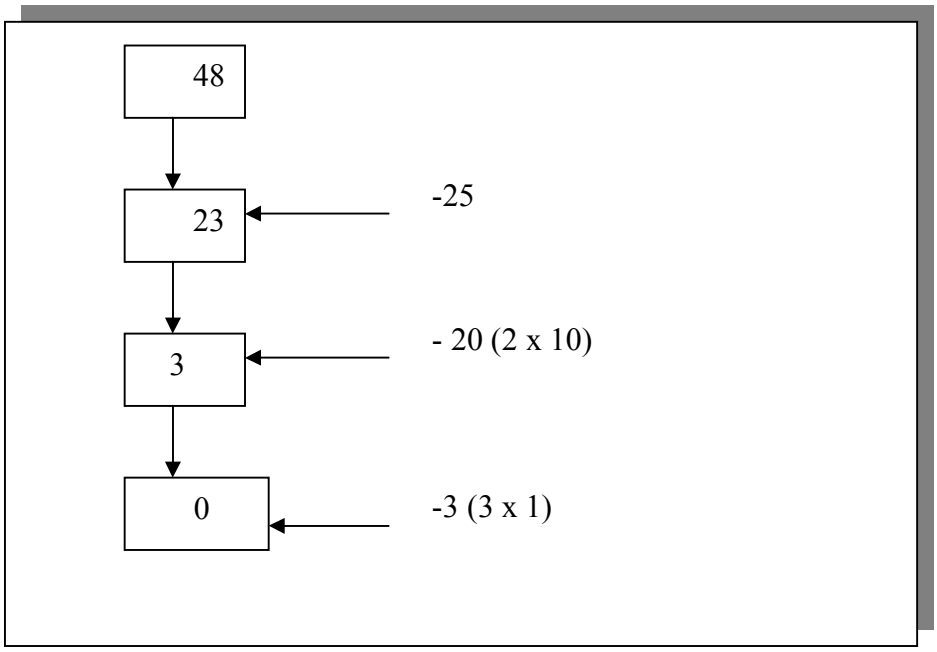
ب) **حل أمثل موضعي**: أي هو الحل الأفضل بين جميع الحلول المعتدلة في هذه المرحلة.

جـ) **لا يمكن استرجاعه**: أي هو حل عند اختياره لا يمكن تغييره في المراحل القادمة من الخوارزمية.

ونلاحظ أن هذه المتطلبات الثلاثة توضح لماذا تم إطلاق " **جشعة** " على هذه الطريقة. في كل مرحلة، تقترح الطريقة الجشعة أفضل الحلول البديلة على أمل أن الاختيار المتتابع للحلول أمثلية الموضع قد تقود في النهاية إلى الحل الأمثل النهائي للمسألة.

وفي الواقع هنالك أمثلة كثيرة لمسائل نقوم باستخدام هذه الطريقة في إيجاد حلولها. فمسألة باقي العملة والتي تواجه المحاسبين تعتبر أحد هذه المسائل، ويمكن التعبير عن المسألة كالآتي: المطلوب وضع باقي العملة لمبلغ، مثلاً n ، وإعطاء أقل عدد ممكن من العملات المتاحة: c_1 أقل من c_2 أقل من c_3 ...أقل من c_m . ومثال لذلك إذا كان لدينا العملات (c_1=1) و (c_2=10) و (c_3=15) و (c_4=25)، وكان لدينا مبلغ قدره (n=48) . فكيف يمكننا توزيع هذا الباقي على العملات وذلك للحصول على أقل عدد من العملات؟ إذا توصلت إلى الحل الذي يعطي **ربع واحد وعشرتين وثلاثة وحدات** تكون قد أتبعت هذه الطريقة الجشعة للحصول على هذا الحل، ولتوضيح أنك في المرحلة الأولى سوف تحاول توزيع المبلغ على أكبر عملة وذلك بغرض تقليل

الباقي إلى 23 وهنا سوف تختار ربعاً واحداً، أما في المرحلة الثانية لا يمكن اختيار ربع ثاني لأننا سوف نخرق شروط المسألة لأن المبلغ هو 48 فقط . ولذا سوف نتحول للعملة الثانية وذلك لتقليل الباقي إلى 3 وحدات فقط، ولذا سوف نحتاج أن نختار عشرتين في هذه المرحلة. وفي المرحلة الثالثة والأخيرة لا يكاد يتبقى سوى ثلاثة وحدات ولذا سوف يتم اختيارها بغرض الوصول إلى الحل النهائي. (أنظر الشكل (1)).



شكل رقم (1)

في الأقسام الجزئية القادمة سوف نقوم بدراسة بعض تطبيقات الخوارزمية الجشعة. التطبيق الأول هو مسألة الجدولة البسيطة، أما في التطبيق الثاني فسوف نتناول استخدام الخوارزمية الجشعة في تقليص الملفات بواسطة رموز هفمان.

2.2 مسألة الجدولة البسيطة

إذا أعطيت المهام $j_1, j_2, j_3, \ldots, j_n$ والمعلومة وقت الإنجـــاز $t_1, t_2, t_3, \ldots, t_n$ على التوالي. وكان لدينا معالج واحد فقط. ما هي طريقة الجدولة المثلى بحيـــث يـــتم تقليل متوسط وقت الإكمال؟

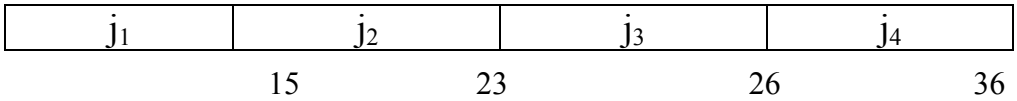
خذ مثلاً المهام الموضحة في جدول (1) أدناه:

المهمة	الوقت
j_1	15
j_2	8
j_3	3
j_4	10

جدول (1) المهام والوقت

أول جدولة محتملة هي الموجودة في الشكل (2) أدناه والتي تعبر عن الجدولـــة حسب الترتيب، والتي تعطينا متوسط وقت إكمال قدره:

$$C = \frac{15 + 23 + 26 + 36}{4} = 25$$

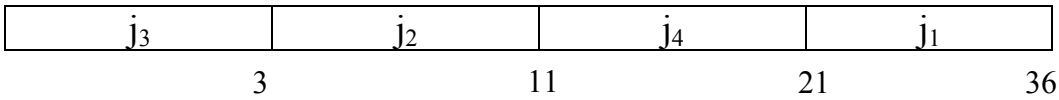


شكل (2) الجدولة الأولى

أما أقل جدولة يمكن الحصول عليها موضحة في الشكل (3) أدناه، وتعطي متوسط وقت إكمال قدره:

$$C = \frac{3 + 11 + 21 + 36}{4} = 17.75$$

ونلاحظ أن هذه الجدولة الأخيرة تقوم على مبدأ الأقصر أولاً، وهـــذه الجدولـــة تعطينا الحل الأمثل لهذه المسألة.



شكل (3)

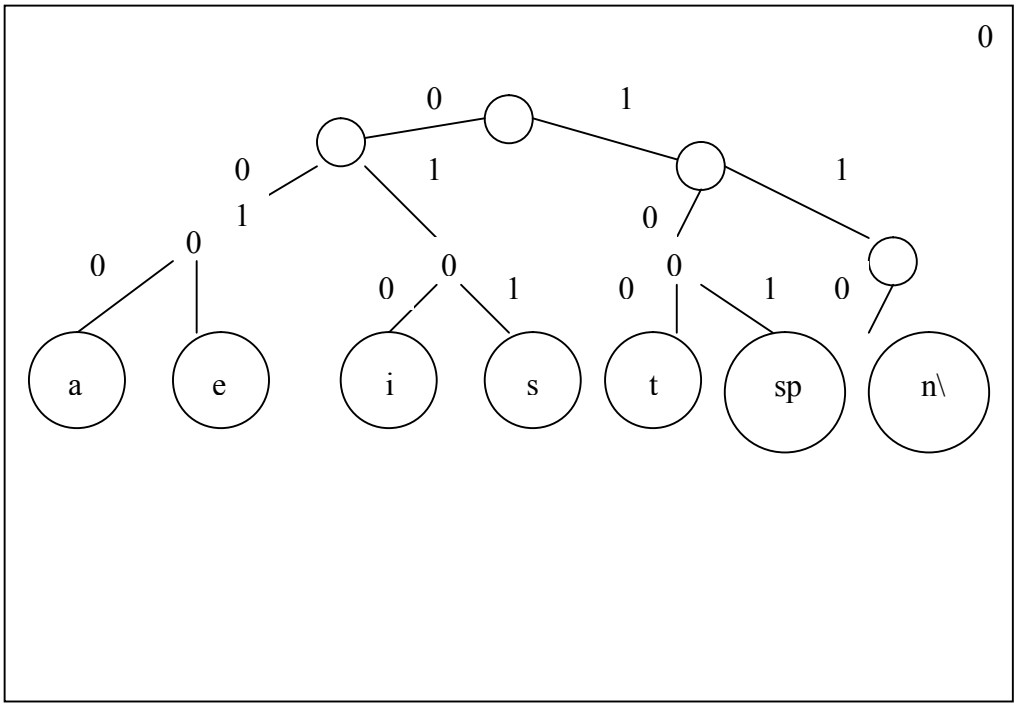
3.2 مسألة تقليص الملفات بواسطة رموز هفمان

من المعروف أننا نحتاج إلى 8 خانات ثنائية (bits) لتمثيل الحروف والرموز في أي ملف نصي (Text file) . إذا فرض أن لدينا ملفاً يحتوي فقط على الحروف t,s,i,e,a بالإضافة للرمزين sp والتي تعادل الفراغ و n والتي تعادل سطر جديد. ولنفترض أننا نود معرفة حجم الخانات الثنائية بالملف والتي تعبر عن شكل الحروف والرموز وذلك حسب ورودها داخل الملف، خذ مثلاً جدول (2) والذي يحتوي على الحرف وتكراره بالملف والرمز الثنائي المقابل له وعدد الخانات الثنائية المطلوبة لتمثيله داخل الملف.

الحرف	التكرار	الرمز	عدد الخانات
a	10	000	30
E	15	001	45
I	12	010	36
S	3	011	9
T	4	100	12
Sp	13	101	39
N	1	110	3
المجموع			174

جدول (2) استخدام الترميز القياسي

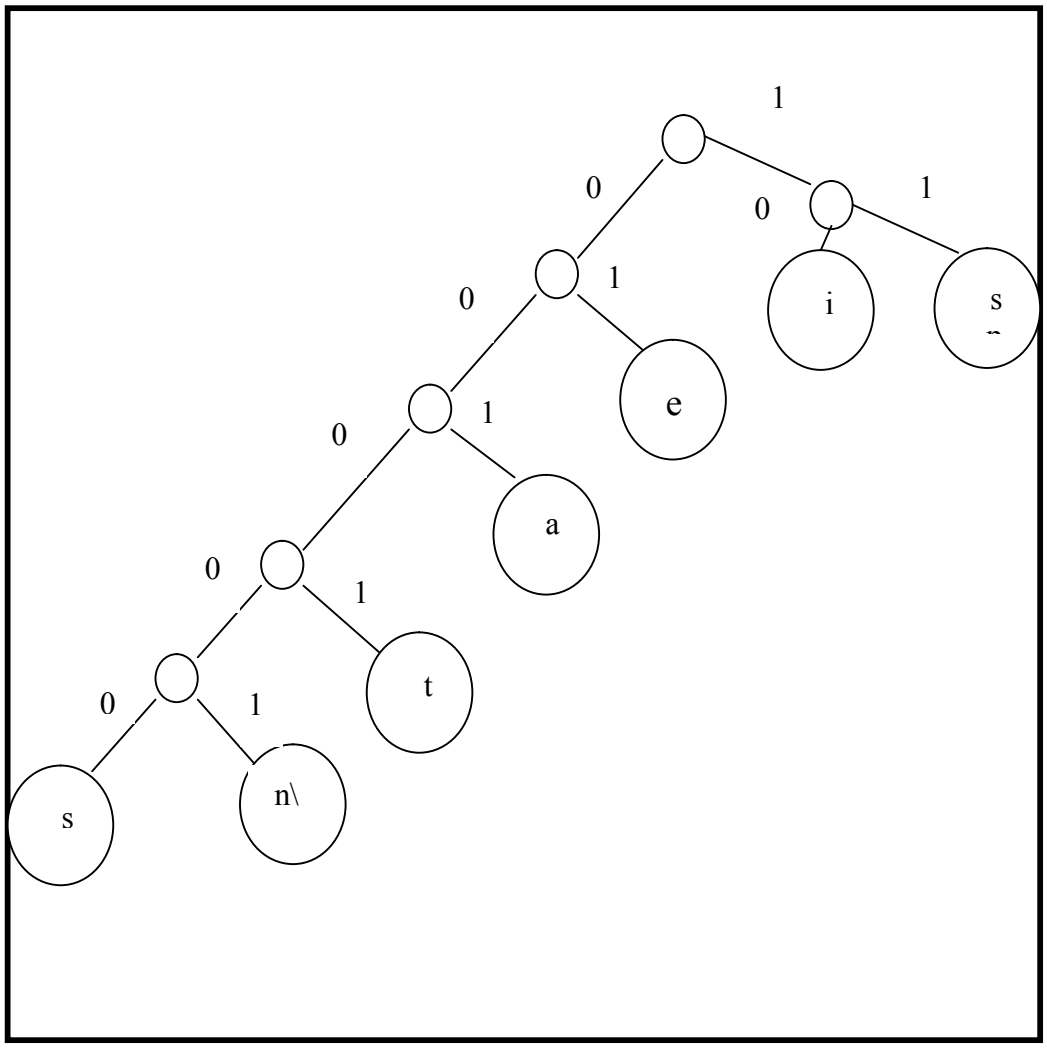
وهنالك طريقة قياسية يمكننا توضيحها على شجرة ثنائية تحتوي على الحروف في الأوراق كما هو الحال في الشكل (4) أدناه:



الشكل (4) تمثيل الحروف بالطريقة القياسية

ونلاحظ أن تمثيل أي حرف برمز ثنائي وذلك عن طريق تسجيل الرموز الثنائية باستخدام (0) للفرع اليسار و (1) للفرع اليمين، مثلاً الحرف s يمكننا تسجيل 0 ثـم 1 وثم 1 يكون 011 .

والآن نود الحصول على تمثيل مثالي لحروف الملـف ليعطينـا أقـل حجـم، أي ضرورة تعديل الشجرة الثنائية. خذ الشكل (5) والذي يعبر عن شـجرة ثنائيـة مثاليـة لتمثيل حروف الملف.



الشكل (5) شجرة ثنائية قياسية

وبواسطة هذه الشجرة يمكننا الوصول لتمثيل حروف الملف والحصول على حجم كلي قدره 146 كما موضح في جدول (3) أدناه.

الحرف	التكرار	الرمز	عدد الخانات
a	10	011	30
E	15	01	30
I	12	10	24
S	3	00000	15
T	4	0001	16
Sp	13	11	26
nl	1	00001	5
المجموع			146

جدول (3) استخدام التمثيل المثالي

الآن السؤال هو كيف يمكننا إنشاء هذه الشجرة الثنائية وذلك للحصول على هـذا التمثيل المثالي؟

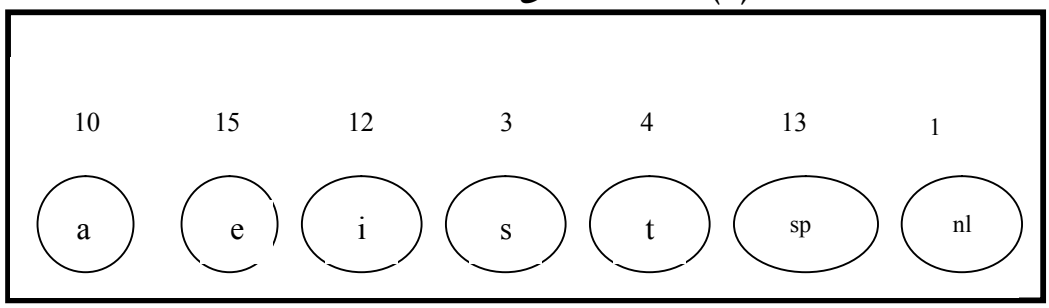
الإجابة عن السؤال ببساطة أن هنالـك خوارزميـة مشـهورة يطلـق عليهـا **خوارزمية رموز هفمان** والتي يمكننا تلخيصها على النحو الآتي:

الخطوة الأولى: أنشئ n شجرة أحادية الرأس وفقاً لعدد الحروف. قم بتسجيل تكرار أي حرف في جزر الشجرة للإشارة على ثقل الشجرة أو بصورة أخرى يعبر هذا الثقل عـن مجموع التكرارات في أوراق الشجرة.

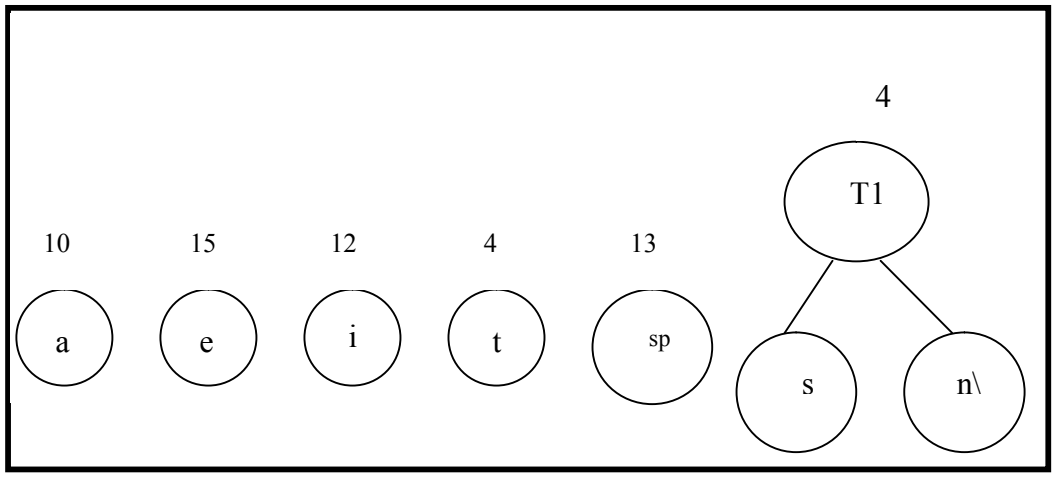
الخطوة الثانية: قم بتكرار هذه العملية وحتي الوصول إلى شجرة واحدة فقط، تـم جـد شجرتين لهما أقل ثقل وقم بدمجهما في شجرة واحدة لها فرعان علـى اليمـين وعلـى اليسار ثم قم بتسجيل مجموع ثقليهما في جزر الشجرة الناتجة.

والآن سنقوم باستخدام الخوارزمية في محاولة لإنشاء الشجرة الثنائية كما هـي موضحة في شكل (5) ففي المرحلة الأولى سوف يكون لدينا 7 شجرات أحادية الـرأس موضحة في شكل (6) ثم نقوم بدمج الشجرتين ذواتي أقل ثقل لتكـوين الشـجرة T1 . وهنا هما الشجرتان اللتان تمثلان الحرف s والرمز n كما هو موضح في الشكل (7).

شكل (6) المرحلة الأولى من خوارزمية هفمان:

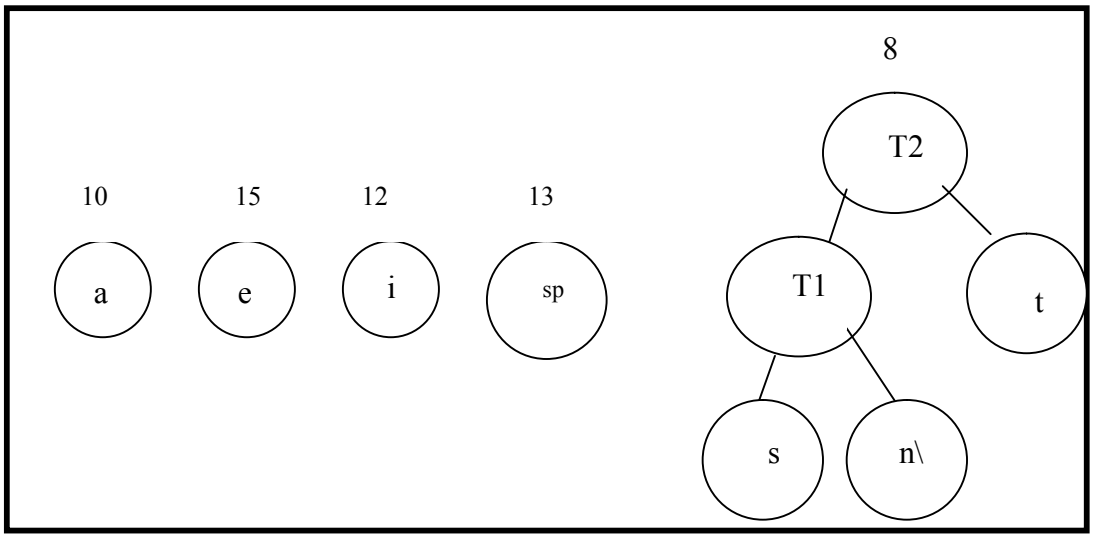


شكل (7) خوارزمية هفمان بعد المرحلة الأولى

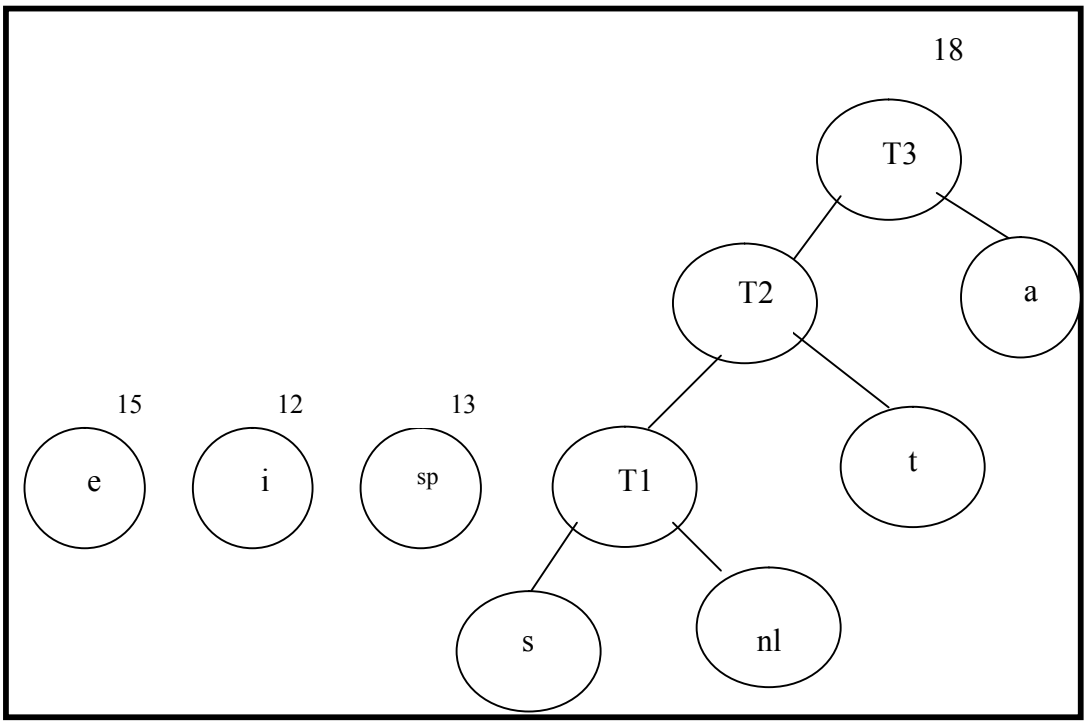


والآن لدينا 6 شجرات، ويمكننا اختيار الشجرتين T1 و t وذلك لدمجهما لتكوين الشجرة T2 كما هو موضح في الشكل (8). ثم نقوم باختيار الشجرتين T2 و a لدمجهما في المرحلة التالية كما هو موضح في الشكل (9).

شكل (8) خوارزمية هفمان بعد المرحلة الثانية



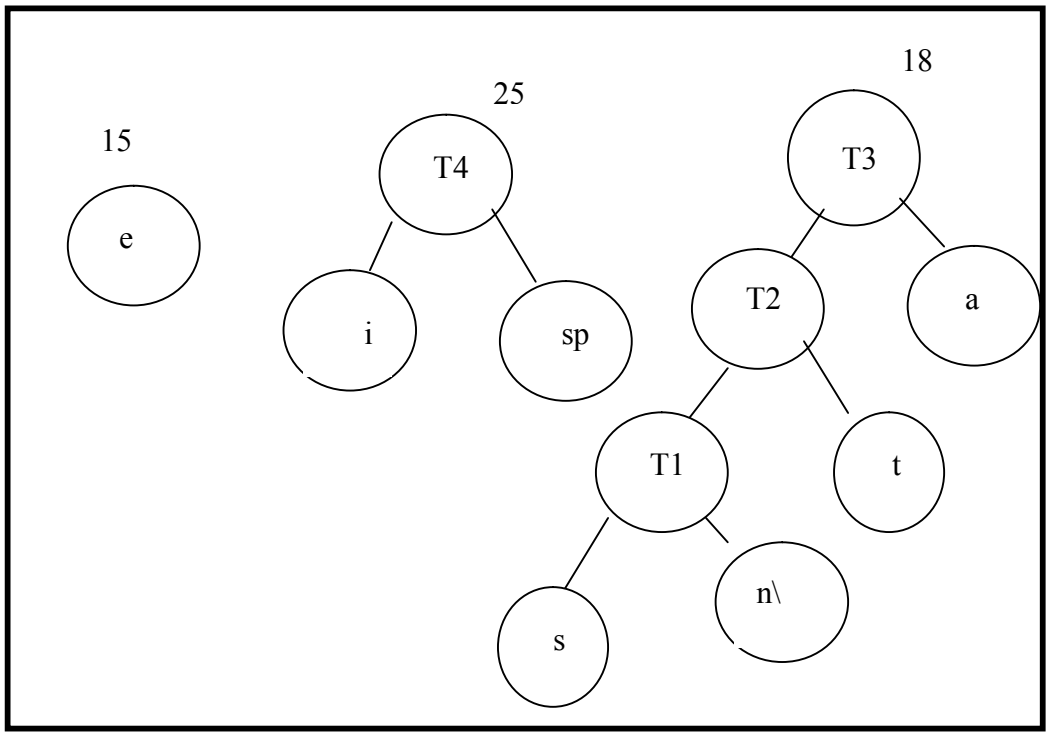
الشكل (9) خوارزمية هفمان بعد المرحلة الثالثة



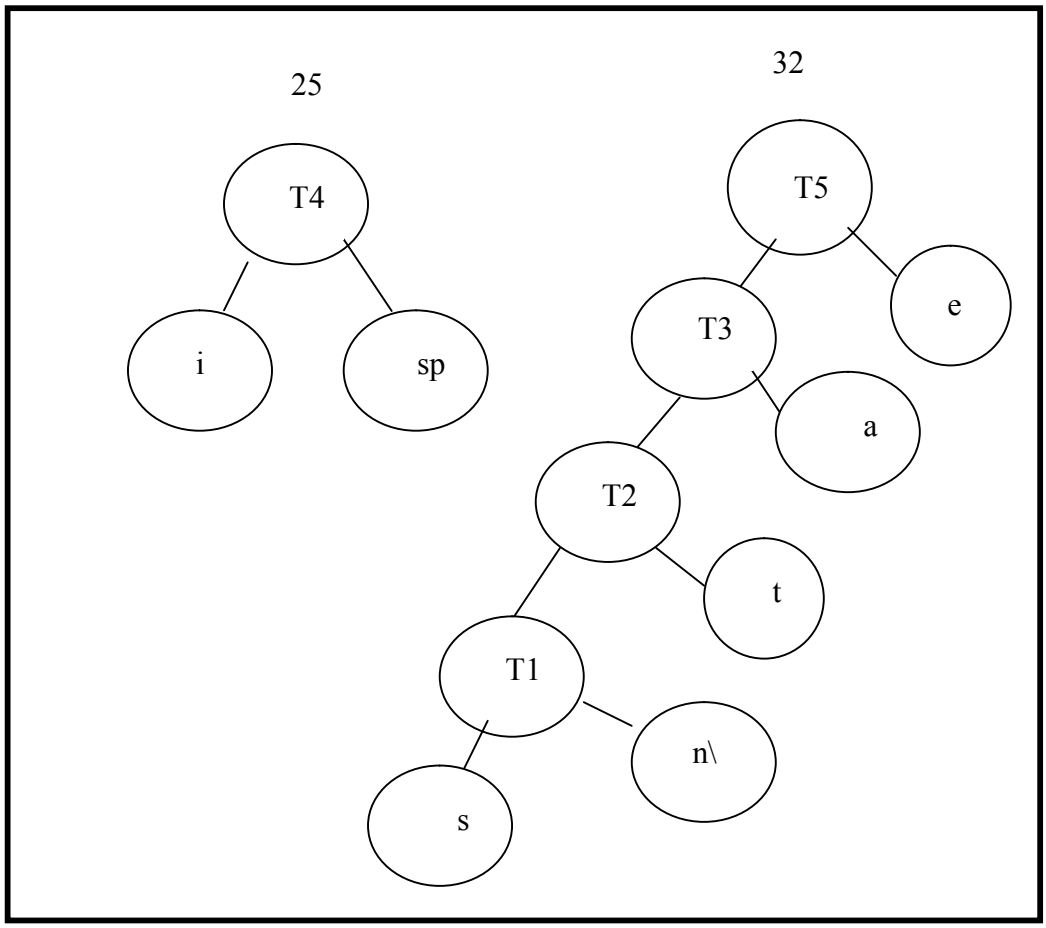
والآن يمكننا الاستمرار في تنفيذ الخوارزمية، وفي المرحلة التالية يمكننا دمج الشجرتين i و sp لتكوين الشجرة T4 كما هو موضح في الشكل (9).

ثم نقوم بدمج الشجرتين T3 و e لتكوين الشجرة Ts كما هو موضح في الشكل (11). وفي المرحلة الأخيرة نقوم بدمج الشجرتين المتبقيتين Ts و T4 لتكوين الشجرة T6 وهي نفس الشجرة الثنائية المطلوبة كما هو موضح في الشكل (12).

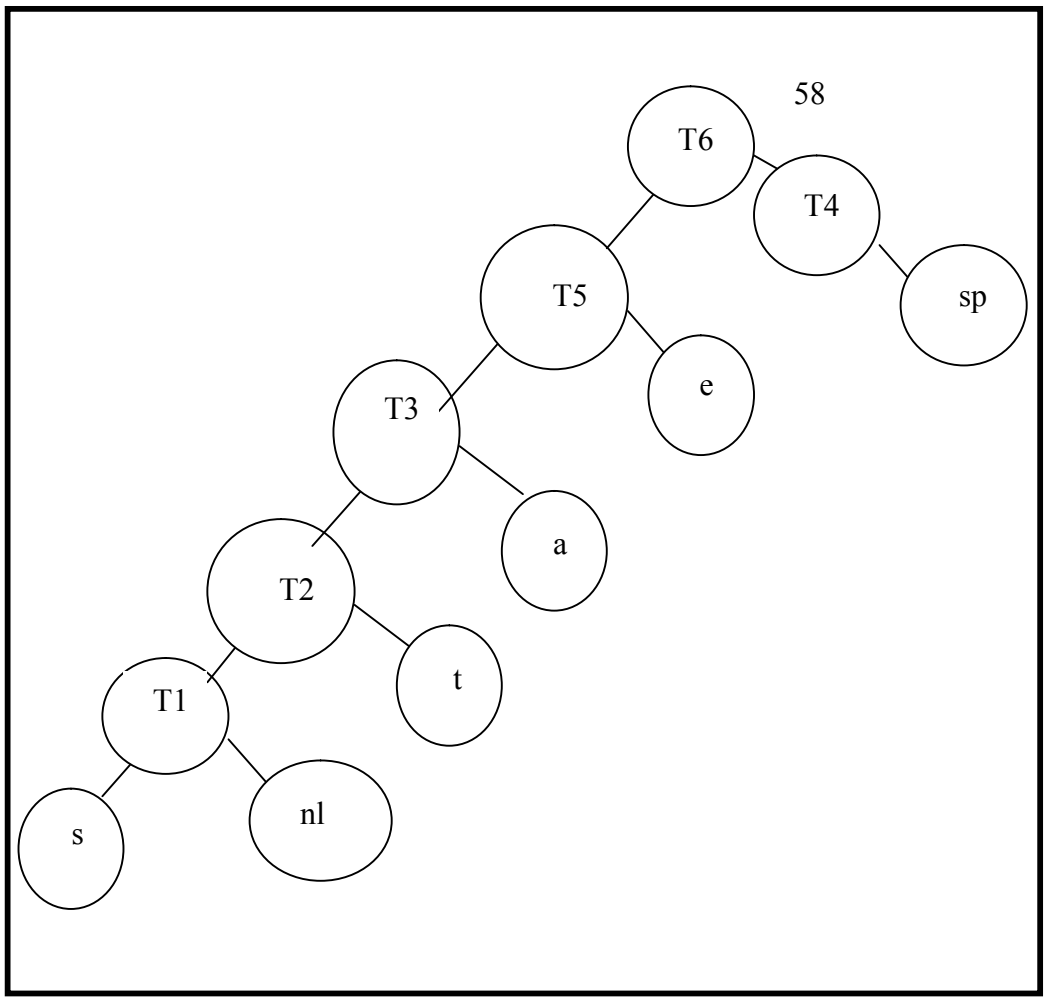
شكل (10) خوارزمية هفمان بعد المرحلة الرابعة



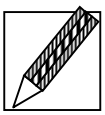
شكل (11) خوارزمية هفمان بعد المرحلة الخامسة



شكل (12) خوارزمية هفمان بعد المرحلة الأخيرة



تدريب (2)



1) إذا كان هنالك 12 مدخل عبارة عن المهام $j1, j2, j3, \ldots, jn$ والتي فيها تحتاج أي مهمة إلى وحدة زمنية واحدة لتكتمل. أي مهمة ji تكسب di دينار إذا اكتملت عند نهاية زمنية قدرها ti ولا تكتسب أي مال إذا اكتملت بعد هذه النهاية. قم بإعطاء خوارزمية جشعة لحل هذه المسألة ذات معدل زمني قدره $O(n^2)$.

2) ملف يحتوي على رمز الفراغ (sp) و الفاصلتين (:) و (,) والسطر الجديد (nl) والأرقام من صفر وحتى 9، كل رمز أو رقم تكراره في الملف على النحو: الفراغ (605) والفاصلة (:) (100) والفاصلة (,) (705) والسطر الجديد (100) و 0 (43) و 1 (242) و 2(176) و 3(59) و 4(185) و 5(250) و 6(174) و 7(199) و 8(205) و 9(217). قم بإنشاء رموز هفمان.

نشاط

إذا أعطيت نظاماً للعملة ذا قيم (من أصغر إلى أكبر) $c_1, c_2, c_3, \ldots, c_n$ وحدة مالية.

قم بكتابة خوارزمية لحساب أقل عدد من العملات لإعطاء k كباقي. وثم قم بتنفيذها على الحاسوب وقم باختيار قيم مناسبة للعددين k, n و قيم العملات المختلفة.

3. البرمجة الفعالة (Dynamic Programming)

1.3 الطريقة

تعتبر خوارزمية البرمجة الفعالة خوارزمية بديلة لخوارزمية الاستدعاء الذاتي، وفي هذه الخوارزمية يتم تسجيل حلول المسائل الفرعية للمسالة الرئيسة، والتي يسبق حلها بالاستدعاء الذاتي، واستخدامها بالتوالي وحتى الوصول إلى الحل النهائي للمسألة، ومن الأمثلة القريبة التي يمثل حلها توضيحاً لهذه الخوارزمية مسالة إيجاد المضروب، فمعلوم أن مضروب العدد يعرف كالآتي:

$$N! = n(n-1)\ ,\ \ 0! = 1$$

أي أن مضروب أي عدد غير الصفر يساوي العدد نفسه في ناتج تكرار نفس العملية أي بمعنى استدعاء نفس الدالة. إذا طبقنا خوارزمية الاستدعاء الذاتي يكون التغيير على النحو الآتي:

سوف نقوم بتسجيل ناتج عملية إيجاد المضروب في مصفوفة a[i] ، حيث يتم تسجيل أول مضروب !0 عند a[0] وهو 1 ثم يتم الحصول على مضروب !1 بالمعادلة a[1]=1*a[0] و هكذا مضروب !2 بالمعادلة a[2]=2*a[1] . أي تم تغيير في الصيغة لتصبح استدعاء لقيمة المضروب السابقة والمسجلة دون استدعاء الدالة نفسها. سوف نقوم بتوضيح مثالين لترسيخ المفاهيم السابقة، في المثال الأول سوف نوضح كيف يتم تطبيق هذه الخوارزمية لتعديل حل مسألة أرقام فايبونسي وبالتالي، تقليل المعدل الزمني السابق والذي تم الحصول عليه بواسطة حل المسالة بواسطة الاستدعاء الذاتي، وفي المثال الثاني سوف نقوم باستخدام هذه الخوارزمية لحل مسألة ايجاد معاملات ذي الحدين.

2.3 مسألة أرقام فايبونسي

سبق أن قمنا بوضع حل لمسالة أرقام فايبونسي في الوحدة الثالثة وذلك للحصول على معدل زمني T(a) قدره $O(2^n)$ بواسطة الاستدعاء الذاتي بالصيغة:

$$T(n) \geq T(n-1) + T(n-2)$$

دعنا نسترجع أن دالة فايبونسي تعرف على النحو:

$$F_n = F_{n-1} + F_{n-2} \ , \ F_0 = F_1 = 1$$

الآن سوف نقوم بوضع حل جديد مبني على تسجيل ناتج الدالة والاستفادة منه في الحصول على ناتج الدالة لقيم n العليا. ولشرح هذا الحل خذ المثال العملي التالي: للحصول على F_4 مثلاًيجب الحصول أولاً على F_2, F_3 بالطبع وحسب التعريف، ولذا للحصول على F_3 يجب الحصول F_1 , F_2 . للحصول على F_2 يجب الحصول على F_0, F_1 ، والآن يجب أن نبدأ بالعكس اي تسجيل قيمة F_0, F_1 في مصفوفة F[0], F[1] ومنها يمكننا صياغة معادلة :

$$F[2] = F[1] + F[0]$$

كما يمكننا الاستمرار بنفس الكيفية للحصول على بقية الأرقام. ولذا يمكننا تعديل الخوارزيمة السابقة بالخوارزمية الجديدة أدناه:

خوارزمية (1)

```
Fibonacci (n)
Begin
    if (n=0)  or (n=1)    then
                fib = 1
       else
              Begin
                   Last =1
                   Next_to_last_1
                   for i=2   to  n  do
                     Begin
                          Fib=last + next_to_last
                           Next_to_last = last
                           Last = fib
                    end for
       end if
    return  fib
end
```

كما يمكننا أن نلاحظ أن المعدل الجديد هو معدل خطي O(n)، وهو بالطبع أفضل من المعدل السابق. وهذا يوصلنا إلى نتيجة هامة وهي أن استخدام هذا الأسلوب يقودنا إلى تقليل المعدلات الزمنية البطيئة التي تسببها خوارزميات الاستدعاء الذاتي.

3.3 مسألة إيجاد معامل "ذو الحدين"

يمكننا استرجاع أن معامل "ذو الحدين" c(n,k) هو عدد التوافيق (أي المجموعات المتوافقة) والناتجة من اختيار k عنصراً من n عنصر F . كما أن مسمى "ذو الحدين" أتى من مساهمة هذا المعامل في مفكوك ذو "الحدين" بالصيغة:

$(a+b)^n = c(n,0)\ a^n + \ldots + c(n,i)\ a^{n-i}\ b^i + \ldots + c(n,n)\ b^n$

حيث أن معامل "ذو الحدين" c(n,k) له الخاصية التالية:

$$c(n,k) = c(n-1,k-1) + c(n-1,k)$$
$$c(n,0) = c(n,n) = 1$$

وهذه الخاصية والتي توضح أنه يمكن إيجاد المعامل c(n,k) بدلالة ناتج المسألة الفرعية لإيجاد المعاملين c(n-1, k-1) و c(n-1, k) تقودنا إلى تطبيـــق الخوارزميـــة الفعالة لحل هذه المسألة.

ولعمل ذلك نحتاج إلى تسجيل نواتج معاملات "ذو الحدين" في جدول به (n+1) صف و (k+1) عمود. وللحصول على c(n,k) . يجب أن نملأ هذا الجدول صفاً بعد صف بدءً بالصف رقم صفر وحتى الصف رقم n . وكل صف يجب أن نملأه من الشمال إلى اليمين ونبدأ بالقيمة 1 لأن c(n,0) =1 . وأيضاً تنتهي الصفوف من الصف رقم صفر وحتى الصف رقم k بالقيمة 1 ، وأيضاً قيم القطر هي 1 لأن c(i,j)=1 . أما بقية المواقع في الجدول فيمكننا استخدام الخاصية أعلاه لملئها بالقيم المطلوبة وحتى الحصول على قيمة c(n,k) . الخوارزمية أدناه توضح هذه العملية:

الخوارزمية (2)

```
Biominal (n,k)
Begin
    For i=o   to  n    do
        For j=0   to  min(i,k)  do
            if (j=0,   or (j=k)   then
                        c[i,j] =1
              else
                        c[i,j] = c[i-1, j-1]
            end if
        end for j
     end for i
      return  c[n,k]
end
```

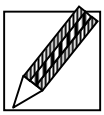
ويمكننا أن نلاحظ أن المعدل الزمني لهذه الخوارزمية يعتمد على عملية الجمع الموجودة بالصيغة القياسية للحصول على c[i,j] . ولذا إذا فرضنا أن A(n,k) هو عدد عمليات الجمع المطلوبة لحساب المعامل c[n,k] ، نجد أن أي حساب للمعامل بالصيغة يحتاج لعملية جمع واحدة ولذا نستطيع أن نصل للصيغة التالية لحساب A(n,k):

$$A(n,k) = \sum_{i=1}^{k}\sum_{j=1}^{i-1} 1 + \sum_{i+k+1}^{n}\sum_{j=1}^{k} 1$$

$$= \sum_{i=1}^{k}(i-1) + \sum_{i+k+1}^{n} k$$

$$= \frac{(k-1k}{2} + k(n-k) = O(nk)$$

تدريب (3)



1) قم بحساب الترتيب النسبي لنمو الدوال التالية:
(i) c(n,1) (ii) c(n,2) (iii) c(n,n/2)
للأعداد الزوجية لـــ n .

2) قارن الخوارزميات التالية من ناحية الكفاءة بالنسبة لحساب معامل ذو الحدين c(n,k) :

(i) باستخدام الصيغة:

c(n,k) = n!/k! (n-k)!

(ii) باستخدام الصيغة:

$$c(n,k) = \frac{n(n-1)(n-2)...(n-k+1)}{k!}$$

(iii) بتطبيق الصيغة التالية بواسطة الاستدعاء الذاتي:

c(n,k)=c(n-1,k-1+c(n-1,k), n>k>0
c(n,0)=c(n,n)=1

(iv) بتطبيق الصيغة الأخيرة بواسطة خوارزمية برمجة فعالة

نشاط

هنالك مجموعتان B, A تلعبان لعبات متتالية حتى يكسب أحدهما n لعبة. افترض أن احتمال أن تكسب المجموعة A هو p في أي لعبة (أي الاحتمال متساوي في جميع اللعبات). واحتمال خسارة المجموعة A هو q=1-p . ضع P(i,j) هو احتمال أن تكسب المجوعة A اللعبات المتتالية إذا كانت تحتاج A إلى i لعبة إضافية لتكسب و b تحتاج إلى j لعبة إضافية لتكسب اللعبات المتتالية:

(i) اكتب علاقة الاستدعاء الذاتي للدالة P(i,j) .

(ii) أحسب احتمال أن تكسب المجموعة A سبعة لعبات متتالية إذا كان احتمال المجموعة لكسب أي لعبة هو p=0.4.

(iii) اكتب خوارزمية برمجة فعالة لهذه المسألة مستخدماً علاقة الاستدعاء ثم أحسب المعدل الزمني لها.

4. خوارزمية فرق تسد

(Divide and Conquer Algorithm)

1.4 الطريقة

هذه الطريقة هي من أفضل طرق تصميم الخوارزميات والتي تعطي في كثير من المسائل أفضل معدلات النمو، وتنقسم خوارزمية فرق تسد إلى قسمين:

فرق: وهو تقسيم المسألة إلى مسائل فرعية وحلها عن طريق الاستدعاء الذاتي.

تسد: وهو عملية الحصول على الحل النهائي للمسألة من خلال تجميع حلول المسائل الفرعية.

وتوجد كثير من المسائل التي تم حلها بواسطة خوارزميات فرق تسد. ومن تلك المسائل مسألة الترتيب السريع ومسألة البحث الثنائي ومسألة التجوال في الشجرة الثنائية

وغيرها من المسائل الشبيهة، وكمثال عملي لهذه الطريقة خذ مسألة جمع n عدد مختلف مختلف $a_n,\ldots,a_2,a_1,a_0$. إذا أخذنا $n>1$ ، فيمكننا أن نقسم القائمة إلى قسمين لحساب مجموع القسم الأول والذي يحتوي على $n/2$ عدد ومجموع القسم الثاني والذي يحتوي على $n/2$ عدد أيضاً. وعندما يتم حساب المجموع بواسطة الاستدعاء الذاتي لكل قسم يتم حساب المجموع الكلي

$$a_0+\ldots a_{n-1} =[a_0+\ldots+a_{(n/2)-1}]+ [a_{n/2}+\ldots a_{n-1}]$$

وبالطبع يمكننا تنفيذ نفس الطريقة على كل قسم على حده، وهكذا حتى الحصول على المجموع الكلي.

2.4 تحليل خوارزمية فرق تسد تحليلاً زمنياً

في معظم المسائل التي نقوم بحلها بواسطة خوارزميات فرق تسد نحتاج إلى تقسيم المسألة الرئيسية والتي حجمها n إلى عدد من الأقسام، مثلاً b وقسم يكون حجمه هو n/b . ومن هذه الأقسام نحتاج إلى استخدام a قسم في كل مرحلة، حيث $a \geq 1$ و $b>1$، ولنتفرض أن حجم المسألة الكلي n يعطي بواسطة $n=b^k$ ، حيث k هو عدد موجب $k>0$ ، لذا سوف نصل إلى علاقة أستدعاء ذاتي لمعدل النمو $T(n)$ كالآتي:

$$T(n) = a\ T(n/b) + f(n)$$

حيث $f(n)$ هو معدل النمو المقابل لعملية تقسيم المسألة ذات الحجم n إلى b قسم ذات حجم n/b، والنظرية التالية هي نظرية هامة في حل هذه العلاقة بالنسبة لجميع أنواع خوارزميات فرق تسد والتي تعتمد على قسمين a,b ومعدل نمو $f(n)$.

نظرية (1)

إذا كانت الدالة $f(n)$ لها معدل $O(n^k)$ عند $k \geq 0$ في العلاقة:

$$T(n)=a\ T(n/b)+f(n)\ ,\qquad a \geq 1,\qquad b>1$$

فإن:

$$T(n) = \begin{cases} O(n^k) \ , \ \text{if } a<b^k \\ O(n^k \log n) \ , \ \text{if } a= b^k \\ O(n^{\log_b a}) \ , \ \text{if } a> b^k \end{cases}$$

وكمثال على صحة هذه النظرية خذ مسألة الحصول على المجموع السابقة والتي لها حجم n حيث علاقة الاستدعاء الذاتي تصبح:

$$T(n)=2\ T(n/2)+O(1)$$

عند a=2 و b=2 و k=0 ولذا فإن:

$$T(n) = O(n^{\log_b a}) = O(n^{\log_b 2}) = O(n)$$

وبواسطة النظرية نستطيع أن نتوصل إلى المعدل التالي:

$$T(n) = O(n^k \log n)$$
$$= O(n \log n)$$

3.4 مسألة ضرب المصفوفات بطريقة ستريسن

كما نعلم فإن أبسط طريقة لضرب المصفوفات تتم بواسطة استخدام ثلاث دارات، فمثلاً إذا كان لدينا مصفوفتين B,A بحجم nxn (أي برتبة n) فإن ناتج عملية ضرب المصفوفتان هو مصفوفة ثالثة c بحجم nxn (أي رتبة n) ، كما هو موضح في الخوارمية التالية:

خوارزمية (3)

```
Matric Mutl (A,B)
Begin
    c[i,j]=0
     for i=j   to   n  do
          for j=1   to   n  do
               for k=1   to    n   do
                    c[i,j]= c[i,j]+A(i,k)*B(k,i)
               end for k
          end for j
      end for i
       return  c
end
```

فملاحظتنا على هذه الخوارزمية أنها تعطينا معدلاً زمنياً قدره $O(n^3)$، ولذا نحتاج إلى خوارزمية فرق تسد لتقليل هذا المعدل. وضع **العالم سترسين** خوارزمية فرق تسد لتحسين هذا المعدل وتعتمد على تقسيم المصفوفات قبل عملية الضرب كالآتي:
يتم تقسيم المصفوفات الثلاثة إلى أربعة أقسام (رباعيات) على النحو:

$$A*B = C$$

$$\begin{bmatrix} [A_{11}] & [A_{12}] \\ [A_{21}] & [A_{22}] \end{bmatrix} * \begin{bmatrix} [B_{11}] & [B_{12}] \\ [B_{21}] & [B_{22}] \end{bmatrix} = \begin{bmatrix} [c_{11}] & [c_{11}] \\ [c_{21}] & [c_{22}] \end{bmatrix}$$

وحيث يمكننا الحصول على 8 ضربيات جزئية كالآت

$$\begin{aligned} c_{11} &= A_{11} * B_{11} + A_{12} * B_{21} \\ c_{12} &= A_{11} * B_{12} + A_{12} * B_{22} \\ c_{21} &= A_{21} * B_{11} + A_{22} * B_{21} \\ c_{22} &= A_{21} * B_{12} + A_{22} * B_{22} \end{aligned}$$

ونلاحظ أن حجم المسألة في كل ضربية جزئية تم تقليصه بالنصف لأن رتبة أي مصفوفة من المصفوفات الرباعيات هو n/2 . لذا وحسب النظرية أعلاه نجد أن b=2، ثم أننا احتجنا لتكرار ذلك 8 مرات أي لعمل 8 ضربيات مختلفة وبالتالي تكون a=8، كما أننا نحتاج عند تقسيم المصفوفة الواحدة إلى معدل قدره $O(n^2)$. إذا المعدل المطلوب هو :

$$T(n) = 8\ T(n/2) + O(n^2)$$

وبالتالي يصبح:

$$T(n) = O(n^{\log_2 8}) = O(n^3)$$

ولذا فإننا لتقليص هذا المعدل يتطلب منا تقليص عدد الضربيات أي تقليص قيمة a، ويمكننا ذلك بواسطة العمليات التحويلية أو بواسطة علاقات رياضية معينة على المصفوفات الفرعية.

قام العالم ستريسن بابتداع طريقة فعالة لتقليص هذا العدد إلى 7 ضربيات فقط وبواسطة علاقات رياضية محددة وموضحة كالآتي:

$M_1 = (A_{12}+A_{22})*(B_{21}+B_{22})$
$M_2 = (A_{11}+A_{22})* (B_{11}+B_{22})$
$M_3 = (A_{11}-A_{21}) *(B_{11}+B_{12})$
$M_4 = (A_{11}+A_{12})*B_{22}$
$M_5 = A_{11}*(B_{12}-B_{22})$
$M_6 = A_{22}*(B_{21}-B_{11})$
$M_7 = (A_{21}+A_{22})*B_{11}$

حيث يمكننا الحصول على رباعيات المصفوفة C على النحو الآتي:

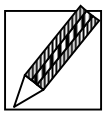
$C_{11} = M_1 + M_2 - M_4 + M_6$
$C_{12} = M_4+M_5$
$C_{21} = M_6+M_7$
$C_{22} = M_2-M_3+M_5-M_7$

ولذا فإن المعدل السابق سوف يصبح:

$$T(n) = 7\,T(n/2) + O(n^2)$$
$$= O(n^{\log_2 7})$$
$$= O(n^{2.8}) < O(n^3)$$

ونلاحظ أن المعدل هو أفضل من المعدل السابق

تدريب (4)



1) اكتب خوارزمية فرق تسد لحل مسألة إيجاد موقع العنصر الأكبر في مصفوفة من n عدد. ثم ضع علامة الاستدعاء الذاتي التي تعبر عن المعدل الزمني لعملية المقارنة في مسألة، ثم قم بحساب المعدل الزمني للخوارزمية.
2) قم بتنفيذ خوارزمية ستريسن لضرب المصفوفتين أدناه:

$$\begin{bmatrix} 1 & 0 & 2 & 1 \\ 4 & 1 & 1 & 0 \\ 0 & 1 & 3 & 0 \\ 5 & 0 & 2 & 1 \end{bmatrix} * \begin{bmatrix} 0 & 1 & 01 & \\ 2 & 1 & 0 & 4 \\ 2 & 0 & 1 & 1 \\ 1 & 3 & 5 & 0 \end{bmatrix}$$

3) قم بتنفيذ خوارزمية ستريسن لضرب المصفوفات على الحاسوب ومن ثم قم بالتأكد من ناتج عملية الضرب بتطبيق برنامج الحاسوب على المسألة رقم (2) في التدريب السابق.

الخلاصة

عزيزي الدارس،

أتوقع أنك الآن قد تحولت من مرحلة التنفيذ إلى مرحلة التصميم، سيتضح لك ذلك عند مراجعتك لأهداف الوحدة وتأكيدك على أنك قد حققتها بشكل ممتاز.

قلنا إن معظم الخوارزميات التي تعرضنا لها بسيطة وواضحة في تنفيذها. أما في هذه الوحدة فقد تعرضنا إلى أنواع من الخوارزميات تستخدم في حل المسائل. وهي الخوارزمية الجشعة والخوارزمية الفعالة وخوارزمية فرق تسد.

فلنقم سوياً بعرض الخوارزميات الثلاث:

أولاً: الخوارزمية الجشعة وهي تستخدم في حل المسائل التي تبحث عن الحل المثالي، وتقوم ببناء الحل في عدة مراحل متتابعة أولها الحل المعقول ثم الحل الأمثل الموضعي ثم الحل الذي لايمكن استرجاعه.

وقد قمنا بعرض تطبيقين من تطبيقات الخوارزمية الجشعة وهما مسألة الجدولة البسيطة ومسألة تقليص الملفات بواسطة رموز هفمان.

ثانياً: الخوارزمية الفعالة وهي خوارزمية بديلة لخوارزمية الاستدعاء الذاتي، وفيها يتم تسجيل جدول المسائل الفرعية للمسألة الرئيسية واستخدامها بالتوالي وحتى الوصول إلى الحل النهائي للمسألة، وقد وضحنا كيف يتم تطبيق هذه الخوارزمية لتعديل حل مسائل أرقام فايبونسي وكذلك استخدمنا الخوارزمية لحل مسألة إيجاد معاملات ذات الحدين.

ثالثاً: خوارزمية فرق تسد وهي أفضل طرق تصميم الخوارزميات والتي تعطي في كثير من المسائل أفضل معدلات النمو، وتستخدم في حل كثير من المسائل منها مسألة ضرب المصفوفات بطريقة ستريسن.

أرجو أن تكون قد أفدت من هذه الوحدة وقد اكتسبت معارف تسهل لك التعامل مع الخوارزميات وتصميمها.

لمحة مسبقة عن الوحدة التالية

سوف نتطرق في الوحدة السابعة والأخيرة إلى دراسة نظرية الحد الأدنى للخوارزميات والتعرف على شجرة القرارات والتي تستخدم في قياس أداء الخوارزميات وخاصة خوارزميات الترتيب والبحث فهيا معا للوحدة التالية لنختم مقررنا ونحقق أهدافه.

مسرد المصطلحات

- **الخوارزمية الجشعة Greedy Algorithm**

هي خوارزمية لحل المسائل بحل مكون من عدة مراحل متتابعة، كل مرحلة تقوم بتقريب الحل الأمثل النهائي.

- **البرمجة الفعالة Dynamic programming**

هي خوارزمية يتم تسجيل حلول المسائل الفرعية للمسألة الرئيسية، والتي يسبق حلها بالاستدعاء الذاتي، واستخدامها بالتوالي حتى الوصول إلى الحل النهائي للمسألة.

- **خوارزمية فرق تسد Divide and Conquer Algorithm**

تنقسم إلى قسمين لحل المسألة:

فرق: وهو تقسيم المسألة إلى مسائل فرعية وحلها عن طريق الاستدعاء الذاتي.

تسد: وهو عملية الحصول على الحل النهائي للمسألة من خلال تجميع حلول المسائل الفرعية.

- **حل أمثل موضعي Local Optimum Solution**

هو الحل الأفضل بين جميع الحلول المعتدلة في المرحلة المعينة.

- **الحل الأمثل Optimum Solution**

الحل النهائي القاطع للمسألة.

- **جدولة Scheduling**

ترتيب الأعمال وفق جدول مرتب ترتيب معين

- **مسألة Problem**

هي ما يجب حله بواسطة الحاسوب

- **رموز هفمان Hufman Codes**

هي رموز مستخدمة في توضيح الخوارزمية

- **تمثيل شجري Tree Presentaion**

هو طريقة لتمثيل الخوارزمية

- **طريقة ستريسن Stressen's Mehthod**

هي الطريقة المستخدمة في خوارزمية فرق تسد وتعتمد على تقسيم المصفوفات قبل الضرب.

المراجع

المراجع العربية

1) مجموعة مؤلفين، تركيب البيانات وتصميم الخوارزميات، منشورات جامعة القدس المفتوحة، 1998.

2) السماني عبد المطلب، هياكل البيانات، منشورات جامعة السودان المفتوحة، 2005.

المراجع الأجنبية

1) Weiss, M.A., Data Structures and Alogrithm Analysis, Benjamin pub., 1992.
2) McConnell, J.J., Analysis of Alogrithms: An Active Approach, Jones pub., 2001.
3) Lavitin, A., Introduction to the Design and Analysis of Alogrithms, Addisions ues by, 2003.
4) Ulman, J.D (etal), The Design and Analysis of Compurte Alogrithms, Addison Wesley, 1974.
5) Basese, S., Compurte Alogrithms : Introduction to Analysis and Design, Addison Wesley, 2000.
6) Sedgewick, R., An Introduction to the Analysis of Alogrithms, Addison Wesley 1996.


الوحدة السابعة
محدودية قوة الخوارزميات
Limitation of Algorithm Power

محتويات الوحدة

المقدمة

تمهيد

في هذا الكتاب تعرفنا على عدد كبير من الخوارزميات المختلفة والتي يتم استخدامها في حل عدد من المسائل المختلفة، ولكن دعنا نعلم بأن قوة هذه الخوارزميات محدودة ومحدوديتها هي موضوعنا الحالي، من المعلوم أن هنالك مسائل لا يمكن حلها بأي خوارزمية، وهنالك مسائل يمكن حلها بواسطة خوارزميات غير محددة الزمن، كما هنالك مسائل يمكن إيجاد حل لها بواسطة خوارزميات محددة الزمن، وبالرغم من ذلك يوجد حد أدنى يعبر عن كفاءة تلك الخوارزميات. وفي هذه الوحدة سوف نتعرض لمفاهيم هامة في تحليل وتصميم الخوارزميات.

تتكون هذه الوحدة من ثلاثة أقسام رئيسية. في القسم الأول سوف نتعرض لنظرية الحد الأدنى للخوارزميات كما سنتعرف على كيفية تحديد الحد الأدنى للخوارزميات المختلفة. أما القسم الثاني فيشتمل على طريقة شجرة القرارات والتي بواسطتها يمكننا قياس أداء خوارزميات الترتيب والبحث. وفي القسم الثالث سوف نقوم بدراسة ثلاثة أصناف للمسائل حسب تعقيدها وهي المسائل محددة الحل والمسائل غير محددة الحل بالتمام.

هذا وقد زودناك في هذه الوحدة بمجموعة من الأنشطة والتدريبات وأسئلة التقويم الذاتي نتمنى أن تستفيد منها كما نتمنى أن تشركنا في نقد هذه الوحدة وتقويمها.

أهداف الوحدة

عزيزي الدارس، بعد دراستك لهذه الوحدة نرجوا أن تكون قادراً على أن:

- ✓ **تحدد** الحد الأدنى للخوارزميات المختلفة.
- ✓ **تقيس** أداء الخوارزميات بواسطة شجرة القرارات.
- ✓ **تصنف** المسائل المختلفة إلى مسائل محددة الحل أو غير محددة الحل بصورة تامة.

1. الحد الأدني للخوارزمية

Lower Bound of Algorithm

يمكننا النظر في قياس كفاءة أي خوارزمية بطريقتين؛ الطريقة الأولى وبصورة واضحة هي تصنيفه حسب الترتيب الزمني الذي يتبع له، فمثلاً ترتيب الإدخال يتبع للصنف التربيعي أي ($O(n^2)$) وهو خوارزمية سريعة مقارنة بخوارزمية أبراج هانوي والتي تتبع للتصنيف الأسّي أي ($O(2^n)$) .

لكننا لايمكن اعتماد هذه المقارنة لأن الخوارزميتين يتم استخدامهما في حل مسألتين مختلفتين. ولذا فإن الطريقة الثانية والأنسب هي أن نبحث في قياس كفاءة أي خوارزمية حسب الخوارزميات الأخرى والمستخدمة في حل شقي المسألة التي يقوم بحلها.

ولهذا فإن خوارزمية الترتيب بالإدخال تكون خوازرمية بطيئة مع خوارزمية ترتيب أخري تتبع لصنف $O(n \log n)$ ، كما أن خوارزمية أبراج هانوي هي خوارزمية سريعة لفرض المسألة التي تقوم بحلها.

عندما نود تحديد كفاءة خوارزمية ما بناءً على كفاءة خوارزميات أخرى لنفس المسألة، فمن الأفضل أن يكون لدينا علم عن الحد الأدني للكفاءة المطلوبة لأي خوارزمية لحل هذه المسألة، وهذا الحد الأدنى يعطينا فكرة عن مدى التحسين المطلوب لأي خوارزمية لتكون أفضل من حل المسألة، وسوف نقوم في الفقرات القادمة بدراسة بعض الطرق المختلفة في تحديد هذا الحد الأدنى مع توضيح بعض الأمثلة.

1.1 الحد الأدني التافه Trivial Lower Bound

أسهل طريقة للحصول على الحد الأدنى تعتمد على حساب عدد العناصر ضمن مدخلات المسألة والتي يجب معالجتها، وعدد العناصر ضمن مخرجات المسألة والتي يجب إنتاجها، وبما أن أي خوارزمية على الأقل يجب أن تعالج جميع مدخلاتها وتكتب جميع مخرجاتها، فإن هذا العدد المطلوب يعطي ما يعرف بالحد الأدنى التافة.

مثال (1)

أي خوارزمية تقوم بتوليد عدد تباديل n عنصر فريد يجب أن تكون $\Omega(n!)$ وذلك لأن حجم المخرجات هو $n!$. وهذا الحد الأدنى هو حد نهائي لأن أي خوارزمية تقوم بتوليد التباديل المطلوبة ستفرق في أي تبديلة زمن ثابت عدا التبديلة الأولى.

مثال (2)

خذ مسألة حساب قيمة كثيرة الحدود من الدرجة n :

$$p(n) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_0$$

من الملاحظ أن المعاملات $a_0, \ldots, a_{n-1}, a_n$، يجب أن تتم معالجتها بواسطة أي خوارزمية، ولذا نصل إلى أن أي خوارزمية يجب أن تكون ضمن المعدل $\Omega(n)$ أيضاً **لاهذا** الحد هو حد أدنى نهائي.

مثال (3)

الحد الأدنى التافة لمسألة إيجاد حاصل ضرب مصفوفتين من الدرجة n هو $\Omega(n^2)$ لأن أي خوارزمية يجب أن يقوم بمعالجة $2n^2$ عنصر من المدخلات وتوليد n^2 عنصر من المخرجات. لكن هذا الحد هو حد أدنى غير نهائي.

2.1 الحد الأدني النظري

Information-Theoretic Lower Bound

هذه الطريقة تعتمد أساساً على كمية المعلومات المطلوب انتاجها من الخوارزمية ولـذا تم إطلاق اسم الطريقة النظرية عليها لأن لها صلة كبيرة بنظرية المعلومات، وهذه الطريقة هي طريقة عملية ومناسبة لكثير من المسائل التي تتطلب مقارنات مثل مسائل الترتيب والبحث، كما أن لها آلية تعرف بشجرة القرارات سوف نقوم بالتعرض لها في القسم القادم.

مثال (4)

خذ مسألة اكتشاف الرقم الصحيح الموجب بين 1 و n والذي يتم اختياره بواسطة أحد الأشخاص وذلك بالإجابة على بعض الأسئلة التي تقدم له بنعم أو لا.

من الواضح أن كمية المعلومات غير المؤكدة التي يجب أن تقوم أي خوارزمية لحل هذه المسألة بمعالجتها هي Log_2n وهو بالضبط عدد الخانات المطلوبة لتعيين عدد ضمن n محاولة. وهنا نحن نركز على كل سؤال (أو بالضبط على الإجابة المطلوبة لكل سؤال) والذي يوضح خانة واحدة من مخرجات الخوارزمية أي الرقم المطلوب، ولهذا فإن أي خوارزمية تحتاج على الأقل Log_2n خطوة لأنتاج المخرجات المطلوبة في الحالة السيئة.

3.1 الحد الأدني بواسطة تقليص المسائل

Lower Bound by Problem Reduction

هذه الطريقة تعتمد على مفهوم تقليص المسائل وتتلخص الطريقة كالآتي: (لإثبات أن المسألة P على الأقل لها صعوبة مثل مسألة أخرى Q والتي لها حد أدنى معروف، نحتاج إلى تقليص المسألة Q إلى المسألة P، أي بمعنى آخر يجب أن نثبت أن أي حالة عشوائية للمسألة Q يمكن تحويلها إلى حالة أخرى للمسألة P، ولذا فإن أي خوارزمية تقوم بحل المسألة Q ، ولهذا فإن الحد الأدنى للمسألة Q هو نفسه الحد الأدنى للمسألةP).

الجدول التالي يوضح بعض المسائل التي يتم استخدامها في هذه الطريقة.

جدول (1) مسائل مختلفة تستخدم في تقليص المسائل

المسألة	الحد الأدنى	نوع الحد الأدنى
1. الترتيب	$\Omega(n \log n)$	نهائي
2. البحث في قائمة مرتبة	$\Omega(\log n)$	نهائي
3. ضرب عددين صحيحين ذوي n خانة	$\Omega(n)$	غير نهائي
4. ضرب المصفوفات المربعة ذوي الدرجة n	$\Omega(n^2)$	غير نهائي
5. مسألة فردية العناصر	$\Omega(n \log n)$	نهائي

مثال (5)

خذ مسألة شجرة إقليدس الصغرى Euclideas Minimum Spanning Tree

والتي تصاغ على النحو : معطى n نقطة في المستوى السيني قم ببناء شجرة ذات أصغر طول كلي والتي رؤوسها هي نفس الـ n نقطة. سوف نستخدم الحد الأدنى المعلوم الخاص بمسألة فردية العناصر. نستطيع تحويل أي مجموعة $x_n6,\ldots x_3,\ x_2,x_1$ من n عدد صحيح إلى مجموعة من n نقطة بواسطة إضافة صفر في مكان الإحداثي الصادي (y) : $(x_1,0), (x_2,0), (x_3,0), \ldots, (x_n,0)$. ضع T هي الشجرة المطلوب بنائها من تلك النقاط وبما أن T لابد أن تحتوي على أصغر حافة، فإن أي خوارزمية لحل المسألة لابد أن تجيب عن سؤال عن فردية النقاط ، وهذا التقليص يقتضي أن أدنى حد للمسألة هو $O(n \log n)$.

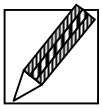
مثال (6)

يمكننا أيضاً استخدام أسلوب تقليص المسائل في مقارنة بتعقيد المسائل النسبي. فمثلاً الصيغ أدناه:

$$x.y = \frac{(x+y)^2 - (x-y)^2}{4} \qquad and \qquad x^2 = x.x$$

يمكن الاستفادة منهما في إثبات أن مسألة إيجاد حاصل ضرب عددين صحيحين ذوي n خانة ومسالة إيجاد مربع عدد صحيح ذي n خانة ينتميان لنفس صنف التعقيد أي O(n) .

تدريب (1)



1) قم بإيجاد الحد الأدنى التافه للمسائل التالية وحدد ما إذا كان هو حداً نهائياً من عدمه:
   أ- مسألة إيجاد العنصر الأكبر من قائمة من الأعداد.
   ب- مسألة توليد جميع المجموعات الجزئية من مجموعة بها n عنصر.
   ج- مسألة تحديد ما إذا كان n عدد صحيح جميعهما فردية.
2) أثبت أن أي خوارزمية تعتمد على المقارنة لإيجاد العدد الأكبر ضمن n عدد يجب أن تأخذ (n-1) مقارنة في حالتها السيئة.
3) مستخدماً نفس الصيغ في المثال 1-6 والخاص بتكافؤ تعقيد ضرب عددين صحيحين ومربع عدد صحيح، وذلك لإثبات تكافؤ تعقيد مسالتي ضرب مصفوفتين ومربع المصفوفة.

2. قياس الأداء بواسطة شجرة القرارات

معظم الخوارزميات وخاصة خوارزميات الترتيب والبحث تعمل بواسطة مقارنة العناصر مع بعضها البعض. يمكننا قياس أداء تلك الخوارزميات بواسطة أداة يطلق عليها شجرة القرارات. تحتوي شجرة القرارات على الجزر والأوراق بحيث أن كل ورقة في كل مرحلة داخل الشجرة تمثل نتيجة محتملة من الخوارزمية التي تعمل على n من المدخلات، وهنالك نقطة هامة وهي أن عدد الأوراق يجب أن يكون أكبر من عدد النتائج المحتملة من الخورازمية، كما أن متابعة أي خوارزمية تعمل على n من المدخلات سوف تتم من الجزر إلى الورقة التي بها القرار، وعدد المقارنات المطلوب في هذه الحالة سوف يكون هو عدد الحواف الموجودة في هذا الممر. ولذا فإن عدد المقارنات الخاص بالحالة السيئة للخوارزمية هو طول شجرة القرارات الناتجة من الخوارزمية.

والفكرة الأساسية من هذا النموذج هي أن أي شجرة قرارات ثنائية بها L ورقة تعبر عن النتائج المحتملة من الخوارزمية يجب أن تكون طويلة بشكل مناسب لتحتوي على كل تلك النتائج، وبالتالي يمكننا أن نثبت أن طول الشجرة h يعبر عنه بواسطة المتباينة التالية:

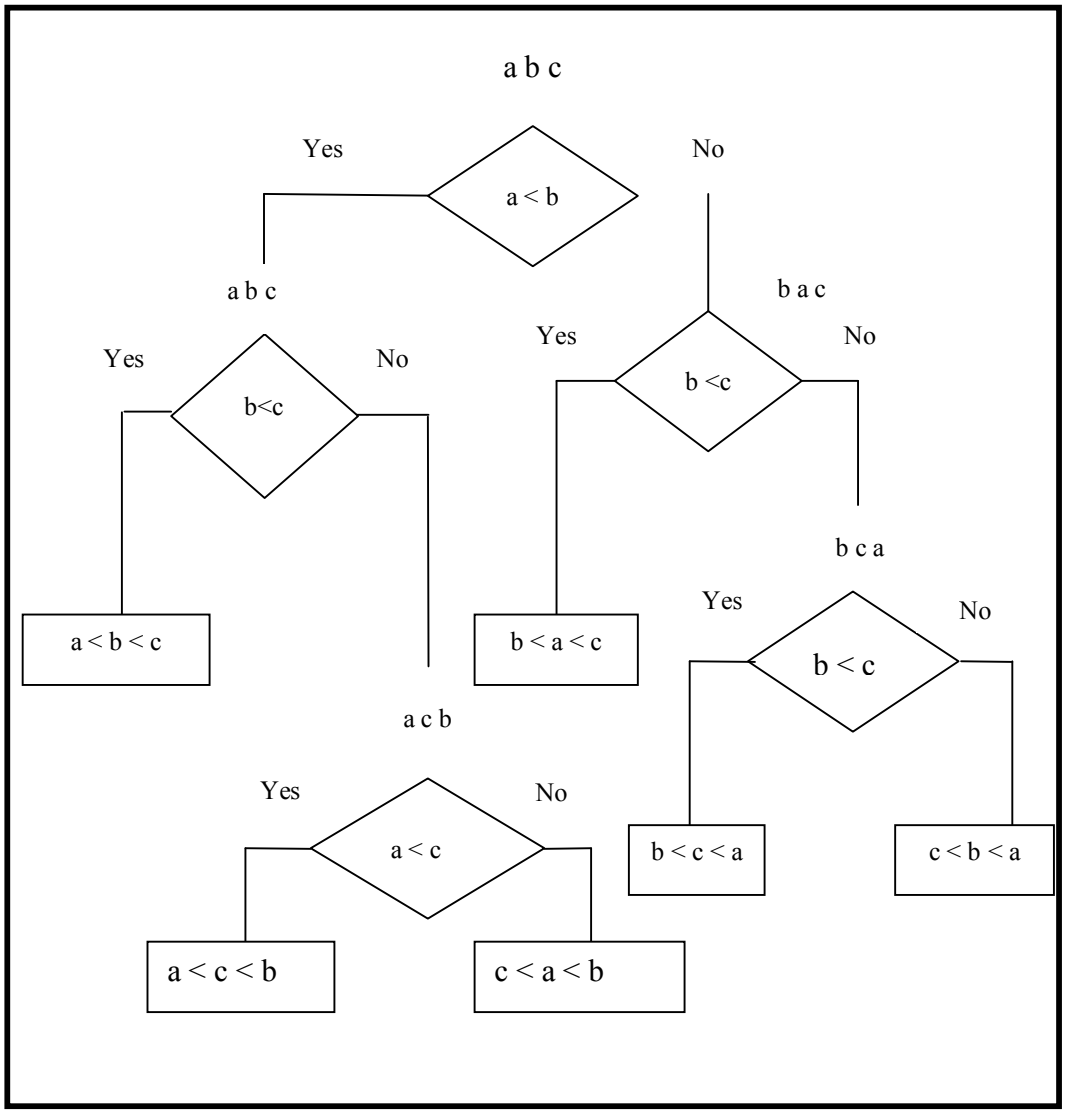
$$h \geq \log_2 L$$

وهذه المتباينة توضح الحد الأدنى من طول شجرة القرارات الثنائية وبالتالي عدد المقارنات للحالة السيئة الناتجة من خوارزمية تقوم أو تعتمد على مقارنة العناصر مع بعضها البعض، وسوف توضح هذه الطريقة على خوارزميات الترتيب والبحث في الفقرات القادمة من هذا القسم.

1.2 شجرة القرارات لخوارزميات الترتيب

كما هو معلوم فأن معظم خوارزميات الترتيب هي خوارزميات تعتمد على عملية المقارنة، أي إنها تقوم بمقارنة عناصر القائمة المراد ترتيبها مع بعضها البعض، لذا يمكننا أن نستخدم طريقة شجرة القرارات وذلك لإيجاد الحد الأدنى لكفاءة خوارزمية تعتمد على عملية المقارنة.

وناتج عملية الترتيب هو إيجاد التبديلة التي تعبر عن العناصر بصورة مرتبة. فمثلاً الناتج $a < c < b$ من عملية ترتيب ثلاثة عناصر a,b,c هو التبديلة 132، وشجرة القرارات أدناه توضح مسار عملية الترتيب بالإدخال كما تم التعرف عليها في الوحدة الرابعة من هذا الكتاب:


a b c
Yes
a < b
No
a b c
b a c
Yes
b<c
No
Yes
b <c
No
a < b < c
b < a < c
b c a
Yes
b < c
No
a c b
Yes
a < c
No
b < c < a
c < b < a
a < c < b
c < a < b

الشكل (1)

إنه وبصورة عامة فإن عدد النواتج المحتملة من ترتيب n عنصر هو !n، ولهذا فإن المتباينة السابقة تقتضي أن خوارزمية الترتيب تحتاج إلى عدد مقارنات في حالتها السيئة لا يقل عن $\log_2 n!$ أي بمعنى:

$$T_{wrs}(n) \geq \log_2 n!$$

وباستخدام صيغة إسترلينج للمضروب !n نحصل على:

$$\log n! \approx \log_2 \sqrt{2\pi n}(n/e)^n$$

$$= n\log_2 n - n\log_2 e + \frac{\log_2 n}{2} + \frac{\log_2 2\pi}{2}$$

$$\approx n\log_2 n$$

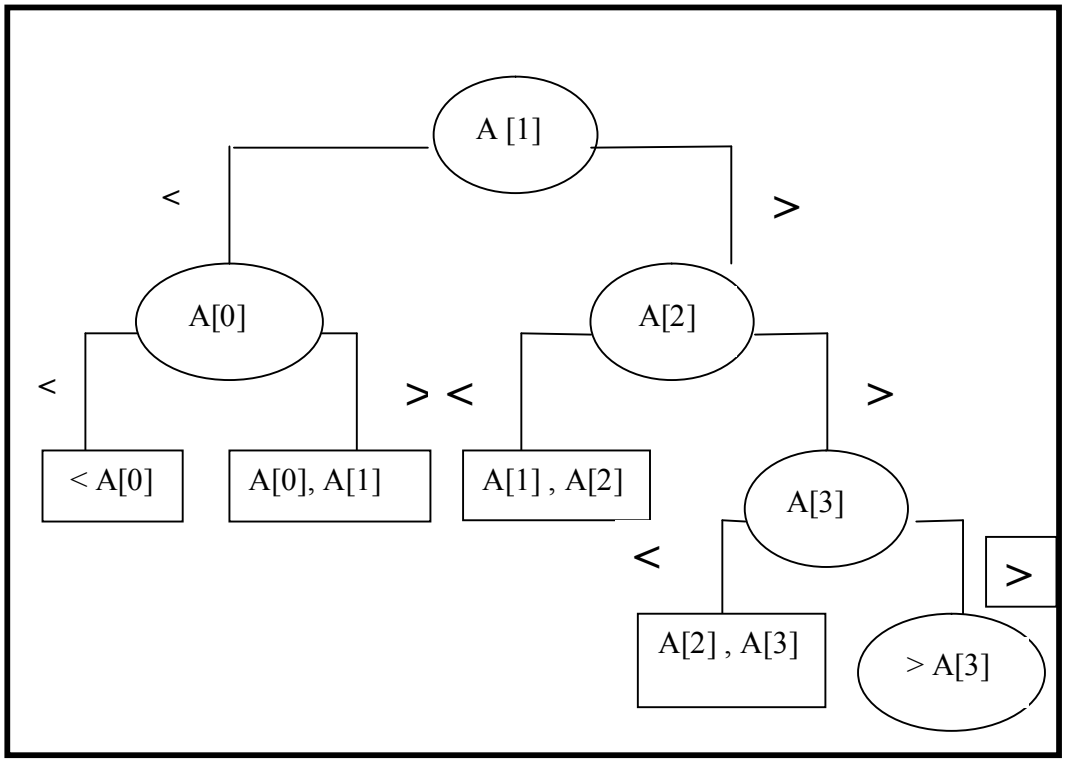
وهذا يعني أن خوارزمية الترتيب تحتاج على الأقل $n \log_n$ مقارنة كحد أدنى لتقوم بترتيب قائمة من n عنصر مختلف أو فريد، ولنا أن نعلم أنَّ هذا الحد هو حد أدنى نهائي ولا يمكن أن نحصل على حد أدنى منه.

2.2 شجرة القرارت لخوارزمية البحث

سوف نستخدم شجرة القرارات لأيجاد الحد الأدنى لعدد المقارنات في خوارزمية البحث من قائمة مرتبة من n عنصر فريد. من المعلوم أن أبسط خوارزمية تقوم بذلك هي خوارزمية البحث الثنائي والتي نعلم أن عدد المقارنات التي تحتاجها في حالتها السيئة يعطى بالصيغة:

$$T_{wrs}(n) = \log_2 n + 1 = \log_2(n+1)$$

فمثلاً للبحث عن عنصر ضمن قائمة من 4 عناصر ولأننا نحتاج إلى مقارنتين في كل مرحلة من مراحل خوارزمية البحث الثنائيين فإننا سوف نستخدم شجرة قرارات ثنائية كما هو موضح أدناه:



الشكل (2)

في هذه الشجرة الرؤوس الداخلية تعبر عن حالات البحث الناجحة، أما الأوراق فتعبر عن حالات البحث غير الناجحة ولذا توجد (n+1) ورقة بصورة عامة للبحث ضمن قائمة بها n عنصر فريد.

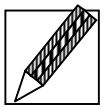
وأيضاً باستخدام المتباينة العامة يمكننا حساب الحد الأدنى كالآتي:

$$T_{wrs}(n) \geq \log_2(n+1)$$

وهذا الحد الأدنى هو نفس المعدل الذي تم الحصول عليه للحالة السيئة لخوارزمية البحث الثنائي. وللعلم فإن هذا الحد غير نهائي، لأننا يمكن وبإضافة بعض الافتراضات

القياسية لعملية البحث إثبات أن عدد المقارنات للحالة الوسطى لا يقل عن $\log_2(n-1)$ لحالات البحث الناجحة ولايقل عن $\log_2(n+1)$ لحالات البحث غير الناجحة.

تدريب (2)



1. خذ مسألة إيجاد الوسيط Median لثلاثة عناصر {a,b,c} .
   أ. حدد عدد المقارنات للحالة السيئة من الخوارزمية.
   ب.ارسم شجرة قرارات لهذه الخوارزمية ثم قم بحساب الحد الأدنى النظري.
2. أرسم شجرة القرارات واحسب الحد الأدني لعدد المقارنات في الحالة السيئة للترتيب السريع والذي يقوم بترتيب قائمة بها 3 عناصر فريدة.
3. ارسم شجرة قرارات ثنائية البحث ضمن قائمة بها 4 عناصر فريدة. ثم أحسب الحد الأدنى لعدد المقارنات للحالة السيئة في الخوارزمية.

3. التعقيد المحوسب للمسائل

Computational Complexity of Problems

يركز التعقيد المحوسب للمسائل المختلفة على تصنيف المسائل إلى مسائل قابلة للحل في زمن محدد بكثيرة حدود Polynomial time أو مسائل غير قابلة للحل في زمن محدد. سوف نتناول التعريف التالي الذي يوضح القاعدة العامة لذلك.

تعريف (1)

يقال أن أي خوارزمية تقوم بحل مسألة في زمن محدد بكثيرة حدود إذا كان المعدل الزمني للحالات السيئة يتبع للمعدل $O(p(n))$ حيث p(n) هي كثيرة حدود من حجم n من المدخلات:

$$p(n) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0$$

(مع ملاحظة أن أي مسألة قابلة للحل في معدل لوغريثمي هي أيضاً قابلة للحل في زمن محدد بكثيرة حدود).

1.3 مسألة P ومسألة NP Pand NP Problems

معظم المسائل التي قمنا بمناقشتها في هذا الكتاب يمكننا إيجاد خوارزمية لحلها في معدل زمني محدد. وتشمل القائمة مسألة القاسم المشترك الأعظم ومسائل الترتيب والبحث وإيجاد الشجرة الصغرى وإيجاد المسار الأقصر للبيانات والتعريف التالي يوضح متى يمكننا أن نطلق على أي مسألة أنها مسألة P أي مسألة قابلة للحل في زمن محدد.

تعريف (2)

الصنف P هو صنف يحتوي على المسائل القرارية decision problems (أي المسائل التي تحتاج إلى إجابة بنعم أو لا) والتي يمكن حلها في زمن محدد بكثيرة حدود Polynomial time وبواسطة خوارزمية محددة Deterministic Algorithm وسوف نطلق على هذا الصنف P اسم محدد بكثيرة حدود Polynomial .

التقيد بالمسائل القرارية في هذا التعريف له سببين:

السبب الأول: لا يمكن تجاهل مسائل هامة لا يمكن التصور بحلها في معدل محدد بكثيرة حدود من أمثال مسألة توليد ا لمجموعات الجزئية لمجموعة ما ومسألة توليد التباديل لعدد n من العناصر.

السبب الثاني: معظم المسائل التي بطبيعتها لا تعتبر مسائل قرارية يمكننا تحويلها إلى متتالية من المسائل القرارية الجزئية.

وهنالك مسائل كثيرة هامة لا يمكن إيجاد خوارزمية ذات معدل زمني محدد بكثيرة حدود لحلها.

ومن أمثلة تلك المسائل الآتي:

1. مسألة حلقة هاملت Hamiltonian Circuit

تحديد ما إذا كان توجد حلقة هاملت في بيان وهي مسار يبدأ وينتهي في نفس الرأس ويمر بجميع الرؤوس في البيانات مرة واحدة فقط.

2. مسألة البائع المتجول Trading Salesman

إيجاد أقصر طريق يمر بعدد n مدينة معلوم المسافة بينها.

3. مسألة حقيبة الظهر Knapsack Problem

تحديد قيم n عنصر جزئي لها أثقال وقيم عددية موجبة لتناسب حقيبة الظهر التي لها سعة عددية محددة.

4. مسألة التجزئة Portion Problem

معطى n عدد صحيح موجب، المطلوب تحديد ما إذا كان يمكن تجزئة الأعداد إلى جزئين لهما نفس المجموع.

5. مسألة تلوين البيان Graph Coloring

معطى بيان، المطلوب إيجاد أقل عدد من الألوان والتي يتم استخدامها لكل رؤوس البيان بحيث لا يتم استخدام لونين متشابهين في أي رأسين متجاورين.

ولإيجاد حل لهذه المسائل بواسطة تحويلها أو استخدام مسائل قرارية، قام علماء الحاسوب بوضع التعريفات التالية:

تعريف (3)

الخوارزمية غير المحددة Non-deterministic Algorithm هي إجراء ذو مرحلتين يأخذ مدخل I كحالة من مسألة قرارية ويقوم بالآتي:

1. مرحلة التخمين (حل غير محدد):

يتم توليد رمز حرفي (String) S كحل للحالة I .

2. مرحلة التحقيق (حل محدد):

يتم استخدام خوارزمية محددة تقوم بإدخال S,I كمدخلات ونقوم بإخراج نعم إذا كان S يمثل حلاً للحالة I ولا إذا كان غيرذلك.

الخوارزمية غير المحددة يطلق عليها خوارزمية غير محددة بكثيرة الحدود Non deterministic Polynomial إذا كان المعدل الزمني لمرحلة التحقيق فيها هو معدل محدد بكثيرة حدود.

والآن سوف نقوم بتعريف المسائل من الصنف NP .

تعريف (4)

الصنف NP هو صنف يحتوي على المسائل القرارية والتي يمكن أن نجد حلاً لها بواسطة خوارزمية غير محددة بكثيرة الحدود. وهذا الصنف سوف نطلق عليه اسم غير محدد بكثيرة الحدود Non deterministic Polynomial

والمعلوم أن معظم المسائل القرارية هي مسائل NP؛ أي أن المسائل P هي جزء من المسائل NP.

$$P \leq NP$$

وهذه النتيجة صحيحة لأن أي مسألة P يمكن حلها بواسطة خوارزمية محددة بكثيرة حدود في مرحلة التحقيق الخاصة بخوارزمية غير محددة بكثيرة حدود تتجاوز المرحلة الأولى أي لا تحتاج إلى توليد رمز حرفي.

وهذا الصنف يشتمل على كل المسائل المذكورة سابقاً وكثير من المسائل المشابهة لها والتي عرفها علماء الحاسوب.

2.3 مسألة NP التامة NP-Complete Problem

تعرف مسألة NP التامة بأنها مسألة تتبع للصنف NP بحيث أنها معقدة ويمكن لكل مسألة أخرى من NP أن يتم تقليصها لمسألة NP التامة والتعريفات التالية تساعدنا في فهم هذا الصنف الجديد.

تعريف (5)

المسألة القرارية D_1 يقال بأنها قابلة للتقليل بكثيرة حدود Polynomial Reducible إلى المسألة القرارية D_2 إذا وجدت دالة t تقوم بتحويل حالات D_1 إلى حالات D_2 بحيث:

1. t تقوم بتحويل حالات كل إجابة نعم في D_1 إلى حالات إجابة نعم في D_2 وكل حالات إجابة لا في D_1 إلى حالات إجابة لا في D_2 .
2. t يمكن حسابها بواسطة خوارزمية محددة بكثيرة حدود.

هذا التعريف يقتضي بأن أي مسألة D_1 وإذا كانت قابلة للتقليص بكثيرة حدود إلى مسألة D_2 والتي هي قابلة للحل في زمن محدد بكثيرة حدود فإن المسألة D_1 أيضاً قابلة للحل في زمن محدد بكثيرة حدود.

تعريف (6)

المسألة القرارية D يقال بإنها مسألة NP تامة.

إذا كان:

1. المسألة D هي مسألة NP أي أنها تتبع للصنف NP .
2. أي مسألة في الصنف NP هي مسألة قابلة للتقليص بكثيرة حدود للمسألة D .

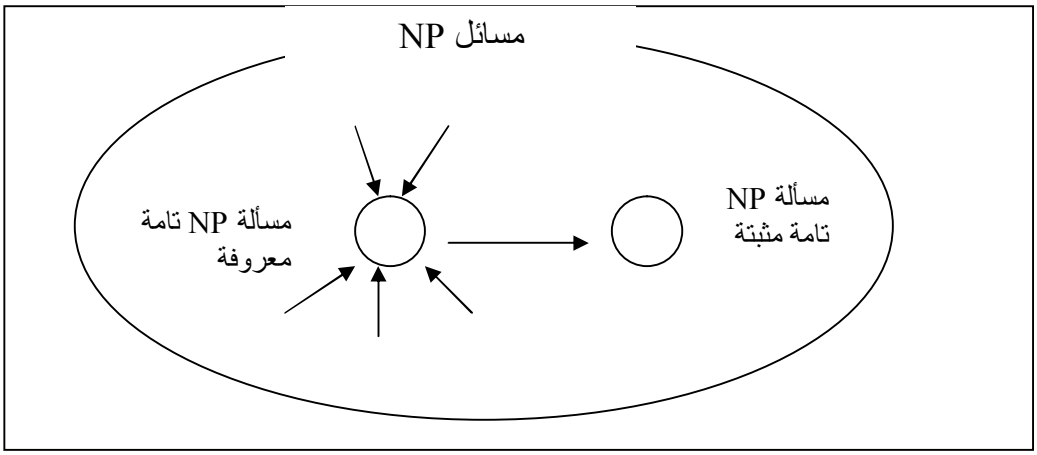
ولتحديد أن مسألة قرارية هي مسألة NP تامة لابد من القيام بالآتي:

أولاً: نحتاج لمعرفة أن المسألة تتبع لصنف NP أي يتم توليد زمن حرفي عشوائياً وتحديد ما إذا كان عند زمن محدد بكثيرة حدود يمثل حلاً للمسألة أم لا.

ثانياً: نحتاج لمعرفة أن كل مسألة في الصنف NP هي مسألة قابلة للتقليص بكثيرة حدود للمسألة الأم.

وبسبب التعدي في علمية التقليص بكثيرة الحدود يمكننا القيام في هذه المرحلة فقط بإثبات أن مسألة NP تامة معروفة يمكن تقليصها بكثيرة حدود للمسألة الأم.

ويمكننا توضيح عملية الإثبات أعلاه في الشكل التالي:



الشكل (3)

مثال (7)

أثبت أن مسألة البائع المتجول هي مسألة NP تامة علماً بأن مسألة حلقة هاملت هي مسألة NP تامة معروفة.

الحل:

أولاً: يمكن إثبات أن مسألة البائع المتجول بأنها مسألة NP بسهولة حيث أن إيجاد رمز حرفي ليمثل حلاً للمسألة بزمن محدد بكثيرة حدود هو أمر بسيط ويمكن أن يتم بالمرور على المدن عشوائياً حتى الوصول إلى المسار الصحيح.

ثانياً: لإثبات أن كل مسائل NP هي مسائل قابلة للتقليص بكثيرة حدود لمسألة البائع المتجول سوف نكتفي باختيار مسألة NP تامة والقيام بإثبات ذلك عليها. بما أن

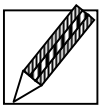
مسألة حلقة هاملت هي معلومة بأنها مسألة NP تامة فإننا سوف نحاول إثبات أنها قابلة للتقليص بكثيرة حدود لمسألة البائع المتجول.

الآن يمكننا صياغة مسألة حلقة هاملت كالآتي: هل توجد حلقة هاملت في بيان تام ذو ثقل صحيح موجب له طول لا يزيد عن عدد صحيح موجب m . يمكن تحويل حالة بيان G لمسألة حلقة هاملت إلى بيان تام بثقل G يمثل حالة لمسألة البائع المتجول بواسطة تعيين العدد واحد لكل ثقل لأي حافة في G وإضافة حافة بثقل 2 بين أي رأسين غير متجاورين في G . كحد أعلى m لطول حلقة هاملت، سوف نأخذ m=n ، حيث n هي عدد الرؤوس في G **(وفي G)** . من الواضح أن عملية التحويل السابقة هي عملية محددة بزمن بكثير حدود.

ولإكمال عملية الإثبات سوف نفترض أن G هو حالة إجابة نعم بالنسبة لمسألة حلقة هاملت. وإن G سوف يكون له حلقة هاملت وبالتالي الصورة المماثلة له في **G** سوف يكون له طول n وسوف تجعل حالة مسألة البائع المتجول المقابلة ذات إجابة نعم، وبالعكس إذا كانت حلقة هاملت لها طول لايزيد عن n في **G** ، فإن طولها سوف يكون بالضبط هو n وبالتالي الحلقة لابد أن تكون مكونة من حواف موجودة في G وهذا سوف يجعل الصورة المماثلة لحالة نعم لمسألة البائع المتجول هي حالة نعم لمسألة حلقة هاملت.

وهذا التحويل يوضح قابلية مسألة حلقة هاملت لمسألة البائع المتجول ، وبالتالي إثبات أن مسألة البائع المتجول هي أيضاً مسألة NP تامة.

تدريب (3)



خذ الخوارزمية البسيطة التالية لحل مسألة الأعداد غير الأولية (Composite Number Problem).

- قم بفحص الأعداد الصحيحة بين 2 و n/2 لتكون قوائم محتملة للعدد n .
- إذا وجد بينهم من يقسم n بصورة زوجية فقم بإرجاع نعم (أي بمعنى العدد غير أولي).
- إذا لم يوجد بينهم منم يقسم n بصورة زوجية فقم بإرجاع ..... (اي بمعنى العدد هو عدد أولي).

لماذا هذه الخوارزمية لا تضع هذه المسألة بين صنف P ؟

أثبت أن مسألة التجزئة هي مسألة NP تامة إذا علمنا أن مسألة حقيبة الظهر هي مسالة NP تامة.

نشاط

قام الملك آرثر بدعوة 150 فارساً للعشاء ولكن قبل العشاء علم الملك بأن بعض الفرسان قاموا بالمشاجرة مع بعضهم ، ولذا قام الملك بطريقة لإجلاس الفرسان حول طاولة بحيث لايجلس أي فارسين تشاجرا بالقرب من بعضهما.

أ)ما هي المسألة القياسية التي يمكن استخدامها كنموذج لمسألة الملك آرثر.

ب) قم بإثبات أن مسألة الملك آرثر يوجد لها حل إذا كان أي فارس لا يتشاجر مع لا يقل عن 75 فارس آخر.

الخلاصة

عزيزي الدارس

لنقم سوياً بتلخيص ما درسناه في هذه الوحدة فقد بدأنا في القسم الأول وعرفنا أنه يمكننا قياس كفاءة الخوارزمية بالبحث عن كفاءة أي خوارزمية حسب الخوارزميات الأخرى والمستخدمة في حل شقي المسألة التي يقوم بحلها، ومن الأفضل أن يكون لدينا علم عن الحد الأدنى للكفاءة المطلوبة لأي خوارزمية لحل المسألة المراد البحث فيها.

ثم ذكرنا الحد الأدنى التافه وعرفناه بأنه عدد العناصر ضمن مدخلات المسألة وعدد العناصر ضمن مخرجاتها والتي يجب إنتاجها.

كما عرفنا الحد الأدنى النظري والذي يعتمد أساساً على كمية المعلومات المطلوب إنتاجها من الخوارزمية.

ثم ذكرنا الطريقة الأخيرة وهي تعتمد أساساً على مفهوم تقليص المسائل لأيجاد الحد الأدنى للخوارزمية المطلوبة.

أما في قسمنا الثاني فقد ناقشنا قياس الأداء بواسطة شجرة القرارات، ووضحنا المتباينة التي تعطي الحد الأدنى من طول شجرة القرارات الثنائية وبالتالي عدد المقارنات للحالة السيئة الناتجة من خوارزمية تقوم أو تعتمد على مقارنة العناصر مع بعضها البعض وهي:

$$N \geq \log_2 L$$

كما شرحنا شجرة القرارات لخوارزميات الترتيب وخوارزمية البحث كيف تعمل لإيجاد الحد الأدنى بمقارنة العناصر.

وفي قسمنا الثالث شرحنا أن التعقيد المحوسب للمسائل يصنف المسائل إلى مسائل قابلة للحل في زمن محدد بكثيرة حدود أو مسائل غير قابلة للحل في زمن

محدد. وعرفنا أن أي مسالة قابلة للحل في زمن محدد. هي مسألة P وهي من صنف المسائل القرارية، ووضعنا أمثلة لهذه المسائل كمسألة حقيبة هاملت ومسألة البائع المتجول ومسألة حقيقة الظهر ومسألة التجزئة ومسألة تلوين البيان، أما الصنف NP فقد علمنا أنه صنف يحتوي على المسائل القرارية والتي يمكن أن نجد حلها حلاً بواسطة خوارزمية غير محددة بكثيرة الحدود وتستخدم مرحلتين الأولى التخمين ثم التحقيق.

كما وضحنا العلاقة القائلة بأن $P \leq NP$

ثم عرفنا مسألة NP التامة بأنها مسألة تتبع للصنف NP حيث إنها معقدة بحيث يمكن تقليص أي مسألة NP إلى مسألة NP التامة.

كما وضعنا تعريفين الأول للمسألة القرارية D_1 القابلة للتقليل بكثيرة حدود والثاني للمسألة القرارية D وهي مسألة NP تامة.

عزيزي الدارس هذه الوحدة ذات أهمية في هذا المقرر يرجى منك أن تكون قد فهمتها الفهم الجيد وحاول دائماً أن تستعين بمشرفك الأكاديمي في الأجزاء التي لم تفهمها وعزز فهمك بالرجوع للمصادر والمراجع في هذا المقرر – وفقك الله.

مسرد المصطلحات

- **الحد الأدنى** **Lower Bound**

هو الحد الأدنى للكفاءة المطلوبة لأي خوارزمية لحل هذه المسألة.

- **الحد الأدنى التافة** **Trivial Lower Bound**

هو الحد الأدنى لعدد المدخلات المعالجة والمخرجات.

- **تقليص المسائل** **Problem Reduction**

إثبات أن أي حالة عشوائية للمسألة Q يمكن تحويلها إلى حالة أخرى للمسألة P.

- **شجرة القرارات** **Decision Tree**

هي شجرة تحتوي على الجزر والأوراق بحيث أن كل ورقة في كل مرحلة داخل الشجرة تمثل نتيجة محتملة في الخوارزمية التي تعمل على n من مدخلاتها.

- **التعقيد المحوسب** **Computational Complexity**

هو تصنيف المسائل الحاسوبية إلى مسائل قابلة للحل في زمن محدد بكثيرة حدود أو مسائل غير قابلة للحل في زمن محدد.

- **مسألة P** **P Problem**

مسألة قابلة للحل في زمن محدد بكثيرة حدود.

- **مسألة NP**

مسألة قرارية يمكن أن نجد لها حلاً بواسطة خوارزمية غير محددة بكثيرة حدود.

- **مسألة NP التامة**

هي مسألة تتبع للصنف NP حيث إنها معقدة ويمكن لكل مسألة NP أن يتم تقليصها لمسألة NP التامة.

- **خوارزمية محددة Deterministic Algorithm**

هي خوارزمية تقوم بإدخال كمدخلات وتقوم بإخراج نعم إذا كان S يمثل حلاً للحالة I وإذا كان غير ذلك.

- **خوارزمية غير محددة بكثيرة حدود**

NonDeterministic Polynomial

إذا كان المعدل الزمني لمرحلة التحقيق فيها هو معدل محدد بكثيرة حدود.

المراجع

المراجع العربية

1) مجموعة مؤلفين، تركيب البيانات وتصميم الخوارزميات، منشورات جامعة القدس المفتوحة، 1998.
2) السماني عبد المطلب، هياكل البيانات، منشورات جامعة السودان المفتوحة، 2005.

المراجع الأجنبية

1) Weiss, M.A., Data Structures and Alogrithm Analysis, Benjamin pub., 1992.
2) McConnell, J.J., Analysis of Alogrithms: An Active Approach, Jones pub., 2001.
3) Lavitin, A., Introduction to the Design and Analysis of Alogrithms, Addisions ues by, 2003.
4) Ulman, J.D (etal), The Design and Analysis of Compurte Alogrithms, Addison Wesley, 1974.
5) Basese, S., Compurte Alogrithms : Introduction to Analysis and Design, Addison Wesley, 2000.
6) Sedgewick, R., An Introduction to the Analysis of Alogrithms, Addison Wesley 1996.