جَامعة المَنارة
MANARA UNIVERSITY

كلية الهندسة
قسم الهندسة المعلوماتية

مقرر خوارزميات بحث ذكية

د. غزوان علي ريّا

محاضرات الأسبوع الثالث
الفصل الأول 2024-2025

محتويات المحاضرة

خوارزمية البحث بالعمق أولاً DFS
تحسينات خوارزميتي DFS, BFS
البحث المحدود بالعمق DLS
البحث بالعمق أولاً المُعمَق بشكل تكراري IDDFS

كما ذكرنا في المحاضرة السابقة، خوارزميات البحث بدون معلومات هي:

البحث بالعرض أولاً Breadth-First Search (BFS)

البحث بالعمق أولاً Depth-First Search (DFS)

البحث المحدود بالعمق Depth-Limited Search (DLS)

البحث بالعمق أولاً المُعمَق بشكل تكراري Iterative Deepening Depth-First Search (IDDFS)

البحث ثنائي الاتجاه Bidirectional Search

البحث ذو التكلفة الموحدة Uniform Cost Search

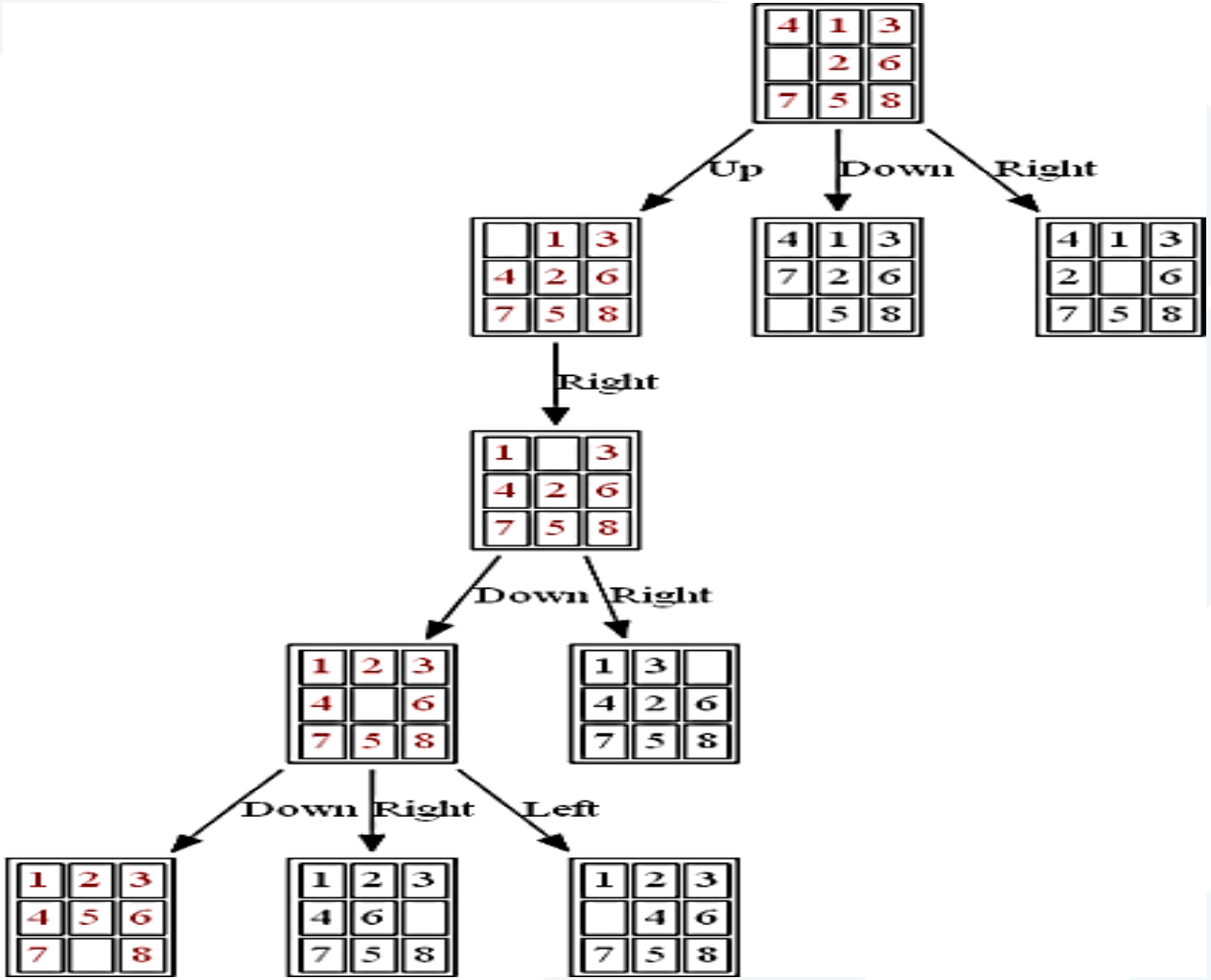
بعدما ناقشنا كيفية عمل خوارزمية BFS سنناقش خوارزمية البحث بالعمق أولاً DFS

الفكرة الأساسية: تبدأ DFS من عقدة جذرية وتتعمق في الفرع الأيسر قدر الامكان قبل العودة إلى العقدة السابقة والانتقال الى الفرع الأيمن.

أي أنها تزور كل خلف للعقدة الحالية قبل الانتقال لعقدة أخرى بحيث يتم وضع العقد المولدة في مكدس والمكدس يعمل حسب مبدأ Last-In-First-Out LIFO أي أن من يدخل إلى المكدس أخيراً يخرج أولاً.

مثال:

توضح الصورة الآتية جزء من شجرة البحث بالعمق لمسألة المربعات المنزلقة حيث يتم توليد كل خلف للعقدة قبل الانتقال لعقدة أخرى.



وخوارزمية البحث بالعمق هي كما يلي:

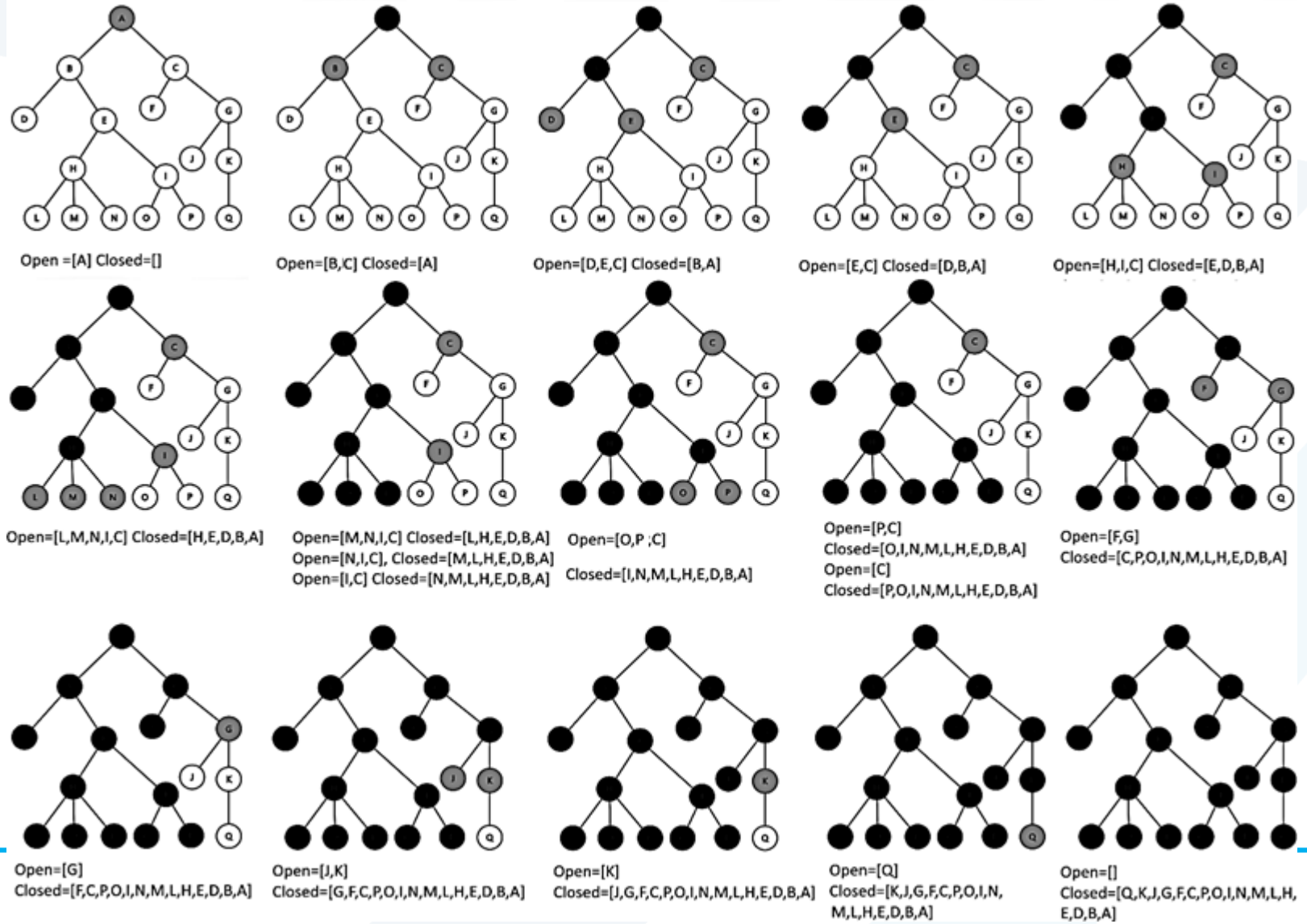
```
Begin                                   %initialize
open:= [Start];
closed:= [];
while open ≠ [] do %states remaining
remove leftmost state from open,
call it X;
If X is a goal then return SUCCESS %goal found
   else begin
    generate children of X;
    put X on closed;
    discard children of X if already on
open or closed;                 %loop check
put remaining children on left end of
Open                                    %stack
   end
   end;
 return FAIL
 end. %no states left
```

سنوضح الآن خطوات البحث بالعمق أولاً عبر مجموعة من التمارين:

تمرين 1:

ما هو ترتيب العقد المكتشفة حسب خوارزمية ال DFS في المثال التالي؟

توضح الصورة الآتية كل خطوة من الخطوات بالتفصيل مع ذكر المجموعتين Open و Closed في كل مرحلة (العقد الملونة باللون الرمادي تمثل المجموعة Open والعقد السوداء هي Closed).

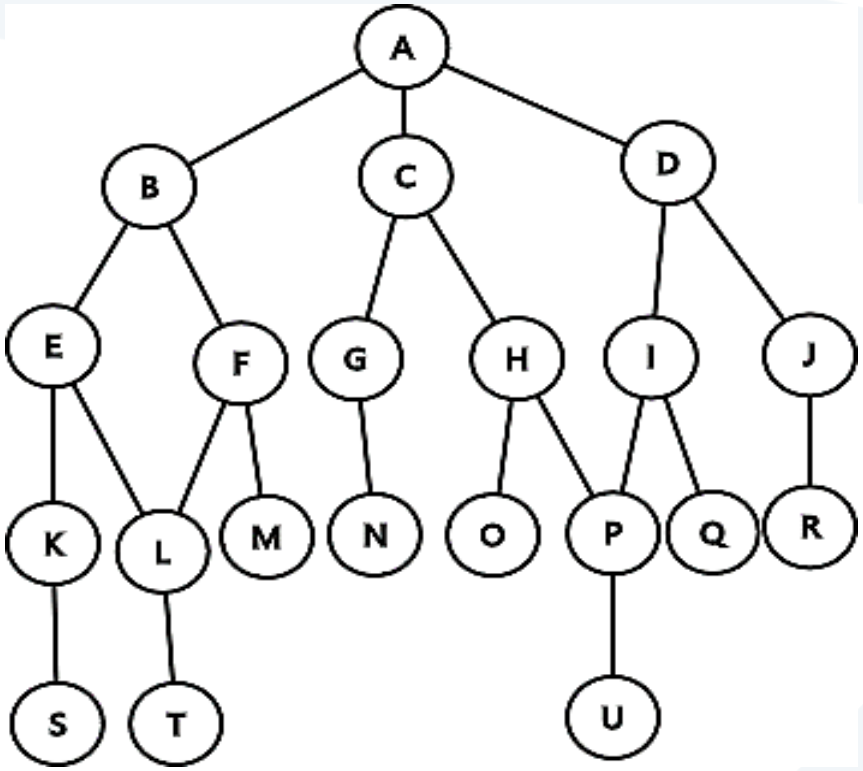


كمقارنة سريعة مع خوارزمية ال BFS خوارزمية ال DFS تضع أبناء العقدة على يسار المجموعة Open وبالتالي تمر على كل خلف للعقدة قبل الانتقال لعقدة أخرى من نفس مستوى العقدة.

المجموعة Closed عند عكس ترتيب عناصرها تعطي ترتيب العقد المكتشفة فهنا الترتيب هو A B D E H L M N I O P C F G J K Q.

تمرين 2:

ليكن لدينا البيان التالي المطلوب تطبيق خوارزمية البحث بالعمق أولاً عليه.



بشكل مشابه لخوارزمية البحث بالعرض أولاً BFS سنقسم العقد لمجموعتين Closed و Open:

Open = [A] Closed = []

Open = [B,C,D] Closed = [A]

Open = [E,F,C,D] Closed = [B,A]

Open = [K,L,F,C,D] Closed = [E,B,A]

Open = [S,L,F,C,D] Closed = [K,E,B,A]

Open = [L,F,C,D] Closed = [S,K,E,B,A]

هنا بما أن S لا تملك أولاد نضيفها على ال Closed مباشرة وننتقل للعقدة L.

Open = [T,F,C,D] Closed = [L,S,K,E,B,A]

Open = [F,C,D] Closed = [T,L,S,K,E,B,A]

هنا بما أن T لا تملك أولاد نضيفها على ال Closed مباشرة وننتقل للعقدة F.

Open = [M,C,D] Closed = [F,T,L,S,K,E,B,A]

لا نضع العقدة L مرة أخرى في Open لأنها موجودة مسبقاً في Closed (نقوم بنفس الشيء إن كانت موجودة مسبقاً في Open).

Open = [C,D] Closed = [M,F,T,L,S,K,E,B,A]

هنا بما أن M لا تملك أولاد نضيفها على ال Closed مباشرة وننتقل للعقدة C.

Open = [G,H,D] Closed = [C,M,F,T,L,S,K,E,B,A]

ونكمل بنفس الطريقة حتى نغطي كل عقد البيان.

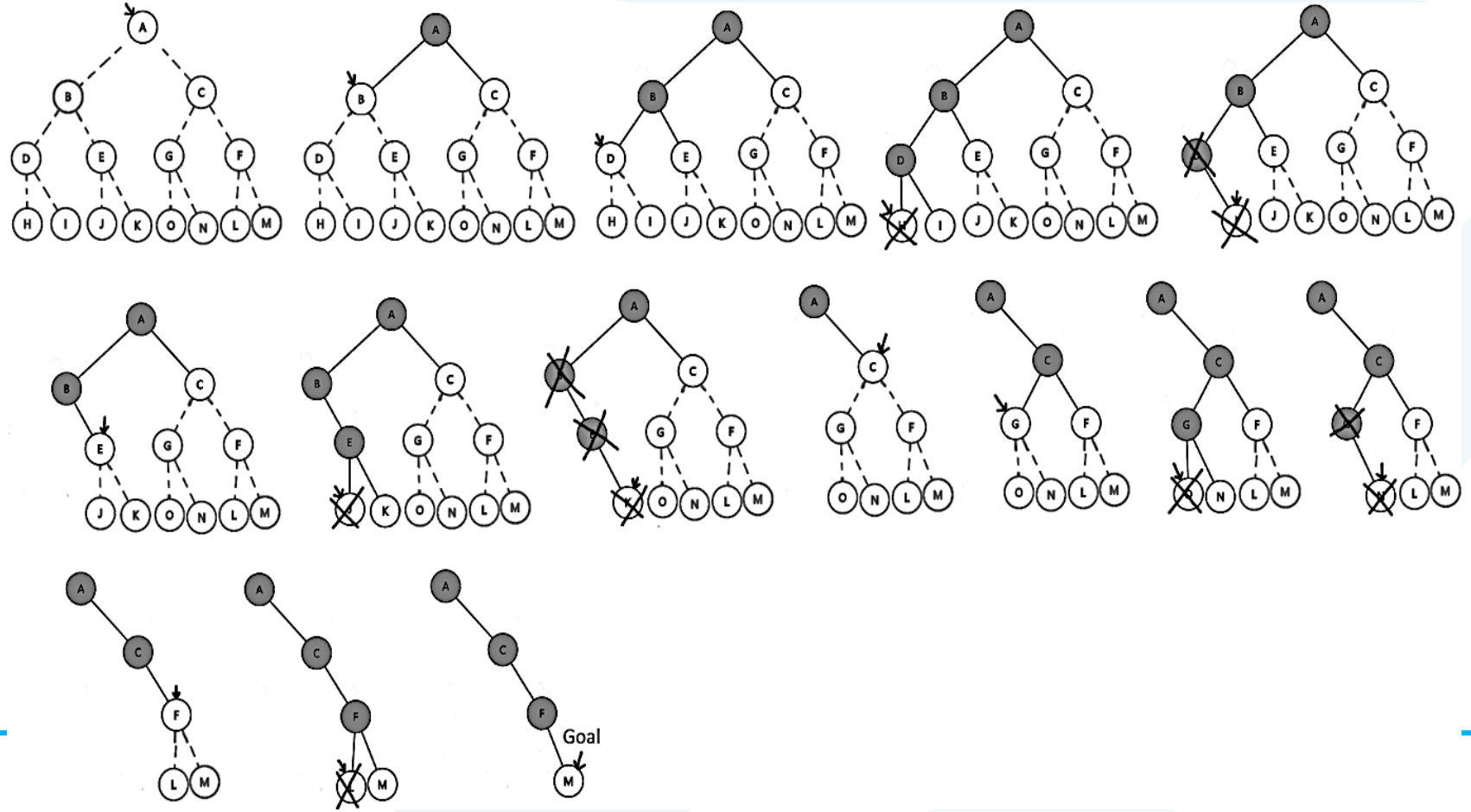
البحث بالعمق أولاً على شجرة ثنائية DFS on a binary tree

الشجرة الثنائية هي شجرة تملك فيها كل عقدة ولدين على الأكثر.

عند تطبيق ال DFS على شجرة ثنائية، يتم حذف العقد المكتشفة في المجموعة الأمامية التي لا تملك أبناء من الذاكرة.

يوضح الشكل التالي العملية حيث الحالة الهدف هي M

تظهر الحالات الغير مكتشفة Unexplored بلون فاتح والمكتشفة بلون رمادي، ويشير السهم للحالة الحالية. نلاحظ حذف العقد في المستوى الثالث التي لا تملك خلفاً.



في المرحلة الرابعة لاحظنا أن العقدة H لا تملك أبناء ولذلك حذفناها في المرحلة الخامسة

وفي المرحلة الخامسة لاحظنا أن العقدة الحالية لا تملك أبناء وبالتالي حذفناها ونتيجة حذفها أصبحت العقدة الأب بلا أبناء وحذفناها وكررنا الخطوات حتى وصلنا للهدف وحصلنا على الطريق.

تحسينات خوارزميتي ال DFS, BFS

خوارزمية البحث بالعرض أولاً **BFS** هي خوارزمية **تامة Complete** ولكن **مكلفة Expensive**.

أي أنه بالرغم من أن خوارزمية البحث بالعرض أولاً تجد حلاً دائماً (تحقق شرط التمام) إلا أنها مكلفة من ناحية التعقيد الزمني أو المساحي.

بالمقارنة خوارزمية البحث بالعمق أولاً **DFS** هي خوارزمية ليست تامة **Incomplete** ولكن **رخيصة Cheap**.

أي أن خوارزمية البحث بالعرض أولاً ليست بالضرورة أن تجد حلاً دائماً (يحدث هذا في فضاء حالات غير منتهي حيث يمكن أن تستمر الخوارزمية بالغوص بالأعماق دون نهاية) ولكنها رخيصة أي أنها ذات تعقيد زمني أو مساحي أقل.

ولهذا هناك تحسينات لخوارزمية الDFS وهما:

البحث المحدود بالعمق Depth limited search

التعميق بشكل تكراري Iterative deepening

ولكن كما سنرى أنه حتى مع مثل هذه التحسينات يمكن أن يكون مثل هذا النمط من البحث غير واقعي بشكل ميؤوس منه للمشاكل الواقعية.

قياس الأداء عند حل المشكلة:

المعيار النموذجي للتعقيد الزمني والمساحي هو حجم بيان فضاء الحالة والذي يعطى كمايلي:

$$[V] + [E]$$

حيث [V] هي عدد العقد و [E] هي عدد الأضلاع.

وهذا مناسب عندما يكون البيان هو بنية معطيات صريحة Explicit data structure.

في الذكاء الصنعي، غالباً ما يتم تمثيل البيان بشكل ضمني عبر حالة البداية Initial state، الأفعال Actions، أنماط الانتقال Transition Models، وغالباً هذا البيان هو غير منته.

ولهذه الأسباب يُعطى التعقيد عبر هذه الكميات:

b: والذي هو معامل التفرع الأعظمي لشجرة البحث أي أنه أكبر عدد من الأبناء لعقدة ما في الشجرة.

في التمرين 1 معامل التفرع الأعظمي Maximum Branching factor هو 3 حيث أن العقدة H والتي تملك أكبر عدد من الأولاد عدد أولادها هو 3.

d: هو عمق أقل حل مكلف.

m: هو عمق الشجرة الأعظمي وقد يكون العمق غير منته.

وغالباً ما يُقاس التعقيد الزمني بعدد العقدة المتولدة خلال البحث، والتعقيد الفضائي/المساحي بعدد العقد المخزنة في الذاكرة.

خصائص البحث حسب العرض أولاً BFS:

هو **تام Complete** بشرط أن يكون معامل التفرع b **منتهياً finite**.

يحتاج **زمن** هو $O(b^{d+1})$ $1 + b + b^2 + b^3 + \ldots + b^d$ أي أنه يحتاج زمن أسي من أجل عمق ما d.

(يعود السبب إلى أن كل عقدة تتفرع إلى b مستوى ففي حال كان معامل التفرع 3 ستتفرع أول عقدة إلى 3 عقد وكل عقدة من هذه العقد ستتفرع إلى 3 عقد بالتالي 9 عقد ثم كل عقدة ستتفرع ل 3 عقد بالتالي 27 عقدة وهذا كما نلاحظ هو تزايد أسي).

يحتاج **مساحة** هي $O(b^{d+1})$ حيث أنه يخزن كل عقدة في الذاكرة.

يعطي **حل أفضلي في حال كانت الكلفة هي 1 من أجل أي انتقال**، ولكن باستثناء هذه الحالة **لا يعطي حل أفضلي بالعموم**.

مثال عن متطلبات الذاكرة والزمن عند تنفيذ خوارزمية BFS:

الذاكرة	الوقت	عدد العقد	العمق
107 Kbytes	0.11 msecond	110	2
10.6 Mbytes	11 msecond	11110	4
1 Gbytes	1.1 seconds	10^6	6
103 Gbytes	2 minutes	10^8	8
10 Tbytes	3 hours	10^{10}	10
1 Petabytes	13 days	10^{12}	12
99 Petabytes	3.5 years	10^{14}	14
10 Exabytes	350 years	10^{16}	16

خصائص البحث حسب العمق أولاً DFS

- هو **ليس تام not complete** حيث أنه يفشل في حال كان فضاء الحالات غير منته أي ذو عمق غير منته Infinite depth.

وفي حالة الفضاءات التي تحوي حلقات loops يجب التعديل عليها لتجنب الحالات المتكررة **وبالتالي نجعله تام complete بجعل الفضاء الاحتمالي منته finite space.**

- يلزمه زمن هو $O(b^m)$ حيث m هي العمق الأعظمي لشجرة البحث،

ويكون زمن التنفيذ كارثي في حال كان العمق الأعظمي m أكبر بكثير من d عمق أقل حل تكلفة (أكثر حل سطحي) **ولكن في حال كانت الحلول كثيفة dense فيمكن أن يكون أسرع بكثير من البحث حسب العرض أولاً BFS.**

- يحتاج **لمساحة** هي $O(bm)$ وهذه المساحة خطية linear (أي أنه من ناحية المساحة أقل استهلاكاً من خوارزمية BFS).
- **ليس بالضرورة أن يجد حل أفضلي not optimal.**

البحث المحدود بالعمق Depth Limited Search DLS:

هو تعديل لخوارزمية DFS يقوم بالتقليل من العمق التي قد تسلكه خوارزمية البحث،

يقوم بإعطاء عمق معين لن تتجاوزه الخوارزمية،

وكل عقدة في مستوى أعمق من العمق المحدد سيتم إهمالها في عملية البحث،

- أي أنه يقوم بتوسعة شجرة البحث حتى عمق أعظمي l فتتم معاملة العقد في المستوى l كأنها لا تملك خلفاً وهذا ما يمنع الخوارزمية من الدخول في حلقة غير محدودة عبر جعل البحث يتوقف بعد العمق المفروض مسبقاً.

في حال وصلت الخوارزمية لعقدة في المستوى l ليست حلاً فتعود الخوارزمية لنقطة اختيار مسبقة في المستويات السابقة (كما في خوارزمية ال DFS) التي عمقها أصغر من l.

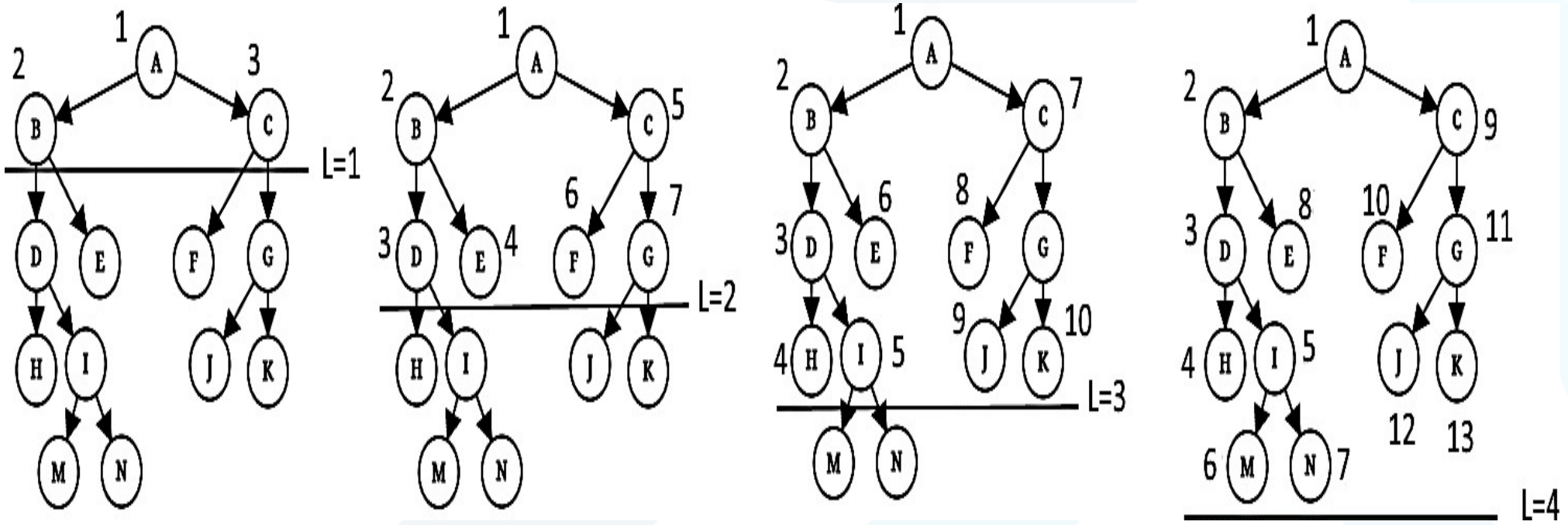
فيمكننا اعتبار أن خوارزمية ال DFS هي حالة خاصة من ال DLS حيث العمق غير منته $l = \infty$

ويمكن اختيار العمق المحدد حسب معرفتنا بالمشكلة.

يجب أيضاً حساب مستوى كل عقدة للتأكد أنها في المستوى المسموح وينتهي البحث المحدود بالعمق في حالتين: **عند إيجاد حل أو عندما لا يكون هناك حل داخل العمق المسموح.**

مثال:

تشير هنا الأرقام إلى ترتيب اكتشاف العقد وتوضح كل صورة المستوى المسموح.



البحث بالعمق أولاً المُعمَق بشكل تكراري Iterative Deepening Depth-First Search :IDDFS

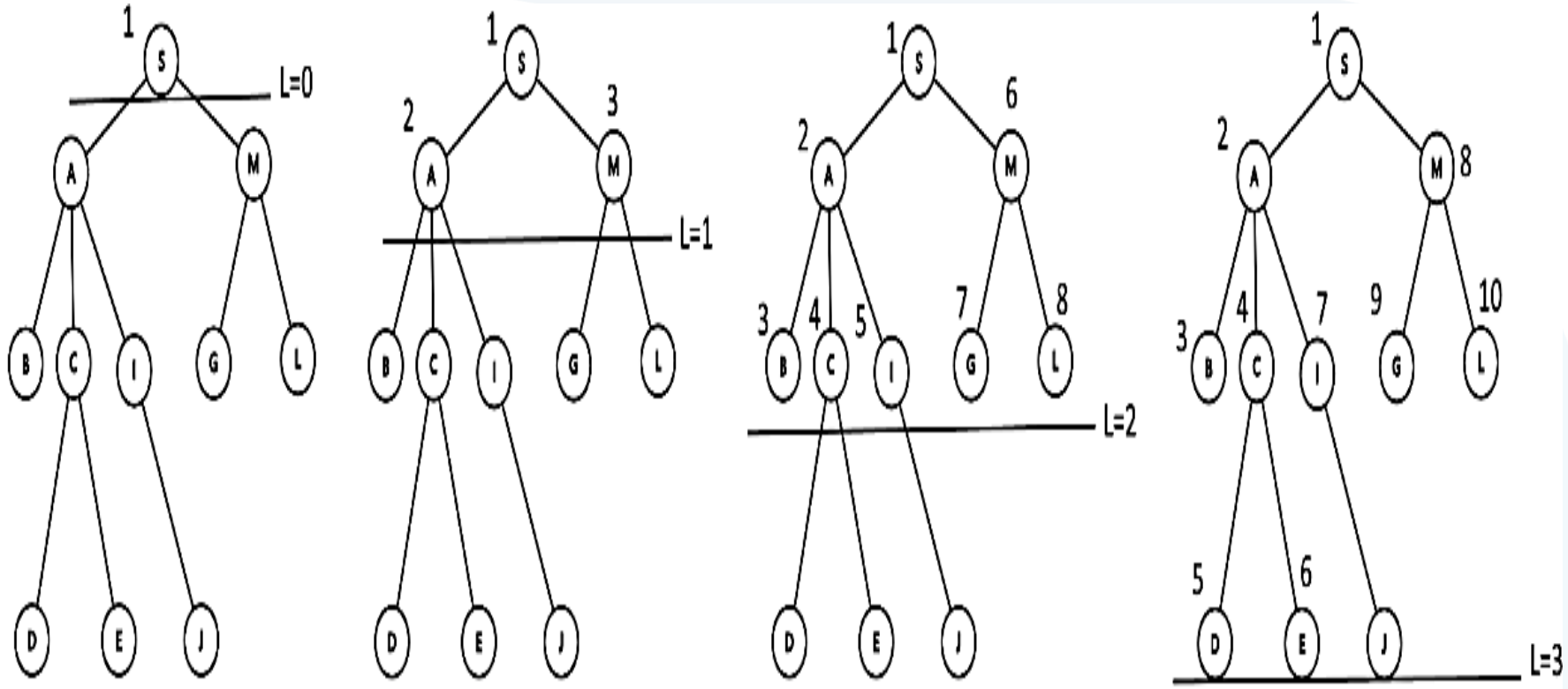
يدمج هذا النمط كفاءة ال DFS من ناحية التعقيد الفضائي/المساحي، وسرعة BFS في البحث (عندما تكون العقد أقرب لجذر الشجرة).

يأتي المصطلح الإنكليزي من قسمين التعميق التكراري iterative deepening ID والبحث بالعمق أولاً Depth-First Search DFS.

فيقوم البحث IDDFS بتطبيق خوارزمية DFS على مستويات مختلفة بدءاً من قيمة ابتدائية وعند كل استدعاء يتم منع ال DFS من تخطي عمق محدد أي أننا نقوم بخوارزمية ال DFS بطريقة ال BFS.

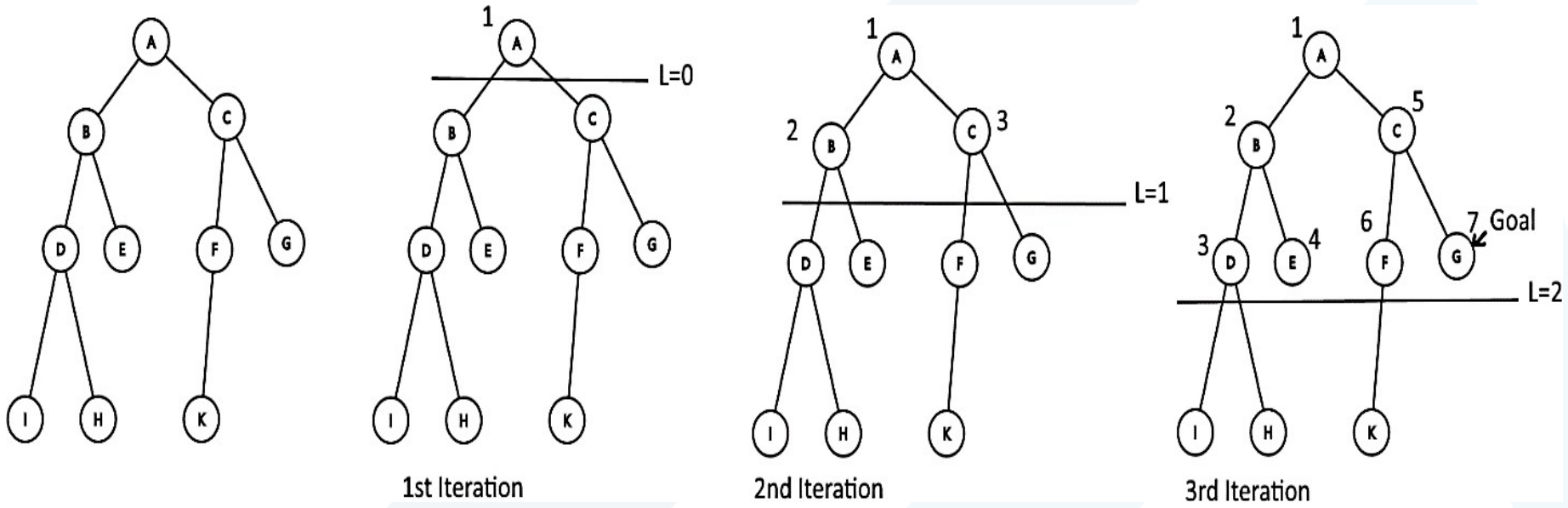
ويجب ملاحظة أننا نزور العقد في المستويات العليا عدة مرات والعقد في المستوى الأخير (العمق الأعظمي) مرة واحدة والعقد في المستوى قبل الأخير تُزار مرتين وهكذا فقد يبدو هذا الأمر مكلفاً ولكن يتضح أنه ليس كذلك حيث أن معظم العقد في شجرة توجد في المستوى الأخير ولهذا ليس مهماً لهذه الدرجة إن زرنا العقد في المستويات العليا عدة مرات.

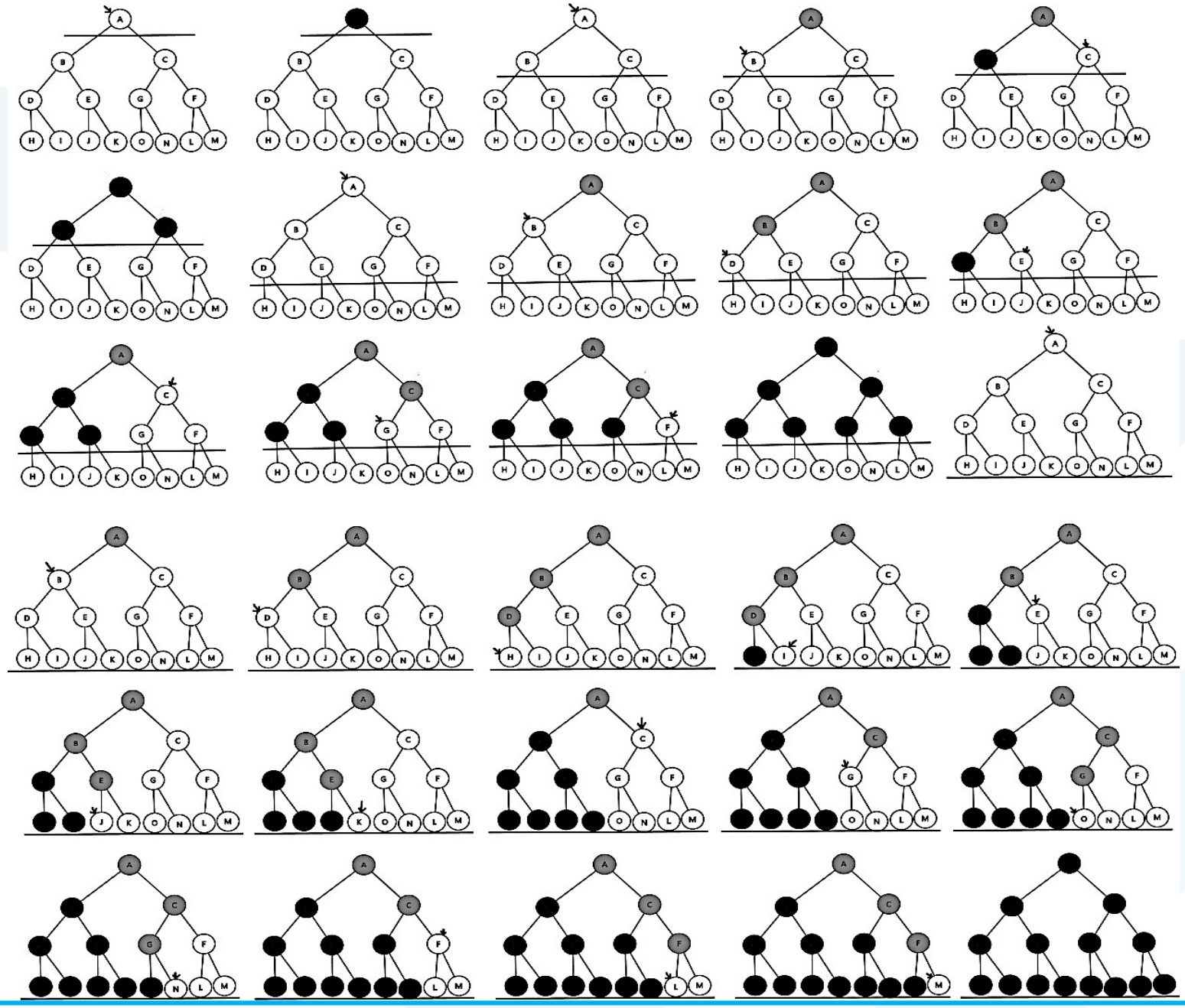
في معظم مسائل البحث، يكون أقل طريق تكلفة للحل d هو مجهول فتقوم خوارزمية ال IDDFS بخوارزمية ال DFS وتزيد العمق المسموح تدريجياً فتبدأ من عمق l يساوي 1 ثم 2 ثم 3 ثم ... حتى نجد حل في مستوى ما $l = d$ ويوضح المثال هنا كيف نقوم بهذه العملية.



كما نلاحظ نقوم عند نهاية كل مرحلة تدريجياً بزيادة العمق المسموح (الفرق بين ال DLS و ال IDDFS هو أنه في ال DLS يكون هناك عمق محدد معطى مع المسألة بينما ال IDDFS نقوم بزيادة العمق عند نهاية كل تكرار).

تمرين 5: طبق خوارزمية IDDFS على البيان التالي علماً أن الحالة الهدف هي G.





كما نرى في هذا المثال نبدأ في أول تكرار بعمق 0 ويكون ترتيب العقد المكتشفة هو A ثم نزيد العمق ليصبح 1 ويكون ترتيب العقد المكتشفة A B C ثم نزيد العمق ليصبح 2 فيكون ترتيب العقد المكتشفة A B D E C F G وبما أننا وصلنا لعقدة هدف فنتوقف عن زيادة العمق.

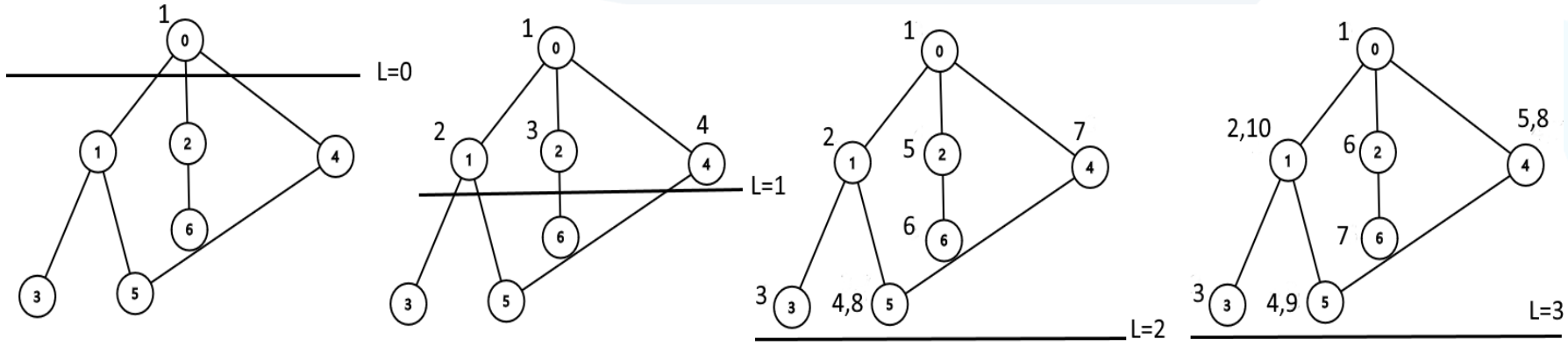
رسم توضيحي لعملية IDDFS في شجرة ثنائية حيث يمثل اللون الرمادي العقد التي لم يتم اكتشاف خلفُها بالأكمل واللون الأسود العقد التي تم اكتشاف كل خلف لها

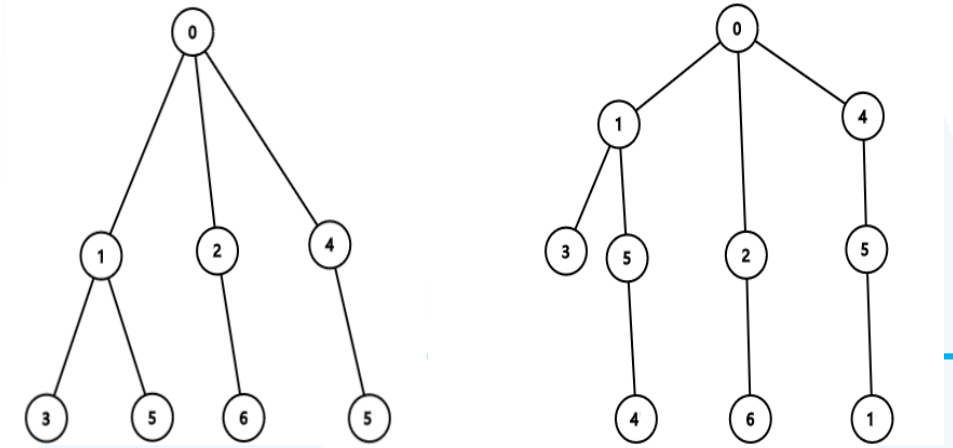
من ناحية البيان في خوارزمية IDDFS هناك حالتان:

عندما لا يحوي البيان على أية حلقات فالحالة بسيطة يمكن أن نستخدم DFS عدة مرات مع حدود ارتفاع مختلفة.

بينما إن حوى البيان على حلقات فيصبح الأمر أعقد حيث أن خوارزمية IDDFS لا تحوي أعلاماً تربطها بالعقد التي تمت زيارتها.

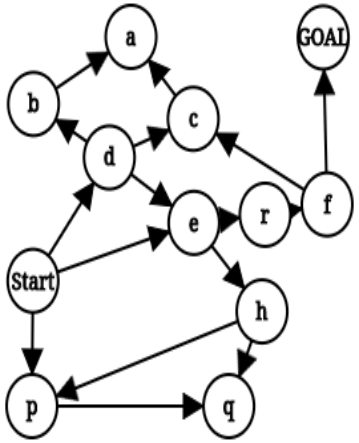
فمثلاً عند تطبيق خوارزمية IDDFS على البيان التالي يكون ترتيب العقد التي تمت زيارتها بالشكل التالي:



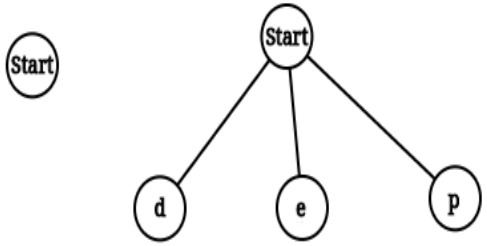


العمق	ترتيب العقد
0	0
1	0 1 2 4
2	0 1 3 5 2 6 4 5
3	0 1 3 5 4 2 6 4 5 1

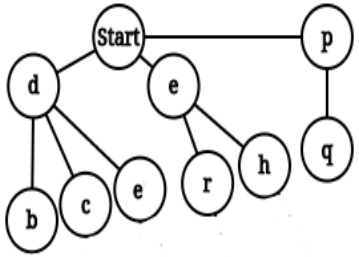
المستوى 0 والمستوى 1 واضحان ولكن المستوى 3 مختلف عن المستوى 2 بالرغم من أن الشخص قد يعتقد أنهما يُفترض أن يحويا نفس الترتيب، ولكن بسبب الحلقة يتغير الترتيب وتوضح الشجرتان سبب اختلاف ترتيب العقد حيث أن الحلقة أدت لتوليد أولاد لبعض العقد في المستوى 3 ما جعل المستوى 2 مختلف عن المستوى 3.



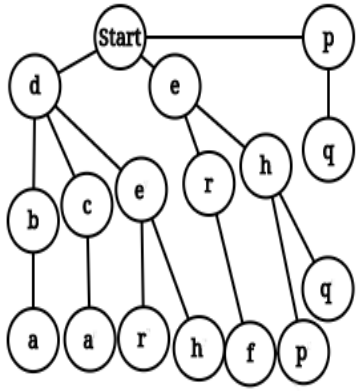
1



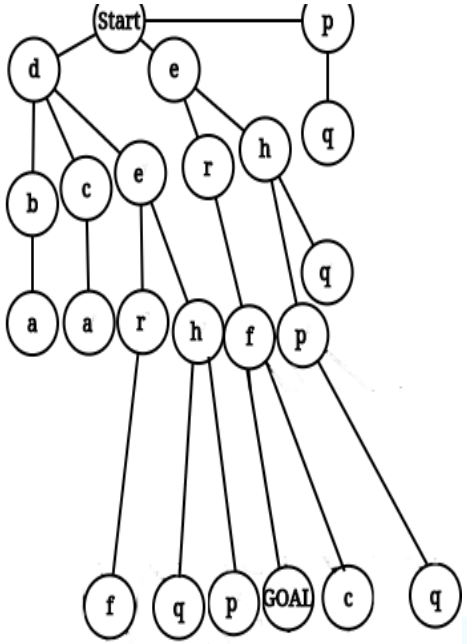
2



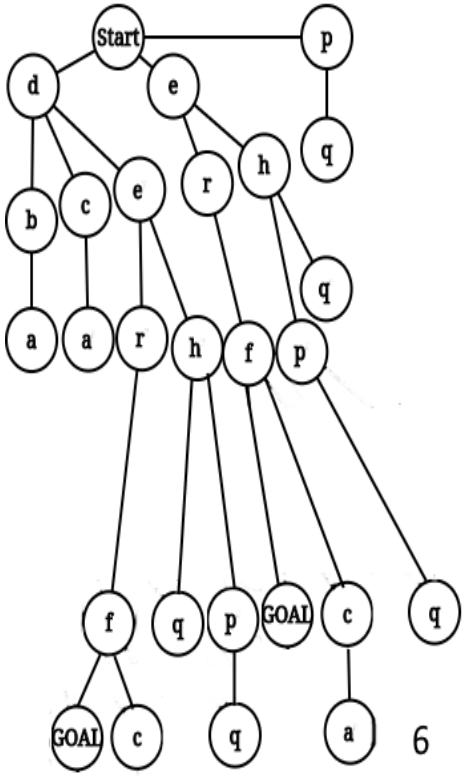
3



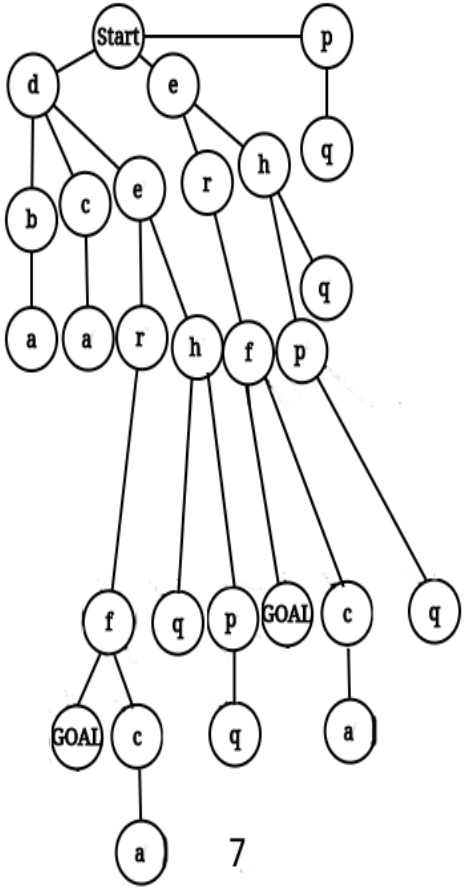
4



5



6



7

تمرين 6:

كما رأينا في المسألة السابقة فقد تؤدي الحلقات في البيان لنتائج معقدة عند تطبيق IDDFS فالهدف من هذا التمرين هو تحويل بيان إلى شجرة حسب المستويات.

كما نرى في هذا التمرين نقوم أولاً بإضافة عقدة البداية start في المرحلة الأولى

ثم نضيف أبناء العقدة start في المرحلة الثانية وهي d p e

وفي المرحلة الثالثة نضيف أبناء العقدة d وهي b c e

وعند التحويل من بيان لشجرة نضيف العقدة ولو كانت مُضافة سابقاً أي أننا نضيف جميع أبناء العقدة

ثم نضيف أبناء العقدة e وهي r h

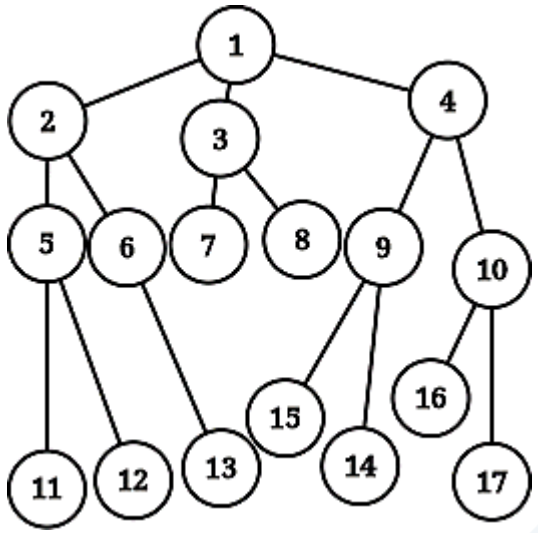
وأبناء العقدة p وهي q ونصبح في المرحلة 4

ونكرر باقي الخطوات حسب المستوى المطلوب (في هذا المثال لا يمكن الوصول لأكثر من مستوى 7 حيث أن أوراق الشجرة رقم 7 لا تملك أطفال في البيان الأصلي).

تمرين 7: وضح كيفية عمل الخوارزميات التالية BFS DFS IDDFS على هذه الشجرة عبر إعطاء الترتيب التي يتم به اكتشاف العقد.

ملاحظة: يمكن حل قسم ال DFS و BFS باستخدام طريقة المجموعتين Closed Open ولكن سنستعمل هنا طريقة أخرى وهي Node Fringe حيث ال Node تمثل العقدة الحالية وال Fringe تكون إما الرتل أو المكدس (حسب نوع الخوارزمية).

بالنسبة ل DFS:

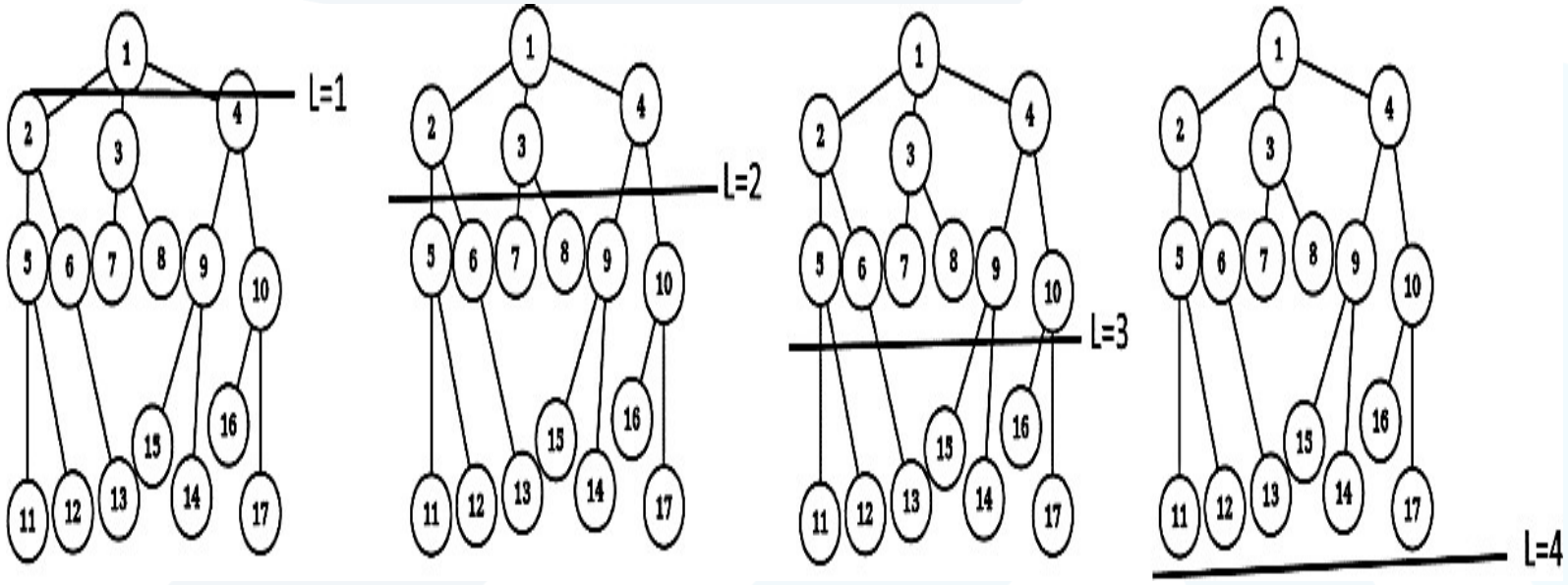


Node	Fringe
-	1
1	2 3 4
2	5 6 3 4
5	11 12 6 3 4
11	12 6 3 4
12	6 3 4
6	13 3 4
13	3 4
3	7 8 4
7	8 4
8	4
4	9 10
9	15 14 10
15	14 10
14	10
10	16 17
16	17
17	-

والآن بالنسبة ل BFS الجدول هو كالتالي:

Node	Fringe
-	1
1	2 3 4
2	3 4 5 6
3	4 5 6 7 8
4	5 6 7 8 9 10
5	6 7 8 9 10 11 12
6	7 8 9 10 11 12 13
7	8 9 10 11 12 13
8	9 10 11 12 13
9	10 11 12 13 15 14
10	11 12 13 15 14 16 17
…….	…….

عند تطبيق خوارزمية IDDFS سنذكر كل مستوى ونتيجة الخوارزمية:



L=1	L=2	L=3	L=4

L=1		L=2	
Node	**Fringe**	**Node**	**Fringe**
-	1	-	1
1	-	1	2 3 4
		2	3 4
		3	4
		4	-

L=3		L=4	
Node	Fringe	Node	**Fringe**
1	2 3 4	1	**2 3 4**
2	5 6 3 4	2	**5 6 3 4**
5	6 3 4	5	**11 12 6 3 4**
6	3 4	11	**12 6 3 4**
3	7 8 4	12	**6 3 4**
7	8 4	6	**13 3 4**
8	4	13	**3 4**
4	9 10	3	**7 8 4**
9	10	7	**8 4**
10	-	8	**4**
		4	**9 10**
		9	**15 14 10**
		15	**14 10**
		14	**10**
		10	**16 17**
		16	**17**
		17	**-**

(قد يأتي سؤال في الامتحان مثل هذا التمرين حيث يطلب تطبيق أكثر من خوارزمية على بيان ما).