الباب الثالث

الذكاء الصنعي

(Artificial Intelligence)

الفصل الأول

حساب الفرضيات

The Propositional Calculus

1 - 1 - أهمية حساب الفرضيات

- تُشكّل الفرضيات ثنائية القيمة وصفاً للعالم (ما هو صحيح في هذا العالم وما هو غير صحيح).
- **أمثلة:**
  - "لا توجد الكتلة A على الأرض".
  - "توجد الكتلة A إما فوق الكتلة B وإما فوق الكتلة C".
- يُمكن صياغة بعض المعلومات عن العالم على شكل قيود على قيم الفرضيات فيه.
- تُمثّل هذه القيود معارف مهمة حول العالم.
- كما يُمكن استخدامها لتوليد قيم فرضيات أخرى غير قابلة للقياس مباشرة.

مثال:

- ليكن لدينا ربوط قادر على حمل كتلة. إذا كانت هذه الكتلة قابلة للحمل (أي غير ثقيلة جداً) وإذا كانت شحنة بطارية الربوط كافية.
- إذا تحقق كل من هذين الشرطين سيقوم الربوط برفع الكتلة التي يُمسكها وذلك بتحريك ذراعه.

• يُمكن تمثيل مختلف هذه الشروط باستخدام الفرضيات ثنائية القيمة التالية:

• *BAT_OK* (البطارية مشحونة)

• *LIFTABLE* (الكتلة قابلة للحمل)

• *MOVES* (الذراع تتحرك)

• لنفترض أن الربوط يتحسس قيمة *BAT_OK* عن طريق قراءة حساس، وقيمة *MOVES* بوساطة حساسات موضعه عليه. إلا أنه لا يتحسس قيمة *LIFTABLE* والتي تُشكّل قيمة مهمة له.

• **لنجري الآن المحاكمة التالية:**

• نعرف أنه إذا كان لكل من *BAT_OK* و *LIFTABLE* القيمة *1* فسيكون لـ *MOVES* القيمة *1* أيضاً.

• كذلك إذا كان لـ *MOVES* القيمة *0* عندما يحاول الربوط أن يحرك الكتلة فإننا نعرف أنه إما لـ *BAT_OK* وإما لـ *LIFTABLE* (أو لكليهما) القيمة *0*.

• إلا أنه إذا كان لـ *BAT_OK* القيمة *1* (حسب الحساس المرتبط) فيبقى أن يكون لـ *LIFTABLE* القيمة *0*.

• **بما أننا نستطيع المحاكمة هكذا، لنجعل الربوط يحاكم مثلنا!**

• نحتاج إذاً للغة لنمذجة العالم.

• كما نحتاج إلى آلية استدلال نستطيع بوساطتها تحقيق المحاكمة المطلوبة.

• يُقدم حساب الفرضيات والمتفرع عن الجبر المنطقي الأدوات الضرورية لذلك.

1 - 2 - الشكل Syntax - مكونات اللغة

- الذرات *Atoms*

- لدينا أولاً الذرتان .T, F

- المجموعة المعدودة وغير المنتهية من السلاسل الحرفية التي تبدأ بحرف كبير. مثلاً:

$P, Q, R, P1, P2, ON\_A\_B, \ldots$

- الروابط *Connectors*

- $\vee$ والتي تُدعى "أو" "or"

- $\wedge$ والتي تُدعى "و" "and"

- $\neg$ والتي تُدعى "لا" "not"

- $\Rightarrow$ والتي تُدعى "يقتضي" "implies"

- الصيغ جيدة التركيب *Well-Formed Formulas WFF*

- أي ذرة هي صيغة جيدة التركيب. مثلاً: R , P

- إذا كانت $w1, w2$ صيغاً جيدة التركيب فإن كل من الصيغ التالية هي صيغة جيدة التركيب:

- $w1 \vee w2$
- $w1 \wedge w2$
- $w1 \Rightarrow w2$
- $\neg w1$

ندعو الذرة أو الذرة المسبوقة بإشارة النفي $\neg$ بحرفي *Literal*.

ندعو $w1$ في $w1 \Rightarrow w2$ بمقدمة الاقتضاء *Antecedent* و $w2$ بنتيجة الاقتضاء *Consequent*.

- لا يوجد صيغ أخرى جيدة التركيب. فمثلاً $P \Rightarrow \neg\neg$ ليست صيغة *wff*.

أمثلة:

- $(P \wedge Q) \Rightarrow \neg P$
- $P \Rightarrow \neg P$
- $P \vee P \Rightarrow P$
- $(P \Rightarrow Q) \Rightarrow (\neg Q \Rightarrow \neg P)$
- $\neg\neg P$

1 - 3 - الدلالة *Semantic*

إذا أُعطيت قيم الذرات في تفسير *interpretation* ما، فيُمكن استخدام جدول الحقيقة لحساب قيمة أي صيغة *wff* في هذا التفسير.
يُعطي جدول الحقيقة دلالة (معنى) الروابط في حساب الفرضيات.

$w1$	$w2$	$w1 \wedge w2$	$w1 \vee w2$	$\neg w1$	$w1 \Rightarrow w2$
True	True	True	True	False	True
True	False	False	True	False	False
False	True	False	True	True	True
False	False	False	False	True	True

- نقول عن تفسير إنه يُحقق صيغة *wff* إذا كان للصيغة القيمة *True* تحت هذا التفسير.
- نقول عن تفسير يُحقق صيغة إنه نموذج لها *Model*.
- نقول عن صيغة أنها غير قابلة للتحقيق *Inconsistent* أو *Unsatisfiable* إذا لم يوجد تفسير يُحققها.
- نقول عن صيغة إنها صالحة *Valid* إذا كان لها القيمة *True* من أجل كل تفسير لذراتها المكونة.

أمثلة:

- الصيغ: *False* و $P\wedge\neg P$ غير قابلة للتحقيق.
- مجموعة الصيغ: $\{P\vee Q,\ P\vee\neg Q,\ \neg P\vee Q,\ \neg P\vee\neg Q\}$ غير قابلة للتحقيق، إذ لا يوجد أي تفسير يجعل جميع صيغ هذه المجموعة صحيحة. (استخدم جدول الحقيقة للتأكد من ذلك).
- جميع الصيغ التالية صالحة:
  - $P\Rightarrow P$
  - T
  - $\neg(P\wedge\neg P)$
  - $Q\vee T$
  - $[(P\Rightarrow Q)\Rightarrow P]\Rightarrow P$
  - $P\Rightarrow(Q\Rightarrow P)$

لاحظ أن استخدام جدول الحقيقة للتأكد من صلاحية صيغة يتطلب تعقيداً أسياً وفق عدد الذرات المكونة لها، إذ يجب إيجاد قيمة الصيغة من أجل جميع القيم الممكنة لذراتها.

1 - 5 - التكافؤ Equivalence

نقول عن صيغتين *wff* إنهما متكافئتان إذا وفقط إذا كانت لهما قيم الحقيقة نفسها من أجل كل التفاسير. (سنرمز للتكافؤ بالإشارة ≡).

يُمكن استخدام جدول الحقيقة لبرهان التكافؤات التالية:

- قوانين دومرغان

$$\neg (w1 \vee w2) \equiv \neg w1 \wedge \neg w2$$

$$\neg (w1 \wedge w2) \equiv \neg w1 \vee \neg w2$$

- قانون عكس الإيجاب

$$(w1 \Rightarrow w2) \equiv (\neg w2 \Rightarrow \neg w1)$$

إذا كانت $w1$ و $w2$ متكافئتين فإن الصيغة التالية صالحة:

$$(w1 \Rightarrow w2) \wedge (w2 \Rightarrow w1)$$

وبسبب هذه الحقيقة فإن المفهوم:

$$w1 \equiv w2$$

كثيراً ما يُستخدم كاختصار لـ

$$(w1 \Rightarrow w2) \wedge (w2 \Rightarrow w1)$$

يوجد عدة طرائق لتوليد صيغ *wff* بدءاً من صيغ أخرى، ندعو هذه العملية بالاستدلال *Inference* أو الاستنتاج. يكون لقاعدة الاستدلال الشكل:

γ يمكن أن تستنتج من α

نعطي فيما يلي مجموعة من قواعد الاستدلال المستخدمة في حساب الفرضيات:

- يمكن للصيغة $w2$ أن تستنتج من الصيغتين $w1$ و $w1 \Rightarrow w2$ (*Modus Ponens*)
- يمكن للصيغة $w1 \wedge w2$ أن تستنتج من الصيغتين $w1$ و $w2$ (إدخال العطف $\wedge$)
- يمكن للصيغة $w2 \wedge w1$ أن تستنتج من $w1 \wedge w2$ (العطف تبديلي)
- يمكن للصيغة $w1$ أن تستنتج من الصيغة $w1 \wedge w2$ (حذف العطف $\wedge$)
- يمكن للصيغة $w1 \vee w2$ أن تستنتج إما من $w1$ وإما من $w2$ (إدخال الفصل $\vee$)
- يمكن للصيغة $w1$ أن تستنتج من الصيغة $\neg(\neg w1)$ (حذف النفي $\neg$)

1 - 7 - البرهان Proof

تُدعى سلسلة الصيغ جيدة التراكيب $\{w_1, w_2, \ldots\ldots w_n\}$ ببرهان أو استنتاج w_n من مجموعة من الصيغ جيدة التركيب Δ إذا وفقط إذا كانت:

كل صيغة w_i في السلسلة هي إما موجودة في Δ وإما يمكن استنتاجها من صيغة أو (عدة صيغ) سابقة في السلسلة باستخدام إحدى قواعد الاستنتاج.

إذا وجد برهان لـ w_n من Δ فإننا نقول إن w_n هي نظرية Theorem للمجموعة Δ ونكتب:

$$\Delta \mid\!- wn$$

مثال: ليكن لدينا مثلاً مجموعة الصيغ:

$$\Delta = \{P, R, P \Rightarrow Q\}$$

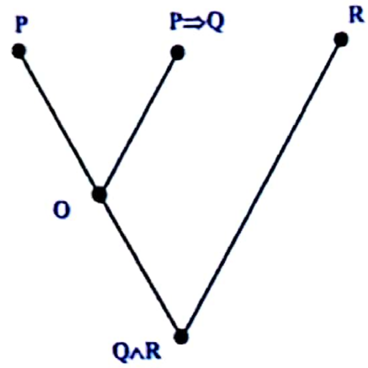
فإن السلسلة التالية هي برهان لـ $Q\grave{U}R$ باستخدام قواعد الاستدلال المعطاة في العبارة السابقة:

$$\{P, P \Rightarrow Q, Q, R, Q \wedge R\}$$

يُمكن تمثيل سلسلة البرهان باستخدام شجرة.

تُعنون كل عقدة في شجرة البرهان بصيغة *wff*.

تكون هذه الصيغة إحدى صيغ Δ أو صيغة مستنتجة من آبائها في الشجرة باستخدام قواعد الاستنتاج. وتكون الشجرة برهاناً لصيغة جذر الشجرة.



1 - 8 - الاستتباع *Entailment*

إذا كان لصيغة *wff* القيمة *True* من أجل كل التفسيرات التي تجعل قيمة كل صيغة في مجموعة Δ من الصيغ *True*. فإننا نقول إن Δ تستتبع *Entail* منطقياً *w*. وإن *w* تتبع *Follow* منطقياً Δ أو إن *w* هي نتيجة منطقية *Logical Consequence* من Δ.

نستخدم الرمز |= للدلالة على الاستتباع المنطقي ونكتب $D|=w$.

أمثلة:

- $P\} \mid= P\}$
- $\{P, P \Rightarrow Q\} \mid= Q$
- $F \supset w$ (أي صيغة :*w*)
- $P \wedge Q \mid= P$

مثال: لنفرض مثلاً، في مثال الربوط السابق، لدينا *BAT_OK* صحيحة (البطارية مشحونة) و $\neg MOVES$ صحيحة (الذراع لا تتحرك). وأننا نمثّل بعض معرفتنا حول العالم بالصيغة:

$$BAT\_OK \wedge LIFTABLE \Rightarrow MOVES$$

وبهذا فإن لدينا ثلاث صيغ: اثنتان منها تصفان حالة معينة للعالم والثالثة تصف معرفة عامة حول العالم.

يُمكن استخدام جدول الحقيقة لإظهار أن $\neg LIFTABLE$ تتبع منطقياً لهذه الصيغ الثلاث.

BAT_OK	LIFTABLE	MOVES	¬MOVES	BAT_OK ∧ LIFTABLE ⇒ MOVES	¬LIFTABLE
F	F	F	T	T	T
F	F	T	F	T	T
F	T	F	T	T	F
F	T	T	F	T	F
T	F	F	T	T	T
T	F	T	F	T	T
T	T	F	T	F	F
T	T	T	F	T	F

بما أن، باستخدام الاستتباع المنطقي، للصيغة *¬LIFTABLE* القيمة *True* في جميع التفسيرات التي تكون فيها هذه الصيغ الثلاث صحيحة. فإنها يجب بالتأكيد أن تأخذ القيمة *True* في تفسيرنا المعتمد للعالم.

وبهذا فإن الفرضية (والتي هي جزء من تفسيرنا المعتمد) "الكتلة غير قابلة للحمل" يجب أن تكون صحيحة.

إن استبدال طرائق جدول الحقيقة بطرائق البرهان يعطي كلفة حسابية أقل بكثير في حساب الفرضيات وحساب الإسناديات.

1 - 9 - قاعدة جديدة للاستدلال: الحل Resolution

تُعطى قاعدة الحل في حساب الفرضيات كما يلي:

لتكن Σ_1 و Σ_2 مجموعتين من الحرفيات (عبارتين) و λ ذرة.

من: $\{\lambda\}\cup\Sigma_1$ و $\{\neg\lambda\}\cup\Sigma_2$ يمكن أن نستنتج: $\Sigma_1 \cup \Sigma_2$

والتي تدعى بناتج الحل *Resolvent* (أو اختصاراً الحل) للعبارتين Σ_1

و Σ_2.

كما تدعى الذرة λ بالمحُل، والإجراء بالحل.

مثال: يؤدي حل $R\vee P$ و $\neg P\vee Q$ إلى $R\vee Q$

يُمكن أن نكتب العبارات التي تمّ حلها كاقتضاءات $\neg R\Rightarrow P$ و $P\Rightarrow Q$.

يؤدي تطبيق قاعدة الاستدلال والمدعوة بالتسلسل *Chaining* على الاقتضائين السابقين إلى $\neg R\Rightarrow Q$ والتي تُكافئ $R\vee Q$. سنرى لاحقاً أن التسلسل هو حالة خاصة من الحل.

مثال: يؤدي حل R و $\neg R\vee P$ إلى P

وبما أن العبارة الثانية تُكافئ $R\Rightarrow P$ فنلاحظ أن مودس بوننس هي أيضاً حالة خاصة من الحل.

مثال: يؤدي حل $P\vee Q\vee R\vee S$ مع: $\neg P\vee Q\vee W$ على P إلى $Q\vee R\vee S\vee W$

لاحظ أن Q تظهر مرة واحدة في عبارة الحل (والتي هي مجموعة).

مثال: يؤدي حل $P\vee Q\vee\neg R$ مع $P\vee W\vee\neg Q\vee R$ على Q إلى $P\vee\neg R\vee R\vee W$

ويؤدي حلهما على R إلى $P\vee Q\vee\neg Q\vee W$

في هذه الحالة. وبما أن لكل من $\neg R \vee R$ و $Q \vee \neg Q$ القيمة *True*، فإن قيمة كل من الحلين *True*.

لاحظ أنه في هذا المثال يجب الحل إما على Q وإما على R وليس على كليهما بالوقت نفسه. أي أن $P \vee W$ ليس حلاً للفقرتين.

يؤدي حل λ (حرفياً موجباً) مع $\neg\lambda$ (حرفياً سالباً) إلى العبارة الخالية. أي من λ و $\neg\lambda$ يمكن استنتاج F لأن λ و $\neg\lambda$ متناقضتان.

تكون أي مجموعة من الصيغ *wffs* و التي تحوي λ و $\neg\lambda$ مجموعة غير قابلة للتحقيق *Unsatisfiable*.

وبالمقابل فإن أي عبارة تحوي ذرة ونفيها (مثل λ و $\neg\lambda$) يكون لها القيمة *True* بغض النظر عن قيمة λ.

1 - 10 - تحويل الصيغ *wffs* إلى عطف عبارات

يمكن تحويل أي صيغة في حساب الفرضيات إلى عطف عبارات مكافئ.

ونقول عن الصيغة المكتوبة كعطف عبارات إنها في شكل العطف النظامي: *Conjunctive Normal Form(CNF)*

نستعرض عبر المثال التالي خطوات تحويل صيغة إلى الشكل *CNF*. لتكن الصيغة المطلوب تحويلها:

$$\neg(P \Rightarrow Q) \vee (R \Rightarrow P)$$

1. حذف إشارات الاقتضاء وذلك باستخدام الشكل المكافئ مع $\vee$:

$$\neg(\neg P \vee Q) \vee (\neg R \vee P)$$

2. خفض مجال النفي ¬ باستخدام قوانين دومرغان وبحذف إشارتي النفي المتتاليتين ¬¬:

$$(P \wedge \neg Q) \vee (\neg R \vee P)$$

3. استخدام قوانين التجميع والتوزيع للوصول إلى الشكل CNF:

$$(P \vee \neg R \vee P) \wedge (\neg Q \vee \neg R \vee P)$$

ثم:

$$(P \vee \neg R) \wedge (\neg Q \vee \neg R \vee P)$$

نقوم عادةً بالتعبير عن عطف عبارات (الشكل النظامي) بمجموعة من العبارات (العطف ضمني بين العبارات):

$$\{(P \vee \neg R), (\neg Q \vee \neg R \vee P)\}$$

1 - 11 - الحـــــل بالنقض Resfutation Resolution

يتمّ الحل بالنقض لإثبات صحة صيغة *wff* من مجموعة من الصيغ Δ باتباع الخطوات التالية:

1- تحويل كل صيغة موجودة في Δ إلى عطف عبارات (الشكل النظامي).

2- تحويل نفي الصيغة *w* والمطلوب إثباتها إلى عطف عبارات.

3- تجميع العبارات الناتجة عن الخطوتين السابقتين في مجموعة واحدة *G.*

4- نعاود تطبيق الحل على عبارات المجموعة Γ ونضيف ناتج الحل في كل مرة إلى المجموعة حتى الوصول إلى حالة لا يبقى فيها عبارات قابلة للحل أو الوصول إلى المجموعة الخالية.

مثال: عالم رفع الكتل

لتكن مجموعة العبارات Δ:

1. *BAT_OK*
2. *¬MOVES*
3. *BAT_OK ∧ LIFTABLE ⇒ MOVES*

يُعطي تحويل العبارة الثالثة إلى الشكل النظامي:

4. *¬BAT_OK ∨ ¬LIFTABLE ∨ MOVES*

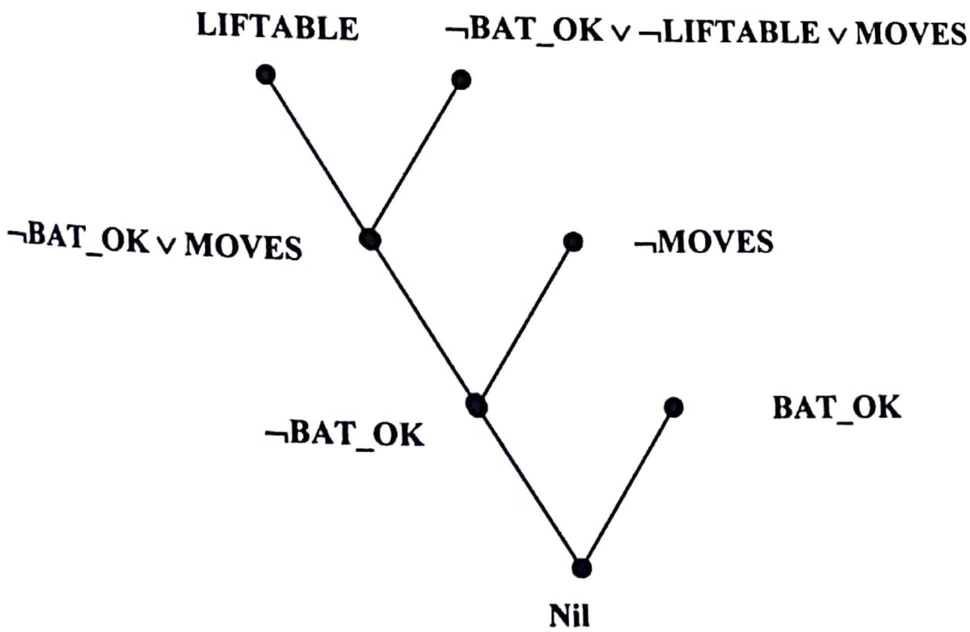
يُعطي نفي الصيغة المطلوب برهانها:

5. *LIFTABLE*

لنطبق الآن الحل للحصول على العبارات التالية:

6. *¬BAT_OK ∨ MOVES (4,5 بحل)*
7. *¬BAT_OK (6,2 بحل)*
8. *Nil (7,1 بحل)*

يُمكن تمثيل الحل بالشجرة التالية:



1 - 12 - أسئلة متعددة الخيارات

1. لتكن لدينا مجموعة القواعد التالية:

$P \Rightarrow (R \vee S)$

$\neg P \Rightarrow (R \vee S)$

$\neg S$

$(R \vee U) \Rightarrow Q$

يُمكن مما سبق برهان:

a. Q

b. $\neg Q$

c. S

d. لا يوجد خيار صحيح

2. حدّد الصيغة غير القابلة للتحقيق *Unsatisfiable* فيما يلي:

a. $((P \wedge Q) \Rightarrow (P \Rightarrow Q))$

b. $((P \Rightarrow Q) \vee (P \wedge \neg Q))$

c. $\neg(\neg (P \wedge Q) \Rightarrow (R \Rightarrow (\neg R \Rightarrow Q)))$

d. <u>لا يوجد خيار صحيح</u>

3. لتكن لدينا مجموعة الحقائق التالية:

1. *"chose one of three roads: short, medium or long"*
2. *"the short road is always crowded"*
3. *"the medium road is not comfortable, but fast"*
4. *"the long road is comfortable"*
5. *"the chosen road should be comfortable."*

والتي يُمكن كتابتها كما يلي:

1. $short \vee medium \vee long$
2. $short \Rightarrow crowded$
3. $medium \Rightarrow (\neg comfortable \wedge fast)$
4. $long \Rightarrow comfortable$
5. $comfortable.$

يُمكن باستخدام الحقائق الخمسة السابقة برهان أن الطريق المختار هو الطريق الطويل *long*

a. ***<u>false</u>***

b. *true*

c. حسب طول الطريق

d. حسب سرعة السيارة

4. الشكل النظامي للصيغة التالية:

$$\neg\{[Smoke \Rightarrow Fire] \Rightarrow [(Smoke \wedge Heat) \Rightarrow Fire]\}$$

a. *true*

b. *false*

c. *Smoke Ú Fire*

d. غير ذلك

5. الشكل النظامي للصيغة التالية:

$$(\neg q \wedge (p \Rightarrow q)) \Leftrightarrow \neg p$$

a. <u>True</u>

b. *False*

c. p

d. <u>غير ذلك</u>

6. الشكل النظامي للصيغة التالية:

$$[(\neg P \vee \neg Q) \wedge (P \vee \neg Q) \wedge R] \Rightarrow [\neg Q \wedge R]$$

a. <u>True</u>

b. *False*

c. R

d. $P \vee R$

7. ليكن لدينا الصيغ الثلاث التالية:

$B \Rightarrow (C \wedge \neg D)$

$(C \wedge A) \Rightarrow \neg E$

$F \Rightarrow (D \vee E)$

ما هي الصيغة التي يُمكن برهانها من الصيغ الثلاث السابقة:

a. **$\underline{A \Rightarrow (B \Rightarrow \neg F)}$**

b. $(B \Rightarrow \neg F) \Rightarrow A$

c. $\neg A \Rightarrow (B \Rightarrow \neg F)$

d. ولا خيار

الفصل الثاني

حساب الإسناديات

The Predicate Calculus

2 - 1 - أهمية حساب الإسناديات

- لا يوفر حساب الفرضيات لغة غنية لتوصيف العالم.
- مثلاً: لا يمكن التعبير في حساب الفرضيات عن أن كل طلاب الصف العاشر نجحوا. أو أن بعض طلاب الصف العاشر نجحوا.

2 - 2 - الشكل *Syntax* - مكونات اللغة

- مجموعة غير منتهية من الثوابت الغرضية *Object Constants*

Aa, 125, 13B, Q, John, EiffelTower

(تبدأ الثوابت الغرضية بحرف كبير).

- مجموعة غير منتهية من الثوابت الوظيفية *Function Constants*

distanceBetween(x,y), times(x,y)

(تبدأ الثوابت الوظيفية بحرف صغير، كما نستخدم الأحرف الصغيرة للمتغيرات).

- مجموعة من الثوابت العلاقاتية (الإسناديات) *Relation Constants*

Parent(x,y), Large(x), Clear(x)

Scanned by CamScanner

- نستخدم أيضاً جميع الروابط $\vee, \wedge, \neg, \Rightarrow$ والمحددات $[.], (.)$ والفاصلة ,.

الحدود *Terms*

- كل ثابت غرضي هو حد.
- الثابت الوظيفي من الدرجة *n* متبوعاً بـ *n* حداً موضوعة بين قوسين ومفصولة بفواصل. هو أيضاً حد.

أمثلة:

fatherOf)John)

times(4, plus(3,6))

الصيغ جيدة التركيب *wffs*

الذرات:

- تتألف من ثابت علاقاتي من البعد *n* متبوعاً بـ *n* حداً موضوعة بين قوسين ومفصولة بفواصل.
  - يُمكن حذف الأقواس في حال كون بعد الثابت العلاقاتي *0*.
  - كل ذرة *Atom* هي صيغة جيدة التركيب *wff*

أمثلة:

Greatherthan(7,2), P(A,B,C,D), Q

الصيغ الفرضية جيدة التركيب WFF

يُدعى كل تعبير يتشكل من صيغ حساب الإسناديات بالطريقة نفسها التي تُشكل بها صيغ حساب الفرضيات، بصيغة فرضية جيدة التركيب.

مثال:

$$[Greatherthan(7,2) \wedge Lessthan(15,4)] \vee \neg Brother(John,Sam) \vee P$$

2 - 3 - الدلالة Semantic

- يحقق تفسير صيغة *wff* إذا كان للصيغة القيمة *True* في هذا التفسير.
- نقول عن تفسير يحقق صيغة إنه نموذج لها.
- نقول عن صيغة تأخذ القيمة *True* من أجل كل التفاسير إنها صالحة *Valid*.
- نقول عن صيغة لا يوجد لها أي نموذج إنها غير قابلة للتحقيق *Unsatisfiable*.
- إذا كان لصيغة *w* القيمة *True* من أجل كل التفاسير التي تجعل لكل صيغة في مجموعة Δ القيمة *True* فنقول إن Δ تؤدي منطقياً لـ *w*

$$D|=w$$

- تكون صيغتان متكافئتين إذ أخذتا قيم الحقيقة نفسها من أجل جميع التفسيرات. (أي إذا وفقط إذا كانت كل منهما تؤدي للأخرى).

2 - 4 - التكميم Quantification

لدينا المكمم العام $\forall$ *universal quantifier* والمكمم الوجودي $\exists$ *existential quantifier*

إذا كانت w صيغة و ξ متغيراً، فإن كلاً من:

$$(\forall \xi)w$$

$$(\exists \xi)w$$

هي صيغ *wff*

أمثلة

$$(\forall x)\ [P(x) \Rightarrow R(x)]$$

$$(\exists x)\ [P(x) \Rightarrow (\exists y)[R(x,y) \Rightarrow S(f(x))]]$$

بعض التكافؤات المفيدة

$$\neg\ (\forall \xi)w(\xi) \equiv (\exists \xi)\neg w(\xi)$$

$$\neg\ (\exists \xi)w(\xi) \equiv (\forall \xi)\neg w(\xi)$$

$$(\forall \xi)w(\xi) \equiv (\forall \eta)w(\eta)$$

الحل *Resolution*

إذا حوت عبارتان حرفيين متطابقين ومتتامين (أحدهما نفي الآخر)، فيُمكن أن نحلهما بطريقة حساب الفرضيات نفسها.

إذا كان لدينا حرف λ (ξ) في عبارة (حيث ξ متغير) ، وحرفي متمم $\neg\lambda$ (τ) في عبارة أخرى (حيث τ هو حد لا يحوي المتغير ξ) . فيُمكن استبدال ξ بـ τ في العبارة الأولى، ومن ثم القيام بالحل للوصول لناتج حل *Resolvent* العبارتين.

مثال:

ليكن لدينا مثلاً العبارتان:

$$P(f(y),A) \vee Q(B,C)$$

و

$$\neg P(x,A) \vee R(x,C) \vee S(A,B)$$

نلاحظ أن العنصرين في أول كل عبارة متتامان وبالتالي يُمكن الحل على الحرفي $P(f(y),A)$ لنحصل على ناتج الحل:

$$R(f(y),C) \vee S(A,B) \vee Q(B,C)$$

يتمّ حساب الاستبدالات المناسبة باستخدام إجرائية تُدعى التوحيد *Unification*.

لتوصيف هذه الإجرائية سنقوم أولاً بمناقشة مسألة الاستبدال *Substitution*.

2 - 5 - الاستبدال *Substitution*

نحصل على مُنتَسَخ الاستبدال *substitution instance* لتعبير باستبدال متغيرات التعبير بحدود.

مثال:

للتعبير $P[x,f(y),B]$ منتسخات الاستبدال التالية:

- $P[z,f(w),B]$
- $P[x,f(A),B]$
- $P[g(z),f(A),B]$
- $P[C,f(A),B]$

يُدعى أول مُنتَسَخ بمغاير أبجدي *alphabetic variant* لأننا قمنا بشكل أساسي باستبدال متغيرات التعبير بمتغيرات أخرى.

يُدعى آخر مُنتَسَخ بالمنتسخ الأرضي *ground instance* إذ لا تحوي حدوده أي متغير.

يُمكن تمثيل أي استبدال بمجموعة مرتبة من الثنائيات:

$$s=\{\tau_1/\xi_1,\ \tau_2/\xi_2,\dots,\ \tau_n/\xi_n\}$$

حيث τ_i/ξ_i يعني أن كل ظهور للمتغير ξ_i سيُستبدل بالحد τ_i.

لاحظ أنه لا يُمكن استبدال متغير بحد يحوي هذا المتغير.

في المثال السابق قمنا باستخدام الاستبدالات التالية:

$$s1=\{z/x,\ w/y\}$$

$$s2=\{A/y\}$$

$$s3=\{g(z)/x,\ A/y\}$$

$$s4=\{C/x,\ A/y\}$$

لتدوين مُنتَسَخ استبدال لتعبير w باستخدام استبدال s نكتب ws.

وبهذا فإن:

$$P[z,f(w),B]=P[x,f(y),B]s1$$

عندما يُطبق الاستبدال s على كل عنصر في مجموعة من التعابير $\{wi\}$ ندون مجموعة منتسخات الاستبدال بـ

$$\{wi\}s$$

نقول إن مجموعة *{wi}* من التعابير قابلة للتوحيد *unifiable* إذا وجد استبدال *s* بحيث يكون:

$$w1s=w2s=w3s=...$$

نقول في هذه الحالة إن *s* هو الموحد *unifier* لـ *{wi}*

مثال:

$$s=\{A/x,B/y\}$$

يوحد المجموعة: *{[P[x,f(y),B], P[x,f(B),B}* ليُعطي: *{[P[A,f(B),B}*

6 - 2 - خوارزمية التوحيد *Algorithm Unification*

نستخدم الخوارزمية *UNIFY* والتي تعمل على تعابير موضوعة بشكل قائمة مهيكلة.

فمثلاً التعبير: *(¬P(x,f(A,y)* يُكتب بالشكل: *((¬P x (f A y)*

ويكون *P¬* التعبير الأول و *(f A x)* التعبير الثالث.

مجموعات عدم التوافق *Disagreement Set*

نستخدم في الخوارزمية *UNIFY* مفهوم مجموعات عدم التوافق *disagreement set*.

نحصل على مجموعة عدم التوافق لمجموعة غير خالية *W* من التعابير بتحديد موقع أول رمز (بدءاً من اليسار) لا تكون عنده رموز كل التعابير متساوية.

ومن ثم نستخرج من كل تعبير التعابير الجزئية التي تبدأ بالرمز الذي يحتل هذا الموقع.

تكوّن هذه التعابير الجزئية مجموعة عدم التوافق لـ W.

مثلاً إذا كانت:

$$W=\{(\neg P\ x\ (f\ A\ y)),(\neg P\ x\ (f\ z\ B))\}$$

فإن مجموعة عدم التوافق لها هي:

$$\{A,z\}$$

خوارزمية التوحيد

$UNIFY(\Gamma)$

(حيث Γ مجموعة تعابير لها بنية القائمة)

1. $k \leftarrow 0;\ G_k \leftarrow \Gamma; \alpha_k \leftarrow e$

(الخطوة الابتدائية: ε هو الاستبدال الفارغ)

2. $If\ Singleton(G_k)\ Then\ Return(\alpha_k)$

(إذا كانت G_k أحادية نخرج من الإجراء و نُرجع α_k)

3. $D_k \leftarrow DisagreementSet(G_k)$

(نحسب D_k مجموعة عدم التوافق لـ G_k)

4. $If\ Exits(v_k,t_k)\ in\ D_k\ and$

$Variable(v_k)\ and\ Not\ Occur(v_k,t_k)\ Then$

(إذا وجد v_k و t_k في المجموعة D_k حيث v_k متغير لا يظهر في t_k)

Continue (نستمر)

Else (و إلا فإن Γ غير قابلة للتوحيد. ونخرج)

Return (FAILURE)

أمثلة:

منتسخ الاستبدال المشترك الأعم	مجموعات من التعابير
$P(A)$	$\{P(x),P(A)\}$
$P[f(x),x,g(x)]$	$\{P[f(x),y,g(y)],P[f(x),z,g(x)]\}$
$P[f(x,g(A,y)),g(A,y)]$	$\{P[f(x,g(A,y)),g(A,y)],P[f(x,z),z]\}$

الحل *Resolution*

لنفرض أن لدينا العبارتين γ_1 و γ_2 (كل عبارة *Clause* ممثلة بمجموعة من الحرفيات *Literals*)

إذا وجد ذرة *atom* Φ في γ_1 وحرفي *literal* $\neg\Psi$ في γ_2 بحيث أن لـ Φ و Ψ الموحد الأعم μ ، فإن لهاتين العبارتين ناتج حل *resolvent* ρ نحصل عليه بتطبيق الاستبدال على اجتماع γ_1 و γ_2 بعد حذف العنصرين المتتامين Φ و $\neg\Psi$

أي:

$$\rho = [(\gamma_1 - \{\Phi\}) \cup (\gamma_2 - \{\neg\Psi\})]\mu$$

نقوم قبل الحل بإعادة تسمية المتغيرات في العبارات بحيث لا يظهر أي متغير لعبارة في عبارة أخرى.

مثال:

إذا كان المطلوب حل:

$$P(x) \vee Q(f(x))$$

مع:

$$R(g(x)) \vee \neg Q(f(A))$$

فإننا نقوم أولاً بإعادة كتابة العبارة الثانية لنحصل على:

$$R(g(y)) \vee \neg Q(f(A))$$

تُدعى عملية إعادة تسمية المتغيرات بتقييس المتغيرات *Standardizing*

أمثلــــة

1- ينتج عن حل: $\{(P(x),Q(x,y)\}$ و $\{(\neg P(A),R(B,z)\}$

المجموعة:

$$\{Q(A,y),R(B,z)\}$$

2- يُمكن حل:

$$\{\neg P(A,z), \neg Q(B)\} \text{ و } \{P(x,x),Q(x),R(x)\}$$

بطريقتين مختلفتين لنحصل على:

$$\{Q(A),R(A), \neg Q(B)\}$$

و

$$\{P(B,B),R(B), \neg P(A,z)\}$$

2 - 7 - تحويل الصيغ إلى عبارات Clause Form

خطوات تحويل الصيغ إلى عبارات

يُمكن وضع أي صيغة *wff* بشكل عبارة *Clause Form* باتباع الخطوات التالية:

(1) **حذف إشارات الاقتضاء** (مثل حساب الفرضيات).

(2) **تقليص مجالات النفي** (مثل حساب الفرضيات).

(3) **تقييس المتغيرات.**

بما أن المتغيرات الواقعة ضمن مجال المكممات هي متغيرات شكلية *Dummy*، فيُمكن إعادة تسميتها بحيث أن لكل مكمم متغيراته الخاصة به.

مثلاً يُمكن كتابة الصيغة:

$$(\forall x)[\neg P(x) \lor (\exists x) Q(x)]$$

كـ

$$(\forall x)[\neg P(x) \lor (\exists y) Q(y)]$$

(4) **حذف المكممات الوجودية**

لاحظ أنه في:

$$(\forall x)[\, (\exists y) Height(x,y)]$$

يتواجد المكمم الوجودي ضمن مجال المكمم العام، وبهذا فإن *y* يُمكن أن يتعلق بقيمة *x*

مثلاً إذا كان المعنى الموافق للتعبير السابق هو:

«لكل شخص *x* الطول *y*"

فمن الواضح أن الطول يتعلق بالشخص. يُمكن إظهار هذا الارتباط بتعريف دالة غير معروفة $h(x)$، والتي تقوم بمقابلة كل قيمة لـ x بقيمة y الموجودة تُدعى مثل هذه الدالة بدالة سكوليم **Skolem function** نسبة إلى عالم المنطق *Thoralf Skolem*.

إذا استعملنا دالة سكوليم عوضاً عن المتغير y فيُمكن أن نحذف المكمم الوجودي لنحصل على:

$$(\forall x)[\, Height(x,h(x))]$$

وتكون القاعدة العامة لحذف المكممات الوجودية أن نستبدل كل متغير مكمم وجودياً بدالة سكوليم.

هذه الدالة هي المتغيرات المكممة *arguments* تكون محدّدات باستخدام $\forall$ والتي يحوي مجالها المتغير المراد حذفه.

يجب أن تكون أسماء دوال سكوليم المستخدمة جديدة وغير مستخدمة في الصيغ.

مثال: يُمكن حذف $(\exists z)$ من:

$$[(\forall w Q(w)] \Rightarrow (\forall x)\{(\forall y)\{(\exists z)[P(x,y,z) \Rightarrow (\forall u) R(x,y,u,z)]\}\}$$

للحصول على:

$$[(\forall w Q(w)] \Rightarrow (\forall x)\{(\forall y)\, [P(x,y,g(x,y)) \Rightarrow (\forall u) R(x,y,u,g(x,y))]\}$$

مثال: يُمكن حذف $(\exists w)$ من:

$$(\forall x)\{\neg P(x) \vee \{(\forall y)[\neg P(y) \vee P(f(x,y))] \wedge (\exists w)[Q(x,w) \wedge \neg P(w)]\}\}$$

لنحصل على:

$$(\forall x)\{\neg P(x) \vee \{(\forall y)[\neg P(y) \vee P(f(x,y))] \wedge [Q(x,h(x)) \wedge \neg P(h(x)]\}\}$$

حيث $g(x,y)$ و $h(x)$ هي دوال سكوليم

إذا لم يكن المتغير المكمم وجودياً ضمن مجال أي مكمم عام، فإننا نستخدم دالة سكوليم بدون محددات أي ثابت

وبهذا فإن: $(\exists x)P(x)$ تصبح: $P(SK)$

حيث يرمز الثابت SK لمقدار نعرف أنه موجود ويجب أن يكون ثابتاً جديداً لا يُستخدم في الصيغ.

لحذف المتغيرات المكممة وجودياً في مجموعة من الصيغ نُطبّق الإجراء السابق على كل صيغة.

ندعو في النهاية مجموعة الصيغ الناتجة بعد ذلك بشكل سكوليم *Skolem form*.

(5) التحويل إلى الشكل الأمامي Prenex Form

نضع جميع المكممات العامة في بداية الصيغة ونجعل مجال كل مكمم يضم كامل الصيغة.

مثال:

الشكل الأمامي للصيغة السابقة:

$$(\forall x)\{\neg P(x)\vee\{(\forall y)[\neg P(y)\vee P(f(x,y))] \wedge [Q(x,h(x))\wedge\neg P(h(x)]\}\}$$

يكون:

$$(\forall x)\ (\forall y)\{\neg P(x)\vee\{\ [\neg P(y)\vee P(f(x,y))] \wedge [Q(x,h(x))\wedge\neg P(h(x)]\}\}$$

(6) وضع المصفوفة في شكل العطف النظامي Conjunctive Normal Form

استخدم الطريقة نفسها المتبعة في حساب الفرضيات لوضع المصفوفة بشكل العطف النظامي وذلك بتكرار تطبيق أحد قوانين التوزيع.

أي عملياً استبدال كل:

$(w1 \vee w2) \wedge (w1 \vee w3)$ بـ $w1 \vee (w2 \wedge w3)$

مثال: تُصبح الصيغة السابقة:

$$(\forall x)(\forall y)\{\neg P(x) \vee \{[\neg P(y) \vee P(f(x,y))] \wedge [Q(x,h(x)) \wedge \neg P(h(x)]\}\}$$

بعد وضعها في شكل العطف النظامي:

$$(\forall x)(\forall y)\{[\neg P(x) \vee \neg P(y) \vee P(f(x,y))] \wedge$$

$$[\neg P(x) \vee Q(x,h(x))] \wedge$$

$$[\neg P(x) \vee \neg P(h(x))]\}$$

(7) حذف المكممات العامة

يُمكن الآن حذف كل المكممات العامة واصطلاح أن كل متغيرات المصفوفة مكممة تكميماً عاماً. وبهذا يتبقى لدينا الآن مصفوفة في شكل العطف النظامي.

(8) حذف إشارات العطف $\wedge$

يُمكن الآن حذف $\wedge$ وذلك باستبدال التعابير من الشكل $(w1 \wedge w2)$ بمجموعة الصيغ جيدة التركيب $\{w1,w2\}$.

نحصل إذاً بعد عدة استبدالات على مجموعة منتهية من الصيغ والتي كل منها فصل بين حرفيات. ندعو هذا الشكل بالعبارة *Clause*.

يُمكن تحويل المثال السابق إلى مجموعة العبارات:

$$\neg P(x) \vee \neg P(y) \vee P[f(x,y)]$$

$$\neg P(x) \vee Q[x,h(x)]$$

$$\neg P(x) \vee \neg P[h(x)]$$

(9) إعادة تسمية المتغيرات

يُمكن إعادة تسمية المتغيرات بحيث لا يظهر أي متغير في أكثر من عبارة.

$\neg P(x1) \vee \neg P(y) \vee P[f(x1,y)]$

$\neg P(x2) \vee Q[x2,h(x2)]$

$\neg P(x3) \vee \neg P[h(x3)]$

مثال: مسألة الربوط موزع الطرود

لنفرض أن هذا الربوط يعرف أن جميع الطرود الموجودة في الغرفة 27 أصغر من أي طرد موجود في الغرفة 28. أي:

1. $(\forall x,y)\{[Package(x) \wedge Package(y) \wedge Inroom(x,27) \wedge Inroom(y,28)] \supset Smaller(x,y)\}$

لنختصر أسماء الثوابت لنجعل الصيغة أكثر صغراً ولنحولها إلى شكل العبارة مما يُعطي:

2. $\neg P(x) \vee \neg P(y) \vee \neg I(x,27) \vee \neg I(y,28) \vee S(x,y)$

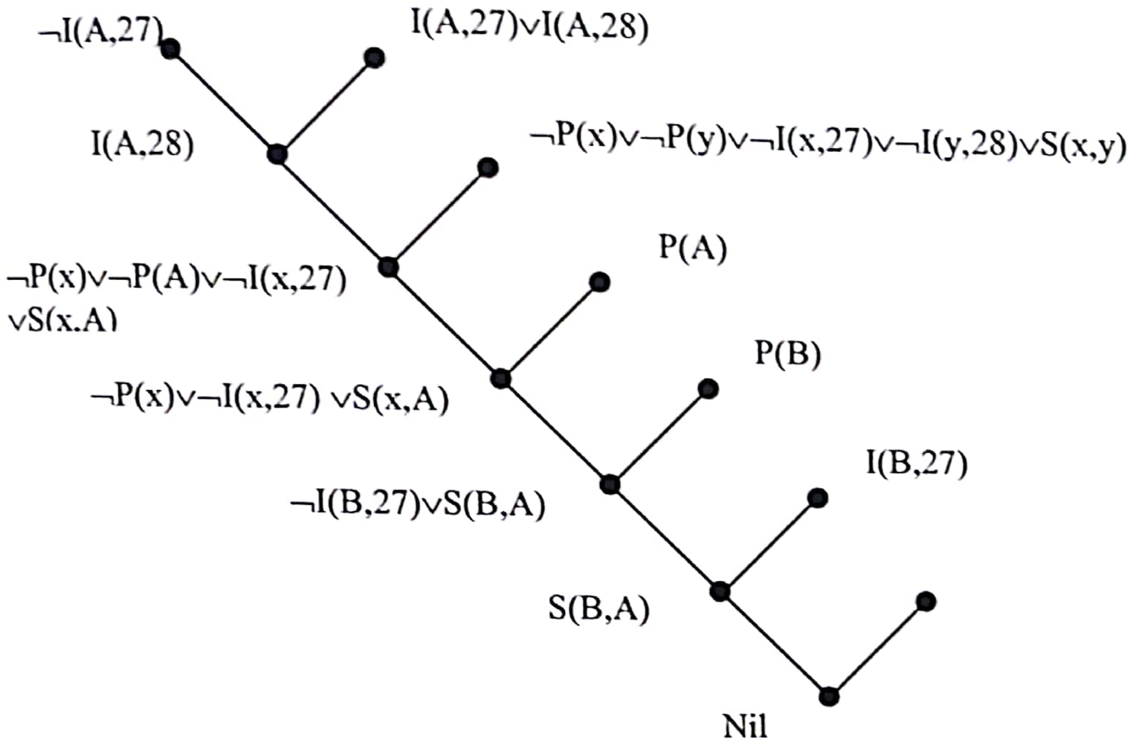
لنفرض أن الربوط يعرف أن الطرد A موجود إما في الغرفة 27 وإما في الغرفة 28 إلا أنه لا يعرف في أي منهما.

كما أن الربوط يعرف أن الطرد B موجود في الغرفة 27 كما أنه ليس أصغر من الطرد A.

3. $P(A)$
4. $P(B)$
5. $I(A,27) \vee I(A,28)$
6. $I(B,27)$
7. $\neg S(B,A)$

يُمكن للربوط، باستخدام الحل بالنقض، إثبات أن الطرد A موجود في الغرفة 27.

يُبين الشكل التالي شجرة البرهان.



إيجاد الحل *Answer Extraction*

مثال: مسألة الربوط موزع الطرود

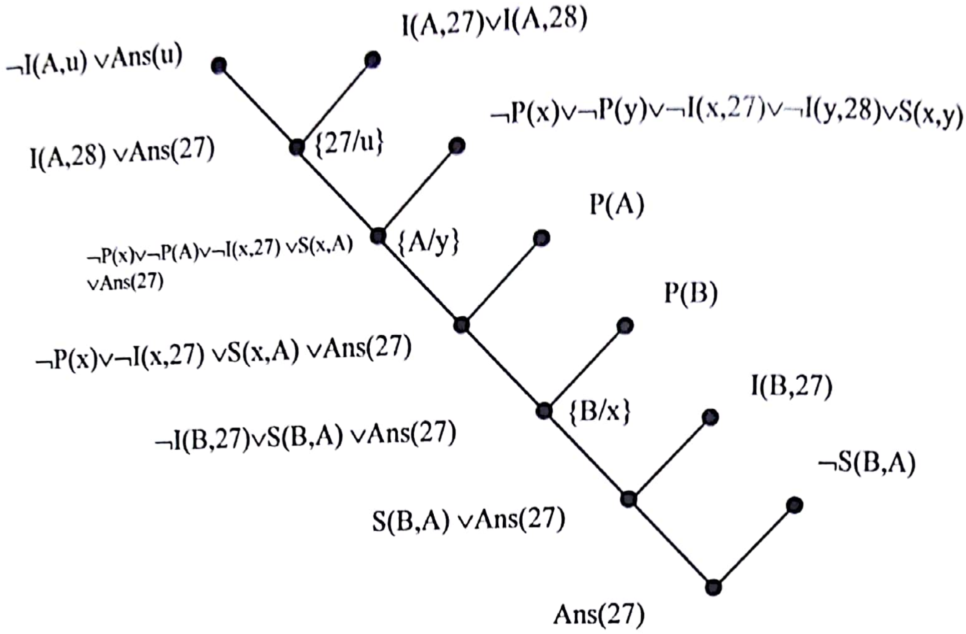
«في أي غرفة توجد الكتلة A"

تُبين الشجرة التالية كيفية استخدام الحرفي *Ans* لبرهان الصيغة:

$$(\exists u)I(A,u)$$

والتي يُمكن عدّها التعبير عن سؤال الربوط لنفسه:

«في أي غرفة توجد الكتلة A".



2 - 8 - تمثيل المعارف باستخدام المنطق من الدرجة الأولى

أمثلة

- *All crows are black*

$$\forall x\ Crow(x) \Rightarrow Black(x)$$

- *Mary likes the color of one of John's ties*

$$\exists x\ Like(Mary, color(x)) \wedge Tie(x) \wedge Owner(x,John)$$

- *Every gardener likes the sun*

$$(\forall x)\ gardener(x) \Rightarrow likes(x,Sun)$$

- *You can fool some of the people all of the time*

$$(\exists x)(\forall t)\ (person(x) \wedge time(t)) \Rightarrow$$

$$can\text{-}fool(x,t)$$

- *You can fool all of the people some of the time*

$$(\forall x)(\exists t)\ (person(x) \wedge time(t) \Rightarrow can\text{-}fool(x,t))$$

- *All purple mushrooms are poisonous*

$$(\forall x)\ (mushroom(x) \wedge purple(x)) \Rightarrow poisonous(x)$$

- *No purple mushroom is poisonous*

$$\neg(\exists x)\ purple(x) \wedge mushroom(x) \wedge poisonous(x)$$

or, equivalently,

$$(\forall x)\ (mushroom(x) \wedge purple(x)) \Rightarrow \sim poisonous(x)$$

- *There are exactly two purple mushrooms*

$$(\exists x)(\exists y)\ mushroom(x) \wedge purple(x) \wedge mushroom(y) \wedge purple(y) \wedge \neg(x=y) \wedge (\forall z)\ (mushroom(z) \wedge purple(z)) \Rightarrow ((x=z) \vee (y=z))$$

- *X is above Y if X is on directly on top of Y or else there is a pile of one or more other objects directly on top of one another starting with X and ending with Y*

$$(\forall x)(\forall y)\ above(x,y)\ \hat{U}\ (on(x,y) \vee (\exists z)\ (on(x,z) \wedge above(z,y)))$$

2 - 9 - أسئلة متعددة الخيارات

1 - ليكن لدينا:

(1) $P(x,y,y)$, $P(f(C),C,v)$

(2) $P(x, f(x))$, $P(x,y)$, $P(g(u),u)$

(3) $p(a,X)$, $p(Y, f(Y))$

(4) $parents(x,father(x),mother(Bill)),parents(Bill,father(y),z)$

حدّد العبارة الخاطئة فيما يلي:

a. الفقرتين في (1) قابلة للتوحيد

b. <u>الفقرات الثلاث في (2) قابلة للتوحيد</u>

c. الفقرتين في (3) قابلة للتوحيد

d. الفقرتين في (4) قابلة للتوحيد

2 - لتكن لدينا مجموعة القواعد التالية:

$\forall x, y, z\ S(x,z) \wedge S(y,z) \Rightarrow U(x,y) \wedge U(y,x)$

$\forall x, y\ U(x, y) \Rightarrow F(x, y)$

$\forall x, y, z\ F(x, y) \wedge F(y, z) \Rightarrow F(x, z)$

ولتكن لدينا الحقائق التالية:

$S(A, F)$

$S(B, F)$

$S(B, G)$

$S(C, G)$

ولتكن لدينا الأهداف الثلاثة التالية:

$G1: F(A, B)$

$G2: F(A, C)$

$G3: F(B, C)$

حدّد فيما يلي الصيغة التي لا يُمكن برهانها من مجموعة القواعد والحقائق السابقة:

a. $G1 \wedge G2 \wedge G3$

b. $\underline{G1 \wedge \neg G2 \wedge G3}$

c. $G1 \vee \neg G2 \vee G3$

d. $G1 \vee G2 \vee G3$

3 - لتكن لدينا الصيغة التالية:

$$(\forall x \exists y\, P(x) \Rightarrow (Q(x) \wedge M(y, x))) \vee \neg (\exists x\, R(x) \wedge S(x))$$

عند وضع الصيغة بالشكل النظامي ينتج ثوابت سكوليم من الشكل:

a. $\underline{f(x)}$

b. C

c. $f(x,y)$

d. ولاخيار مما سبق

4 - لتكن لدينا مجموعة القواعد التالية:

$$\forall x\, \forall y\, \forall z\, (R(x,y) \wedge R(x,z)) \Rightarrow R(y,z)$$

$$\forall x\, \forall y\, R(x,y) \Rightarrow \exists u\, (R(u,x) \wedge R(u,y))$$

يُمكن مما سبق برهان:

$$\forall x\, \forall y\, R(x,y) \Rightarrow R(y,x)$$

a. **<u>صح</u>**

b. خطأ

c. بعض الأحيان

d. حسب قيمة ثوابت سكوليم

5 - حدّد فيما يلي الصيغة الصالحة *valid*

a. $(\forall x, y\, (p(x,y) \Rightarrow p(y,x))) \Rightarrow \forall z\, p(z,z)$

b. $\forall x, y\, (p(x,y) \Rightarrow (p(y,x) \Rightarrow \forall z\, p(z,z)))$

c. $(\forall x\, p(x)) \Rightarrow \exists y\, p(y)$

d. **<u>ولا خيار</u>**

6 - ليكن لدينا الصيغ الثلاث التالية:

$P(x) \vee Q(F(x), x)$

$R(y) \vee Q(y, z)$

$R(F(A))$

ما هي الصيغة التي لا يُمكن برهانها من الصيغ الثلاث السابقة:

a. $P(A)$

b. $\exists x\, P(x)$

c. $\forall x\, P(x)$

d. **<u>ولا خيار مما سبق</u>**

7 - ليكن لدينا القواعد التالية:

$Guilty(b) \Rightarrow \neg Guilty(a)$

$Guilty(b) \Rightarrow \neg Guilty(c)$

$Guilty(a) \Rightarrow \neg Guilty(c)$

$Knows(a, d) \Rightarrow Lies(b)$

$\neg Knows(a, d) \Rightarrow Lies(a)$

$Knows(a, d) \Rightarrow Lies(c)$

$Lies(x) \Rightarrow Guilty(x)$

يُمكن من هذه القواعد برهان:

a. **$Guilty(a)$**

b. $Guilty(b)$

c. $Guilty(c)$

d. $Guilty(d)$

8 - أي ما يلي هو نمذجة للحقيقة التالية:

John has exactly one brother

a. $\exists x, y\ brother(John, x) \wedge brother(John, y) \wedge x = y$

b. $\exists x\ brother(John, x) \Rightarrow \forall y(brother(John, y) \Rightarrow x = y)$

c. $\exists x\ brother(John, x) \Rightarrow \forall y(brother(John, y) \wedge x = y)$

d. **غير ذلك**

9 - لتكن لدينا مجموعة القواعد التالية:

$$\forall x, y, z\ SpeakLang(x,z) \wedge SpeakLang(y,z) \Rightarrow Understand(x,y) \wedge Understand(y,x)$$

$$\forall x, y\ Understand(x, y) \Rightarrow Friend(x, y)$$

$$\forall x, y, z\ Friend(x, y) \wedge Friend(y, z) \Rightarrow Friend(x, z)$$

ولتكن لدينا الحقائق التالية:

SpeakLang(Ann, French)

SpeakLang(Bob, French)

SpeakLang(Bob, German)

SpeakLang(Cal, German)

ولتكن لدينا الأهداف الثلاث التالية:

G1: Friend(Ann, Bob)

G2: Friend(Ann, Cal)

G3: Friend(Bob, Cal)

حدّد فيما يلي الصيغة التي لا يُمكن برهانها من مجموعة القواعد والحقائق السابقة:

a. $G1 \wedge G2 \wedge G3$

b. $\underline{G1 \wedge \neg G2 \wedge G3}$

c. $G1 \vee \neg G2 \vee G3$

d. $G1 \vee G2 \vee G3$

10 - لتكن لدينا مجموعة القواعد التالية:

$$\forall x \forall y (p(x,y) \Rightarrow \exists z\, q(x,y,z))$$

$$\exists x \forall y \forall z (r(y,z) \Leftrightarrow q(x,y,z))$$

$$\forall x \exists y (\neg p(x,y) \Rightarrow \forall z\, q(x,y,z))$$

عند وضع القواعد السابقة بالشكل النظامي ينتج ثوابت سكوليم التالية:

a. <u>C, g(x), f(x, y)</u>

b. C1, C2, g(x)

c. C1, C2, f(x,y)

d. ولاخيار مما سبق

11 - لتكن لدينا مجموعة القواعد التالية:

$$\forall x \forall y (p(x,y) \Rightarrow \exists z\, q(x,y,z))$$

$$\exists x \forall y \forall z (r(y,z) \Leftrightarrow q(x,y,z))$$

$$\forall x \exists y (\neg p(x,y) \Rightarrow \forall z\, q(x,y,z))$$

يُمكن مما سبق برهان:

$$\exists w \exists x \exists y \exists z\,)\, r(x,y) \wedge q(x,w,z))$$

a. <u>صح</u>

b. خطأ

c. بعض الأحيان

d. حسب قيمة ثوابت سكوليم

12 - حدّد عدد الحالات غير قابلة للتوحيد من الثنائيات الأربعة التالية:

p(g(y), x, f(g(y)) and p(z, h(z,w), f(w))

R(f(x), y) and R(z, g(w))

R(f(x), x) and R(y, g(y))

P[f(x),y,g(y)] and P[f(x),z,g(x)]

a. 0

b. 1

c. 2

d. 3

13 - ليكن لدينا المعارف التالية:

∀x Person(x) ^Wood(x) Þ Witch(x)

∀y Duck(y) Þ Wood(y)

∀x, y Duck(y) ^ Equals(weight(x),weight(y)) Þ Wood(x)

Equals(weight(A),weight(D))

Person(A)

Duck(D)

يُمكن من هذه المعارف برهان:

a. Witch(A)

b. ¬Witch(A)

c. Witch(D)

d. ولا خيار

الفصل الثالث

النظم الخبيرة

Expert Systems

- تُعتبر النظم الخبيرة من أكثر تطبيقات تقانات الحاكمة في الذكاء الصنعي، والتي تستخدم الحقائق والقواعد لتضمين المعارف حول حقل معين من حقول المعارف البشرية مثل الطب والهندسة والأعمال.

- تُصمم النظم الخبيرة عموماً لحل مسائل التصنيف واتخاذ القرارات مثل (التشخيص الطبي، الوصفات العلاجية، تنظيم البورصات، وغيرها ...).

- النظم الخبيرة هي أدوات ذكاء صنعي، وهذا يعني أننا لا نستعملها إلا في المسائل التي ليس لها أي خوارزمية واضحة أكيدة لحلها.

- تتطلب النظم الخبيرة، وجود خبرة نودُّ نمذجتها. أي أنه، لا معنى للنظم الخبيرة إلا في المجالات التي توجد فيها خبرةٌ بشرية. والخبير هو الشخص الذي يعرف مجال التطبيق، ويعرف، نوعاً ما، كيف ينقل معرفته للآخرين.

التفكير البشري معقد جداً ومن الصعب تمثيله بخوارزمية. ومع ذلك فإن معظم الخبراء قادرين على وضع معارفهم لحل المسائل على شكل قواعد من الشكل (إلا أنهم غير قادرين على ضمان كمال هذه القواعد ولا تكاملها معاً!):

IF <antecedent> مقدمة *THEN <consequent>* النتيجة

مثال تعليمي:

يُمكن أن نتصور مثلاً موظف في بنك مسؤول عن إعطاء القروض للمتعاملين. يستخدم نظام خبير يُساعده في تقرير فيما إذا كان من المناسب إعطاء قرض لمتعامل. لنفرض أن الذرات التالية تستخدم المعاني الموافقة:

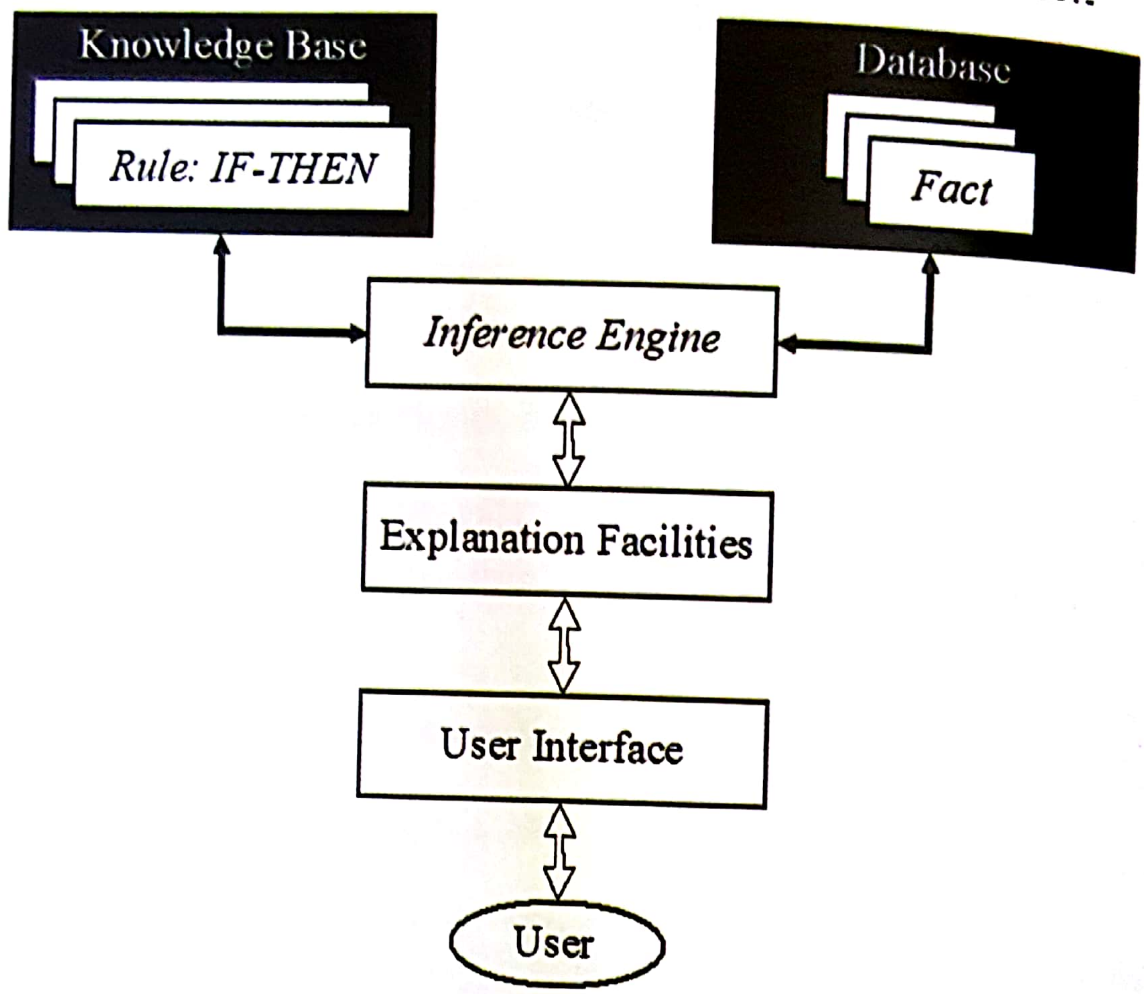
OK	يجب أن يتمّ الموافقة على القرض
COLLAT	ضمانة القرض مقنعة
PYMT	العميل قادر على تسديد دفعات القرض
REP	للعميل سمعة مالية جيدة
APP	تخمين الضمانة أكبر من مبلغ القرض
RATING	للعميل دفعات دورية منتظمة
INC	يتجاوز دخل العميل مصاريفه
BAL	للعميل صفحة متوازنة ممتازة

يُمكن استخدام القواعد التالية بهدف إقرار القرض:

1. $COLLAT \wedge PYMT \wedge REP \Rightarrow OK$
2. $APP \Rightarrow COLLAT$
3. $RATING \Rightarrow REP$
4. $INC \Rightarrow PYMT$
5. $BAL \wedge REP \Rightarrow OK$

3 - 1 - البنية العامة للنظام الخبير

يُبين الشكل التالي البنية العامة للنظام الخبير:



- **قاعدة المعرفة *Knowledge Base***

تحوي قاعدة المعرفة معارف المجال والمفيدة في حل المسائل. تُمثّل المعارف على شكل قواعد استنتاج.

IF <antecedent> مقدمة THEN <consequent> النتيجة

عندما يكون الشرط محققاً يتمّ تطبيق القاعدة وتنفيذ الفعل الموافق.

- **قاعدة البيانات *Data Base***

تحوي قاعدة البيانات مجموعة من الحقائق *Facts* والتي تُستخدم كدخل لشروط القواعد المخزنة في قاعدة المعرفة.

- **محرك الاستدلال *Inference engine***

يقوم محرك الاستدلال بالمحاكمة اللازمة للوصول إلى الحل، عبر تطبيق قواعد قاعدة المعرفة على حقائق قاعدة البيانات لاستنتاج حقائق جديدة.

- **نظام الشرح *Explanation System***

يتنصت نظام الشرح على محرك الاستدلال ليعطي للمستخدم تفسيراً لكيفية وصوله للحقائق الجديدة انطلاقاً من حقائق قاعدة البيانات.

- **واجهة الاستخدام *User Interface***

وهو وسيلة التخاطب بين المستخدم والنظام الخبير.

3 - 2 - استراتيجيات محرك الاستدلال

تُمثّل المعرفة في النظم الخبيرة باستخدام القواعد *Rules*، وتوضع البيانات على شكل حقائق *Facts*.

يقوم محرك الاستدلال بمقارنة كل قاعدة موجودة في قاعدة المعرفة مع الحقائق الموجودة في قاعدة البيانات، وعندما يجد المحرك أن شرط قاعدة محقق يقوم بتنفيذ القاعدة.

تولد عمليات المطابقة والتنفيذ سلسلة معينة ندعوها سلسلة الاستدلال *chains Inference*.

تُحدّد سلسلة الاستدلال كيفية تطبيق القواعد للوصول إلى نتيجة.

3 - 3 - السَلسَلة الأمامية Forward Chaining

تعتمد هذه الاستراتيجية على الفكرة التالية: «ولّد ما يُمكن توليده حتى الوصول للنتيجة المطلوبة».

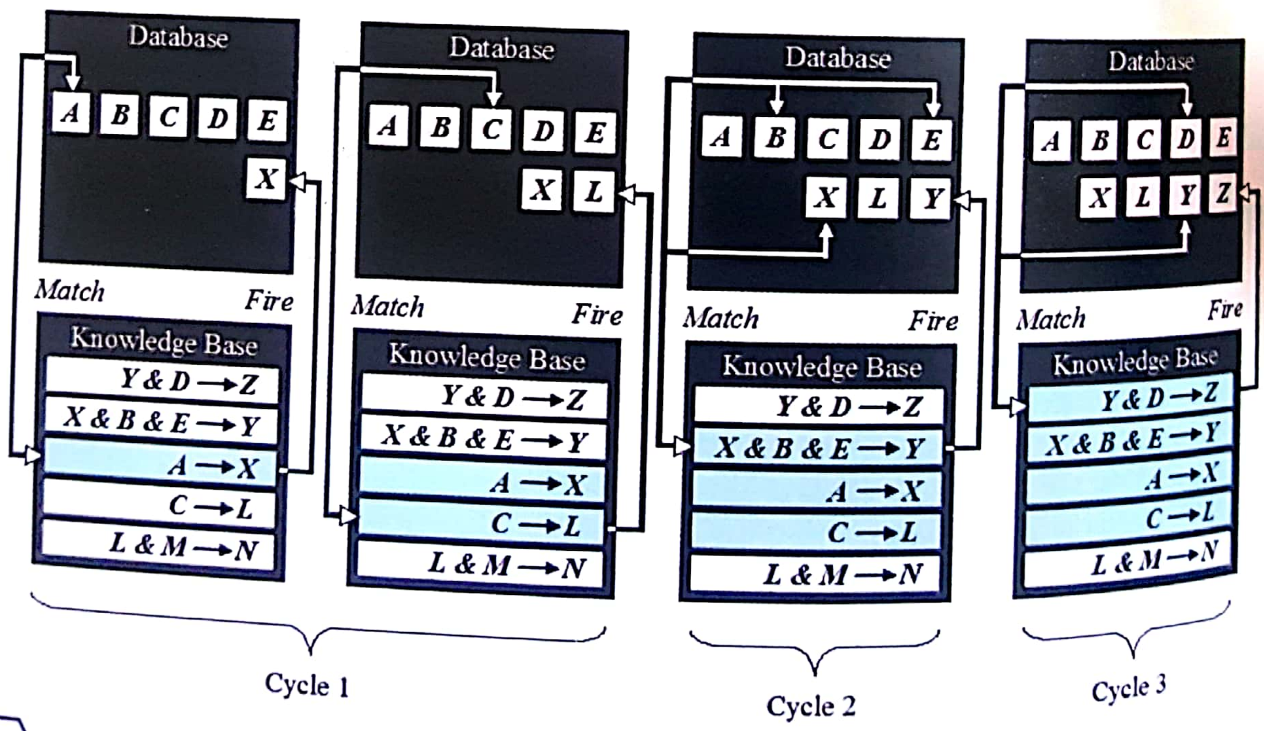
يقوم محرك الاستدلال في هذه الاستراتيجية وفي كل دورة *Cycle* بمسح القواعد لتحديد القواعد القابلة للتطبيق مع قاعدة الحقائق.

يُطبّق محرك الاستدلال هذه القواعد اعتباراً من القاعدة الأعلى مما سيُضيف حقائق جديدة إلى قاعدة الحقائق.

ثم ينتقل إلى الدورة التالية، وهكذا، حتى توليد النتيجة المطلوبة.

- تولّد السَلسَلة الأمامية كل ما يُمكن توليده.
- يُمكن في السَلسَلة الأمامية تطبيق قواعد لا علاقة لها بالمسألة المطلوبة.
- يُمكن في بعض الأحيان وحين يكون الهدف استنتاج حقيقة معينة ألا تكون هذه الاستراتيجية فعالة.

مثال *1*:



مثال 2:

ليكن لدينا قاعدة الحقائق $\{B,C\}$، والقواعد:

(1) $B \wedge D \wedge E \Rightarrow F$

(2) $G \wedge D \Rightarrow A$

(3) $C \wedge F \Rightarrow A$

(4) $B \Rightarrow X$

(5) $D \Rightarrow E$

(6) $X \wedge A \Rightarrow H$

(7) $C \Rightarrow D$

(8) $X \wedge C \Rightarrow A$

(9) $X \wedge B \Rightarrow D$

والمطلوب استنتاج الحقيقة H.

في الدورة الأولى: القاعدة 4 والقاعدة 7 قابلتين للتطبيق.

تضيف القاعدة 4 الحقيقة X إلى قاعدة الحقائق، وتضيف القاعدة 7 الحقيقة D إلى قاعدة الحقائق.

في الدورة الثانية: القاعدة 8 والقاعدة 5 قابلتين للتطبيق.

تضيف القاعدة 8 الحقيقة A إلى قاعدة الحقائق، وتضيف القاعدة 5 الحقيقة E إلى قاعدة الحقائق.

في الدورة الثالثة: القاعدة 1 والقاعدة 7 قابلتين للتطبيق.

تضيف القاعدة 1 الحقيقة F إلى قاعدة الحقائق، وتضيف القاعدة 7 الحقيقة H إلى قاعدة الحقائق.

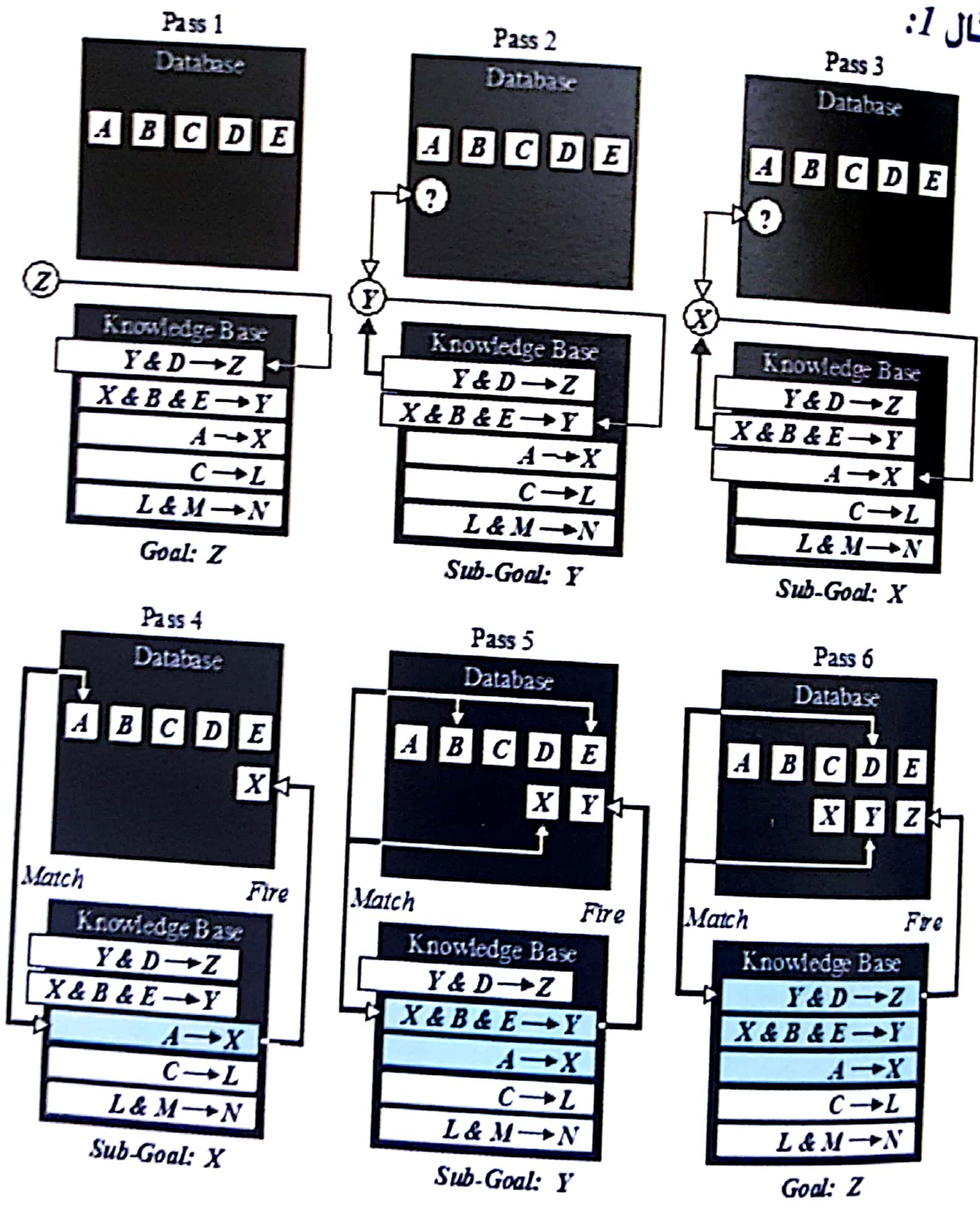
وبذلك تتحقق H وتتوقف عملية البحث.

3 - 4 - السَلسَلة الخلفية Backward Chaining

تعتمد هذه الاستراتيجية على الفكرة التالية: "ابدأ من الهدف وحاول برهان كل ما يلزم لتحقيقه".

تعتمد آلية السَلسَلة الخلفية على البدء من الحقيقة الهدف. والبحث في مجموعة القواعد عن القواعد التي تقع هذه الحقيقة في نتيجتها. ومن ثم إنشاء قائمة بالحقائق الواجب برهانها لنتمكن من تطبيق القواعد السابقة ثم نعاود تطبيق هذه الآلية عودياً على الحقائق الموجودة في هذه القوائم. وهكذا.

مثال 1:



مثال 2:

ليكن لدينا قاعدة الحقائق $\{B,C\}$، والقواعد:

(1) $B \wedge D \wedge E \Rightarrow F$

(2) $G \wedge D \Rightarrow A$

(3) $C \wedge F \Rightarrow A$

(4) $B \Rightarrow X$

(5) $D \Rightarrow E$

(6) $X \wedge A \Rightarrow H$

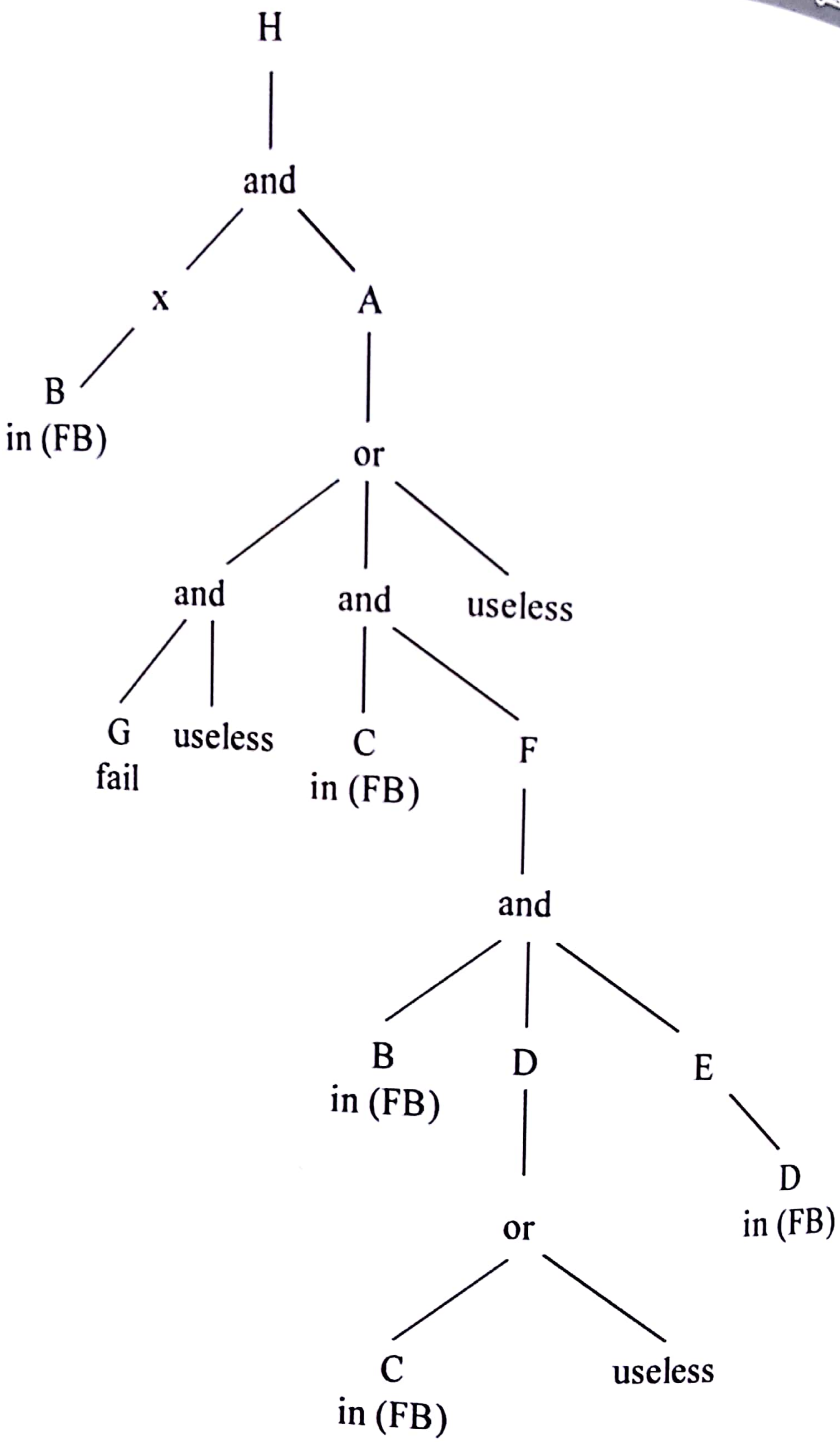
(7) $C \Rightarrow D$

(8) $X \wedge C \Rightarrow A$

(9) $X \wedge B \Rightarrow D$

والمطلوب استنتاج الحقيقة H.

يمكن توصيف تنفيذ خوارزمية السَلسَلة الخلفية بشجرة عقدها إما حقائق وإما إحدى *and* أو *or*. نُسمّي هذه الشجرة، شجرة *and-or*.


H
and
x
A
B
in (FB)
or
and
and
useless
G
fail
useless
C
in (FB)
F
and
B
in (FB)
D
E
or
D
in (FB)
C
in (FB)
useless

3 - 5 - حل التضارب *Conflict Resolution*

ليكن لدينا القواعد التالية في قاعدة المعرفة:

Rule 1:

IF the 'traffic light' is green THEN the action is go

Rule 2:

IF the 'traffic light' is red THEN the action is stop

Rule 3:

IF the 'traffic light' is red THEN the action is go

نلاحظ أن لكلا القاعدتين 2 و 3 نفس الشرط. وبالتالي فيُمكن تطبيق كل منهما في حال تحقق الشرط.

تؤلف هذه القواعد مجموعة تضارب. وعلى محرك الاستدلال أن يُقرر أي من القواعد يجب تطبيقه من هذه المجموعة.

تدعى الطريقة التي تقود إلى اختيار القاعدة الواجب تطبيقها من مجموعة التضارب بحل التضارب *Resolution Conflict*

ستقوم استراتيجية السَلسَلة الأمامية بتطبيق كلتا القاعدتين 2 و 3.

تُطبق القاعدة 2 أولاً لأن ترتيبها قبل القاعدة 3. وبالتالي سيتم توليد النتيجة *Stop*. كذلك فإن القاعدة 3 ستُطبق لأن شرطها محقق، وسيتم توليد النتيجة *Go*!

3 - 6 - طرق حل التضارب

- طبّق القاعدة ذات الأولوية الأكبر *highest priority*

يُمكن في تطبيق بسيط إعطاء الأولويات للقواعد عن طريق ترتيبها في قاعدة المعرفة (حيث ستُطبّق القواعد حسب ترتيبها).

- طبّق القاعدة الخاصة أكثر *most specific rule*

تُدعى هذه الطريقة أيضاً باستراتيجية المطابقة الأكبر *longest matching strategy*. وتعتمد على فرضية أن القاعدة الأكثر خصوصية تعالج معلومات أكثر من القواعد العامة.

- طبّق القاعدة التي يستخدم البيانات الأحدث *most recently entered data*

تقوم هذه الطريقة بإضافة معرف الزمن إلى كل حقيقة في قاعدة البيانات. في حال التضارب، نُطبّق القاعدة التي تعتمد شروطها على الحقائق الأحدث.

3 - 7 - تعلم القواعد *Rules Learning*

يوجد العديد من الطرق المقترحة لتعلم القواعد بشكل استنتاجي.

سنشرح واحدةً منها فيما يلي.

ولتوضيح إجرائية التعلم، سنستخدم ثانيةً مثال الموافقة على قرض مصرفي. إلا أنه عوضاً عن البدء بإعطاء القواعد لهذه المسألة، سنفترض أننا أُعطينا مجموعة أمثلة للتعلم تتألف من قيم الحقائق لعدد من العملاء.

لنعتبر مثلاً مجموعة التعلم المعطاة في الجدول التالي. (نستخدم *1* لـ *True* و *0* لـ *False*).

OK	BAL	INC	RATING	APP	
0	1	0	0	1	1
0	0	1	0	0	2
1	1	0	1	1	3
1	1	1	1	0	4
0	0	1	1	0	5
1	0	1	1	1	6
1	1	1	1	1	7
0	0	1	0	1	8
0	0	0	1	1	9

بيانات المصرف

يُمكن الحصول على هذا الجدول بالرجوع إلى سجلات طلبات القروض المصرفية والقرارات التي اتخذها موظفو المصرف بشأن الموافقة على القرض. ندعو الأمثلة التي تأخذ فيها *OK* القيمة *True* بأمثلة موجبة، والتي تأخذ فيها القيمة *False* بأمثلة سالبة. ونريد باستخدام مجموعة التعلم السابقة استنتاج قواعد من الشكل:

$$\alpha_1 \wedge \alpha_2 \wedge \ldots \alpha_n \Rightarrow OK$$

إذا كان لمقدمة قاعدة القيمة *True* من أجل مثال معين من مجموعة التعلم، فإننا نقول بأن القاعدة تُغطي *cover* المثال. يُمكن أن نُغير في أي قاعدة لجعلها تُغطي عدداً أقل من الأمثلة وذلك بإضافة ذرة لمقدمتها. سيجعل مثل هذا التغيير القاعدة أكثر خصوصية *specific*.

يُمكن لقاعدتين أن تُغطيا أمثلة أكثر مما تُغطيه قاعدة واحدة. يجعل إضافة قاعدة النظام الذي يستخدم هذه القواعد أكثر عمومية *General*. سنبحث عن مجموعة من القواعد التي تغطي كل وفقط الأمثلة الموجبة في مجموعة التعلم.

يُمكن أن تكون عملية البحث عن مثل هذه القواعد مكلفة حسابياً.

نشرح هنا طريقة «شرهه *Greedy*» ندعوها «فرّق تسُد *and Separate Conquer*».

نحاول في هذه الطريقة أولاً أن نجد قاعدة واحدة تُغطي الأمثلة الموجبة فقط (حتى لو كانت لا تُغطي جميع الأمثلة الموجبة). وذلك بالبدء بقاعدة تُغطي جميع الأمثلة (الموجبة والسالبة). ثم نجعلها بالتدريج أكثر خصوصية بإضافة ذرات لمقدمتها.

وبما أن قاعدة واحدة قد لا تكفي لتغطية جميع الأمثلة الموجبة، سنقوم بالتدريج بإضافة القواعد حتى الوصول إلى مجموعة كاملة من القواعد التي تُغطي كل وفقط الأمثلة الموجبة.

لنرى تطبيق هذه الطريقة على مثالنا. نبدأ أولاً بالقاعدة المؤقتة التالية والتي تُغطي جميع الأمثلة:

$$T \Rightarrow OK$$

يجب أن نُضيف الآن ذرة لنجعلها تغطي أمثلة سالبة بشكل أقل، وبهدف الوصول لقاعدة تُغطي الأمثلة الموجبة فقط. والسؤال المطروح هنا هو أي من الذرات *{APP, RATING, INC, BAL}* يجب إضافتها؟.

يوجد العديد من المعايير الممكن استخدامها للقيام بالاختيار. ومن هذه المعايير البسيطة النسبة سهلة الحساب التالية:

$$r_\alpha = n^+_\alpha / n_\alpha$$

حيث:

n_α هي العدد الكلي للأمثلة (الموجبة والسالبة) المغطاة من قبل المقدمة (الجديدة) للقاعدة بعد إضافة الذرة α للمقدمة.

n^+_α هو العدد الكلي للأمثلة الموجبة المغطاة من قبل المقدمة (الجديدة) للقاعدة بعد إضافة α لمقدمتها.

نختار الذرة α التي تُعطي أكبر قيمة لـ r_α.

تكون هذه القيم في مثالنا:

$$r_{APP} = 3 / 6 = 0.5$$

$$r_{RATING} = 4 / 6 = 0.667$$

$$r_{INC} = 3 / 6 = 0.5$$

$$r_{BAL} = 3 / 4 = 0.75$$

ولذا، سنختار الذرة BAL، مما يُعطي القاعدة المؤقتة:

$$BAL \Rightarrow OK$$

تُغطي هذه القاعدة الأمثلة الموجبة 3 و 4 و 7. إلا أنها تُغطي المثال السالب 1، ولذا فيجب أن نُخصصها أكثر.

نستخدم نفس الأسلوب السابق لاختيار ذرة أخرى. وبالطبع، فإن حساب r_α الآن يجب أن يأخذ بالاعتبار أننا قررنا أن مقدمة القاعدة هو BAL. وبالتالي يكون:

$$r_{APP} = 2 / 3 = 0.667$$

$$r_{RATING} = 3 / 3 = 1$$

$$r_{INC} = 2 / 2 = 1$$

لدينا هنا تعادل بين $RATING$ و INC. نختار $RATING$ لأنها تعتمد على مجموعة أكبر من الأمثلة. تُغطي القاعدة الجديدة الأمثلة الموجبة فقط:

$$BAL \wedge RATNG \Rightarrow OK$$

ولذا، فلسنا بحاجة لإضافة ذرات جديدة إلى مقدمة هذه القاعدة.

إلا أن القاعدة:

$$BAL \wedge RATNG \Rightarrow OK$$

لا تُغطي جميع الأمثلة الموجبة، إذ أنها لا تُغطي المثال الموجب *6*. ولذا فعلينا إضافة قاعدة أخرى.

لتعلم القاعدة التالية، نقوم أولاً بحذف جميع الأمثلة الموجبة المغطاة بالقاعدة الأولى من الجدول للحصول على الجدول التالي:

OK	BAL	INC	RATING	APP	
0	1	0	0	1	1
0	0	1	0	0	2
0	0	1	1	0	5
1	0	1	1	1	6
0	0	1	0	1	8
0	0	0	1	1	9
البيانات الباقية					

ونُعيد الآن تطبيق الطريقة مع الجدول الجديد بدءاً من القاعدة $T \Rightarrow OK$ والتي تُغطي الأمثلة السالبة *1* و *2* و *5* و *8* و *9*.

ولاختيار الذرة الواجب إضافتها للقاعدة نحسب:

$$r_{APP} = 1/4 = 0.25$$

$$r_{RATING} = 0/3 = 0$$

$$r_{INC} = 1/4 = 0.25$$

$$r_{BAL} = 0/1 = 0$$

يوجد لدينا مرة أخرى تعادل بين APP و INC.

لنختار بشكل عشوائي APP مما يُعطي القاعدة:

$$APP \Rightarrow OK$$

تُغطي هذه القاعدة الأمثلة السالبة *1* و *8* و *9*. ولذا فيجب إضافة ذرة جديدة لمقدمتها. نحسب إذاً النسب:

$$r_{RATING} = 1/2 = 0.5$$

$$r_{INC} = 1/2 = 0.5$$

$$r_{BAL} = 0/1 = 0$$

ونختار $RATING$ مما يُعطي القاعدة:

$$APP \wedge RATING \Rightarrow OK$$

تُغطي هذه القاعدة المثال السالب *9*. وبجعل هذه القاعدة أكثر خصوصية (باتباع نفس الطريقة) تنتج القاعدة:

$$APP \wedge RATING \wedge INC \Rightarrow OK$$

تُغطي القواعد المستنتجة:

$$BAL \wedge \ RATNG \Rightarrow OK$$

$$APP \wedge RATING \wedge INC \Rightarrow OK$$

جميع وفقط الأمثلة الموجبة. وبهذا نكون قد انتهينا.

3 - 8 - لغة البرمجة المنطقية Prolog

نقوم عند استخدام *Prolog* بكتابة الحقائق والقواعد الضرورية. يقوم *Prolog* بعدها باستخدام الحاكمة الاستنتاجية لحل المسألة المعينة.

يحوي *Prolog* على محرك استدلال يقوم باستخدام استراتيجية بحث من النمط أعلى-أسفل و يسار-يمين *right-left and down-top* على الحقائق والقواعد لاستكشاف الحلول.

تُستخدم الحقائق لتمثيل المعارف. فمثلاً يُمكن كتابة المعارف التالية:

Bill likes Cindy.

Cindy likes Bill.

Bill likes dogs.

بلغة *Prolog*:

likes(bill, cindy).

likes(cindy, bill).

likes(bill, dogs).

ليكن لدينا مثلاً باللغة الطبيعية:

Cindy likes everything that Bill likes.

Caitlin likes everything that is green.

نكتب ذلك في *Prolog*:

likes(cindy, Something):- likes(bill, Something).

likes(caitlin, Something):- green(Something).

مثال: من يستطيع أن يشتري

```
person(kelly).
person(judy).
person(ellen).
person(mark).
car(lemon).
car(hot_rod).
likes(kelly, hot_rod).
likes(judy, pizza).
likes(ellen, tennis).
likes(mark, tennis).
for_sale(pizza).
for_sale(hot_rod).
can_buy(X,Y):- person(X), car(Y), likes(X,Y), for_sale(Y).
```

سيُعطي الهدف *(can_buy(X,Y* **النتائج التالية:**

```
can_buy(judy, pizza)
can_buy(kelly, hot_rod)
```

مثال: ماهي أوراق الشدة التي مجموعها عدد معين

```
card(ace,14).
card(king,13).
card(queen,12).
card(knight,11).
card(10,10).
card(9,9).
card(8,8).
card(7,7).
card(6,6).
card(5,5).
card(4,4).
card(3,3).
card(2,2).
/* find three cards giving the value of N */
find(One, Two, Three, N):-
card(One,Val1),
card(Two,Val2),
card(Three,Val3),
Val1 + Val2 + Val3 = N.
```

مثال: حساب العاملي لعدد طبيعي

```
factorial(1,1):-!.
factorial(X,FactX):-
Y=X-1,
factorial(Y,FactY),
FactX = X*FactY.
```

مثال: الرفع لقوة

```
p(_,0,1):-!.
p(X,1,X):-!.
p(X,N,Result):-
N1 = N-1,
p(X,N1,XN1),
Result = X * XN1.
```

3 - 9 - استخدام القطع *Cut*

يلغي استخدام *Cut* نقاط التراجع على يسارها وعن القواعد السابقة المشابهة.

مثال:

```
cutcount2(X) :-
   X >= 0, !,
   console::write('\r',X),
```

```
    NewX = X+1,
    cutcount2(NewX).
cutcount2(X) :-
console::write(«X is negative.»).
```

3 - 10 - القوائم Lists

أمثلة:

[1, 2, 3, 4]

["toto", "titi", "tata"]

[]

- يُمكن أن تكون عناصر قائمة قوائم
- يُميز رأس القائمة عن بقية العناصر باستخدام |

أمثلة:

- *L=[a,b,c,d,e,f], L=[Head|Tail].*

  Head = a, Tail=[b,c,d,e,f].

- *L=[a,b,c,d,e,f], L=[X,Y|_].*

  X=a, Y=c.

- *L=[a,b,c,d,e,f], L=[_,_|Tail].*

  Tail=[c,d,e,f].

مثال: توليد قائمة من الأعداد الصحيحة

```
genl(0,[]):-!.
genl(N,[N|L]):-
N1=N-1,
genl(N1,L).
```

مثال: عدّ عناصر قائمة

```
length_of([], 0).
length_of([_|T],L):-
    length_of(T,TailLength),
    L = TailLength + 1.
```

مثال: انتماء عنصر لقائمة

```
member(X,[X|_]).
member(X,[_|List]) :-
 member(X,List).
```

مثال: إضافة عنصر لقائمة

```
append([],L,L).
append([H|L1],L2,[H|L3]):-
 append(L1,L2,L3).
```

3 - 11 - عدم التوكيد Uncertainty

نستعمل في حياتنا لغة يشوبها الالتباس وتعاني من عدم الدقة. فنحن نصف الحقائق باستخدام تعابير مثل غالباً أو بعض الأحيان أو نادراً. وبالتالي فمن الصعوبة التعبير عن معارفنا باستخدام الشكل THEN-IF لقواعد الاستدلال.

3 - 12 - محاكمة بايز Bayesian Reasoning

لنفرض أن جميع قواعد الاستدلال في قاعدة المعرفة لها الشكل التالي:

IF H is true THEN E is true with probability p

تُمثّل عادة H في النظم الخبيرة فرضية hypothesis، بينما يكون E دليل evidence يؤيد هذه الفرضية.

يكون لدينا من أجل مجموعة من الفرضيات Hm,...,H2,H1 ومجموعة من الأدلة En,...,E2,E1:

$$p(H_i|E_1\,E_2\ldots E_n)=\frac{p(E_1|H_i)\times p(E_2|H_i)\times\ldots\times p(E_n|H_i)\times p(H_i)}{\sum_{k=1}^{m}p(E_1|H_k)\times p(E_2|H_k)\times\ldots\times p(E_n|H_k)\times p(H_k)}$$

مثال:

ليكن لدينا المسألة التالية:

لنفترض أن خبير أُعطي ثلاثة أدلة مستقلة:

E_1, E_2, E_3

مع ثلاث فرضيات:

H_1, H_2, H_3

مع الاحتمالات الأولية لهذه الفرضيات:

$p(H_1), p(H_2), p(H_3)$

يُحدّد الخبير أيضاً الاحتمالات الشرطية لحدوث كل دليل من أجل كل الفرضيات المحتملة.

يُبين الجدول التالي مختلف الاحتمالات المعطاة من قبل الخبير:

Probability	Hypothesis		
	$i=1$	$i=2$	$i=3$
$p(H_i)$	0.40	0.35	0.25
$p(E_1 \mid H_i)$	0.3	0.8	0.5
$p(E_2 \mid H_i)$	0.9	0.0	0.7
$p(E_3 \mid H_i)$	0.6	0.7	0.9

لنفرض أننا لاحظنا أولاً الدليل E_3، يقوم النظام الخبير بحساب الاحتمالات اللاحقة لكل الفرضيات:

$$p(H_i \mid E_3) = \frac{p(E_3 \mid H_i) \times p(H_i)}{\sum_{k=1}^{3} p(E_3 \mid H_k) \times p(H_k)}, \qquad i = 1, 2, 3$$

وبهذا يكون:

$$p(H_1 \mid E_3) = \frac{0.6 \cdot 0.40}{0.6 \cdot 0.40 + 0.7 \cdot 0.35 + 0.9 \cdot 0.25} = 0.34$$

$$p(H_2 \mid E_3) = \frac{0.7 \cdot 0.35}{0.6 \cdot 0.40 + 0.7 \cdot 0.35 + 0.9 \cdot 0.25} = 0.34$$

$$p(H_3 \mid E_3) = \frac{0.9 \cdot 0.25}{0.6 \cdot 0.40 + 0.7 \cdot 0.35 + 0.9 \cdot 0.25} = 0.32$$

لاحظ أنه بعد ملاحظة الدليل E_3، فإن الاعتقاد بالفرضية H_1 ينخفض ويصبح مساوياً للاعتقاد بالفرضية H_2. كذلك فإن الاعتقاد بالفرضية H_3 يرتفع ويصبح تقريباً مساوياً للاعتقاد بـ H_1 و H_2.

لنفرض الآن ظهور الدليل E_1. تُصبح الاحتمالات اللاحقة:

$$p(H_i|E_1E_3)=\frac{p(E_1|H_i)\times p(E_3|H_i)\times p(H_i)}{\sum_{k=1}^{3}p(E_1|H_k)\times p(E_3|H_k)\times p(H_k)},\qquad i=1,2,3$$

وبالتالي:

$$p(H_1|E_1E_3)=\frac{0.3\cdot0.6\cdot0.40}{0.3\cdot0.6\cdot0.40+0.8\cdot0.7\cdot0.35+0.5\cdot0.9\cdot0.25}=0.19$$

$$p(H_2|E_1E_3)=\frac{0.8\cdot0.7\cdot0.35}{0.3\cdot0.6\cdot0.40+0.8\cdot0.7\cdot0.35+0.5\cdot0.9\cdot0.25}=0.52$$

$$p(H_3|E_1E_3)=\frac{0.5\cdot0.9\cdot0.25}{0.3\cdot0.6\cdot0.40+0.8\cdot0.7\cdot0.35+0.5\cdot0.9\cdot0.25}=0.29$$

لاحظ أن الفرضية H_2 أصبحت الآن هي المرجحة.

وبعد ظهور الدليل E_2، تُصبح الاحتمالات اللاحقة النهائية لجميع الفرضيات:

$$p(H_i|E_1E_2E_3)=\frac{p(E_1|H_i)\times p(E_2|H_i)\times p(E_3|H_i)\times p(H_i)}{\sum_{k=1}^{3}p(E_1|H_i)\times p(E_2|H_i)\times p(E_3|H_i)\times p(H_k)},\qquad i=1,2,3$$

وبالتالي:

$$p(H_1|E_1E_2E_3)=\frac{0.3\cdot0.9\cdot0.6\cdot0.40}{0.3\cdot0.9\cdot0.6\cdot0.40+0.8\cdot0.0\cdot0.7\cdot0.35+0.5\cdot0.7\cdot0.9\cdot0.25}=0.45$$

$$p(H_2|E_1E_2E_3)=\frac{0.8\cdot0.0\cdot0.7\cdot0.35}{0.3\cdot0.9\cdot0.6\cdot0.40+0.8\cdot0.0\cdot0.7\cdot0.35+0.5\cdot0.7\cdot0.9\cdot0.25}=0$$

$$p(H_3|E_1E_2E_3)=\frac{0.5\cdot0.7\cdot0.9\cdot0.25}{0.3\cdot0.9\cdot0.6\cdot0.40+0.8\cdot0.0\cdot0.7\cdot0.35+0.5\cdot0.7\cdot0.9\cdot0.25}=0.55$$

نلاحظ أن الترتيب الأولي للفرضيات كان:

H_1, H_2, H_3

إلا أنه، وبعد ظهور الأدلة، تبقى فقط الفرضيتان H_1 و H_3 ممكنتين.

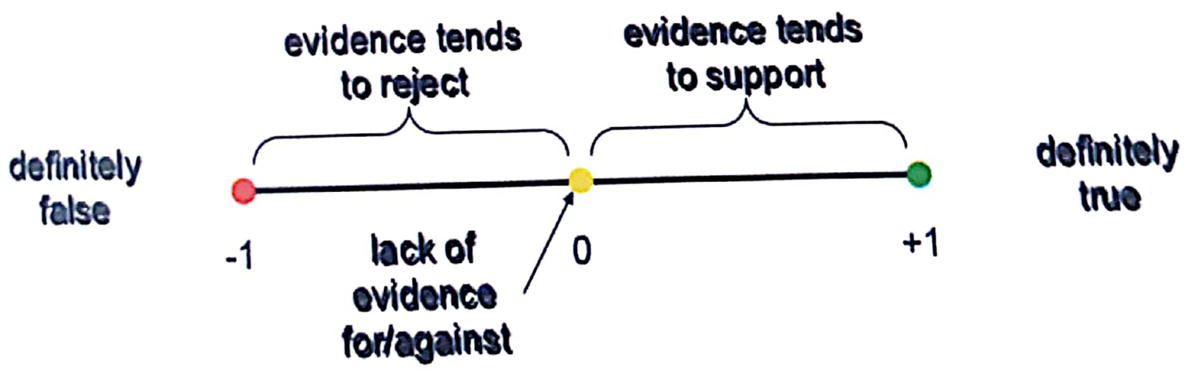
3 - 13 - نظرية معامل الثقة Certainty Factors Theory

تُمثل نظرية معامل الثقة بديلاً مهماً ومنتشراً لطرق محاكمة بايز.

يقيس معامل الثقة *factor certainty* لفرضية H درجة اعتقاد الخبير *belief s 'expert*.

تتراوح قيمة معامل الثقة بين *1* (مؤكد صحيح) وبين *-1* (مؤكد خطأ).

- **$CF=1$ الفرضية صحيحة تماماً**
- **$CF=-1$ الفرضية خاطئة تماماً**
- **$CF=0$ لا يوجد دليل مع-ضد الفرضية**
- **$CF>0$ يوجد دلائل تدعم صحة الفرضية**
- **$CF<0$ لا يوجد دلائل تدعم صحة الفرضية**



تتألف قاعدة المعرفة في النظم الخبيرة التي تعتمد على معامل الثقة من قواعد استدلال لها الشكل التالي:

IF E THEN H {cf}

حيث يُمثل معامل الثقة *cf* الاعتقاد بالفرضية *H* إذا علمنا وقوع الدليل *E*.

3 - 14 - حساب معامل الثقة

يجب حساب ثقة نتيجة القاعدة عندما يكون الدليل في مقدمة القاعدة غير مؤكد.

$$cf(H,E) = cf(E) * cf$$

مثلاً، إذا كان لدينا القاعدة:

IF sky is clear THEN the forecast is sunny {cf 0.8}

وكانت قيمة معامل الثقة لـ *"sky is clear"* هي 0.5 فإن:

$$cf(H,E) = 0.5 * 0.8 = 0.4$$

حالة قواعد الربط

IF	<evidence E_1>
	⋮
AND	<evidence E_n>
THEN	<hypothesis H> {cf}

يُحسب معامل الثقة لفرضية *H* كما يلي:

$$cf(H, E1 \wedge E2 \wedge \ldots \wedge En = (\min\{cf(E1), cf(E2), \ldots, cf(En)\} * cf$$

مثال:

IF sky is clear

AND the forecast is sunny

THEN the action is 'wear sunglasses' cf 0.8

وإذا كان معامل الثقة لـ *"sky is clear»* هو *0.9* ومعامل الثقة لـ *"the forecast is sunny»* هو *0.7* يكون:

$$cf\,(H,E_1 \wedge E_2) = \min\{0.9, 0.7\} * 0.8 = 0.56$$

حالة قواعد الفصل

IF <evidence E_1>

⋮

OR <evidence E_n>

THEN <hypothesis H> $\{cf\}$

يُحسب معامل الثقة لفرضية H كما يلي:

$$cf\,(H,E_1\text{Ú}E_2\text{Ú}\ldots\ \text{Ú}E_n) = \max\{cf\,(E_1), cf\,(E_2),\ldots, cf\,(E_n)\} * cf$$

مثال:

IF sky is overcast

OR the forecast is rain

THEN the action is 'take an umbrella' {cf 0.9}

وإذا كان معامل الثقة لـ *"sky is clear»* هو *0.9* ومعامل الثقة لـ *"the forecast is sunny»* هو *0.7* يكون:

$$cf\,(H,E_1\text{Ú}E_2) = \max\{0.9, 0.7\} * 0.9 = 0.81$$

دمج معاملات القواعد

عندما نحصل على نفس النتيجة من أكثر من قاعدة، فيجب القيام بدمج معاملات الثقة التي نحصل عليها جراء تطبيق كل قاعدة بطريقة معينة للوصول إلى معامل الثقة النهائي للنتيجة.

لنفرض مثلاً أن لدينا القاعدتين التاليتين في قاعدة المعرفة:

Rule 1: IF A is X THEN C is Z {cf 0.8}

Rule 2: IF B is Y THEN C is Z {cf 0.6}

والسؤال: ما هو معامل الثقة لـ Z إذا كانت كلتا القاعدتين قابلتين للتطبيق؟

دمج معاملات القواعد

لحساب عامل الثقة الناتج عن أكثر من قاعدة، نستخدم:

$$cf(cf_1, cf_2) = \begin{cases} cf_1 + cf_2 \times (1 - cf_1) & \text{if } cf_1 > 0 \text{ and } cf_2 > 0 \\ \dfrac{cf_1 + cf_2}{1 - min\,[|cf_1|, |cf_1|]} & \text{if } cf_1 < 0 \text{ or } cf_2 < 0 \\ cf_1 + cf_2 \times (1 + cf_1) & \text{if } cf_1 < 0 \text{ and } cf_2 < 0 \end{cases}$$

حيث:

- cf_1 هو معامل الثقة بالفرضية H والناتج عن القاعدة $Rule_1$
- cf_2 هو معامل الثقة بالفرضية H والناتج عن القاعدة $Rule_2$
- $|cf_1|$ و $|cf_2|$ هي القيم المطلقة لـ cf_1 و cf_2.

مثال:

ليكن لدينا:

IF A and B or C and not D

THEN X with CF 0.6

ومعاملات الثقة:

$CF(A) = 0.3$

$CF(B) = 0.5$

$CF(C) = 0.4$

$CF(D) = -0.7$

والمطلوب حساب معامل الثقة للنتيجة X.

$CF(E) = CF\ (A \text{ and } B \text{ or } C \text{ and not } D)$

$= max(min(0.3,0.5), min(0.4,0.7))$

$= max(0.3,0.4) = 0.4$

$CF(H,E)=0.6$

$CF(X)=CF(H) = CF(E) * CF(H,E) = 0.4 * 0.6 = 0.24$

مثال:

ليكن لدينا:

Rule 1: IF A OR B THEN C (certainty factor 0.3)

Rule 2: IF C OR D THEN H (certainty factor 0.8)

Rule 3: IF E OR F THEN H (certainty factor 0.2)

مع معاملات الثقة:

A is 0.2, B is 0.5, D is 0.3, E is 0.6, F is 0.7

والمطلوب حساب معامل الثقة لـ *H*.

نقوم بالحل كما يلي:

IF A OR B , so the max value of A and B: max (0.2, 0.5) = 0.5

then the certainty of C = 0.5 * certainty factor 0.3 = 0.15

IF C OR D, so the max value of C and D: max (0.15, 0.3) = 0.3

then the certainty of H = 0.3 * certainty factor 0.8 = 0.24.

IF E OR F, so the max value of E and F: max (0.6, 0.7) = 0.7

then the certainty of H = 0.7 * certainty factor 0.2 = 0.14

وبالتالي يكون معامل الثقة:

$$0.24 + 0.14\ (1 - 0.24) = 0.3464$$

3 - 15 - أسئلة متعددة الخيارات

1. لتكن مجموعة القواعد التالية واحتمالاتها:

R1: If Battery is bad Then horn does not work with prob=0.3

R2: If Battery is bad Then engine does not start with prob=0.6

R3: If Spark plugs are bad Then horn does not work with prob=0.6

R4: If Spark plugs are bad Then engine does not start with prob=0.4

مع الاحتمالات:

prob(Battery is bad)= 0.4

prob(Spark plugs are bad)= 0.6

وبفرض أن لدينا المشاهدات التالية:

"horn does not work"

"engine does not start"

احسب احتمال الفرضية التالية:

"Spark plugs are bad"

a. *0.33*

b. ***0.66***

c. *0*

d. ولا خيار مما سبق

2. ليكن لدينا القواعد التالية في *Prolog*

```
A([X|_], 0, X).
A([_|T], I, X) :- (I > 0) , A(T, I - 1, X).
B(_, [ ], [ ]).
B(L, [I|T], [X|R]) :- A(L, I, X), B(L, T, R).
```

تكون قيمة *R* بعد الاستدعاء التالي:

B([1,3,5,7], [0,3], R)

a. **<u>R = [1,7]</u>**

b. *R = [1,6]*

c. *R = [2,7]*

d. غير ذلك

3. لتكن مجموعة القواعد التالية:

Rule 1 : IF A OR B THEN C (certainty factor 0.3)

Rule 2 : IF C OR D THEN H (certainty factor -0.8)

Rule 3 : IF E OR F THEN H (certainty factor 0.2)

مع معاملات الثقة:

$CF(A) = 0.2$

$CF(B) = 0.5$

$CF(D) = 0.3$

$CF(E) = -0.6$

$CF(F) = -0.7$

يكون معامل الثقة لـ H مساوياً إلى:

a. 0.66

b. -0.66

c. 0

d. **ولا خيار مما سبق**

4. لتكن مجموعة الأمثلة التالية:

	A	B	C	D	OK
1	0	1	0	1	0
2	0	0	1	0	0
3	1	1	0	1	1
4	1	0	1	1	1
5	1	0	1	0	0
6	1	1	1	0	1
7	1	1	1	1	1
8	0	1	1	0	0
9	1	1	0	0	0

تعطي خوارزمية التعلم كقاعدة أولى:

a. $A \wedge B \Rightarrow OK$

b. **$A \wedge D \Rightarrow OK$**

c. $B \wedge C \Rightarrow OK$

d. $B \wedge D \Rightarrow OK$

5. لتكن مجموعة القواعد التالية:

Rule 1 : IF A OR B THEN C (certainty factor 0.3)

Rule 2 : IF C OR D THEN H (certainty factor 0.8)

Rule 3 : IF E OR F THEN H (certainty factor 0.2)

مع معاملات الثقة:

$CF(A) = 0.2$

$CF(B) = 0.5$

$CF(D) = 0.3$

$CF(E) = -0.6$

$CF(F) = -0.7$

يكون معامل الثقة لـ H مساوياً إلى:

a. 0.66

b. <u>**0.13**</u>

c. 0.33

d. -0.13

6. ليكن برنامج *Prolog* التالي:

```
x(0,[ ]).
x(N,[N|L]):- x(N-2,L).
```

يطبع الاستدعاء:

```
x(10, L)
console::write(L)
```

a. *[2 ,6,4 ,8 ,10]*

b. *Stack Overflow* ومن ثم القائمة *[10, 8, 4 ,6, 2]*

c. <u>**[2 ,6,4 ,8 ,10] ومن ثم الرسالة Stack Overflow**</u>

d. ولا خيار مما سبق

7. لتكن مجموعة القواعد التالية:

Rule 1 : IF A OR B THEN C (certainty factor 0.3)

Rule 2 : IF C OR D THEN H (certainty factor 0.8)

Rule 3 : IF E OR F THEN H (certainty factor 0.2)

مع معاملات الثقة:

$CF(A) = 0.2$

$CF(B) = 0.5$

$CF(D) = 0.3$

$CF(E) = -0.6$

$CF(F) = -0.7$

يكون معامل الثقة لـــ H مساوياً إلى:

a. 0.66

b. -0.13

c. 0.33

d. <u>**0.13**</u>

8. لنكن مجموعة القواعد التالية:

$R1: \quad A \rightarrow B$

$R2: \quad A \rightarrow C$

$R3: \quad B \rightarrow C$

$R4: \quad C \rightarrow D$

$R5: \quad D \rightarrow E$

$R6: \quad G \rightarrow F$

$R7: \quad E \rightarrow F$

$R8: \quad H \rightarrow G$

وحيث الحقيقة الوحيدة الصحيحة هي A. والهدف هو F.

في استراتيجية السلسلة الأمامية تكون عدد الحلقات *cycles* للوصول للهدف:

a. 3

b. <u>4</u>

c. 5

d. 6

9. لتكن مجموعة القواعد التالية:

$R1: \quad A \rightarrow B$

$R2: \quad A \rightarrow C$

$R3: \quad B \rightarrow C$

$R4: \quad C \rightarrow D$

$R5: \quad D \rightarrow E$

$R6: \quad G \rightarrow F$

$R7: \quad E \rightarrow F$

$R8: \quad H \rightarrow G$

وحيث الحقيقة الوحيدة الصحيحة هي A. والهدف هو F.

في استراتيجية السلسلة الخلفية يكون عدد القواعد المنفذة *Fired* للوصول للهدف:

a. 3

b. <u>4</u>

c. 5

d. 6

10. لتكن لدينا مجموعة القواعد التالية مع الاحتمالات الموافقة:

Let the following rules:

R1:If Battery is bad *Then engine does not start*
prob=0.3

R2:If Battery is bad *Then lights do not come on*
prob=0.8

R3: If Battery is bad *Then horn does not work*
prob=0.5

R4: If Spark plugs are bad *Then engine does not start*
prob=0.9

R5: If Spark plugs are bad *Then lights do not come on*
prob=0.0

R6: If Spark plugs are bad *Then horn does not work*
prob=0.7

R7: If Electricity cable is cut *Then engine does not start*
prob=0.6

R8: If Electricity cable is cut *Then lights do not come on*
prob=0.7

R9: If Electricity cable is cut *Then horn does not work*
prob=0.9

علما أن:

prob(Battery is bad)=0.4 , prob(Spark plugs are bad)=0.35, prob(Electricity cable is cut)=0.25

وبفرض أن لدينا المشاهدات التالية:

"horn does not work", "engine does not start", and "lights do not come on".

احسب احتمال الفرضية:

"Electricity cable is cut".

a. <u>***55%***</u>

b. *45%*

c. *0%*

d. **ولاخيار مما سبق**

11. لتكن لدينا مجموعة القواعد التالية مع معاملات التوكيد:

R1: If engine does not start Then Battery is bad
CF=0.3

R2: If lights do not come on Then Battery is bad
CF=0.8

R3: If horn does not work Then Battery is bad
CF=0.5

وبفرض أن لدينا معاملات التوكيد التالية للمشاهدات:

- *CF("horn does not work")= 1*
- *CF("engine does not start")= -1*
- *CF("lights do not come on")= -1*

احسب معامل التوكيد للفرضية:

CF("Battery is bad")

a. <u>***-0.72***</u>

b. *0.72*

c. *-0.27*

d. **ولاخيار مما سبق**

12. ليكن لدينا القواعد التالية في *Prolog*

fis(L, X, C) :- fi(L, 0, X, C).

fi([ ], _, _, [ ]).

fi([X|T], I, X, [I|C]) :- fi(T, I + 1 , X, C).

fi([H|T], I, X, C) :- (X != H), fi(T, I + 1, X, C).

تكون قيمة C بعد الاستدعاء التالي:

fis([1,5,5,1,3,5], 5, C)

a. C = [1,2,5]

b. C = [2,1,5]

c. C = [5,2,1]

d. غير ذلك

الفصل الرابع

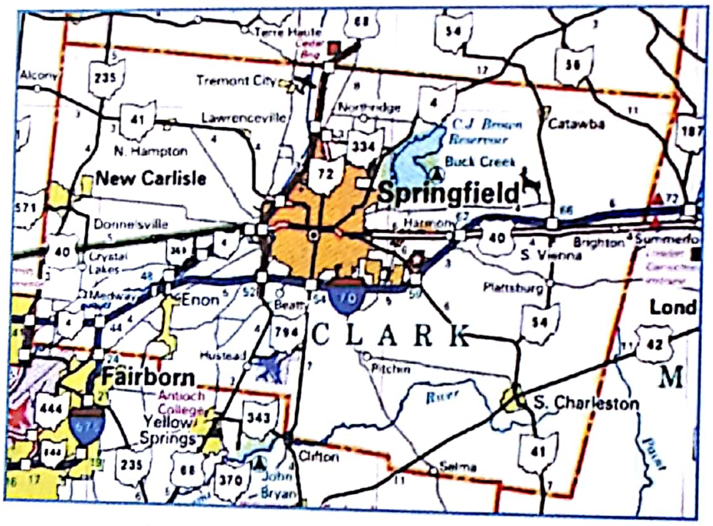
البحث *Search*

يلعب البحث دوراً أساسياً في الكثير من مسائل وتطبيقات الذكاء الصنعي. تُعتبر خوارزميات البحث العمود الفقري لجميع المسائل التي تحتاج لاستكشاف الخيارات المتعددة بشكل منهجي.

يرتبط مفهوم حل المسائل في الذكاء الصنعي ارتباطاً وثيقاً بتقانات البحث الموجه في البيان (سنرى أن البحث الموجه يعني وجود آلية "ذكية" تقود عملية سبر البيان).

4 - 1 - البيانات *Graphs*

- تتواجد البيانات *Graphs* في كل مكان: شبكات الطرق، الخطوط الجوية، شبكات الحواسيب.
- نهتم في جميع الحالات بإيجاد مسار *Path* عبر البيان يحقق خصائص معينة.



- يُمكن أن نكتفي في بعض الحالات بأي مسار، أو نريد إيجاد المسار ذي الكلفة الأقل.
- من أهم منهجيات حل المسائل في الذكاء الصنعي تحويل المسألة إلى مسألة بحث في بيان.

- لاستخدام هذه المنهجية يجب تحديد الحالات والأفعال واختبار الهدف.
- نفترض أن الحالة كاملة *Complete* أي أنها تُمثل جميع جوانب المسألة.
- نفترض أن الأفعال حتمية *Deterministic* أي أننا نعرف الحالة تماماً بعد تطبيق فعل.

4 - 2 - خوارزميات البحث في بيان الحالات

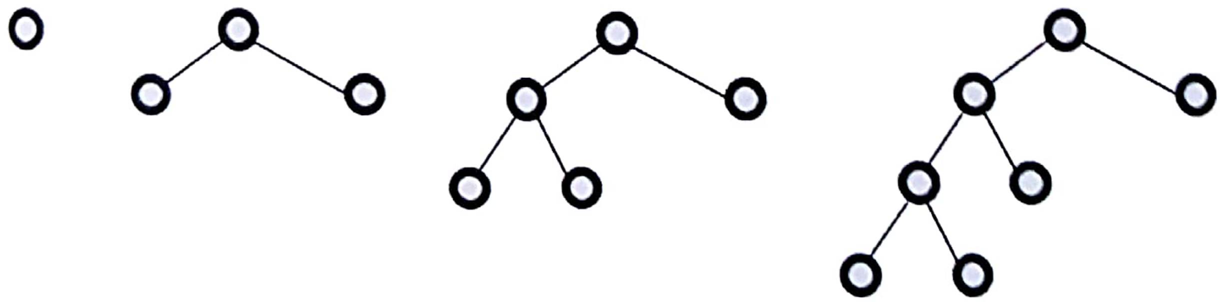
- بعد تعريف فضاء المسألة (تمثيل الحالات، الأفعال أو المعاملات، الحالة الابتدائية، الحالة النهائية) يبدأ البحث!
- انطلاقاً من الحالة الابتدائية نعاود تطبيق الأفعال الممكنة حتى الوصول إلى الحالة النهائية.
- إلا أن فضاء البحث عادةً ضخم جداً جداً !
- وبالتالي نحتاج إلى منهجيات تقود البحث.

4 - 3 - خوارزميات البحث الأعمى *Blind Search*

لا تستخدم هذه الخوارزميات معلومات متعلقة بالمسألة.

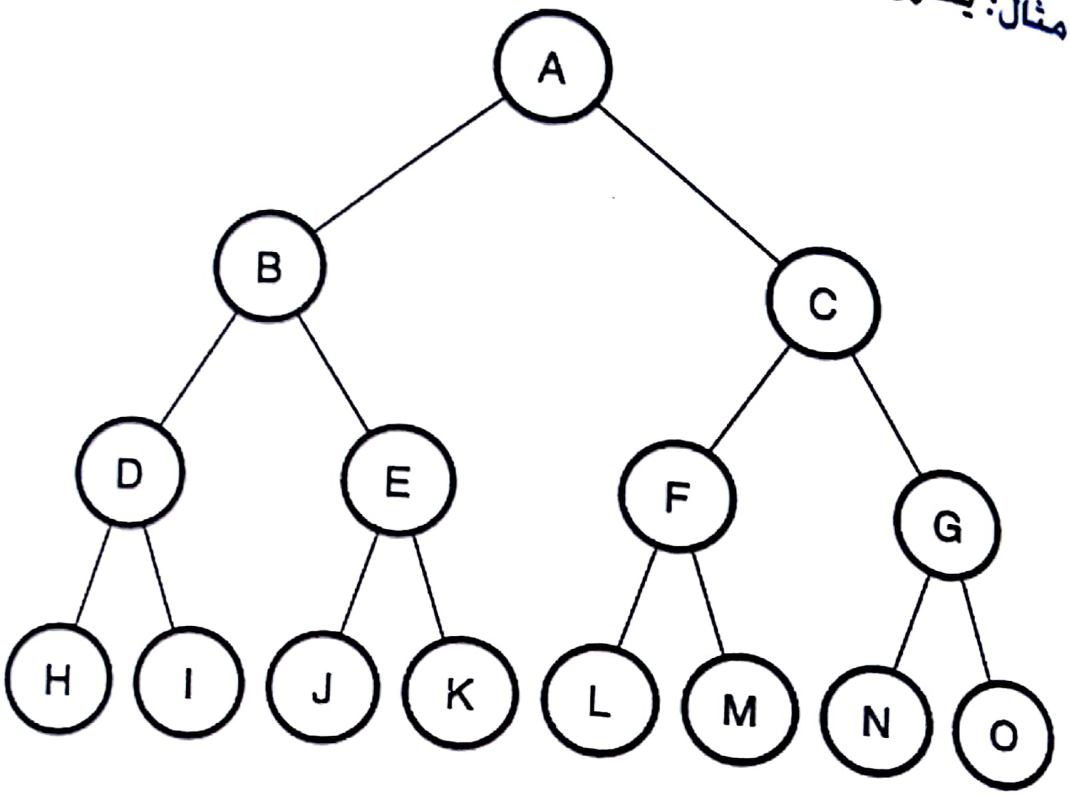
4 - 4 - البحث بالعمق - أولاً *Depth-First Search*

تعتمد هذه الخوارزمية على المبدأ: طوّر العقدة الأعمق غير المطورة.



نوضع العقد الحدود في مكدّس من النوع *LIFO* أي الداخل أخيراً الخارج أولاً.

مثال: يكون ترتيب تطوير العقد في الشجرة التالية:

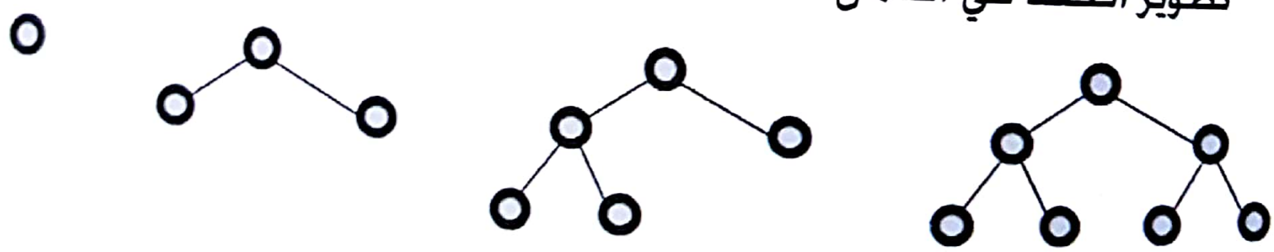


هو:

A, B, D, H, I, E, J, K, C, F, L, M, G, N, O

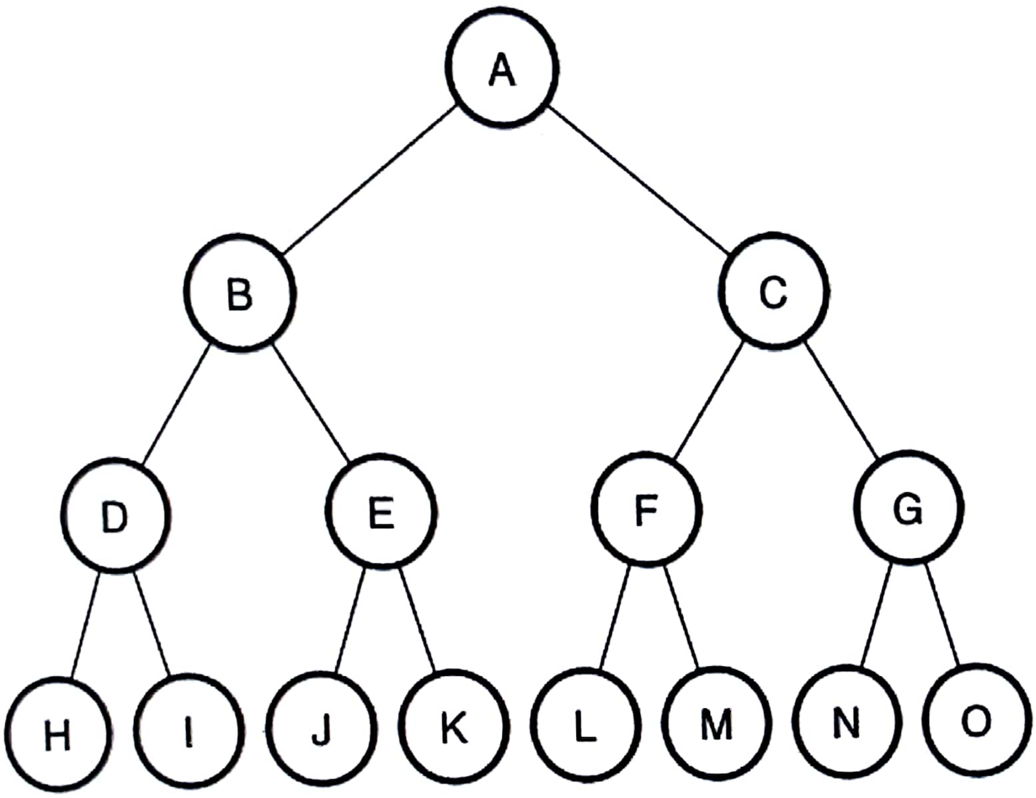
4 - 5 - البحث بالعرض - أولاً *Breadth-First Search*

تعتمد هذه الخوارزمية على المبدأ: طوّر جميع العقد في العمق i قبل تطوير العقد في العمق $i+1$.



توضع العقد الحدود في رتل من النوع *FIFO* أي الداخل أولاً الخارج أولاً.

مثال: يكون ترتيب تطوير العقد في الشجرة التالية:

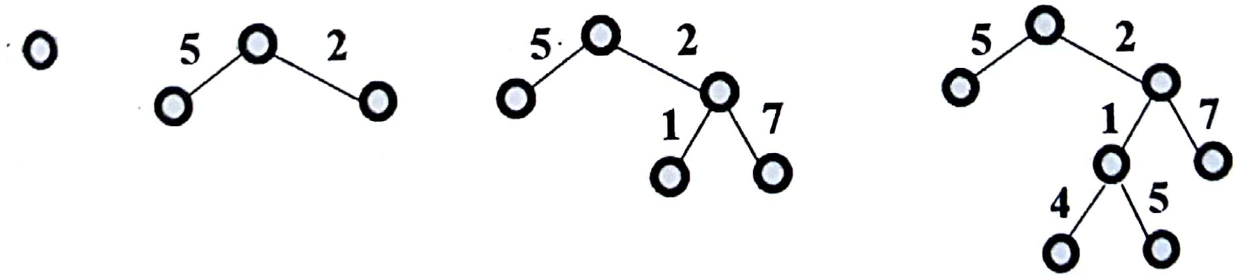


هو:

A ,B ,C ,D ,E ,F ,G ,H ,I ,J ,K ,L ,M ,N ,O

4 - 6 - البحث وفق الكلفة المنتظمة *Uniform-Cost Search*

تعتمد هذه الخوارزمية على المبدأ: طوّر العقدة ذات الكلفة الأقل. أي أن الخوارزمية تقوم بتطوير العقد وفق الكلف المتزايدة. وبالتالي فإن أي عقدة هدف نصل إليها تكون الحل الأمثل.



توضع إذاً العقد الحدود في رتل مرتب تصاعدياً وفق الكلفة. أي أن إضافة العقد الجديدة المولدة تتمّ وبحيث نحافظ على الترتيب التصاعدي للكلف.

4 - 7 - التجريبيات *Heuristics*

إن طرائق البحث العمياء هي طرائق شاملة تهدف إلى إيجاد طريق حل. إلا أن تطبيقها غير واقعي، من أجل معظم المسائل، لأنه يتطلب معالجة عدد هائل من العقد قبل مصادفة الحل.

تكمن الفكرة الأساسية لخوارزميات البحث المطّلعة مع بيان الحل في استعمال معلومة تجريبية خاصة بالمسألة المعالجة تؤدي إلى تخفيض عدد العقد المعالجة.

التجريبية هي تابع عندما يُطبّق على حالة يُعيد رقم يُقدّر قرب هذه الحالة من الهدف.

أي أن التجريبية تُخبر تقريباً كم تبقى للوصول للهدف (الأرقم الأصغر أفضل).

يُمكن أن تبخس التجريبية *underestimate* أو تغالي *overestimate* في تقدير البعد عن الهدف.

سوف نرى أن التجريبيات التي نعتبرها مقبولة *admissible* هي التجريبيات التي يكون تقديرها دائماً أصغر من الكلفة الحقيقية.

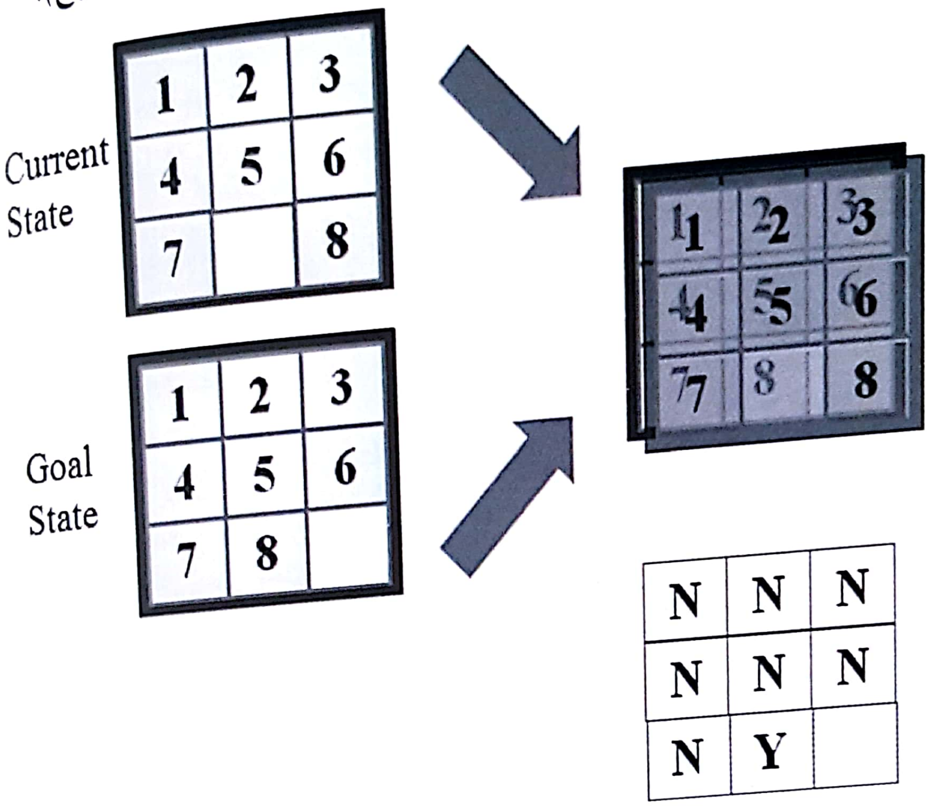
مثال*1*: أقصر طريق

يُمكن في مسألة البحث عن المسار الأمثلي لخريطة من المدن اعتماد مسافة خط النظر بين مدينة ما والمدينة الهدف كمقياس لقرب هذه المدينة من الهدف.

مثال2: لعبة التكوتان

يُمكن في مسألة التاكوتان اعتماد التجريبية التالية لقياس قرب حالة من الهدف:

عدد الخانات الموجودة في غير مكانها الصحيح (ماعدا الفراغ).

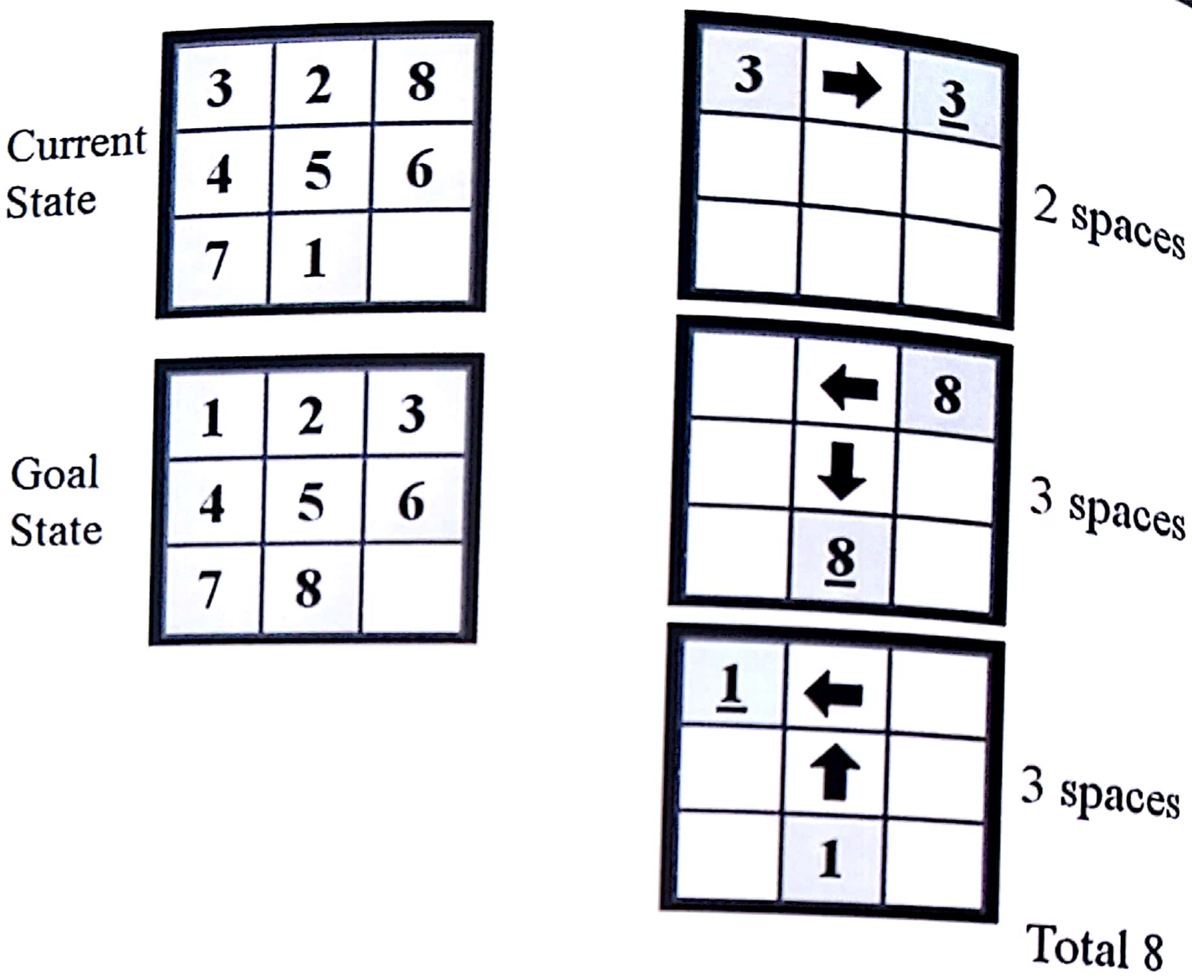


في مثالنا تكون قيمة التجريبية *1* إذ فقط لا توجد الخانة *8* في موضعها الصحيح.

مثال3: لعبة التكوتان

يُمكن في مسألة التاكوتان مسافة مانهاتن *Distance Manhattan* لقياس قرب حالة من الهدف:

(مجموع عدد الخانات التي يجب أن نعبرها لنوصل كل خانة إلى مكانها الصحيح).



4 - 8 - خوارزمية تسلق التلة *Hill Climbing Search*

ليكن التابع h تقديراً للكلفة من العقدة n إلى الهدف.

ندعو هذا التابع بالتجريبية *heuristic*.

مثلاً، يُمكن في مسألة التنقل بين المدن أن يكون هذا التابع مسافة خط النظر بين المدينة الحالية والمدينة الهدف.

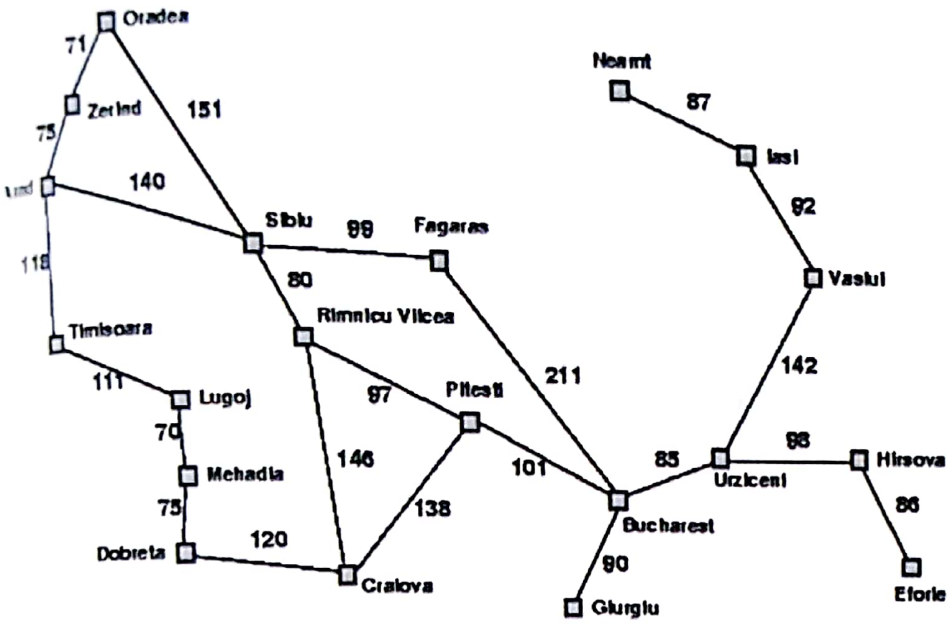
$$h_{SLD}(n) = \text{straight-line distance from } n \text{ to goal}$$

تقوم خوارزمية تسلق التلة بتطوير العقدة التي يكون تقديرها أقرب للهدف.

(بخلاف خوارزمية البحث وفق الكلفة المنتظمة والتي تطور العقدة ذات الكلفة الأقل).

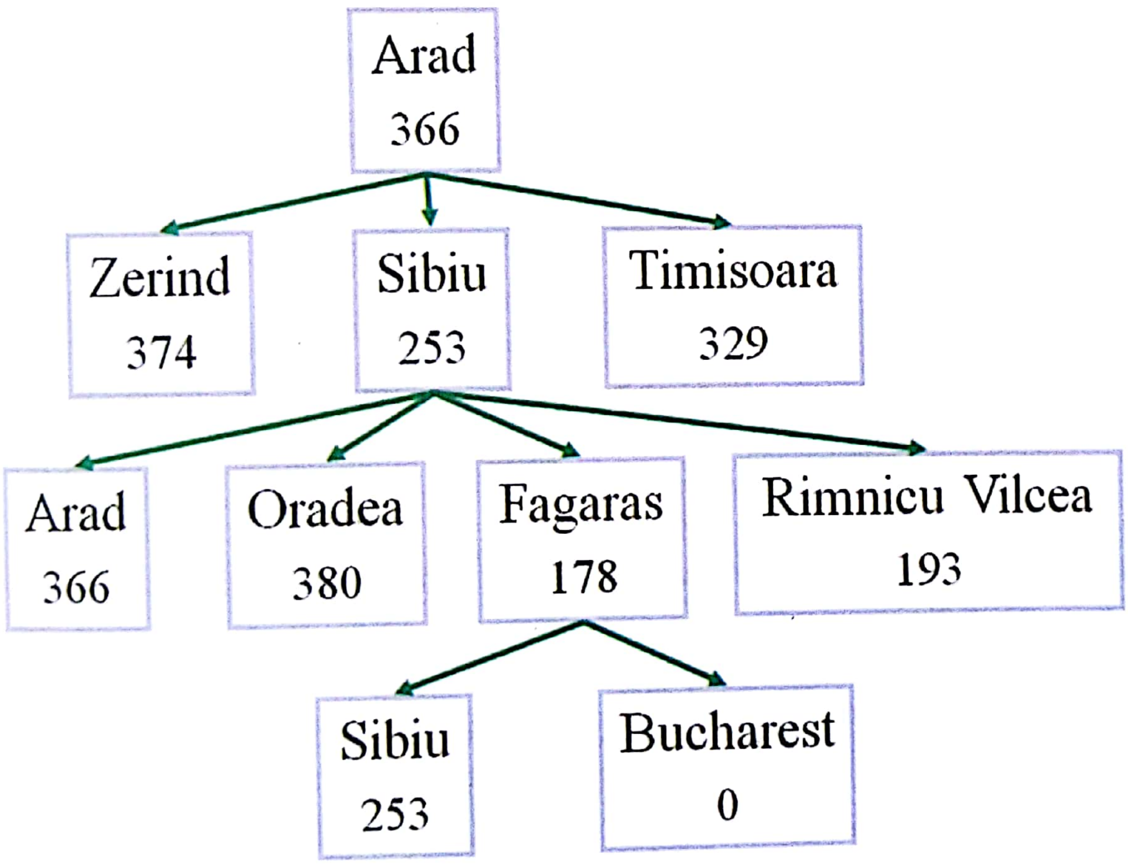
مثال: الانتقال من *Arad* إلى *Bucharest*

نستخدم في هذا المثال مسافة خط النظر كتقدير للكلفة المتبقية من مدينة إلى المدينة الهدف.



Straight-line distance to Bucharest	
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	178
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	98
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

ينتج عن تطبيق الخوارزمية شجرة البحث التالية:



مثال: لعبة التكوتان

ينتج عن تطبيق خوارزمية تسلق التلة مع اعتماد مسافة مانهاتن شجرة البحث التالية:

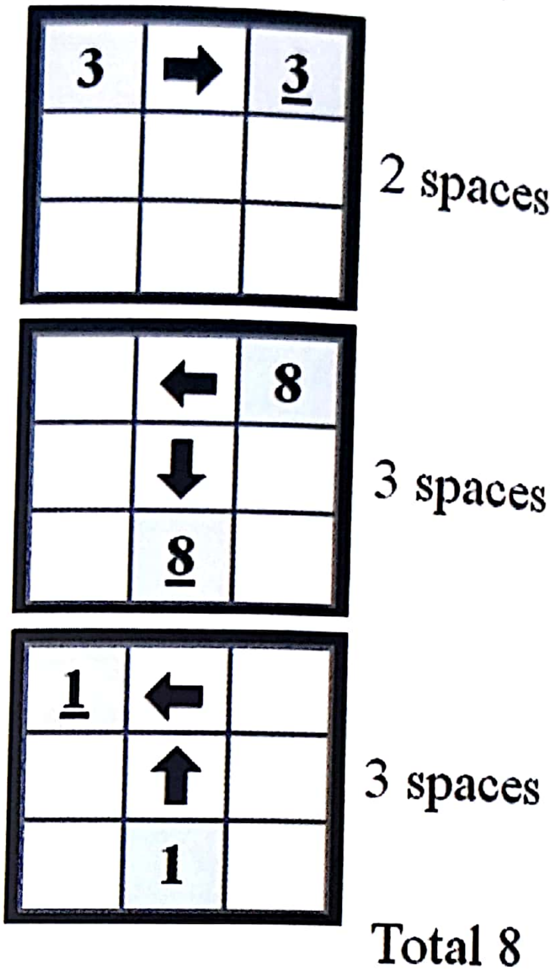
(لاحظ سرعة الوصول إلى الحل)

Current State

3	2	8
4	5	6
7	1	

Goal State

1	2	3
4	5	6
7	8	



4 - 9 - خوارزمية *A

تقوم هذه الخوارزمية بدمج كل من الخوارزميتين: البحث وفق الكلفة المنتظمة وخوارزمية تسلق التلة للوصول إلى خوارزمية كاملة وأمثليه وسريعة. نستخدم في هذه الخوارزميتين التابعين التاليين:

- g كلفة العقدة الحالية
- h تقدير كلفة الوصول إلى الهدف اعتباراً من العقدة الحالية

ونستخدم المحصلة:

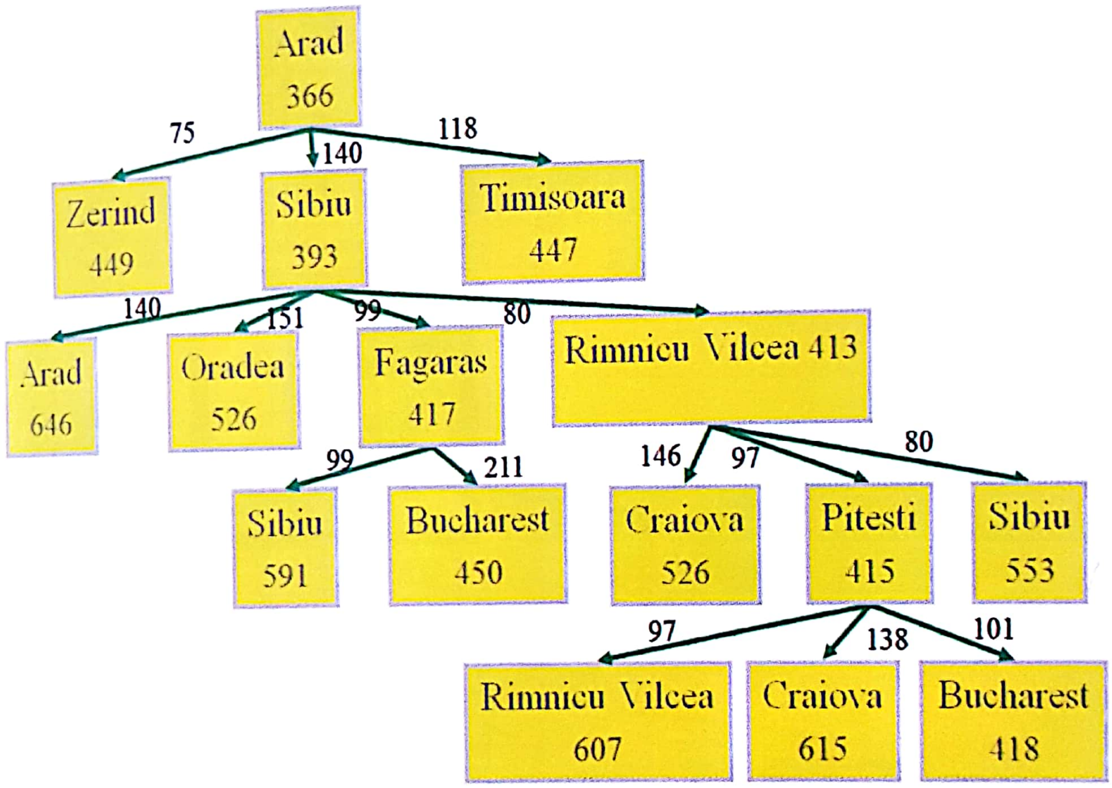
$$f(n) = g(n) + h(n)$$

لترتيب العقد الحدود وفق قيمة التابع f بشكل تصاعدي.

تكون هذه الخوارزمية أمثلية إذا كانت تقدير الكلفة للوصول إلى الهدف دائماً أصغر أو يساوي الكلفة الحقيقية.

مثال: الانتقال من *Arad* إلى *Bucharest*

ينتج عن تطبيق الخوارزمية شجرة البحث التالية:

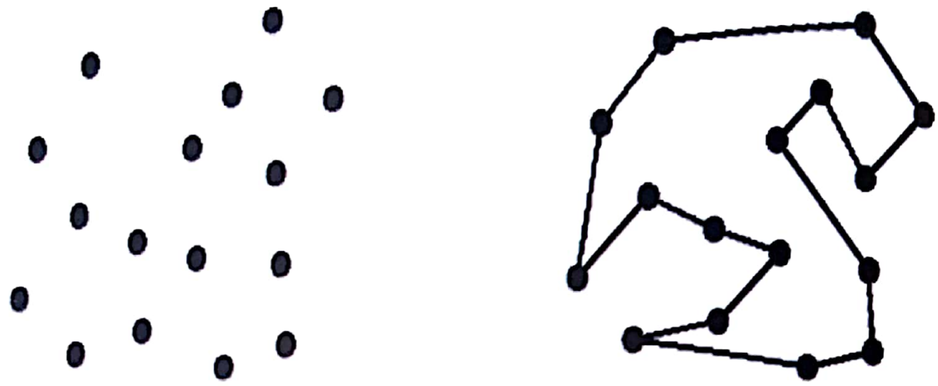


4 - 10 - مسألة البائع الجوال

بفرض أن لدينا مجموعة من العقد. نُعرّف المسافات بين هذه العقد.

والمطلوب:

إيجاد مسار ذو كلفة أصغرية وبحيث نزور كل عقدة مرة واحدة.



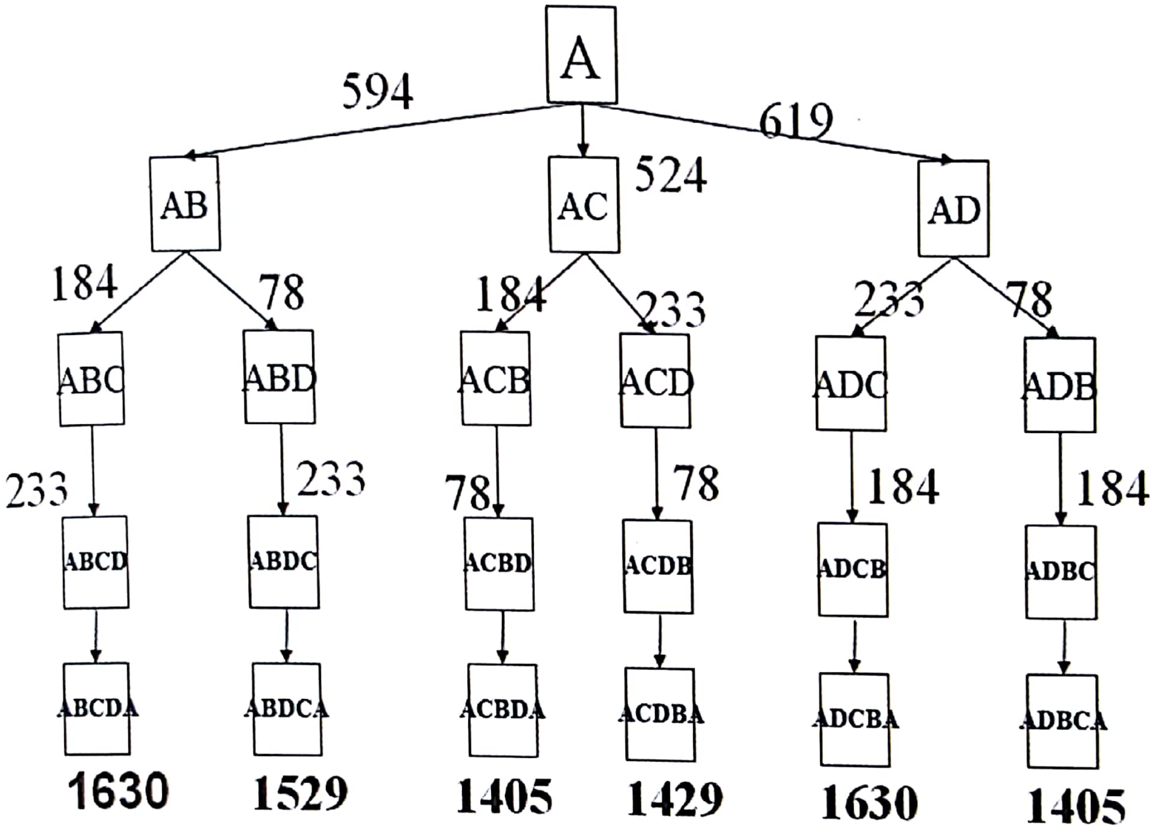
مثال:

نريد زيارة كل من المدن الخمس التالية وبكلفة أصغرية.

	Aberdeen	Brighton	Cardiff	Dover	Edinburgh
Aberdeen	0	594	524	619	127
Brighton	594	0	184	78	467
Cardiff	524	184	0	233	395
Dover	619	78	233	0	493
Edinburgh	127	467	395	493	0

وذلك بتطبيق الخوارزمية *A**.

ينتج عن تطبيق الخوارزمية من أجل *N=4* شجرة البحث التالية:



لاحظ أن عدد المسارات هو:

No of paths =(1- n)!

أي مثلاً:

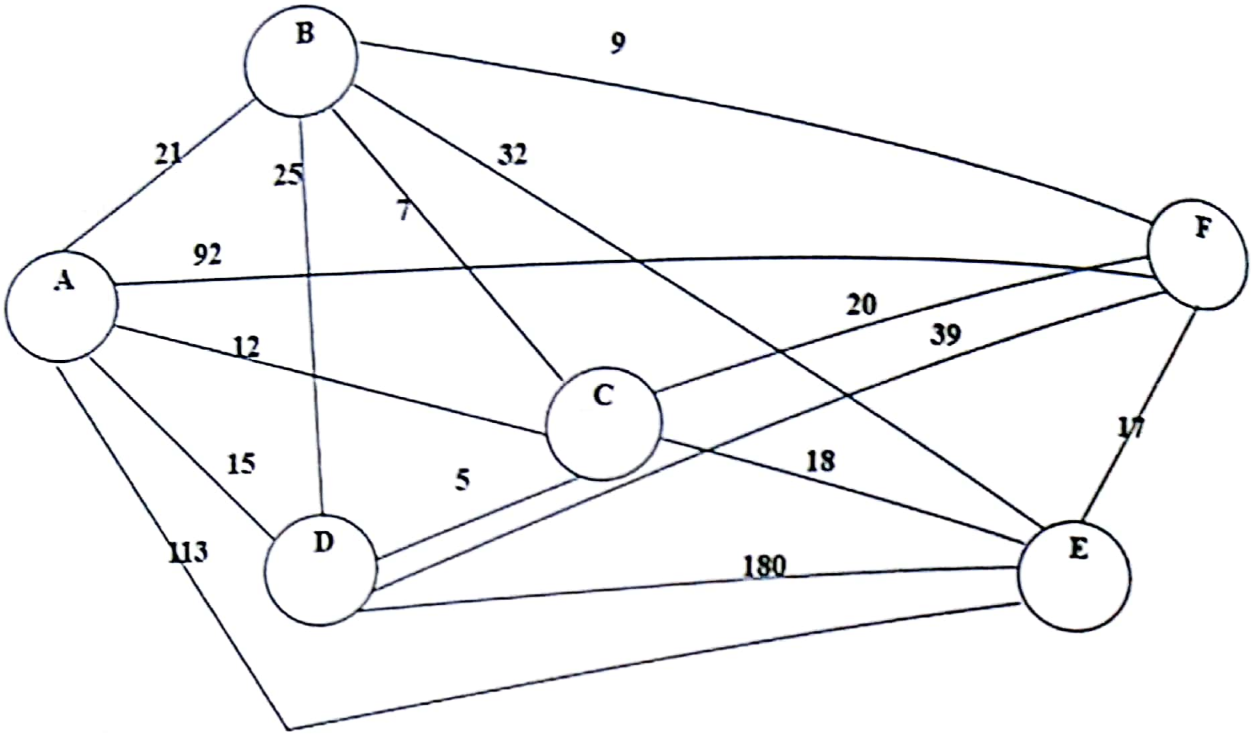
n ,4=p6=

n=5, p=24

n=10, p=362,880

مثال:

لنطبق الخوارزمية *A على المسألة التالية مع تجريبيات مختلفة:



التجريبيات

لنستخدم التجريبيات التالية:

H1=0

التجريبية1 = صفر

*H2 = No missed arcs * cost of minimal arc*

التجريبية2 = عدد الأقواس الناقصة * كلفة أصغر قوس

H3 = Sum of p most short arcs) p is No missed arcs(

التجريبية3 = مجموع أقصر p قوس (حيث p هو عدد الأقواس الناقصة)

H4 = Sum of p most short arcs connected to nodes that remained to be exited from

Ex: For H4(ABD)=36 Since we have to exit from D,C,E and F

التجريبية4 = مجموع أقصر *p* قوس مرتبط بالعقد المتبقي الخروج منها

H5 = Sum of p most short arcs connected to nodes that remained to be visited

Ex: For H4(ABD)=43 Since we have to visit from C ,E ,F and A.

التجريبية5 = مجموع أقصر *p* قوس مرتبط بالعقد الواجب الدخول فيها

تُعطي جميع التجريبيات السابقة الحل الأمثل إذ أن تقدير الكلفة في كل منها أصغر من الكلفة الحقيقية.

FBADCEF or FECDABF

Cost: 85

يُبين الجدول التالي عدد العقد المولدة وعدد العقد المطورة من أجل كل تجريبية:

Heuristic	Developed Nodes	Created Nodes
H1	88	159
H2	72	138
H3	62	115
H4	47	97
H5	69	133

في حال استخدام التجريبية التالية والتي لاتحقق شرط الأمثلية:

*H6: No of missed arcs * cost of average arc*

التجريبية6 = عدد الأقواس الناقصة * الكلفة الوسطية للأقواس

نحصل على الحل:

FEADCBF

Cost: 166

مع الكلف التالية:

Heuristic	Developed Nodes	Created Nodes
H6	9	22

وفي حال استخدام التجريبية التالية والتي لاتحقق شرط الأمثلية:

*H7: No of missed arcs * cost of max arc*

التجريبية7 = عدد الأقواس الناقصة * كلفة أكبر قوس

نحصل على الحل:

FBCDAEF

Cost: 166

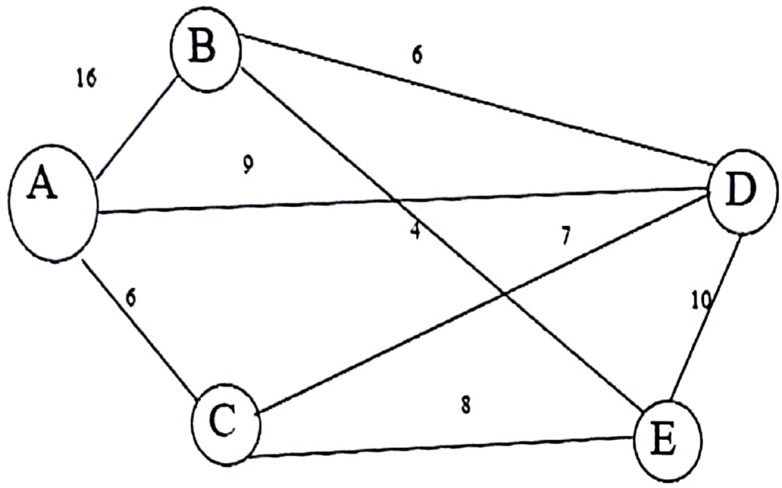
مع الكلف التالية:

Heuristic	Developed Nodes	Created Nodes
H7	13	17

4 - 11 - تمارين محلولة

1. ليكن البيان التالي وحيث المطلوب الانتقال من A إلى E.

ماذا سيكون الناتج باعتماد كل من الخوارزميات التالية: العمق أولاً، العرض أولاً، الكلفة المنتظمة.



الحل:

العمق أولاً	A, B, D, C, E
العرض أولاً	A, B, E
الكلفة المنتظمة	A, C, E

2. ماذا سيكون الناتج باستخدام خوارزمية تسلق التلة وخوارزمية A^* علماً بأن لدينا تقدير للمسافة بين كل عقدة والعقدة E كما يلي:

A	B	C	D	E
10	2	8	5	0

الحل:

تسلق التلة	A, B, E
A^*	A, C, E

4 - 12 - أسئلة متعددة الخيارات

3. تعطي خوارزمية البحث من النمط العمق أولاً *Depth-First Search*:

a. حل أمثلي دائماً فقط في مسألة البائع الجوال.

b. حل أمثلي دائماً فقط في مسائل الألعاب بلاعب واحد (مثل *8-puzzle*).

c. حل أمثلي دائماً في جميع مسائل البحث.

d. **ولا خيار من الخيارات الثلاث السابقة.**

4. تعطي خوارزمية البحث من النمط العرض أولاً *Breadth-First Search*:

a. حل أمثلي دائماً فقط في مسألة البائع الجوال.

b. **حل أمثلي دائماً فقط في مسائل الألعاب بلاعب واحد (مثل *8-puzzle*).**

c. حل أمثلي دائماً في جميع مسائل البحث.

d. ولا خيار من الخيارات الثلاث السابقة.

5. تعطي خوارزمية البحث من النمط البحث المنتظم *Uniform Search*:

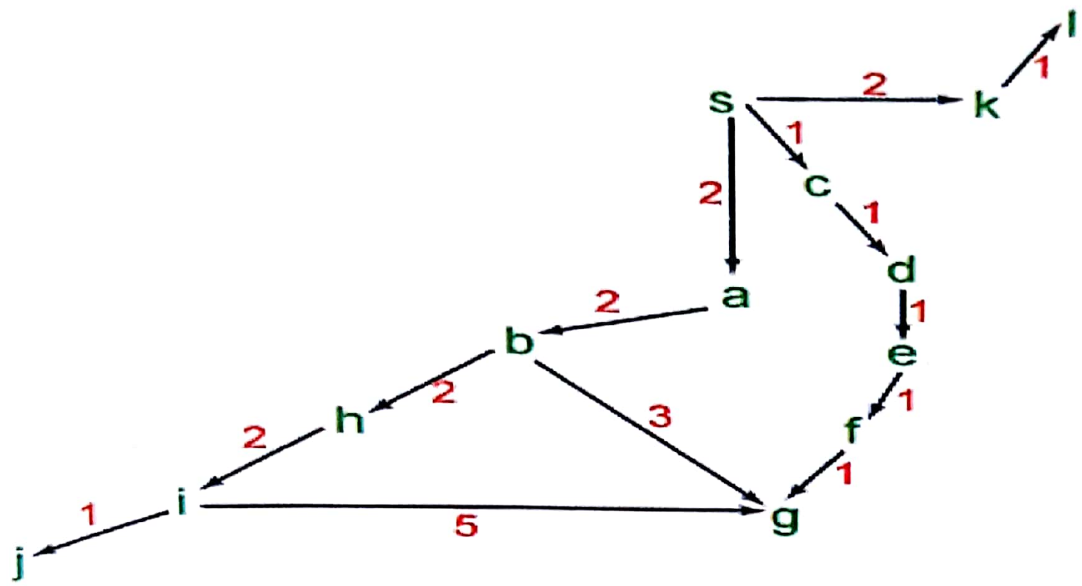
a. حل أمثلي دائماً فقط في مسألة البائع الجوال.

b. حل أمثلي دائماً فقط في مسائل الألعاب بلاعب واحد (مثل *8-puzzle*).

c. **حل أمثلي دائماً في جميع مسائل البحث.**

d. ولا خيار من الخيارات الثلاث السابقة.

6. ليكن لدينا مسألة البحث التالية من s إلى g:



مع قيمة التجريبية كما يلي:

h(a,2), h(b,3), h(c,4), h(d,3), h(e,2), h(f,1), h(g,0), h(h,4), h(i,5), h(j,6), h(k,5), h(l,6), h(s,4).

تكون كلفة المسار الناتج بتطبيق خوارزمية تسلق التلة *Hill Climbing*

a. 5

b. 7

c. 13

d. غير ذلك

ليكن لدينا اللعبة البسيطة التالية *(4-puzzle)* :

وليكن لدينا التجريبيتين التاليتين:

Start

-	1
3	2

Goal

1	2
-	3

مكانه.

$h1$ مسافة مانهاتن لوضع الفراغ في مكانه.

$h2$ عدد الخانات الموجودة في غير مكانها الصحيح (لا نحسب الفراغ).

مثلاً للحالة الابتدائية يكون: $h1=1, h2=3$

7. يكون عدد الحالات المختلفة التي سيتم توليدها حتى الوصول للهدف (بما فيها الحالة الابتدائية والنهائية) بتطبيق الخوارزمية *A مع التجريبية $h1$:

a. 7

b. 6

c. 5

d. ولا خيار مما سبق

8. يكون عدد الحالات المختلفة التي سيتم توليدها حتى الوصول للهدف (بما فيها الحالة الابتدائية والنهائية) بتطبيق الخوارزمية *A مع التجريبية $h2$:

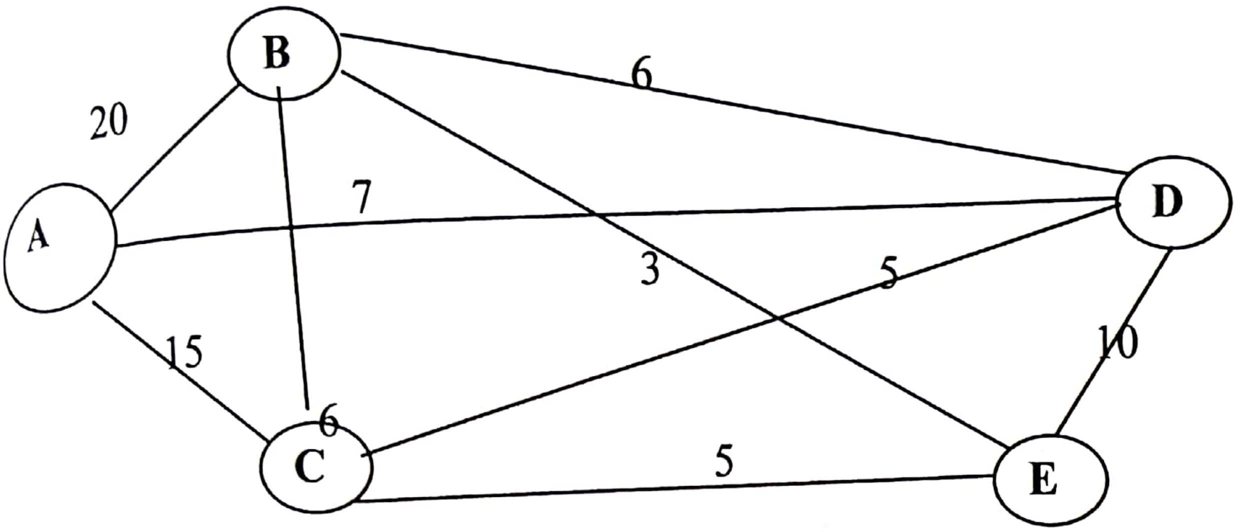
a. 4

b. 5

c. 6

d. ولا خيار مما سبق

ليكن البيان التالي والذي يُمثل الطرق الممكنة بين مجموعة من المدن والمسافات بينها. وليكن المطلوب الانتقال من المدينة A إلى المدينة E:



9. يعطي تطبيق خوارزمية البحث العرض أولاً *Breadth-First* الحل:

a. <u>*ABE*.</u>

b. *ABCE*.

c. *ACE*.

d. ولا خيار من الخيارات الثلاث السابقة.

10. يعطي تطبيق خوارزمية *A** مع تجريبية *h=0* الحل:

a. *ABE*.

b. *ACE*.

c. *ADE*.

d. <u>ولا خيار من الخيارات الثلاث السابقة.</u>

ليكن المطلوب الآن مسألة البائع الجوال على نفس البيان السابق والذي ينطلق من *A* ويعود إلى *A*:

11. يعطي تطبيق خوارزمية A^* مع تجريبية عدد الأقواس الناقصة * كلفة أكبر قوس الحل:

a. $ADBECA$.

b. $ADEBCA$.

c. **$ADCEBA$.**

d. $ADECBA$.

12. يعطي تطبيق خوارزمية A^* مع تجريبية عدد الأقواس الناقصة * كلفة أصغر قوس الحل:

a. **$ADBECA$.**

b. $ADEBCA$.

c. $ADCEBA$.

d. $ADECBA$.

13. تكون كلفة الحل الأمثل للمسألة:

a. 34.

b. **36.**

c. 38.

d. 40.

14. يوجد للمسالة بنفس الكلفة:

a. حل وحيد أمثل.

b. حلين أمثلين.

c. ثلاثة حلول أمثليه.

d. أربعة حلول أمثليه.

الفصل الخامس

مسائل الألعاب Games

نعالج الألعاب الحتمية مع لاعبين والتي تتميز:

- يتناوب اللاعبان باللعب، فيلعب اللاعب الأول، ثم الآخر، وهكذا دواليك.
- يعلم كل لاعب ماذا لعب اللاعب الآخر وما يُمكنه أن يلعب.
- تنتهي اللعبة بربح أحد اللاعبين (وخسارة اللاعب الآخر) أو التعادل.

مثال: لدينا بدايةً كومة مؤلفة من سبع ليرات، والهدف تقسيم هذه الكومة إلى عدة أكوام؛ على كل لاعب أن يقسم أحد الأكوام إلى كومتين غير متساويتين حصراً؛ الخاسر هو من لا يستطيع أن يلعب.

5 - 1 - شجرة اللعب

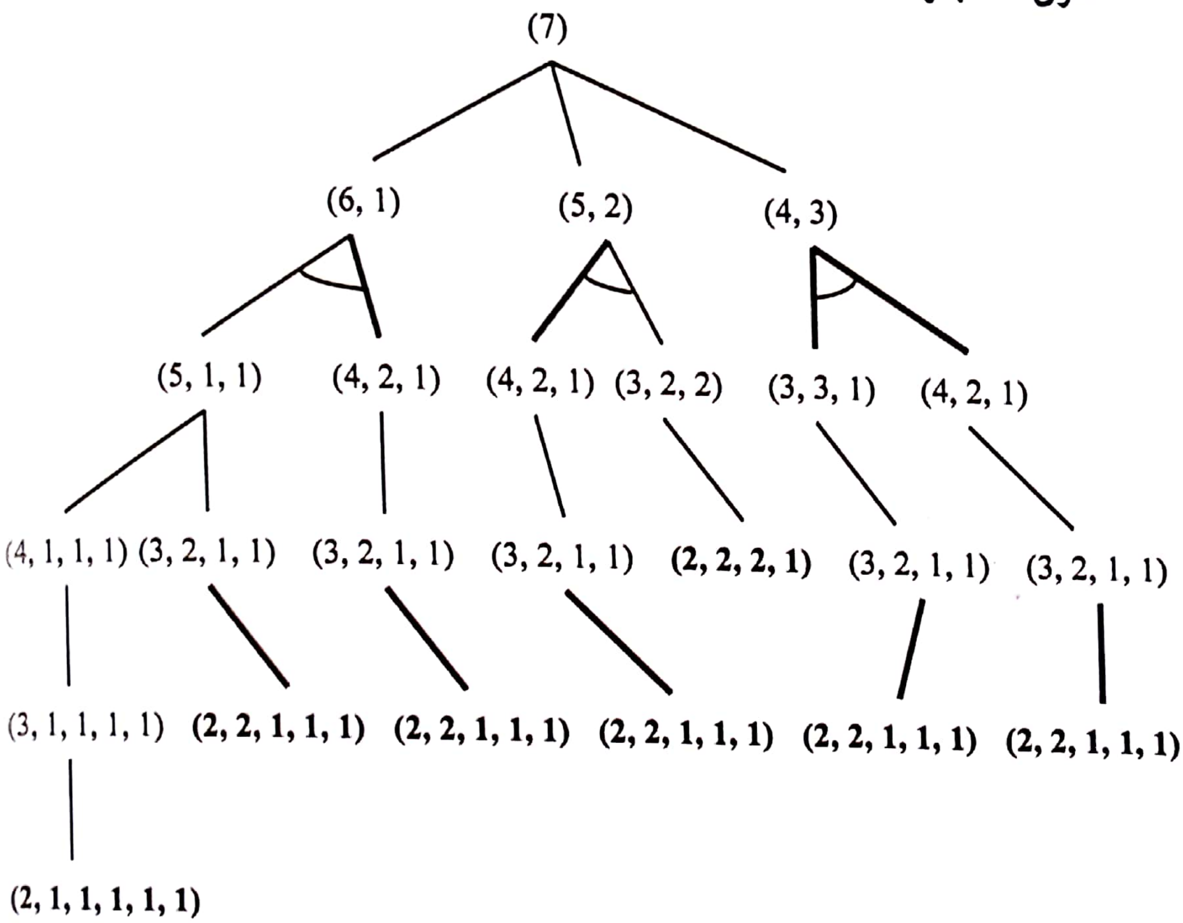
ليكن اللاعبان $J1$ و $J2$، وليكن $J1$ هو البادئ في اللعب. **تفيد قواعد اللعبة بإنشاء شجرة اللعب على النحو التالي:**

- يُمثّل الجذر (ذو المستوى 0) موضع البداية.
- تُمثّل العقد ذات المستوى الزوجي المواضع التي على اللاعب $J1$ أن يلعب فيها
- تُمثّل العقد ذات المستوى الفردي المواضع التي على اللاعب $J2$ أن يلعب فيها

• تُمثّل الأقواس الصادرة من عقدة ما مختلف الضربات الممكنة والتي يمكن لعبها انطلاقاً من الموضع الممثل بهذه العقدة، وذلك من قبل اللاعب المعني (*J1* أو *J2* حسب شفعية المستوى للعقدة زوجياً أو فردياً).

• تُمثّل الأوراق المواضع الفائزة، أو الخاسرة، أو التي بلا مخرج.

تكون شجرة اللعب للعبة السابقة:



5 - 2 - خوارزمية *MinMax*

نسمّي اللاعبين من الآن فصاعداً *MAX* و *MIN*. وستكون مهمتنا إيجاد "أفضل" حركة للاعب *MAX* (الصديق). لنفترض أن اللاعب *MAX* سيلعب أولاً، ومن ثم يلعب اللاعبان بالتناوب.

تتألف الخوارزمية من إجرائيتين تستدعي كل منهما الأخرى (عودية متصالبة):

- الإجرائية *Maxmin* والتي تُستدعى من أجل عقد الصديق.
- الإجرائية *MinMax* والتي تُستدعى من أجل عقد الخصم.

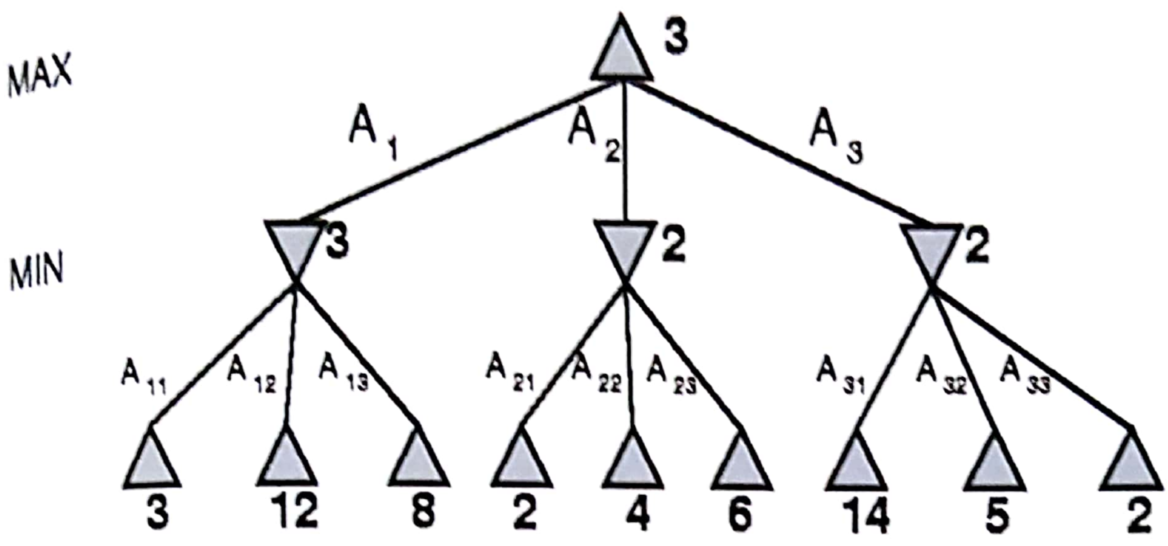
```
α ← Maxmin (R)
If  Leaf (R) Then
  α ← Evaluate (R)
Else
  α ← Max (Minmax (Succ1 (R)),
      Minmax (Succ2 (R)), ... Minmax (Succn (R)))
EndIf
```

```
β ← Minmax (R)
If  Leaf (R) Then
      β ← Evaluate (R)
Else
      β ← Min (Maxmin (Succ1 (R)),
    Maxmin (Succ2 (R)), ... Maxmin (Succn (R)))
EndIf
```

حيث:

- *Leaf(R)*: تُعيد *True* إذا كانت العقدة *R* ورقة.
- $Succ_i(R)$: تُعيد العقدة *i* الخَلَف لـ *R*.

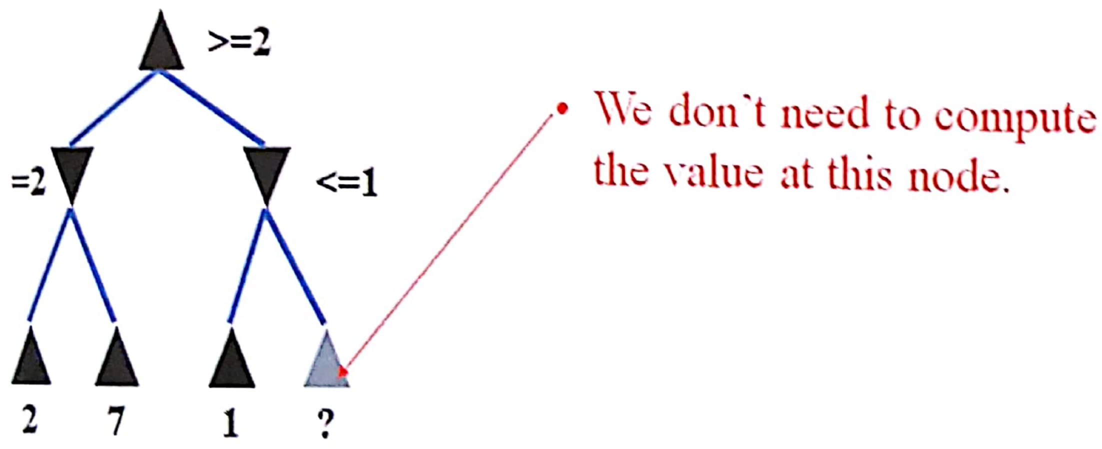
مثال:



5 - 3 - خوارزمية ألفا-بيتا

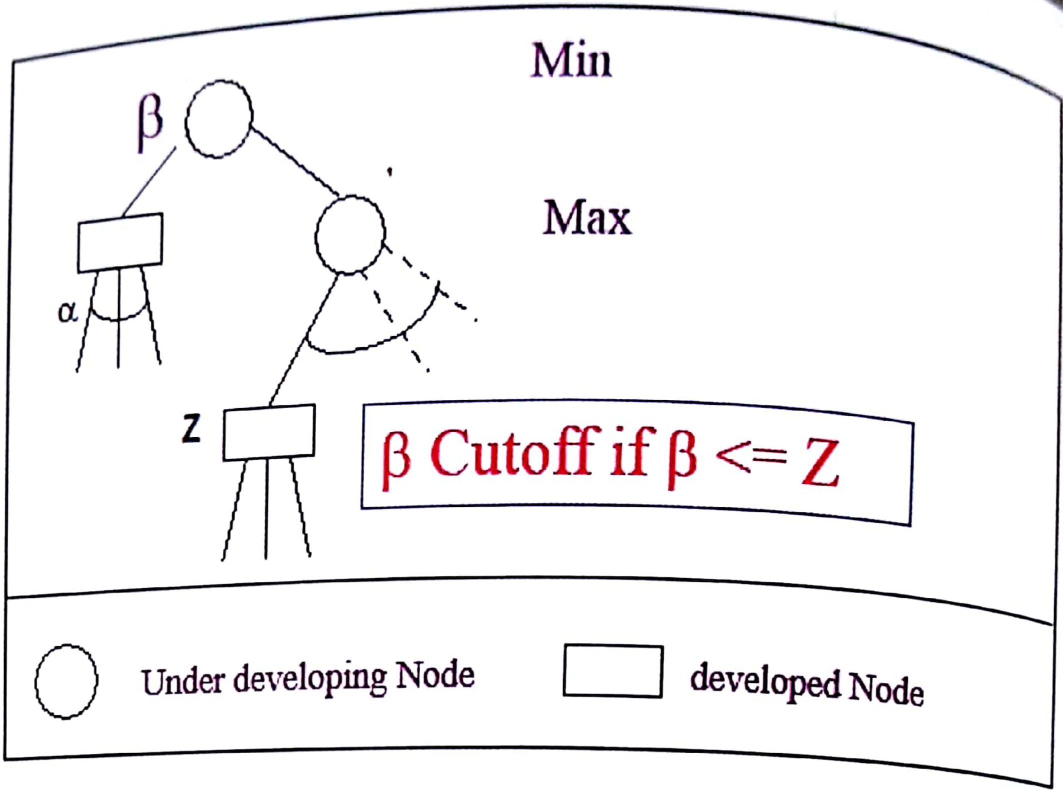
يُمكن تسريع وبشكل كبير أداء الخوارزمية *MinMax* السابقة باستخدام التشذيب α-β.

مثال:



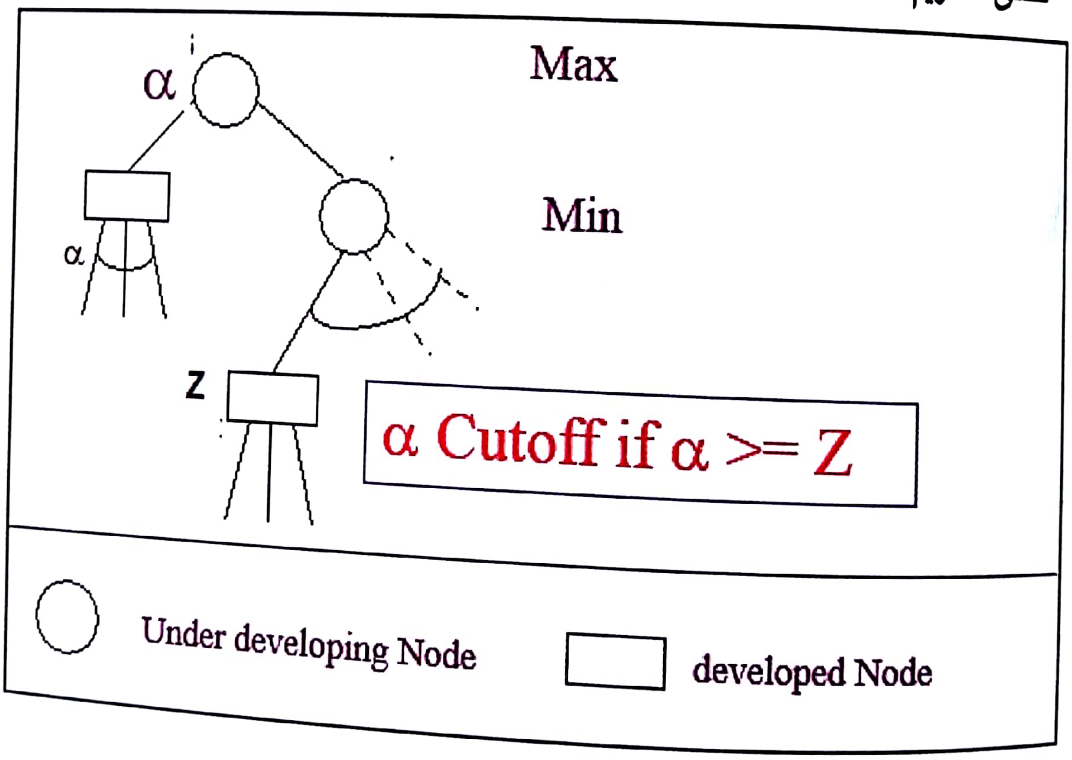
5 - 4 - القطع بيتا *Beta Cutoff*

عندما نكون في عقدة *Max* تحت عقدة *Min* (تمّ حساب بيتا لها) فيُمكن التوقف عن تقويم العقد المتبقية للعقدة *Max* بمجرد الحصول على تقويم *Z* لأحد أبناء العقدة وبحيث: $Z \leq \beta$



5 - 5 - القطع ألفا *Alpha Cutoff*

عندما نكون في عقدة *Min* تحت عقدة *Max* (تمّ حساب ألف لها) فيُمكن التوقف عن تقويم العقد المتبقية للعقدة *Min* بمجرد الحصول على تقويم *Z* لأحد أبناء العقدة وبحيث: $\alpha \geq Z$

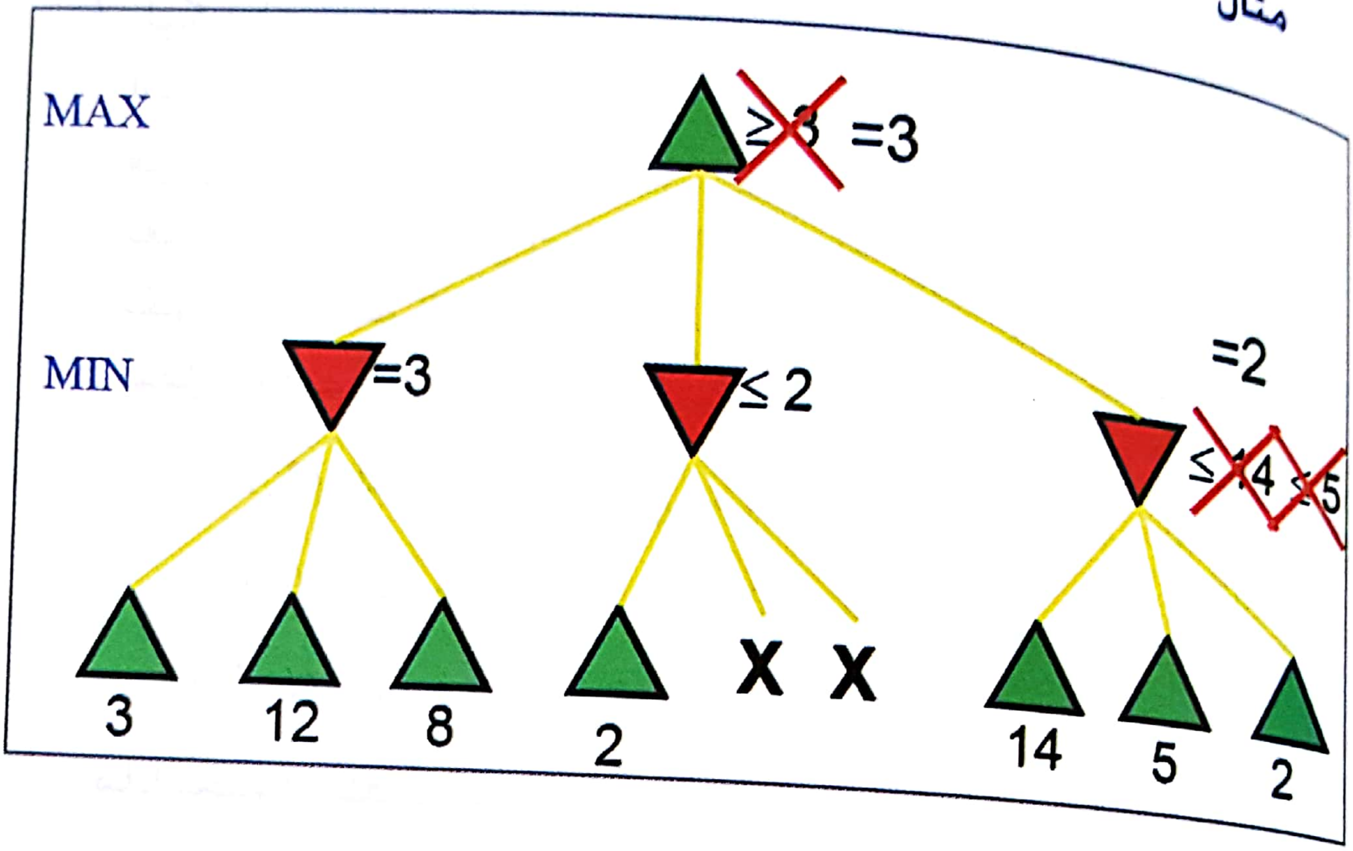


تتألف الخوارزمية من تابعين يستدعي كل منهما الآخر (عودية متصالبة):

- تابع يُستدعى عند عقد الصديق *Max*
- تابع يُستدعى عند عقد الخصم *Min*
- في البداية (α=-∞, β=∞).

```
function MAX-value (n, alpha, beta)
if n is a leaf node then return f(n);
for each child m of n do
alpha :=max{alpha, MIN-value(m, alpha, beta)};
if alpha >= beta then return beta /* pruning */
end{do}
return alpha
function MIN-value (n, alpha, beta)
if n is a leaf node then return f(n);
for each child m of n do
beta :=min{beta, MAX-value(m, alpha, beta)};
if beta <= alpha then return alpha /* pruning */
end{do}
return beta
```

مثال


MAX
MIN
$\cancel{\geq 3}$ =3
=3
≤ 2
=2
$\cancel{\leq 14}$ $\cancel{\leq 5}$
3
12
8
2
X
X
14
5
2

5 - 6 - أسئلة متعددة الخيارات

1. ليكن لدينا اللعبة التالية بثلاثة لاعبين: يوجد ثلاثة أبراج يحوي البرج الأول قرص والثاني قرص والثالث قرصين. يُمكن للاعب أن يأخذ قرص أو أكثر إنما من نفس البرج. اللاعب الذي يربح هو اللاعب الذي بعد أن يلعب لا يبقى أي قرص في الأبراج أي يجعل اللعبة (0,0,0). تكون إذاً الحالة الابتدائية هي (1,1,2) ومنها يُمكن للاعب الأول أن ينقل اللعبة إلى أحد الحالات الثلاث التالية:

(2 ,1 ,0) يأخذ قرص من برج يحوي قرص

(0 ,1 ,1) يأخذ قرصين من البرج الذي يحوي قرصين

(1 ,1 ,1) يأخذ قرص من البرج الذي يحوي قرصين

ماذا يجب أن يلعب اللاعب الذي يبدأ كي يربح في النهاية:

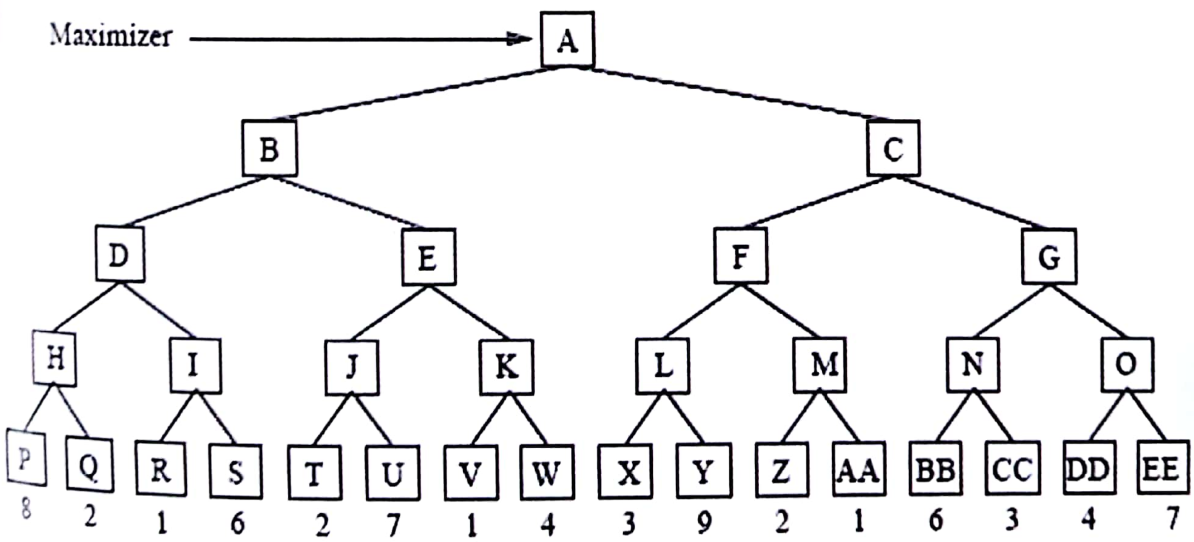
a. يأخذ قرص من برج يحوي قرص

b. يأخذ قرص من البرج الذي يحوي قرصين

c. يأخذ قرصين من البرج الذي يحوي قرصين

d. و لا خيار مما سبق

2. لتكن شجرة اللعب التالية وحيث أن اللاعب MAX هو الذي يبدأ:



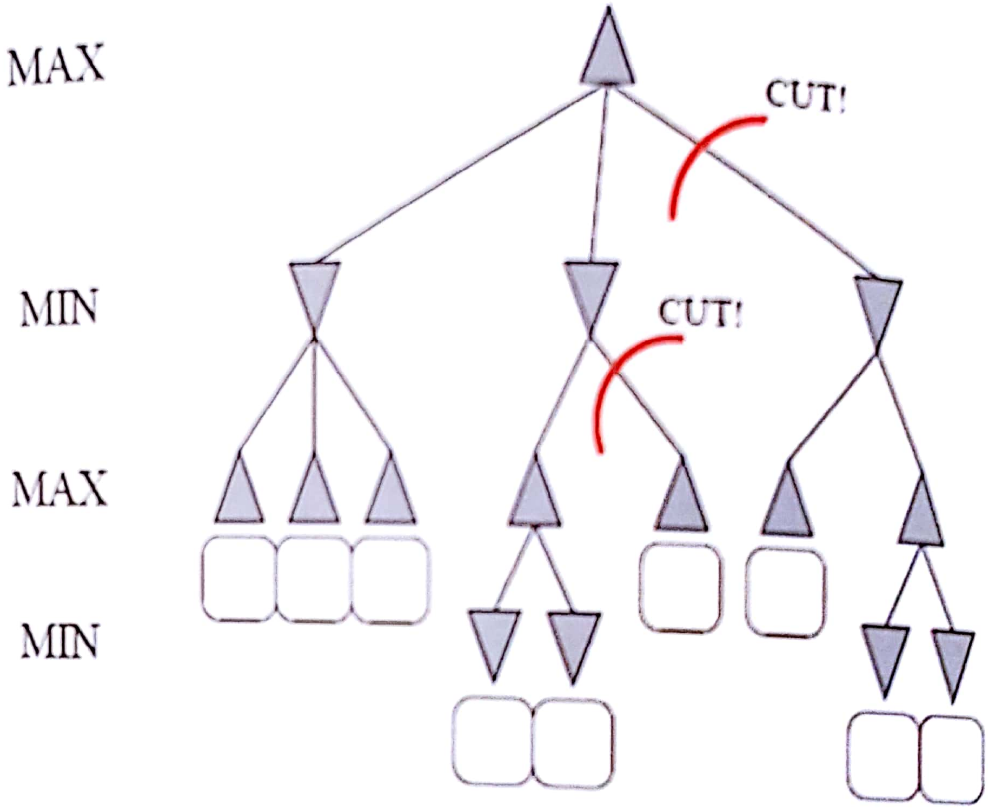
تقوم الخوارزمية ألفا-بيتا بحساب قيم الأوراق التالية

a. P, Q, R, T, U, V, X, Y, Z, BB, CC, DD

b. <u>P, Q, R, T, U, X, Y, Z, BB, CC</u>

c. P, Q, R, T, U, V, X, Y, Z, BB

3. لتكن شجرة اللعب التالية والمطلوب إيجاد قيم للأوراق كي نستطيع تحقيق القطعين المبينين



a. نضع قيم متساوية في جميع الأوراق.

b. نضع قيم بترتيب تنازلي لعقد *Max* و بترتيب تصاعدي لعقد *Min*.

c. نضع قيم بترتيب تنازلي لعقد *Min* و بترتيب تصاعدي لعقد *Max*.

d. <u>ولا خيار من الخيارات الثلاث السابقة.</u>

4. حدّد العبارة الصحيحة

a. لا تختلف قيم عقد *Min* و *Max* في خوارزميتي *Min-Max* و *Alpha-Beta*

b. لا يؤثر ترتيب الأوراق على فعالية خوارزمية *Alpha-Beta*

c. لا تتأثر فعالية الخوارزمية *Alpha-Beta* بمستوى الخصم

d. ولا خيار مما سبق

5. ليكن لدينا اللعبة التالية: يوجد كومة من n قطعة. يقوم اللاعب حين يأتي دوره بأخذ على الأقل قطعة واحدة وعلى الأكثر m قطعة من الكومة. يربح اللاعب الذي يأخذ آخر قطعة من الكومة. حدّد الجملة الصحيحة فيما يلي

a. يربح اللاعب الذي يبدأ باللعب دائما

b. يخسر اللاعب الذي يبدأ باللعب دائما

c. يربح اللاعب الذي يبدأ إذا كانت n من مضاعفات m

d. يربح اللاعب الذي يبدأ إذا كانت n ليست من مضاعفات $m+1$

مثال لفهم اللعبة: إذا كانت مثلا $n=20$ و $m=3$ فإن اللاعب الذي يبدأ يمكن أن يأخذ قطعة أو اثنتين أو ثلاثة.

يربح اللاعب الذي حين يأتي دوره يكون في الكومة قطعة (فيأخذها) أو قطعتين (فيأخذهما) أو ثلاثة قطع (فيأخذهم).