دانشگاه صنعتی امیرکبیر
دانشکده مهندسی کامپیوتر و فناوری اطلاعات

# عامل‌های منطقی

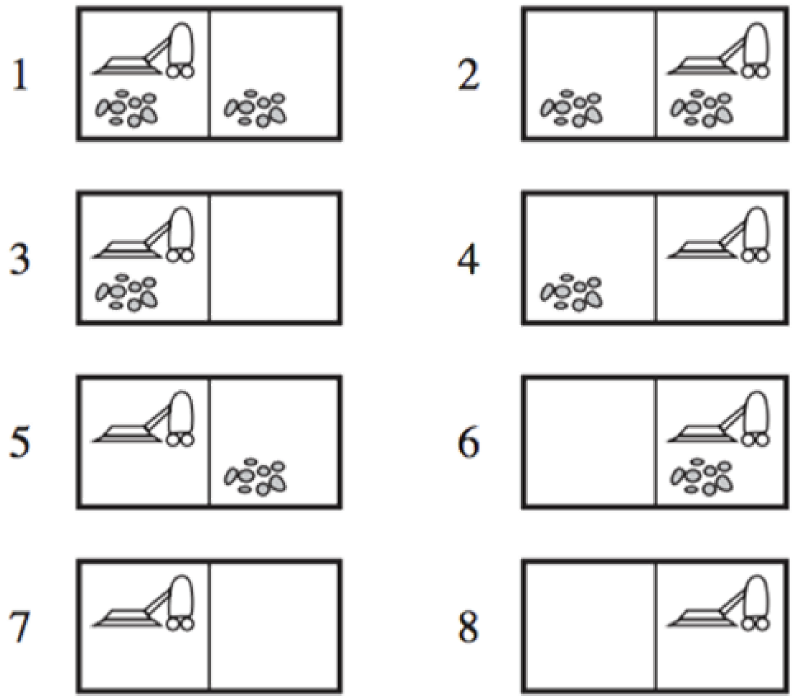
**«هوش مصنوعی: یک رهیافت نوین»، فصل ۷**

**ارائه‌دهنده: سیده فاطمه موسوی**

**نیم‌سال دوم ۱۴۰۰–۱۳۹۹**

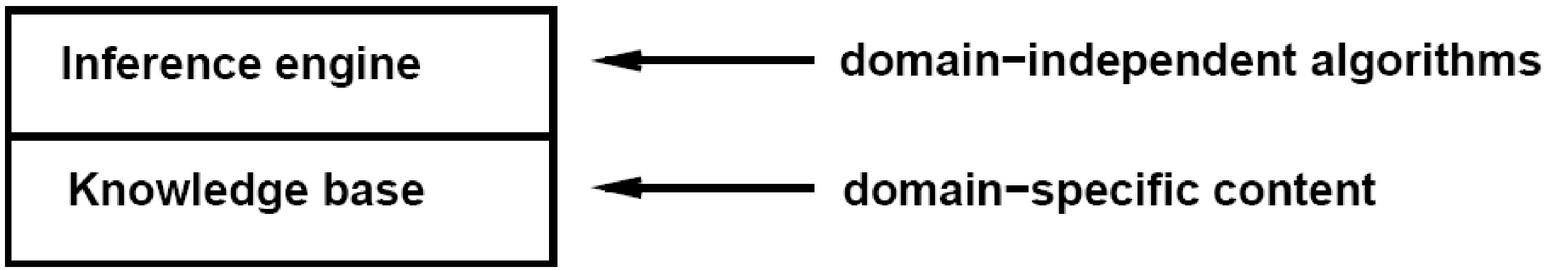
# رئوس مطالب

- عامل‌های مبتنی‌بر دانش
- محیط وامپوس
- منطق – مدل‌ها و ایجاب
- منطق گزاره‌ای (بولین)
- هم‌ارزی، اعتبار و ارضاپذیری
- قوانین استنتاج و اثبات تئوری
  - رزولوشن
  - زنجیره استنتاج رو به جلو
  - زنجیره استنتاج رو به عقب




$$L_A^0 \vee L_B^0 \vee D_A^0 \vee D_B^0$$

# عامل‌های مبتنی‌بر دانش




- جزء اصلی یک عامل مبتنی‌بر دانش، پایگاه دانش (Knowledge Base) است.
- پایگاه دانش = مجموعه ای از جملات (حقایق و قوانین) در زبان بازنمایی دانش
  - به جملاتی که حقیقتی را در مورد دنیای عامل اظهار می‌کنند axiom یا حقیقت می‌گویند.
- پایگاه دانش می‌تواند شامل دانش پیش‌زمینه (background) باشد یا شامل دانشی باشد که عامل به مرور یاد می‌گیرد.

# عامل‌های مبتنی‌بر دانش ...

- بر روی پایگاه دانش دو عمل اصلی انجام می‌شود:
  - TELL: جمله‌ای را به پایگاه دانش اضافه می‌کند.
  - ASK: اطلاعاتی را از KB استخراج می‌کند. (پرس‌وجو از دانسته‌ها)
- هر دو عمل فوق می‌توانند شامل استنتاج (Inference) باشند.
  - یعنی به‌دست آوردن جملات جدید از قدیم
- استنتاج باید از این الزام اساسی پیروی کند که هرگاه سوالی از پایگاه دانش پرسیده شود، پاسخ باید از آن‌چه که قبلاً به پایگاه دانش گفته شده (TELL) نتیجه گردد.
  - فرایند استنتاج نباید از خودش مفهوم جدیدی تولید نماید.

# عامل‌های مبتنی‌بر دانش ...

• طرح کلی برنامه عامل مبتنی‌بر دانش:

**function** KB-AGENT(*percept*) **returns** an *action*
  **persistent**: $KB$, a knowledge base
    $t$, a counter, initially 0, indicating time

  TELL($KB$, MAKE-PERCEPT-SENTENCE($percept, t$))
  $action \leftarrow$ ASK($KB$, MAKE-ACTION-QUERY($t$))
  TELL($KB$, MAKE-ACTION-SENTENCE($action, t$))
  $t \leftarrow t + 1$
  **return** *action*

# عامل‌های مبتنی‌بر دانش ...

- طرح کلی برنامه عامل مبتنی‌بر دانش:

```
function KB-AGENT(percept) returns an action
   persistent: KB, a knowledge base
               t, a counter, initially 0, indicating time

   TELL(KB, MAKE-PERCEPT-SENTENCE(percept, t))
   action ← ASK(KB, MAKE-ACTION-QUERY(t))
   TELL(KB, MAKE-ACTION-SENTENCE(action, t))
   t ← t + 1
   return action
```

$$D_A^0 \wedge Suck^0 \rightarrow \sim D_A^1$$
$$D_B^0 \wedge Suck^0 \rightarrow \sim D_B^1$$
$$L_A^0 \wedge Right^0 \rightarrow L_B^1 \wedge \sim L_A^1$$
$$L_B^0 \wedge Right^0 \rightarrow L_B^1$$
$$L_A^0 \wedge Left^0 \rightarrow L_A^1$$
$$L_B^0 \wedge Left^0 \rightarrow L_A^1 \wedge \sim L_B^1$$
$$\dots$$

# عامل‌های مبتنی‌بر دانش ...

- طرح کلی برنامه عامل مبتنی‌بر دانش:

**function** KB-AGENT(*percept*) **returns** an *action*
  **persistent**: *KB*, a knowledge base
      *t*, a counter, initially 0, indicating time

  TELL(*KB*, MAKE-PERCEPT-SENTENCE(*percept*, *t*))
  *action* ← ASK(*KB*, MAKE-ACTION-QUERY(*t*))
  TELL(*KB*, MAKE-ACTION-SENTENCE(*action*, *t*))
  $t \leftarrow t + 1$
  **return** *action*

$$L_A^0$$
$$D_A^0$$

$$D_A^0 \wedge Suck^0 \rightarrow \sim D_A^1$$
$$D_B^0 \wedge Suck^0 \rightarrow \sim D_B^1$$
$$L_A^0 \wedge Right^0 \rightarrow L_B^1 \wedge \sim L_A^1$$
$$L_B^0 \wedge Right^0 \rightarrow L_B^1$$
$$L_A^0 \wedge Left^0 \rightarrow L_A^1$$
$$L_B^0 \wedge Left^0 \rightarrow L_A^1 \wedge \sim L_B^1$$
$$\ldots$$

# عامل‌های مبتنی‌بر دانش ...

- طرح کلی برنامه عامل مبتنی‌بر دانش:

**function** KB-AGENT(*percept*) **returns** an *action*
    **persistent**: *KB*, a knowledge base
                *t*, a counter, initially 0, indicating time

    TELL(*KB*, MAKE-PERCEPT-SENTENCE(*percept*, *t*))
    *action* ← ASK(*KB*, MAKE-ACTION-QUERY(*t*))
    TELL(*KB*, MAKE-ACTION-SENTENCE(*action*, *t*))
    $t \leftarrow t + 1$
    **return** *action*

$$L_A^0$$
$$D_A^0$$
$$suck^0$$

$$D_A^0 \wedge Suck^0 \rightarrow {\sim}D_A^1$$
$$D_B^0 \wedge Suck^0 \rightarrow {\sim}D_B^1$$
$$L_A^0 \wedge Right^0 \rightarrow L_B^1 \wedge {\sim}L_A^1$$
$$L_B^0 \wedge Right^0 \rightarrow L_B^1$$
$$L_A^0 \wedge Left^0 \rightarrow L_A^1$$
$$L_B^0 \wedge Left^0 \rightarrow L_A^1 \wedge {\sim}L_B^1$$
$$\ldots$$

# عامل‌های مبتنی‌بر دانش ...

- طرح کلی برنامه عامل مبتنی‌بر دانش:

```
function KB-AGENT(percept) returns an action
   persistent: KB, a knowledge base
               t, a counter, initially 0, indicating time

   TELL(KB, MAKE-PERCEPT-SENTENCE(percept, t))
   action ← ASK(KB, MAKE-ACTION-QUERY(t))
   TELL(KB, MAKE-ACTION-SENTENCE(action, t))
   t ← t + 1
   return action
```

$$L_A^0$$
$$D_A^0$$
$$suck^0$$
$$\sim D_A^1$$
$$D_A^0 \wedge Suck^0 \rightarrow \sim D_A^1$$
$$D_B^0 \wedge Suck^0 \rightarrow \sim D_B^1$$
$$L_A^0 \wedge Right^0 \rightarrow L_B^1 \wedge \sim L_A^1$$
$$L_B^0 \wedge Right^0 \rightarrow L_B^1$$
$$L_A^0 \wedge Left^0 \rightarrow L_A^1$$
$$L_B^0 \wedge Left^0 \rightarrow L_A^1 \wedge \sim L_B^1$$
$$\ldots$$

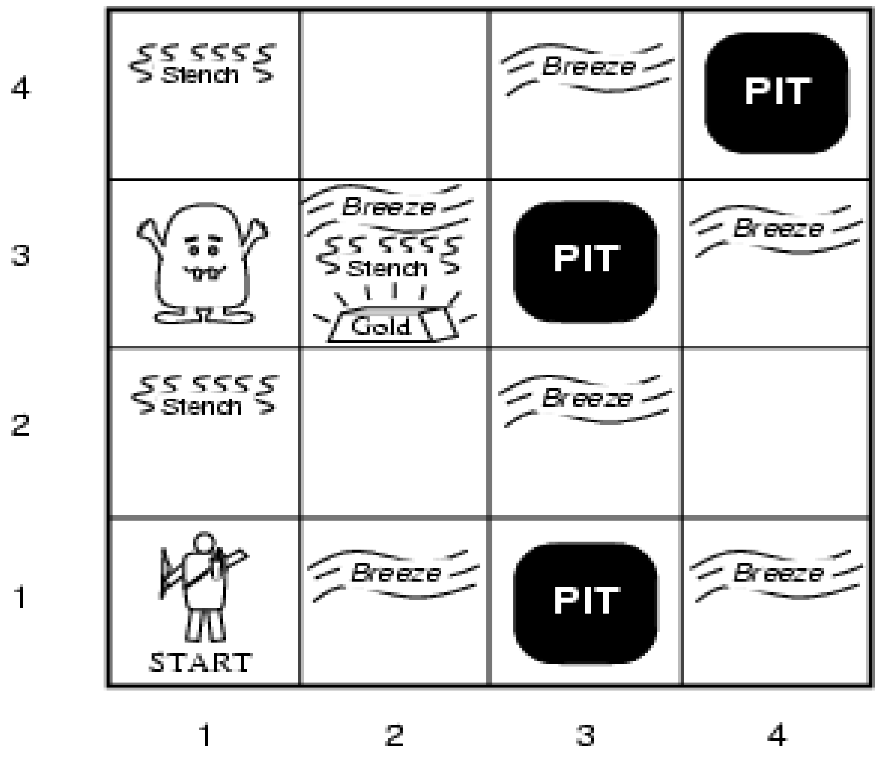
# دو سطح بررسی عامل مبتنی‌بر دانش

- سطح دانش: عامل چه چیزی می‌داند، بدون توجه به چگونگی پیاده‌سازی
- سطح پیاده‌سازی: چه ساختمان‌داده‌هایی برای ذخیره دانش استفاده می‌کند و از چه الگوریتم‌هایی برای استنتاج استفاده می‌کند.

# دو روش ساخت یک عامل مبتنی‌بر دانش

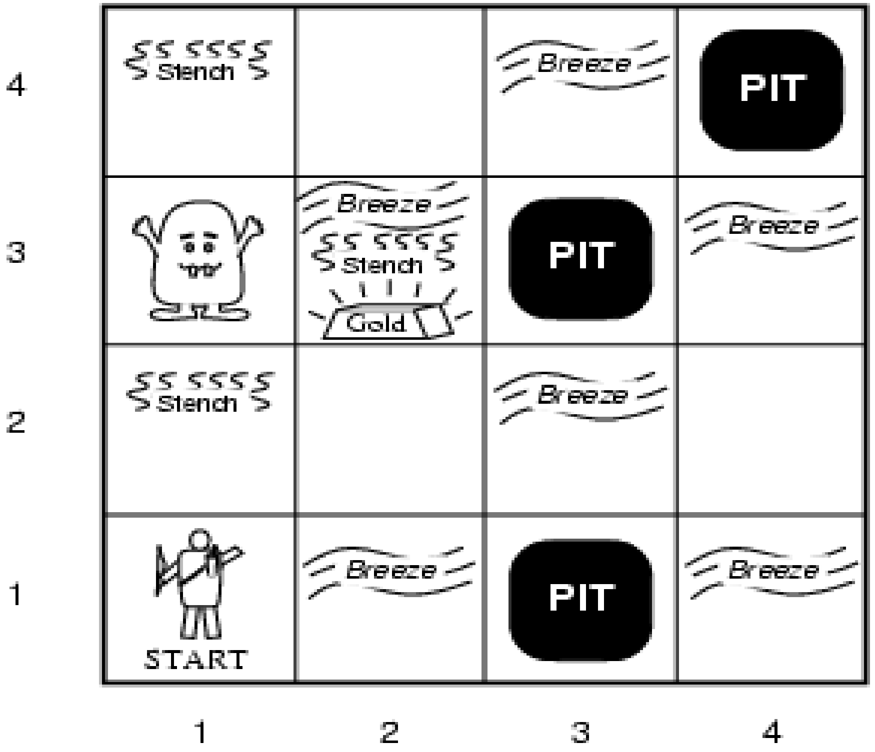
- روش اعلانی (declarative)
  - طراح عامل هر آن‌چه را عامل برای انجام وظایفش نیاز دارد یک به یک هر کدام به صورت یک جمله به عامل می‌گوید (TELL)
  - در زمان اجرا عامل با انجام استدلال و استنتاج بر روی این دانش تصمیم می‌گیرد چه کاری انجام دهد.
  - مزیت: امکان استفاده از دانش به شیوه‌هایی که طراح عامل پیش‌بینی نمی‌کرده است.
- روش رویه‌ای (procedural)
  - طراح نحوه رسیدن به هر نتیجه خاص را به صورت یک رویه کد می‌کند.
  - مزیت: سرعت اجرای بیشتر
- یک عامل موفق می‌تواند از ترکیب هر دو روش رویه‌ای و توصیفی ساخته شده باشد.

# دنیای وامپوس (Wumpus world)



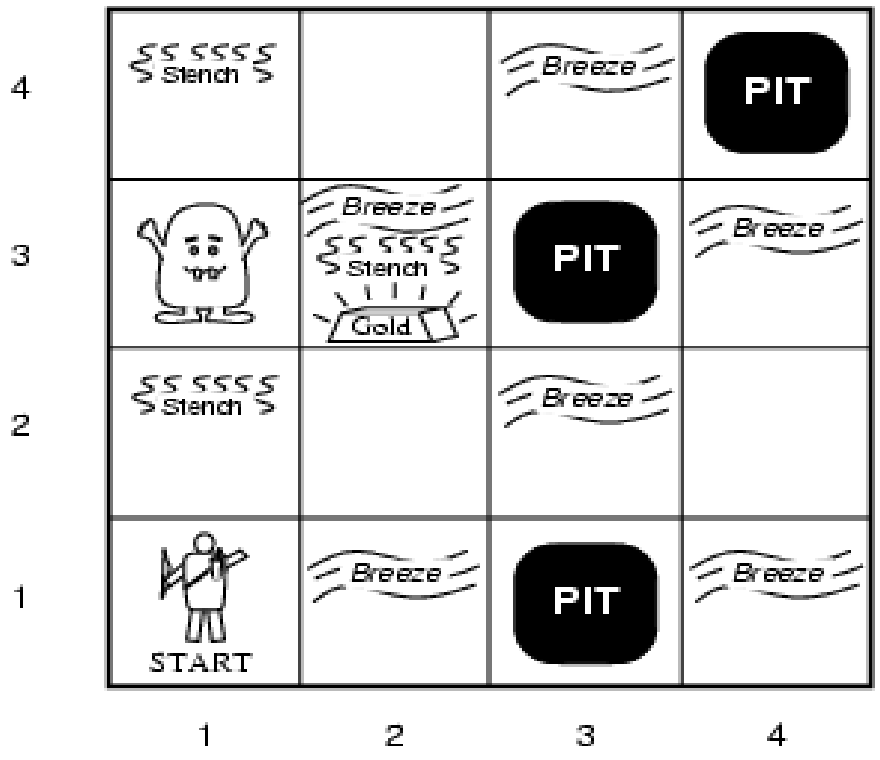

- معیار کارایی
  - طلا ۱۰۰۰ + ، مرگ ۱۰۰۰−
  - ۱− به ازای هر قدم و ۱۰− به ازای استفاده از تیر
- محیط
  - یک خانه مشبک ۴×۴
  - عامل همواره از مربع [1,1] و جهت روبه راست شروع می‌کند.
  - مکان طلا و وامپوس به‌طور تصادفی و در خانه‌ای غیر از خانه اول می‌باشد.
  - هر خانه به جز خانه اول با احتمال 0.2 گودال است.

# دنیای وامپوس (Wumpus world)




- اقدام‌ها:
  - حرکت مستقیم، ۹۰ درجه چرخش به راست یا چرخش به چپ.
    - حرکت به خانه‌ی دارای وامپوس و یا گودال منجر به مرگ می‌شود.
    - حرکت رو به جلو زمانی که دیوار وجود دارد، بی‌اثر است.
  - اقدام Grab، برای برداشتن طلا در همان خانه است.
  - اقدام Shoot، تیر را در جهتی که عامل ایستاده است شلیک می‌کند.
    - تیر تا زمانی که به وامپوس یا دیوار بخورد به راه خود ادامه می‌دهد.
    - عامل فقط یک تیر دارد، بنابراین فقط اولین اقدام Shoot اثرگذار است.
  - اقدام Climb برای زمانی که عامل از مربع [۱,۱] بیرون می‌آید.

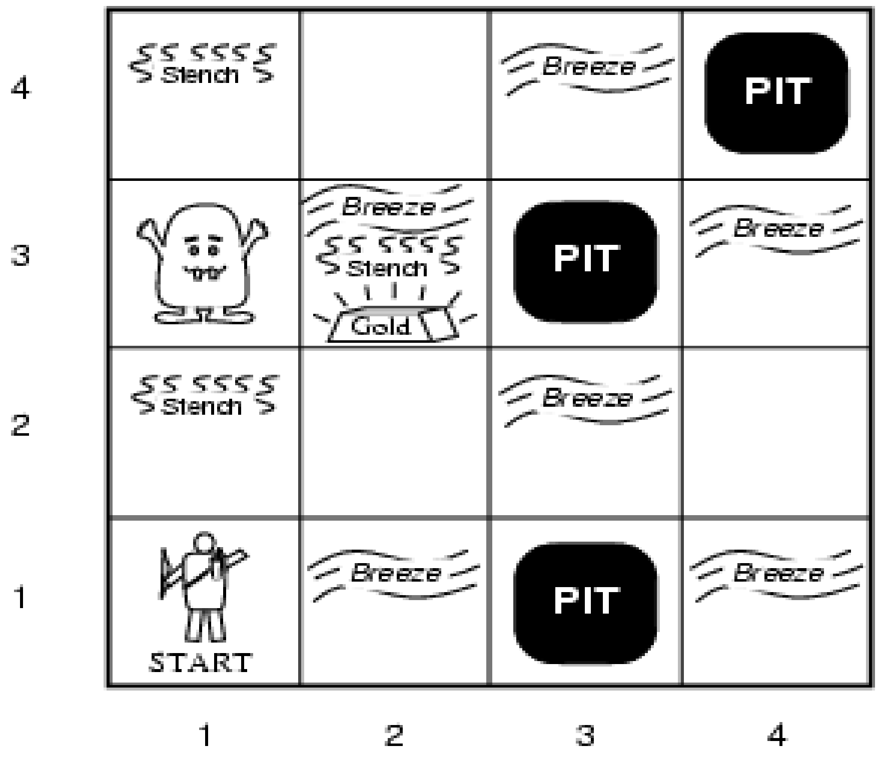
# دنیای وامپوس (Wumpus world)




- حس‌گرها:
  - در خانه‌ای که وامپوس است و خانه‌های همسایه غیرقطری، عامل بوی بد دریافت می‌کند.
  - در خانه‌های همسایه غیرقطری گودال، عامل نسیم خنکی را حس می‌کند.
  - در خانه‌ای که طلا وجود دارد، عامل درخشش طلا را می‌بیند.
  - هنگامی که عامل به دیوار بر می‌خورد، ضربه‌ای حس خواهد کرد.
  - هنگامی که وامپوس کشته می‌شود، فریادی اندوهناک می‌کشد که صدایش در همه جای غار شنیده می‌شود.
- ادراکات با یک بردار پنج تایی بو، نسیم، درخشش، ضربه و جیغ نشان داده می‌شود.

[Stench, Breeze, None, None, None]

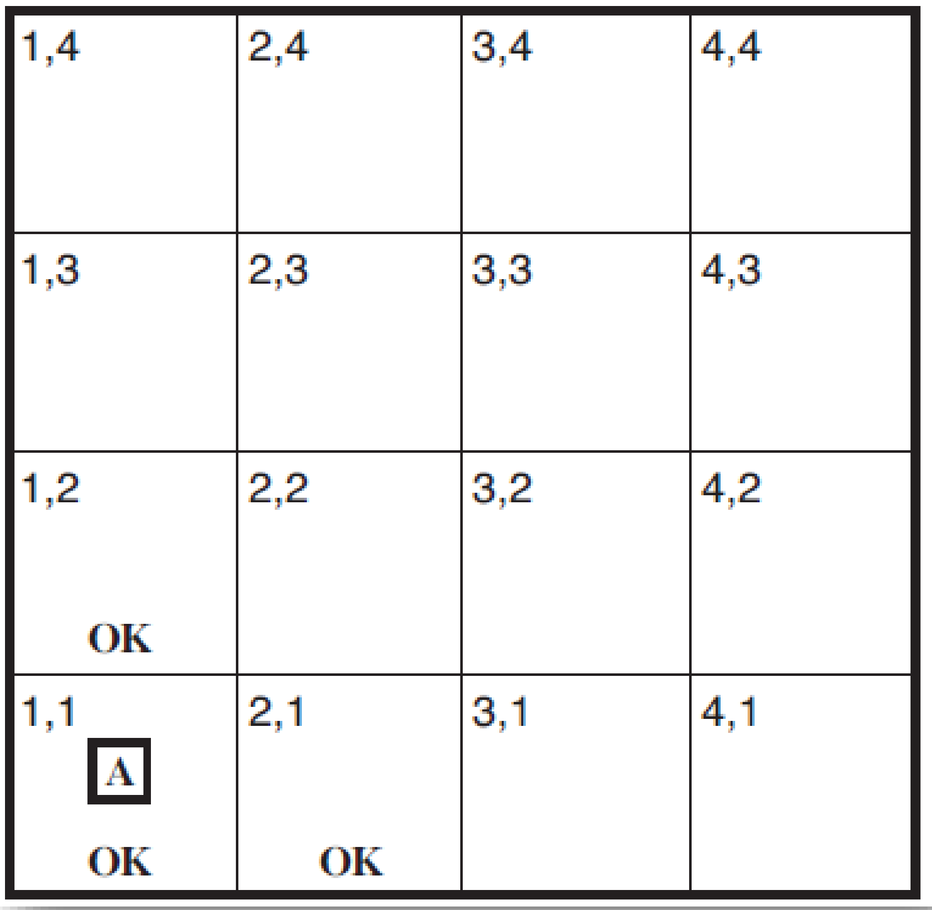
# مشخصات دنیای وامپوس




- مشاهده‌پذیر یا نیمه مشاهده‌پذیر؟
- قطعی یا غیرقطعی؟
- اپیزودیک یا مرحله‌ای؟
- ایستا یا پویا؟
- گسسته یا پیوسته؟
- تک عاملی یا چند عاملی؟

# مثال دنیای وامپوس

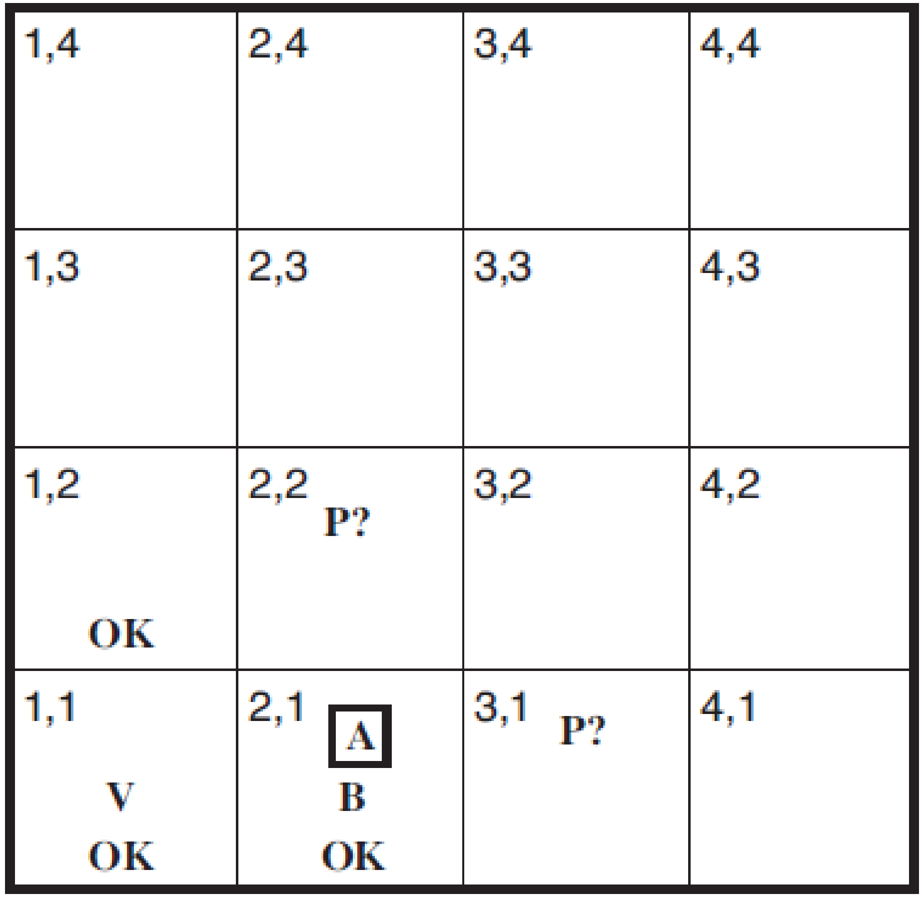
- **A** = *Agent*
- **B** = *Breeze*
- **G** = *Glitter, Gold*
- **OK** = *Safe square*
- **P** = *Pit*
- **S** = *Stench*
- **V** = *Visited*
- **W** = *Wumpus*




[None, None, None, None, None]

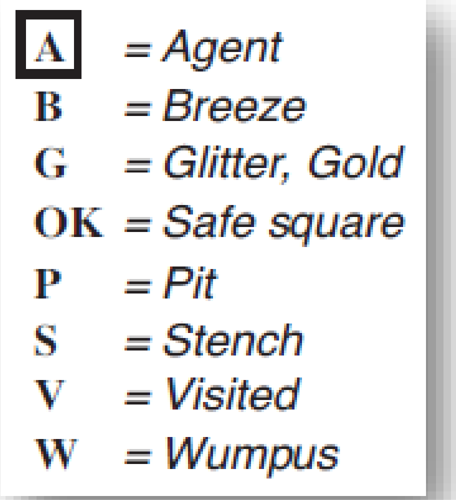
# مثال دنیای وامپوس

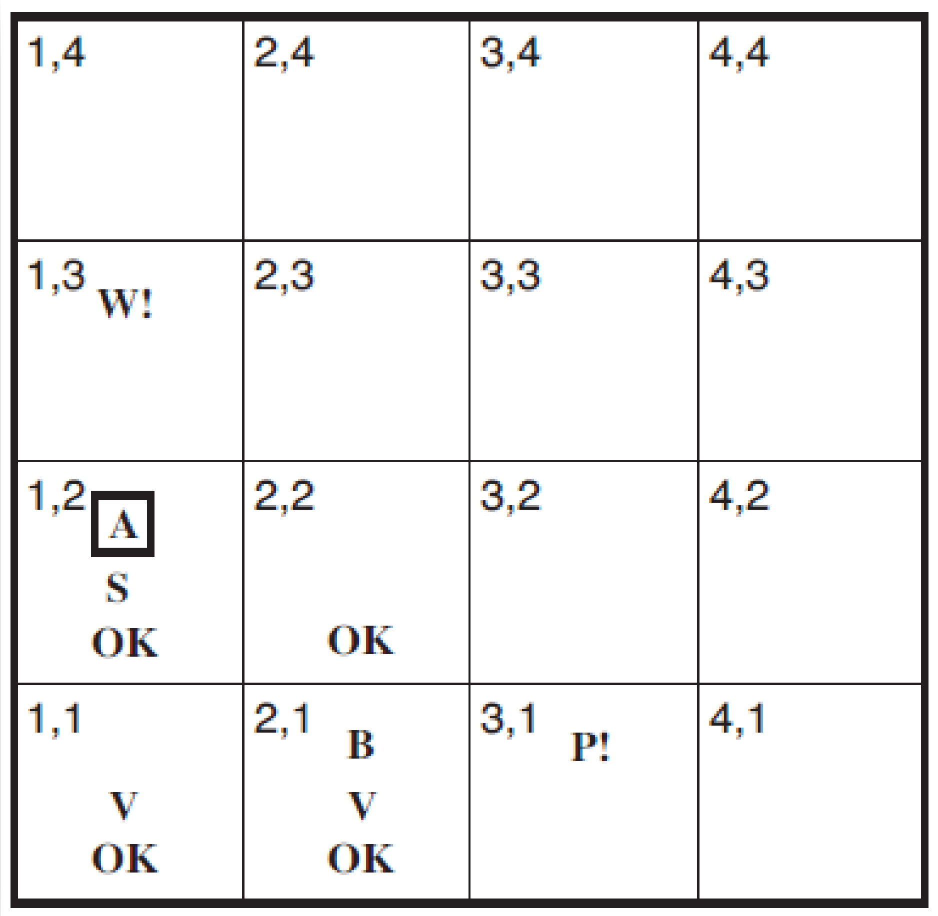
A = *Agent*
B = *Breeze*
G = *Glitter, Gold*
OK = *Safe square*
P = *Pit*
S = *Stench*
V = *Visited*
W = *Wumpus*




[None, Breeze, None, None, None]

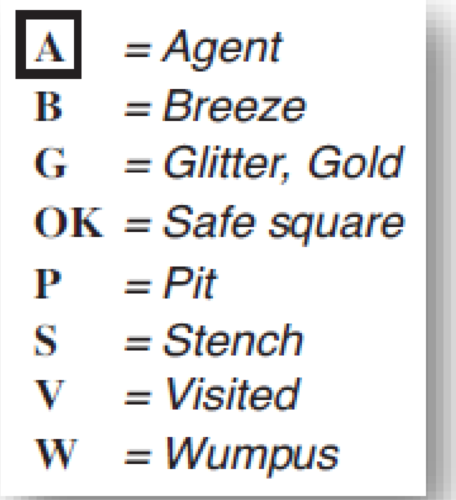
# مثال دنیای وامپوس

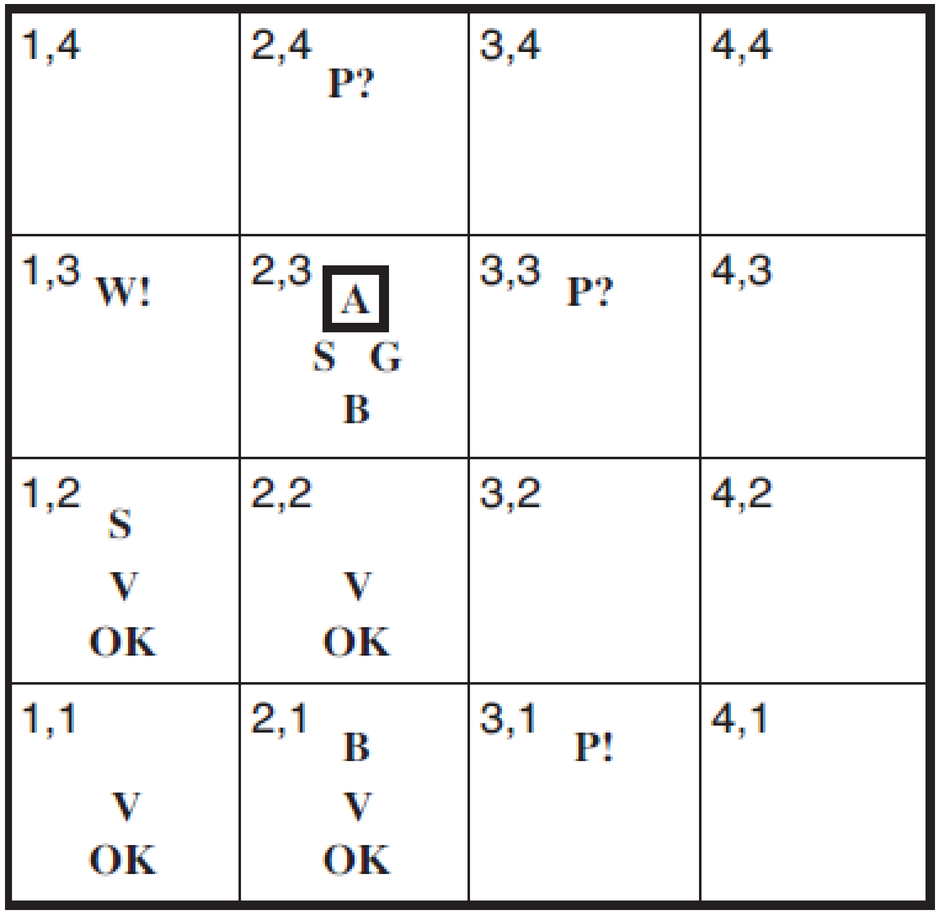



[Stench, None, None, None, None]

# مثال دنیای وامپوس




[Stench, Breeze, Glitter, None, None]

# منطق

- منطق، یک زبان رسمی برای بازنمایی دانش است به‌طوری‌که بتوان از آن نتیجه‌گیری نمود.
- نحو (Syntax) ساختار جملات زبان را تعریف می‌کند.
  - تمامی جملات خوش فرم(Well-formed) را مشخص می‌کند.
- معنا (Semantic) معنای جملات را تعریف می‌کند. درستی یک جمله را نسبت به هر دنیای ممکن تعریف می‌کند.
- مثال: زبان ریاضی
  - $x + 2 \geq y$ جمله است.
  - $x2 + y \geq$ جمله نیست.
  - جمله $x + 2 \geq y$ در دنیایی با $x = 7, y = 1$ درست و در دنیایی با $x = 0, y = 6$ نادرست است.

# مدل (Model)

- منطق‌دانان به هر دنیای ممکن یک مدل (مفاهیم انتزاعی ریاضی) می‌گویند.
- توجه: درستی یک جمله را نسبت به یک مدل می‌سنجیم، یعنی تعیین می‌کنیم آیا آن جمله به ازای مقادیری که در آن مدل به سمبل‌ها داده شده است درست است یا خیر.
  - مثال: دنیایی با x مرد و y زن و جمله $x+y=4$ را درنظر بگیرید.
  - مدل‌های ممکن: همه انتساب‌های عدد حقیقی به x و y هستند.
  - هر یک از این انتساب‌ها ارزش درستی جمله را تعیین می‌کنند.
    - جمله برای مدل‌هایی درست است که در آن مجموع تعداد افراد ۴ گردد مثلا $\{x=1, y=3\}$.

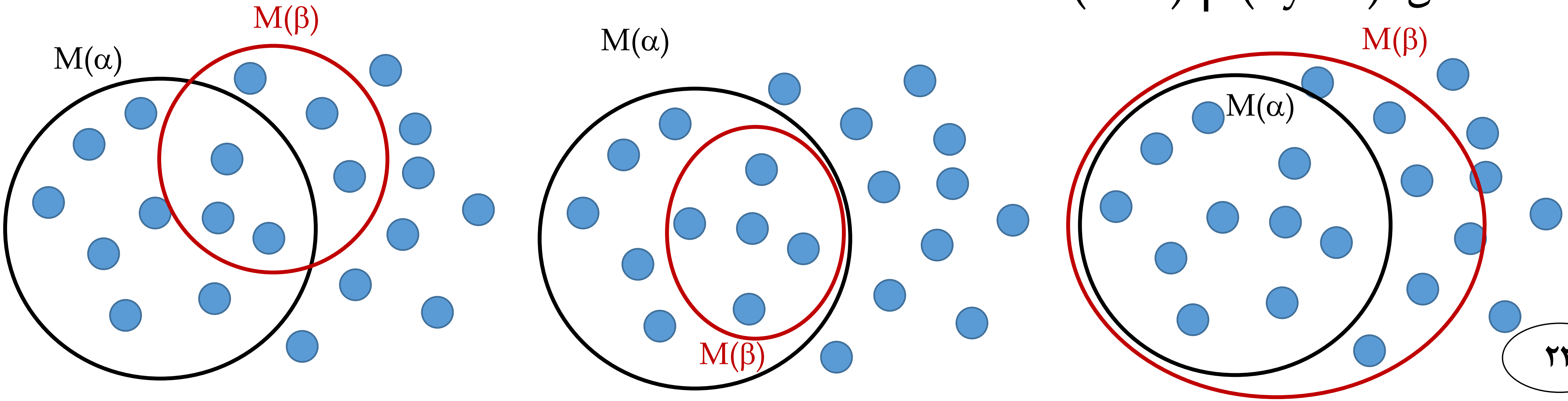
# مدل (Model) ...

- "$m$ مدلی از $\alpha$ است" یعنی "جمله $\alpha$ در مدل $m$ درست است". به‌عبارت دیگر یعنی جمله $\alpha$ به‌ازای مقادیری که مدل $m$ به سمبل‌ها می‌دهد برابر true می‌شود.
- If a sentence α is true in model m, we say that m satisfies α or sometimes m is a model of α.

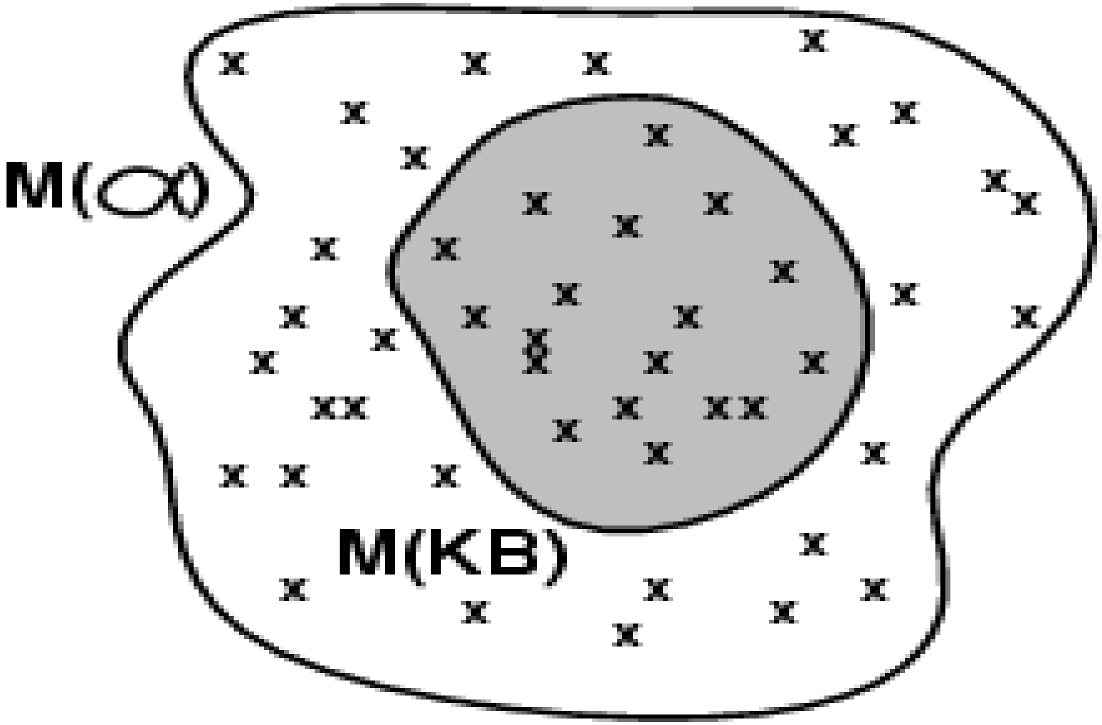
- یک جمله می‌تواند مدل‌های زیادی داشته باشد. مجموعه تمام مدل‌های جمله $\alpha$ را با $M(\alpha)$ نشان می‌دهیم.
  - به‌عبارت دیگر $M(\alpha)$ مجموعه تمام دنیاهایی است که $\alpha$ در آن‌ها درست است.
  - $m \in M(\alpha)$ if α is true in model $m$

# استدلال منطقی: ایجاب (Entailment)

- ایجاب منطقی میان جملات یعنی یک جمله به‌طور منطقی از جمله دیگر استنباط شود.
- می‌گوییم جمله $\alpha$ جمله $\beta$ را ایجاب می‌کند $\alpha \models \beta$ اگر و تنها اگر در هر مدلی که $\alpha$ در آن درست باشد، $\beta$ نیز در آن درست باشد.
  - $\alpha \models \beta$ if and only if $M(\alpha) \subseteq M(\beta)$
- مثال: $(x=0) \models (xy=0)$



# استدلال منطقی: ایجاب (Entailment)




- پایگاه دانش را می‌توان به‌صورت مجموعه‌ای از جملات و یا یک جمله که بیان‌کننده‌ی تمامی جملات است درنظر گرفت.
  - پایگاه دانش در مدل‌هایی false است که با آن‌چه که عامل می‌داند در تناقض باشد.
- رابطه‌ی زیر برقرار است:

$$KB \models \alpha \text{ iff } M(KB) \subseteq M(\alpha)$$

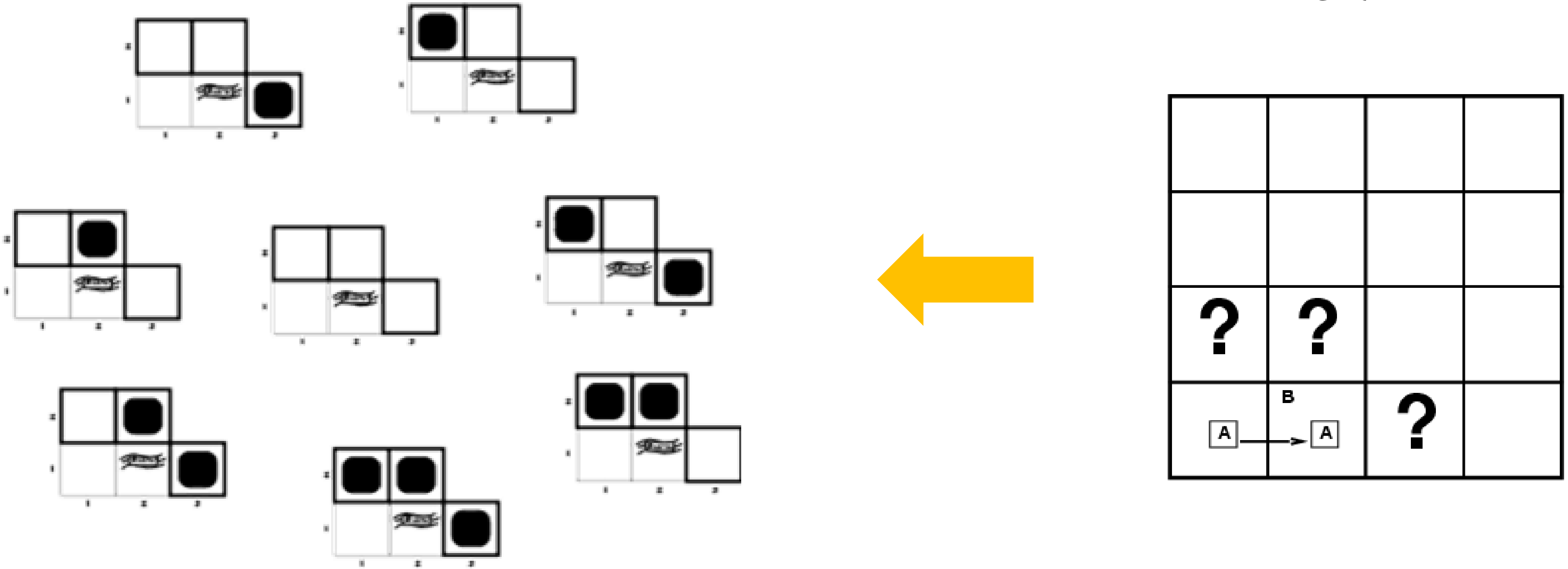
  - مثال:

KB= “A is red” and “B is blue”

α: “A is red”

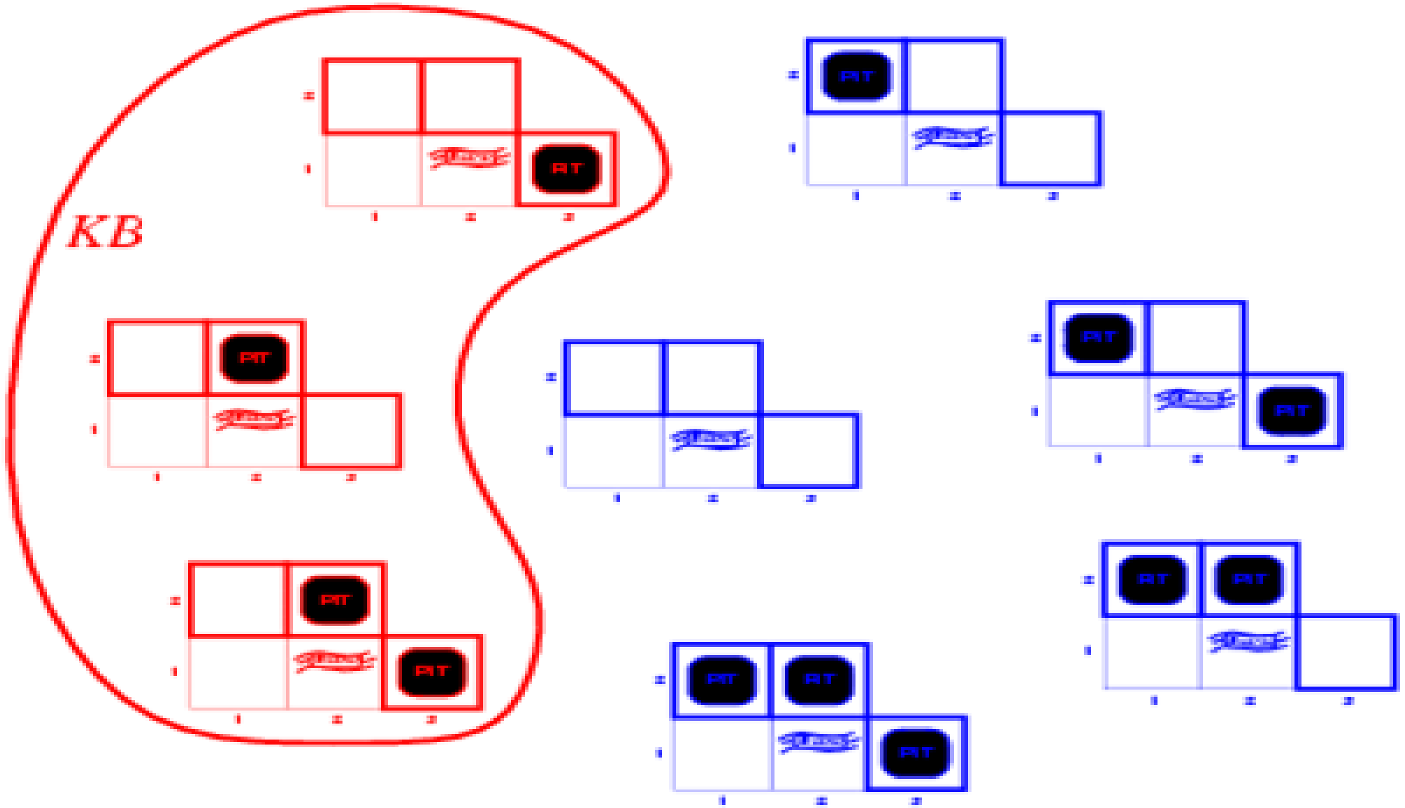
# ایجاب در دنیای وامپوس

- مدل‌های ممکن در ?ها را برای چاله‌ها در نظر بگیرید.
  - سه انتخاب بولین: هشت مدل مختلف



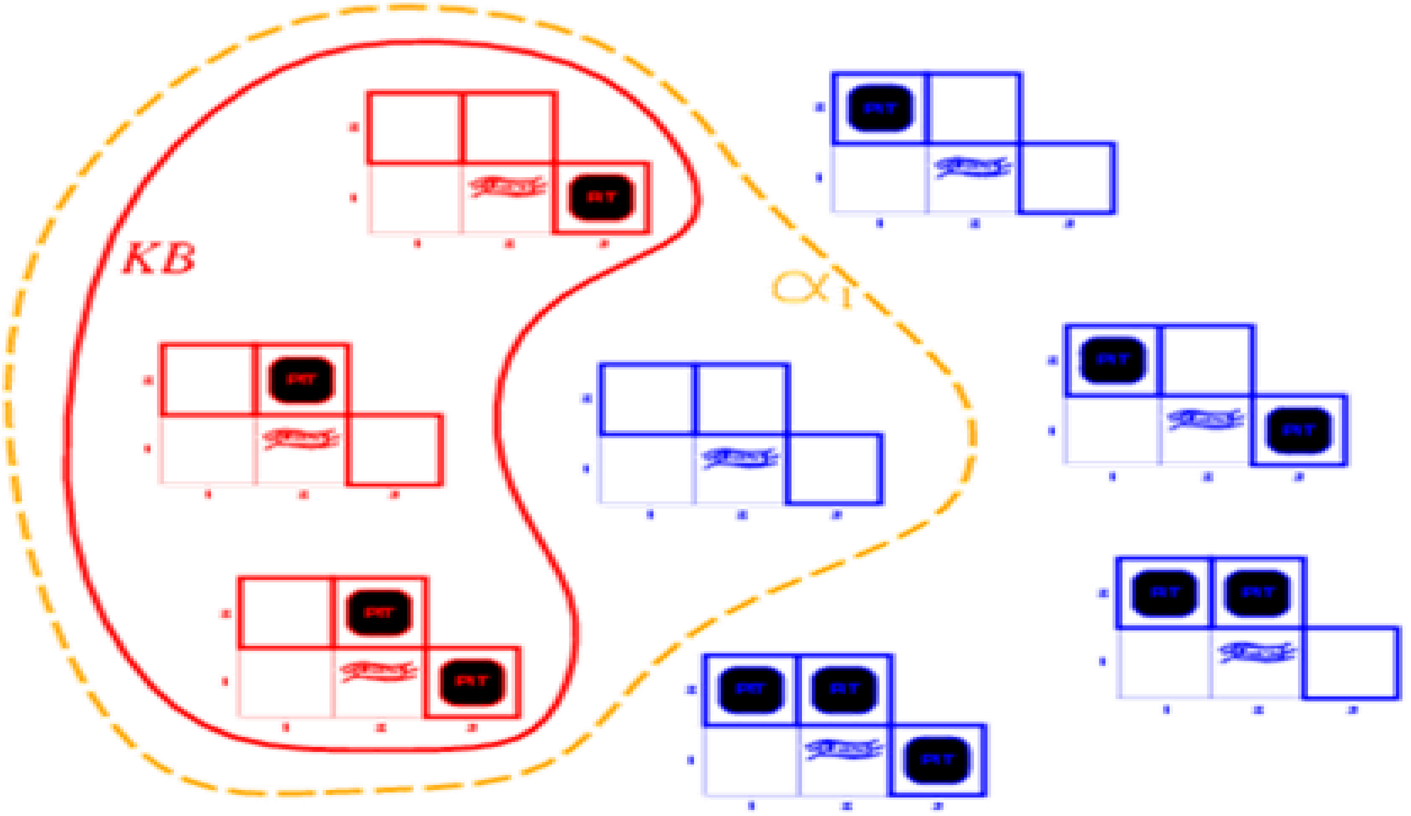
# ایجاب در دنیای وامپوس ...

- قبل از ادراک: KB شامل قوانین دنیای وامپوس است.
- ادراک: پس از آن‌که هیچ چیز در [1,1] درک نکرد به به [2,1] می‌رود و نسیم احساس می‌کند.




$KB$ = wumpus-world rules + observations

# ایجاب در دنیای وامپوس ...

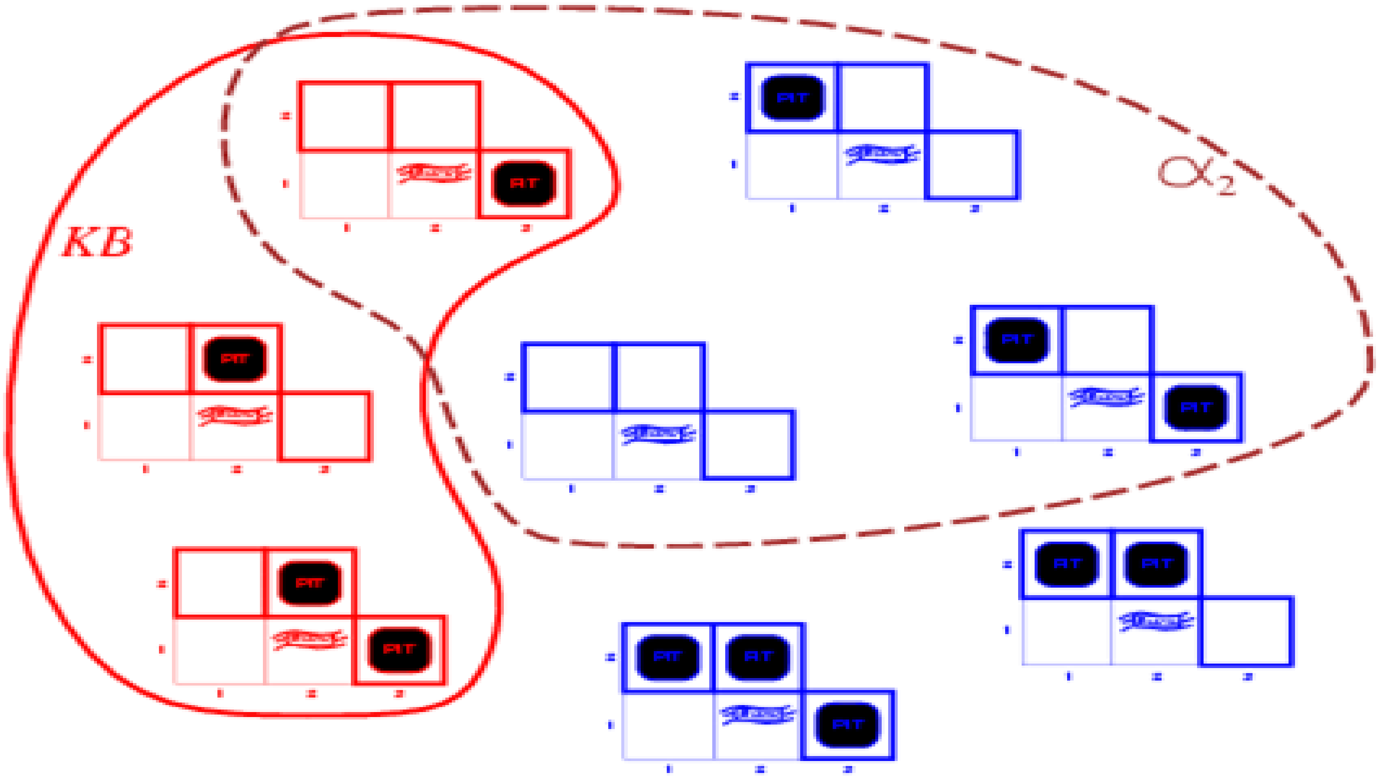



$KB$ = wumpus-world rules + observations

$\alpha_1$ = "[1,2] is safe" $\rightarrow$ M(KB) $\subseteq$ M($\alpha_1$) $\rightarrow KB \models \alpha_1$

(proved by model checking)

# ایجاب در دنیای وامپوس ...




$KB$ = wumpus-world rules + observations

$\alpha_2$ = "[2,2] is safe" $\rightarrow$ M(KB) $\not\subseteq$ M($\alpha_2$) $\rightarrow$ $KB \not\models \alpha_2$

# استنتاج (Inference)

- هدف از استنتاج این است که برای جمله‌ای مانند $\alpha$ تعیین کنیم آیا $KB \models \alpha$ برقرار است یا خیر.
  - آیا یک جمله خاص مانند $\alpha$ را می‌توان از KB نتیجه گرفت یا خیر.
- برای استنتاج الگوریتم‌های مختلفی می‌تواند وجود داشته باشد. اگر الگوریتم استنتاج i بتواند جمله $\alpha$ را از KB نتیجه بگیرد می‌نویسیم: $KB \vdash_i \alpha$
  - جمله $\alpha$ از KB بوسیله رویه i قابل استخراج می‌باشد.
- برای مثال الگوریتمی که در مثال قبل به‌کار بردیم را وارسی مدل (Model checking) می‌گویند. زیرا تمامی مدل‌های ممکن را برشماری می‌کند تا وارسی کند که آیا جمله در تمام مدل‌های درست KB درست است یا خیر ($M(KB) \subseteq M(\alpha)$).

# استنتاج (Inference) ...

- دو ویژگی مهم الگوریتم‌های استنتاجی صحت و کامل بودن آن‌ها است.
- الگوریتم استنتاجی که فقط جملات ایجابی را به دست آورد صحیح است.

Soundness:

$i$ is sound if whenever $KB \vdash_i \alpha$, it is also true that $KB \models \alpha$

- الگوریتم استنتاجی که بتواند هر جمله ایجاب شدنی را به دست آورد کامل است.

Completeness:

$i$ is complete if whenever $KB \models \alpha$, it is also true that $KB \vdash_i \alpha$

# منطق گزاره‌ای یا منطق بولی (Propositional Logic)

منطق گزاره‌ای ساده‌ترین منطق برای تشریح ایده‌های اساسی در مورد منطق و استدلال است.

# نحو منطق گزاره‌ای

- جملات اتمیک (Atomic): عناصر نحوی غیر قابل تجزیه هستند و از یک نماد گزاره‌ای تشکیل شده‌اند.
  - نمادهای گزاره‌ای را با حروف بزرگ P، Q و ... نشان می‌دهیم.
  - مثال: $W_{1,3}$ یعنی وامپوس در خانه [1,3] است.
  - True و False هر کدام یک جمله هستند و به آن‌ها ثابت‌های گزاره‌ای می‌گویند.
    - True گزاره‌ای همواره درست و False گزاره‌ای همواره غلط است.
- جملات مرکب (Complex): ترکیبی از جملات ساده‌تر بوسیله‌ی رابطهای منطقی هستند.
  - رابطهای منطقی: not، and، or و ...

# نحو منطق گزاره‌ای ...

- جملات مرکب:

- If S is a sentence, $\neg S$ is a sentence (negation یا نقیض)
- If $S_1$ and $S_2$ are sentences, $S_1 \wedge S_2$ is a sentence (conjunction یا عطفی)
- If $S_1$ and $S_2$ are sentences, $S_1 \vee S_2$ is a sentence (disjunction یا فصلی)
- If $S_1$ and $S_2$ are sentences, $S_1 \Rightarrow S_2$ is a sentence (implication یا شرطی)
- If $S_1$ and $S_2$ are sentences, $S_1 \Leftrightarrow S_2$ is a sentence (biconditional یا دوشرطی)

- مثال:

$$(W_{1,3} \wedge P_{3,1}) \vee W_{2,2}$$

$$\underbrace{\boxed{(W_{1,3} \wedge P_{3,1})}}_{\text{(Antecedent) مقدم یا فرض}} \Rightarrow \underbrace{\boxed{\neg W_{2,2}}}_{\text{(Consequent) تالی یا نتیجه}}$$

# دستور زبان BNF جملات در منطق گزاره‌ای

$$
\begin{aligned}
Sentence &\rightarrow Atomic\ Sentence \mid Complex\ Sentence \\
AtomicSentence &\rightarrow True \mid False \mid P \mid Q \mid R \mid \ldots \\
ComplexSentence &\rightarrow (Sentence) \\
&\quad \mid \neg\ Sentence \\
&\quad \mid (Sentence \wedge Sentence) \\
&\quad \mid (Sentence \vee Sentence) \\
&\quad \mid (Sentence \Rightarrow Sentence) \\
&\quad \mid (Sentence \Leftrightarrow Sentence)
\end{aligned}
$$

برای بدون ابهام کردن نحو از پرانتزگذاری و ترتیب اولویت رابطها استفاده می‌شود:

$Precedence: \neg\ , \wedge\ , \vee\ , \Rightarrow, \Leftrightarrow$

# معانی منطق گزاره‌ای

- معانی، قواعد تعیین درستی یک جمله نسبت به یک مدل خاص هستند.
  - یک مدل در منطق گزاره‌ای مقدار درستی (true یا false) را برای هر نماد گزاره‌ای مشخص می‌کند.
  - مثال: برای نمادهای گزاره‌ای $P_{1,2}$، $P_{2,2}$ و $P_{3,1}$ یک مدل ممکن می‌تواند به‌صورت زیر باشد:

$$M_1=\{P_{1,2}=false, P_{2,2}=false, P_{3,1}=true\}$$

  - تعیین درستی یک جمله به‌صورت بازگشتی انجام می‌شود. (بررسی درستی یک جمله مرکب به بررسی درستی جملات ساده‌تر تبدیل می‌شود.)
  - می‌توان برای هر رابط (connective) قواعدی به‌صورت جدول درستی (truth table) خلاصه نمود که درستی هر جمله مرکب را به ازای تمام انتساب‌های ممکن مقادیر درستی به مولفه‌هایش تعیین می‌کند.

# معانی منطق گزاره‌ای– جدول درستی

| $P$ | $Q$ | $\neg P$ | $P \wedge Q$ | $P \vee Q$ | $P \Rightarrow Q$ | $P \Leftrightarrow Q$ |
|---|---|---|---|---|---|---|
| *false* | *false* | *true* | *false* | *false* | *true* | *true* |
| *false* | *true* | *true* | *false* | *true* | *true* | *false* |
| *true* | *false* | *false* | *false* | *true* | *false* | *false* |
| *true* | *true* | *false* | *true* | *true* | *true* | *true* |

- قوانین درستی جملات ترکیبی در مدل $m$:

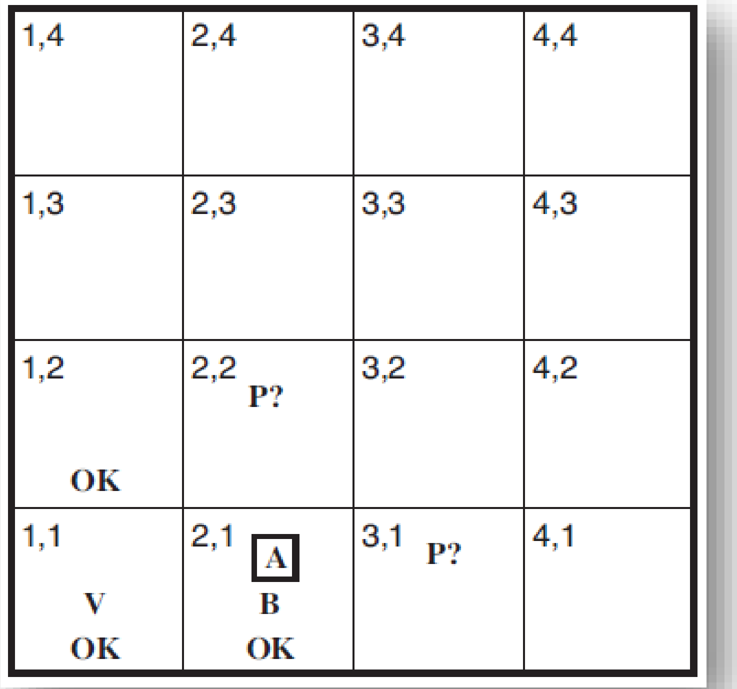
- $\neg$P is true iff P is false in $m$
- P $\wedge$ Q is true iff P is true and Q is true in $m$
- P $\vee$ Q is true iff P is true or Q is true in $m$
- P $\Rightarrow$ Q is true unless P is true and Q is false in $m$
- P $\Leftrightarrow$ Q is true iff P and Q are both true or both false in $m$

# تعریف پایگاه دانش

- اگر KB شامل گزاره‌های $S_1$، $S_2$، ... و $S_n$ باشد می‌توان KB را معادل با ترکیب عطفی این گزاره‌ها درنظر گرفت.
- یعنی پایگاه دانش ما باید همیشه شامل جملات یا حقیقت‌های صحیح باشند.

$$KB = S_1 \wedge S_2 \wedge \ldots \wedge S_n$$

# جملات دنیای وامپوس – پایگاه دانش ساده




- نمادهای زیر برای هر مکان [x,y] تعریف می‌شوند:
- $P_{x,y}$ is true if there is a pit in [x,y]
- $W_{x,y}$ is true if there is a wumpus in [x,y], dead or alive
- $B_{x,y}$ is true if the agent perceives a breeze in [x,y]
- $S_{x,y}$ is true if the agent perceives a stench in [x,y]
- قوانین کلی
- $R_1$: $\neg P_{1,1}$
- $R_2$: $B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$
- $R_3$: $B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$
- ادراکات
- $R_4$: $\neg B_{1,1}$
- $R_5$: $B_{2,1}$

# یک رویه استنتاج ساده

- هدف استنتاج منطقی: آیا برای جمله‌ای مانند α، می‌توان گفت: $KB \models \alpha$
- ساده‌ترین راه‌حل: پیاده‌سازی مستقیم تعریف ایجاب است یعنی برشماری مدل‌ها و وارسی این‌که آیا α در هر مدلی که KB در آن درست است، درست می‌باشد.
  - این کار از طریق بررسی سطرهای مختلف جدول درستی انجام می‌شود.
- مثال: پایگاه دانش زیر را در نظر بگیرید.
  - آیا $KB \models \neg P_{1,2}$؟
  - آیا $KB \models P_{2,2}$؟

KB

$R_1$: $\neg P_{1,1}$
$R_2$: $B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$
$R_3$: $B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$
$R_4$: $\neg B_{1,1}$
$R_5$: $B_{2,1}$

# استنتاج بوسیله‌ی جدول درستی – مثال

| $B_{1,1}$ | $B_{2,1}$ | $P_{1,1}$ | $P_{1,2}$ | $P_{2,1}$ | $P_{2,2}$ | $P_{3,1}$ | $R_1$ | $R_2$ | $R_3$ | $R_4$ | $R_5$ | $KB$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| *false* | *false* | *false* | *false* | *false* | *false* | *false* | *true* | *true* | *true* | *true* | *false* | *false* |
| *false* | *false* | *false* | *false* | *false* | *false* | *true* | *true* | *true* | *false* | *true* | *false* | *false* |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |
| *false* | *true* | *false* | *false* | *false* | *false* | *false* | *true* | *true* | *false* | *true* | *true* | *false* |
| *false* | *true* | *false* | *false* | *false* | *false* | *true* | *true* | *true* | *true* | *true* | *true* | <u>*true*</u> |
| *false* | *true* | *false* | *false* | *false* | *true* | *false* | *true* | *true* | *true* | *true* | *true* | <u>*true*</u> |
| *false* | *true* | *false* | *false* | *false* | *true* | *true* | *true* | *true* | *true* | *true* | *true* | <u>*true*</u> |
| *false* | *true* | *false* | *false* | *true* | *false* | *false* | *true* | *false* | *false* | *true* | *true* | *false* |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |
| *true* | *true* | *true* | *true* | *true* | *true* | *true* | *false* | *true* | *true* | *false* | *true* | *false* |

- آیا در هر سطری که KB درست است α نیز درست است.

☆ آیا $\mathbf{P_{1,2}} \neg \models KB$؟

☆ آیا $\mathbf{P_{2,2}} \models KB$؟

# استنتاج بوسیله‌ی جدول درستی (یا وارسی مدل)

```
function TT-ENTAILS?(KB, α) returns true or false
   inputs: KB, the knowledge base, a sentence in propositional logic
           α, the query, a sentence in propositional logic

   symbols ← a list of the proposition symbols in KB and α
   return TT-CHECK-ALL(KB, α, symbols, { })
```

---

```
function TT-CHECK-ALL(KB, α, symbols, model) returns true or false
   if EMPTY?(symbols) then
      if PL-TRUE?(KB, model) then return PL-TRUE?(α, model)
      else return true // when KB is false, always return true
   else do
      P ← FIRST(symbols)
      rest ← REST(symbols)
      return (TT-CHECK-ALL(KB, α, rest, model ∪ {P = true})
              and
              TT-CHECK-ALL(KB, α, rest, model ∪ {P = false }))
```

# استنتاج بوسیله‌ی جدول درستی – ویژگی‌ها

- الگوریتم استنتاج TT-ENTAILS? یک برشماری بازگشتی از فضای محدود انتساب‌ها را برای متغیرها اجرا می‌کند.
- آیا این الگوریتم صحیح است؟
  - بله زیرا به‌طور مستقیم تعریف ایجاب را پیاده‌سازی می‌کند.
- آیا این الگوریتم کامل است؟
  - بله زیرا تعداد محدودی مدل برای آزمایش وجود دارد.
- پیچیدگی زمانی؟ $O(2^n)$
  - اگر جمله $\alpha$ و KB در مجموع $n$ متغیر (سمبل) داشته باشد، تعداد مدل‌ها (یا همان تعداد سطرهای جدول درستی) برابر با $2^n$ خواهد بود.
- پیچیدگی فضایی؟ $O(n)$
  - زیرا از نوع اول عمق است.

# اثبات قضیه گزاره‌ای

PROPOSITIONAL THEOREM PROVING

# استنتاج از طریق اثبات قضیه

- الگوریتم استنتاج در این روش سعی می‌کند جمله را اثبات کند.
- منظور از اثبات یک جمله، به‌کارگیری دنباله‌ای از قوانین استنتاج برای رسیدن از KB به آن جمله است.
- جستجوی برهان‌ها می‌تواند کارآمدتر از وارسی مدل باشد.
  - می‌تواند از گزاره‌های نامرتبط صرف نظر کند.
  - هنگامی که تعداد مدل‌ها زیاد است اما طول برهان‌ها کوتاه است، ایجاب با اثبات قضیه مفید است.

KB

$R_1: \neg L$
$R_2: P \Rightarrow (Q \wedge R)$
$R_3: M \wedge R \Rightarrow L$
$R_4: P$
$R_5: R \vee N$

# هم‌ارزی منطقی

- دو جمله $\alpha$ و $\beta$ از نظر منطقی هم‌ارز هستند اگر و تنها اگر در مجموعه یکسانی از مدل‌ها درست باشند.
  - با استفاده از هم‌ارزی‌های منطقی ما می‌توانیم به قوانین استنتاج زیادی دست یابیم.

$\alpha \equiv \beta$ if and only if $\alpha \models \beta$ and $\beta \models \alpha$

$$M(\alpha) \subseteq M(\beta) \qquad M(\beta) \subseteq M(\alpha)$$

$$M(\alpha) = M(\beta)$$

# هم‌ارزی منطقی ...

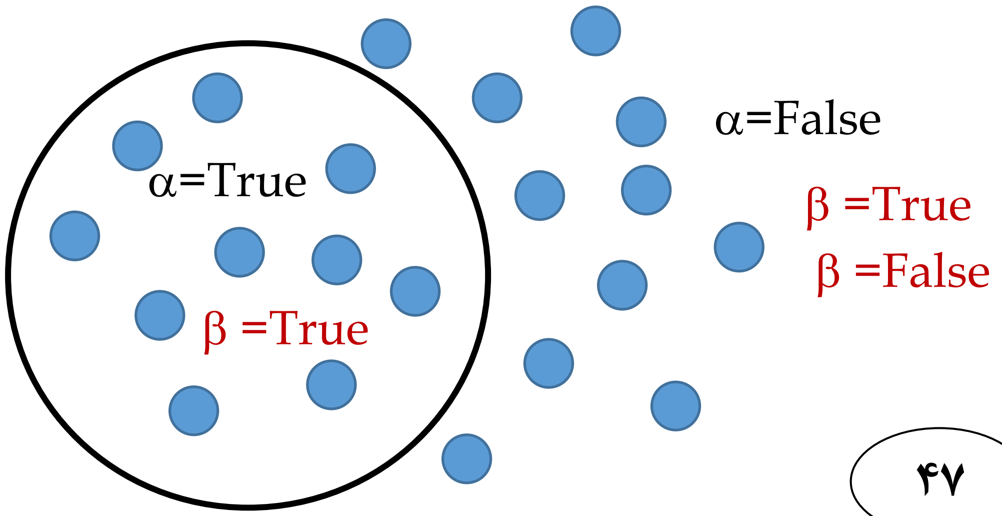
$$(\alpha \wedge \beta) \equiv (\beta \wedge \alpha) \quad \text{commutativity of } \wedge$$
$$(\alpha \vee \beta) \equiv (\beta \vee \alpha) \quad \text{commutativity of } \vee$$
$$((\alpha \wedge \beta) \wedge \gamma) \equiv (\alpha \wedge (\beta \wedge \gamma)) \quad \text{associativity of } \wedge$$
$$((\alpha \vee \beta) \vee \gamma) \equiv (\alpha \vee (\beta \vee \gamma)) \quad \text{associativity of } \vee$$
$$\neg(\neg\alpha) \equiv \alpha \quad \text{double-negation elimination}$$
$$(\alpha \Rightarrow \beta) \equiv (\neg\beta \Rightarrow \neg\alpha) \quad \text{contraposition}$$
$$(\alpha \Rightarrow \beta) \equiv (\neg\alpha \vee \beta) \quad \text{implication elimination}$$
$$(\alpha \Leftrightarrow \beta) \equiv ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)) \quad \text{biconditional elimination}$$
$$\neg(\alpha \wedge \beta) \equiv (\neg\alpha \vee \neg\beta) \quad \text{De Morgan}$$
$$\neg(\alpha \vee \beta) \equiv (\neg\alpha \wedge \neg\beta) \quad \text{De Morgan}$$
$$(\alpha \wedge (\beta \vee \gamma)) \equiv ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma)) \quad \text{distributivity of } \wedge \text{ over } \vee$$
$$(\alpha \vee (\beta \wedge \gamma)) \equiv ((\alpha \vee \beta) \wedge (\alpha \vee \gamma)) \quad \text{distributivity of } \vee \text{ over } \wedge$$

| $\alpha$ | $\beta$ | $\alpha \Rightarrow \beta$ | $\neg\alpha \vee \beta$ |
|---|---|---|---|
| True | True | True | True |
| True | False | False | False |
| False | True | True | True |
| False | False | True | True |

# اعتبار (Validity)

- یک جمله معتبر (valid) است اگر در تمام مدل‌ها درست باشد.
  - به‌عبارت دیگر یک جمله معتبر است اگر به ازای تمام سطرهای جدول درستی، آن جمله درست باشد.
  - مثال: $P \vee \neg P$
  - نام دیگر: تاتولوژی (tautology)، بدیهیات
- قضیه قیاس (deduction theorem)

$\alpha \models \beta$ if and only if $\alpha \Rightarrow \beta$ is valid



# ارضاپذیری (Satisfiability)

- یک جمله ارضاپذیر (satisfiable) است اگر در بعضی از مدل‌ها درست باشد.
  - به‌عبارت دیگر اگر حداقل به ازای یکی از سطرهای جدول درستی برابر با true باشد.
  - مثال: جمله $P \vee Q$ ارضاشدنی است چون سه مدل وجود دارد که در آن‌ها درست است.
- یک جمله ارضاناپذیر است، هرگاه در هیچ مدلی درست نباشد.
  - به‌عبارت دیگر اگر به ازای تمام سطرهای جدول درستی برابر با false باشد.
  - مثال: $P \wedge \neg P$
- جمله $\alpha$ معتبر (همیشه درست) است اگر و تنها اگر $\neg\alpha$ ارضانشدنی باشد.
- رابطه ایجاب و ارضاناپذیری (برهان خلف)

$\alpha \models \beta$ if and only if $\alpha \wedge \neg \beta$ is unsatisfiable

# تست ۱

کدام یک از جملات زیر نادرست است؟ (مهندسی کامپیوتر دولتی ۹۵)

۱) اگر $\alpha \wedge \beta \Rightarrow \gamma$ همیشه درست باشد، آن‌گاه حداقل یکی از دو جمله $\alpha \Rightarrow \gamma$ و $\beta \Rightarrow \gamma$ همیشه درست است.

۲) اگر $\alpha \vee \beta \Rightarrow \gamma$ همیشه درست باشد، آن‌گاه هر دو جمله $\alpha \Rightarrow \gamma$ و $\beta \Rightarrow \gamma$ همیشه درست هستند.

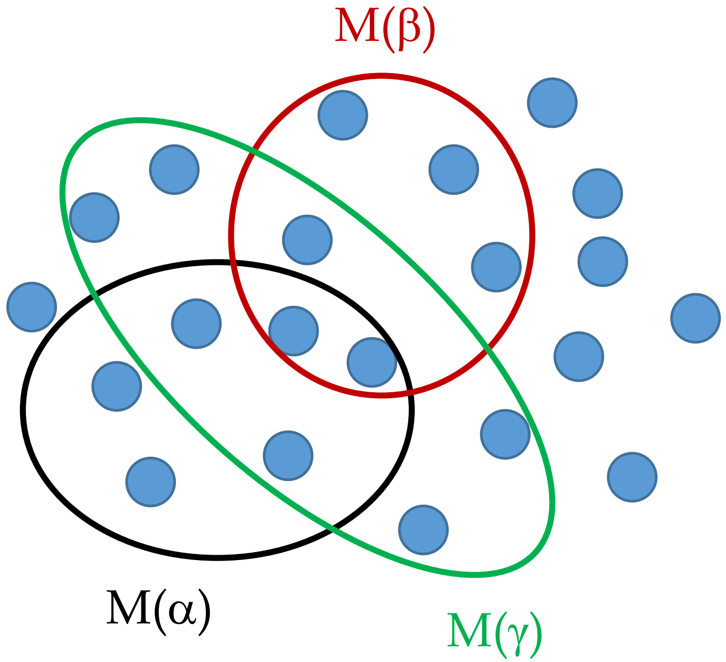
۳) $\alpha$ یک جمله همیشه درست اگر و فقط اگر $\text{True} \Rightarrow \alpha$ همیشه درست باشد.

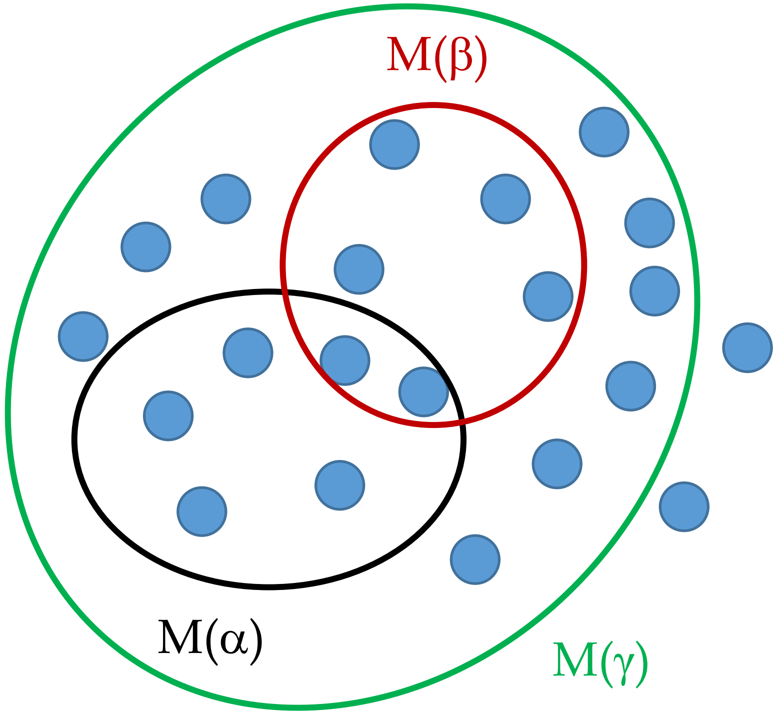
۴) $\alpha \Rightarrow \beta$ همیشه درست است اگر و فقط اگر $\alpha \wedge \neg \beta$ یک جمله غیرقابل ارضا (unsatisfiable) باشد.

# تست ۱

✓ ۱) اگر $\alpha \wedge \beta \Rightarrow \gamma$ همیشه درست باشد، آن‌گاه حداقل یکی از دو جمله $\alpha \Rightarrow \gamma$ و $\beta \Rightarrow \gamma$ همیشه درست است.

۲) اگر $\alpha \vee \beta \Rightarrow \gamma$ همیشه درست باشد، آن‌گاه هر دو جمله $\alpha \Rightarrow \gamma$ و $\beta \Rightarrow \gamma$ همیشه درست هستند.



# چند قانون استنتاج

- قوانین استنتاج می‌توانند برای حصول زنجیره‌ای از نتایج که به هدف مطلوب منجر می‌شوند، به‌کار برده می‌شوند.
- تمامی هم‌ارزی‌های منطقی می‌توانند به‌عنوان قواعد استنتاجی به‌کار برده شوند.
  - هم‌ارزی حذف دوشرطی (Biconditional Elimination)

$$\frac{\alpha \Leftrightarrow \beta}{(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)} \qquad \frac{(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)}{\alpha \Leftrightarrow \beta}$$

- قانون Modus Ponens:

$$\frac{\alpha \Rightarrow \beta, \alpha}{\beta} \qquad \frac{(WumpusAhead \wedge WumpusAlive) \Rightarrow Shoot, (WumpusAhead \wedge WumpusAlive)}{Shoot}$$

- قانون حذف عطف (And Elimination):

$$\frac{\alpha \wedge \beta}{\alpha} \qquad \frac{(WumpusAhead \wedge WumpusAlive)}{WumpusAlive}$$

# استفاده از قوانین استنتاج و هم‌ارزی در دنیای وامپوس

- آیا می‌توان با استفاده از قوانین موجود در پایگاه دانش زیر اثبات کرد در [1,2] گودالی نیست؟
  - باید اثبات کنیم: $\neg P_{1,2}$

$R_1$: $\neg P_{1,1}$
$R_2$: $B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$
$R_3$: $B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$
$R_4$: $\neg B_{1,1}$
$R_5$: $B_{2,1}$

$R_6$: $(B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$ — اعمال حذف دوشرطی بر روی $R_2$

$R_7$: $((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$ — اعمال حذف And بر روی $R_6$

$R_8$: $(\neg B_{1,1} \Rightarrow \neg(P_{1,2} \vee P_{2,1}))$ — $(P \Rightarrow Q) \equiv (\neg Q \Rightarrow \neg P)$

$R_9$: $\neg(P_{1,2} \vee P_{2,1})$ — قانون Modus Ponens با قوانین $R_8$ و $R_4$

$R_{10}$: $\neg P_{1,2} \wedge \neg P_{2,1}$ — قانون De Morgan

# یافتن اثبات به صورت یک مسئله جستجو

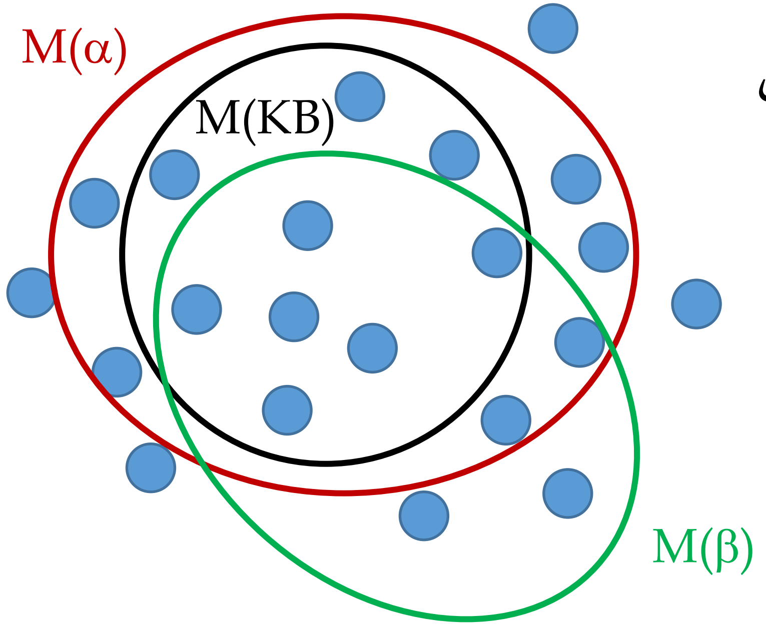
- INITIAL STATE: the initial knowledge base.
- ACTIONS: the set of actions consists of all the inference rules applied to all the sentences that match the top half of the inference rule.
- RESULT: the result of an action is to add the sentence in the bottom half of the inference rule.
- GOAL: the goal is a state that contains the sentence we are trying to prove.

# خاصیت یکنواختی (Monotonicity)

- با اضافه شدن جمله‌ای جدید به KB، نتایجی که قبلا از جملات قبلی KB گرفته شده‌اند معتبر باقی می‌مانند.

if $KB \models \alpha$ then $KB \wedge \beta \models \alpha$

- به‌عبارت دیگر با اضافه شدن جمله جدید به KB ممکن نیست یکی از جملات قبلی KB نامعتبر شود و از KB حذف شود، بلکه برعکس، اندازه KB رشد می‌کند.
- قواعد استنتاج می‌توانند تا هر زمان که مقدمه‌های مناسب در پایگاه دانش یافت می‌شوند، به‌کار برده شوند.



# رزولوشن (Resolution)

- قواعد استنتاج گفته‌شده صحیح هستند.
- با این وجود، آیا هر مجموعه‌ای از قوانین برای استنتاج، کامل است؟
  - خیر، در صورتی که قواعد استنتاج ناکافی باشد هدف دست‌نیافتنی خواهد بود.
  - برای مثال اگر قانون حذف دوشرطی را در مثال دنیای وامپوس نادیده بگیریم، اثبات انجام نخواهد شد.
- قوانین رزولوشن را می‌توان به‌تنهایی در هر یک از الگوریتم‌های جستجوی کامل استفاده کرد و یک الگوریتم جستجوی کامل به‌دست آورد.

# رزولوشن ...

- قانون رزولوشن واحد (Unit): اگر $l_i$ و $m$ لیترال‌های نقیض هم باشند $(l_i = {\sim}m)$ آن‌گاه

$$\frac{l_1 \vee \ldots \vee l_k , m}{l_1 \vee \ldots \vee l_{i-1} \vee l_{i+1} \ldots \vee l_k}$$

$$\frac{{\sim}p \vee q \vee r \vee s \quad , {\sim}r}{{\sim}p \vee q \vee s}$$

- قانون رزولوشن کامل (Full): اگر $l_i$ و $m_j$ لیترال‌های نقیض هم باشند $(l_i = {\sim}m_j)$ آن‌گاه

$$\frac{l_1 \vee \ldots \vee l_k , \quad m_1 \vee \ldots \vee m_n}{l_1 \vee \ldots \vee l_{i-1} \vee l_{i+1} \vee \ldots \vee l_k \vee m_1 \vee \ldots \vee m_{j-1} \vee m_{j+1} \vee \ldots \vee m_n}$$

$$\frac{{\sim}p \vee q \vee r \vee s \quad , t \vee q \vee {\sim}r}{{\sim}p \vee q \vee s \vee t}$$

# رزولوشن ...

- آیا قانون رزولوشن صحیح است؟چرا؟

$$\frac{l_1 \vee \ldots \vee l_k, \quad m_1 \vee \ldots \vee m_n}{l_1 \vee \ldots \vee l_{i-1} \vee l_{i+1} \vee \ldots \vee l_k \vee m_1 \vee \ldots \vee m_{j-1} \vee m_{j+1} \vee \ldots \vee m_n}$$

$$\begin{array}{ccc} l_i = False & & l_i = True \\ \Big\downarrow & & \downarrow \\ & & m_j = False \\ \Big\downarrow & & \downarrow \\ l_1 \vee l_2 \ldots l_{i-1} \vee l_{i+1} \vee \cdots \vee l_k & \vee & m_1 \vee m_2 \ldots m_{j-1} \vee m_{j+1} \vee \cdots \vee m_n \end{array}$$

# کاربرد رزولوشن در اثبات

- یک اثبات‌کننده قضیه‌ی مبتنی‌بر رزولوشن می‌تواند برای هر جمله‌ی $\alpha$ و $\beta$ در منطق گزاره‌ای تصمیم بگیرد که آیا $\alpha \models \beta$ یا خیر.
- برای این منظور باید به دو بحث بپردازیم:

۱- شکل نرمال عطفی (Conjunctive Normal Form - CNF)

۲- ارائه یک الگوریتم مبتنی‌بر رابطه ایجاب و ارضاناپذیری (برهان خلف)

$\alpha \models \beta$ if and only if $\alpha \wedge \neg \beta$ is unsatisfiable

# شکل نرمال عطفی (CNF)

- جمله‌ای در CNF است که به صورت ترکیب عطفی یک یا چند عبارت ساده نوشته شده باشد به‌طوری‌که هر یک از این عبارات ساده، یک نماد یا ترکیب فصلی چند نماد باشد.

$$CNFSentence \rightarrow Clause_1 \wedge \ldots \wedge Clause_n$$

$$Clause \rightarrow Literal_1 \vee \ldots \vee Literal_m$$

$$Literal \rightarrow Symbol \mid \neg Symbol$$

$$Symbol \rightarrow P \mid Q \mid R \mid \ldots$$

- برای مثال:

$$(P_1 \vee P_2) \wedge (\neg P_3 \vee P_4 \vee P_5) \wedge P_6$$

# مراحل تبدیل یک گزاره به شکل نرمال عطفی (CNF)

۱- حذف $\alpha \Leftrightarrow \beta$ با استفاده از هم‌ارزی $\alpha \Leftrightarrow \beta \equiv (\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$

۲- حذف $\alpha \Rightarrow \beta$ با استفاده از هم‌ارزی $\alpha \Rightarrow \beta \equiv (\neg \alpha \vee \beta)$

۳- انتقال $\neg$ به داخل پرانتز با استفاده از هم‌ارزی‌های زیر

$$\neg(\neg \alpha) \equiv \alpha$$

$$\neg(\alpha \vee \beta) \equiv \neg \alpha \wedge \neg \beta$$

$$\neg(\alpha \wedge \beta) \equiv \neg \alpha \vee \neg \beta$$

۴- استفاده از توزیع‌پذیری $\vee$ نسبت به $\wedge$

# مراحل تبدیل یک گزاره به شکل نرمال عطفی (CNF)

- مثال: جمله $A \Leftrightarrow (B \vee C)$ را به شکل CNF درآورید.

$$A \Leftrightarrow (B \vee C)$$

$$\equiv (A \Rightarrow (B \vee C)) \wedge ((B \vee C) \Rightarrow A)$$

$$\equiv (\neg A \vee (B \vee C)) \wedge (\neg(B \vee C) \vee A)$$

$$\equiv (\neg A \vee B \vee C) \wedge ( (\neg B \wedge \neg C) \vee A)$$

$$\equiv (\neg A \vee B \vee C) \wedge (\neg B \vee A) \wedge (\neg C \vee A)$$

# الگوریتم رزولوشن

- $KB\wedge\neg\alpha$ را به فرم CNF نوشته و هر دو بندی که شامل لیترال‌های مکمل باشند را در هم resolve کرده تا در صورت وجود بند جدیدی حاصل شود. این فرایند تا زمان رخ دادن یکی از حالات زیر ادامه می‌یابد:

۱- با انجام قاعده رزولوشن به عبارت تهی برسیم. در این صورت عبارت $KB \wedge \neg\alpha$ برابر با False است. یعنی $\alpha$ را می‌توان ایجاب کرد.

- توجه: بند تهی با False هم‌ارز است. بند تهی تنها از حل دو بند واحد مکمل مانند P و $\neg P$ ناشی می‌شود.

۲- هیچ بند جدیدی را نتوان با انجام رزولوشن ایجاد کرد. در این حالت $\alpha$ را نمی‌توان استنباط کرد.

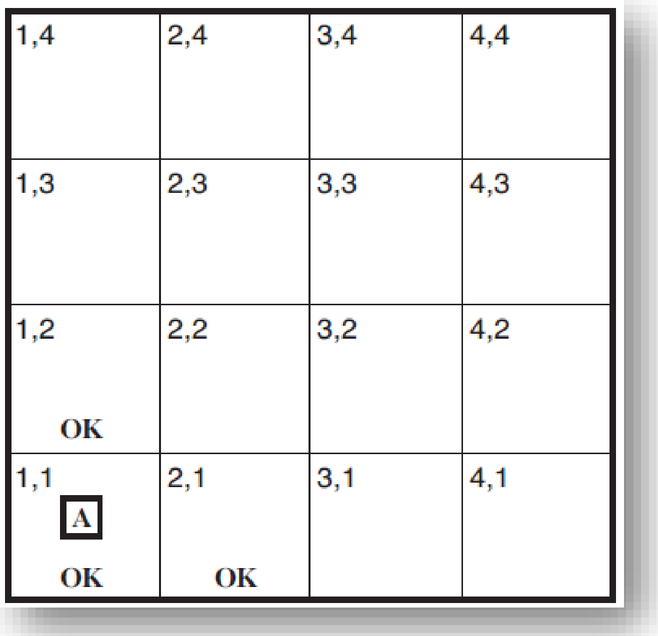
# الگوریتم رزولوشن – مثال

- فرض کنید پایگاه دانش به شکل زیر باشد:
  - در $[1,1]$ هیچ نسیمی وجود ندارد.
  - اگر عامل در $[1,1]$ باشد و نسیمی حس کند آن‌گاه ممکن است گودالی در همسایه‌های غیرقطری آن وجود داشته باشد.

$$KB = (B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})) \wedge \neg B_{1,1}$$

- آیا می‌توان نتیجه گرفت در $[1,2]$ گودال نیست؟

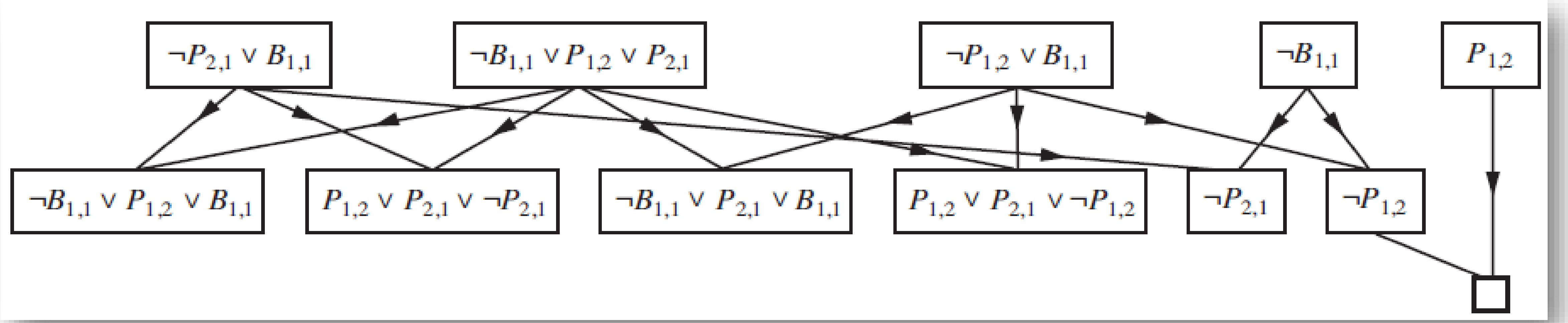
$$\alpha = \neg P_{1,2}$$



# الگوریتم رزولوشن – مثال ...

- تبدیل به شکل نرمال عطفی

$$(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg P_{1,2} \vee B_{1,1}) \wedge (\neg P_{2,1} \vee B_{1,1}) \wedge (\neg B_{1,1}) \wedge P_{1,2}$$



# الگوریتم رزولوشن

**function** PL-RESOLUTION($KB, \alpha$) **returns** *true* or *false*
  **inputs**: $KB$, the knowledge base, a sentence in propositional logic
    $\alpha$, the query, a sentence in propositional logic

  $clauses \leftarrow$ the set of clauses in the CNF representation of $KB \wedge \neg\alpha$
  $new \leftarrow \{\ \}$
  **loop do**
    **for each** pair of clauses $C_i, C_j$ **in** $clauses$ **do**
      $resolvents \leftarrow$ PL-RESOLVE($C_i, C_j$)
      **if** $resolvents$ contains the empty clause **then return** *true*
      $new \leftarrow new \cup resolvents$
    **if** $new \subseteq clauses$ **then return** *false*
    $clauses \leftarrow clauses \cup new$

یک الگوریتم استنتاج صحیح و کامل است.

# تست ۲

پایگاه دانش زیر مفروض است. کدام یک از گزینه‌های زیر با استفاده از روش رزولوشن از این پایگاه دانش قابل استنتاج است؟ (مهندسی کامپیوتر دولتی ۹۲)

۱) $S$ ✓

۲) $W$

۳) $V$

۴) $T$

$$P$$
$$V \vee T$$
$$\neg P \vee U$$
$$R \vee \neg Q$$
$$V \Rightarrow W$$
$$P \Rightarrow Q$$
$$S \Rightarrow (U \vee T)$$
$$(P \wedge R) \Rightarrow S$$

# شکل نرمال هورن (HNF)

- یک بند هورن، ترکیب فصلی لیترال‌هایی است که حداکثر یکی از آن‌ها مثبت است.
- یک بند معین، ترکیب فصلی لیترال‌هایی که فقط یک لیترال آن مثبت است.
- به بندهای هورن که لیترال مثبت ندارند، بند هدف می‌گویند
  - بند هدف $A \wedge B \wedge C \Rightarrow False \equiv \neg(A \wedge B \wedge C) \vee False \equiv \neg(A \wedge B \wedge C) \equiv \neg A \vee \neg B \vee \neg C$
  - بند معین $A \wedge B \Rightarrow C \equiv \neg(A \wedge B) \vee C \equiv \neg A \vee \neg B \vee C$
  - بند معین $A \Rightarrow B \equiv \neg A \vee B \equiv A \vee \neg B$
  - بند معین $True \Rightarrow A \equiv \neg True \vee A \equiv False \vee A \equiv A$
- بندهای هورن نسبت به رزولوشن بسته‌اند. یعنی اگر قانون رزولوشن را بر روی دو بند هورن اعمال کنیم، یک بند هورن خواهیم داشت.

# شکل نرمال هورن (HNF) ...

$\text{Clause} \rightarrow \text{Literal}_1 \lor \ldots \lor \text{Literal}_m$

$\text{Literal} \rightarrow \text{Symbol} \mid \neg\text{Symbol}$

$\text{Symbol} \rightarrow P \mid Q \mid R \mid \ldots$

$\text{HornClauseForm} \rightarrow \text{DefiniteClauseForm} \mid \text{GoalClauseForm}$

$\text{DefiniteClauseForm} \rightarrow (\text{Symbol}_1 \land \ldots \land \text{Symbol}_l) \Rightarrow \text{Symbol}$

$\text{GoalClauseForm} \rightarrow (\text{Symbol}_1 \land \ldots \land \text{Symbol}_l) \Rightarrow \text{False}$

- هر پایگاه دانش را می‌توان به شکل ترکیب عطفی جملات به فرم HNF نوشت.

# دلایل استفاده از بند معین در پایگاه دانش

۱- هر بند معین می‌تواند به‌صورت یک استلزام که مقدم آن ترکیب عطفی لیترال‌های مثبت و نتیجه‌ی آن یک لیترال مثبت است، نوشته شود.

۲- استنتاج در بندهای معین می‌تواند از طریق الگوریتم‌های زنجیره‌ای روبه‌جلو و زنجیره‌ای پس‌رو انجام شود.

۳- زمان تصمیم‌گیری در مورد ایجاب در بندهای معین می‌تواند برحسب اندازه‌ی پایگاه دانش به‌صورت خطی باشد.

$P$

$V$

$V \Rightarrow W$

$P \wedge W \Rightarrow R$

$W \wedge V \wedge R \Rightarrow S$

# الگوریتم زنجیره‌ای روبه‌جلو

**function** PL-FC-ENTAILS?($KB$, $q$) **returns** *true* or *false*
  **inputs**: $KB$, the knowledge base, a set of propositional definite clauses
      $q$, the query, a proposition symbol
  $count \leftarrow$ a table, where $count[c]$ is the number of symbols in $c$'s premise
  $inferred \leftarrow$ a table, where $inferred[s]$ is initially *false* for all symbols
  $agenda \leftarrow$ a queue of symbols, initially symbols known to be true in $KB$

  **while** $agenda$ is not empty **do**
    $p \leftarrow$ POP($agenda$)
    **if** $p = q$ **then return** *true*
    **if** $inferred[p]$ = *false* **then**
      $inferred[p] \leftarrow true$
      **for each** clause $c$ in $KB$ where $p$ is in $c$.PREMISE **do**
        decrement $count[c]$
        **if** $count[c] = 0$ **then** add $c$.CONCLUSION to $agenda$
  **return** *false*

# الگوریتم زنجیره‌ای روبه‌جلو

یک پایگاه دانش KB به‌صورت ترکیب عطفی عبارات معین و لیترال $\alpha$ داده شده است می‌خواهیم نشان دهیم آیا می‌توان $\alpha$ را از KB ایجاب کرد؟

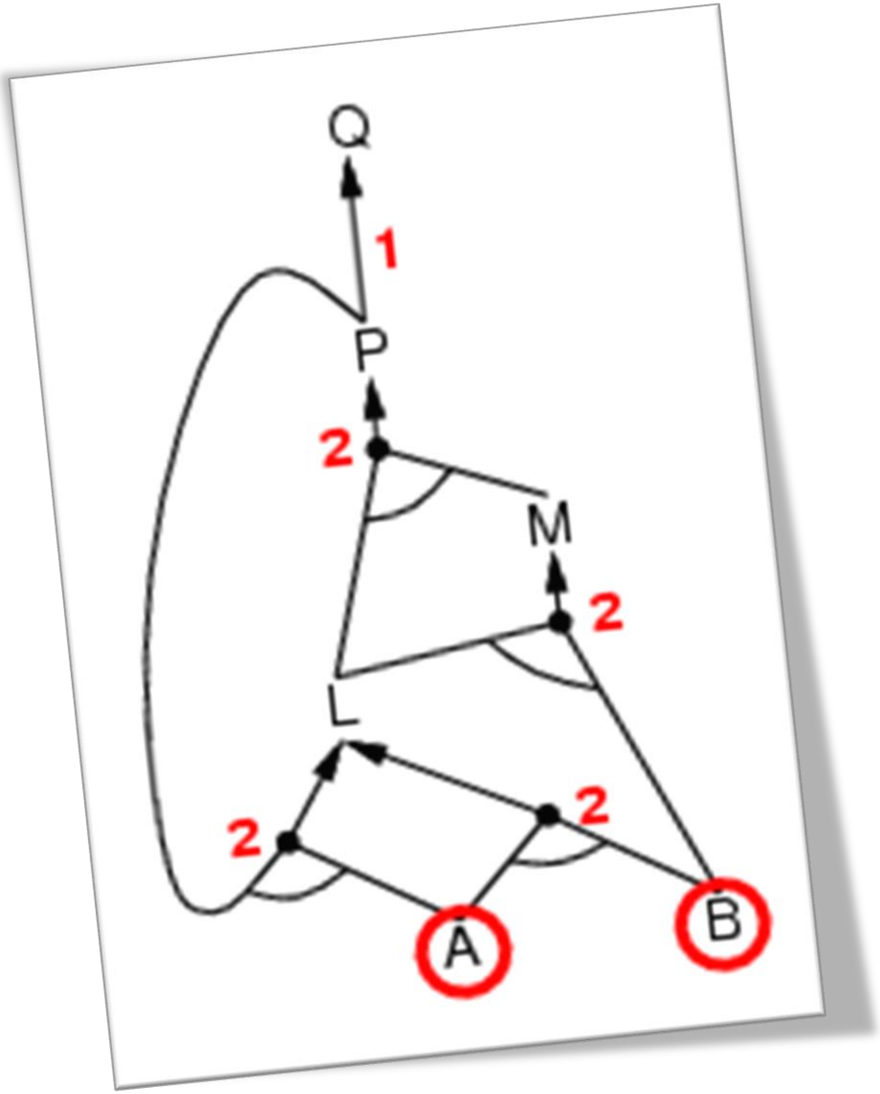
۱- تمام حقایق را به عنوان مجموعه لیترال‌های درست (agenda) در نظر می‌گیرد.

۲- به‌ازای هر قانون شرطی که تمام لیترال‌های موجود در مقدم آن شرط در مجموعه inferred باشد، لیترال موجود در تالی آن شرط به agenda اضافه می‌شود.

۳- مرحله ۲ تکرار می‌شود تا یکی از دو مورد زیر رخ دهد:

- لیترال $\alpha$ به agenda اضافه شود در این صورت $\alpha$ از KB ایجاب می‌شود.
- نتوان لیترال جدیدی به agenda اضافه کرد. در این صورت KB مستلزم $\alpha$ نیست.

# استنتاج روبه‌جلو – مثال



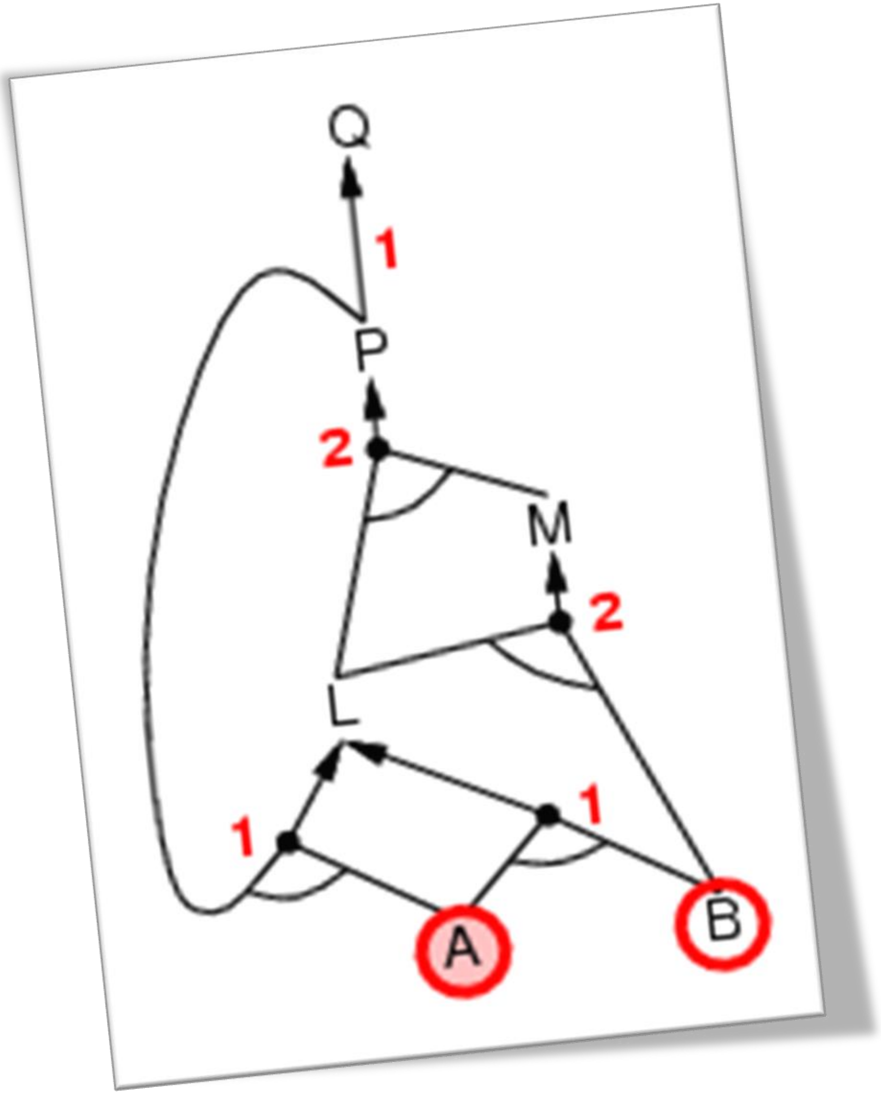

agenda = {A, B}

inferred = [false, false, false, false, false, false]

| | A | B | L | M | P | Q |
|---|---|---|---|---|---|---|

| | |
|---|---|
| $P \Rightarrow Q$ | $count(P \Rightarrow Q) = 1$ |
| $L \wedge M \Rightarrow P$ | $count(L \wedge M \Rightarrow P) = 2$ |
| $B \wedge L \Rightarrow M$ | $count(B \wedge L \Rightarrow M) = 2$ |
| $A \wedge P \Rightarrow L$ | $count(A \wedge P \Rightarrow L) = 2$ |
| $A \wedge B \Rightarrow L$ | $count(A \wedge B \Rightarrow L) = 2$ |
| A | A |
| B | B |

# استنتاج روبه‌جلو – مثال



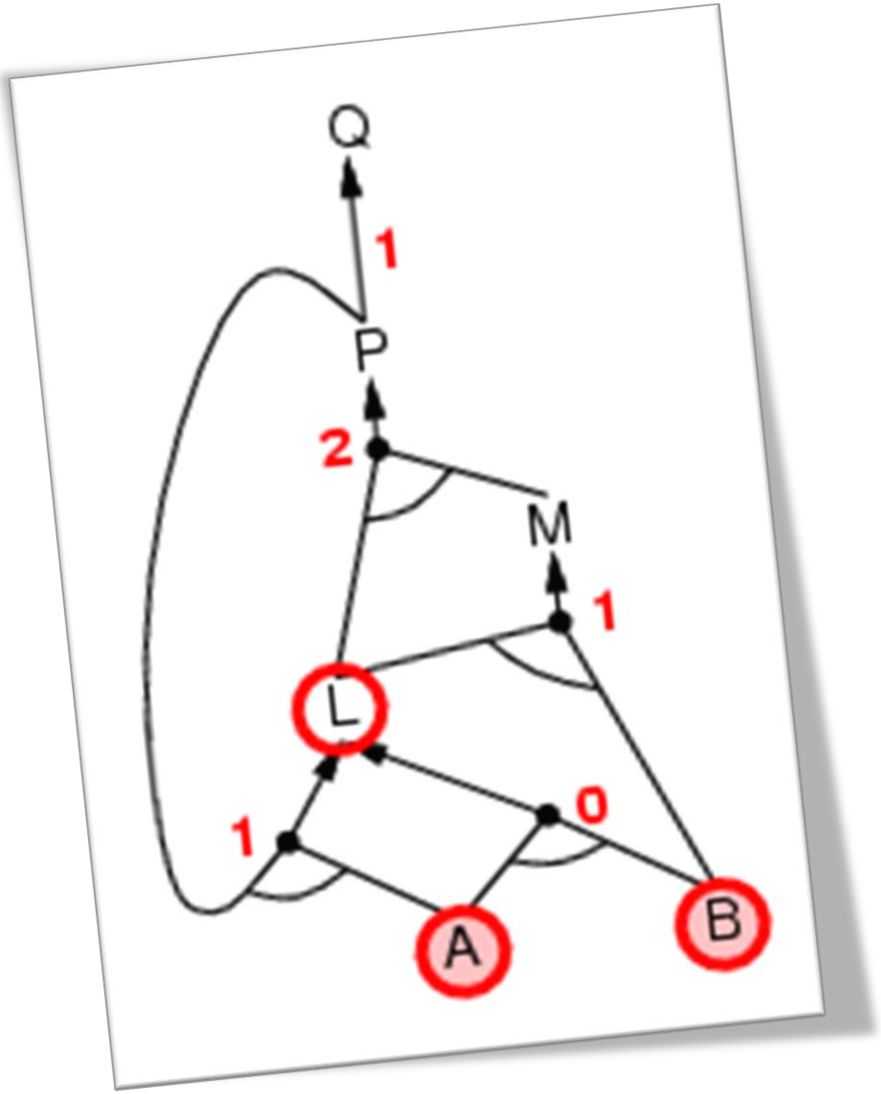

agenda = {B}

inferred = [true, false, false, false, false, false]

A B L M P Q

| | |
|---|---|
| $P \Rightarrow Q$ | count($P \Rightarrow Q$) = 1 |
| $L \wedge M \Rightarrow P$ | count($L \wedge M \Rightarrow P$) = 2 |
| $B \wedge L \Rightarrow M$ | count($B \wedge L \Rightarrow M$) = 2 |
| $A \wedge P \Rightarrow L$ | count($A \wedge P \Rightarrow L$) = 1 |
| $A \wedge B \Rightarrow L$ | count($A \wedge B \Rightarrow L$) = 1 |
| A | A |
| B | B |

# استنتاج روبه‌جلو – مثال



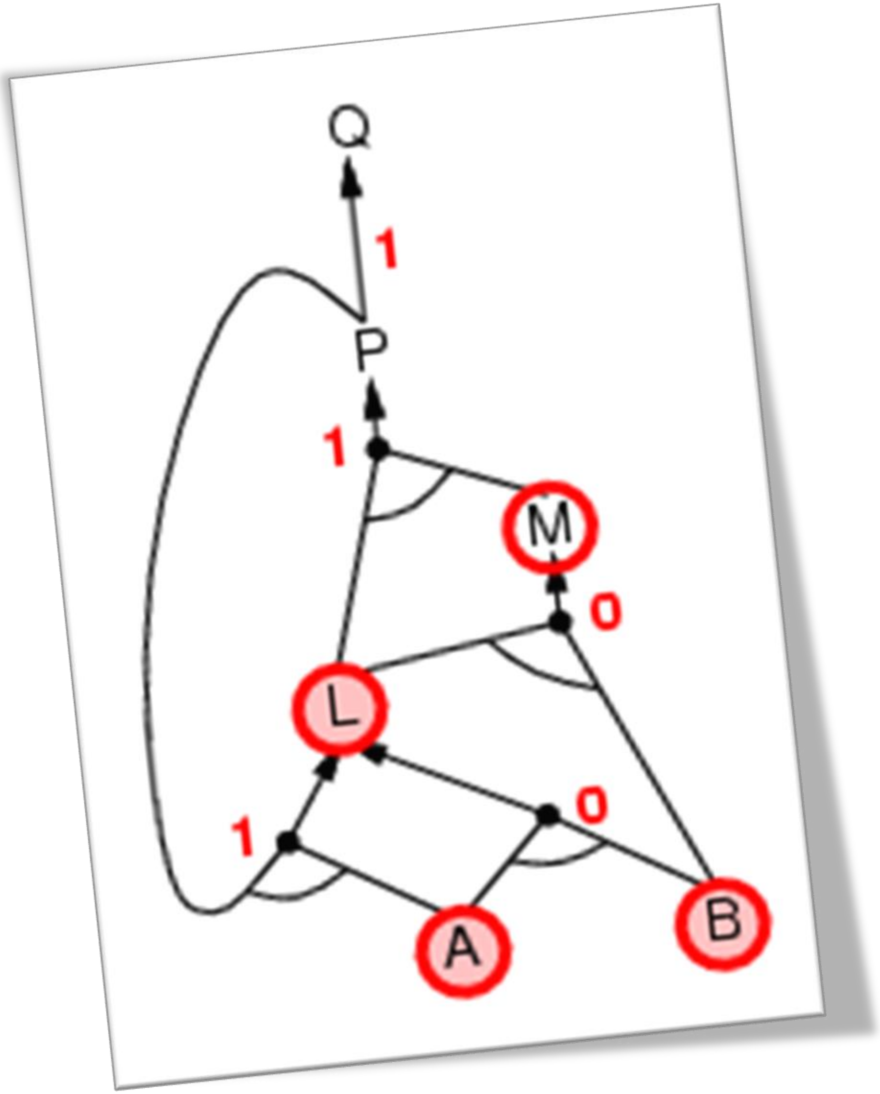

agenda = {L}

inferred = [true, true, false, false, false, false]

| | A | B | L | M | P | Q |
|---|---|---|---|---|---|---|

| | |
|---|---|
| $P \Rightarrow Q$ | count($P \Rightarrow Q$) = 1 |
| $L \wedge M \Rightarrow P$ | count($L \wedge M \Rightarrow P$) = 2 |
| $B \wedge L \Rightarrow M$ | count($B \wedge L \Rightarrow M$) = 1 |
| $A \wedge P \Rightarrow L$ | count($A \wedge P \Rightarrow L$) = 1 |
| $A \wedge B \Rightarrow L$ | count($A \wedge B \Rightarrow L$) = 0 |
| A | A |
| B | B |

# استنتاج روبه‌جلو – مثال



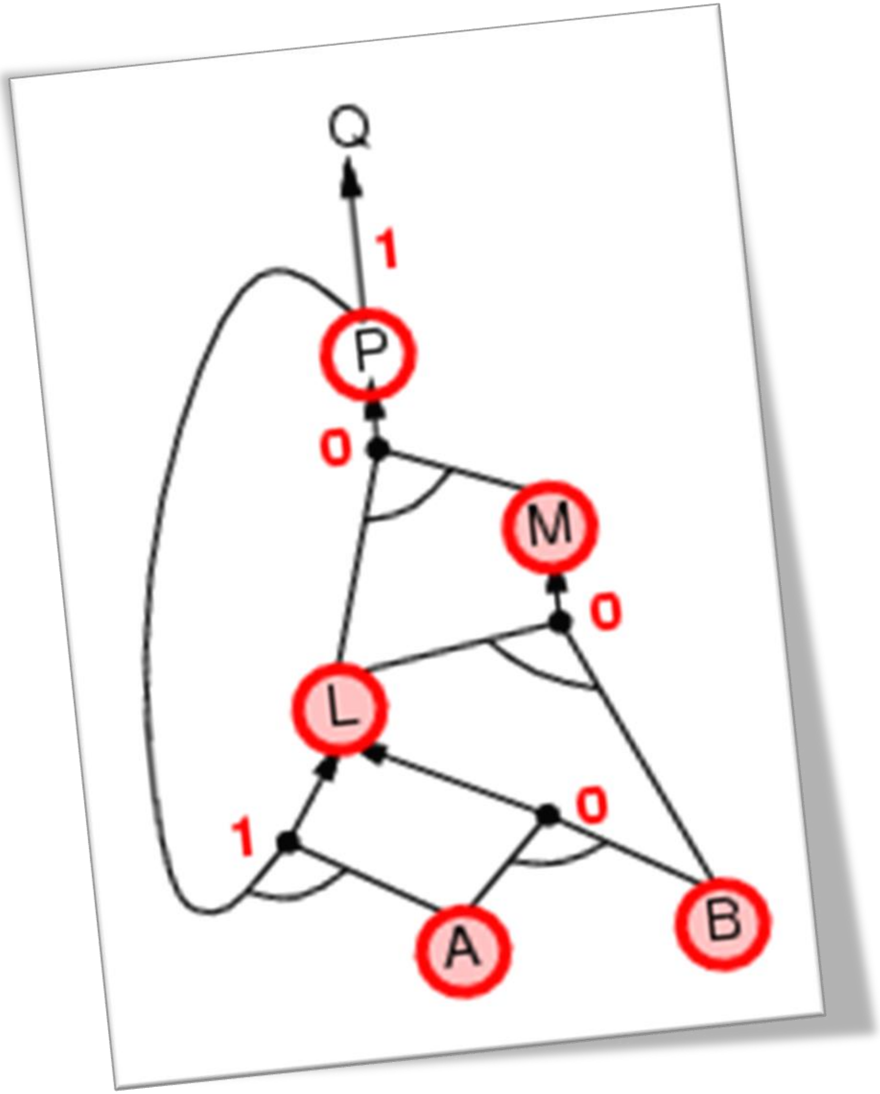

agenda = {M}

inferred = [true, true, true, false, false, false]

| | A | B | L | M | P | Q |
|---|---|---|---|---|---|---|

| | |
|---|---|
| $P \Rightarrow Q$ | count($P \Rightarrow Q$) = 1 |
| $L \wedge M \Rightarrow P$ | count($L \wedge M \Rightarrow P$) = 1 |
| $B \wedge L \Rightarrow M$ | count($B \wedge L \Rightarrow M$) = 0 |
| $A \wedge P \Rightarrow L$ | count($A \wedge P \Rightarrow L$) = 1 |
| $A \wedge B \Rightarrow L$ | count($A \wedge B \Rightarrow L$) = 0 |
| A | A |
| B | B |

# استنتاج روبه‌جلو – مثال



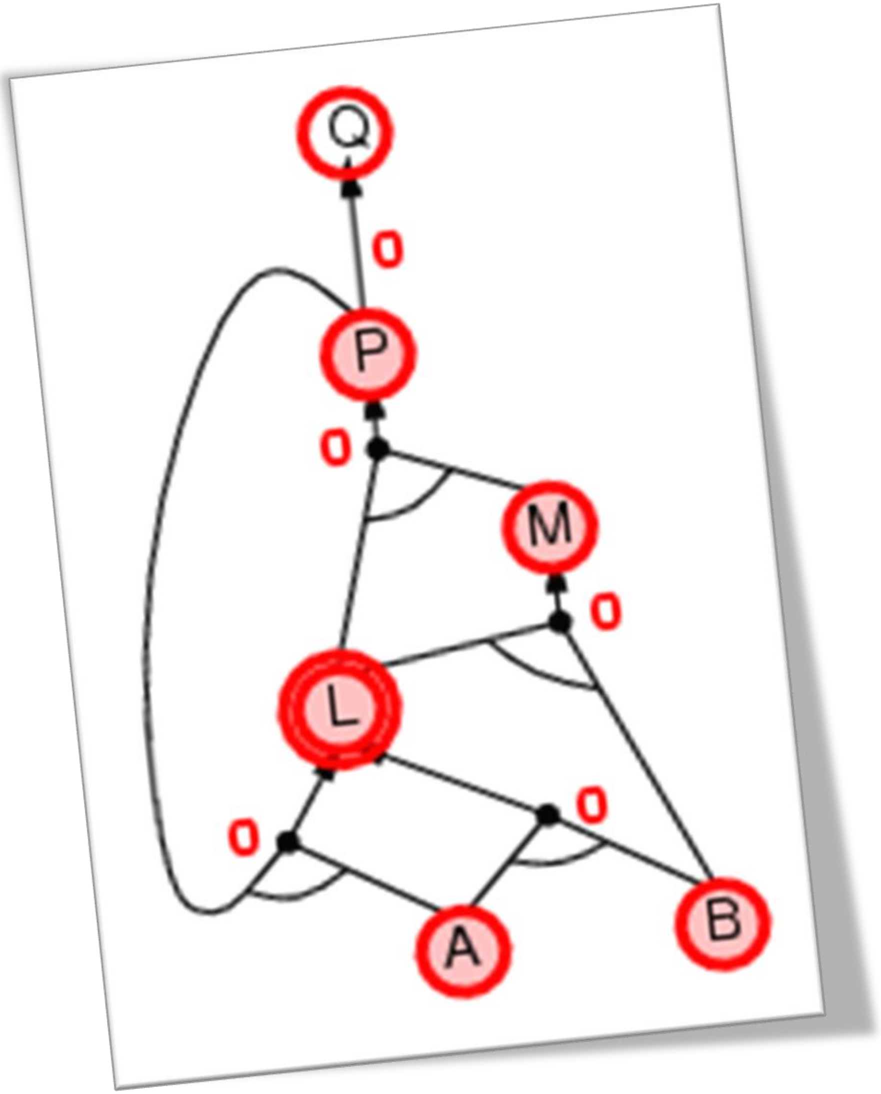

agenda = {P}

inferred = [true, true, true, true, false, false]

| | A | B | L | M | P | Q |
|---|---|---|---|---|---|---|

| | |
|---|---|
| $P \Rightarrow Q$ | count($P \Rightarrow Q$) = 1 |
| $L \wedge M \Rightarrow P$ | count($L \wedge M \Rightarrow P$) = 0 |
| $B \wedge L \Rightarrow M$ | count($B \wedge L \Rightarrow M$) = 0 |
| $A \wedge P \Rightarrow L$ | count($A \wedge P \Rightarrow L$) = 1 |
| $A \wedge B \Rightarrow L$ | count($A \wedge B \Rightarrow L$) = 0 |
| A | A |
| B | B |

# استنتاج روبه‌جلو – مثال



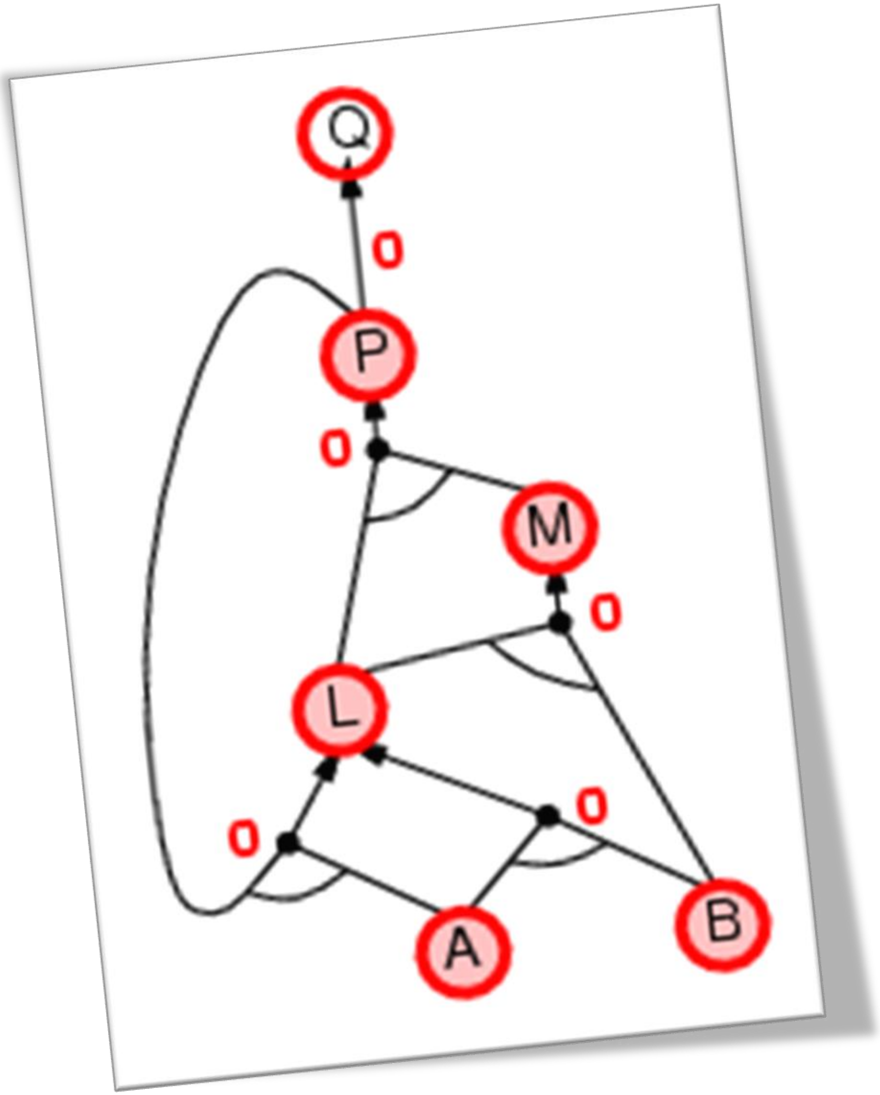

agenda = {L, Q}

inferred = [true, true, true, true, true, false]

| | A | B | L | M | P | Q |
|---|---|---|---|---|---|---|

| | |
|---|---|
| $P \Rightarrow Q$ | $count(P \Rightarrow Q) = 0$ |
| $L \wedge M \Rightarrow P$ | $count(L \wedge M \Rightarrow P) = 0$ |
| $B \wedge L \Rightarrow M$ | $count(B \wedge L \Rightarrow M) = 0$ |
| $A \wedge P \Rightarrow L$ | $count(A \wedge P \Rightarrow L) = 0$ |
| $A \wedge B \Rightarrow L$ | $count(A \wedge B \Rightarrow L) = 0$ |
| A | A |
| B | B |

# استنتاج روبه‌جلو – مثال



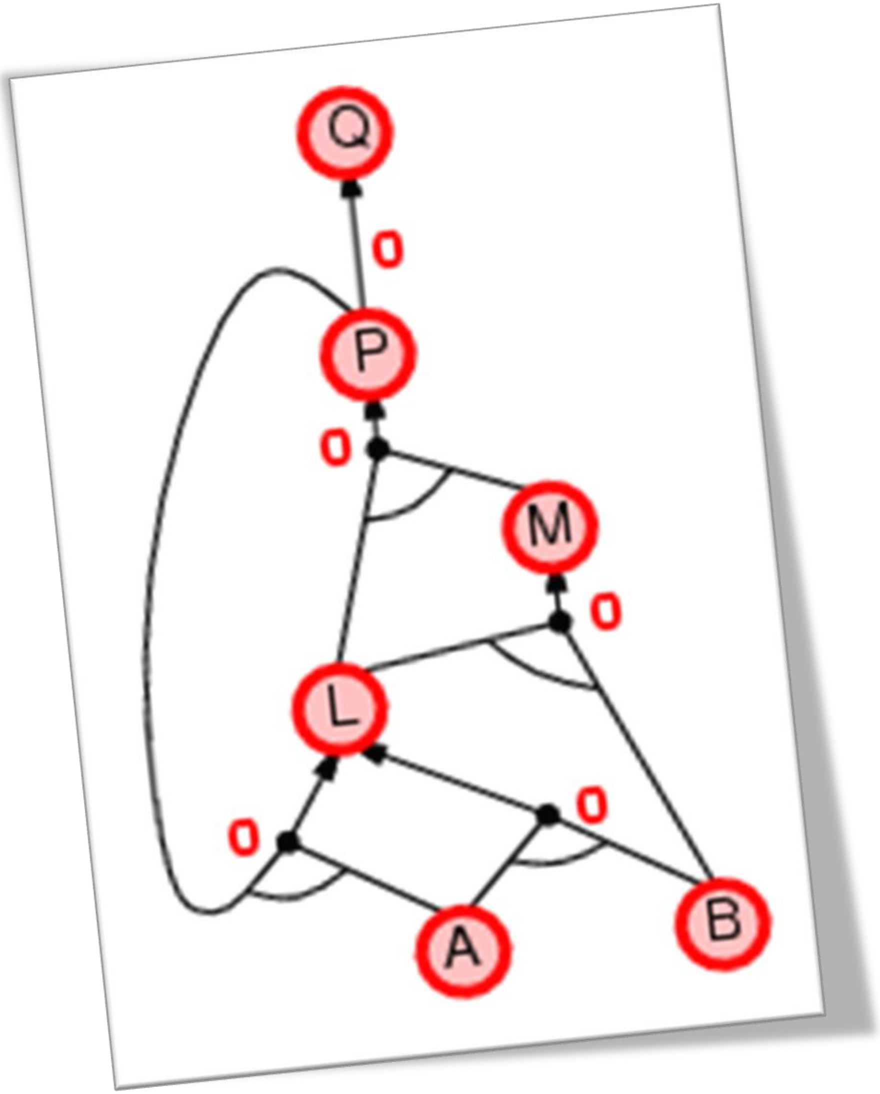

agenda = {Q}

inferred = [true, true, true, true, true, false]

| | A | B | L | M | P | Q |
|---|---|---|---|---|---|---|

| | |
|---|---|
| $P \Rightarrow Q$ | count($P \Rightarrow Q$) = 0 |
| $L \wedge M \Rightarrow P$ | count($L \wedge M \Rightarrow P$) = 0 |
| $B \wedge L \Rightarrow M$ | count($B \wedge L \Rightarrow M$) = 0 |
| $A \wedge P \Rightarrow L$ | count($A \wedge P \Rightarrow L$) = 0 |
| $A \wedge B \Rightarrow L$ | count($A \wedge B \Rightarrow L$) = 0 |
| A | A |
| B | B |

# استنتاج روبه‌جلو – مثال




agenda = {Q}

inferred = [true, true, true, true, true, false]

| | | | | | | |
|---|---|---|---|---|---|---|
| | A | B | L | M | P | Q |

| | |
|---|---|
| $P \Rightarrow Q$ | count($P \Rightarrow Q$) = 0 |
| $L \wedge M \Rightarrow P$ | count($L \wedge M \Rightarrow P$) = 0 |
| $B \wedge L \Rightarrow M$ | count($B \wedge L \Rightarrow M$) = 0 |
| $A \wedge P \Rightarrow L$ | count($A \wedge P \Rightarrow L$) = 0 |
| $A \wedge B \Rightarrow L$ | count($A \wedge B \Rightarrow L$) = 0 |
| A | A |
| B | B |

# ویژگی‌های الگوریتم زنجیره‌ای روبه‌جلو

- زنجیره‌سازی روبه‌جلو، یک نوع استدلال مبتنی‌بر داده است و می‌توان از آن برای استخراج دانش از روی ادراکات عامل استفاده کرد.
- زمان؟
  - زمان خطی نسبت به اندازه پایگاه دانش
- صحت؟
  - اساس کار الگوریتم زنجیره‌ای روبه‌جلو، استفاده از Modus Ponens است که صحیح می‌باشد.
- کامل؟
  - بله، اثبات در اسلاید بعد

# اثبات کامل بودن الگوریتم زنجیره‌ای روبه‌جلو

- حالت پایانی inferred را درنظر بگیرید. (یعنی نقطه ثابتی (fixed point) که هیچ جمله اتمیک جدیدی به‌دست نمی‌آید.)
- حالت پایانی را به‌عنوان یک مدل $m$ درنظر بگیرید که در آن به نمادهای استنتاج شده true و به نمادهای دیگر false تخصیص داده شده است.
- هر بند معین در KB در $m$ نیز درست است
  - اگر $a_1 \wedge \ldots \wedge a_n \Rightarrow b$ غلط باشد آن‌گاه $a_1 \wedge \ldots \wedge a_n$ درست و $b$ غلط است. این موضوع با رسیدن به یک نقطه ثابت در تناقض است.
- بنابراین $m$ یک مدل از KB است.
- اگر $\mathrm{KB} \models q$، جمله اتمیک $q$ در هر مدل از KB شامل $m$ درست است.

# الگوریتم زنجیره‌ای روبه عقب

- ایده: از query شروع کن و به سمت عقب حرکت کن.

if q is known to be true then

  no work is needed

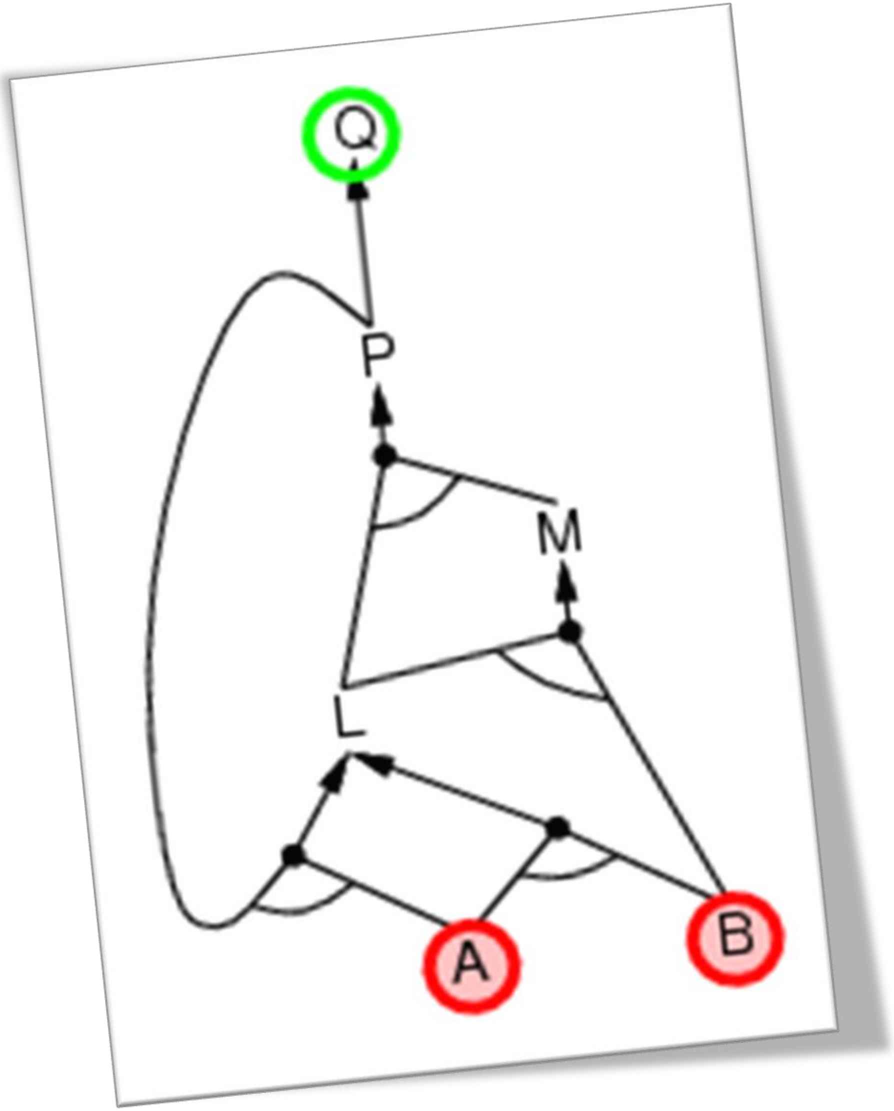
else

  find those rule concluding q

  if all premises of one of them can be proved (by backward chaining) then q is true

- اجتناب از حلقه‌ها: بررسی آن‌که آیا زیرهدف جدید قبلا بر روی پشته هدف بوده است.
- اجتناب از کار تکراری: بررسی آن‌که آیا قبلا درستی زیرهدف جدید اثبات شده یا با شکست روبه‌رو شده است.

# استنتاج روبه‌عقب – مثال




$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

$B \wedge L \Rightarrow M$

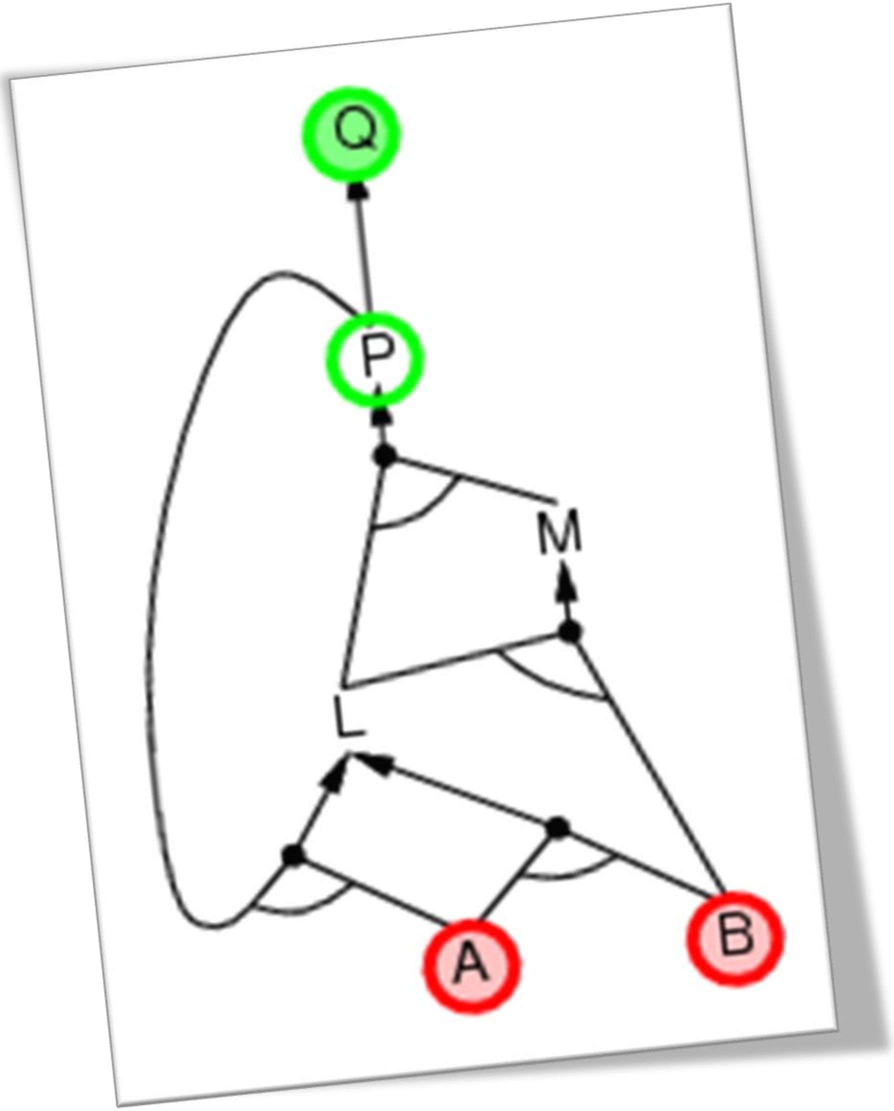
$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

$A$

$B$

# استنتاج روبه‌عقب – مثال




$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

$B \wedge L \Rightarrow M$

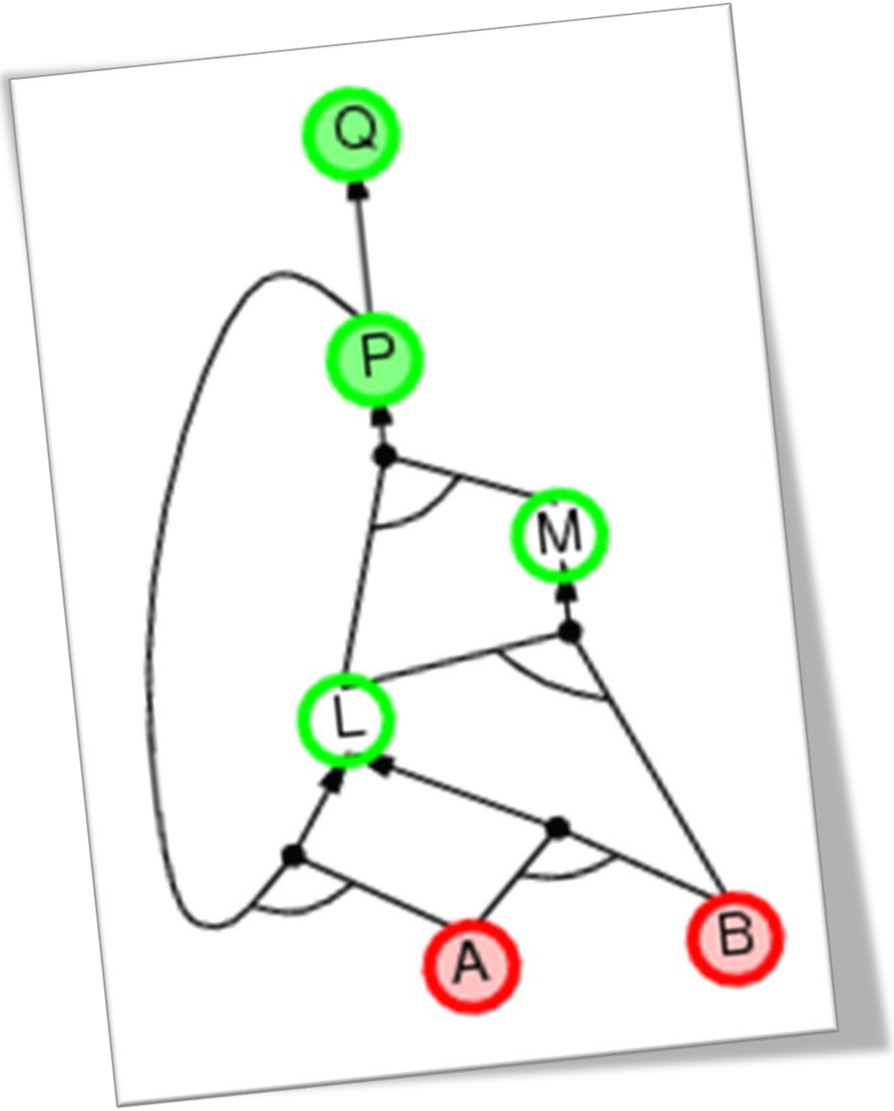
$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

$A$

$B$

# استنتاج روبه‌عقب – مثال




$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

$B \wedge L \Rightarrow M$

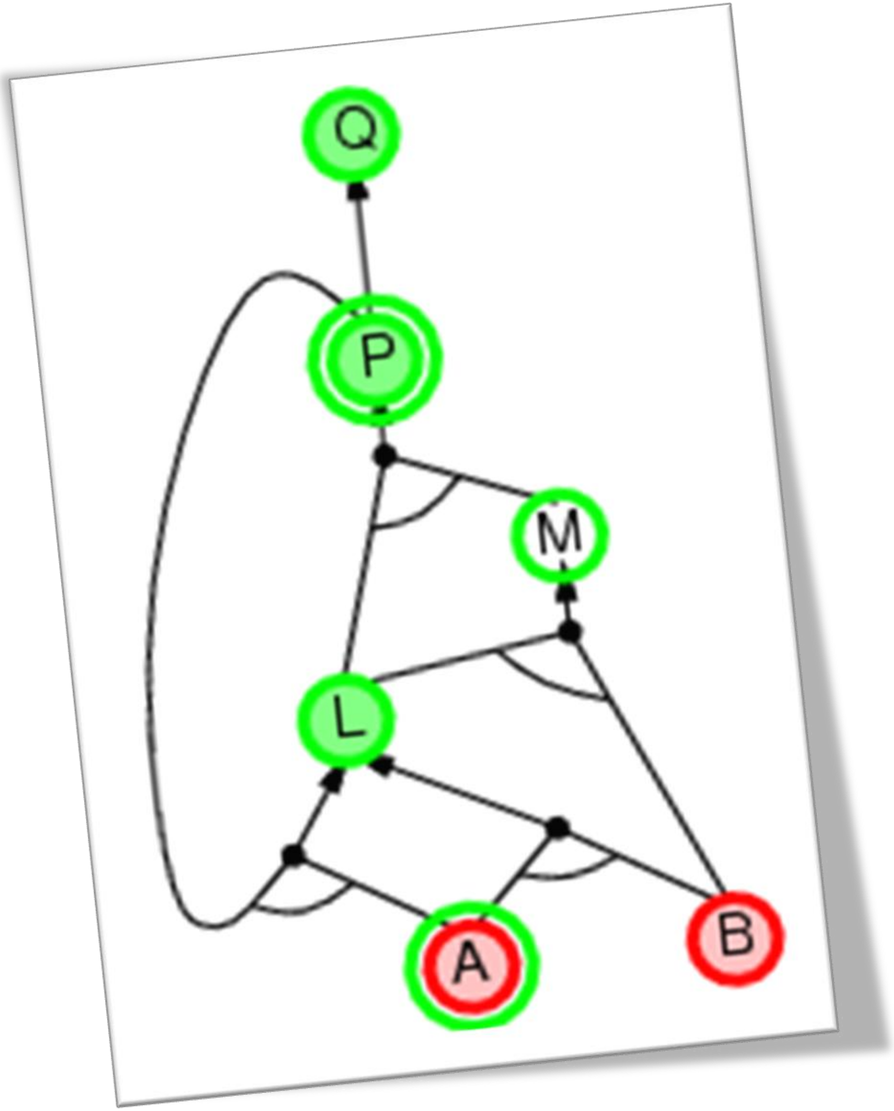
$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

$A$

$B$

# استنتاج روبه‌عقب – مثال




$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

$B \wedge L \Rightarrow M$

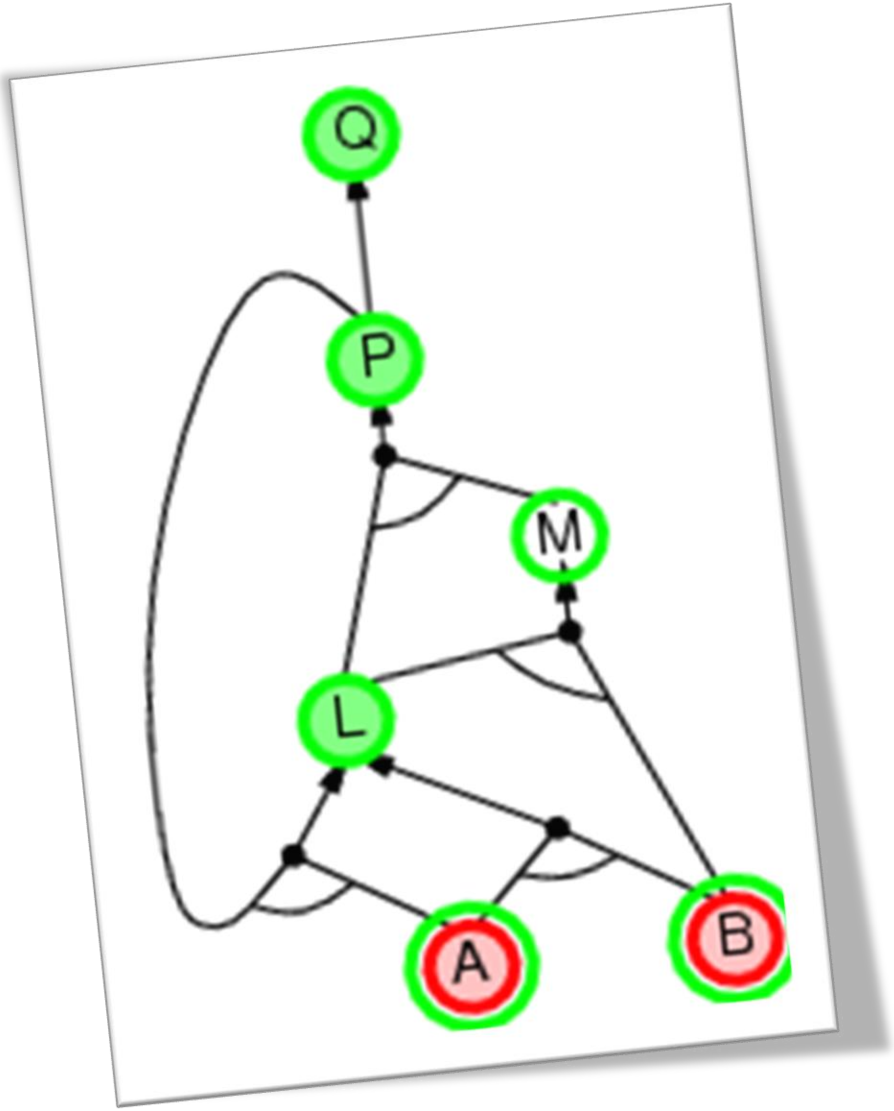
$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

$A$

$B$

# استنتاج روبه‌عقب – مثال




$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

$B \wedge L \Rightarrow M$

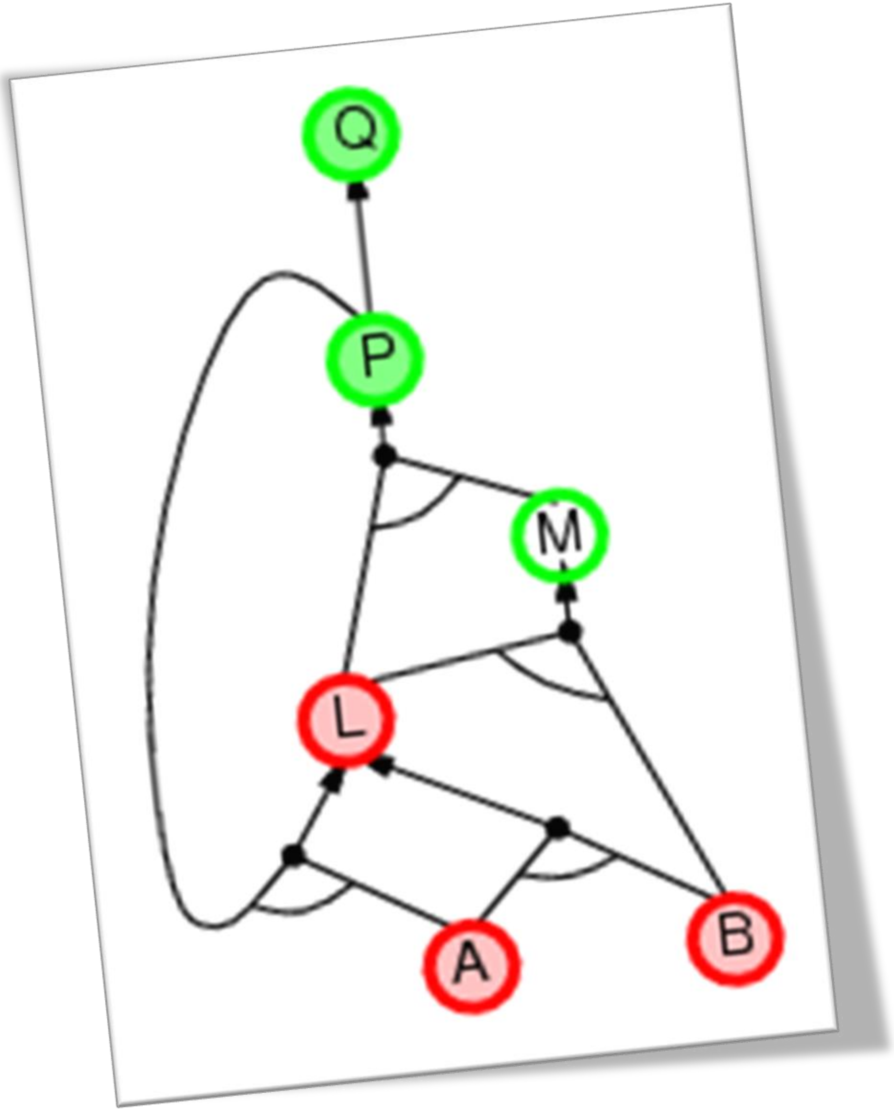
$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

$A$

$B$

# استنتاج روبه‌عقب – مثال




$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

$B \wedge L \Rightarrow M$

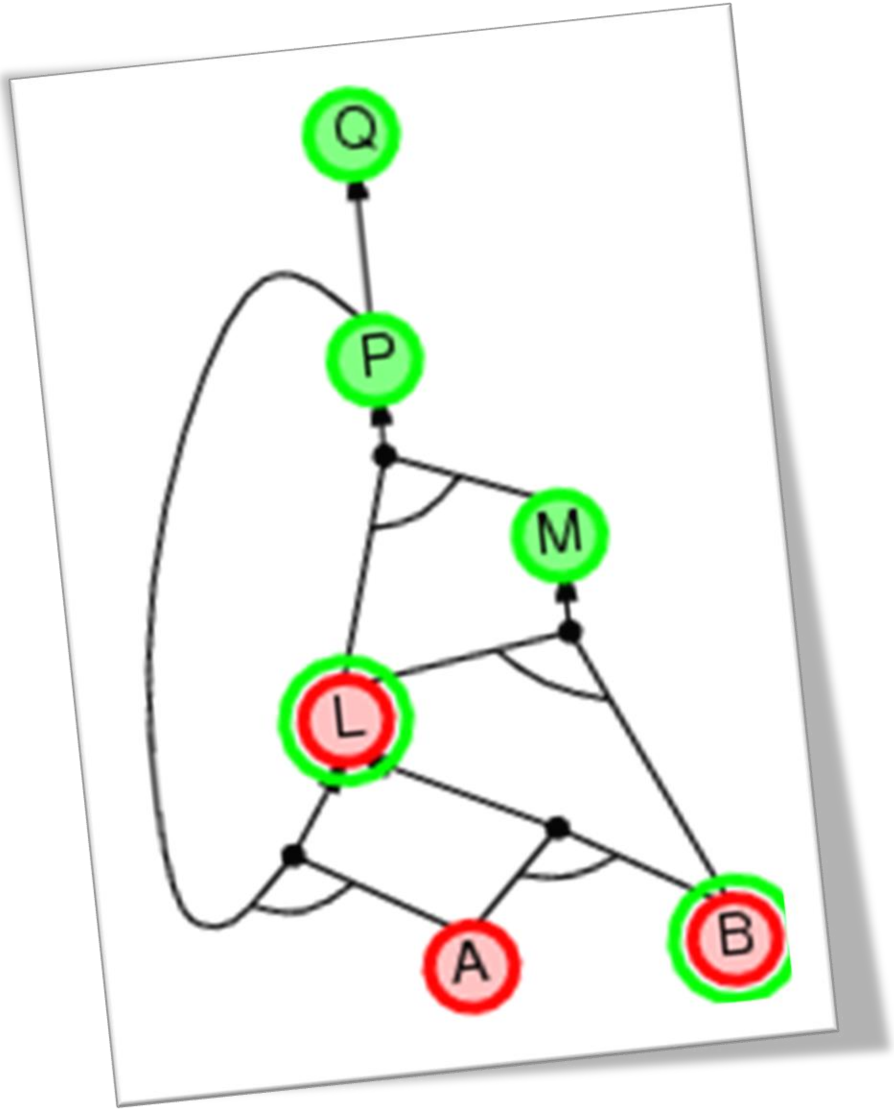
$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

$A$

$B$

# استنتاج روبه‌عقب – مثال




$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

$B \wedge L \Rightarrow M$

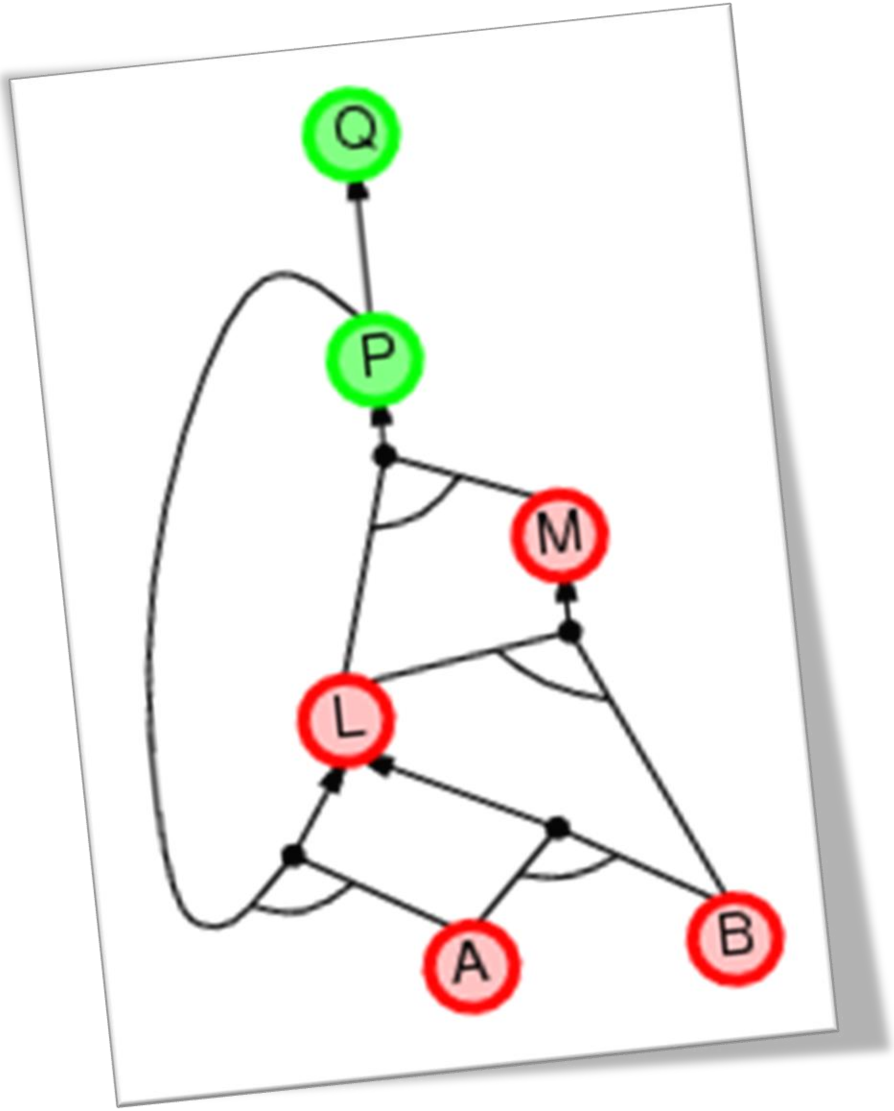
$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

$A$

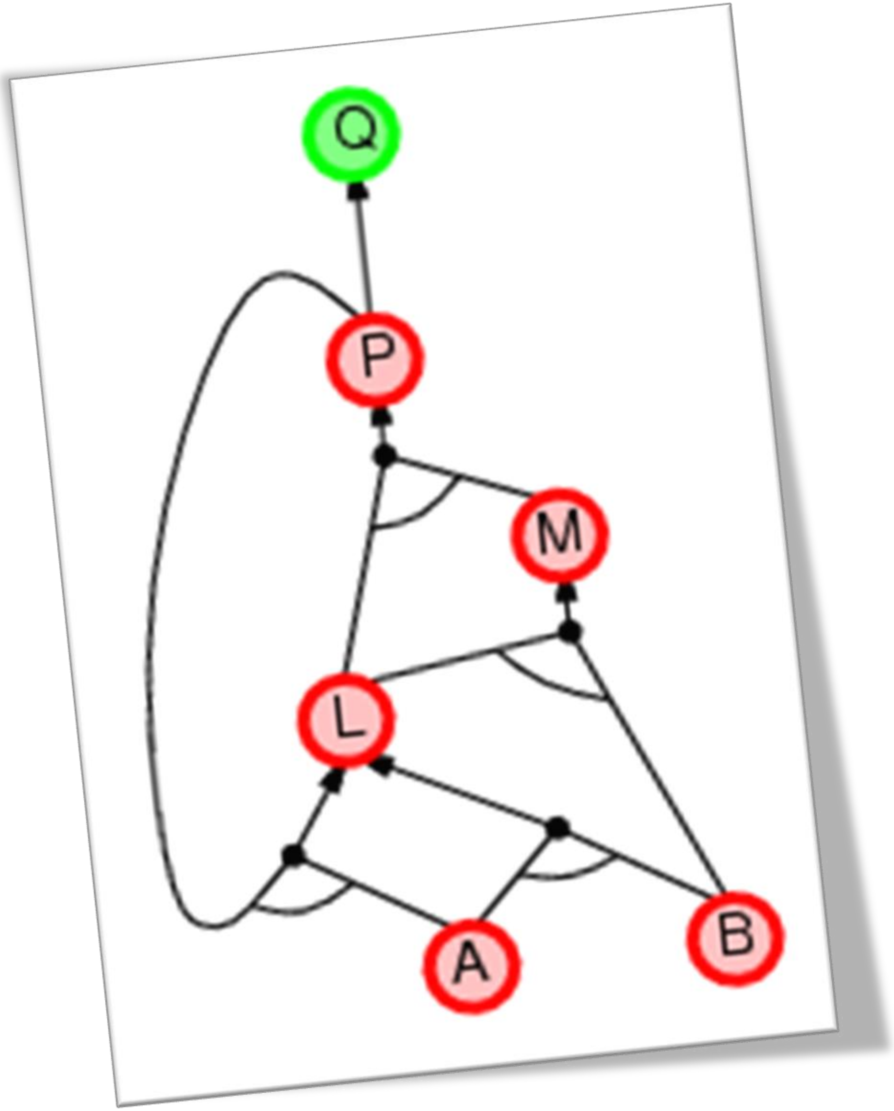
$B$

# استنتاج روبه‌عقب – مثال




$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$B \wedge L \Rightarrow M$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$A$$
$$B$$

# استنتاج روبه‌عقب – مثال




$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

$B \wedge L \Rightarrow M$

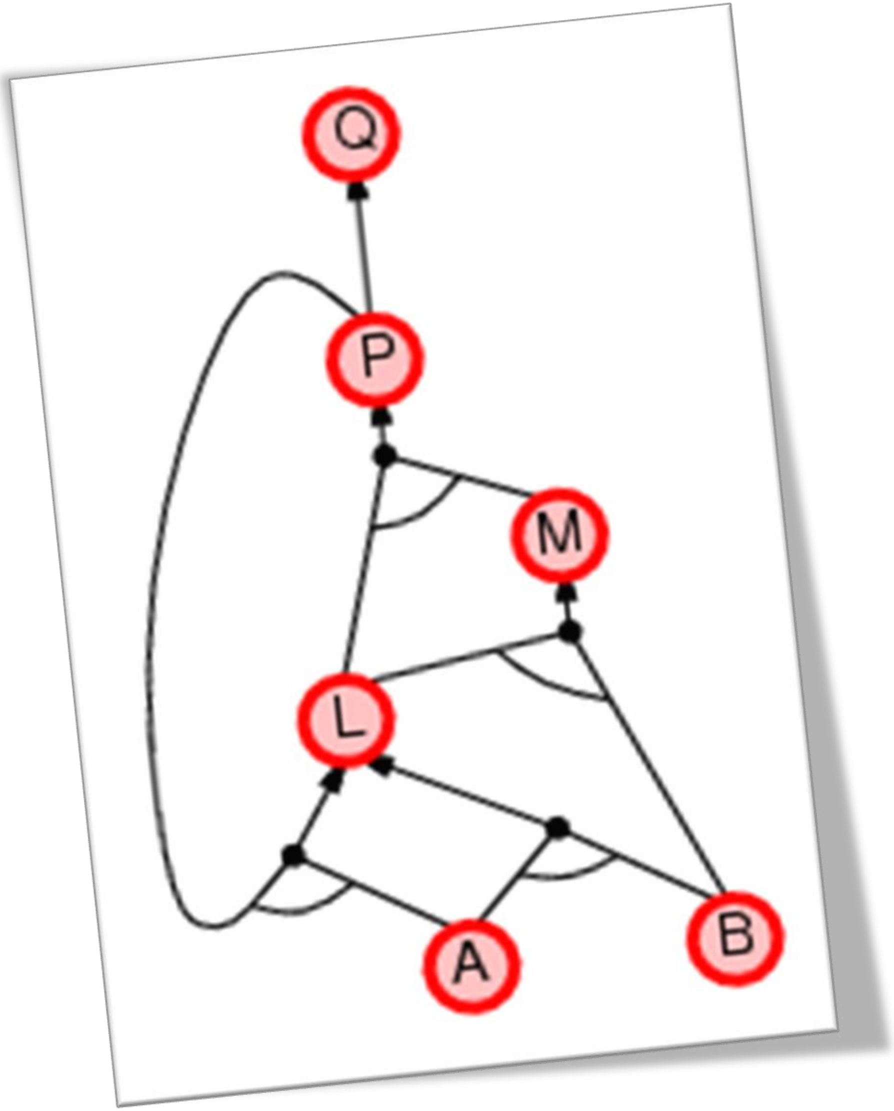
$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

$A$

$B$

# استنتاج روبه‌عقب – مثال




$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

$A$

$B$

# ویژگی‌های الگوریتم زنجیره‌ای روبه عقب

- الگوریتم زنجیره ای روبه عقب یک الگوریتم مبتنی‌بر هدف است.
- زمان اجرای زنجیره‌سازی روبه عقب مانند زنجیره‌سازی روبه‌جلو، برحسب اندازه پایگاه دانش خطی است. اما چون فقط واقعیت‌های مرتبط با هدف را بررسی می‌کند، اغلب سریع‌تر از آن عمل می‌کند.

# استنتاج گزاره‌ای کارا

# استنتاج گزاره‌ای کارا

- هدف بررسی دو خانواده از الگوریتم‌های کارا در استنتاج گزاره‌ای مبتنی‌بر وارسی مدل است.
  - الگوریتم‌های کامل جستجوی عقبگرد
    - الگوریتم DPLL (Davis, Putnam, Logemann, Loveland)
  - الگوریتم ناکامل جستجوی محلی
    - الگوریتم WalkSAT
- این الگوریتم‌ها برای تعیین ارضاپذیری یک جمله استفاده می‌شوند.
  - برای تعیین آن‌که $KB \models \alpha$ یا خیر می‌توان ناسازگار بودن $KB \wedge \neg \alpha$ را بررسی کرد.

# الگوریتم DPLL

- ورودی DPLL یک جمله به شکل نرمال عطفی (CNF) یعنی مجموعه‌ای از بندها (clauses) است.
- اساسا مشابه با Backtracking-search و ?TT-Entail یک برشماری بازگشتی و اول عمق از مدل‌های ممکن انجام می‌دهد.
- بهبود آن نسبت به این روش‌ها به‌دلیل استفاده از سه هیوریستیک زیر برای تعیین امیدبخش یا غیر امیدبخش بودن گره‌ها در فضای جستجو است:
  - خاتمه زود هنگام (*Early termination*)
  - هیوریستیک نماد محض (*Pure symbol heuristic*)
  - هیوریستیک بند واحد (*Unit clause heuristic*)

# الگوریتم DPLL– هیوریستک‌ها

- خاتمه زود هنگام
  - یک بند true است اگر حداقل یکی از لیترال‌های آن true باشد.
    - $(A \vee B) \wedge (A \vee C)$ در صورت درست بودن A صرف نظر از مقادیر B و C درست است
    - $(\neg A \vee B \vee \neg C)$ در صورت نادرست بودن C صرف نظر از مقادیر A و B درست است
  - یک جمله false است اگر حداقل یکی از بندهای آن false باشد.
    - $(A \vee B) \wedge (A \vee C)$ در صورت نادرست بودن A و B صرف نظر از مقادیر B و C نادرست است
  - خاتمه زودهنگام از آزمایش کل زیردرخت‌ها در فضای جستجو جلوگیری می‌کند.

# الگوریتم DPLL– هیوریستک‌ها

- هیوریستیک نماد محض
  - نماد محض: نمادی که در تمام بندها با یک علامت ظاهر شود.
    - در دو بند $(C \vee A \vee B), (\neg B \vee \neg C)$ نماد A محض و B و C غیر محض است.
    - در سه بند $(C \vee A), (\neg B \vee \neg C), (A \vee \neg B)$ نمادهای A و B محض و C غیر محض است.
  - اگر جمله‌ای که به فرم CNF است ارضاشدنی باشد حتما برای آن مدلی وجود دارد که در آن مقدار نمادهای محض مثبت برابر true و مقدار نمادهای محض منفی برابر false است.
  - در تعیین نماد محض از بندهایی که قبلا با مقدار true شناخته شده‌اند صرف نظر می‌شود.
  - در عبارت بالا با تعیین B=false بندهای $(\neg B \vee \neg C)$ و $(A \vee \neg B)$ مقدار true پیدا می‌کند و در نتیجه C نماد محض خواهد بود.

# الگوریتم DPLL– هیوریستک‌ها

- هیوریستیک بند واحد
  - بند واحد: تنها شامل یک لیترال می‌باشد. یا بندی است که تمام لیترال‌های آن غیر از یک لیترال، تا کنون مقدار False گرفته باشد.
  - در این صورت، تنها لیترال موجود در یک بند واحد باید True باشد تا بند نیز True گردد.
  - انتساب به یک بند واحد می‌تواند بند واحد دیگری تولید کند به این کار انتشار واحد می‌گویند. ( Unit propagation)
    - در دو بند $B, (C \vee A \vee \neg B)$ نماد B یک بند واحد است که به آن باید مقدار true داد
    - و اگر تا کنون C مقدار false گرفته باشد A نیز یک بند واحد خواهد شد.

- هیوریستیک‌های دیگری نیز وجود دارد (صفحه ۲۶۱ کتاب مرجع ویرایش ۳).
  - هیوریستیک درجه: انتخاب متغیری که در تعداد بندهای باقی‌مانده‌ی بیشتری ظاهر شده باشد.
  - آنالیز مؤلفه‌های مجزا: مجموعه بندهای مجزا یعنی آن‌هایی که هیچ متغیر مشترکی ندارند.

# الگوریتم DPLL

```
function DPLL-SATISFIABLE?(s) returns true or false
   inputs: s, a sentence in propositional logic

   clauses ← the set of clauses in the CNF representation of s
   symbols ← a list of the proposition symbols in s
   return DPLL(clauses, symbols, { })
```

---

```
function DPLL(clauses, symbols, model) returns true or false

   if every clause in clauses is true in model then return true
   if some clause in clauses is false in model then return false
   P, value ← FIND-PURE-SYMBOL(symbols, clauses, model)
   if P is non-null then return DPLL(clauses, symbols – P, model ∪ {P=value})
   P, value ← FIND-UNIT-CLAUSE(clauses, model)
   if P is non-null then return DPLL(clauses, symbols – P, model ∪ {P=value})
   P ← FIRST(symbols); rest ← REST(symbols)
   return DPLL(clauses, rest, model ∪ {P=true}) or
          DPLL(clauses, rest, model ∪ {P=false}))
```

# تست ۳

می‌خواهیم مسئله ارضاپذیری زیر را با استفاده از الگوریتم DPLL حل کنیم. اگر انتساب مقدار «صفر» به متغیرها، بر انتساب مقدار «یک» به آن‌ها اولویت داشته باشد، از کدام یک از دو تکنیک Unit Clause (UC) و Pure Literal (PL)، در حل این مسئله خاص استفاده خواهد شد؟
(مهندسی کامپیوتر دولتی ۹۴)

✓ ۱) فقط از UC استفاده می‌شود.

۲) فقط از PL استفاده می‌شود.

۳) هم از UC هم از PL استفاده می‌شود.

۴) از UC و یا PL استفاده نمی‌شود.

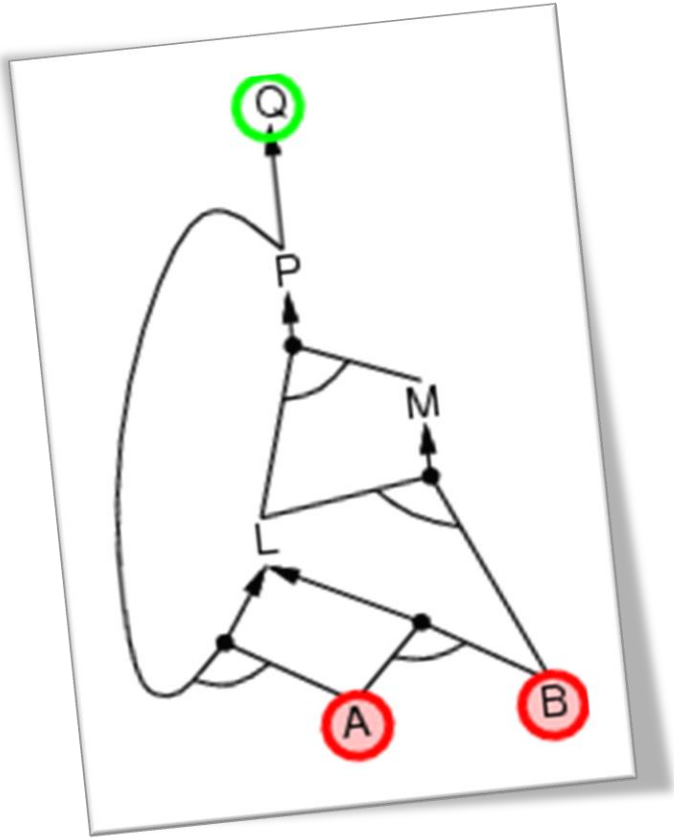
$$\{\neg A \vee B \vee C\}, \{A \vee \neg B \vee C\}, \{A \vee B \vee \neg C\}, \{A \vee B \vee C\}$$

$$\{\neg B \vee C\}, \quad \{B \vee \neg C\}, \quad \{B \vee C\}$$

$$\{\neg C\}, \quad \{C\}$$

# الگوریتم DPLL و زنجیره‌سازی روبه‌جلو

- تمرین: بررسی کنید که اگر عبارت CNF فقط شامل جملات هورن باشد آن‌گاه DPLL دقیقا مانند زنجیره پیش‌رو عمل می‌کند.
  - برای مثال رفتار DPLL در این پایگاه دانش متناظر با رفتار FC است.




$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

$A$

$B$

# الگوریتم DPLL و زنجیره‌سازی روبه‌جلو

| | $\neg Q$ | $A$ | $B$ | $\neg P \vee Q$ | $\neg L \vee \neg M \vee P$ | $\neg B \vee \neg L \vee M$ | $\neg A \vee \neg P \vee L$ | $\neg A \vee \neg B \vee L$ |
|---|---|---|---|---|---|---|---|---|
| A=T | | | | | | | $\neg P \vee L$ | $\neg B \vee L$ |
| B=T | | | | | | $\neg L \vee M$ | | $L$ |
| L=T | | | | | $\neg M \vee P$ | $M$ | True | |
| M=T | | | | | $P$ | | | |
| P=T | | | | $Q$ | | | | |
| Q=T | False | | | | | | | |

# الگوریتم جستجوی محلی WalkSat

- تابع ارزیابی: هیوریستیک حداقل تناقض (min-conflict) یعنی تعداد بندهایی که در مدل فعلی True نمی‌شود.
- در هر تکرار یک بند ارضا نشده را انتخاب می‌کند و یک نماد آن را برمی‌گزیند تا مقدار آن را عکس کند.
- برای انتخاب یک نماد به صورت تصادفی از یکی از دو راه زیر استفاده می‌کند:
  - یک گام حداقل تناقض که تعداد بندهای ارضا نشده در حالت جدید را حداقل می‌کند.
  - یک گام راه رفتن تصادفی که نمادها را به صورت تصادفی انتخاب می‌کند. این روش سعی می‌کند از گیر افتادن در نقطه بهینه محلی جلوگیری کند.

# الگوریتم جستجوی محلی WalkSat

**function** WALKSAT(*clauses*, *p*, *max_flips*) **returns** a satisfying model or *failure*
  **inputs**: *clauses*, a set of clauses in propositional logic
    *p*, the probability of choosing to do a “random walk” move, typically around 0.5
    *max_flips*, number of flips allowed before giving up

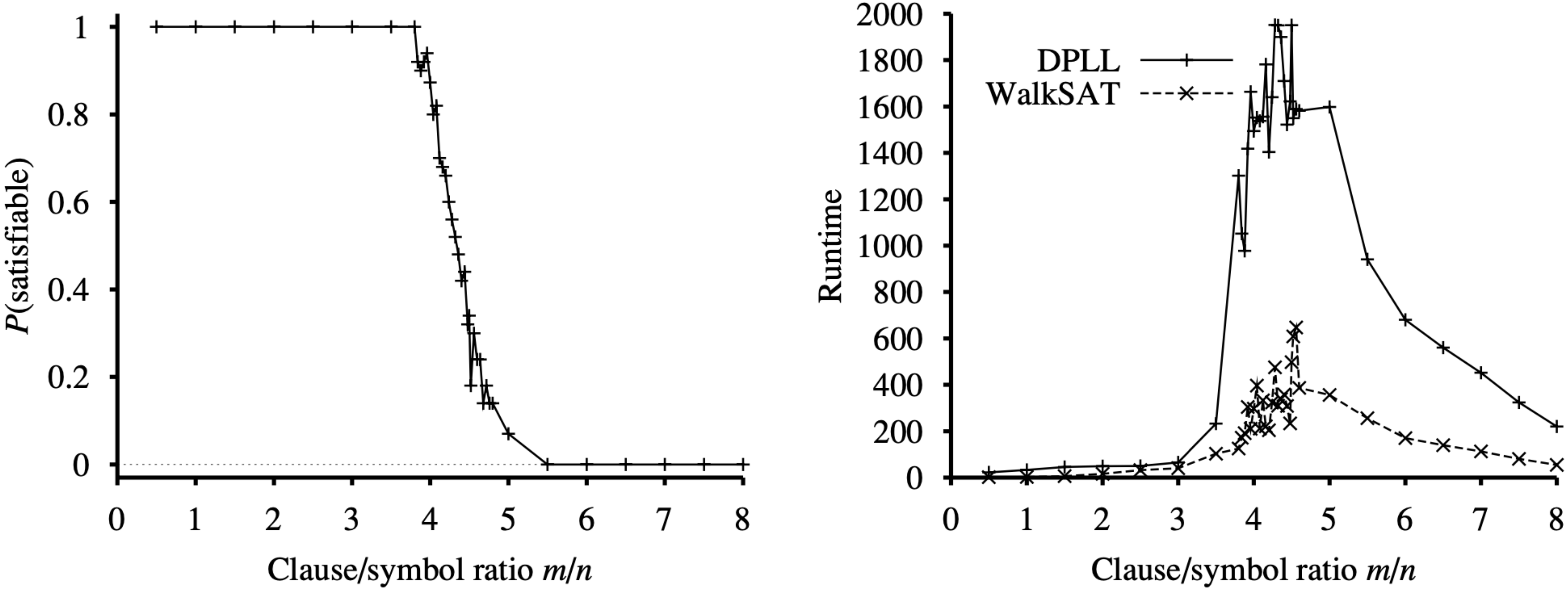
  *model* ← a random assignment of *true*/*false* to the symbols in *clauses*
  **for** $i = 1$ **to** *max_flips* **do**
    **if** *model* satisfies *clauses* **then return** *model*
    *clause* ← a randomly selected clause from *clauses* that is false in *model*
    **with probability** *p* flip the value in *model* of a randomly selected symbol from *clause*
    **else** flip whichever symbol in *clause* maximizes the number of satisfied clauses
  **return** *failure*

# الگوریتم جستجوی محلی WalkSat

- با توجه به نحوه‌ی انتخابش، تعادلی مناسب میان میزان حریصانه بودن و تصادفی بودن برقرار می‌کند.
- اگر الگوریتم مدلی را بازگرداند جمله ورودی ارضاپذیر است ولی اگر failure را بازگرداند نمی‌توانیم تعیین کنیم که آن جمله ارضاناپذیر است.
  - اگر جواب وجود داشته باشد و max-flips مقدار بینهایت داشته باشد و $p>0$ الگوریتم یک مدل را برمی‌گرداند.
  - اگر جمله ناسازگار باشد و max-flips مقدار بینهایت داشته باشد الگوریتم هیچگاه پایان نمی‌پذیرد.
  - باید یک مقدار محدود برای max-flips مشخص کرد.

# میزان سختی مسائل SAT

- جملات 3-CNF(m,n) (هر عبارت ۳ لیترال دارد).
  - n=50 تعداد متغیرها و m تعداد عبارات




(a) (b)

# Wumpus world example: propositional symbols

- Atemporal variables that may be needed
  - $P_{i,j}$ is true if there is a pit in $[i,j]$.
  - $W_{i,j}$ is true if there is a wumpus in $[i,j]$, dead or alive.
  - $B_{i,j}$ is true if the agent perceives a breeze in $[i,j]$.
  - $S_{i,j}$ is true if the agent perceives a stench in $[i,j]$.

- Some of the temporal variables that may be needed
  - Variables for percepts and actions
  - $L_{i,j}^t, FacingEast^t, FacingWest^t, FacingNorth^t, FacingSouth^t$
  - $HaveArrow^t$
  - $WumpusAlive^t$

# Wumpus world example: axioms on atemporal aspect of the world

- Axioms: general knowledge about how the world works
- General rules on atemporal variables

$$\neg P_{1,1}$$

$$\neg W_{1,1}$$

$$B_{x,y} \Leftrightarrow (P_{x,y+1} \vee P_{x,y-1} \vee P_{x+1,y} \vee P_{x-1,y})$$

$$S_{x,y} \Leftrightarrow (W_{x,y+1} \vee W_{x,y-1} \vee W_{x+1,y} \vee W_{x-1,y})$$

A distinct rule is needed for each square

$$W_{1,1} \vee W_{1,2} \vee \ldots \vee W_{4,4}$$

$$\neg W_{1,1} \vee \neg W_{1,2}$$

$$\neg W_{1,1} \vee \neg W_{1,3}$$

...

# Wumpus world example: perceptions & actions

- Perceptions are converted to facts and are Telled to KB
  - $MAKE\_PERCEPT\_SENTENCE([Breeze, Stench, None, None, None], t)$
  - $Breeze^t, Stench^t$ are added to KB

- Selected action is converted to a fact and is Telled to KB
  - $MAKE\_ACTION\_SENTENCE(Forward, t)$
  - $Forward^t$ is added to KB

# Wumpus world example: transition model

- Changing aspect of world (Temporal variables)
  - They are also called as fluents or state variables
- Initial KB includes the initial status of some of the temporal variables:
  - $L_{1,1}^0, FacingEast^0, HaveArrow^0, WumpusAlive^0$
- Percepts are connected to fluents describing the properties of squares
  - $L_{x,y}^t \Rightarrow (Breeze^t \Leftrightarrow B_{x,y})$
  - $L_{x,y}^t \Rightarrow (Stench^t \Leftrightarrow S_{x,y})$
  - …
- Transition model: fluents can change as the results of agent's actions

# Wumpus world example: transition model Frame problem

- Transition model:

  $L_{1,1}^0 \land FacingEast^0 \land Forward^0 \Rightarrow (L_{2,1}^1 \land \neg L_{1,1}^1)$

  $L_{1,1}^0 \land FacingEast^0 \land TurnLeft^0 \Rightarrow (L_{1,1}^1 \land FacingNorth^1 \land \neg FacingEast^1)$

  …

- After doing $Forward$ action at time 0:
  - $ASK(KB, HaveArrow^1) = false$
    - Why?

- Frame problem: representing the effects of actions without having to represent explicitly a large number of intuitively obvious non-effects

# Wumpus world example
## Solution to frame problem: successor-state axioms

- <u>A Solution to frame problem</u>: Successor-state axioms

  Instead of writing axioms about actions, we write axioms about fluents

  - $F^{t+1} \Leftrightarrow ActionCausesF^t \vee (F^t \wedge \neg ActionCausesNotF^t)$

$$HaveArrow^{t+1} \Leftrightarrow (HaveArrow^t \wedge \neg Shoot^t)$$

$$\begin{aligned} L_{1,1}^{t+1} \Leftrightarrow & (L_{1,1}^t \wedge (\neg Forward^t \vee Bump^{t+1})) \\ & \vee (L_{1,2}^t \wedge (FacingSouth^t \wedge Forward^t)) \\ & \vee (L_{2,1}^t \wedge (FacingWest^t \wedge Forward^t)) \end{aligned}$$

# Limitations of propositional logic

- Wumpus world
  - Distinct rules "for each square [x,y]" and "for each time t"
  - $L_{x,y}^{t} \wedge FacingEast^{t} \wedge TurnLeft^{t} \Rightarrow (L_{x,y}^{t+1} \wedge FacingUp^{t+1} \wedge \neg FacingEast^{t+1})$
- Many rules are needed to describe the world axioms
  - Rather impractical for bigger world
- We can not express general knowledge about "physics" of the world directly in propositional language
  - More expressive language is needed.
- Qualification problem: specifying all the exceptions
  - Probability theory allows us to summarize all the exceptions (without explicitly naming them)