

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ



هوش مصنوعی

فصل ۷

عوامل‌های منطقی

Logical Agents

کاظم فولادی قلعه
دانشکده مهندسی، پردیس فارابی
دانشگاه تهران

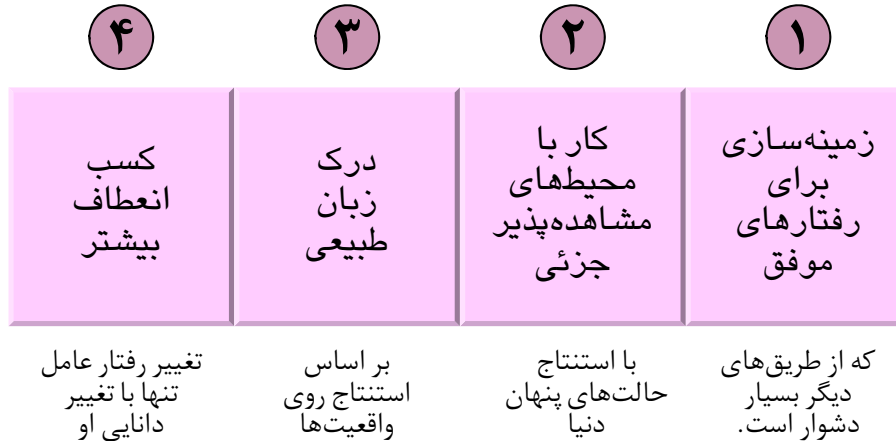
<http://courses.fouladi.ir/ai>

هوش مصنوعی و دانایی

AI AND KNOWLEDGE

اهمیت «استدلال»

چرا استدلال در هوش مصنوعی مهم است؟



البته منطق محدودیت‌هایی هم دارد: مثل کار با دانش غیرمطمئن

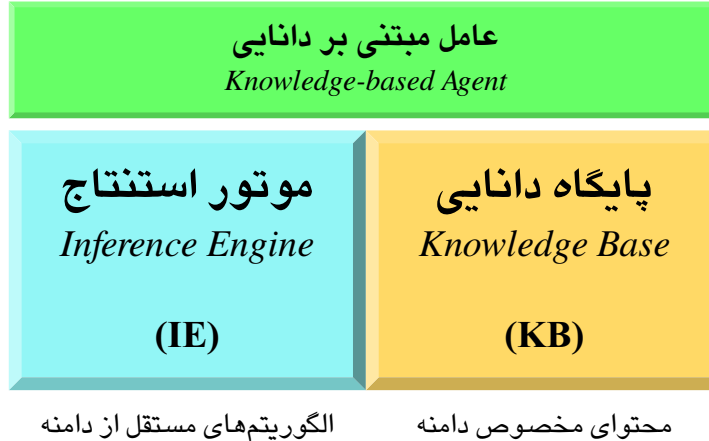
عوامل‌های منطقی

۱

عوامل‌های
مبتنی بر
دانشی

عامل مبتنی بر دانایی

ساختار



عامل مبتنی بر دانایی

پایگاه دانایی

مجموعه‌ای از جمله‌ها در یک زبان صوری

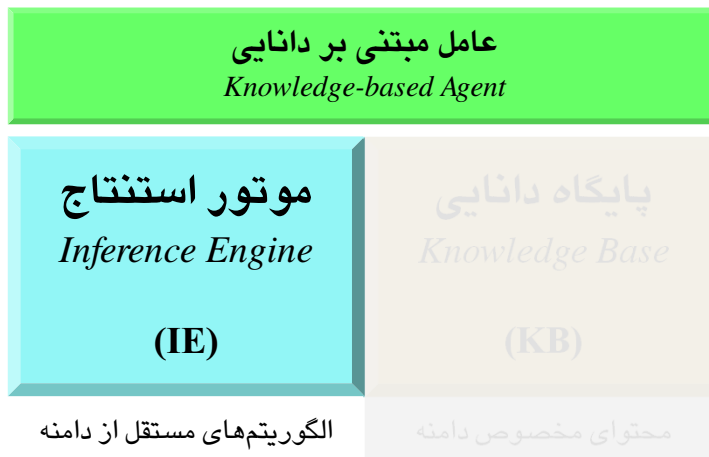
↓

زبان بازنمایی دانایی

جزء اصلی یک عامل مبتنی بر دانایی، پایگاه دانایی آن می‌باشد.

عامل مبتنی بر دانایی

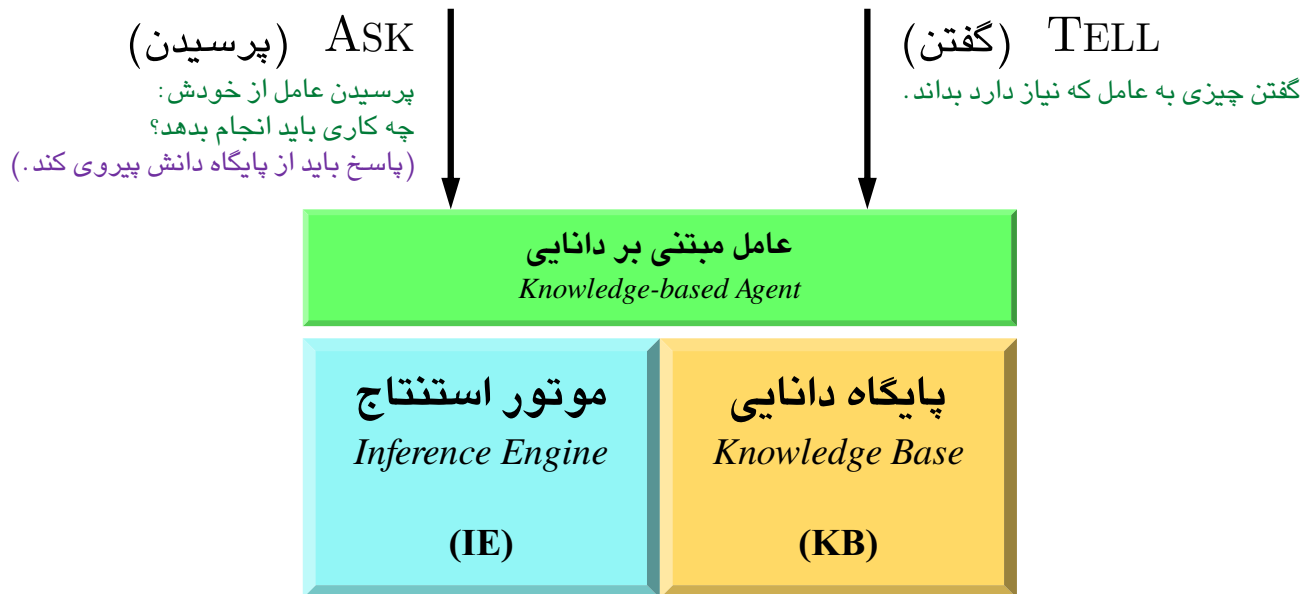
موتور استنتاج



الگوریتم استخراج **دانایی جدید** از دانایی فعلی

عامل مبتنی بر دانایی

عملیات: گفتن و پرسیدن



عامل مبتنی بر دانایی

سطوح طراحی عامل مبتنی بر دانایی

عامل‌های مبتنی بر دانایی

Knowledge-based Agents

آنچه عامل می‌داند، صرف نظر از چگونگی پیاده‌سازی
(جملات خام، مثلاً: «پل P بین جزیره A و B قرار دارد»)

سطح دانایی

Knowledge Level

به صورت تعدادی از جملات کدگذاری شده
(مثلاً: $\text{Links}(P, A, B)$)

سطح منطقی

Logical Level

سطح قابل اجرا در معماری عامل
(بیت‌های کد شده، ساختمان داده‌ها و ...)

سطح پیاده‌سازی

Implementation Level

روی‌کرد اعلانی (declarative) برای ساخت یک عامل: TELL

عامل مبتنی بر دانایی

شبه کد

```
function KB-AGENT(percept) returns an action
  static: 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
```

بسیار مشابه با عامل واکنشی با حالت داخلی (عامل مبتنی بر مدل)

عامل مبتنی بر دانایی

توانایی‌های لازم



عوامل‌های منطقی

۲

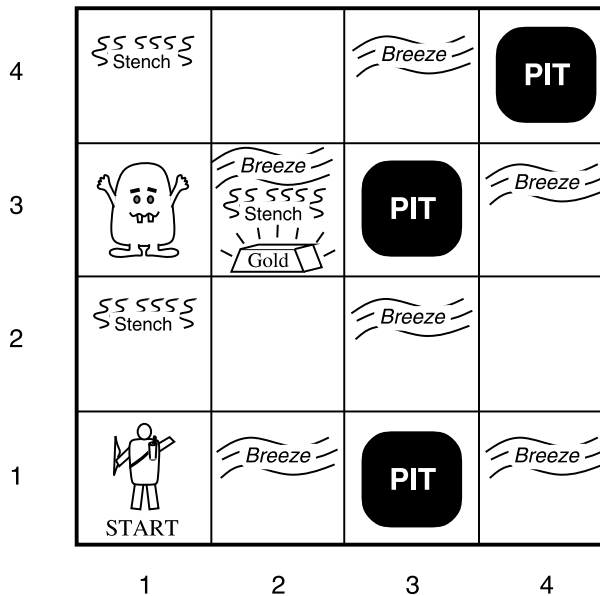
دنیای
اژدها

(Wumpus World)

عامل مبتنی بر دانایی

مثال: دنیای اژدها

WUMPUS WORLD



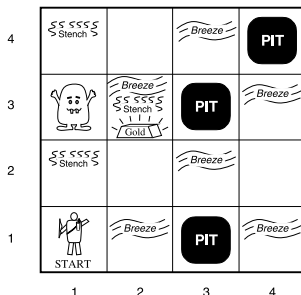
عامل مبتنی بر دانایی

مثال: دنیای اژدها

WUMPUS WORLD

P	E	A	S
معيار کارآیی	محیط	کنش‌گرها	حسگرها
Performace Measure	Environment	Actuators	Sensors

کشف طلا: +1000
مرگ: -1000
هر گام: -1
استفاده از تیر: -10



بو در خانه‌های مجاور اژدها
نسیم در خانه‌های مجاور چاله
درخشش طلا در خانه‌ی طلا
کشتن اژدها با شلیک اگر در
مقابل تیر باشد (کنش Shoot)
فقط یک تیر موجود برای شلیک
امکان برداشتن طلا فقط در
خانه‌ی آن (کنش Grab)
رها کردن طلا موجب افتادن در
همان خانه (کنش Release)
شنیده شدن صدای ناله‌ی
اژدها (Scream) در همه‌ی
محیط در صورت زخمی شدن
شنیده شدن صدای Bump در
صورت برخورد عامل با دیوار
مرگ عامل در صورت ورود به
خانه‌ی اژدهای سالم یا چاله
وجود چاله در یک خانه با

احتمال 0.2

انجام کنش‌های
گردش به راست
(Right Turn)
گردش به چپ
(Left Turn)
حرکت به جلو
(Forward)
برداشتن
(Grab)
رها کردن
(Release)
شلیک کردن
(Shoot)

نسیم، درخشش، بو
بردار ادراکی ۵ تایی:
بوی بد اژدها
(Stench)
نسیم
(Breeze)
درخشش
(Glitter)
صدای برخورد
(Bump)
صدای ناله
(Scream)

$$Percept = [Stench, Breeze, Glitter, Bump, Scream]$$

عامل مبتنی بر دانایی

مثال: دنیای اژدها: خصوصیات محیط وظیفه

PROPERTIES OF TASK ENVIRONMENTS

مشاهده‌پذیر کامل <i>Fully Observable</i>	تنها ادراک‌های محلی وجود دارند.	مشاهده‌پذیر جزئی <i>Partially Observable</i>
تک عاملی <i>Single-agent</i>	با اژدها به عنوان یک پدیده‌ی طبیعی و جزئی از محیط برخورد می‌شود.	چند عاملی <i>Multiagent</i>
قطعی <i>Deterministic</i>	برآمد همه‌ی کنش‌ها به طور قطعی مشخص شده است.	اتفاقی <i>Stochastic</i>
مقطعی <i>Episodic</i>	در سطح کنش‌ها دنباله‌ای است.	دنباله‌ای <i>Sequential</i>
ایستا <i>Static</i>	محیط شامل اژدها و چاله‌ها حرکت نمی‌کنند.	پویا <i>Dynamic</i>
گسسته <i>Discrete</i>	ادراک‌ها، کنش‌ها، زمان و حالت‌های مسئله، گسسته هستند.	پیوسته <i>Continuous</i>
شناخته‌شده <i>Known</i>	برآمد همه‌ی کنش‌ها داده شده است.	ناشناخته <i>Unknown</i>

عامل مبتنی بر دانایی

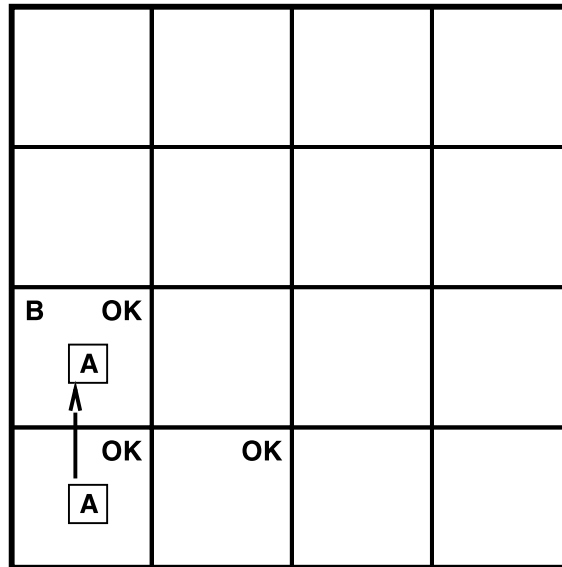
مثال: دنیای اژدها: اکتشاف در دنیای اژدها (۱ از ۸)

OK			
OK A	OK		

$Percept = [Stench, Breeze, Glitter, Bump, Scream] = [None, None, None, None, None]$

عامل مبتنی بر دانایی

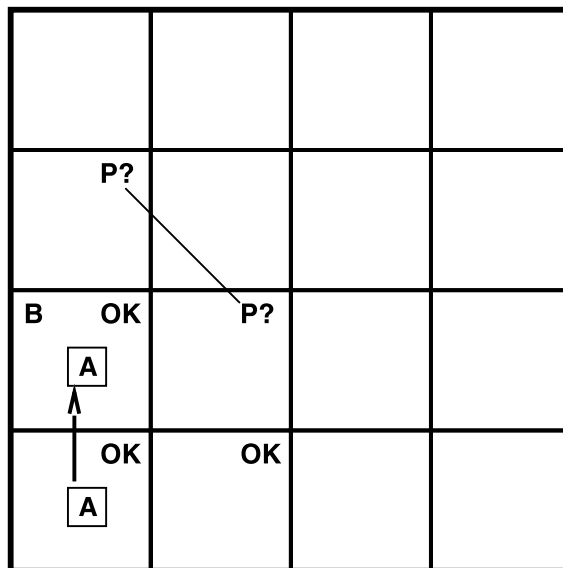
مثال: دنیای اژدها: اکتشاف در دنیای اژدها (۲ از ۸)



$Percept = [Stench, Breeze, Glitter, Bump, Scream] = [None, Breeze, None, None, None]$

عامل مبتنی بر دانایی

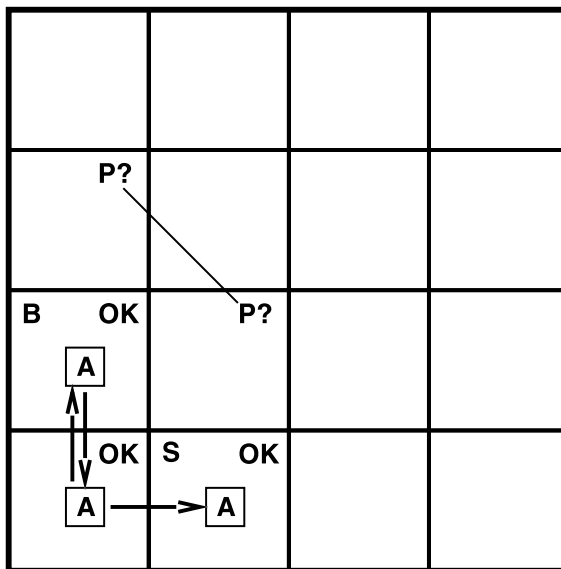
مثال: دنیای اژدها: اکتشاف در دنیای اژدها (۳ از ۸)



$Percept = [Stench, Breeze, Glitter, Bump, Scream] = [None, Breeze, None, None, None]$

عامل مبتنی بر دانایی

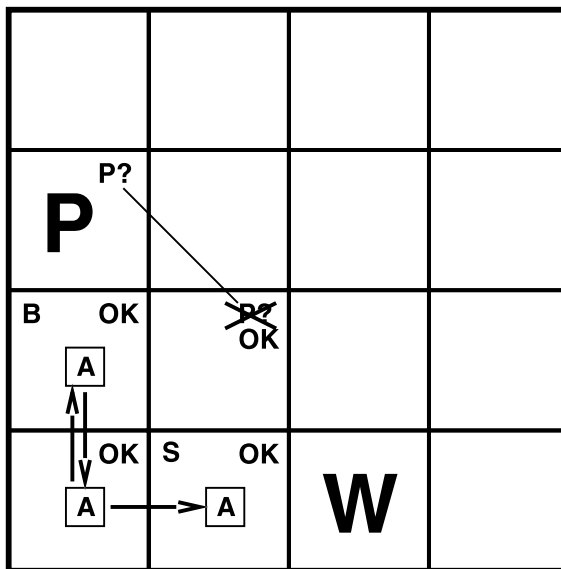
مثال: دنیای اژدها: اکتشاف در دنیای اژدها (۴ از ۸)



$Percept = [Stench, Breeze, Glitter, Bump, Scream] = [Stench, None, None, None, None]$

عامل مبتنی بر دانایی

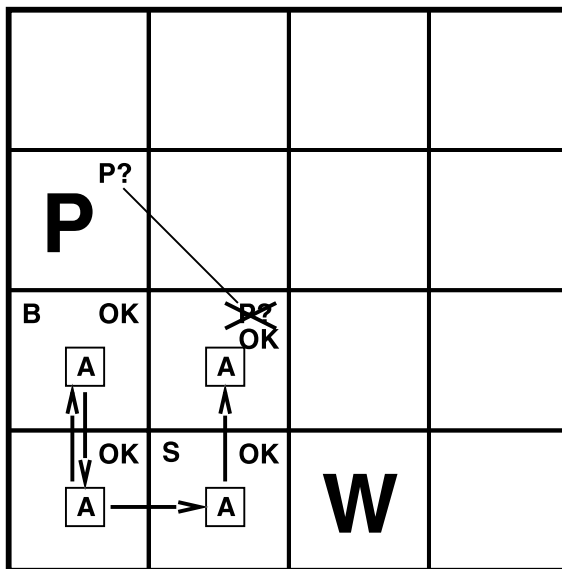
مثال: دنیای اژدها: اکتشاف در دنیای اژدها (۵ از ۸)



$Percept = [Stench, Breeze, Glitter, Bump, Scream] = [Stench, None, None, None, None]$

عامل مبتنی بر دانایی

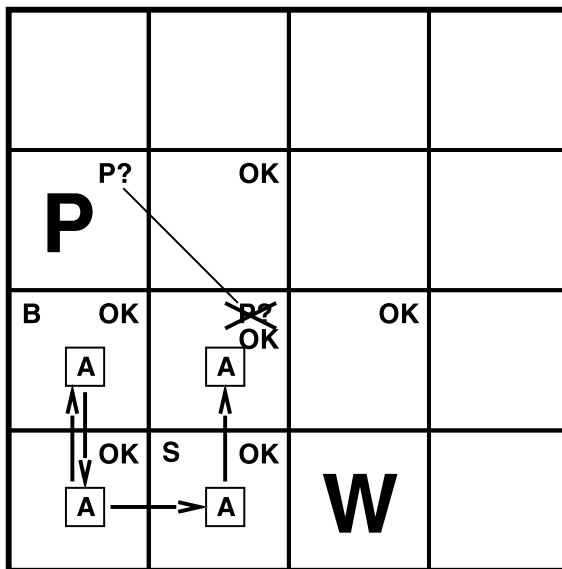
مثال: دنیای اژدها: اکتشاف در دنیای اژدها (۶ از ۸)



$Percept = [Stench, Breeze, Glitter, Bump, Scream] = [None, None, None, None, None]$

عامل مبتنی بر دانایی

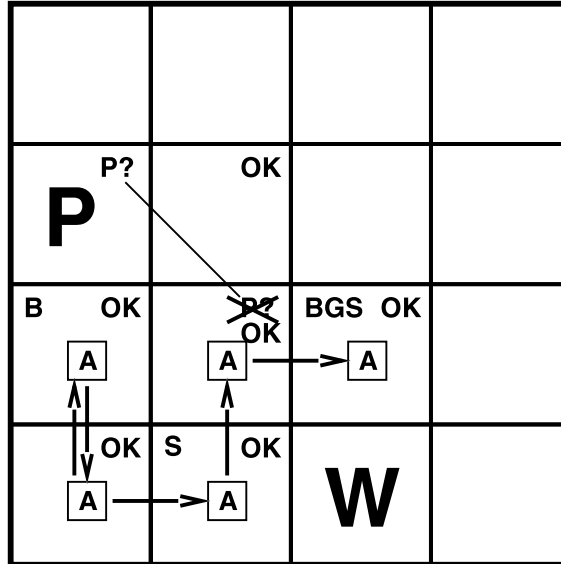
مثال: دنیای اژدها: اکتشاف در دنیای اژدها (۷ از ۸)



$Percept = [Stench, Breeze, Glitter, Bump, Scream] = [None, None, None, None, None]$

عامل مبتنی بر دانایی

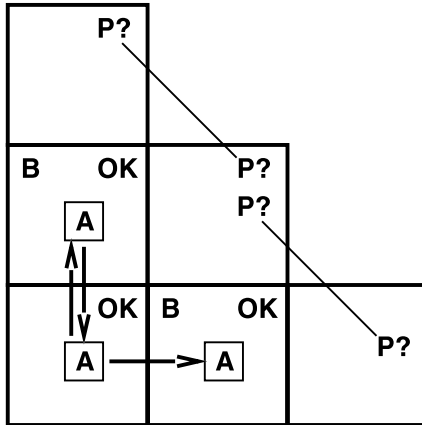
مثال: دنیای اژدها: اکتشاف در دنیای اژدها (۸ از ۸)



$Percept = [Stench, Breeze, Glitter, Bump, Scream] = [Stench, Breeze, Glitter, None, None]$

عامل مبتنی بر دانایی

مثال: دنیای اژدها: موقعیت عدم وجود کنش امن



حس نسیم در $(1,2)$ و $(2,1)$



هیچ کنش امنی وجود ندارد.

راه حل:

استفاده از استدلال احتمالاتی
(احتمال وجود چاله)

با فرض توزیع یکنواخت چاله‌ها

$(2,2)$ چاله دارد با احتمال 0.86 در مقابل 0.31

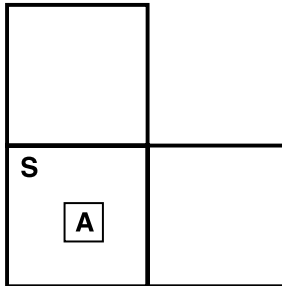
عامل مبتنی بر دانایی

مثال: دنیای اژدها: موقعیت عدم وجود کنش امن: استراتژی اضطرار

حس بوی بد اژدها در خانه‌ی شروع (1,1)



هیچ کنش امنی برای حرکت وجود ندارد.



راه حل:

استفاده از استراتژی اضطرار (Coercison)

عامل یک تیر به جلو شلیک می‌کند:

اگر اژدها در مقابل باشد $\Leftarrow$ اژدها کشته می‌شود $\Leftarrow$ وضعیت امن

اگر اژدها در مقابل نباشد $\Leftarrow$ وضعیت امن

$Percept = [Stench, Breeze, Glitter, Bump, Scream] = [Stench, None, None, None, None]$

هوش مصنوعی

عوامل‌های منطقی

۳

منطق

منطق

منطق

Logic

قوانین تفکر:

ابزار درست فکر کردن

تفکیک‌کننده‌ی **درستی** از **نادرستی**

منطق

در حالت عمومی

LOGIC IN GENERAL

منطق‌ها: زبان‌هایی صوری برای بازنمایی اطلاعات
به طوری که نتایجی می‌تواند از آن استخراج شود.

اجزای یک منطق

تئوری اثبات

Proof Theory

مجموعه قواعدی برای
استخراج جملات جدید

معناشناسی

Semantics

تعریف معنای جملات
(مثلاً: تعریف درستی یک جمله در
یک دنیا)

نحو

Syntax

تعریف جملات در زبان
(ظاهر جملات)

منطق

مثال

منطق: زبان حساب
Language of Arithmetic

اجزای منطق «زبان حساب»

تئوری اثبات

Proof Theory

مجموعه قواعدی برای
استخراج جملات جدید

معناشناسی

Semantics

تعریف معنای جملات
(مثلاً: تعریف *درستی* یک جمله در
یک دنیا)

نحو

Syntax

تعریف جملات در زبان
(ظاهر جملات)

$x + 2 \geq y$ is a sentence; $x^2 + y >$ is not a sentence

$x + 2 \geq y$ is true iff the number $x + 2$ is no less than the number y

$x + 2 \geq y$ is true in a world where $x = 7$, $y = 1$

$x + 2 \geq y$ is false in a world where $x = 0$, $y = 6$

منطق

تعهدات یک منطق

تعهدات یک منطق	
تعهدات معرفت‌شناختی <i>Epistemological Commitment</i>	تعهدات هستی‌شناختی <i>Ontological Commitment</i>
(آنچه عامل در مورد موجودها باور دارد)	(آنچه در جهان وجود دارد)
دانایی چه حالت‌هایی دارد؟	واقعیت‌ها، اشیا، زمان، باورها، ...

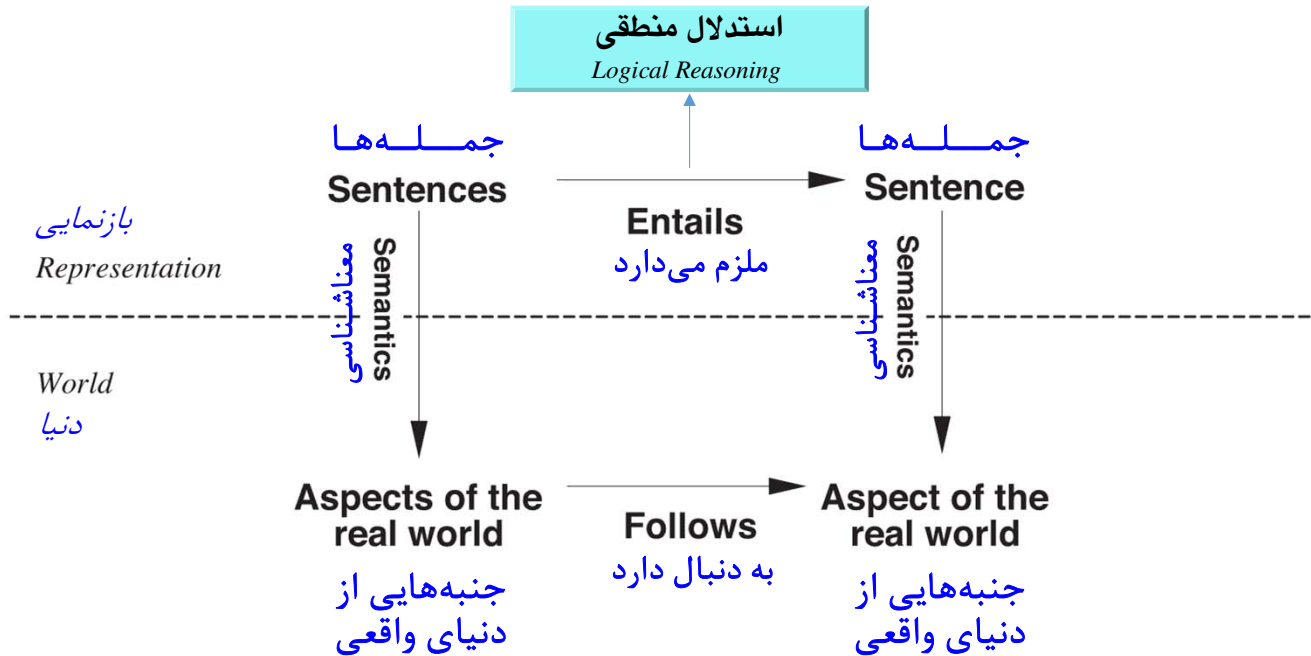
انواع منطق‌ها

و تعهدات آنها

تعهدات منطق

تعهدات معرفت‌شناختی <i>Epistemological Commitment</i>	تعهدات هستی‌شناختی <i>Ontological Commitment</i>	
درست / نادرست / ناشناخته <i>True / False / Unknown</i>	واقعیت‌ها <i>Facts</i>	منطق گزاره‌ای <i>Propositional Logic</i>
درست / نادرست / ناشناخته <i>True / False / Unknown</i>	واقعیت‌ها، اشیاء، رابطه‌ها <i>Facts, Objects, Relations</i>	منطق مرتبه اول <i>First-Order Logic</i>
درست / نادرست / ناشناخته <i>True / False / Unknown</i>	واقعیت‌ها، اشیاء، رابطه‌ها، زمان‌ها <i>Facts, Objects, Relations, Times</i>	منطق زمانی <i>Temporal Logic</i>
درجه‌ی باور بین صفر تا یک <i>Degree of Belief 0...1</i>	واقعیت‌ها <i>Facts</i>	نظریه‌ی احتمال <i>Probability Theory</i>
بازه‌ی معلومی از مقادیر <i>Known interval value</i>	واقعیت‌هایی با درجه‌ی درستی بین ۰ و ۱ <i>Facts with Degree of Truth in [0,1]</i>	منطق فازی <i>Fuzzy Logic</i>

نسبت بین منطق (بازنمایی) و واقعیت (دنیا)



استلزام

ENTAILMENT

استلزام: به معنی پیروی یک چیز از دیگری است.
(یعنی: اگر یک چیز باشد، دیگری هم به دنبال آن می‌آید.)

$$\beta \models \alpha$$

جمله‌ی β جمله‌ی α را ملزم می‌دارد
(جمله‌ی β مستلزم جمله‌ی α است)
(β entails α)

اگر و فقط اگر

α در همه‌ی دنیاهایی که در آن‌ها β درست است، درست باشد.

استلزام یک رابطه بین جملات (یعنی نحو) است که مبتنی بر معنا می‌باشد.

استلزام

از پایگاه دانایی

ENTAILMENT

استلزام: به معنی **پیروی** یک چیز **از** دیگری است.
(یعنی: اگر یک چیز باشد، دیگری هم به دنبال آن می‌آید.)

$$KB \models \alpha$$

پایگاه دانایی KB جمله‌ی α را ملزم می‌دارد
(پایگاه دانایی KB مستلزم جمله‌ی α است)
(KB entails α)

اگر و فقط اگر

α در همه‌ی دنیاهایی که در آن‌ها KB درست است، درست باشد.

استلزام

مثال

ENTAILMENT

استلزام: به معنی پیروی یک چیز از دیگری است.
(یعنی: اگر یک چیز باشد، دیگری هم به دنبال آن می‌آید.)

$$KB \models \alpha$$

در زبان حساب

$$x + y = 4 \quad \models \quad 4 = x + y$$

مدل‌ها

MODELS

مدل‌ها: دنیا‌های ساخت‌یافته‌ی صوری هستند که
با توجه به آنها **درستی** جملات می‌تواند ارزیابی شود.
(منطق‌دان‌ها معمولاً بر حسب مدل‌ها فکر می‌کنند.)

می‌گوییم m یک مدل از جمله‌ی α است

اگر

α در m درست باشد.

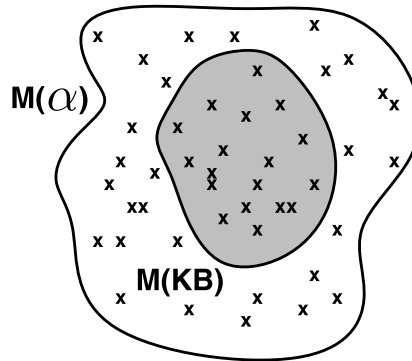
$$M(\alpha) = \text{مجموعه‌ی همه‌ی مدل‌های } \alpha$$

مدل‌ها و استلزام

$$KB \models \alpha$$

اگر و فقط اگر

$$M(KB) \subseteq M(\alpha)$$



مدل‌ها و استلزام

مثال: استلزام در دنیای اژدها (۱ از ۶)

ENTAILMENT IN THE WUMPUS WORLD

وضعیت:

پس از تشخیص هیچ چیز در $[1,1]$ ،
حرکت به راست،
تشخیص نسیم **B** در $[2,1]$

در نظر گرفتن مدل‌های ممکن برای $?$ ها
با فرض وجود فقط چاله **P**

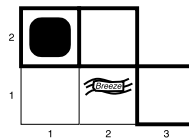
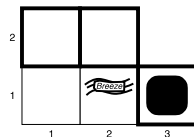
۳ متغیر بولی $\Leftarrow$ ۸ مدل ممکن

?	?		
A → A	B	?	

مدل‌ها و استلزام

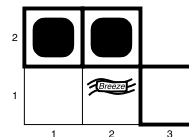
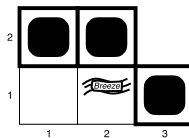
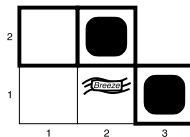
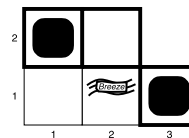
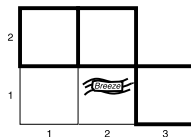
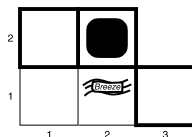
مثال: استلزام در دنیای اژدها (۲ از ۶)

ENTAILMENT IN THE WUMPUS WORLD



?	?		
A	B	?	

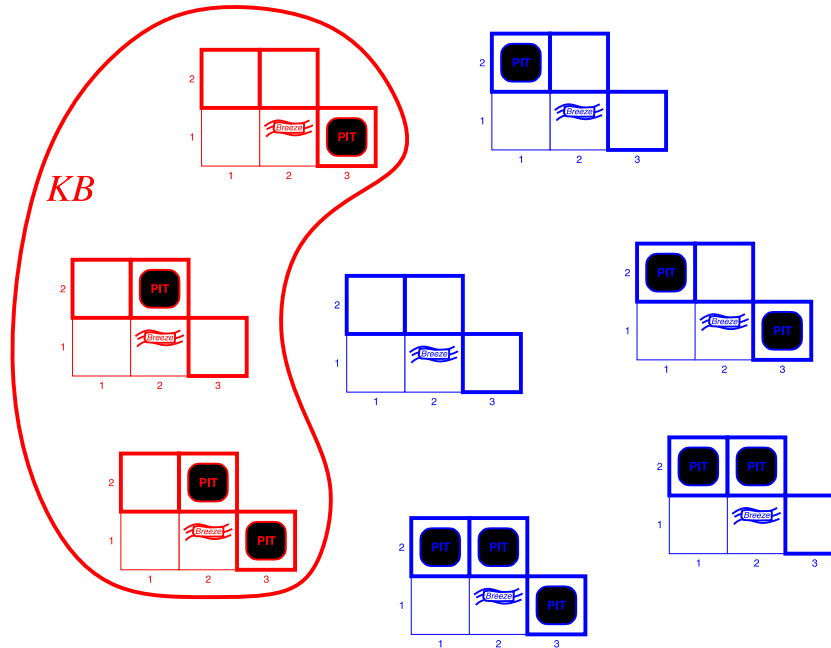
مدل‌های
دنیای اژدها
در وضعیت فوق:



مدل‌ها و استلزام

مثال: استلزام در دنیای اژدها (۳ از ۶)

ENTAILMENT IN THE WUMPUS WORLD



?	?		
A	B	?	

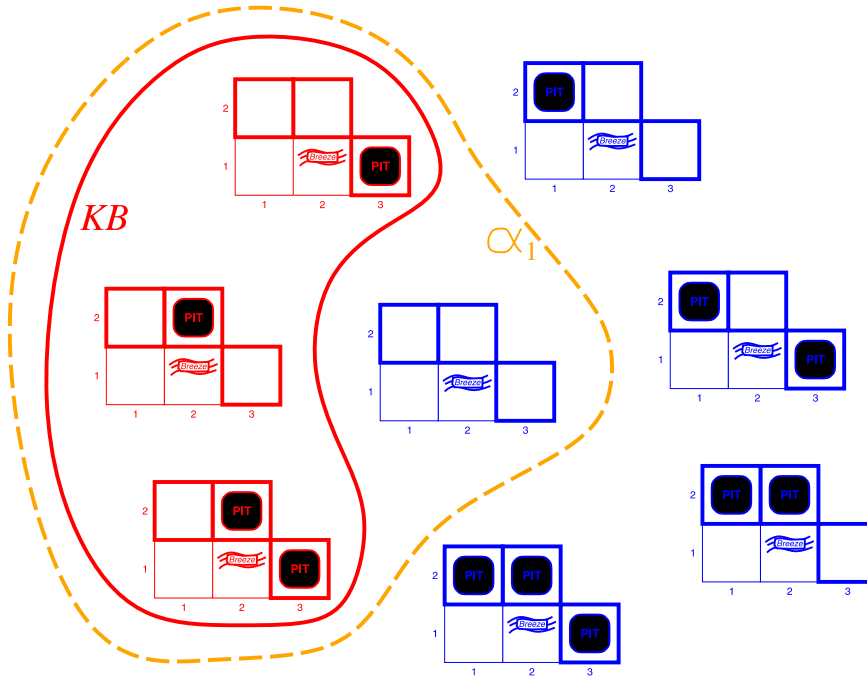
مدل‌های
دنیای اژدها
در وضعیت فوق:

پایگاه دانایی KB = قواعد دنیای اژدها + مشاهدات

مدل‌ها و استلزام

مثال: استلزام در دنیای اژدها (۴ از ۶)

ENTAILMENT IN THE WUMPUS WORLD



?	?		
A	B	?	

مدل‌های
دنیای اژدها
در وضعیت فوق:

پایگاه دانایی KB = قواعد دنیای اژدها + مشاهدات

اثبات با بررسی مدل
Model Checking

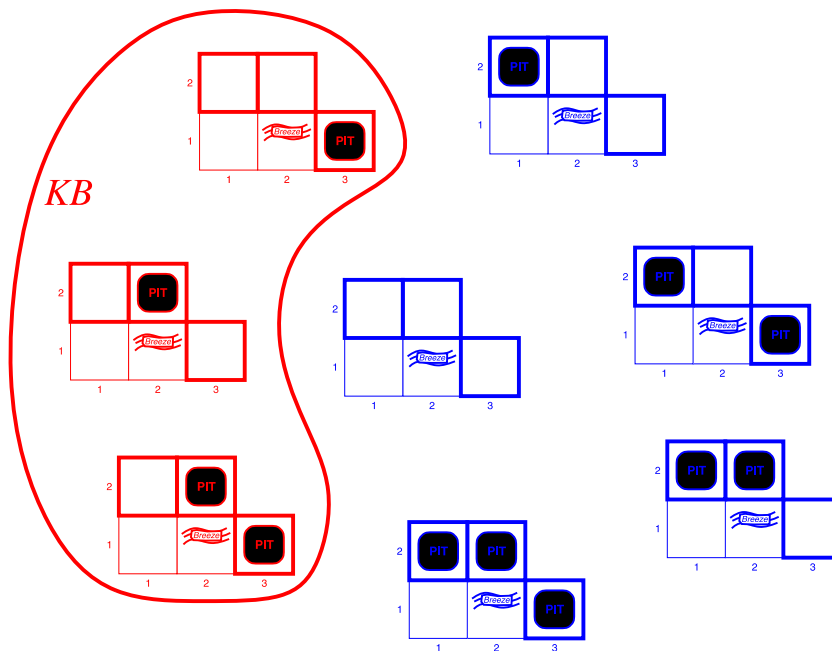
$$KB \models \alpha_1$$

$\alpha_1 = [1, 2]$ «امن است»

مدل‌ها و استلزام

مثال: استلزام در دنیای اژدها (۵ از ۶)

ENTAILMENT IN THE WUMPUS WORLD



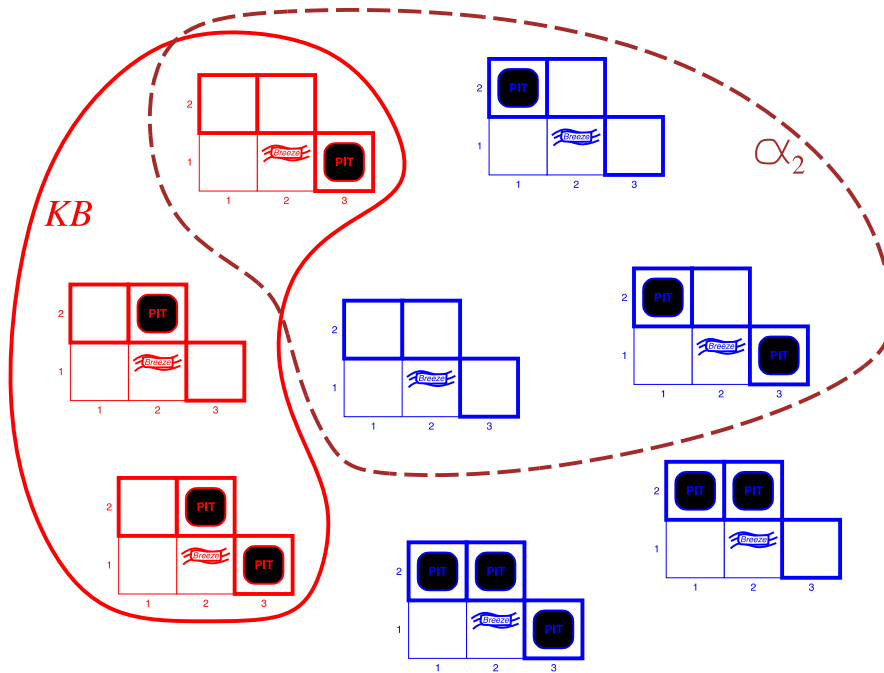
?	?		
A	B	?	

مدل‌های
دنیای اژدها
در وضعیت فوق:

پایگاه دانایی KB = قواعد دنیای اژدها + مشاهدات

مدل‌ها و استلزام

مثال: استلزام در دنیای اژدها (۶ از ۶)

ENTAILMENT IN THE WUMPUS WORLD

?	?		
A	B	?	

مدل‌های
دنیای اژدها
در وضعیت فوق:

پایگاه دانایی KB = قواعد دنیای اژدها + مشاهدات

$KB \not\models \alpha_2$ « α_2 [2,2] امن است»

استنتاج

INFERENCE

استنتاج: به معنی اشتقاق یک جمله از دیگری توسط یک روال خاص است.
(یعنی: استخراج یک جمله از جمله‌ی دیگر بر اساس یک قاعده)

$$\beta \vdash_i \alpha$$

جمله‌ی β جمله‌ی α را مشتق می‌کند توسط روال i
(جمله‌ی α از جمله‌ی β مشتق می‌شود توسط روال i)
(β derives α by procedure i)

استنتاج

از پایگاه دانایی

INFERENCE

استنتاج: به معنی **اشتقاق** یک جمله **از** دیگری توسط یک روال خاص است.
(یعنی: استخراج یک جمله از جمله‌ی دیگر بر اساس یک قاعده)

$$KB \vdash_i \alpha$$

پایگاه دانایی KB جمله‌ی α را مشتق می‌کند توسط روال i
(جمله‌ی α از پایگاه دانایی KB مشتق می‌شود توسط روال i)
(KB derives α by procedure i)

نسبت بین استنتاج و استلزام

$$KB \vdash_i \alpha$$

پی‌آمدهای پایگاه دانش:
کاهدان
haystack

سوزن
needle

استنتاج
Inference

یافتن سوزن در کاهدان

$$KB \vdash_i \alpha$$

استلزام
Entailment

انداختن یک سوزن در کاهدان
Needle in haystack

$$KB \models \alpha$$

استنتاج

خصوصیات یک روال استنتاج

برای روال استنتاج i	
تمامیت <i>Completeness</i>	صحیح بودن <i>Soundness</i>
همه‌ی جمله‌های درست مشتق شوند.	هر جمله‌ی مشتق شده درست باشد.
i کامل است اگر هرگاه $KB \models \alpha$ درست بود، آن‌گاه $KB \vdash_i \alpha$ نیز درست باشد.	i صحیح است اگر هرگاه $KB \vdash_i \alpha$ درست بود، آن‌گاه $KB \models \alpha$ نیز درست باشد.

عوامل‌های منطقی

۴

منطق گزاره‌ای

یک منطق بسیار ساده

منطق گزاره‌ای

PROPOSITIONAL LOGIC

منطق گزاره‌ای: ساده‌ترین منطق ترسیم‌کننده‌ی ایده‌های اصلی		
اجزای منطق گزاره‌ای		
تئوری اثبات <i>Proof Theory</i>	معناشناسی <i>Semantics</i>	نحو <i>Syntax</i>
استفاده از قواعد استنتاج بررسی مدل	قواعد درست / نادرست بودن گزاره‌ها	نمادهای گزاره‌ای رابطها گزاره‌های ترکیبی

منطق گزاره‌ای

نحو

PROPOSITIONAL LOGIC: SYNTAX



The proposition symbols P_1, P_2 etc are sentences

نمادهای گزاره‌ای

If S is a sentence, $\neg S$ is a sentence (negation)

نقیض

If S_1 and S_2 are sentences, $S_1 \wedge S_2$ is a sentence (conjunction)

عطف

If S_1 and S_2 are sentences, $S_1 \vee S_2$ is a sentence (disjunction)

فصل

If S_1 and S_2 are sentences, $S_1 \Rightarrow S_2$ is a sentence (implication)

ایجاب

If S_1 and S_2 are sentences, $S_1 \Leftrightarrow S_2$ is a sentence (biconditional)

دو شرطی

منطق گزاره‌ای

نحو: گرامر

PROPOSITIONAL LOGIC: SYNTAX: GRAMMAR



$$\begin{aligned}
 \text{Sentence} &\rightarrow \text{AtomicSentence} \mid \text{ComplexSentence} \\
 \text{AtomicSentence} &\rightarrow \mathbf{True} \mid \mathbf{False} \mid \text{Symbol} \\
 \text{Symbol} &\rightarrow \mathbf{P} \mid \mathbf{Q} \mid \mathbf{R} \mid \dots \\
 \text{ComplexSentence} &\rightarrow \neg \text{Sentence} \\
 &\mid (\text{Sentence} \wedge \text{Sentence}) \\
 &\mid (\text{Sentence} \vee \text{Sentence}) \\
 &\mid (\text{Sentence} \Rightarrow \text{Sentence}) \\
 &\mid (\text{Sentence} \Leftrightarrow \text{Sentence})
 \end{aligned}$$

منطق گزاره‌ای

معناشناسی

PROPOSITIONAL LOGIC: SEMANTICS

منطق گزاره‌ای: ساده‌ترین منطق

ترسیم‌کننده‌ی ایده‌های اصلی

اجزای منطق گزاره‌ای

تئوری اثبات

Proof Theory

معناشناسی

Semantics

نحو

Syntax

به هر گزاره **true** یا **false** نسبت داده می‌شود: معنای هر گزاره = درستی/نادرستی آن

$\neg S$ is true iff	S is false	
$S_1 \wedge S_2$ is true iff	S_1 is true and	S_2 is true
$S_1 \vee S_2$ is true iff	S_1 is true or	S_2 is true
$S_1 \Rightarrow S_2$ is true iff	S_1 is false or	S_2 is true
i.e., is false iff	S_1 is true and	S_2 is false
$S_1 \Leftrightarrow S_2$ is true iff	$S_1 \Rightarrow S_2$ is true and	$S_2 \Rightarrow S_1$ is true

منطق گزاره‌ای

معناشناسی: مثال

منطق گزاره‌ای: ساده‌ترین منطق

ترسیم‌کننده‌ی ایده‌های اصلی

اجزای منطق گزاره‌ای

تئوری اثبات

Proof Theory

معناشناسی

Semantics

نحو

Syntax

هر مدل **true** یا **false** را برای هر نماد گزاره‌ای مشخص می‌کند.

E.g. $P_{1,2}$ $P_{2,2}$ $P_{3,1}$ در «دنیای اژدها»:
 $true$ $true$ $false$

(With these symbols, 8 possible models, can be enumerated automatically.)

Simple recursive process evaluates an arbitrary sentence, e.g.,

$$\neg P_{1,2} \wedge (P_{2,2} \vee P_{3,1}) = true \wedge (false \vee true) = true \wedge true = true$$

منطق گزاره‌ای

جدول درستی برای رابطها

TRUTH TABLES FOR CONNECTIVES

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
<i>false</i>	<i>false</i>	<i>true</i>	<i>false</i>	<i>false</i>	<i>true</i>	<i>true</i>
<i>false</i>	<i>true</i>	<i>true</i>	<i>false</i>	<i>true</i>	<i>true</i>	<i>false</i>
<i>true</i>	<i>false</i>	<i>false</i>	<i>false</i>	<i>true</i>	<i>false</i>	<i>false</i>
<i>true</i>	<i>true</i>	<i>false</i>	<i>true</i>	<i>true</i>	<i>true</i>	<i>true</i>

منطق گزاره‌ای

مثال: جملاتی برای دنیای اژدها

WUMPUS WORLD SENTENCES

Let $P_{i,j}$ be true if there is a pit in $[i, j]$.

Let $B_{i,j}$ be true if there is a breeze in $[i, j]$.

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

$$\neg B_{1,1}$$

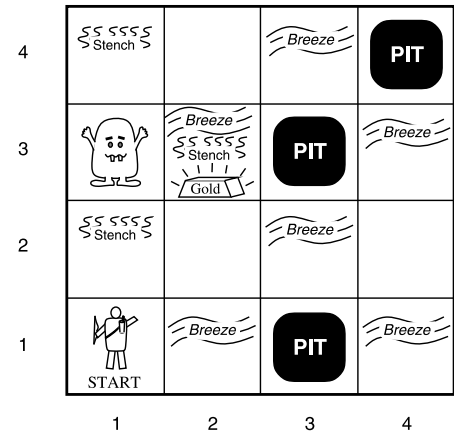
$$B_{2,1}$$

“Pits cause breezes in adjacent squares”

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

$$B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$$

“A square is breezy **if and only if** there is an adjacent pit”



استنتاج با «جدول درستی»

TRUTH TABLES FOR INFERENCE

$B_{1,1}$	$B_{2,1}$	$P_{1,1}$	$P_{1,2}$	$P_{2,1}$	$P_{2,2}$	$P_{3,1}$	KB	α_1
false	false	false	false	false	false	false	false	true
false	false	false	false	false	false	true	false	true
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
false	true	false	false	false	false	false	false	true
false	true	false	false	false	false	true	<u>true</u>	<u>true</u>
false	true	false	false	false	true	false	<u>true</u>	<u>true</u>
false	true	false	false	false	true	true	<u>true</u>	<u>true</u>
false	true	false	false	true	false	false	false	true
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
true	true	true	true	true	true	true	false	false

هر سطر یک ترکیب انتساب متفاوت به نمادهای گزاره‌ای را نشان می‌دهد.

سطرها را بررسی می‌کنیم:

در هر سطری که KB درست بود، بررسی می‌کنیم که α هم درست باشد.

در این صورت

$$KB \models \alpha$$

استنتاج با «شمارش»

شبه‌کد

INFERENCE BY ENUMERATION

شمارش عمق-اول همه‌ی مدل‌ها: روش استنتاج صحیح (sound) و کامل (complete)

```
function TT-ENTAILS?(KB,  $\alpha$ ) returns true or false
```

```
  symbols  $\leftarrow$  a list of the proposition symbols in KB and  $\alpha$ 
```

```
  return TT-CHECK-ALL(KB,  $\alpha$ , symbols, [])
```

```
function TT-CHECK-ALL(KB,  $\alpha$ , symbols, model) returns true or false
```

```
  if EMPTY?(symbols) then
```

```
    if PL-TRUE?(KB, model) then return PL-TRUE?( $\alpha$ , model)
```

```
    else return true
```

```
  else do
```

```
    P  $\leftarrow$  FIRST(symbols); rest  $\leftarrow$  REST(symbols)
```

```
    return TT-CHECK-ALL(KB,  $\alpha$ , rest, EXTEND(P, true, model)) and  
          TT-CHECK-ALL(KB,  $\alpha$ , rest, EXTEND(P, false, model))
```

$O(2^n)$ for n symbols; problem is **co-NP-complete**

همارزی منطقی

جمله‌های هم‌ارز

LOGICAL EQUIVALENCE

دو جمله از نظر منطقی هم‌ارز هستند

اگر و فقط اگر

هر دو در مدل‌های مشابهی درست باشند.

$$\alpha \equiv \beta \text{ if and only if } \alpha \models \beta \text{ and } \beta \models \alpha$$

همارزی منطقی

قوانین همارزی منطقی

LOGICAL EQUIVALENCE

$$(\alpha \wedge \beta) \equiv (\beta \wedge \alpha) \quad \text{commutativity of } \wedge \quad \text{جابجاپذیری}$$

$$(\alpha \vee \beta) \equiv (\beta \vee \alpha) \quad \text{commutativity of } \vee \quad \text{جابجاپذیری}$$

$$((\alpha \wedge \beta) \wedge \gamma) \equiv (\alpha \wedge (\beta \wedge \gamma)) \quad \text{associativity of } \wedge \quad \text{شرکت پذیری}$$

$$((\alpha \vee \beta) \vee \gamma) \equiv (\alpha \vee (\beta \vee \gamma)) \quad \text{associativity of } \vee \quad \text{شرکت پذیری}$$

$$\neg(\neg\alpha) \equiv \alpha \quad \text{double-negation elimination} \quad \text{حذف نقیض مضاعف}$$

$$(\alpha \Rightarrow \beta) \equiv (\neg\beta \Rightarrow \neg\alpha) \quad \text{contraposition} \quad \text{عکس نقیض}$$

$$(\alpha \Rightarrow \beta) \equiv (\neg\alpha \vee \beta) \quad \text{implication elimination} \quad \text{حذف ایجاب}$$

$$(\alpha \Leftrightarrow \beta) \equiv ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)) \quad \text{biconditional elimination} \quad \text{حذف دوشروطی}$$

$$\neg(\alpha \wedge \beta) \equiv (\neg\alpha \vee \neg\beta) \quad \text{de Morgan} \quad \text{دمورگان}$$

$$\neg(\alpha \vee \beta) \equiv (\neg\alpha \wedge \neg\beta) \quad \text{de Morgan} \quad \text{دمورگان}$$

$$(\alpha \wedge (\beta \vee \gamma)) \equiv ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma)) \quad \text{distributivity of } \wedge \text{ over } \vee \quad \text{توزیع پذیری}$$

$$(\alpha \vee (\beta \wedge \gamma)) \equiv ((\alpha \vee \beta) \wedge (\alpha \vee \gamma)) \quad \text{distributivity of } \vee \text{ over } \wedge \quad \text{توزیع پذیری}$$

۵

منطق گزاره‌ای و اثبات قضیه

منطق گزاره‌ای

تئوری اثبات

PROPOSITIONAL LOGIC: PROOF THEORY

منطق گزاره‌ای: ساده‌ترین منطق

ترسیم‌کننده‌ی ایده‌های اصلی

اجزای منطق گزاره‌ای

تئوری اثبات

Proof Theory

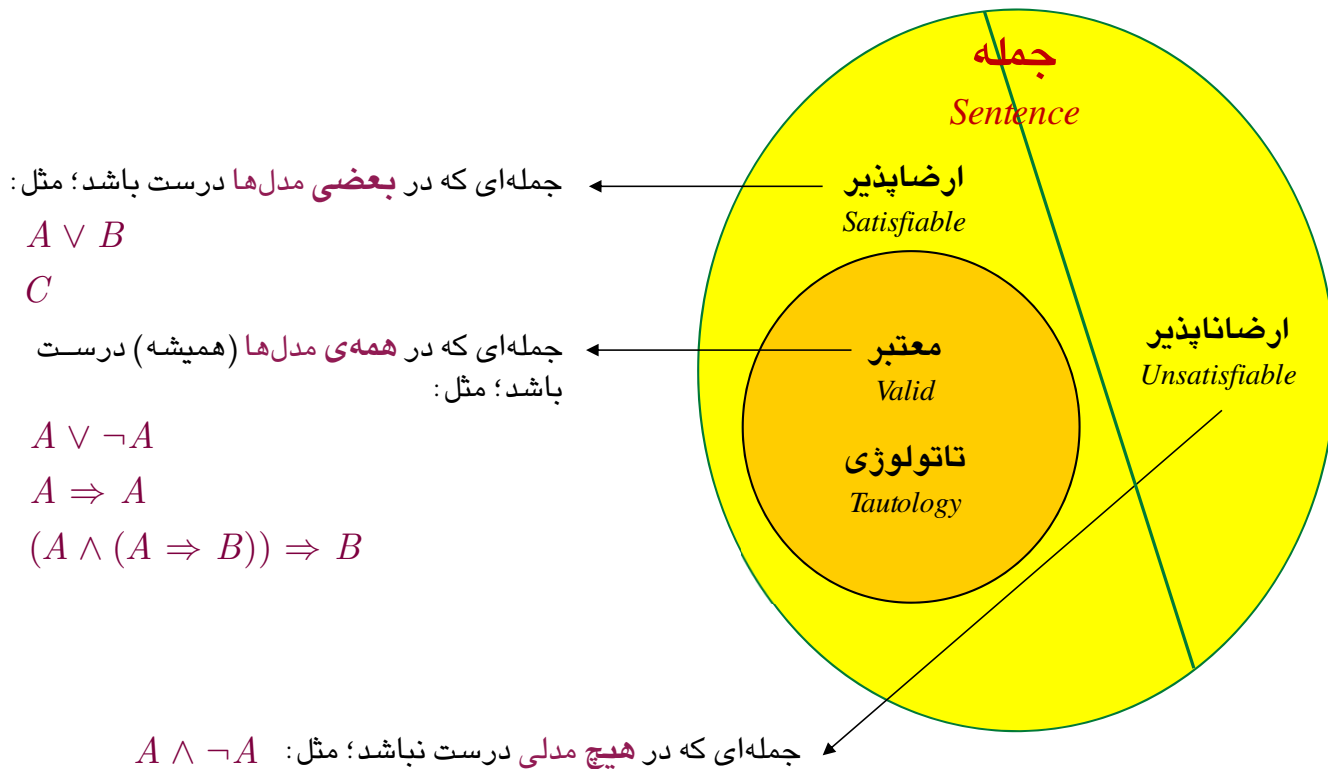
معناشناسی

Semantics

نحو

Syntax

اعتبار و ارضاپذیری



اتصال استنتاج با اعتبار

قضیه‌ی استنباط

DEDUCTION THEOREM

$KB \models \alpha$ if and only if $(KB \Rightarrow \alpha)$ is valid

اتصال استنتاج با ارضاناپذیری

برهان خلف

REDUCTIO AD ABSURDUM

$KB \models \alpha$ if and only if $(KB \wedge \neg\alpha)$ is unsatisfiable

برهان خلف

reductio ad absurdum

قواعد استنتاج

الگوهای استدلال

INFERENCE RULES (REASONING PATTERNS)

◇ Modus Ponens

$$\frac{\alpha \Rightarrow \beta, \quad \alpha}{\beta}$$

◇ And-Elimination

$$\frac{\alpha_1 \wedge \alpha_2 \wedge \dots \wedge \alpha_n}{\alpha_i}$$

◇ And-Introduction

$$\frac{\alpha_1, \alpha_2, \dots, \alpha_n}{\alpha_1 \wedge \alpha_2 \wedge \dots \wedge \alpha_n}$$

◇ Or-Introduction

$$\frac{\alpha_i}{\alpha_1 \vee \alpha_2 \vee \dots \vee \alpha_n}$$

◇ Double Negation-Elimination

$$\frac{\neg \neg \alpha}{\alpha}$$

◇ Unit Resolution

$$\frac{\alpha \vee \beta, \quad \neg \beta}{\alpha}$$

◇ Resolution

$$\frac{\alpha \vee \beta, \quad \neg \beta \vee \gamma}{\alpha \vee \gamma}$$

or equivalently

$$\frac{\neg \alpha \Rightarrow \beta, \quad \beta \Rightarrow \gamma}{\neg \alpha \Rightarrow \gamma}$$

فرم‌های نرمال

NORMAL FORMS

◇ **Conjunctive Normal Form (CNF-universal)**
conjunction of disjunctions of literals
clauses

E.g., $(A \vee \neg B) \wedge (B \vee \neg C \vee \neg D)$

◇ **Disjunctive Normal Form (DNF-universal)**
disjunction of conjunctions of literals
terms

E.g., $(A \wedge B) \vee (A \wedge \neg C) \vee (A \wedge B \wedge \neg C)$

◇ **Horn Form (restricted)**
conjunction of Horn clauses
 Horn clause: a clause with ≤ 1 positive literal.

E.g., $(A \vee \neg B) \wedge (B \vee \neg C \vee \neg D)$

Often written as set of implications: $B \Rightarrow A$ and $(C \wedge D) \Rightarrow B$

فرم نرمال عطفی

Conjunctive Normal Form (CNF)

عطف کلاوزها

فرم نرمال فصلی

Disjunctive Normal Form (DNF)

فصل ترم‌ها

فرم هورن

Horn Form

عطف کلاوزهای هورن

اثبات

اثبات α : دنباله‌ای از به‌کارگیری قواعد استنتاج منتهی به α
(یافتن اثبات دقیقاً مثل یافتن راه‌حل‌های یک مسئله‌ی جستجو است.)

روش‌های اثبات

بررسی مدل <i>Model Checking</i>	به‌کارگیری قواعد استنتاج <i>Application of inference rules</i>
شمارش جدول درستی • همیشه نمایی بر حسب n • عقب‌گرد بهبودیافته، مانند: • مثل: الگوریتم DPLL • جستجوی هیوریستیک در فضای مدل (صحیح اما ناکامل) • مثل: مشابه تپه‌نوردی (حداقل تداخل‌ها)	تولید جملات جدید صحیح از قبلی‌ها • اثبات = دنباله‌ای از به‌کارگیری قواعد استنتاج (قواعد استنتاج = کنش‌ها در هر مرحله از الگوریتم جستجو) • معمولاً نیازمند تبدیل جملات به یک فرم نرمال

خاصیت یکنوایی در منطق‌های یکنوا

و کاربرد آن در اثبات

MONOTONICITY

اگر منطقی یکنوا باشد؛
برای هر جمله‌ی α و β ، داریم

اگر

$$KB \models \alpha$$

آن‌گاه

$$KB \wedge \beta \models \alpha$$

نتیجه‌ی یکنوایی:

- اضافه شدن جملات جدید به پایگاه دانایی، استلزام جملات قبلی را نقض نمی‌کند.
- قواعد استنتاج می‌توانند هرگاه که پیش‌فرض‌های مناسب در KB پیدا شد، استفاده شوند.
- نتایج استفاده از قاعده باید از KB پیروی کند، بدون توجه به مابقی آنچه در KB قرار دارد.

منطق‌های غیریکنوا، خاصیت یکنوایی را نقض می‌کنند.

اثبات

روش‌ها

اثبات α : دنباله‌ای از به‌کارگیری قواعد استنتاج منتهی به α
(یافتن اثبات دقیقاً مثل یافتن راه‌حل‌های یک مسئله‌ی جستجو است.)

روش‌های اثبات

روش‌های اثبات		
بررسی مدل <i>Model Checking</i>	به‌کارگیری قواعد استنتاج <i>Application of inference rules</i>	
شمارش جدول درستی	تولید جملات جدید صحیح از قبلی‌ها	
الگوریتم DPLL <i>DPLL Algorithm</i>	HF + Modus Ponens	زنجیره‌سازی پیش‌رو <i>Forward Chaining</i>
الگوریتم WalkSAT <i>WalkSAT Algorithm</i>		زنجیره‌سازی پس‌رو <i>Backward Chaining</i>
	CNF	رزولوشن <i>Resolution</i>

اثبات

روش‌های به‌کارگیری قواعد استنتاج

اثبات α : دنباله‌ای از به‌کارگیری قواعد استنتاج منتهی به α
 (یافتن اثبات دقیقاً مثل یافتن راه‌حل‌های یک مسئله‌ی جستجو است.)

روش‌های اثبات

روش‌های اثبات		
بررسی مدل <i>Model Checking</i>	به‌کارگیری قواعد استنتاج <i>Application of inference rules</i>	
شمارش جدول درستی	تولید جملات جدید صحیح از قبلی‌ها	
الگوریتم DPLL <i>DPLL Algorithm</i>	HF + Modus Ponens	زنجیره‌سازی پیش‌رو <i>Forward Chaining</i>
الگوریتم WalkSAT <i>WalkSAT Algorithm</i>		زنجیره‌سازی پس‌رو <i>Backward Chaining</i>
	CNF	رزولوشن <i>Resolution</i>

استنتاج با «قیاس استثنائی»

خصوصیات روال استنتاج

$$\frac{\alpha_1, \dots, \alpha_n, \quad \alpha_1 \wedge \dots \wedge \alpha_n \Rightarrow \beta}{\beta}$$

برای روال استنتاج «قیاس استثنائی» «Modus Ponens»

تمامیت <i>Completeness</i>	صحیح بودن <i>Soundness</i>
همه‌ی جمله‌های درست مشتق شوند. فقط برای جملات به فرم هورن (Horn)	هر جمله‌ی مشتق شده درست باشد. همیشه

قابل استفاده در زنجیره‌سازی پیشرو و زنجیره‌سازی پسرو
زمان اجرای خطی

فرم هورن

HORN FORM

فرم هورن

Horn Form

عطف کلاوزهای هورن

Horn Form (restricted)

KB = **conjunction** of **Horn clauses**

Horn clause =

◇ proposition symbol; or

◇ (conjunction of symbols) $\Rightarrow$ symbol

E.g., $C \wedge (B \Rightarrow A) \wedge (C \wedge D \Rightarrow B)$

الگوریتم اثبات: «زنجیره‌سازی پیش‌رو»

بر اساس قاعده‌ی استنتاج «قیاس استثنائی»

FORWARD CHAINING

ایده:

هر قاعده در KB که مقدم‌هایش ارضا شده است را **فایر** کنید
و نتیجه‌ی آن را به KB اضافه کنید تا اینکه هدف پرس وجو پیدا شود.

زنجیره‌سازی پیش‌رو

Forward Chaining

الگوریتم اثبات: «زنجیره سازی پیش رو»

شبهه‌کد

FORWARD CHAINING

```

function PL-FC-ENTAILS?(KB, q) returns true or false
  local variables: count, a table, indexed by clause, initially the number of premises
                    inferred, a table, indexed by symbol, each entry initially false
                    agenda, a list of symbols, initially the symbols known to be true

  while agenda is not empty do
    p ← POP(agenda)
    unless inferred[p] do
      inferred[p] ← true
      for each Horn clause c in whose premise p appears do
        decrement count[c]
        if count[c] = 0 then do
          if HEAD[c] = q then return true
          PUSH(HEAD[c], agenda)

  return false
  
```

الگوریتم اثبات: «زنجیره‌سازی پیشرو»

مثال (۱ از ۹)

FORWARD CHAINING

ایده:

هر قاعده در KB که مقدم‌هایش ارضا شده است را **فایر** کنید و نتیجه‌ی آن را به KB اضافه کنید تا اینکه هدف پرس وجو پیدا شود.

زنجیره‌سازی پیشرو
Forward Chaining

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

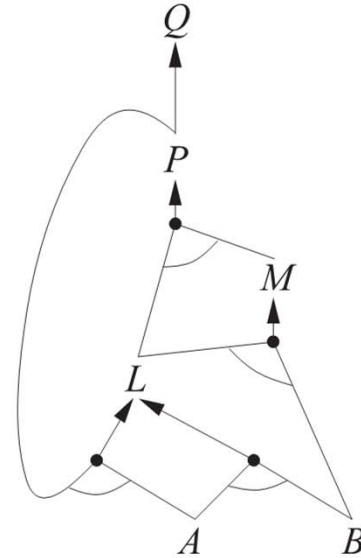
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

A

B



KB (Knowledge Base)

AND-OR Graph

الگوریتم اثبات: «زنجیره‌سازی پیش‌رو»

مثال (۲ از ۹)

FORWARD CHAINING

ایده:

هر قاعده در KB که مقدم‌هایش ارضا شده است را **فایر** کنید و نتیجه‌ی آن را به KB اضافه کنید تا اینکه هدف پرس وجو پیدا شود.

زنجیره‌سازی پیش‌رو

Forward Chaining

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

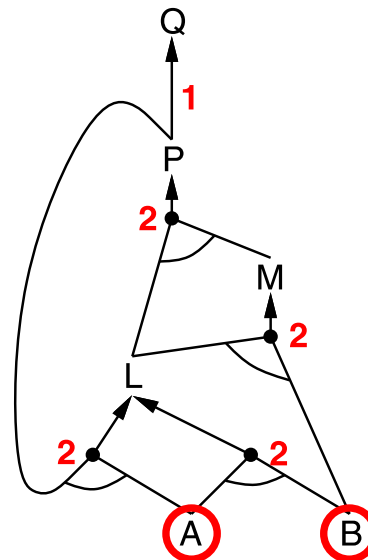
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

A

B



الگوریتم اثبات: «زنجیره‌سازی پیش‌رو»

مثال (۳ از ۹)

FORWARD CHAINING

ایده:

هر قاعده در KB که مقدم‌هایش ارضا شده است را **فایر** کنید و نتیجه‌ی آن را به KB اضافه کنید تا اینکه هدف پرس وجو پیدا شود.

زنجیره‌سازی پیش‌رو
Forward Chaining

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

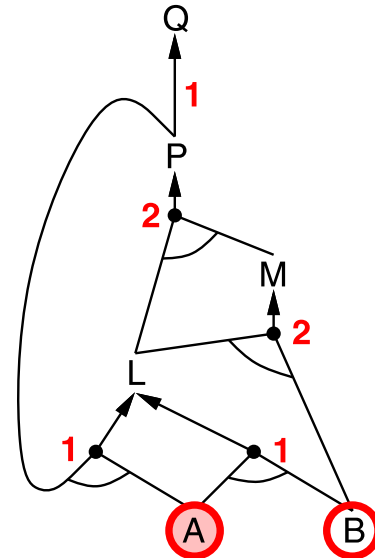
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

A

B



الگوریتم اثبات: «زنجیره‌سازی پیش‌رو»

مثال (۴ از ۹)

FORWARD CHAINING

ایده:

هر قاعده در KB که مقدم‌هایش ارضا شده است را **فایر** کنید و نتیجه‌ی آن را به KB اضافه کنید تا اینکه هدف پرس وجو پیدا شود.

زنجیره‌سازی پیش‌رو
Forward Chaining

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

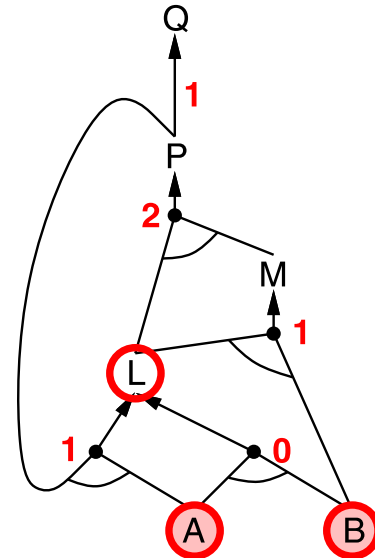
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

A

B



الگوریتم اثبات: «زنجیره‌سازی پیش‌رو»

مثال (۵ از ۹)

FORWARD CHAINING

ایده:

هر قاعده در KB که مقدم‌هایش ارضا شده است را **فایر** کنید و نتیجه‌ی آن را به KB اضافه کنید تا اینکه هدف پرس وجو پیدا شود.

زنجیره‌سازی پیش‌رو
Forward Chaining

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

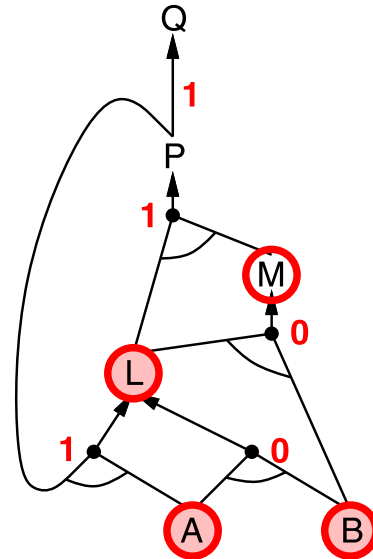
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

A

B



الگوریتم اثبات: «زنجیره‌سازی پیش‌رو»

مثال (۶ از ۹)

FORWARD CHAINING

ایده:

هر قاعده در KB که مقدم‌هایش ارضا شده است را **فایر** کنید و نتیجه‌ی آن را به KB اضافه کنید تا اینکه هدف پرس وجو پیدا شود.

زنجیره‌سازی پیش‌رو

Forward Chaining

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

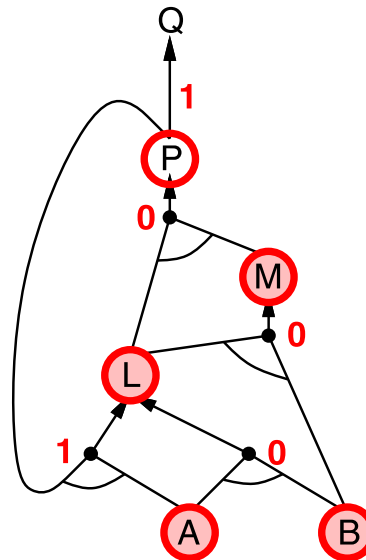
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

A

B



الگوریتم اثبات: «زنجیره‌سازی پیش‌رو»

مثال (۷ از ۹)

FORWARD CHAINING

ایده:

هر قاعده در KB که مقدم‌هایش ارضا شده است را **فایر** کنید و نتیجه‌ی آن را به KB اضافه کنید تا اینکه هدف پرس وجو پیدا شود.

زنجیره‌سازی پیش‌رو

Forward Chaining

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

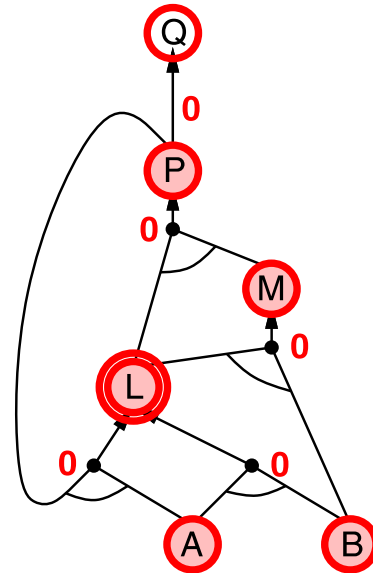
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

A

B



الگوریتم اثبات: «زنجیره‌سازی پیش‌رو»

مثال (۸ از ۹)

FORWARD CHAINING

ایده:

هر قاعده در KB که مقدم‌هایش ارضا شده است را **فایر** کنید و نتیجه‌ی آن را به KB اضافه کنید تا اینکه هدف پرس وجو پیدا شود.

زنجیره‌سازی پیش‌رو
Forward Chaining

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

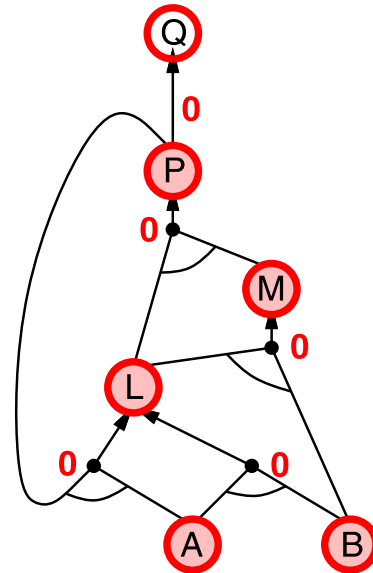
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

A

B



الگوریتم اثبات: «زنجیره‌سازی پیش‌رو»

مثال (۹ از ۹)

FORWARD CHAINING

ایده:

هر قاعده در KB که مقدم‌هایش ارضا شده است را **فایر** کنید و نتیجه‌ی آن را به KB اضافه کنید تا اینکه هدف پرس وجو پیدا شود.

زنجیره‌سازی پیش‌رو
Forward Chaining

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

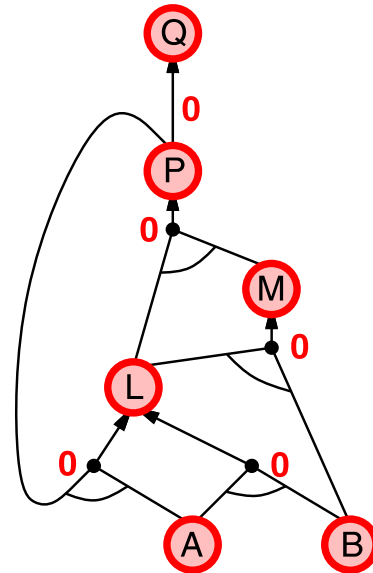
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

A

B



الگوریتم اثبات: «زنجیره سازی پیش رو»

اثبات تمامیت

FC derives every atomic sentence that is entailed by KB

1. FC reaches a **fixed point** where no new atomic sentences are derived
2. Consider the final state as a model m , assigning true/false to symbols
3. Every clause in the original KB is true in m

Proof: Suppose a clause $a_1 \wedge \dots \wedge a_k \Rightarrow b$ is false in m

Then $a_1 \wedge \dots \wedge a_k$ is true in m and b is false in m

Therefore the algorithm has not reached a fixed point!

4. Hence m is a model of KB
5. If $KB \models q$, q is true in **every** model of KB , including m

General idea: construct any model of KB by sound inference, check α

الگوریتم اثبات: «زنجیره‌سازی پس‌رو»

بر اساس قاعده‌ی استنتاج «قیاس استثنائی»

BACKWARD CHAINING

زنجیره‌سازی پس‌رو *Backward Chaining*

ایده: از هدف پرس‌وجوی q به صورت پس‌رو شروع می‌کنیم:
یا درستی q معلوم است
یا مقدم‌های برخی قواعد که q را نتیجه می‌دهند باید ثابت شود (با همین روش).

ملزومات روش زنجیره‌سازی پس‌رو

اجتناب از کار تکراری

Avoid repeated work

- بررسی می‌کنیم که
اگر یک زیرهدف جدید وجود دارد
(۱) آیا درستی آن تاکنون ثابت شده است؟ یا
(۲) اثبات درستی آن تاکنون با شکست مواجه شده است؟

اجتناب از حلقه‌ها

Avoid loops

- بررسی می‌کنیم که
آیا یک زیرهدف جدید هم‌اکنون بر روی پشته‌ی هدف
قرار دارد؟

الگوریتم اثبات: «زنجیره‌سازی پس‌رو»

مثال (۱ از ۱۱)

BACKWARD CHAINING

ایده: از هدف پرس‌وجوی q به صورت پس‌رو شروع می‌کنیم:
یا درستی q معلوم است
یا مقدم‌های برخی قواعد که q را نتیجه می‌دهند باید ثابت شود (با همین روش).

زنجیره‌سازی پس‌رو
Backward Chaining

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

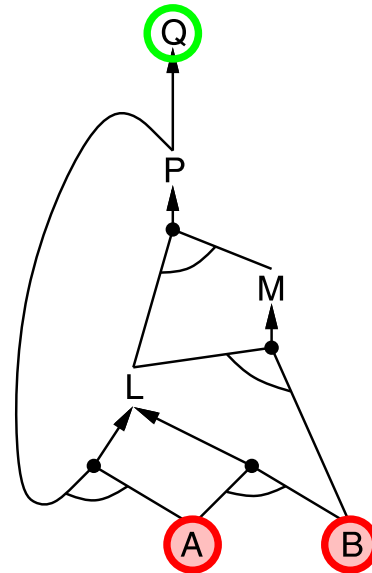
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

A

B



الگوریتم اثبات: «زنجیره‌سازی پس‌رو»

مثال (۲ از ۱۱)

BACKWARD CHAINING

ایده: از هدف پرس‌وجوی q به صورت پس‌رو شروع می‌کنیم:
یا درستی q معلوم است
یا مقدم‌های برخی قواعد که q را نتیجه می‌دهند باید ثابت شود (با همین روش).

زنجیره‌سازی پس‌رو
Backward Chaining

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

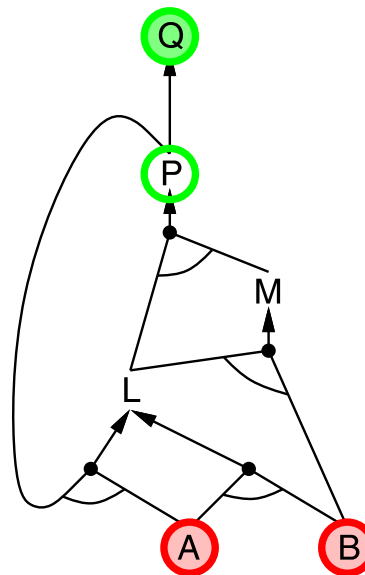
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

A

B



الگوریتم اثبات: «زنجیره‌سازی پس‌رو»

مثال (۳ از ۱۱)

BACKWARD CHAINING

ایده: از هدف پرس‌وجوی q به صورت پس‌رو شروع می‌کنیم:
یا درستی q معلوم است
یا مقدم‌های برخی قواعد که q را نتیجه می‌دهند باید ثابت شود (با همین روش).

زنجیره‌سازی پس‌رو
Backward Chaining

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

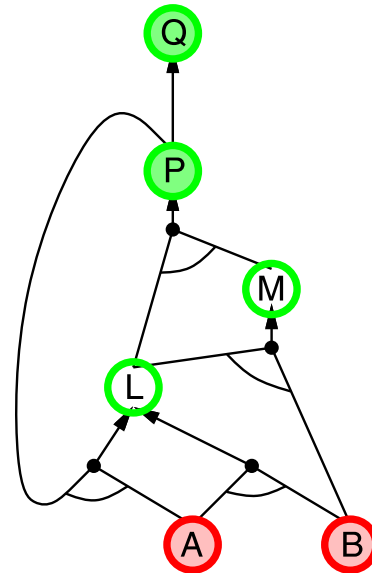
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

A

B



الگوریتم اثبات: «زنجیره‌سازی پس‌رو»

مثال (۴ از ۱۱)

BACKWARD CHAINING

ایده: از هدف پرس‌وجوی q به صورت پس‌رو شروع می‌کنیم:
یا درستی q معلوم است
یا مقدم‌های برخی قواعد که q را نتیجه می‌دهند باید ثابت شود (با همین روش).

زنجیره‌سازی پس‌رو
Backward Chaining

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

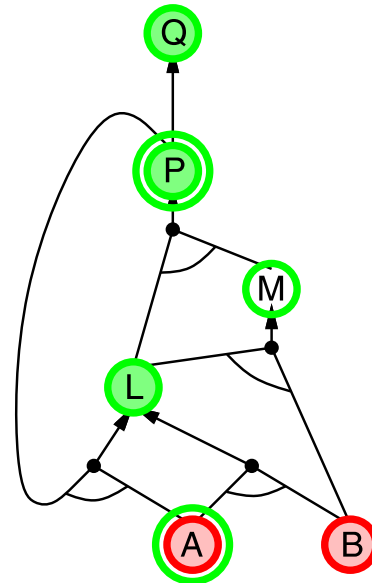
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

A

B



الگوریتم اثبات: «زنجیره‌سازی پس‌رو»

مثال (۵ از ۱۱)

BACKWARD CHAINING

ایده: از هدف پرس‌وجوی q به صورت پس‌رو شروع می‌کنیم:
یا درستی q معلوم است
یا مقدم‌های برخی قواعد که q را نتیجه می‌دهند باید ثابت شود (با همین روش).

زنجیره‌سازی پس‌رو
Backward Chaining

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

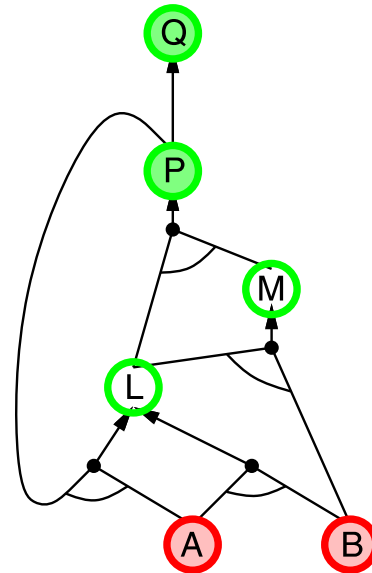
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

A

B



الگوریتم اثبات: «زنجیره‌سازی پس‌رو»

مثال (۶ از ۱۱)

BACKWARD CHAINING

ایده: از هدف پرس‌وجوی q به صورت پس‌رو شروع می‌کنیم:
یا درستی q معلوم است
یا مقدم‌های برخی قواعد که q را نتیجه می‌دهند باید ثابت شود (با همین روش).

زنجیره‌سازی پس‌رو
Backward Chaining

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

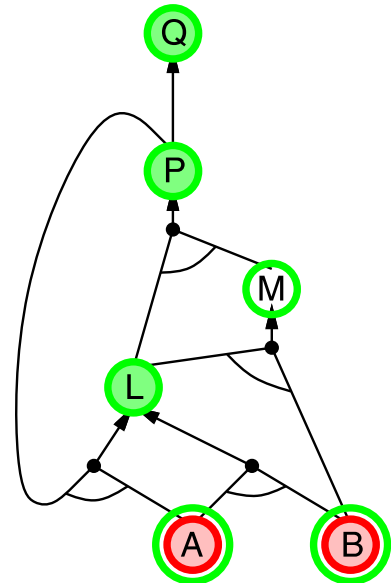
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

A

B



الگوریتم اثبات: «زنجیره‌سازی پس‌رو»

مثال (۷ از ۱۱)

BACKWARD CHAINING

ایده: از هدف پرس‌وجوی q به صورت پس‌رو شروع می‌کنیم:
یا درستی q معلوم است
یا مقدم‌های برخی قواعد که q را نتیجه می‌دهند باید ثابت شود (با همین روش).

زنجیره‌سازی پس‌رو
Backward Chaining

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

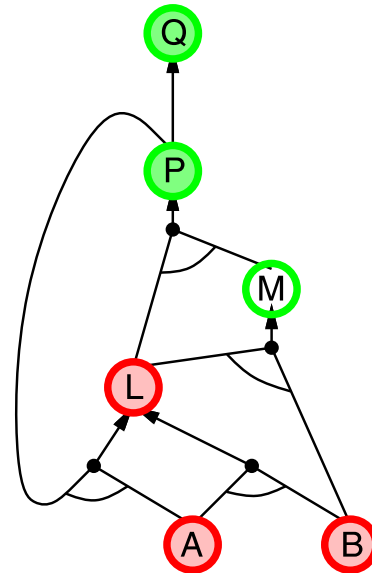
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

A

B



الگوریتم اثبات: «زنجیره‌سازی پس‌رو»

مثال (۸ از ۱۱)

BACKWARD CHAINING

ایده: از هدف پرس‌وجوی q به صورت پس‌رو شروع می‌کنیم:
یا درستی q معلوم است
یا مقدم‌های برخی قواعد که q را نتیجه می‌دهند باید ثابت شود (با همین روش).

زنجیره‌سازی پس‌رو
Backward Chaining

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

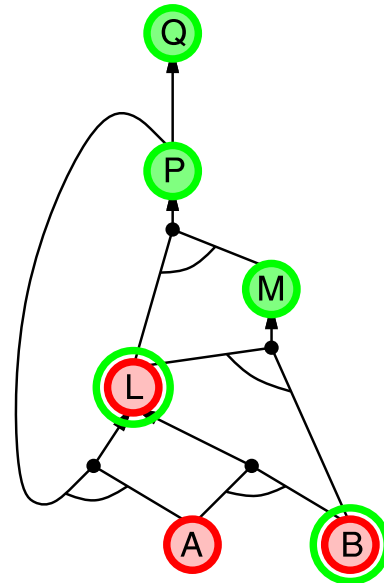
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

A

B



الگوریتم اثبات: «زنجیره‌سازی پس‌رو»

مثال (۹ از ۱۱)

BACKWARD CHAINING

ایده: از هدف پرس‌وجوی q به صورت پس‌رو شروع می‌کنیم:
یا درستی q معلوم است
یا مقدم‌های برخی قواعد که q را نتیجه می‌دهند باید ثابت شود (با همین روش).

زنجیره‌سازی پس‌رو
Backward Chaining

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

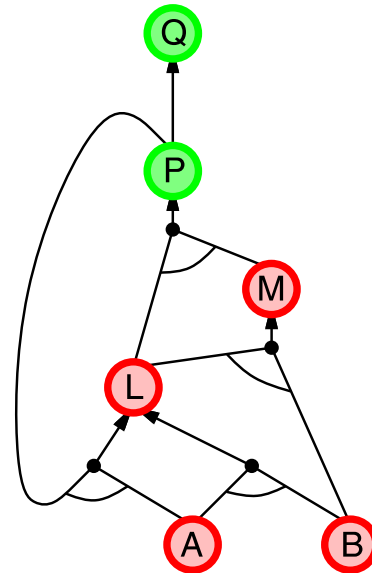
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

A

B



الگوریتم اثبات: «زنجیره‌سازی پس‌رو»

مثال (۱۰ از ۱۱)

BACKWARD CHAINING

ایده: از هدف پرس‌وجوی q به صورت پس‌رو شروع می‌کنیم:
یا درستی q معلوم است
یا مقدم‌های برخی قواعد که q را نتیجه می‌دهند باید ثابت شود (با همین روش).

زنجیره‌سازی پس‌رو
Backward Chaining

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

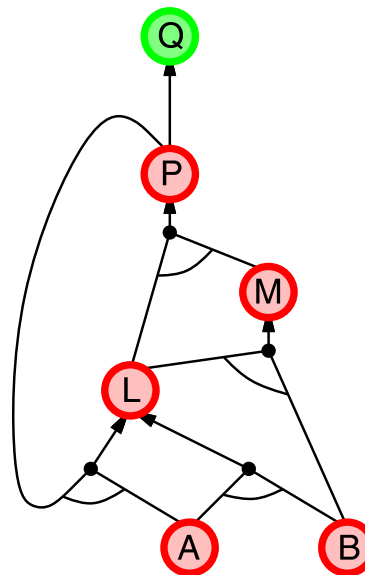
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

A

B



الگوریتم اثبات: «زنجیره‌سازی پس‌رو»

مثال (۱۱ از ۱۱)

BACKWARD CHAINING

ایده: از هدف پرس‌وجوی q به صورت پس‌رو شروع می‌کنیم:
یا درستی q معلوم است
یا مقدم‌های برخی قواعد که q را نتیجه می‌دهند باید ثابت شود (با همین روش).

زنجیره‌سازی پس‌رو
Backward Chaining

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

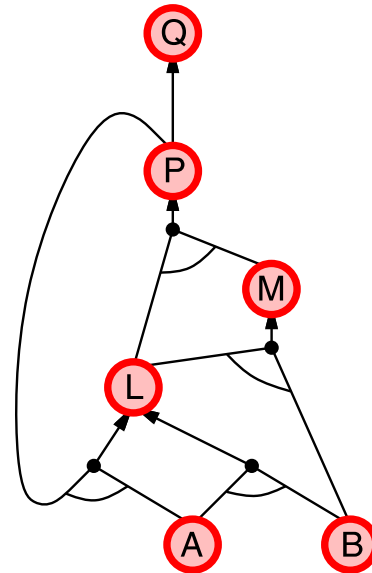
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

A

B



الگوریتم‌های اثبات «زنجیره‌سازی پیش‌رو» و «زنجیره‌سازی پس‌رو»

مقایسه

مقایسه	
زنجیره‌سازی پس‌رو <i>Backward Chaining</i>	زنجیره‌سازی پیش‌رو <i>Forward Chaining</i>
هدف محور <i>goal-driven</i>	داده محور <i>data-driven</i>
مناسب برای حل مسئله	پردازش خودکار، ناخودآگاه (<i>unconscious</i>)
کاربرد: پاسخ به یک پرسش مشخص	کاربرد: بازشناسی شیء، تصمیم‌گیری‌های روتین
پیچیدگی زمانی: بسیار کمتر از خطی (بر حسب اندازه‌ی KB)	پیچیدگی زمانی: خطی (بر حسب اندازه‌ی KB)

استنتاج با «رزولوشن»

خصوصیات روال استنتاج

$$\frac{\ell_1 \vee \dots \vee \ell_k, \quad m_1 \vee \dots \vee m_n}{\ell_1 \vee \dots \vee \ell_{i-1} \vee \ell_{i+1} \vee \dots \vee \ell_k \vee m_1 \vee \dots \vee m_{j-1} \vee m_{j+1} \vee \dots \vee m_n}$$

where ℓ_i and m_j are complementary literals.

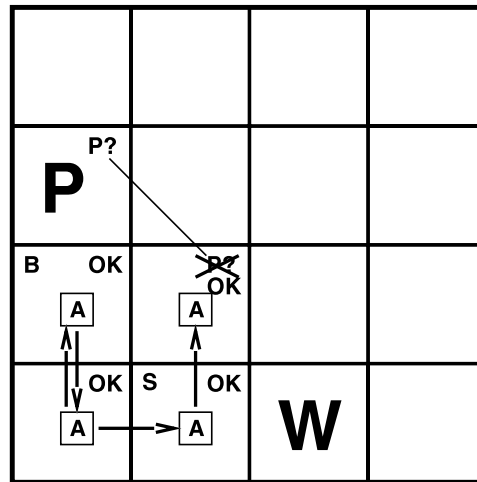
برای روال استنتاج «رزولوشن» <i>“Resolution”</i>	
تمامیت <i>Completeness</i>	صحیح بودن <i>Soundness</i>
همه‌ی جمله‌های درست مشتق شوند. برای منطق گزاره‌ای: همیشه (CNF)	هر جمله‌ی مشتق شده درست باشد. برای منطق گزاره‌ای: همیشه (CNF)

قابل استفاده در الگوریتم اثبات «رزولوشن»

استنتاج با «رزولوشن»

مثال

$$\frac{P_{1,3} \vee P_{2,2}, \quad \neg P_{2,2}}{P_{1,3}}$$



فرم نرمال عطفی

CONJUNCTIVE NORMAL FORM (CNF)

فرم نرمال عطفی

Conjunctive Normal Form (CNF)

عطف کلاوزها

Conjunctive Normal Form (CNF—universal)
conjunction of **disjunctions** of **literals**
clauses

E.g., $(A \vee \neg B) \wedge (B \vee \neg C \vee \neg D)$

فرم نرمال عطفی

تبدیل یک گزاره به CNF

CONJUNCTIVE NORMAL FORM (CNF)

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

- ۱) حذف دوشروطی: Eliminate $\Leftrightarrow$, replacing $\alpha \Leftrightarrow \beta$ with $(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$.

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

- ۲) حذف شرطی: Eliminate $\Rightarrow$, replacing $\alpha \Rightarrow \beta$ with $\neg\alpha \vee \beta$.

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

- ۳) حرکت نقیض‌ها به داخل: Move $\neg$ inwards using de Morgan's rules and double-negation:

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

- ۴) اعمال قانون توزیع‌پذیری (و روی یا): Apply distributivity law ($\vee$ over $\wedge$) and flatten:

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

الگوریتم اثبات: «رزولوشن»

بر اساس قاعده‌ی استنتاج «رزولوشن»

RESOLUTION

ایده:

اثبات از طریق تناقض (*Proof by contradiction*)
 برای اثبات $KB \models \alpha$ نشان می‌دهیم $KB \wedge \neg \alpha$ ارضاناپذیر است.

رزولوشن
Resolution

الگوریتم اثبات: «رزولوشن»

شبه‌کد

RESOLUTION

function PL-RESOLUTION(KB, α) **returns** *true* or *false*

clauses $\leftarrow$ the set of clauses in the CNF representation of $KB \wedge \neg \alpha$

new $\leftarrow \{ \}$

loop do

for each 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*

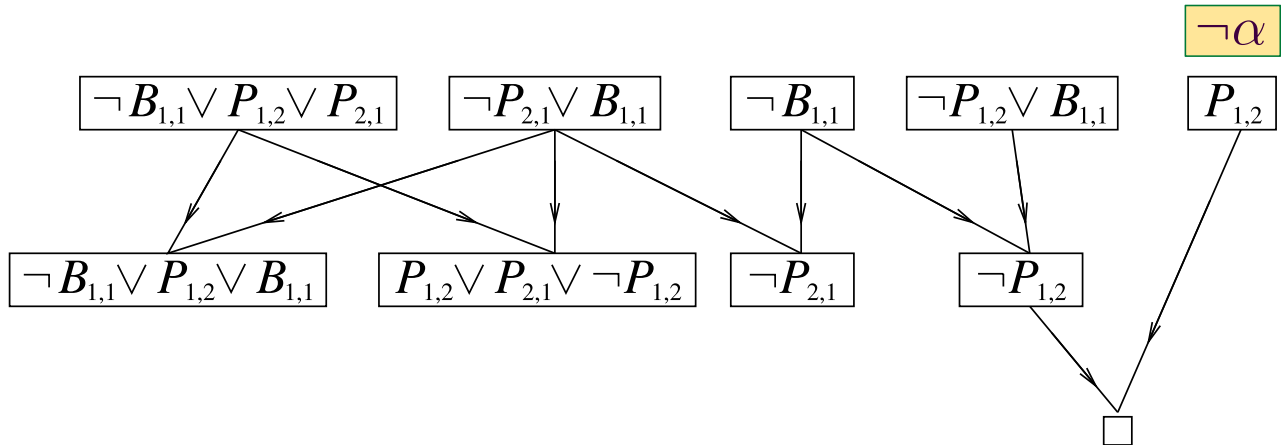
clauses $\leftarrow clauses \cup new$

الگوریتم اثبات: «رزولوشن»

مثال ۱

RESOLUTION

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

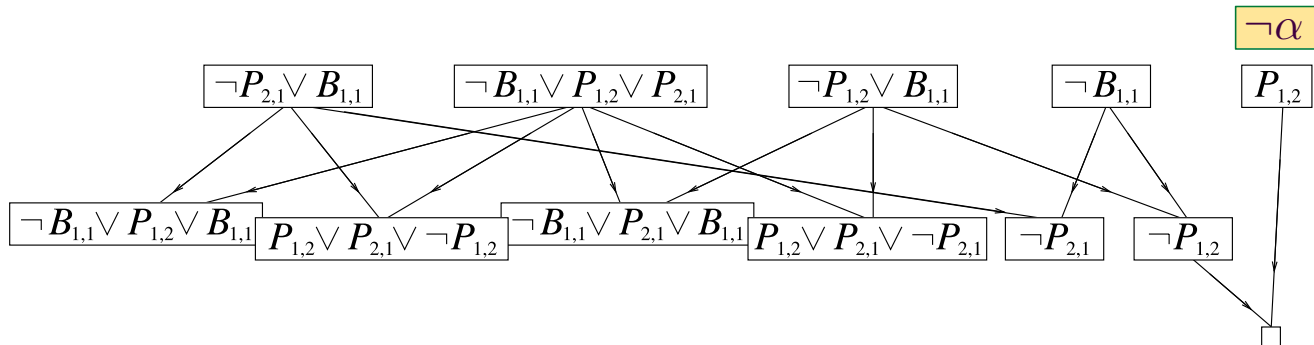


الگوریتم اثبات: «رزولوشن»

مثال ۲

RESOLUTION

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



۶

منطق گزاره‌ای

بررسی مدل مؤثر گزاره‌ای

اثبات

روش‌های شمارش جدول درستی

اثبات α : دنباله‌ای از به‌کارگیری قواعد استنتاج منتهی به α
(یافتن اثبات دقیقاً مثل یافتن راه‌حل‌های یک مسئله‌ی جستجو است.)

روش‌های اثبات

بررسی مدل <i>Model Checking</i>	به‌کارگیری قواعد استنتاج <i>Application of inference rules</i>
شمارش جدول درستی	تولید جملات جدید صحیح از قبلی‌ها
الگوریتم DPLL <i>DPLL Algorithm</i>	HF
الگوریتم WalkSAT <i>WalkSAT Algorithm</i>	HF + Modus Ponens
	CNF
	زنجیره‌سازی پیش‌رو <i>Forward Chaining</i>
	زنجیره‌سازی پس‌رو <i>Backward Chaining</i>
	رزولوشن <i>Resolution</i>

الگوریتم DPLL: عقب‌گرد++

DPLL: BACKTRACKING++

اعمال الگوریتم جستجوی عقب‌گرد به مسائل ارضاییری (SAT)

(متغیرها = نمادهای گزاره‌ای قیدها = کلاوزها)

الگوریتم DPLL

DPLL Algorithm

موارد اضافه شده به الگوریتم عقب‌گرد

توقف کنید اگر همه‌ی کلاوزها **true** یا همه‌ی آنها **false** باشند.

$$\{A = true\} \text{ satisfies } (A \vee B) \wedge (A \vee C)$$

(۱)

پایان زودهنگام

Early Termination

نمادهای دارای علامت یکسان در همه‌ی کلاوزهای «هنوز ارضاننده»
به نماد مقداری نسبت دهید که لیترال مربوطه **true** شود.

$$A, B \text{ are pure in } (A \vee \neg B) \wedge (\neg B \vee \neg C) \wedge (C \vee A)$$

(۲)

نمادهای خالص

Pure Symbols

کلاوزهای دارای دقیقاً یک لیترال «هنوز unfalsify نشده»
به نماد مقداری نسبت دهید که کلاوز مربوطه **true** شود.

if $\{A = true\}$ already $(\neg A \vee \neg B)$ is a unit clause

(۳)

کلاوزهای واحد

Unit Clauses

الگوریتم DPLL: عقب‌گرد++

شبیه‌کد

DPLL: BACKTRACKING++

```

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, [P = value | model])
  P, value ← FIND-UNIT-CLAUSE(clauses, model)
  if P is non-null then return DPLL(clauses, symbols−P, [P = value | model])
  P ← FIRST(symbols); rest ← REST(symbols)
  return DPLL(clauses, rest, [P = true | model]) or
    DPLL(clauses, rest, [P = false | model])

```

الگوریتم DPLL: عقب‌گرد++

ارزیابی کارایی

DPLL: BACKTRACKING++

پیاده‌سازی بسیار بهینه‌شده + نگهداری زیرانتساب‌های غیر قابل حل



- امکان حل مسئله با ده‌ها میلیون کلاوز
- قابل استفاده‌ی عملی برای واریسی سخت‌افزاری بزرگ
- قابل استفاده‌ی عملی برای واریسی نرم‌افزاری متوسط

الگوریتم WalkSAT

به کارگیری الگوریتم جستجوی محلی برای مسئله‌ی ارضاپذیری

WALKSAT ALGORITHM

اعمال الگوریتم جستجوی محلی به مسائل ارضاپذیری (SAT)

تابع ارزیابی = تعداد کلاوزهای ارضاننده

با استفاده از الگوریتم MIN-CONFLICTS برای CSPها

الگوریتم WalkSAT

WalkSAT Algorithm

الگوریتم WalkSAT در هر تکرار دو کار انجام می‌دهد:

(۱)

انتخاب یک کلاوز ارضاننده

picks an unsatisfied clause

انتخاب تصادفی بین دو راه برای انتخاب نماد برای برگردان

Min-conflicts step

گام حداقل تداخلها

تعداد کلاوزهای ارضاننده در حالت جدید را می‌نیم می‌کند.

Random walk step

گام گام‌برداری تصادفی

یک نماد را به صورت تصادفی انتخاب می‌کند.

(۲)

انتخاب یک نماد در کلاوز برای برگردان

picks a symbol in the clause to flip

الگوریتم WalkSAT

شبه‌کد

WALKSAT ALGORITHM

```

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  $\leftarrow$  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  $\leftarrow$  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 WalkSAT

ارزیابی کارایی: مسئله‌های ارضاپذیری سخت

WALKSAT ALGORITHM: HARD SATISFIABILITY PROBLEM

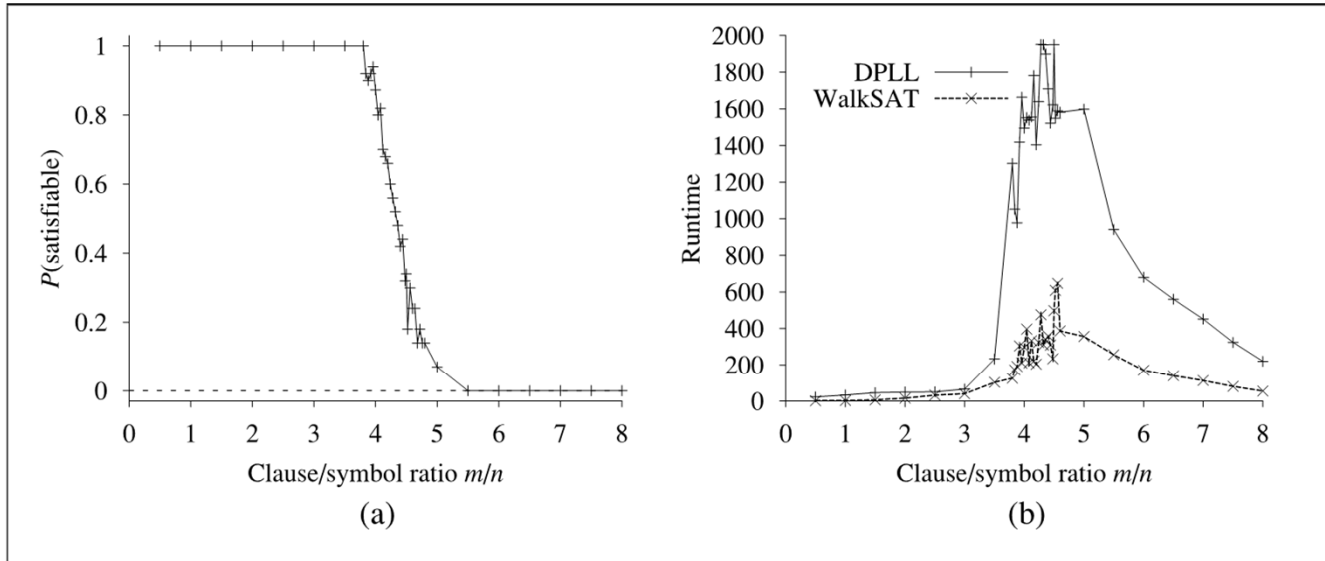


Figure 7.19 (a) Graph showing the probability that a random 3-CNF sentence with $n = 50$ symbols is satisfiable, as a function of the clause/symbol ratio m/n . (b) Graph of the median run time (measured in number of recursive calls to DPLL, a good proxy) on random 3-CNF sentences. The most difficult problems have a clause/symbol ratio of about 4.3.

۷

عوامل‌های
مبتنی بر
منطق
گزاره‌ای

گزاره‌ها و زمان

ارزش یک نماد گزاره‌ای می‌تواند متغیر با زمان باشد



باید برای هر گام زمانی، یک نماد گزاره‌ای در نظر گرفت
(احتمالاً بی‌نهایت گام زمانی ...)

گزاره‌ها و زمان

مثال: دنیای اژدها

Suppose the wumpus-world agent wants to keep track of its location

A sentence such as $L_{1,1} \wedge FacingRight \wedge Forward \Rightarrow L_{2,1}$
doesn't work: after one inference step, $L_{1,1}$ and $L_{2,1}$ are in KB!!

Changeable aspects of world need separate symbols for each time step
e.g., $L_{1,1}^1$ means "Agent is at $[1, 1]$ at time step 1", and

$$L_{1,1}^1 \wedge FacingRight^1 \wedge Forward^1 \Rightarrow L_{2,1}^2$$

Reflex rules: for every t , we have, e.g., $Glitter^t \Rightarrow Grab^t$

دنبال کردن تغییرات در دنیا

TRACKING CHANGES IN THE WORLD

وظیفه‌ی عمومی دنبال کردن حالت محیط
با داشتن یک جریان از ادراک‌ها

تخمین حالت

State Estimation

در سیستم‌های مبتنی بر منطق:

نگهداری یک بازنمایی از مجموعه‌ی همه‌ی حالت‌های ممکن دنیا،
با داشتن اصول موضوع (axioms) و ادراک‌ها (percepts)

ترفند پایه

اصول موضوع حالت-مابعد

Successor-State Axioms

تعریف درستی گزاره در زمان $t + 1$ از روی گزاره‌های زمان t

دنبال کردن تغییرات در دنیا

مثال: دنیای اژدها

TRACKING CHANGES IN THE WORLD

اصول موضوع حالت-مابعد

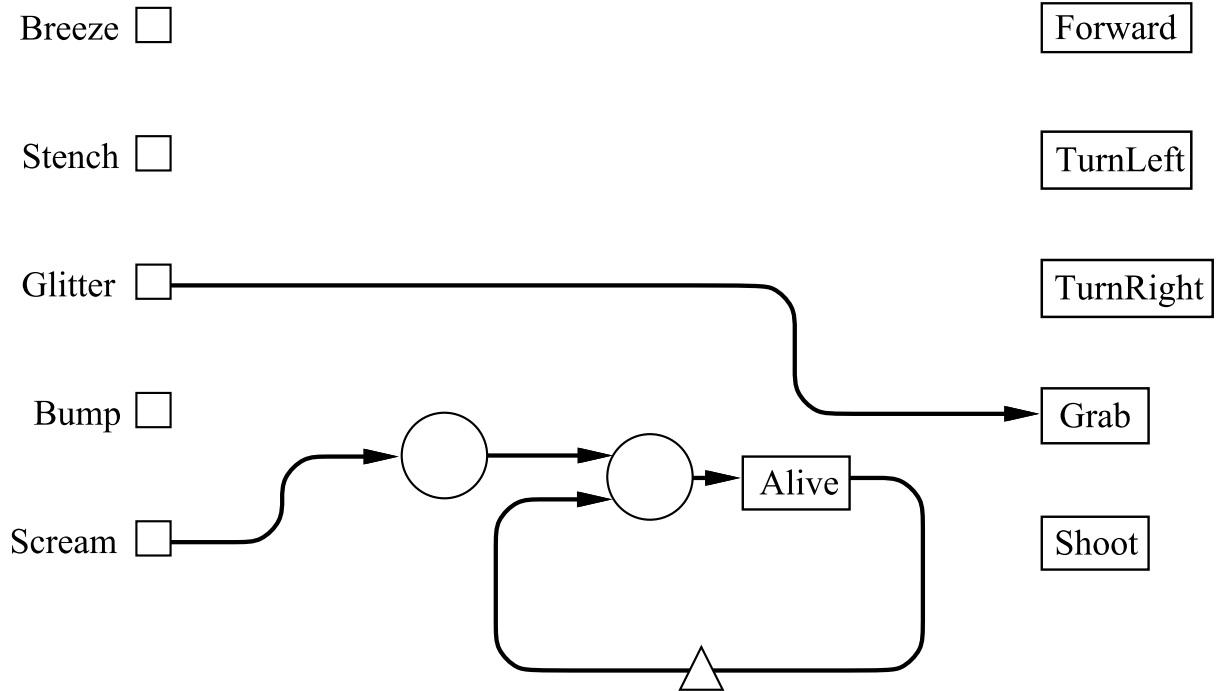
*Successor-State Axioms*تعریف درستی گزاره در زمان $t + 1$ از روی گزاره‌های زمان t

$$\text{E.g., } \text{Alive}^t \Leftrightarrow \neg \text{Scream}^t \wedge \text{Alive}^{t-1}$$

$$\begin{aligned} L_{1,1}^t \Leftrightarrow & (L_{1,1}^{t-1} \wedge (\neg \text{Forward}^{t-1} \vee \text{Bump}^t)) \\ & \vee (L_{1,2}^{t-1} \wedge (\text{FacingDown}^{t-1} \wedge \text{Forward}^{t-1})) \\ & \vee (L_{1,2}^{t-1} \wedge (\text{FacingLeft}^{t-1} \wedge \text{Forward}^{t-1})) \end{aligned}$$

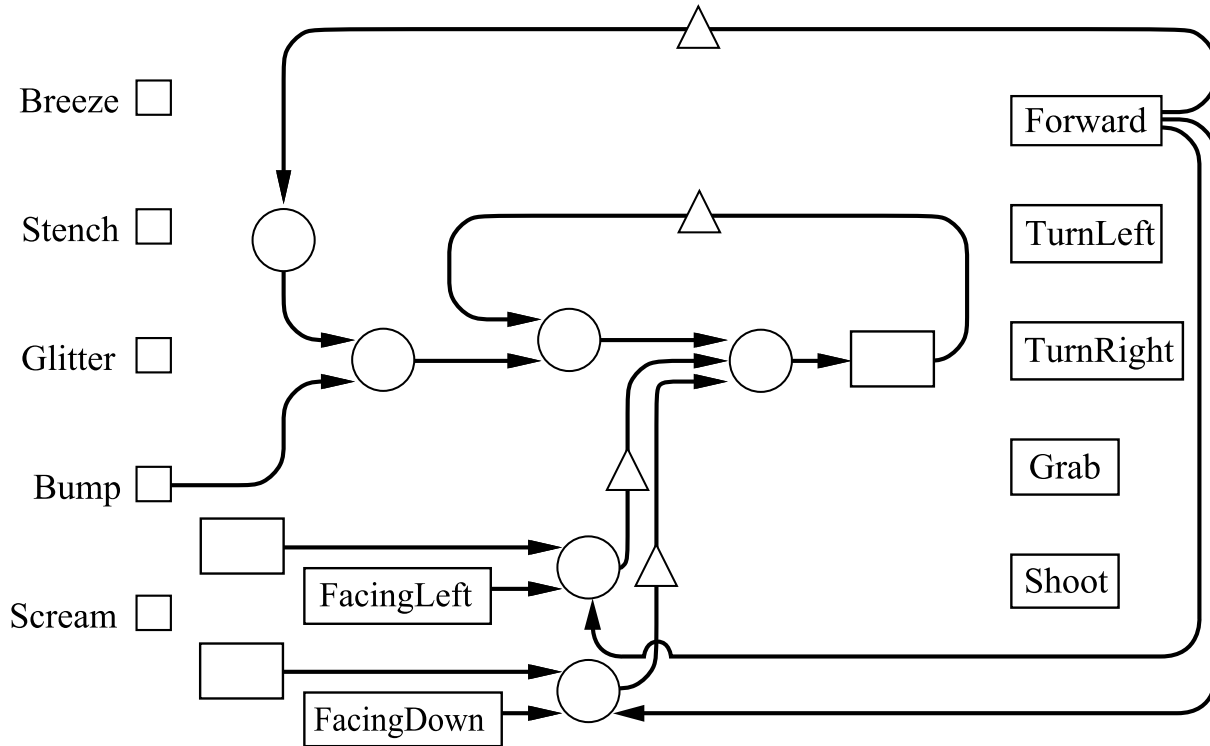
عامل‌های مبتنی بر مدارهای بولی

مثال (۱ از ۲)

BOOLEAN CIRCUIT AGENTS

عامل‌های مبتنی بر مدارهای بولی

مثال (۲ از ۲)

BOOLEAN CIRCUIT AGENTS

عامل‌های مبتنی بر مدارهای بولی

چند نکته

BOOLEAN CIRCUIT AGENTS

عامل‌های مبتنی بر مدار، راه ساده‌ای برای کار با زمان فراهم می‌کنند، اما معمولاً نسبت به عامل‌های مبتنی بر استنتاج کمتر کامل هستند.

- مقادیر نامعلوم باید در نظر گرفته شوند
با استفاده از گزاره‌ی دانایی (knowledge proposition)

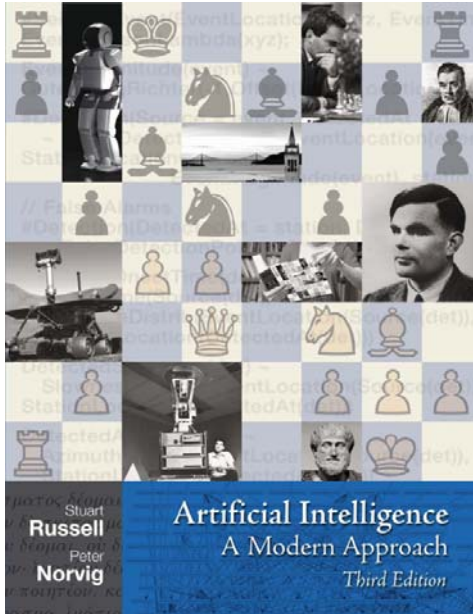
$$K(B) \quad \text{and} \quad K(\neg B) \quad \text{for} \quad B$$

- مدار باید بدون چرخه (acyclic) باشد.
- محیط مسئله باید محلی (local) باشد.

عوامل‌های منطقی



منابع،
مطالعه،
تکلیف



Stuart Russell and Peter Norvig,
Artificial Intelligence: A Modern Approach,
 3rd Edition, Prentice Hall, 2010.

Chapter 7

7 LOGICAL AGENTS

In which we design agents that can form representations of a complex world, use a process of inference to derive new representations about the world, and use these new representations to deduce what to do.

REASONING
 REPRESENTATION
 KNOWLEDGE-BASED
 AGENTS

Humans, it seems, know things; and what they know helps them do things. These are not empty statements. They make strong claims about how the intelligence of humans is achieved—not by purely reflex mechanisms but by processes of **reasoning** that operate on internal **representations** of knowledge. In AI, this approach to intelligence is embodied in **knowledge-based agents**.

The problem-solving agents of Chapters 3 and 4 know things, but only in a very limited, inflexible sense. For example, the transition model for the 8-puzzle—knowledge of what the actions do—is hidden inside the domain-specific code of the **RESULT** function. It can be used to predict the outcome of actions but not to deduce that two tiles cannot occupy the same space or that states with odd parity cannot be reached from states with even parity. The atomic representations used by problem-solving agents are also very limiting. In a partially observable environment, an agent's only choice for representing what it knows about the current state is to list all possible concrete states—a hopeless prospect in large environments.

Chapter 6 introduced the idea of representing states as assignments of values to variables; this is a step in the right direction, enabling some parts of the agent to work in a domain-independent way and allowing for more efficient algorithms. In this chapter and those that follow, we take this step to its logical conclusion, so to speak—we develop **logic** as a general class of representations to support knowledge-based agents. Such agents can combine and recombine information to suit myriad purposes. Often, this process can be quite far removed from the needs of the moment—as when a mathematician proves a theorem or an astronomer calculates the earth's life expectancy. Knowledge-based agents can accept new tasks in the form of explicitly described goals; they can achieve competence quickly by being told or learning new knowledge about the environment; and they can adapt to changes in the environment by updating the relevant knowledge.

We begin in Section 7.1 with the overall agent design. Section 7.2 introduces a simple new environment, the wumpus world, and illustrates the operation of a knowledge-based agent without going into any technical detail. Then we explain the general principles of **logic**

LOGIC