هوش مصنوعی

فصل ۹

استنتاج در منطق مرتبه اول

Inference in First-Order Logic

کاظم فولادی

دانشکده مهندسی برق و کامپیوتر

دانشگاه تهران

http://courses.fouladi.ir/ai

تاریخچه‌ی مختصر استدلال

A BRIEF HISTORY OF REASONING

450B.C.	Stoics	propositional logic, inference (maybe)
322B.C.	Aristotle	"syllogisms" (inference rules), quantifiers
1565	Cardano	probability theory (propositional logic + uncertainty)
1847	Boole	propositional logic (again)
1879	Frege	first-order logic
1922	Wittgenstein	proof by truth tables
1930	Gödel	$\exists$ complete algorithm for FOL
1930	Herbrand	complete algorithm for FOL (reduce to propositional)
1931	Gödel	$\neg\exists$ complete algorithm for arithmetic
1960	Davis/Putnam	"practical" algorithm for propositional logic
1965	Robinson	"practical" algorithm for FOL—resolution

۱

استنتاج گزاره‌ای دربرابر مرتبه اول

نمونه‌سازی عمومی

UNIVERSAL INSTANTIATION (UI)

نمونه‌سازی عمومی
Universal instantiation (UI)

هر نمونه‌سازی یک جمله‌ی مسور عمومی، از آن استلزام می‌شود.

$$\frac{\forall v \ \ \alpha}{\text{SUBST}(\{v/g\}, \alpha)}$$

for any variable v and ground term g

E.g., $\forall x \ \ King(x) \wedge Greedy(x) \Rightarrow Evil(x)$ yields

$$King(John) \wedge Greedy(John) \Rightarrow Evil(John)$$
$$King(Richard) \wedge Greedy(Richard) \Rightarrow Evil(Richard)$$
$$King(Father(John)) \wedge Greedy(Father(John)) \Rightarrow Evil(Father(John))$$
$$\vdots$$

واقعیت زمینی

GROUND FACT

واقعیت زمینی *Ground Fact*	واقعیتی که در آن هیچ متغیری وجود ندارد. (فقط ثابت‌ها حضور دارند)

$$King(John),\ Greedy(John),\ Evil(John), King(Richard)$$

نمونه‌سازی وجودی

EXISTENTIAL INSTANTIATION (EI)

نمونه‌سازی وجودی
Existential instantiation (EI)

به‌جای هر متغیر سور وجودی می‌توان یک نماد ثابت انحصاری قرار داد.
(ثابت اسکولم)

For any sentence α, variable v, and constant symbol k
that does not appear elsewhere in the knowledge base:

$$\frac{\exists v \;\; \alpha}{\text{SUBST}(\{v/k\}, \alpha)}$$

E.g., $\exists x \;\; Crown(x) \wedge OnHead(x, John)$ yields

$$Crown(C_1) \wedge OnHead(C_1, John)$$

provided C_1 is a new constant symbol, called a Skolem constant

اسکولمیزاسیون

ثابت اسکولم

SKOLEMIZATION: SKOLEM CONSTANT

ثابت اسکولم
Skolem Constant

ثابتی که به‌جای یک متغیر سور وجودی قرار می‌گیرد.

E.g., $\exists x \;\; Crown(x) \wedge OnHead(x, John)$ yields

$$Crown(C_1) \wedge OnHead(C_1, John)$$

provided C_1 is a new constant symbol, called a Skolem constant

Another example: from $\exists x \;\; d(x^y)/dy = x^y$ we obtain

$$d(e^y)/dy = e^y$$

provided e is a new constant symbol

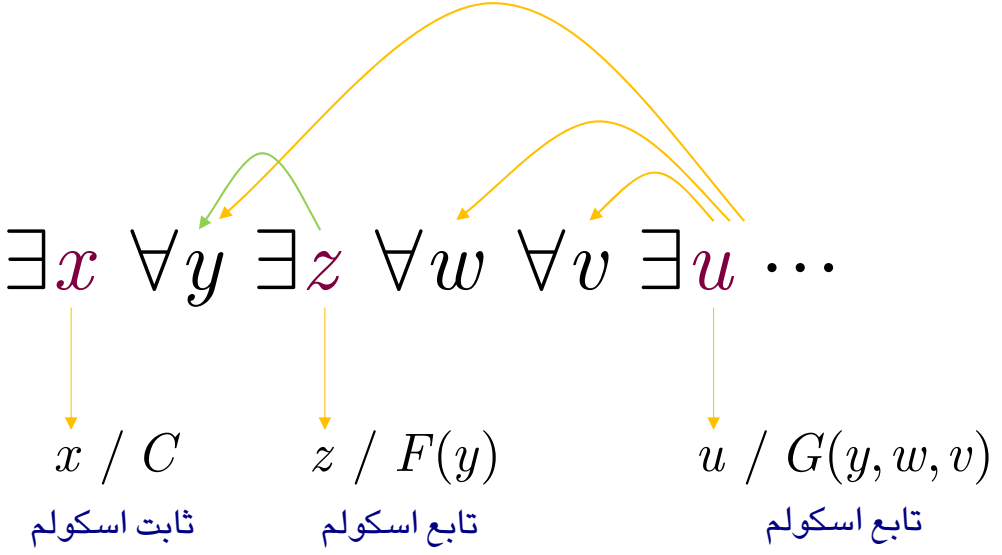
اسکولمیزاسیون

تابع اسکولم

SKOLEMIZATION: SKOLEM FUNCTION

تابع اسکولم
Skolem Function

تابعی که به‌جای یک متغیر سور وجودی قرار می‌گیرد.
(مواردی که متغیر سور وجودی در حوزه‌ی دید متغیر سور عمومی قرار دارد.)



نمونه‌سازی

مقایسه‌ی نمونه‌سازی عمومی با نمونه‌سازی وجودی

INSTANTIATION

نمونه‌سازی عمومی *Universal instantiation* *(UI)*	نمونه‌سازی وجودی *Existential instantiation* *(EI)*
می‌تواند به دفعات انجام شود تا جمله‌های جدیدی **اضافه** شوند.	فقط یک بار می‌تواند انجام شود تا جمله‌ی وجودی **جایگزین** شود.
KB جدید هم‌ارز منطقی KB قبلی است.	KB جدید هم‌ارز منطقی KB قبلی نیست.
	KB جدید ارضاپذیر است اگر KB قبلی ارضاپذیر باشد.

گزاره‌ای‌سازی

کاهش پایگاه دانایی منطق مرتبه اول به پایگاه دانایی منطق گزاره‌ای

PROPOSITIONALIZATION

گزاره‌ای‌سازی *Propositionalization*	نمونه‌سازی همه‌ی متغیرهای سور عمومی با **همه‌ی راه‌های ممکن**

گزاره‌ای‌سازی

مثال

PROPOSITIONALIZATION

فرض کنید یک KB فقط حاوی جملات زیر باشد:

$$\forall x \;\; King(x) \wedge Greedy(x) \;\Rightarrow\; Evil(x)$$
$$King(John)$$
$$Greedy(John)$$
$$Brother(Richard, John)$$

با نمونه‌سازی جمله‌ی عمومی با همه‌ی راه‌های ممکن (گزاره‌ای‌سازی) داریم:

$$King(John) \wedge Greedy(John) \;\Rightarrow\; Evil(John)$$
$$King(Richard) \wedge Greedy(Richard) \;\Rightarrow\; Evil(Richard)$$
$$King(John)$$
$$Greedy(John)$$
$$Brother(Richard, John)$$

KB جدید **گزاره‌ای‌سازی** شده است؛ نمادهای گزاره‌ای عبارتند از:

$$King(John),\;\; Greedy(John),\;\; Evil(John), King(Richard)$$

گزاره‌ای‌سازی

ملاحظات «کاهش پایگاه دانایی منطق مرتبه اول به پایگاه دانایی منطق گزاره‌ای»

PROPOSITIONALIZATION

ادعا: یک جمله‌ی زمینی (ground) به‌وسیله‌ی KB جدید استلزام می‌شود
اگر و فقط اگر
توسط KB اصلی استلزام شود.

ادعا: هر KB منطق مرتبه اول می‌تواند گزاره‌ای‌سازی شود، به‌گونه‌ای که استلزام حفظ شود.

ایده برای استلزام: ۱) گزاره‌ای‌سازی KB و پرسش، ۲) اعمال رزولوشن، ۳) برگرداندن نتیجه

مشکل: با وجود نمادهای تابع، بی‌نهایت ترم زمینی وجود دارد:

$$Father(John)$$
$$Father(Father(John))$$
$$Father(Father(Father(John)))$$
$$\vdots$$

قضیه‌ی استلزام در منطق مرتبه اول

قضیه هربرند (۱۹۳۰م) ـ قضیه‌ی چرچ–تورینگ (۱۹۳۶م)

قضیه
Herbrand (1930)

اگر یک جمله‌ی α توسط یک **FOL KB** استلزام شود،
آن‌گاه با زیرمجموعه‌ای **متناهی** از **KB گزاره‌ای** استلزام می‌شود.

ایده

به‌ازای $n = 0$ تا ∞
۱) یک KB گزاره‌ای بسازید (با نمونه‌سازی با ترم‌های عمق n)
۲) ببینید آیا α از این KB استلزام می‌شود یا خیر؟

مشکل

اگر α استلزام شود، کار می‌کند
اگر α استلزام نشود، ممکن است در حلقه‌ی نامتناهی بیفتد!

قضیه
Turing & Chruch (1936)

استلزام در منطق مرتبه اول نیمه‌تصمیم‌پذیر (semidecidable) است.

گزاره‌ای‌سازی

مشکلات «کاهش پایگاه دانایی منطق مرتبه اول به پایگاه دانایی منطق گزاره‌ای»

PROPOSITIONALIZATION

گزاره‌ای‌سازی، تعداد بسیار زیادی جمله‌ی نامربوط تولید می‌کند.

اگر p محمول k–تایی و n نماد ثابت داشته باشیم، تعداد نمونه‌سازی‌سازی‌ها عبارت است از:

$$p \cdot n^k$$

با وجود نمادهای تابع، اوضاع بسیار بدتر است!

گزاره‌ای‌سازی

مشکلات «کاهش پایگاه دانایی منطق مرتبه اول به پایگاه دانایی منطق گزاره‌ای»: مثال

PROPOSITIONALIZATION

گزاره‌ای‌سازی، تعداد بسیار زیادی جمله‌ی نامربوط تولید می‌کند.

برای مثال، از

$$\forall x \;\; King(x) \wedge Greedy(x) \Rightarrow Evil(x)$$
$$King(John)$$
$$\forall y \;\; Greedy(y)$$
$$Brother(Richard, John)$$

بدیهی است که $Evil(John)$،
اما گزاره‌ای‌سازی تعداد بسیار زیادی واقعیت مانند $Greedy(Richard)$
تولید می‌کند که نامربوط هستند.

هوش مصنوعی

استنتاج در منطق مرتبه اول

۲

یکسان‌سازی

یکسان‌سازی

UNIFICATION

یکسان‌سازی *Unification*	تعیین مقادیری برای متغیرهای دو عبارت به‌گونه‌ای که آن دو عبارت یکسان شوند.

$$\text{UNIFY}(\alpha, \beta) = \theta \text{ if } \alpha\theta = \beta\theta$$

یکسان‌سازی امکان استنتاج بدون نیاز به گزاره‌ای‌سازی را فراهم می‌کند.

عمومی‌ترین یکسان‌ساز (most general unifier) آن است که تا جای ممکن متغیرها را با ثابت‌ها جایگزین نکند.

یکسان‌سازی

شبه‌کد

UNIFICATION

function UNIFY(x, y, θ) **returns** a substitution to make x and y identical
  inputs: x, a variable, constant, list, or compound expression
    y, a variable, constant, list, or compound expression
    θ, the substitution built up so far (optional, defaults to empty)

  if θ = failure **then return** failure
  else if $x = y$ **then return** θ
  else if VARIABLE?(x) **then return** UNIFY-VAR(x, y, θ)
  else if VARIABLE?(y) **then return** UNIFY-VAR(y, x, θ)
  else if COMPOUND?(x) **and** COMPOUND?(y) **then**
    return UNIFY(x.ARGS, y.ARGS, UNIFY(x.OP, y.OP, θ))
  else if LIST?(x) **and** LIST?(y) **then**
    return UNIFY(x.REST, y.REST, UNIFY(x.FIRST, y.FIRST, θ))
  else return failure

function UNIFY-VAR(var, x, θ) **returns** a substitution

  if $\{var/val\} \in \theta$ **then return** UNIFY(val, x, θ)
  else if $\{x/val\} \in \theta$ **then return** UNIFY(var, val, θ)
  else if OCCUR-CHECK?(var, x) **then return** failure
  else return add $\{var/x\}$ to θ

یکسان‌سازی

مثال ۱

UNIFICATION

We can get the inference immediately if we can find a substitution θ such that $King(x)$ and $Greedy(x)$ match $King(John)$ and $Greedy(y)$

$\theta = \{x/John, y/John\}$ works

یکسان‌سازی

مثال ۲

UNIFICATION

p	q	θ
$Knows(John, x)$	$Knows(John, Jane)$	$\{x/Jane\}$
$Knows(John, x)$	$Knows(y, OJ)$	$\{x/OJ, y/John\}$
$Knows(John, x)$	$Knows(y, Mother(y))$	$\{y/John, x/Mother(John)\}$
$Knows(John, x)$	$Knows(x, OJ)$	$fail$

Standardizing apart eliminates overlap of variables, e.g., $Knows(z_{17}, OJ)$

یکسان‌سازی

شکست

UNIFICATION: FAILURE

شکست در یکسان‌سازی	
یکسان‌سازی یک ثابت با ثابت متفاوت دیگر	یکسان‌سازی با جانشانی یک متغیر با تابعی از آن متغیر
$\{C \ / \ D\}$	$\{x \ / \ F(x)\}$
※	※

استانداردسازی با جداسازی

STANDARDIZING APART

استانداردسازی با جداسازی *Standardizing apart*	نام‌گذاری متغیرهای هر سور با شناسه‌های مجزا

قیاس استثنائی تعمیم‌یافته

GENERALIZED MODUS PONENS (GMP)

$$\frac{p_1',\ p_2',\ \ldots,\ p_n',\ \ (p_1 \wedge p_2 \wedge \ldots \wedge p_n \Rightarrow q)}{q\theta}$$

where $p_i'\theta = p_i\theta$ for all i

قابل استفاده با KBهای **کلاوزهای معین** (دقیقاً یک لیترال مثبت)

با فرض: همه‌ی متغیرها با سور عمومی

قیاس استثنائی تعمیم‌یافته

مثال

GENERALIZED MODUS PONENS (GMP)

$$\frac{p_1',\ p_2',\ \ldots,\ p_n',\ \ (p_1 \wedge p_2 \wedge \ldots \wedge p_n \Rightarrow q)}{q\theta}$$

where $p_i'\theta = p_i\theta$ for all i

p_1' is $King(John)$ $\quad$ p_1 is $King(x)$
p_2' is $Greedy(y)$ $\quad$ p_2 is $Greedy(x)$
θ is $\{x/John, y/John\}$ $\quad$ q is $Evil(x)$
$q\theta$ is $Evil(John)$

استنتاج با «قیاس استثنائی تعمیم‌یافته»

خصوصیات روال استنتاج

$$\frac{p_1',\ p_2',\ \ldots,\ p_n',\ \ (p_1 \wedge p_2 \wedge \ldots \wedge p_n \Rightarrow q)}{q\theta}$$

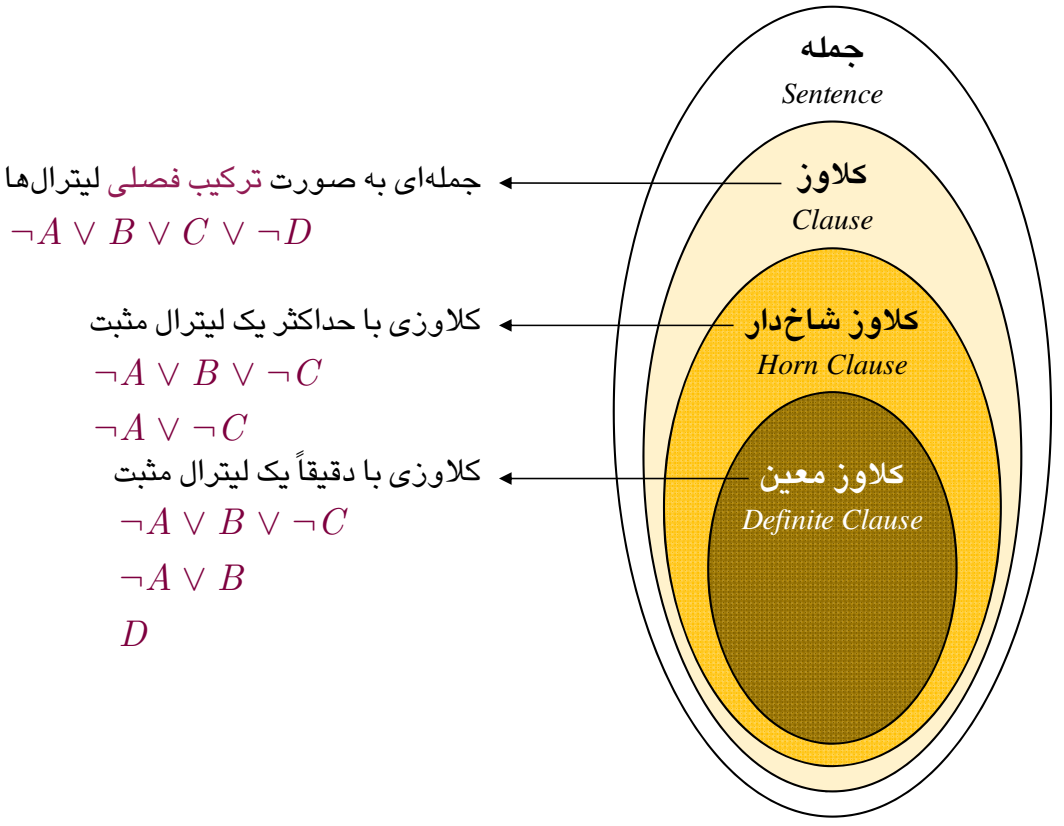
where $p_i'\theta = p_i\theta$ for all i

برای روال استنتاج «قیاس استثنائی تعمیم‌یافته» *"Generalized Modus Ponens (GMP)"*	
صحیح بودن *Soundness*	**تمامیت** *Completeness*
هر جمله‌ی مشتق شده درست باشد. همیشه	**همه‌ی جمله‌های درست مشتق شوند.** فقط برای جملات به فرم کلاوز معین (Definite Clause) • استلزام با کلاوزهای معین **نیمه‌تصمیم‌پذیر** است. • استلزام برای Datalog **تصمیم‌پذیر** است.

قابل استفاده در **زنجیره‌سازی پیشرو** و **زنجیره‌سازی پسرو**

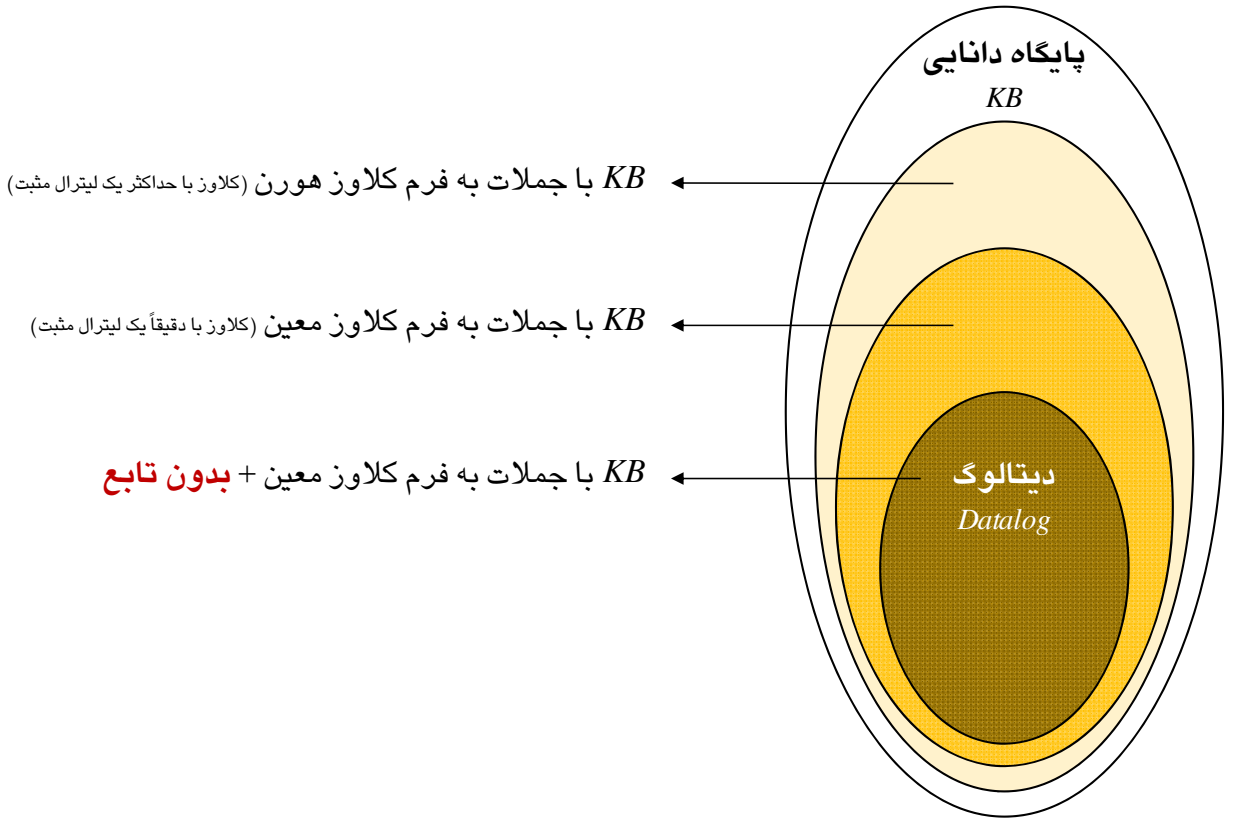
کلاوز معین

DEFINITE CLAUSE



دیتالوگ

DATALOG



قیاس استثنائی تعمیم‌یافته

اثبات درستی

GENERALIZED MODUS PONENS (GMP)

Need to show that

$$p_1', \ldots, p_n', \quad (p_1 \wedge \ldots \wedge p_n \Rightarrow q) \models q\theta$$

provided that $p_i'\theta = p_i\theta$ for all i

Lemma: For any definite clause p, we have $p \models p\theta$ by UI

1. $(p_1 \wedge \ldots \wedge p_n \Rightarrow q) \models (p_1 \wedge \ldots \wedge p_n \Rightarrow q)\theta = (p_1\theta \wedge \ldots \wedge p_n\theta \Rightarrow q\theta)$
2. $p_1', \ldots, p_n' \models p_1' \wedge \ldots \wedge p_n' \models p_1'\theta \wedge \ldots \wedge p_n'\theta$
3. From 1 and 2, $q\theta$ follows by ordinary Modus Ponens

اثبات در منطق مرتبه اول

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

به‌کارگیری قواعد استنتاج *Application of inference rules*	بررسی مدل *Model Checking*
تولید جملات جدید صحیح از قبلی‌ها	

- HF + Gen. Modus Ponens
  - زنجیره‌سازی پیش‌رو — *Forward Chaining*
  - زنجیره‌سازی پس‌رو — *Backward Chaining*
- CNF
  - رزولوشن — *Resolution*

هوش مصنوعی

استنتاج در منطق مرتبه اول

۳

زنجیره‌سازی پیش‌رو

پایگاه دانایی با «منطق مرتبه اول»

مثال

The law says that
it is a crime for an American to sell weapons to hostile nations.
The country Nono, an enemy of America, has some missiles,
and all of its missiles were sold to it by Colonel West,
who is American.

Prove that Col. West is a criminal

قانون ایالات متحده می‌گوید که:
برای یک آمریکایی این یک جرم است که به کشورهای متخاصم سلاح بفروشد.
کشور Nono، یک دشمن آمریکا، تعدادی موشک دارد،
و همه‌ی موشک‌های آن توسط سرهنگ وست فروخته شده است،
که یک آمریکایی است.

ثابت کنید که سرهنگ وست مجرم است.

پایگاه دانایی با «منطق مرتبه اول»

مثال: تبدیل جملات به منطق مرتبه اول

... برای یک آمریکایی این یک جرم است که به کشورهای متخاصم سلاح بفروشد:

$$American(x) \wedge Weapon(y) \wedge Sells(x, y, z) \wedge Hostile(z) \Rightarrow Criminal(x)$$

... نونو ... تعدادی موشک دارد: $\exists x\ Owns(Nono, x) \wedge Missile(x)$:

$$Owns(Nono, M_1) \text{ and } Missile(M_1)$$

... همه‌ی موشک‌های آن توسط سرهنگ وست فروخته شده است:

$$\forall x\ \ Missile(x) \wedge Owns(Nono, x) \ \Rightarrow\ Sells(West, x, Nono)$$

Missiles are weapons: موشک‌ها، سلاح هستند

$$Missile(x) \Rightarrow Weapon(x)$$

An enemy of America counts as "hostile": دشمن آمریکا «متخاصم» شمرده می‌شود

$$Enemy(x, America) \ \Rightarrow\ Hostile(x)$$

West, who is American ... ... وست، آمریکایی است

$$American(West)$$

The country Nono, an enemy of America ... ... کشور نونو، یک دشمن آمریکا

$$Enemy(Nono, America)$$

الگوریتم اثبات: «زنجیره‌سازی پیش‌رو»

بر اساس قاعده‌ی استنتاج «قیاس استثنائی تعمیم‌یافته»

FORWARD CHAINING

زنجیره‌سازی پیش‌رو
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

شبه‌کد الگوریتم

FORWARD CHAINING ALGORITHM

```
function FOL-FC-Ask(KB, α) returns a substitution or false
  repeat until new is empty
    new ← { }
    for each sentence r in KB do
      (p1 ∧ ... ∧ pn ⇒ q) ← Standardize-Apart(r)
      for each θ such that (p1 ∧ ... ∧ pn)θ = (p'1 ∧ ... ∧ p'n)θ
                  for some p'1, ..., p'n in KB
        q' ← Subst(θ, q)
        if q' is not a renaming of a sentence already in KB or new then do
          add q' to new
          φ ← Unify(q', α)
          if φ is not fail then return φ
    add new to KB
  return false
```

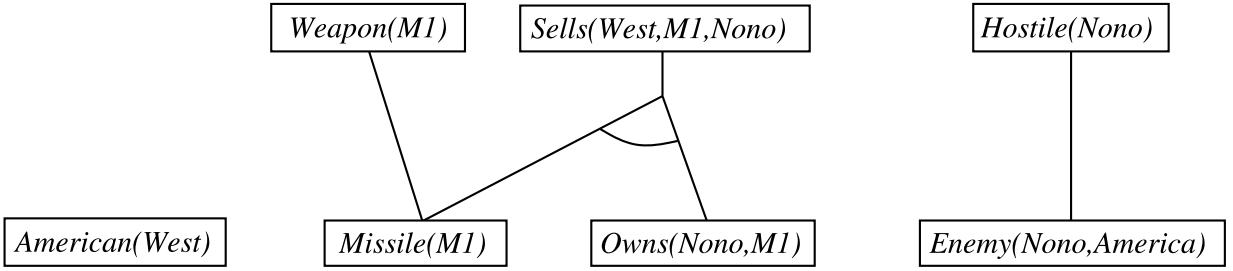
اثبات با «زنجیره‌سازی پیش‌رو»

مثال (۱ از ۳)

American(West)	*Missile(M1)*	*Owns(Nono,M1)*	*Enemy(Nono,America)*

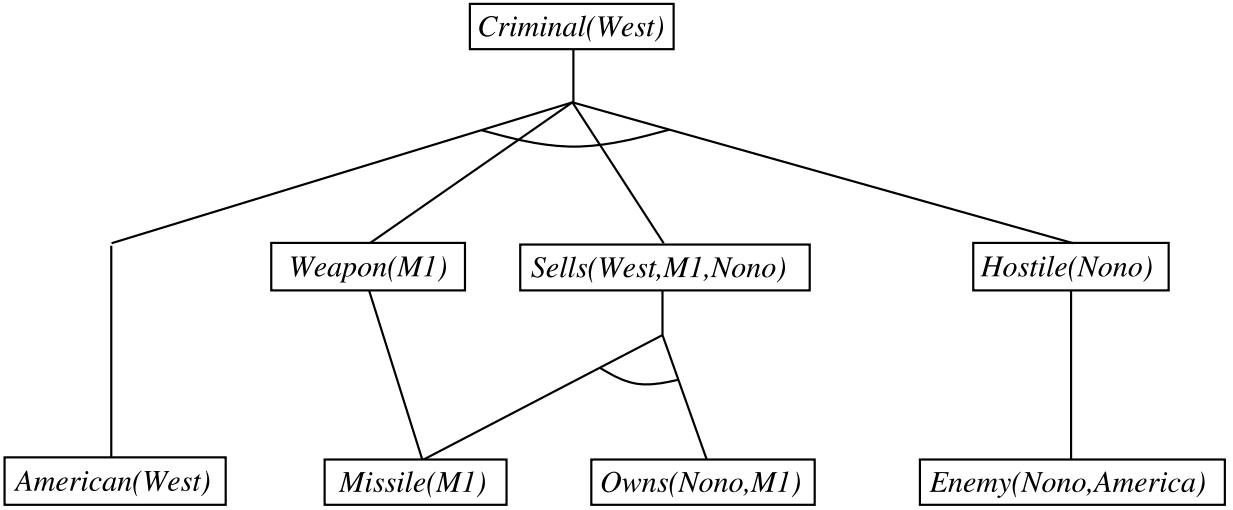
اثبات با «زنجیره‌سازی پیش‌رو»

مثال (۲ از ۳)



اثبات با «زنجیره‌سازی پیش‌رو»

مثال (۳ از ۳)



اثبات با «زنجیره‌سازی پیش‌رو»

خصوصیات «زنجیره‌سازی پیش‌رو»

PROPERTIES OF FORWARD CHAINING

زنجیره‌سازی پیش‌رو

برای دیتالوگ، در تعداد تکرارهای چندجمله‌ای خاتمه می‌یابد

(در دیتالوگ حداکثر $p \cdot n^k$ لیترال داریم)

استلزام با کلاوزهای معین، **نیمه‌تصمیم‌پذیر** است:

* اگر α استلزام شود، خاتمه می‌یابد.
* اگر α استلزام نشود، ممکن است در حلقه‌ی نامتناهی بیفتد.

اثبات با «زنجیره‌سازی پیش‌رو»

کارآمدی «زنجیره‌سازی پیش‌رو»

EFFICIENCY OF FORWARD CHAINING

عملیات تطابق با مقدم‌های قواعد در «زنجیره‌سازی پیش‌رو» هزینه‌بر است!

راه حل

نیازی به تطابق یک قاعده در تکرار k نداریم
اگر یک پیش‌فرض مقدم در تکرار $k-1$ اضافه نشده باشد.
$\Downarrow$
فقط قواعدی را تطبیق می‌دهیم که مقدم‌های آنها حاوی یک لیترال به‌تازگی اضافه شده باشد.

نمایه‌سازی پایگاه‌داده **(database indexing)** امکان بازیابی واقعیت‌های معلوم را در $O(1)$ فراهم می‌کند.

مثلاً: پرس‌وجوی $Missile(x)$ واقعیت $Missile(M_1)$ را بازیابی می‌کند.

تطابق **مقدم‌های عطفی** با **واقعیت‌های معلوم**، NP سخت **(NP-hard)** است.

زنجیره‌سازی پیشرو در **پایگاه‌های داده‌ی استنباطی (deductive databases)** به‌طور گسترده استفاده می‌شود.

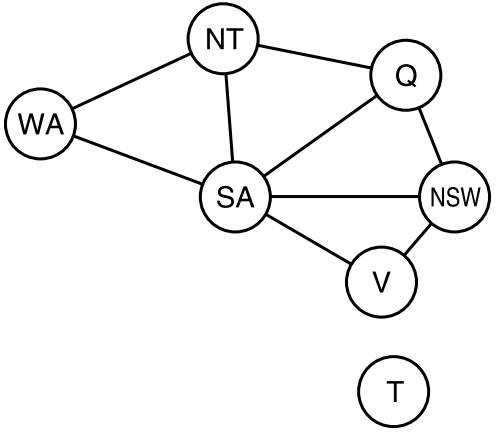
اثبات با «زنجیره‌سازی پیش‌رو»

کارآمدی «زنجیره‌سازی پیش‌رو»: مثال از تطابق سخت

HARD MATCHING EXAMPLE

تطابق **مقدم‌های عطفی** با **واقعیت‌های معلوم**، NP سخت **(NP-hard)** است.

هر CSP با دامنه‌ی متناهی، به صورت یک کلاوز معین واحد + تعدادی واقعیت زمینی مرتبط قابل بیان است.



$$Diff(wa, nt) \wedge Diff(wa, sa) \wedge$$
$$Diff(nt, q)Diff(nt, sa) \wedge$$
$$Diff(q, nsw) \wedge Diff(q, sa) \wedge$$
$$Diff(nsw, v) \wedge Diff(nsw, sa) \wedge$$
$$Diff(v, sa) \Rightarrow Colorable()$$
$$Diff(Red, Blue) \quad Diff(Red, Green)$$
$$Diff(Green, Red) \quad Diff(Green, Blue)$$
$$Diff(Blue, Red) \quad Diff(Blue, Green)$$

$Colorable()$ is inferred iff the CSP has a solution
CSPs include 3SAT as a special case, hence matching is NP-hard

هوش مصنوعی

استنتاج در منطق مرتبه اول

۴

زنجیره‌سازی پس‌رو

الگوریتم اثبات: «زنجیره‌سازی پس‌رو»

بر اساس قاعده‌ی استنتاج «قیاس استثنائی تعمیم‌یافته»

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).

زنجیره‌سازی پس‌رو

شبه‌کد الگوریتم

BACKWARD CHAINING ALGORITHM

function FOL-BC-ASK(KB, goals, θ) **returns** a set of substitutions
  inputs: KB, a knowledge base
    $goals$, a list of conjuncts forming a query
    θ, the current substitution, initially the empty substitution $\{\}$
  local variables: ans, a set of substitutions, initially empty

  if $goals$ is empty **then return** $\{\theta\}$
  $q' \leftarrow$ SUBST(θ, FIRST($goals$))
  for each r **in** KB where STANDARDIZE-APART(r) $= (p_1 \wedge \ldots \wedge p_n \Rightarrow q)$
    and $\theta' \leftarrow$ UNIFY(q, q') succeeds
    $ans \leftarrow$ FOL-BC-ASK(KB, $[p_1, \ldots, p_n | \text{REST}(goals)]$, COMPOSE($\theta', \theta$)) $\cup$ ans
  return ans

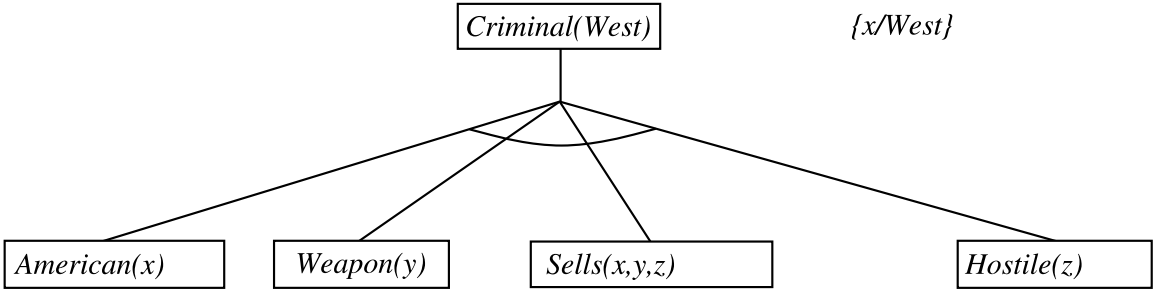
اثبات با «زنجیره‌سازی پس‌رو»

مثال (۱ از ۷)

$$Criminal(West)$$

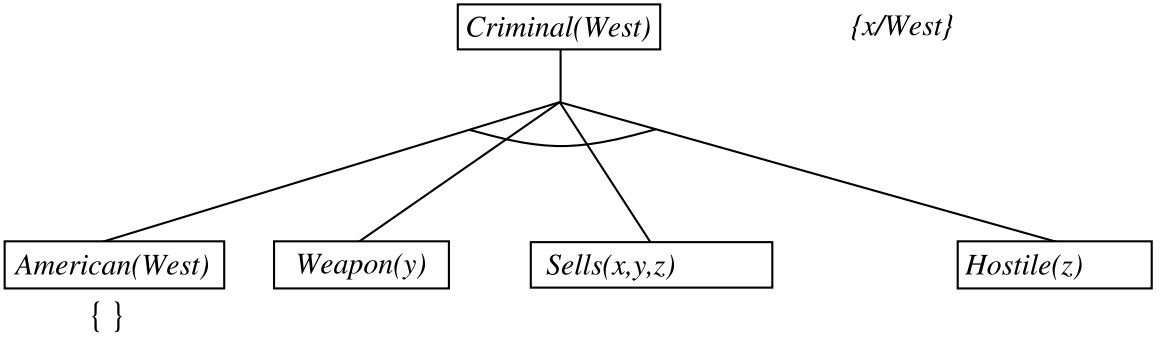
اثبات با «زنجیره‌سازی پس‌رو»

مثال (۲ از ۷)



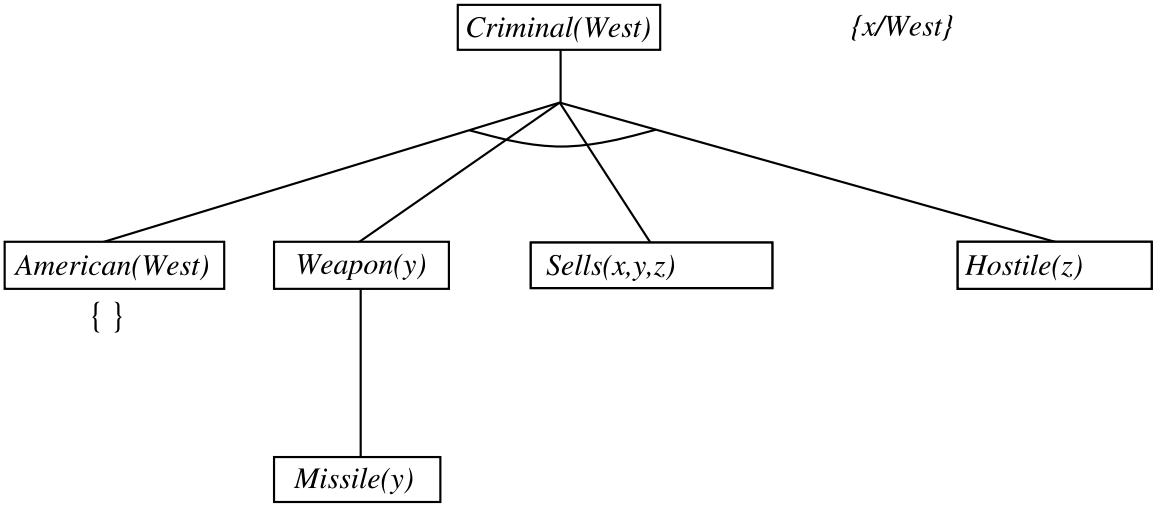
اثبات با «زنجیره‌سازی پس‌رو»

مثال (۳ از ۷)



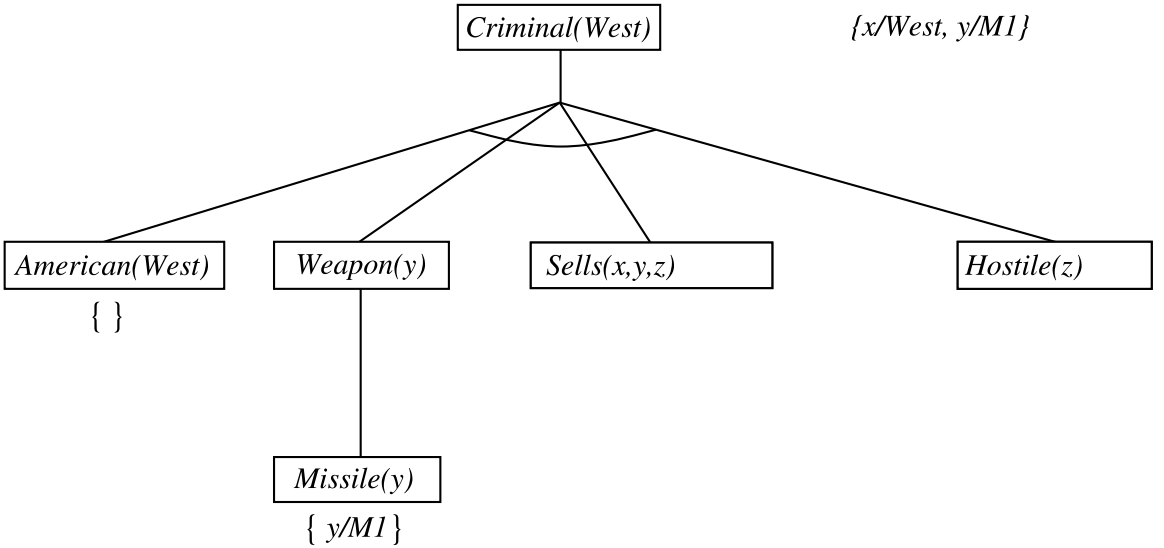
اثبات با «زنجیره‌سازی پس‌رو»

مثال (۴ از ۷)



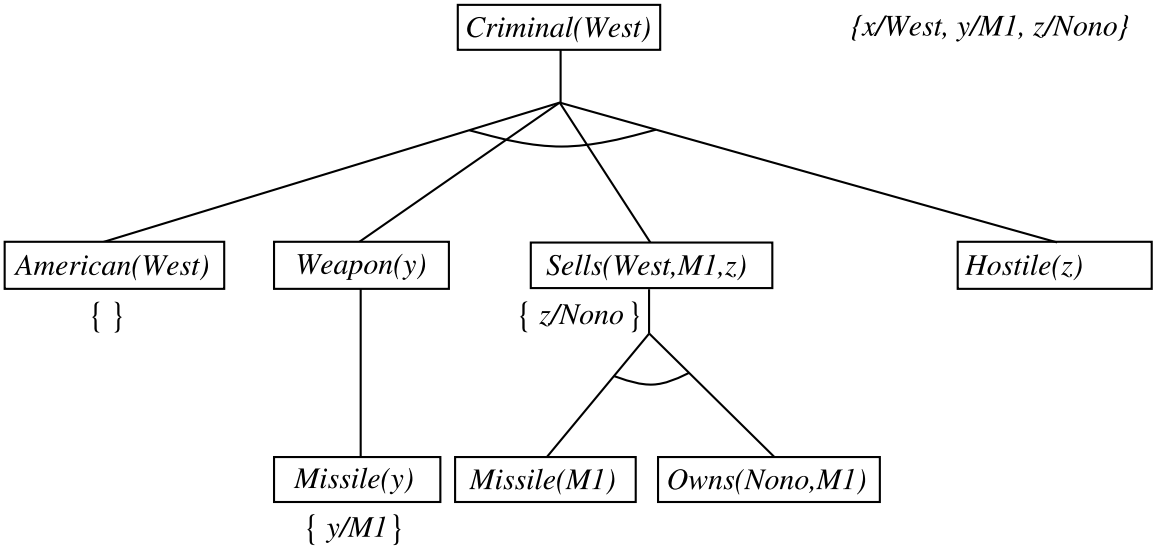
اثبات با «زنجیره‌سازی پس‌رو»

مثال (۵ از ۷)



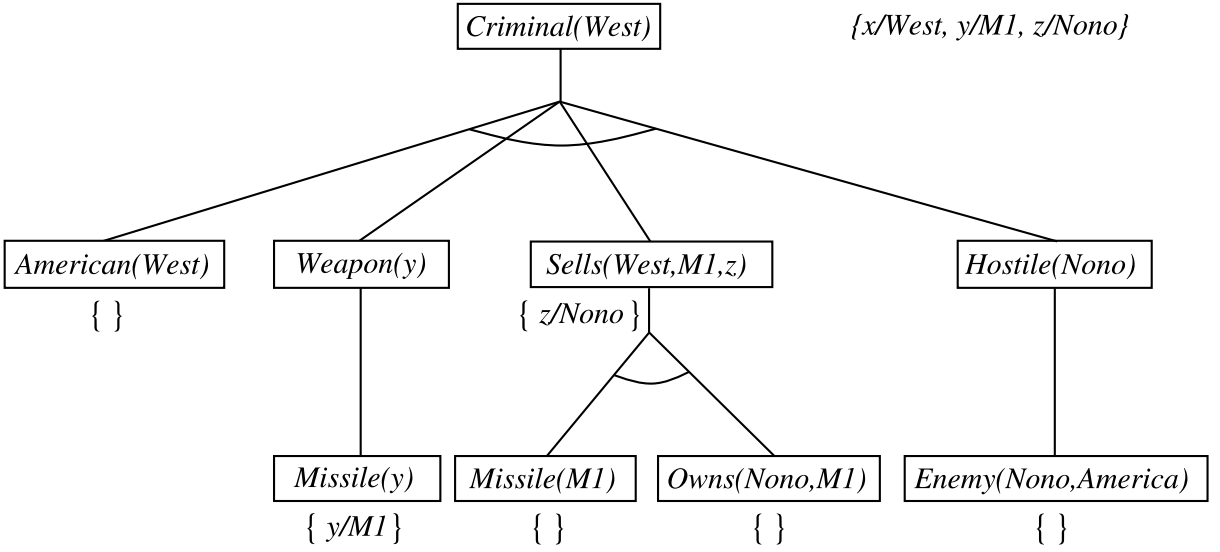
اثبات با «زنجیره‌سازی پس‌رو»

مثال (۶ از ۷)



اثبات با «زنجیره‌سازی پس‌رو»

مثال (۷ از ۷)



اثبات با «زنجیره‌سازی پس‌رو»

خصوصیات «زنجیره‌سازی پس‌رو»

PROPERTIES OF BACKWARD CHAINING

زنجیره‌سازی پس‌رو

جستجوی: اثبات بازگشتی عمق–اول
(فضای خطی بر حسب اندازه‌ی اثبات)

ملزومات روش زنجیره‌سازی پس‌رو

اجتناب از حلقه‌ها *Avoid loops*	اجتناب از کار تکراری *Avoid repeated work*
ناتمامیت در اثر حلقه‌های نامتناهی	**ناکارآمد بودن در اثر زیرهدف‌های تکراری** (موفقیت/شکست)
رفع مشکل با: بررسی هدف فعلی در برابر همه‌ی هدف‌های موجود بر روی پشته	رفع مشکل با: ذخیره‌سازی نتایج قبلی (نیازمند فضای اضافی!)

زنجیره‌سازی پس‌رو، در **برنامه‌نویسی منطقی (logic programming)** به‌طور گسترده (بدون بهبود!) استفاده می‌شود.

برنامه‌نویسی منطقی

LOGIC PROGRAMMING

برنامه‌نویسی منطقی: محاسبات به صورت استنتاج بر روی پایگاه‌های دانایی منطقی

	برنامه‌نویسی منطقی Logic programming	برنامه‌نویسی معمولی Ordinary programming
1.	Identify problem	Identify problem
2.	Assemble information	Assemble information
3.	Tea break	Figure out solution
4.	Encode information in KB	Program solution
5.	Encode problem instance as facts	Encode problem instance as data
6.	Ask queries	Apply program to data
7.	Find false facts	Debug procedural errors

Should be easier to debug $Capital(NewYork, US)$ than $x := x + 2$!

برنامه‌نویسی منطقی

سیستم پرولوگ

PROLOG SYSTEM

زنجیره‌سازی پس‌رو | فرم هورن + ...

برنامه = مجموعه‌ای از کلاوزها

$$\texttt{head :- literal}_1\texttt{, ... literal}_n\texttt{.}$$

```
criminal(X) :- american(X), weapon(Y), sells(X,Y,Z), hostile(Z).
```

ویژگی‌ها

- **زنجیره‌سازی پس‌رو، چپ به راست، عمق–اول**
- **یکسان‌سازی** کارآمد با تکنیک open coding
- بازیابی کارآمد کلاوزهای تطابق‌کننده با تکنیک direct linking
- محمول‌های درونی برای حساب مثل `X is Y*Z+3`
- **فرض دنیای بسته** (Closed-world assumption): **negation as failure**

e.g., given `alive(X) :- not dead(X).`
`alive(joe)` succeeds if `dead(joe)` fails

استفاده‌ی گسترده در اروپا و ژاپن (پایه‌ی پروژه‌ی نسل پنجم)

برنامه‌نویسی منطقی

سیستم پرولوگ: مثال

PROLOG SYSTEM

Depth-first search from a start state X: جستجوی عمق–اول از حالت شروع X

```
dfs(X) :- goal(X).
dfs(X) :- successor(X,S),dfs(S).
```

No need to loop over S: `successor` succeeds for each

Appending two lists to produce a third: الحاق دو لیست برای ایجاد لیست سوم

```
append([],Y,Y).
append([X|L],Y,[X|Z]) :- append(L,Y,Z).

query:   append(A,B,[1,2]) ?
answers: A=[]    B=[1,2]
         A=[1]   B=[2]
         A=[1,2] B=[]
```

هوش مصنوعی

استنتاج در منطق مرتبه اول

۵

رزولوشن

رزولوشن

RESOLUTION

Full first-order version:

$$\frac{\ell_1 \vee \cdots \vee \ell_k, \qquad m_1 \vee \cdots \vee m_n}{(\ell_1 \vee \cdots \vee \ell_{i-1} \vee \ell_{i+1} \vee \cdots \vee \ell_k \vee m_1 \vee \cdots \vee m_{j-1} \vee m_{j+1} \vee \cdots \vee m_n)\theta}$$

where $\text{UNIFY}(\ell_i, \neg m_j) = \theta$.

قابل استفاده با KBهای در **فرم نرمال عطفی** (CNF)

با فرض: همه‌ی متغیرها با سور عمومی

رزولوشن

مثال

RESOLUTION

Full first-order version:

$$\frac{\ell_1 \vee \cdots \vee \ell_k, \qquad m_1 \vee \cdots \vee m_n}{(\ell_1 \vee \cdots \vee \ell_{i-1} \vee \ell_{i+1} \vee \cdots \vee \ell_k \vee m_1 \vee \cdots \vee m_{j-1} \vee m_{j+1} \vee \cdots \vee m_n)\theta}$$

where $\text{UNIFY}(\ell_i, \neg m_j) = \theta$.

For example,

$$\frac{\neg Rich(x) \vee Unhappy(x) \qquad Rich(Ken)}{Unhappy(Ken)}$$

with $\theta = \{x/Ken\}$

استنتاج با «رزولوشن»

خصوصیات روال استنتاج

Full first-order version:

$$\frac{\ell_1 \vee \cdots \vee \ell_k, \qquad m_1 \vee \cdots \vee m_n}{(\ell_1 \vee \cdots \vee \ell_{i-1} \vee \ell_{i+1} \vee \cdots \vee \ell_k \vee m_1 \vee \cdots \vee m_{j-1} \vee m_{j+1} \vee \cdots \vee m_n)\theta}$$

where $\text{UNIFY}(\ell_i, \neg m_j) = \theta$.

برای روال استنتاج «رزولوشن» *"Resolution"*	
صحیح بودن *Soundness*	**تمامیت** *Completeness*
هر جمله‌ی مشتق شده درست باشد. برای منطق مرتبه اول: همیشه (CNF)	**همه‌ی جمله‌های درست مشتق شوند.** برای منطق مرتبه اول: همیشه (CNF) (نیمه‌تصمیم‌پذیر)

قابل استفاده در **روال اثبات رزولوشن**

الگوریتم اثبات: «رزولوشن»

بر اساس قاعده‌ی استنتاج «رزولوشن»

RESOLUTION

رزولوشن
Resolution

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

Apply resolution steps to $CNF(KB \wedge \neg\alpha)$; complete for FOL

فرم نرمال عطفی

تبدیل یک گزاره به CNF

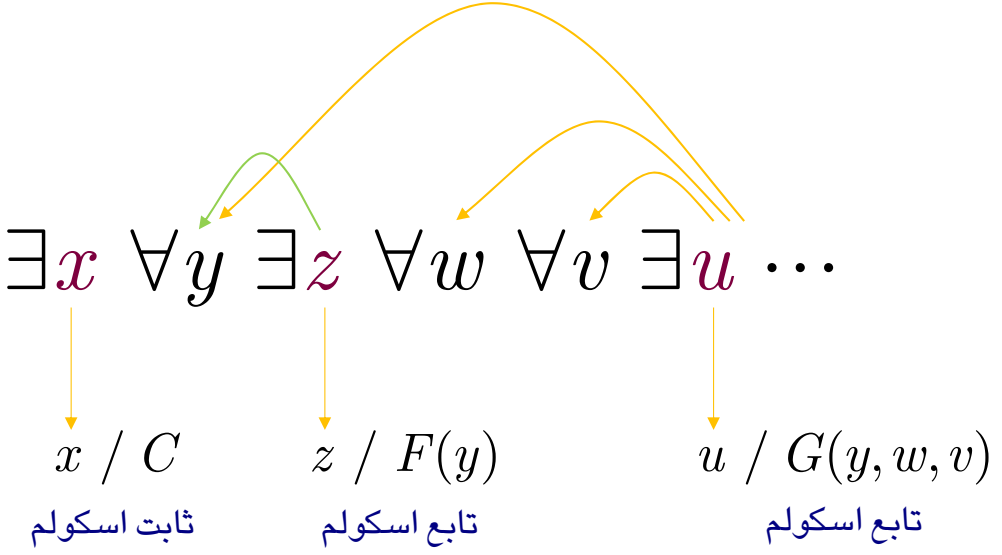
CONJUNCTIVE NORMAL FORM (CNF)

اسکولمیزاسیون

SKOLEMIZATION

اسکولمیزاسیون
Skolemization

هر متغیر وجودی، با یک تابع **اسکولم**
از متغیرهای مسور عمومی دربردارنده جایگزین می‌شود.
(شکل کلی‌تری از نمونه‌سازی وجودی)



فرم نرمال عطفی

تبدیل یک گزاره به CNF : مثال (۱ از ۲)

CONJUNCTIVE NORMAL FORM (CNF)

هر کسی که همه‌ی حیوانات را دوست دارد، توسط کسی دوست داشته می‌شود.

Everyone who loves all animals is loved by someone:

$$\forall x \ [\forall y \ \ Animal(y) \Rightarrow Loves(x,y)] \Rightarrow [\exists y \ \ Loves(y,x)]$$

۱) حذف دوشرطی‌ها و شرطی‌ها (ایجاب‌ها)

1. Eliminate biconditionals and implications

$$\forall x \ [\neg\forall y \ \ \neg Animal(y) \vee Loves(x,y)] \vee [\exists y \ \ Loves(y,x)]$$

۲) حرکت نقیض‌ها به داخل: {با قواعد دمورگان}

2. Move $\neg$ inwards: $\neg\forall x,p \equiv \exists x \ \neg p$, $\neg\exists x,p \equiv \forall x \ \neg p$:

$$\forall x \ [\exists y \ \neg(\neg Animal(y) \vee Loves(x,y))] \vee [\exists y \ \ Loves(y,x)]$$
$$\forall x \ [\exists y \ \neg\neg Animal(y) \wedge \neg Loves(x,y)] \vee [\exists y \ \ Loves(y,x)]$$
$$\forall x \ [\exists y \ \ Animal(y) \wedge \neg Loves(x,y)] \vee [\exists y \ \ Loves(y,x)]$$

فرم نرمال عطفی

تبدیل یک گزاره به CNF : مثال (۲ از ۲)

CONJUNCTIVE NORMAL FORM (CNF)

3. Standardize variables: each quantifier should use a different one ۳) استانداردسازی متغیرها

$$\forall x \ [\exists y \ Animal(y) \wedge \neg Loves(x, y)] \vee [\exists z \ Loves(z, x)]$$

4. Skolemize: a more general form of existential instantiation. ۴) اسکولمیزاسیون
   Each existential variable is replaced by a Skolem function of the enclosing universally quantified variables:

$$\forall x \ [Animal(F(x)) \wedge \neg Loves(x, F(x))] \vee Loves(G(x), x)$$

5. Drop universal quantifiers: ۵) حذف سورهای عمومی

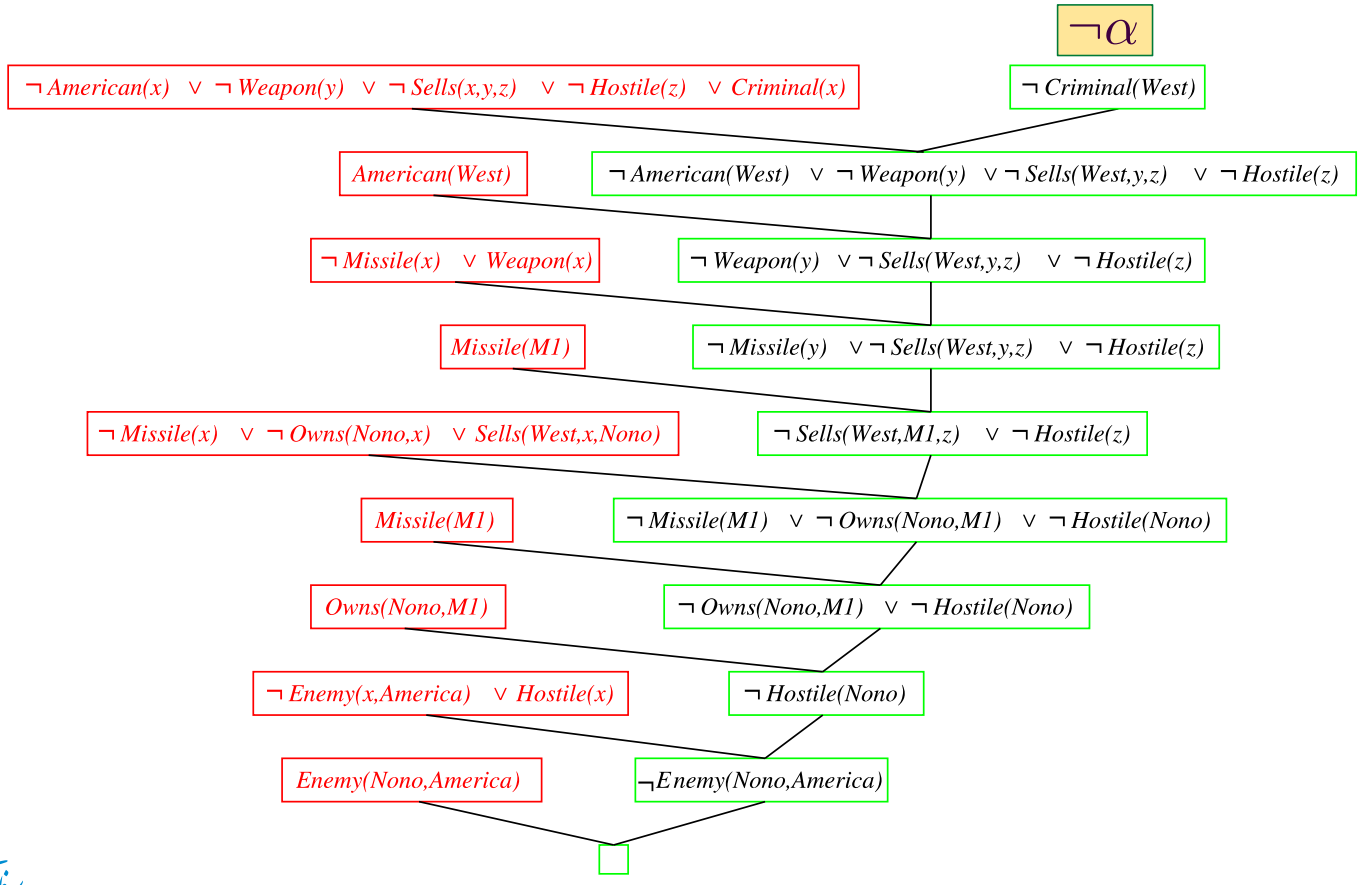
$$[Animal(F(x)) \wedge \neg Loves(x, F(x))] \vee Loves(G(x), x)$$

6. Distribute $\wedge$ over $\vee$: ۶) اعمال قانون توزیع‌پذیری (**و** روی **یا**)

$$[Animal(F(x)) \vee Loves(G(x), x)] \wedge [\neg Loves(x, F(x)) \vee Loves(G(x), x)]$$

الگوریتم اثبات: «رزولوشن»

مثال



مثال

Everyone who loves all animals is loved by someone.
Anyone who kills an animal is loved by no one.
Jack loves all animals.
Either Jack or Curiosity killed the cat, who is named Tuna.

Did Curiosity kill the cat?

هر کسی که همه‌ی حیوانات را دوست دارد، توسط کسی دوست داشته می‌شود.
هر کسی که یک حیوان را بکشد، توسط هیچ کس دوست داشته نمی‌شود.
جک همه‌ی حیوانات را دوست دارد.
یا جک یا کوریوسیتی گربه‌ای به نام تونا را کشته است.

آیا کوریوسیتی گربه را کشته است؟

پایگاه دانایی با «منطق مرتبه اول»

مثال: تبدیل جملات به منطق مرتبه اول

A. $\forall x \ [\forall y \ \ Animal(y) \Rightarrow Loves(x, y)] \Rightarrow [\exists y \ \ Loves(y, x)]$

B. $\forall x \ [\exists z \ \ Animal(z) \wedge Kills(x, z)] \Rightarrow [\forall y \ \neg Loves(y, x)]$

C. $\forall x \ \ Animal(x) \Rightarrow Loves(Jack, x)$

D. $Kills(Jack, Tuna) \vee Kills(Curiosity, Tuna)$

E. $Cat(Tuna)$

F. $\forall x \ \ Cat(x) \Rightarrow Animal(x)$

$\neg\alpha$ ¬G. $\neg Kills(Curiosity, Tuna)$

پایگاه دانایی با «منطق مرتبه اول»

مثال: تبدیل جملات به فرم نرمال عطفی

A. $\forall x \ [\forall y \ \ Animal(y) \Rightarrow Loves(x, y)] \Rightarrow [\exists y \ \ Loves(y, x)]$

B. $\forall x \ [\exists z \ \ Animal(z) \wedge Kills(x, z)] \Rightarrow [\forall y \ \neg Loves(y, x)]$

C. $\forall x \ \ Animal(x) \Rightarrow Loves(Jack, x)$

D. $Kills(Jack, Tuna) \vee Kills(Curiosity, Tuna)$

E. $Cat(Tuna)$

F. $\forall x \ \ Cat(x) \Rightarrow Animal(x)$

$\neg\alpha$ ¬G. $\neg Kills(Curiosity, Tuna)$

Now we apply the conversion procedure to convert each sentence to CNF:

A1. $Animal(F(x)) \vee Loves(G(x), x)$

A2. $\neg Loves(x, F(x)) \vee Loves(G(x), x)$

B. $\neg Loves(y, x) \vee \neg Animal(z) \vee \neg Kills(x, z)$

C. $\neg Animal(x) \vee Loves(Jack, x)$

D. $Kills(Jack, Tuna) \vee Kills(Curiosity, Tuna)$

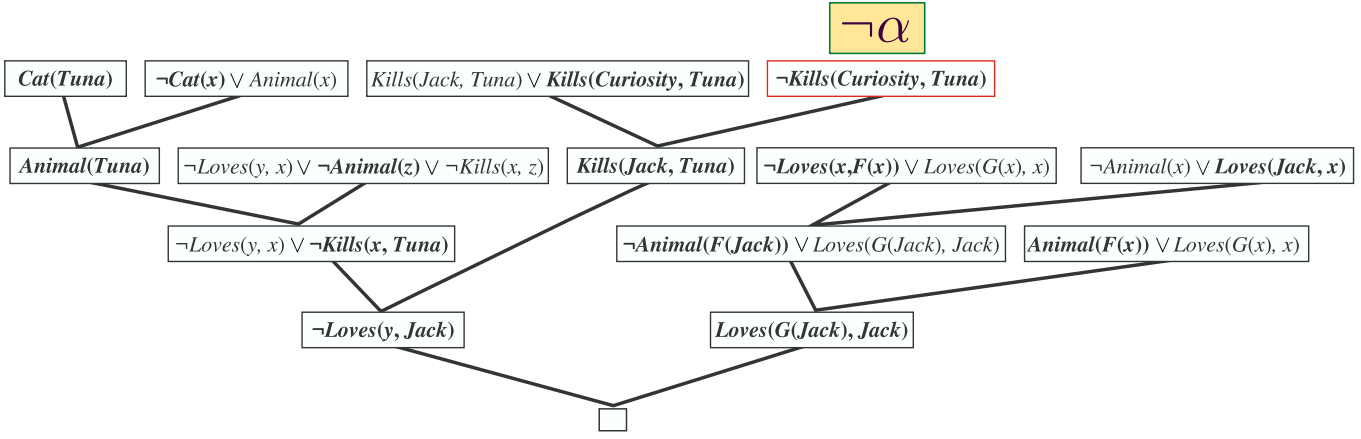
E. $Cat(Tuna)$

F. $\neg Cat(x) \vee Animal(x)$

$\neg\alpha$ ¬G. $\neg Kills(Curiosity, Tuna)$

الگوریتم اثبات: «رزولوشن»

مثال



A resolution proof that **Curiosity killed the cat**.

- Notice the use of factoring in the derivation of the clause $Loves(G(Jack), Jack)$.
- Notice also in the upper right, the unification of $Loves(x, F(x))$ and $Loves(Jack, x)$ can only succeed after the variables have been standardized apart.

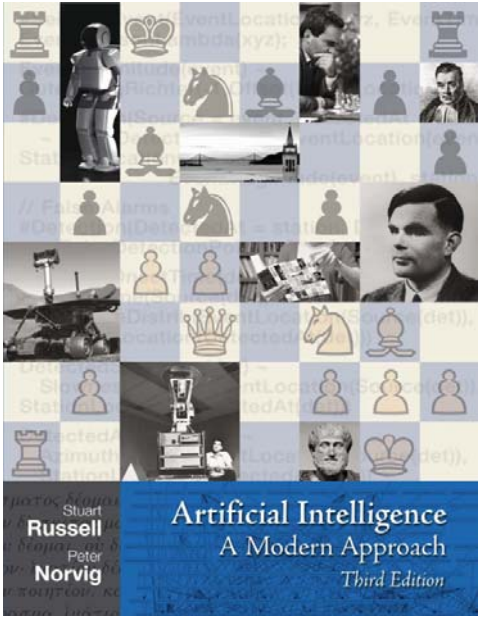
هوش مصنوعی

استنتاج در منطق مرتبه اول

۶

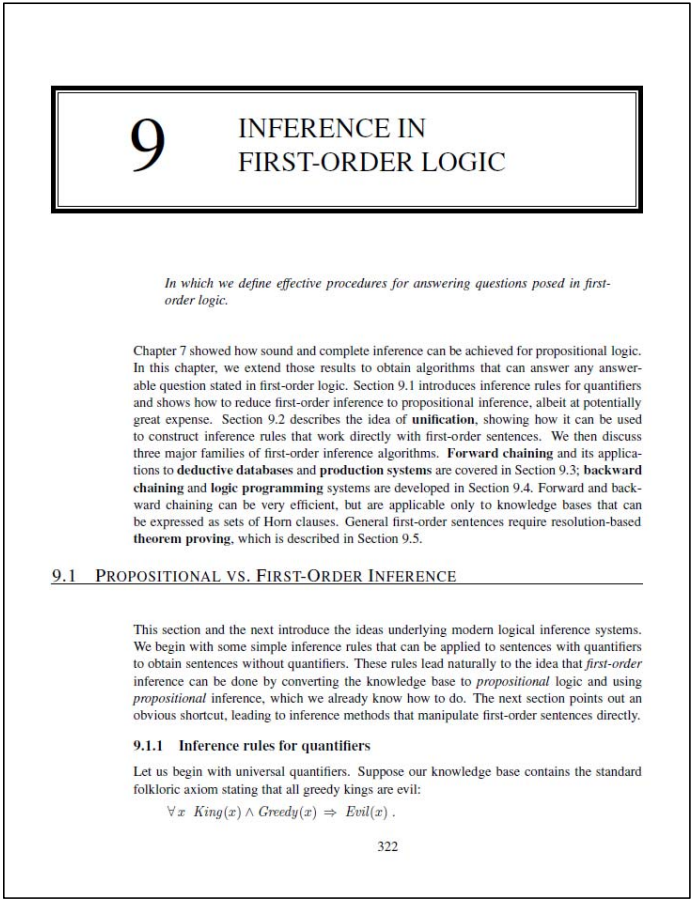
منابع، مطالعه، تکلیف

منبع اصلی



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

Chapter 9



9 INFERENCE IN FIRST-ORDER LOGIC

In which we define effective procedures for answering questions posed in first-order logic.

Chapter 7 showed how sound and complete inference can be achieved for propositional logic. In this chapter, we extend those results to obtain algorithms that can answer any answerable question stated in first-order logic. Section 9.1 introduces inference rules for quantifiers and shows how to reduce first-order inference to propositional inference, albeit at potentially great expense. Section 9.2 describes the idea of **unification**, showing how it can be used to construct inference rules that work directly with first-order sentences. We then discuss three major families of first-order inference algorithms. **Forward chaining** and its applications to **deductive databases** and **production systems** are covered in Section 9.3; **backward chaining** and **logic programming** systems are developed in Section 9.4. Forward and backward chaining can be very efficient, but are applicable only to knowledge bases that can be expressed as sets of Horn clauses. General first-order sentences require resolution-based **theorem proving**, which is described in Section 9.5.

9.1 PROPOSITIONAL VS. FIRST-ORDER INFERENCE

This section and the next introduce the ideas underlying modern logical inference systems. We begin with some simple inference rules that can be applied to sentences with quantifiers to obtain sentences without quantifiers. These rules lead naturally to the idea that *first-order* inference can be done by converting the knowledge base to *propositional* logic and using *propositional* inference, which we already know how to do. The next section points out an obvious shortcut, leading to inference methods that manipulate first-order sentences directly.

9.1.1 Inference rules for quantifiers

Let us begin with universal quantifiers. Suppose our knowledge base contains the standard folkloric axiom stating that all greedy kings are evil:

$$\forall x \ \ King(x) \wedge Greedy(x) \Rightarrow Evil(x) .$$

322

تکلیف ۸

هوالحکیم

دانشگاه تهران
پردیس فارابی
دانشکده‌ی مهندسی

هوش مصنوعی
نیمسال اول ۹۵-۱۳۹۴
http://courses.fouladi.ir/ai

تکلیف شماره‌ی ۸

فصل هشتم و نهم

منطق مرتبه اول و استنتاج در آن

۱) جمله‌های زیر که در منطق مرتبه‌ی اول نوشته شده است را به زبان طبیعی ترجمه کنید.

(الف) $\forall x(Bird(x) \Rightarrow Flies(x))$

(ب) $\forall x \exists y(Person(x) \Rightarrow Mother(y, x))$

(ج) $\exists x \forall y(Person(x) \wedge Mother(x, y))$

در جمله‌های فوق، $Bird(x)$ یعنی x پرنده است؛ $Flies(x)$ یعنی x پرواز می‌کند؛ $Person(x)$ یعنی x یک شخص است؛ $Mother(x, y)$ یعنی x مادر y است.

۲) جملات انگلیسی زیر را به عباراتی در منطق مرتبه‌ی اول تبدیل کنید. از نام‌های بامعنا برای محمول‌ها استفاده کنید.

(الف) All cats are mammals.

(ب) No cat is a reptile.

(ج) All computer scientists like some operating system.

۳) سه جمله‌ی زیر را در نظر بگیرید.

(A) There is a computer scientist who likes every operating system.

(B) Linux is an operating system.

(C) Someone likes Linux.

می‌خواهیم رابطه‌ی منطقی میان این سه جمله را پیدا کنیم.

(الف) فرمولی را در منطق مرتبه‌ی اول بنویسید که هر یک از واقعیت‌های داده‌شده را بیان کند و آنها را A، B و C بنامید.

(ب) مجموعه‌ای از clauseها را بنویسید که با A، B و $\neg C$ مرتبط باشند.

(ج) با استفاده از resolution از این مجموعه یک clause تهی (معادل False) استخراج کنید.

(د) توضیح دهید که resolution چه رابطه‌ی استنتاجی بین این سه جمله استخراج می‌کند؟

۴) عمومی‌ترین یکسان‌ساز (most general unifier) را برای هر یک از زوج عبارت‌های زیر بیابید.

(الف) $Rel(z, C, P(x, F(x)), x)$, $Rel(P(y, y), y, z, D)$

(ب) $P(A, B, B)$, $P(x, y, z)$

(ج) $Older(Father(y), y)$, $Older(Father(x), John)$

(د) $Q(y, G(A, B))$, $Q(G(x, x), y)$

(ه) $Knows(Father(y), y)$, $Knows(x, x)$

۵) نقیض عبارت زیر را بیابید.

$$\forall x \forall y \forall z[P(x) \wedge Q(x, y) \Rightarrow R(x, y, z)]$$

۶) چگونه با استفاده از resolution می‌توان «ارضاناپذیری» و «معتبر بودن» یک گزاره را ثابت کرد؟

کار مطالعاتی ۸

هوالحکیم

دانشگاه تهران
پردیس فارابی
دانشکده‌ی مهندسی

هوش مصنوعی
نیمسال اول ۹۵-۱۳۹۴
http://courses.fouladi.ir/ai

کار مطالعاتی شماره‌ی ۷

فصل هفتم، هشتم و نهم

عامل‌های منطقی، منطق مرتبه اول و استنتاج در آن

	عنوان موضوع	دانشجویان
۱	پایگاه‌های داده‌ی استنباطی Deductive Databases	۱) ..... ۲) .....
۲	قضیه‌ی ناتمامیت گودل Gödel Incompleteness Theorem	۱) ..... ۲) .....
۳	الگوریتم رته Rete Algorithm	۱) ..... ۲) .....
۴	برنامه‌نویسی منطقی جدولی Tabled Logic Programming	۱) ..... ۲) .....
۵	منطق محاسباتی Computational Logic	۱) ..... ۲) .....
۶	حدس مقدار آستانه‌ی ارضاپذیری Satisfiability Threshold Conjecture	۱) ..... ۲) .....
۷	حساب وضعیت Situation Calculus	۱) ..... ۲) .....
۸	منطق زمانی Temporal Logic	۱) ..... ۲) .....
۹	منطق فازی Fuzzy Logic	۱) ..... ۲) .....
۱۰	فرضیه سپیر-هورف Sapir-Whorf Hypothesis	۱) ..... ۲) .....
۱۱	منطق مرتبه بالاتر Higher-Order Logic (HOL)	۱) ..... ۲) .....
۱۲	برنامه‌نویسی اعلانی Declarative Programming	۱) ..... ۲) .....
۱۳	جبر روبینز Robbins Algebra	۱) ..... ۲) .....
۱۴	اینفوگرافی عامل‌های منطقی و منطق ریاضی Logical Agents & Mathematical Logic: Infographies	۱) ..... ۲) .....

- گزارش کار مطالعاتی در حدود دو صفحه تنظیم و در محل مشخص شده در وب‌سایت درس بارگذاری شود.
- همه‌ی فایل‌های پیوست به صورت آرشیو شده یا فشرده شده در قالب یک فایل بارگذاری شود و در صورت زیاد بودن حجم آن، برای استاد درس ایمیل شود یا بر روی CD به طور حضوری تحویل داده شود.
- ارائه‌ی منابع مطالعه به عنوان پیوست و گردآوری تصویر، انیمیشن، فیلم، متن و هر مطلب خام از هر منبعی اعم از کتابخانه، اینترنت، آرشیو و ... که مرتبط با موضوع باشد، اکیداً پیشنهاد می‌شود و بدیهی است که در نمره‌ی تخصیص داده شده به کار مطالعاتی تأثیر مثبت دارد.