هوش مصنوعی

فصل نهم

استنتاج در منطق مرتبه اول

ایمان مختاری فرد
دانشگاه پیام نور شهرکرد

استنتاج مرتبه‌ی اول

- بر حسب مطالب فصل ۷ ، برای استنتاج در منطق مرتبه اول ، ابتدا باید عبارت منتطق مرتبه اول را به گزاره ای تبدیل کنیم و سپس عبارت گزاره ای را با تابع Pl_resolution استنتاج کنیم
- در روند گزاره ای کردن عبارت FOL باید سورها و.... حذف شوند چون عبارت گزاره ای سور ندارد

مراحل تبدیل مرتبه اول به گزاره ای (مهم)

$\forall x\ (P(x) \rightarrow \exists y\ (\forall z\ (\ q(y,z))))$

۱. حذف دو شرطی و شرطی
۲. اعمال نقیض (طبق قوانین دمورگان)

$\forall x\ (\exists y\ (\forall z\ (P(x) \rightarrow q(y,z))))$

۳. انتقال سورها به اول عبارت
۴. حذف سور وجودی و جایگزین کردن اسکلم

۱. اگر متغیر سور وجودی به صورت $\exists x\ P(x)$ باشد متغیر اسکلم a جایگزین و سور حذف $P(a)$
۲. اگر سور وجودر در یک سور عمومی باشد از تابع اسکلم برای حذف استفاده می کنیک

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

$\forall x\ (q(f(x),z))$

$\forall x\ (\forall z\ (P(x) \rightarrow q(f(x),z)))$

۵. حذف ضمنی سور عمومی(بدون جایگزین کردن)

$(P(x) \rightarrow q(f(x),z))$

۶. تبدیل به شکل نرمال عطفی

استنتاج مرتبه‌ی اول

- **هدف از استنتاج در منطق مرتبه اول ، یافتن الگوریتمی است که بتواند به هر سوالی قابل پاسخ گویی با منطق مرتبه اول ، پاسخ دهد و تعیین کند که آیا استلزام و ایجاب به وجود می آید یا خیر؟**

حذف سورها

- **روش اول برای استنتاج در منطق مرتبه اول حذف سور و تبدیل پایگاه دانش به منطق گزاره ای و استفاده از قوانین استنتاج در منطق گزاره ای است.لذا روشهای زیر به عنوان الگوریتمهای استنتاج در منطق مرتبه اول ارائه شده اند :**

۱. **نمونه‌سازی عمومی (حذف سور عمومی یا** Universal instantiation (UI))
۲. **نمونه‌سازی وجودی (حذف سور وجودی یا** Existential instantiation (EI))

۱- نمونه‌سازی عمومی

هر نمونه از یک جمله دارای سور عمومی، توسط آن جمله قابل استلزام است.
نمونه‌سازی عمومی برای هر متغیّر v و اصطلاح زمینه g، به‌صورت زیر نوشته می‌شود:

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

که در آن $Subst(\theta,\alpha)$ نتیجه اعمال جایگزینی θ به جای α است.

به عنوان مثال جملات

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

از نمونه‌سازی عمومی جمله

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

با جایگزینی های $\{x/John\}$، $\{x/Richard\}$ و $\{x/Father(John)\}$ به‌دست‌می‌آید.

۲- نمونه‌سازی وجودی

- در مورد $\exists x : LivesIn(x, Springfield) \wedge knows(x, Homer)$ ما می دانیم که این باید برای لااقل یک شی ، درست باشد . بیایید این شی را K بنامیم ؛ داریم ، $LivesIn(k, Springfield) \wedge knows(k, Homer)$ که در این مورد، K یک ثابت اسکُولِم نامیده می شود.
- پس ، برای هر عبارت α و متغیر v و سمبل ثابت k که در جای دیگری در پایگاه دانش وجود ندارد داریم :

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

مثال : نتیجه ی $\exists x Crown(x) \wedge OnHead(x, John)$ به صورت $Crown(C1) \wedge OnHead(C1,John)$ می تواند باشد که C1 به وجود آمده از یک سمبل ثابت جدید به نام ثابت اسکلم می باشد.

مثالی دیگر : از $d(x^y) / dy = x^y$ ما نتیجه می گیریم ، $\exists x : d(x^y) / dy = e^y$ ، e ای که به وجود آمده است یک سمبل ثابت جدید می باشد)

گزاره بندی

ما می توانیم هر عبارت دارای سور وجودی را با یک نوع اسکلمایز شده جایگزین نماییم . برای عبارت های با سور عمومی ، می توانید هر جایگزین ممکن را جایگزین نمایید ؛ که این کار به ما اجازه می دهد که از قانون های استنتاج گزاره ای استفاده نماییم . البتّه این کار خیلی ناکارآمد می باشد ؛ تا سال ۱۹۶۰ از همین روش استفاده می کردند .

ساده سازی به استنتاج گزاره ای

- پایگاه دانش مقابل را در نظر بگیرید:

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

با اعمال تمامی جایگزینی های ممکن به اوّلین جمله، داریم: (پایگاه دانش گزاره‌ای شده)(UI) ({x/John} و {x/Richard})

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

KB جدید گزاره ای سازی شده است. سیمبول های(نمادها) گزاره ای عبارتند از:
King(John) , Greedy(John) , Evil(John) , King(Richard)

هر پایگاه دانش با منطق مرتبه اول را می توان گزاره ای کرد به گونه‌ای که ایجاب در آن حفظ شود، یعنی هر جمله زمینه‌ای که از پایگاه دانش جدید به دست می آید، اگر و فقط اگر از پایگاه دانش جدید نیز حاصل شود.
یک روش برای استنتاج: پایگاه دانش و پرس‌وجوی مرتبه‌ی اوّل را گزاره‌ای نمود، و از الگوریتمهای استنتاج گزاره ای برای استنتاج استفاده کرد.

- مشکل: در مورد سیمبول های تابعی(دارای خروجی) تعداد نامحدودی ترم زمینی وجود دارد، مثلاً:

Father(Father(Father(John)))

قضیه هربراند (۱۹۳۰) : اثبات با یک زیر مجموعه...

- **قضیه هربراند** (۱۹۳۰): اگر جمله‌ای به وسیله پایگاه دانش مرتبه‌ی اوّل اصلی ایجاب شود، آنگاه یک اثبات با زیرمجموعه‌ی متناهی از پایگاه دانش گزاره‌ای شده، وجود دارد.

- با توجه به متناهی بودن زیرمجموعه، یک مقدار حداکثر برای عمق تودرتو بودن اصطلاحات زمینه موجود می‌باشد.

تئوری Herbrand (۱۹۳۰): اگر جمله α توسط پایگاه دانش FOL قابل استلزام باشد، آنگاه این جمله توسط زیرمجموعه محدودی از پایگاه دانش گزاره ای سازی شده قابل استلزام است

ایده:

For $n = 0$ to ∞ do
  create a propositional KB by instantiating with depth-n terms
  see if α is entailed by this KB

مشکل: اگر KB مستلزم α باشد کار می کند، در غیر اینصورت در حلقه بی نهایت می افتد

می‌توان ابتدا زیرمجموعه‌ای با استفاده از تمامی نمونه‌سازیهای نمادهای ثابت (*Richard* و *John*) را تولید کنیم. سپس، با تمامی اصطلاحات با عمق ۱ (*Father (Richard)* و *Father (john)*)، و بعد از آن تمامی اصطلاحات با عمق ۲ و ... را بیابیم تا زمانی که قادر به اثبات گزاره‌ای جمله ایجاب شده بشویم.

قضیه ی تورینگ و چورچ (۱۹۳۶) : فقط بلی می تواند بگوید

قضیه تورینگ و چرچ (۱۹۳۶): *مسأله ایجاب در منطق مرتبه‌ی اوّل، یک مسأله‌ی* ***نیمه‌تصمیم‌پذیر*** *است، یعنی الگوریتمهایی وجود دارد که به هر جمله ایجاب شونده پاسخ مثبت می‌دهد. امّا هیچ الگوریتمی وجود ندارد که به هر جمله ایجاب‌ناپذیر پاسخ منفی بدهد.*

مشکلات گزاره ای سازی نمودن

• به نظر می رسد که گزاره ای سازی کردن جملات نامربوط زیادی تولید می کند. برای مثال از جملات زیر:

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

حاصل Evil(John) است ، اما گزاره ای سازی کردن حقایق زیادی مانند Greedy(Richard) تولید می کند که نامربوط می باشند

یکسان سازی (Unification) : حل مشکل گزاره سازی:سازگار بودن دو جمله

• اگر یک جایگزینی θ وجود داشته باشد که مقدم استلزام را با جملات موجود در پایگاه دانش، یکسان‌سازی کند، یا به عبارتی یکسان سازی، این است که ما فقط می خواهیم جایگزین هایی برای عباراتی که به ما کمک می کنند چیزهایی را ثابت نماییم پیدا کنیم .آنگاه می‌توان با اعمال θ، تالی استلزام را اظهار نمود. مانند جایگزینی {x / John} در پایگاه دانش، Evil (John) را نتیجه می‌دهد.

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

ابتدا باید بتوان روشی برای انتخاب مناسب x پیدا کرد (یکسان سازی).

$Unify(\alpha,\beta) = \theta$ if $\alpha\theta = \beta\theta$

Unification: دو ترم s و t اصطلاحاً unifiableهستند اگر یک جانشینی σمانند
وجود داشته باشد به طوری که . $\sigma(s) = \sigma(t)$

p	θ	q
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} عدم استاندارد به خاطر ترکیب x و y

قیاس استثنائی تعمیم یافته

- آیا می‌توان در پایگاه دانش با انتخاب مقداری برای x ، یک دفعه یک واقعیت را استنتاج کرد؟

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

برای این کار باید قانون

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

p_1' is *King*(*John*) p_1 is *King*(*x*)
p_2' is *Greedy*(*y*) p_2 is *Greedy*(*x*)
θ is {x/John,y/John} q is *Evil*(*x*)
q θ is *Evil*(*John*)

در این قانون اگر جمله هایی به صورت ^ با هم ترکیب شوند و جمله ی دیگری را ایجاد کنند. در را الزام می کرد با وجود چند جمله به تعداد جملات ترکیبی می توان جایگزاری مقادیرθ در q را طبق شرط یکسان سازی نتیجه گرفت که به آن قیاس استثنایی توسعه یافته می گویند

ذخیره و بازیابی

- **توابع ابتدایی Askو Tellو یا استفاده از نسخه ابتدایی تر مثل Storeو Fetch**
- **پرس و جوها ((ASK) از طریق یک شبکه شمول به دست می آیند. در این شبکه هر گره در شبکه تنها با یک جابجایی از والد خود بدست می آید(شکل صفحه ۱۳۰)**

ذخیره و بازیابی

- قانون می گوید که برای یک آمریکایی، فروش تسلیحات به ملل متخاصم جرم است.کشور نونو که دشمن آمریکا است، تعدادی موشک دارد و تمامی موشکهایش را از سرهنگ وست که یک آمریکایی است، خریده است.

- ثابت کنید سرهنگ وست مجرم است.

مثالی از یک پایگاه دانش

- فروش تسلیحات به ملل متخاصم برای یک آمریکایی جرم است
- $American(x) \wedge Weapon(y) \wedge Sells(x,y,z) \wedge Hostile(z) \Rightarrow Criminal(x)$
- نونو تعدادی موشک دارد:$\exists x\ Owns(Nono,x) \wedge Missile(x)$.,
- $Owns(Nono,M_1)\ and\ Missile(M_1)$
- سرهنگ وست تمامی موشک ها را را به نونو فروخته بود
- $Missile(x) \wedge Owns(Nono,x) \Rightarrow Sells(West,x,Nono)$
- موشک ها جزو تسلیحات هستند
- $Missile(x) \Rightarrow Weapon(x)$
- هر دشمن آمریکا به عنوان متخاصم تلقی می شود
- $Enemy(x,America) \Rightarrow Hostile(x)$
- وست یک آمریکایی است
- $American(West)$
- کشور نونو دشمن آمریکاست
- $Enemy(Nono,America)$

مثالی از یک پایگاه دانش

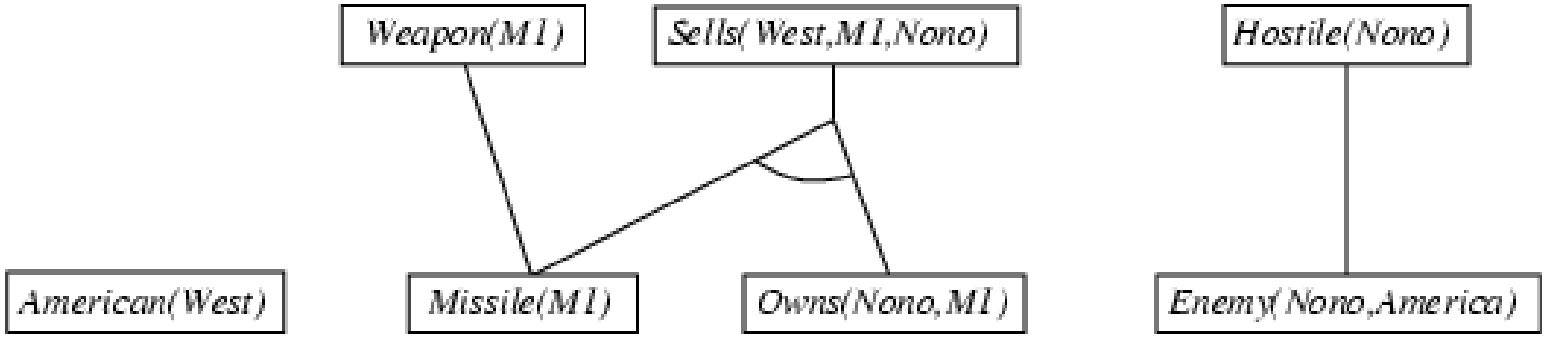
- فروش تسلیحات به ملل متخاصم برای یک آمریکایی جرم است
- $American(x) \wedge Weapon(y) \wedge Sells(x,y,z) \wedge Hostile(z) \Rightarrow Criminal(x)$
- نونو تعدادی موشک دارد:$\exists x\ Owns(Nono,x) \wedge Missile(x)$.,
- $Owns(Nono,M_1)\ and\ Missile(M_1)$
- سرهنگ وست تمامی موشک ها را را به نونو فروخته بود
- $Missile(x) \wedge Owns(Nono,x) \Rightarrow Sells(West,x,Nono)$
- موشک ها جزو تسلیحات هستند
- $Missile(x) \Rightarrow Weapon(x)$
- هر دشمن آمریکا به عنوان متخاصم تلقی می شود
- $Enemy(x,America) \Rightarrow Hostile(x)$
- وست یک آمریکایی است
- $American(West)$
- کشور نونو دشمن آمریکاست
- $Enemy(Nono,America)$

الگوریتم زنجیره ای پیش رو
از جملات بسیط حرکت می کند و مبتنی بر داده هاست

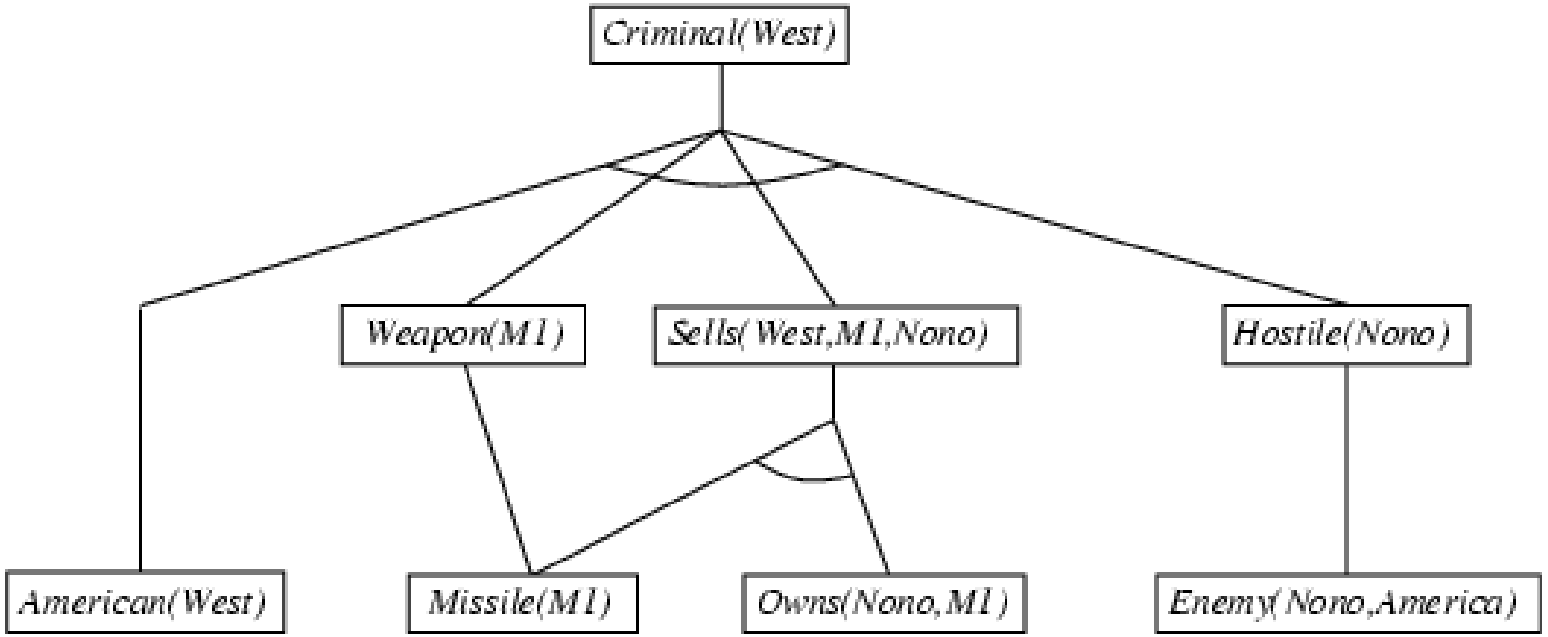
```
function FOL-FC-Ask(KB, α) returns a substitution or false
  repeat until new is empty
    new ← { }
    for each sentence r in KB do
      (p1 ∧ ... ∧ pn ⇒ q) ← Standardize-Apart(r)
      for each θ such that (p1 ∧ ... ∧ pn)θ = (p'1 ∧ ... ∧ p'n)θ
                  for some p'1, ..., p'n in KB
        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
```

الگوریتم زنجیره ای پیش رو

الگوریتم زنجیره ای پیش رو



الگوریتم زنجیره ای پیش رو



ویژگی های الگوریتم زنجیره ای پیش رو

- این الگوریتم صحیح است به دلیل آنکه هر استنتاج از به کارگیری قیاس استثنائی تعمیم یافته ناشی می شود که آن نیز به نوبه خود صحیح می باشد.

- این الگوریتم در مورد پایگاه های دانش با **بندهای معین کامل** است.

الگوریتم زنجیره ای **پس رو**

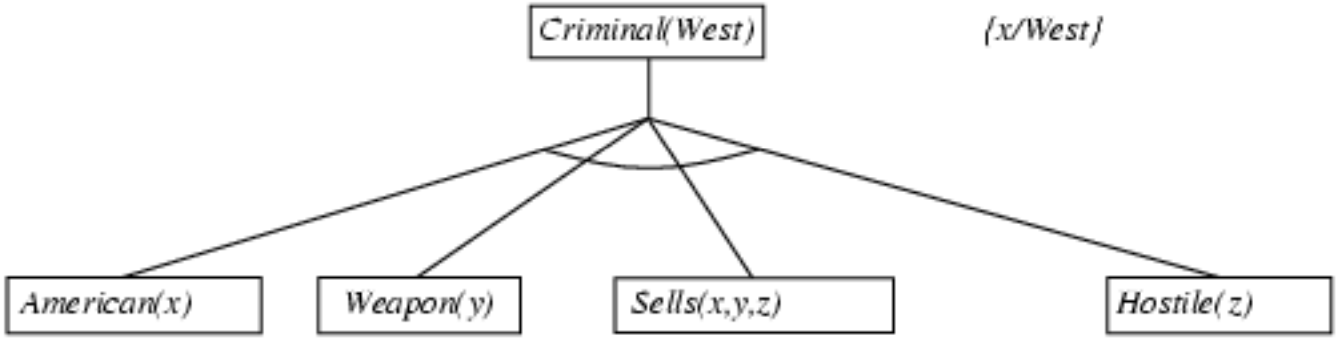
از هدف حرکت می کند تا به داده ها برسد یعنی هدف گراست

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

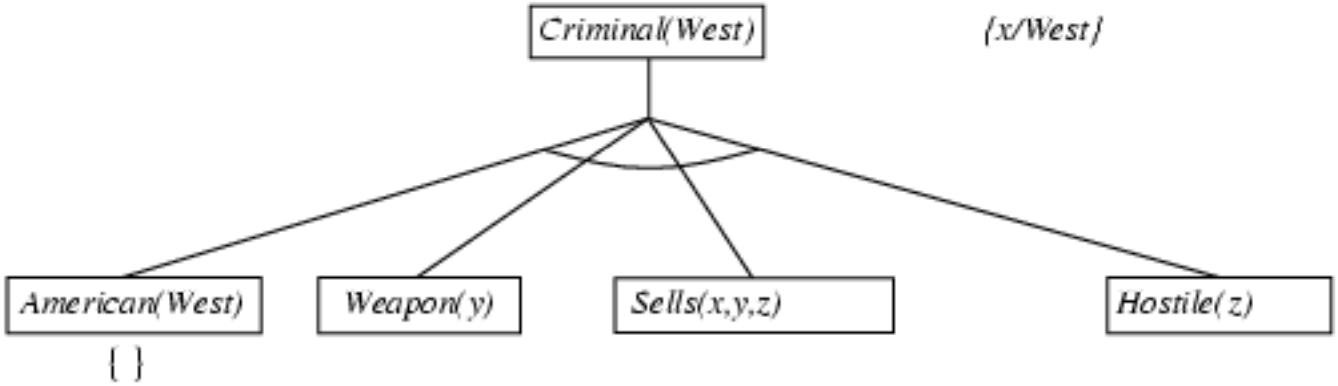
  if $goals$ is empty **then return** $\{\theta\}$
  $q' \leftarrow$ SUBST(θ, FIRST($goals$))
  for each r **in** KB where STANDARDIZE-APART(r) $= (p_1 \wedge \ldots \wedge p_n \Rightarrow q)$
    and $\theta' \leftarrow$ UNIFY(q, q') succeeds
    $ans \leftarrow$ FOL-BC-ASK(KB, $[p_1, \ldots, p_n | \text{REST}(goals)]$,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

مثالی از الگوریتم زنجیره ای پس رو

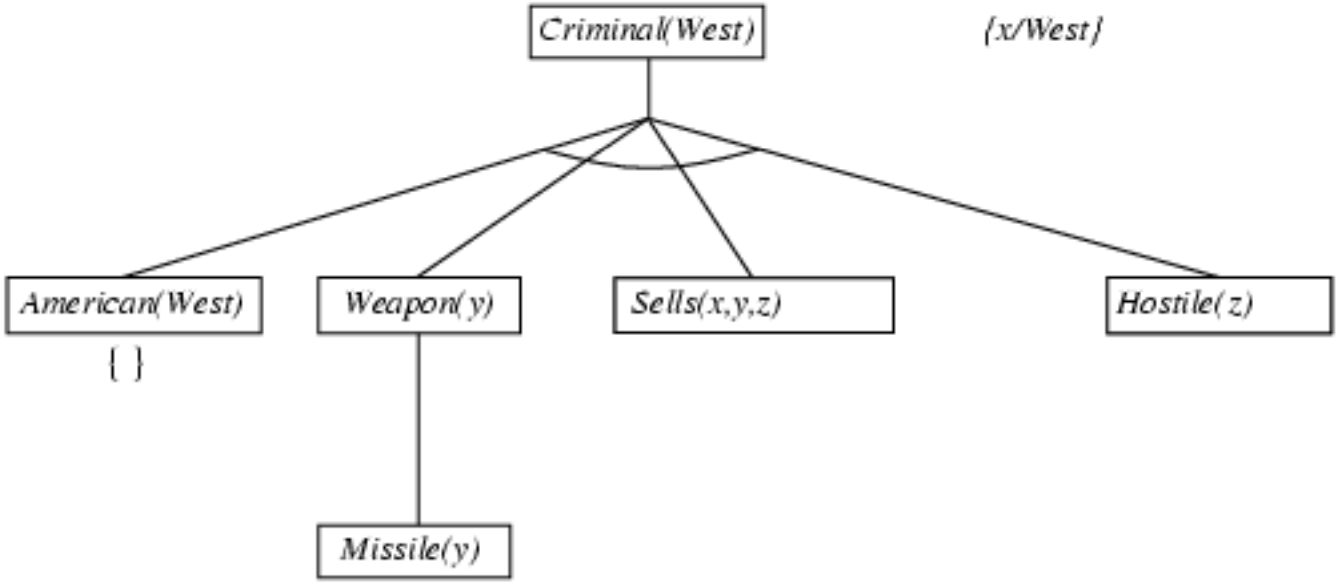
مثالی از الگوریتم زنجیره ای پس رو



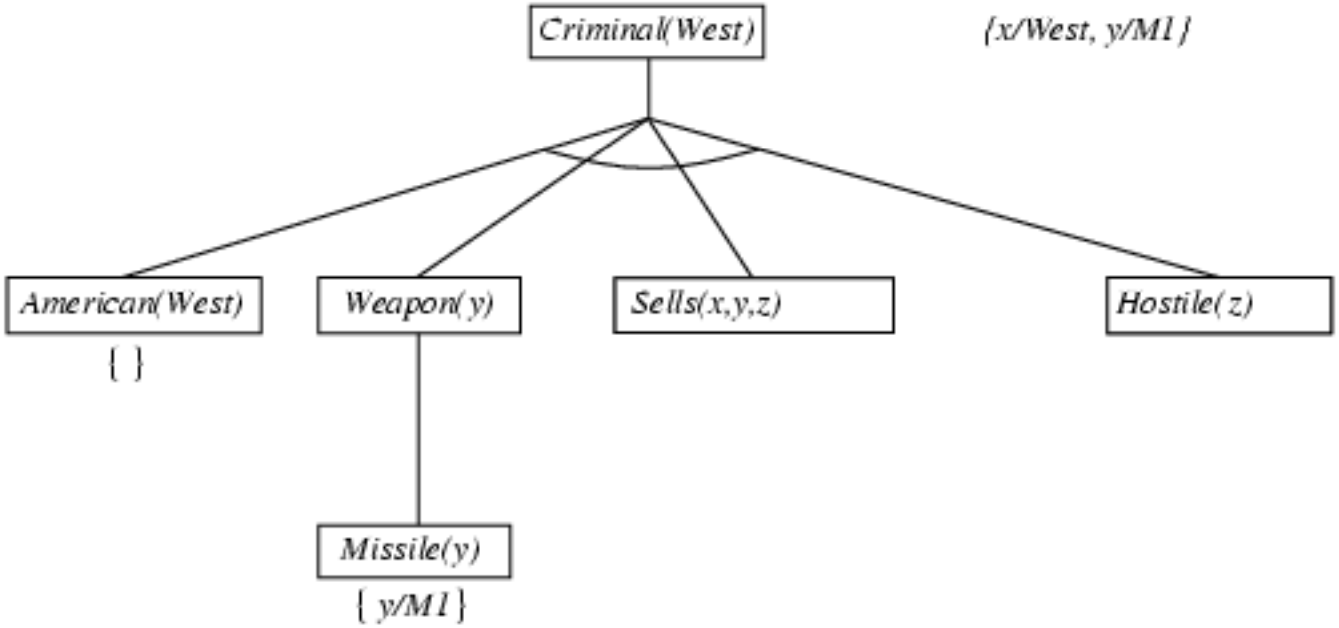
مثالی از الگوریتم زنجیره ای پس رو



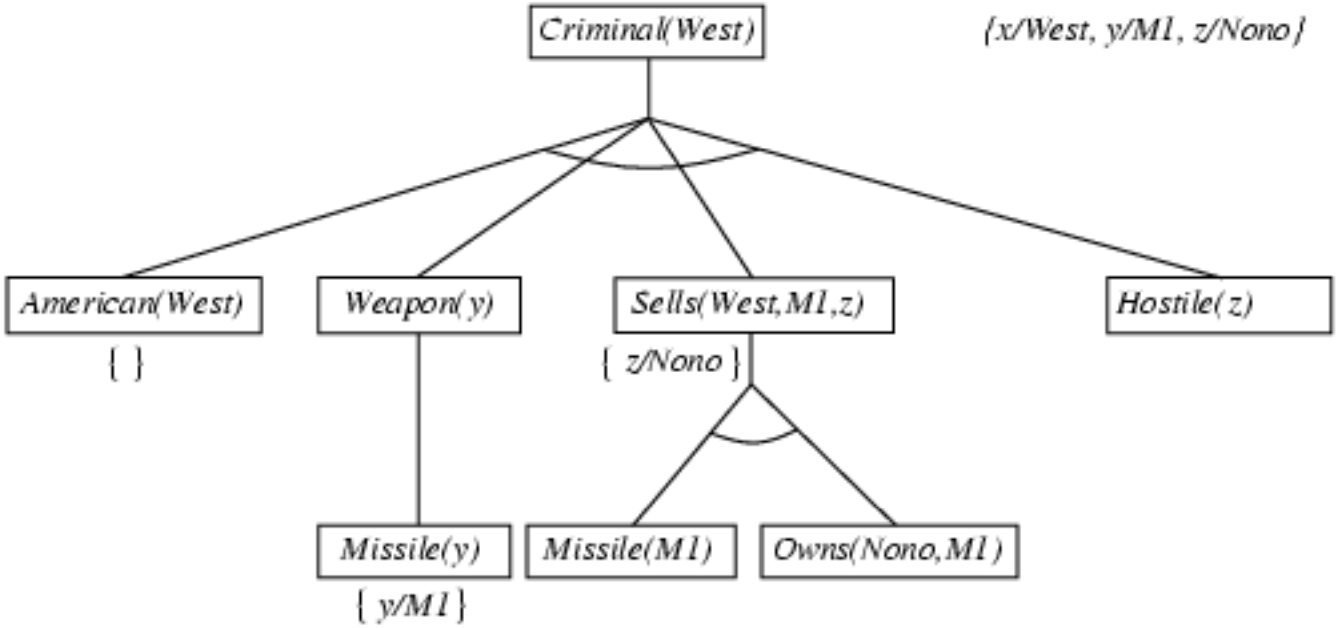
مثالی از الگوریتم زنجیره ای پس رو



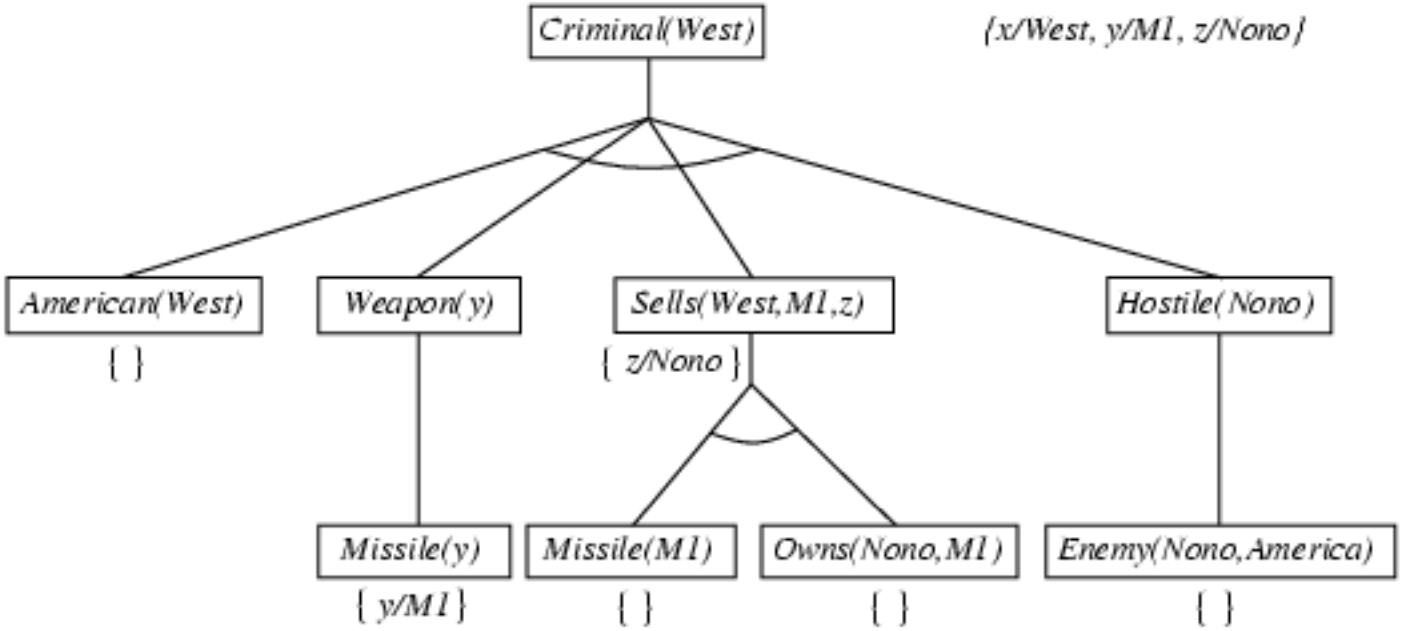
مثالی از الگوریتم زنجیره ای پس رو



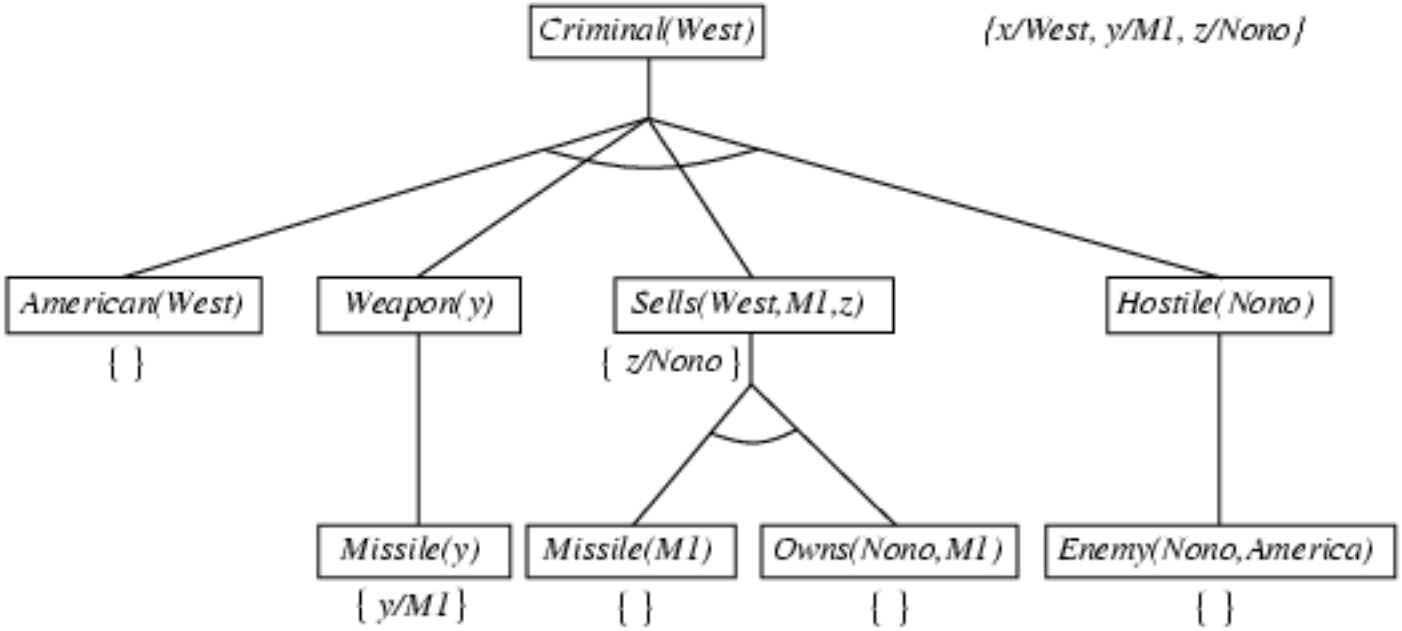
مثالی از الگوریتم زنجیره ای پس رو



مثالی از الگوریتم زنجیره ای پس رو



مثالی از الگوریتم زنجیره ای پس رو



برنامه نویسی منطقی (اثبات استلزام و ...) – پرولوگ

- الگوریتم = منطق + کنترل
- بر اساس : الگوریتم زنجیره ای پس گرد با استفاده از عبارت های هورن
- برنامه =مجموعه ای از عبارات. $head :- literal_1, \ldots literal_n$

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

یک برنامه بسیار ساده پرولوگ

یک برنامه بسیار ساده پرولوگ را در نظر می گیریم. این برنامه شامل چهار کلاز است که در یک دیتابیس ذخیره شده اند.

likes(hossein, food).
likes(hossein, icecream).
likes(naeem, icecream).
likes(naeem, hossein).
?- likes(hossein, X) , likes(naeem, X).

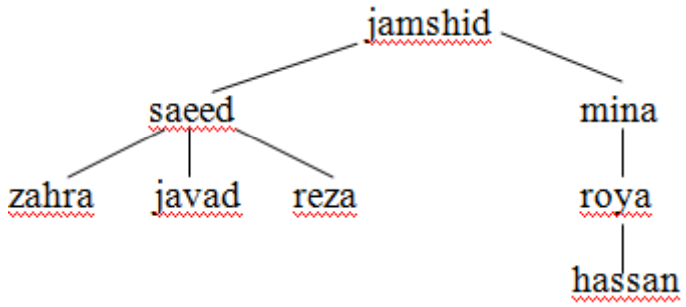
Query که در انتها آمده می پرسد که آیا حسین چیزی را دوست دارد که نعیم هم آنرا دوست دارد؟ پرولوگ اولین ترم از query را می گیرد likes(hossein, X) و تلاش می کند که آنرا بایک کلاز در دیتابیس unify کند. پرولوگ با تولید جانشینی $\{X \leftarrow food\}$ موفق می شودکه دو جمله likes(hossein, food)و likes(hossein, X)را بایکدیگر unifyکند. سپس پرولوگ جانشینی به دست آمده را به تمامی ترم های queryاعمال می کند. سپس به سراغ دومین ترم می رود، که اکنون likes(naeem, food)شده است. این جمله با هیچ جمله دیگری در دیتابیس unifyنمی شود.

بعد از هر شکست، پرولوگ backtrackمی کند. یعنی به unificationقبلی باز می گردد، که در این مورد unify کردن likes(hossein, X)با likes(hossein, food)است. اکنون پرولوگ تلاش می کند که اولین ترم queryرابا یک کلاز دیگر در دیتابیس unifyکند، که likes(hossein, icecream)است.

این دو ترم با جانشینی $\{X \leftarrow icecream\}$ با یکدیگر unifyمی شوند، که به ترم دوم queryیعنی likes(naeem, X)هم اعمال می شود تا likes(naeem, icecream)را ایجاد کند. ترم جدید با سومین کلاز دیتابیس قابل unifyشدن است.

بعد از تمام شدن کار با تمامی ترم های query، پرولوگ جانشینی های به دست آمده را خروجی می دهد، که در این جا X=icecreamاست.

یکی از مسائل کلاسیکی که معمولاً در ابتدای آموزش پرولوگ مطرح می کنند مساله شجره نامه خانوادگی می باشد.
برای مثال شجره نامه زیر را درنظر بگیرید:



این شجره نامه را به زبان پرولوگ توصیف می نماییم. برای این کار سه محمول پایه یعنی male ،female و parent را اختیار می کنیم و با یک سری درخت را نمایش می دهیم.

```
male(jamshid).
male(saeed).
male(javad).
male(reza).
male(hassan).
female(zahra).
female(mina).
female(roya).
parent( jamshid, saeed).
parent( jamshid, mina).
parent( saeed, javad).
parent( saeed, zahra).
parent( saeed, reza).
parent( mina, roya).
parent( roya, hassan).
```

پس از ساختن پایگاه دانش یک سری query به سیستم می دهیم.

- Is hassan the parent of saeed?

Query: parent(hassan, saeed).

- Who is saeed's parent?

Query: parent(X, saeed).

- Who were the *children* of saeed?

Query: parent(saeed, X).

- **لیست ها در پرولوگ**

```
append([],Y,Y)
```
الحاق با لیست خالی برابر با خود لیست

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

- پرس و جو :
```
append(A,B,[1,2]) ?
```

- پاسخ ها :

```
A=[]    B=[1,2]
A=[1]   B=[2]
A=[1,2] B=[]
```

تحلیل

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

زمانی که $\texttt{Unify}(\ell_i, \neg m_j) = \theta$.

- مثال

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

با $\theta = \{x/Ken\}$

تبدیل به فر م CNF

هر کس همه حیوانات را دوست داشته باشد، توسط یک نفردوست داشته خواهد شد.

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

۱. حذف یک دوشرطی ها و یک شرطی ها

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

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

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

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

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

۳. استاندارد ساز متغیرها

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

٤. اسکولم سازی

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

٥-حذف سورها (عمومی)

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

٦-توزیع $\vee$ بر $\wedge$

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

اثبات به روش تحلیل