
madsage
IRan Education
Research
NETwork
(IRERNET)
شبکه آموزشی-پژوهشی مادسیج
با هدف بهبود پیشرفت علمی
و دسترسی راحت به اطلاعات
برای جامعه بزرگ علمی ایران
ایجاد شده است
مادسیج
شبکه آموزشی –پژوهشی ایران
madsg.com
مادسیج

استنتاج در منطق مرتبه اول

فصل نهم

سید ناصر رضوی

E-mail: razavi@Comp.iust.ac.ir

۱۳۸۴

مقدمه

- تقلیل استنتاج مرتبه–اول به استنتاج گزاره ای
- یکسان سازی (Unification)
- MP تعمیم یافته (GMP)
- زنجیره استنتاج روبه جلو
- زنجیره استنتاج رو به عقب
- رزولوشن

نمونه سازی عمومی (UI)

- هر نمونه از یک جمله دارای سور عمومی، توسط آن جمله قابل استلزام است.

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

- مثال: $\forall x\ King(x) \land Greedy(x) \Rightarrow Evil(x)$

$King(John) \land Greedy(John) \Rightarrow Evil(John)$

$King(Richard) \land Greedy(Richard) \Rightarrow Evil(Richard)$

$King(Father(John)) \land Greedy(Father(John)) \Rightarrow Evil(Father(John))$

.
.
.

نمونه سازی وجودی (EI)

- برای هر جمله α، متغیر v و سیمبول ثابت k که در جای دیگری از پایگاه دانش ظاهر نشده باشد:

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

- مثال: $\exists x\ Crown(x) \land OnHead(x,John)$

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

به شرطی که C_1 یک سیمبول جدید باشد، که به آن ثابت اسکولم می گویند.

تقلیل به استنتاج گزاره ای

- فرض کنید که KB فقط شامل جملات زیر باشد:

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

- با نمونه سازی جمله دارای سور عمومی به تمام طرق ممکن، 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 گزاره ای سازی شده است. سیمبول های گزاره ای عبارتند از:

King(John), Greedy(John), Evil(John), King(Richard), etc

تقلیل به استنتاج گزاره ای (ادامه)

- هر پایگاه دانش در FOL می تواند به گونه ای گزاره ای سازی شود که استلزام را حفظ کند
- یک جمله توسط KB جدید قابل استلزام است اگر و فقط اگر توسط KB اصلی قابل استلزام باشد
- ایده: پایگاه دانش و کوئری را گزاره ای سازی کن، رزولوشن را اعمال و نتیجه را برگردان
- مشکل: در مورد سیمبول های تابعی تعداد نامحدودی ترم زمینی وجود دارد، مثلاً:

$$Father(Father(Father(John)))$$

تقلیل به استنتاج گزاره ای (ادامه)

- 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
- تئوری تورینگ و چرچ (۱۹۳۶): استلزام برای FOL **نیمه تصمیم پذیر** است
  - الگوریتم هایی وجود دارند که برای هر جمله قابل استلزام پاسخ بله را تولید می کنند، ولی الگوریتمی وجود ندارد که برای هر جمله ای که قابل استلزام نباشد پاسخ خیر تولید کند.

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)
$\forall y\ Greedy(y)$
Brother(Richard,John)

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

یکسان سازی (Unification)

- اگر بتوانیم یک جانشینی مانند θ بیابیم به طوری که King(x) و Greedy(x) با King(John) و Greedy(y) تطبیق یابند، آنگاه استنتاج بلافاصله بدست می آید.
  $\theta = \{x/John, y/John\}$
- $Unify(\alpha,\beta) = \theta$ if $\alpha\theta = \beta\theta$

p	q	θ
Knows(John,x)	Knows(John,Jane)	
Knows(John,x)	Knows(y,OJ)	
Knows(John,x)	Knows(y,Mother(y))	
Knows(John,x)	Knows(x,OJ)	

یکسان سازی (Unification)

- اگر بتوانیم یک جانشینی مانند θ بیابیم به طوری که King(x) و Greedy(x) با King(John) و Greedy(y) تطبیق یابند، آنگاه استنتاج بلافاصله بدست می آید.
  $\theta = \{x/John, y/John\}$
- $Unify(\alpha,\beta) = \theta$ if $\alpha\theta = \beta\theta$

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

یکسان سازی (Unification)

- اگر بتوانیم یک جانشینی مانند θ بیابیم به طوری که King(x) و Greedy(x) با King(John) و Greedy(y) تطبیق یابند، آنگاه استنتاج بلافاصله بدست می آید.
  $\theta = \{x/John, y/John\}$
- $Unify(\alpha,\beta) = \theta$ if $\alpha\theta = \beta\theta$

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))	
Knows(John,x)	Knows(x,OJ)	

یکسان سازی (Unification)

- اگر بتوانیم یک جانشینی مانند θ بیابیم به طوری که King(x) و Greedy(x) با King(John) و Greedy(y) تطبیق یابند، آنگاه استنتاج بلافاصله بدست می آید.
  $\theta = \{x/John, y/John\}$
- $Unify(\alpha,\beta) = \theta$ if $\alpha\theta = \beta\theta$

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)	

یکسان سازی (Unification)

- اگر بتوانیم یک جانشینی مانند θ بیابیم به طوری که King(x) و Greedy(x) با King(John) و Greedy(y) تطبیق یابند، آنگاه استنتاج بلافاصله بدست می آید.
  $\theta = \{x/John, y/John\}$
- Unify(α,β) = θ if αθ = βθ

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}

استاندارد سازی متغیرها: مثلاً Knows(z, OJ)

یکسان سازی

- برای یکسازی سازی Knows(John, x) و Knows(y, z) دو جانشینی وجود دارد یا

1) $\theta = \{y/John, x/z\}$
2) $\theta = \{y/John, x/John, z/John\}$

- اولین یکسان ساز عمومی تر از دومی می باشد
- فقط یک عمومی ترین یکسان ساز (MGU) وجود دارد که منحصر به فرد می باشد.

MGU = { y/John, x/z }

الگوریتم یکسان سازی

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

   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(ARGS[x], ARGS[y], UNIFY(OP[x], OP[y], θ))
   else if LIST?(x) and LIST?(y) then
       return UNIFY(REST[x], REST[y], UNIFY(FIRST[x], FIRST[y], θ))
   else return failure
```


الگوریتم یکسان سازی

```
function UNIFY-VAR(var, x, θ) returns a substitution
   inputs: var, a variable
           x, any expression
           θ, the substitution built up so far

   if {var/val} ∈ θ then return UNIFY(val, x, θ)
   else if {x/val} ∈ θ then return UNIFY(var, val, θ)
   else if OCCUR-CHECK?(var, x) then return failure
   else return add {var/x} to θ
```

Generalized Modus Ponens (GMP)

$$\frac{p_1', p_2', \ldots, p_n', (p_1 \wedge p_2 \wedge \ldots \wedge p_n \Rightarrow q)}{q\theta} \quad \text{where } p_i'\theta = p_i\theta \text{ for all } i$$

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 θ is *Evil*(*John*)

- GMP با پایگاه دانشی از فراکردهای معین (دقیقاً یک لیترال مثبت) کار می کند
- فرض می شود که تمام متغیرها دارای سور عمومی هستند.

صحت GMP

- باید نشان دهیم:

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

به شرطی که برای هر i داشته باشیم: $p_i'\theta = p_i\theta$

- لم: برای هر جمله p توسط UIداریم: $p \models p\theta$

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.
2. p_1', \; …, \;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

پایگاه دانش نمونه

- 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 Colonel West is a criminal

پایگاه دانش نمونه (ادامه)

... it is a crime for an American to sell weapons to hostile nations:

American(x) ∧ Weapon(y) ∧ Sells(x,y,z) ∧ Hostile(z) ⇒ Criminal(x)

Nono … has some missiles, i.e., ∃x Owns(Nono,x) ∧ Missile(x):

Owns(Nono,M1) and Missile(M1)

… all of its missiles were sold to it by Colonel West

Missile(x) ∧ Owns(Nono,x) ⇒ Sells(West,x,Nono)

Missiles are weapons:

Missile(x) ⇒ Weapon(x)

An enemy of America counts as "hostile“:

Enemy(x,America) ⇒ Hostile(x)

West, who is American …

American(West)

The country Nono, an enemy of America

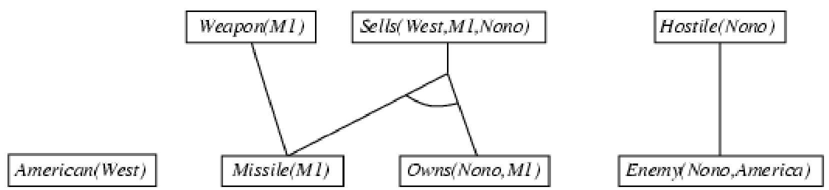
الگوریتم استنتاج رو به جلو

function FOL-FC-ASK(KB, α) **returns** a substitution or *false*

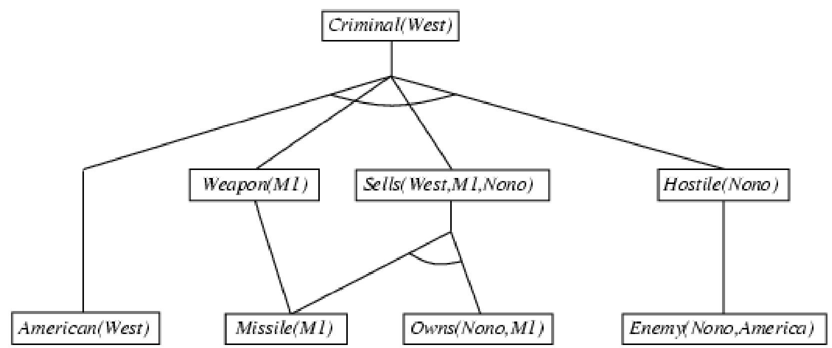
  repeat until *new* is empty
    $new \leftarrow \{\,\}$
    for each sentence r **in** KB **do**
      $(p_1 \wedge \ldots \wedge p_n \Rightarrow q) \leftarrow$ STANDARDIZE-APART(r)
      for each θ such that $(p_1 \wedge \ldots \wedge p_n)\theta = (p'_1 \wedge \ldots \wedge p'_n)\theta$
          for some $p'_1, \ldots, p'_n$ in KB
        $q' \leftarrow$ SUBST(θ, q)
        if q' is not a renaming of a sentence already in KB or *new* **then do**
          add q' to *new*
          $\phi \leftarrow$ UNIFY(q', α)
          if ϕ is not *fail* **then return** ϕ
    add *new* to KB
  return *false*

اثبات توسط استنتاج رو به جلو

اثبات توسط استنتاج رو به جلو



اثبات توسط استنتاج رو به جلو



خصوصیات استنتاج رو به جلو

- برای فراکردهای معین FOL، کامل و صحیح است
- Datalog = فراکردهای معین FOL بدون سیمبول تابعی
- FC برای Datalog با تعداد تکرارهای محدودی متوقف می شود.
- در حالت کلی، اگر پایگاه دانش مستلزم α نباشد، ممکن است متوقف نشود
- این اجتناب ناپذیر است: استلزام با فراکردهای معین نیمه تصمیم پذیر است

کارآیی استنتاج رو به جلو

- استنتاج رو به جلوی افزایشی: در تکرار k اُم نیاز به تطبیق قانونی که هیچ کدام از شرایطش در تکرار k-1 اضافه نشده اند، نیست
- خود انطباق می تواند پرهزینه باشد:
  – شاخص بندی پایگاه داده اجازه می دهد که حقایق شناخته شده در زمان O(1) بازیابی شوند.
- استنتاج رو به جلو به طور گسترده ای در پایگاه های داده استنتاجی به کار می رود

الگوریتم استنتاج رو به عقب

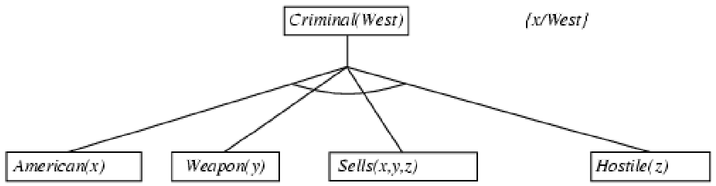
```
function FOL-BC-ASK(KB, goals, θ) returns a set of substitutions
   inputs: KB, a knowledge base
           goals, a list of conjuncts forming a query
           θ, the current substitution, initially the empty substitution { }
   local variables: ans, a set of substitutions, initially empty

   if goals is empty then return {θ}
   q' ← SUBST(θ, FIRST(goals))
   for each r in KB where STANDARDIZE-APART(r) = (p1 ∧ ... ∧ pn ⇒ q)
           and θ' ← UNIFY(q, q') succeeds
      ans ← FOL-BC-ASK(KB, [p1, ..., pn|REST(goals)], COMPOSE(θ, θ')) ∪ ans
   return ans
```

SUBST(COMPOSE(θ1, θ2), p) = SUBST(θ2, SUBST(θ1, p))

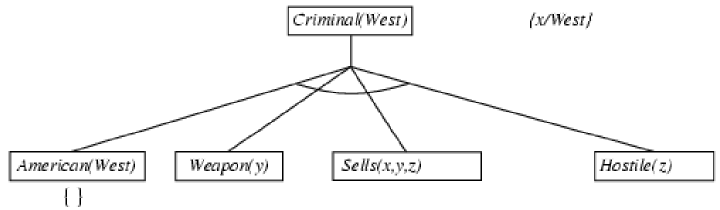
مثال استنتاج رو به عقب

Criminal(West)

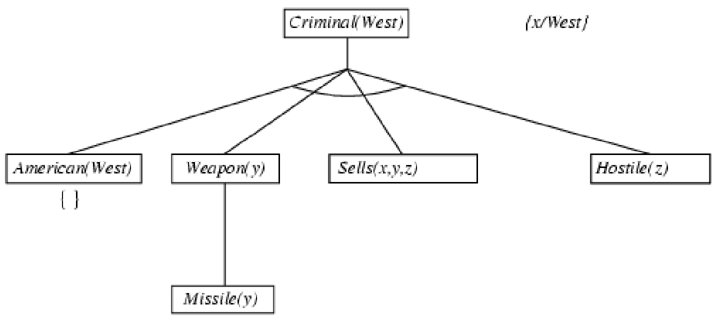
مثال استنتاج رو به عقب



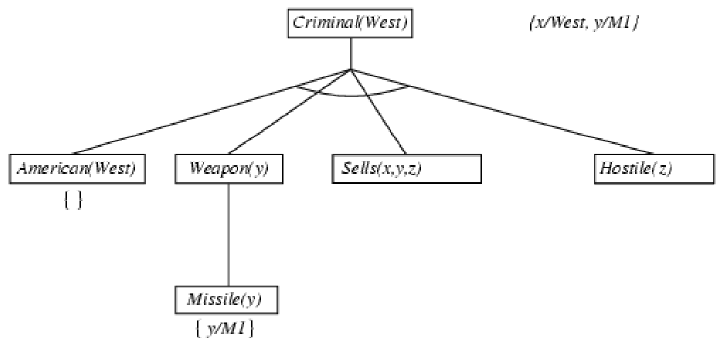
مثال استنتاج رو به عقب



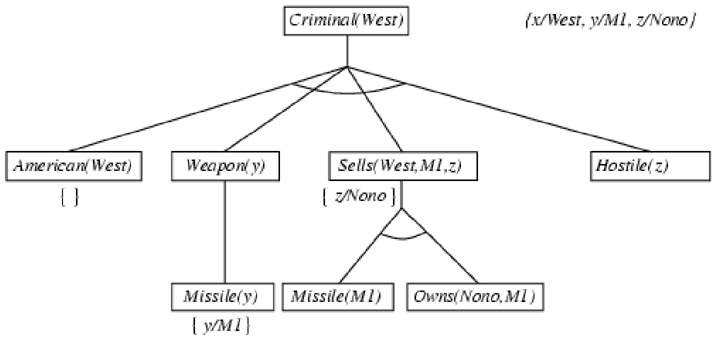
مثال استنتاج رو به عقب



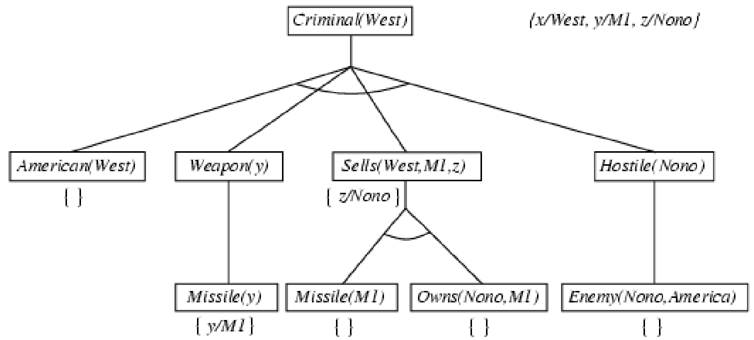
مثال استنتاج رو به عقب



مثال استنتاج رو به عقب



مثال استنتاج رو به عقب



خصوصیات استنتاج روبه عقب

- اثبات: جستجوی اول-عمق بازگشتی – فضای حالت بر حسب اثبات خطی
- ناکامل به دلیل وجود حلقه ها
  - راه حل: مقایسه هدف فعلی با تمام اهداف موجود در پشته
- ناکارآ به دلیل زیرهدف های تکرای (چه موفقیت و چه شکست)
  - راه حل: ذخیره نتایج قبلی (با مصرف حافظه بیشتر) – نوعی memoization
- به طور گسترده ای در **برنامه نویسی منطقی** استفاده می شود

برنامه نویسی منطقی: پرولوگ

- Algorithm = Logic + Control
- اصول: عقبگرد با فراکردهای هورن- به طور گسترده ای در اروپا و ژاپن استفاده شده است (در پروژه نسل پنجم کامپیوترها)
- برنامه = مجموعه ای از فراکردها
- فراکرد =

head :- literal$_1$, … literal$_n$.

- مثال:

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

خصوصیات پرولوگ

- استنتاج روبه عقب، اول-عمق و از چپ به راست
- عملگرهایی برای محاسبات ریاضی مثلاً : X is Y * Z + 3
- مسندهایی با عوارض جانبی (مثلاً ورودی و خروجی)
- فرض دنیای بسته (‹نقیض به عنوان شکست›)
  - مثال: با داشتن

alive(X) :- not dead(X).

alive(Joe) موفق می شود اگر not dead(Joe) شکست بخورد

پرولوگ

- اضافه کردن یک لیست به انتهای لیست دیگر برای تولید یک لیست جدید (عمل Append)

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

- کوئری:

```
append(A,B,[1,2]) ?
```

- پاسخ ها:

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

A=[1]     B=[2]

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


رزولوشن: خلاصه

- رزولوشن در منطق مرتبه اول

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

که $\text{Unify}(l_i, \neg m_j) = \theta$

- فرض می شود که دو فراکرد استاندارد شده اند به گونه ای که شامل متغیر یکسانی نباشند
- مثال:

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

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

تبدیل به CNF

- Everyone who loves all animals is loved by someone:
  $\forall x\ [\forall y\ Animal(y) \Rightarrow Loves(x,y)] \Rightarrow [\exists y\ Loves(y,x)]$
- 1. Eliminate biconditionals and implications
- $\forall x\ [\neg\forall y\ \neg Animal(y) \vee Loves(x,y)] \vee [\exists y\ Loves(y,x)]$
- 2. Move $\neg$ inwards: $\neg\forall x\ p \equiv \exists x\ \neg p$, $\neg\exists x\ p \equiv \forall x\ \neg p$
- $\forall x\ [\exists y\ \neg(\neg Animal(y) \vee Loves(x,y))] \vee [\exists y\ Loves(y,x)]$
  $\forall x\ [\exists y\ \neg\neg Animal(y) \wedge \neg Loves(x,y)] \vee [\exists y\ Loves(y,x)]$

تبدیل به CNF (ادامه)

3. Standardize variables: each quantifier should use a different one
4.

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

4. Skolemize: a more general form of existential instantiation.

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:
6.
6.

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

6. Distribute $\vee$ over $\wedge$:

اثبات بوسیله رزولوشن: مثال