مترجم: سهراب جلوه گر
ویرایش دوّم ، بهار ۱۳۸۸

ریوس مطالب

- چرا منطق مرتبه ی اوّل ؟
- ظاهر ومعنای منطق مرتبه ی اوّل
- بازی با عبارات
- دنیای wumpus در منطق مرتبه ی اوّل

موافقین و مخالفین منطق گزاره ای

موافقین

☺ منطق گزاره ای ، اظهاری (اعلانی) است : اجزای گرامری با واقعیّات برابرند . ☺ منطق گزاره ای ، به اطّلاعات جزیی / فصلی / منفی شده اجازه می دهد (برخلاف بیش تر ساختارهای داده ای

و پایگاه های داده ای) .☺ منطق گزاره ای ، **ترکیبی** [۱] می باشد ؛ به عنوان مثال ، مفهوم $B_{1,1} \wedge P_{1,2}$ از $B_{1,1}$ و از $P_{1,2}$ به وجود می آید . ☺ مفهوم در منطق گزاره ای ، **مستقل از متن** [۲] است . (برخلاف زبان طبیعی ، که مفهوم وابسته به متن می باشد) .

مخالفین

☹ منطق گزاره ای محدودیّت زیادی در قدرت بیان دارد (برخلاف زبان طبیعی) . مثال ، ما نمی توانیم بگوییم " چاله ها باعث هوای مطلوب در خانه های مجاور می شوند " به جز با نوشتن یک عبارت برای هر خانه .

منطق مرتبه ی اوّل – از آنجایی که منطق گزاره ای ، محتویات واقعی دنیا را به خود می گیرد ، منطق گزاره ای (مثل زبان طبیعی) محتویات جهان را به خود می گیرد .

اشیاء : مردم ، خانه ها ، اعداد ، تئوری ها ، رونالد مک دونالد [۳] ، رنگ ها ، بازی های بیسبال [۴] ، جنگ ها ، قرن ها و ..... می باشند .

وابستگی ها (ارتباطات یا روابط) : قرمز ، گرد [۵] ، ساختگی [۶](جعلی یا قلابی) ، اوّل [۷] (عمده) ، ساختمان چند طبقه [۱] و ..... می باشند .

---

[۱] compositional
[۲] context-independent
[۳] Ronald McDonald
[۴] baseball
[۵] round
[۶] bogus
[۷] prime

- برادر ... ، بزرگ تر از ، درون ... ، قطعه ای از ... ، به رنگ ، اتفاق افتاده بعد از ، داشتن ، آمدن میان و ..... می باشند .

- **عملکردها [۲](تابع ها) :** پدر ... ، بهترین دوست ، تصدی ... ، یکی بیش تر از ، پایان ..... می باشند .

منطق در حالت کلّی

زبان	هستی شناسی [۳]	معرفت شناسی [٤]
منطق گزاره ای	واقعیّات	**درست / غلط / ناشناخته**
منطق مرتبه ی اوّل	واقعیّات ، اشیاء ، رابطه ها	**درست / غلط / ناشناخته**
منطق زمانی [۵]	واقعیّات ، اشیاء ، رابطه ها ، زمان ها	**درست / غلط / ناشناخته**
تیوری احتمال [٦]	واقعیّات	**اندازه ی اعتقاد [۷]**
منطق فازی [۸]	**واقعیّات + درجه ی درستی [۹]**	**مقدار فاصله ی شناخته شده** [۱۰]

---

[۱] multistoried
[۲] functions
[۳] Ontological Commitment
[۴] Epistemological Commitment
[۵] Temporal logic
[۶] Probability logic
[۷] degree of belief
[۸] Fuzzy logic
[۹] degree of truth
[۱۰] known interval value

ظاهر منطق مرتبه ی اوّل - عناصر اساسی عبارتند از :

- ثابت ها مثل : KingJohn,2,UCB,Koblenz,C,...

- گزاره ها مثل : Brother,>,=,...

- توابع مثل : Sqrt,LeftLegOf,...

- متغیّرها مثل : x,y,a,b,...

- رابط ها مثل : $\wedge,\vee,\neg,\Rightarrow,\Leftrightarrow$

- تساوی : =

- کمّیّت سنج ها (سورها) : $\forall,\exists$

عبارات اتمیک [۱]

- عبارت اتمیک = $predicate(term_1,...,term_n)$ یا $term_1 = term_2$

واژه = $function(term_1,...,term_n)$ ، که یا ثابت است و یا متغیّر می باشد .

مثال :

Brother(KingJohn,RichardTheLionheart) >
(Length(LeftLegOf(Richard)),Length(LeftLegOf(KingJohn)))

---

[۱] Atomic

در عبـارت Brother(KingJohn,RichardTheLionheart) ، Brother گـزاره ، KingJohn ثابـــــــــت ، RichardTheLionheart ثابـــــــــت ، KingJohn واژه ، RichardTheLionheart واژه و کل عبارت یک عبارت اتمیک می باشد .

> (Length(LeftLegOf(Richard)),Length(LeftLegOf(KingJohn)))

در عبارت فوق ، > گزاره ، Length تابع ، LeftLegOf تابع ، Richard ثابت ، Length تابع ، مجددا LeftLegOf تابع ، KingJohn ثابت ، Length(LeftLegOf(Richard)) واژه ، Length(LeftLegOf(KingJohn)) واژه و کل عبارت (> (Length(LeftLegOf(Richard)),Length(LeftLegOf(KingJohn)))) ، عبارت اتمیک می باشد .

عبارات پیچیده – عبارات پیچیده ، همانند منطق گزاره ای ، از عبارات اتمیک با اسـتفاده از رابط ها ساخته می شوند ، مثل :

$$\neg S, S_1 \wedge S_2, S_1 \vee S_2, S_1 \Rightarrow S_2, S_1 \Leftrightarrow S_2$$

مثال :

$Sibling\,(KingJohn\,,Richard\,) \Rightarrow Sibling\,(Richard\,,KingJohn\,)$
$> (1,2) \vee \leq (1,2)$
$> (1,2) \wedge \neg > (1,2)$

در عبارت فوق ، Sibling ، گـزاره مـی باشـد . KingJohn واژه اسـت . Richard واژه است . Sibling طـرف راسـت گـزاره اسـت . Richard طـرف راسـت واژه اسـت . KingJohn طـــرف راســـت واژه اســـت . Sibling(KingJohn,Richard) ، عبـــارت اتمیـــک اســـت . Sibling(Richard,KingJohn) عبارت اتمیک است و کل عبارت ، عبارت پیچیده می باشد .

صحّت در منطق مرتبه ی اوّل – عبارت ها با رعایت یک مدل و یک تفسیر [۱] ، درست می باشند . مدل ، شامل اشیای بزرگ تر یا مساوی یک (عناصر دامنه) و رابط های میان آن ها می باشد ($\text{contains}\geq 1$) . تفسیر موارد مراجعه را برای موارد زیر تایین می نماید :

سمبل های ثابت [۲] ⟵ اشیاء

سمبل های گزاره ای [۳] ⟵ رابط ها

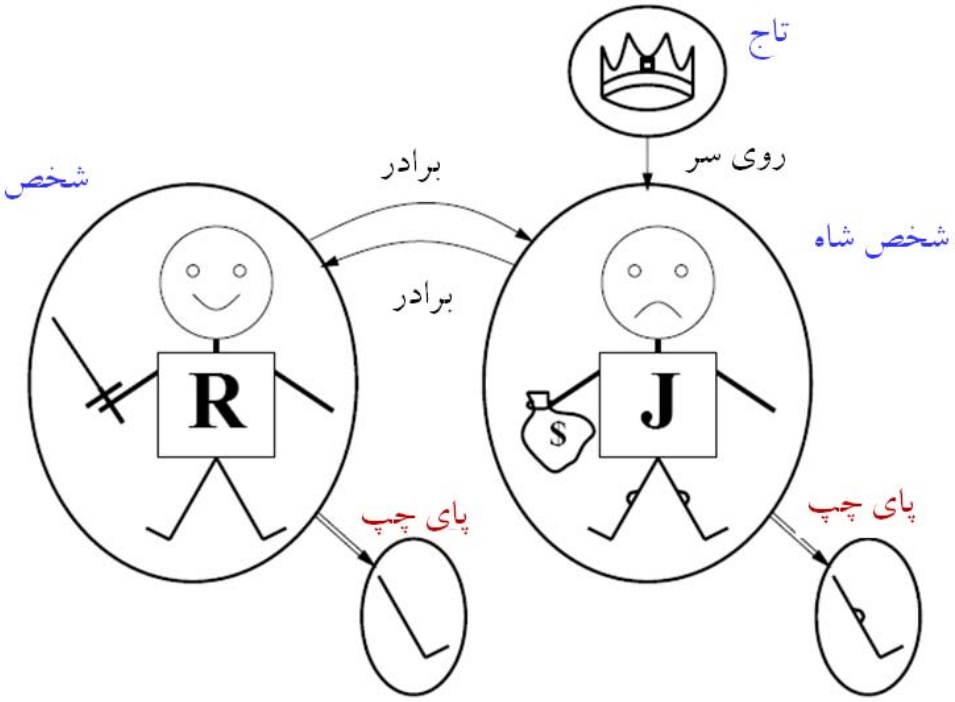
سمبل های تابعی [۴] ⟵ رابط های تابعی [۵]

یک عبارت اتمیک $predicate(term_1,...,term_n)$ در صورتی درست است که اشیاء ارجاع شده به آن توسط $term_1,...,term_n$ با ارتباط های ارجاع شده توسط گزاره در ارتباط باشند .

مثالی برای مدل های منطق مرتبه ی اوّل

---

[۱] interpretation
[۲] constant symbols
[۳] predicate symbols
[۴] function symbols
[۵] functional relations



مثال صحت – به تفسیری به صورت زیر توجّه نمایید :

ریچارد [۱] ⟵ فردی دلیر [۲] است

جان [۳] ⟵ پادشاهی بد

برادر ⟵ ارتباط برادری [۴]

---

[۱] Richard

[۲] Lionheart

[۳] John

[۴] brotherhood relation

تحت این تفسیر ، Brother(Richard,John) فقط در مورد **ریچـارد فـردی دلیـر است** درست است و **جان پادشاهی بد اسـت** درسـت اسـت ، کـه در ایـن مـدل دارای ارتبـاط برادری هستند .

سور عمومی – همه در برکلی [۱] زرنگ [۲] هستند : $\forall x : At(x, Berkeley) \Rightarrow Smart(x)$

$\forall x\ P$ ، در یک مدل m درست است در صورتی که P به ازای هـر x ممکـن در مـدل درسـت باشد .

$$(At(KingJohn, Berkeley) \Rightarrow Smart(KingJohn)) \wedge (At(Richard, Berkeley) \Rightarrow Smart(Richard)) \wedge (At(Berkeley, Berkeley) \Rightarrow Smart(Berkeley)) \wedge \ldots$$

$\forall x : At(x, Berkeley) \wedge Smart(x)$ معنای این عبارت این است که " همه در برکلی هستند و همه باهوش هستند " .

تعیین خواص وجودی [۳]

شخصـی در Stanford بـاهوش اسـت را بـه ایـن صـورت مـی تـوانیم بیـان نمـاییم : $\exists x At(x, S\tan ford) \wedge Smart(x)$. که با مثال های زیر برابر است :

$$(At(KingJohn, S\tan ford) \wedge Smart(KingJohn))$$
$$\vee (At(Richard, S\tan ford) \wedge Smart(Richard))$$
$$\vee (At(S\tan ford, S\tan ford) \wedge Smart(S\tan ford))$$

خصوصیّات کمّیّت سنج ها (سورها)

---

[۱] Berkely

[۲] smart

[۳] Existential quantification

$\forall x \forall y$ مثل $\forall y \forall x$ می باشد . $\exists x \exists y$ مثل $\exists y \exists x$ می باشد . $\exists x \forall y$ مثل $\forall y \exists x$ نمی باشد .

$\exists x \forall y Loves(x, y)$ ؛ یعنی : " شخصی وجود دارد که همه را در جهان دوست دارد "

$\forall y \exists x Loves(x, y)$ ؛ یعنی : " همه در جهان با لااقل یک فرد دوست می شوند "

دوگانگی کمّیّت سنج ها [۱] – هر چیزی می تواند با استفاده از دیگری بیان شود :

$$\forall x Likes(x, IceCream) \equiv \neg \exists x \neg Likes(x, IceCream)$$
$$\exists x Likes(x, Broccoli) \equiv \neg \forall x \neg Likes(x, Broccoli)$$

بازی با عبارات

برادران ، هم نژاد [۲] هستند را به این صورت بیان می نماییم :

$$\forall x,y \ Brother(x,y) \Rightarrow Sibling(x,y)$$

" برادری (هم نژادی) " [رابطه ای] [۳] متقارن [۴] است ؛ یعنی :

$$\forall x, y : Sibling(x, y) \Leftrightarrow Sibling(y, x)$$

مادر یکی والده ی یکی دیگر است :

$$\forall x, y : Mother(x, y) \Leftrightarrow (Female(x) \wedge Parent(x, y))$$

---

[۱] Quantifier duality
[۲] Sibling
[۳] مترجم
[۴] symmetric

اوّلین عموزاده بچه ای هم نژاد با والد است :

$$\forall x, y : FirstCou\sin(x, y) \Leftrightarrow \exists p,\ psParent(p, x) \wedge Sibling(ps, p) \wedge Parent(ps, y)$$

مترجم: سهراب جلوه گر
ویرایش دوّم ، بهار ۱۳۸۸

فصل دوازدهم

استنتاج در منطق مرتبه ی اوّل [1]

[1] Inference in First-order Logic

مترجم: سهراب جلوه گر
ویرایش دوّم ، بهار ۱۳۸۸

ریوس مطالب

- کاهنده ی استنتاج منطق مرتبه ی اوّل به استنتاج گزاره ای
- یک شکلی [۱]
- طرز عمل کلّی [۲]
- زنجیره ی مستقیم و معکوس
- برنامه نویسی منطقی [۳]
- تحلیل [٤]

---

[۱] unification
[۲] Generalized Modus Ponens
[۳] Logic programming
[۴] Resolution

مترجم: سهراب جلوه گر
ویرایش دوّم ، بهار ۱۳۸۸

تاریخچه ی مختصری از استدلال

در سال ۴۵۰ قبل از میلاد مسیح ، پیروان فلسفه ی رواقیون [۱] ، منطق گزاره ای و(شاید) استدلال را بنیان نهادند . | در سال ۳۲۲ قبل از میلاد مسیح ، ارسطو ، قیاس منطقی [۲] (قوانین استدلال) و سورها را بیان کرد . | در سال ۱۵۶۵ ، کاردانو [۳] ، تیوری احتمال [۴] (منطق گزاره ای [۵] + نامعلومی [۶]) را به وجود آورد . | در سال ۱۸۴۷ ، بول [۷] ، دوباره منطق گزاره ای را بررسی کرد . | سال ۱۸۷۹ ، فرگ [۸] ، منطق مرتبه ی اوّل را به وجود آورد . | در سال ۱۹۲۲ ، ویتجنستین [۹] ، اثبات توسط جدول صحت را بررسی کرد . | در سال ۱۹۳۰ ، گدل [۱۰] ، گفت : الگوریتم کاملی برای منطق مرتبه ی اوّل وجود دارد . | در سال ۱۹۳۰ ، هربرند [۱۱] ، الگوریتم کاملی برای منطق مرتبه ی اوّل به وجود آورد . | در سال ۱۹۳۱ ، گدل گفت : الگوریتم کاملی برای محاسبه وجود

---

[۱] Stoics
[۲] syllogisms
[۳] Cardano
[۴] probability theory
[۵] propositional logic
[۶] uncertainty
[۷] Boole
[۸] Frege
[۹] Wittgenstein
[۱۰] Godel
[۱۱] Herbrand

ندارد . | در سال ۱۹۶۰ ، دیویس- پوتنام [۱] ، الگوریتمی کاربردی برای منطق گزاره ای ارایه کرد . | در سال ۱۹۶۵ ، رابینسن [۲] الگوریتمی کاربردی برای منطق مرتبه ی اوّل – تحلیل به وجود آورد .

استنتاج در منطق مرتبه ی اوّل

می توانیم انواع استنتاج را با منطق مرتبه ی اوّل داشته باشیم .

برداشتن سورها

نمونه سازی (معرفی) عمومی [۳]

ما همیشه می توانیم یک واژه ی زمینه را با یک متغیّر جانشین نماییم مثلا در مثال زیر ؛ $\forall x LivesIn(x, Springfield) \Rightarrow knows(x, Homer)$ ، از آنجایی که این عبارت برای همه ی x ها صحیح می باشد ، ما می توانیم {x=Bart} را جایگزین نماییم : $LivesIn(Bart, Springfield) \Rightarrow knows(Bart, Homer)$ ، پس : $\frac{\forall \upsilon : \alpha}{Subs(\{\upsilon / g\}, \alpha)}$ برای هر متغیّر v و واژه ی زمینه ی g برقرار می باشد .

مثال ، نتیجه های $\forall x : King(x) \wedge Greedy(x) \Rightarrow Evil(x)$ عبارتند از :

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

---

[۱] Davis/Putnam

[۲] Rabinson

[۳] Universal instantiation (UI)

مترجم: سهراب جلوه گر
ویرایش دوّم ، بهار ۱۳۸۸

نمونه سازی (معرفی) وجودی [۱]

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 اِسکَلِم نامیده می شود . پس ، برای هر عبارت α و متغیّر v و سمبل ثابت k *که در جای دیگری در پایگاه دانش وجود ندارد* داریم :

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

مثال : نتیجه ی $\exists x Crown(x) \wedge OnHead(x, John)$ به صورت $Crown(C_1) \wedge OnHead(C_1, John)$ می تواند باشد که C_1 به وجود آمده از یک سمبل ثابت جدید به نام ثابت اسکلم [۲] می باشد .

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

گزاره بندی [۳]

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

---

[۱] Existential instantiation (EI)

[۲] Skolem constant

[۳] propositionalization

دهد که از قانون های استنتاج گزاره ای استفاده نماییم . البتّه این کار خیلی ناکارآمد می باشد ؛ تـا سـال ۱۹۶۰ از همین روش استفاده می کردند .

تبدیل به استدلال گزاره ای

تصور نمایید که پایگاه دانش فقط شامل عبارات زیر باشد :

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

معرفی عبارت عمومی در **همه ی موارد ممکن است** و داریم :

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

که پایگاه دانش جدید ، گزاره ای می باشد ، سمبل های گزاره ای عبارتند از :

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

و .....

تبدیل

– **ادعا :** یک عبارت زمینه به وسیله ی پایگاه دانش جدید در صورتی که به وسـیله ی پایگـاه دانـش اصلی به وجود آمده باشد ، موجود می شود .

– **ادعا :** هر پایگاه دانش منطق مرتبه ی اوّل می تواند برای حفظ موجودیت ، بـه صـورت گـزاره ای بیان شود .

مترجم: سهراب جلوه گر
ویرایش دوّم ، بهار ۱۳۸۸

– **ایده :** پایگاه دانش را به صورت گزاره ای و پرس و جو بیان نمایید ، تحلیل را به کار ببرید و نتایج را برگردانید .

– **مسأله :** با سمبل های توابع ، واژگان زمینه به صورت نامحدود وجود خواهند داشت .

مثال : Father(Father(Father(John)))

قضیّه ی هربرند (۱۹۳۰) : در صورتی که عبارت α توسط یک پایگاه دانش منطق مرتبه ی اوّل به وجود بیاید ، توسط یک زیرمجموعه ی محدود از پایگاه دانش های گزاره ای موجود می شود .

ایده : برای $n = 0$ تا بینهایت کارهای زیر را انجام بده

یک KB (پایگاه دانش) گزاره ای با عمق n بسازید

بررسی نمایید که آیا α توسط این KB تولید می شود

قضیّه ی تورینگ و چورچ [۱](۱۹۳۶) : به وجود آمدن [۲] در منطق مرتبه ی اوّل تقریبا قابل تصمیم گیری [۳] می باشد .

ایرادهای گزاره ها

گزاره ، برای تولید تعداد زیادی عبارات که به هم نامربوط هم هستند به کار می رود .

مثال :

---

۱ Turing,Church

۲ entailment

۳ semidecidable

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

حاصل به نظر می رسد که Evil(John) باشد ، اما گزاره بندی ، تعداد زیادی واقعیّات نظیر Greedy(Richard) که نامربوط هستند را به وجود می آورد . با گزاره های k - تایی p و n های ثابت ، $p.n^k$ معرفی وجود خواهد داشت . با سمبل های تابعی ، معرّفی های با مقادیر به مراتب بدتری وجود خواهد داشت !

یک شکلی

یک شکلی ، این است که ما فقط می خواهیم جایگزین هایی برای عباراتی که به ما کمک می کنند چیزهایی را ثابت نماییم پیدا کنیم . برای مثال ، اگر ما بدانیم :

$$\forall x : LivesIn(x, Springfield) \wedge WorksAt(x, PowerPlant) \Rightarrow know(x, Homer)$$

$$\forall y : LivesIn(y, Springfield)$$

$$WorksAt(MrSmithers, PowerPlant)$$

آن گاه ، ما باید قادر باشیم مستقیما نتیجه بگیریم knows(MrSmithers,Homer) با جایگزینی : {x/MrSmithers,y/MrSmithers} .

پس در مثال قبلی ، در صورتی که بتوانیم یک زیر وضعیت θ را پیدا کنیم ، می توانیم بلافاصله استدلال نماییم King (x) و Greedy (x) با King (John) و Greedy(y) مطابقت می کنند .

$\alpha\theta = \beta\theta$ اگر $\theta = \{x/John,y/John\}$, $Unify(\alpha,\beta) = \theta$

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)	fail

استاندارد کردن از روی هم افتادن [۱] متغیّرها جلوگیری می کند ، مثال : $Knows(z_{17}, OJ)$

طرز عمل کلّی [۲]

ایده ی اساسی این است که فرض کنیم یک استنتاج به شکل $P_1 \wedge P_2 \wedge ... \wedge P_i \Rightarrow Q$ را داریم و عبارات به صورت $P_1', P_2', ..., P_i'$ می باشند ، اگر مجموعه ای از جانشین ها به صورت $P_1 = P_1', P_2 = P_2', ..., P_n = P_n'$ وجود داشته باشند ، سپس ما می توانیم جانشین را به کار بگیریم و روش طرز عمل کلّی را برای به دست آوردن Q به کار بگیریم . *این تکنیک استفاده از جانشین ها برای جفت عبارات بالا ، برای به دست آوردن استنتاج ،* ***یک شکلی*** *نام دارد* . پردازه ی استنتاج ما حالا پیداکردن جانشین هایی که به ما اجازه خواهند داد عبارات جدید را به وجود آوریم می باشد .

الگوریتم یگانی : دو عبارت را می پذیرد و مجموعه ای از جانشین هایی که عبارات را به صورت یکتا می سازند برمی گرداند .

---

[۱] overlap

[۲] Generalized Modus Ponens (GMP)

WorksAt(x,PowerPlant),WorksAt(Homer,PowerPlant) ، {x/Homer} را تولیـد مـی نماید .

WorksAt(x,PowerPlant),WorksAt(Homer,y) ، {x/Homer,y/PowerPlant} را تولیـد می نماید .

WorksAt(x,PowerPlant),WorksAt(FatherOf(Bart),y) ، {x/FatherOf(Bart),y/PowerPlant} را تولید می کند .

WorksAt(x,PowerPlant),WorksAt(Homer,x) غلط می باشد – x نمی تواند هـم بـه Homer اشاره کند و هم به PowerPlant .

آخرین عبارت یک اشتباه است ؛ فقط به خاطر این که ما خواسته ایم x را در هـر دو عبـارت اسـتفاده نماییم . ما می توانیم x را با یک متغیّر منحصر به فرد (x_{21}) که یک عبارت است جایگزین نماییم که این کـار استانداردسازی نامیده می شود . اگر بیش از یک جانشین وجود داشته باشد که بتوانند دو عبـارت کـه بـه نظـر می رسد یکسان هستند را بسازند چطور ؟ ؛ مثلا :

Sibling(Bart,x) و Sibling(y,z) می تواند {Bart/y,x/z} یا {x/Bart,y/Bart,z/Bart} را تولید نماید که اوّلین یکسان سازی ، کلّی تر از دومی است – چون الزام های کم تری را سبب می شـود . پـس ، ***مـا می خواهیم کلّی ترین یکسان کننده ها را در زمانی که استنتاج را انجـام مـی دهـیم پیـدا نماییم .***

الگوریتم یکسان سازی

برای یگانه کردن دو عبارت ، به صورت بازگشتی جلو بروید . اگر هر دو عبارت تک منغیری است ، یک یگانه سازی که متغیّر را به یک ثابت متصل می کند را پیدا نمایید . در غیر این صورت یکانگی درون واژه ی اوّل را ، که توسط باقی مانده ی هر عبارت دنبال شده است را صدا بزنید .

مثال :

$Sibling(x, Bart) \wedge PlaysSax(x)$ و Sibling(Lisa,y) که ما می توانیم Sibling(x,Bart) و Sibling(Lisa,y) را با {x/Lisa,y/Bart} یگانه نماییم . پردازه ی پیدا کردن مجموعه ی کاملی از جانشین ها ، یک پردازه ی جستجو می باشد که وضعیت ، لیستی از جانشین ها می باشد و تابع جانشین ، لیستی از یکسان سازی های بالقوه و لیست های جانشین جدید است . پس در مورد یکسان سازی داریم :

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

در جایی که $p_i'\theta = p_i\theta$ برای همه ی i ها برقرار باشد . به عنوان مثال ، اگر King(John) ، p_1' می باشد . King(x) ، p_1 می باشد . Greedy(y) ، p_2' می باشد . Greedy(x) ، p_2 می باشد . {x/John,y/John} ، Θ می باشد و q ، Evil(x) می باشد و Evil(John) ، qΘ می باشد .

روش طرز عمل کلّی ، با شرط های نامحدود پایگاه دانش استفاده می شود (**دقیقا** با یک لفظ مثبت) . تمام متغیّرها ، کمّیّت های عمومی را به خود می گیرند .

اثبات روش طرز عمل کلّی – باید نشان بدهیم که : $p_1', ..., p_n', (p_1 \wedge ... \wedge p_n \Rightarrow q) \models q\theta$

مشروط بر این که برای همه ی i ها ی موجود ، $p_i'\theta = p_i\theta$ باشد .

اثبات : با استفاده از نمونه سازی عمومی (UI) برای هر شرط نامحدود p داریم $p \models p\theta$

$1.(p_1 \wedge ... \wedge p_n \Rightarrow q) \models (p_1 \wedge ... \wedge p_n \Rightarrow q)\theta = (p_1\theta \wedge ... \wedge p_n\theta \Rightarrow q\theta)$

$2. p_1', ..., p_n' \models p_1' \wedge ... \wedge p_n' \models p_1'\theta \wedge ... \wedge p_n'\theta$

۳. از ۱ و ۲ نتیجه می شود که $q\theta$ توسط روش های معمولی دنبال می شود .

مثال : پایگاه دانش – قانون می گوید که این یک جنایت است که یک آمریکایی بـه ملّـت هـای دشمن ، اسلحه بفروشد . کشور نونو [۱] یک دشمن آمریکا می باشد و دارای تعدادی موشک مـی باشـد و تمـام این موشک ها توسط کلنل وست [۲] که یک آمریکایی است به این کشور فروخته شده است . ثابـت کنیـد کـه کلنل وست یک جنایت کار می باشد .

... برای یک آمریکایی این جنایت است که به ملل دشمن اسلحه بفروشد :

$$American(x) \wedge Weapon(y) \wedge Sells(x, y, z) \wedge Hostile(z) \Rightarrow Cri\min al(x)$$

کشور نونو دارای تعدادی موشک می باشد ... . توجه کنید :

$$\exists x Owns\,(Nono\,, x) \wedge Missle\,(x) : Missle\,(M_1), Owns\,(Nono\,, M_1)$$

... تمام این موشک ها توسط کلنل وست فروخته شده اند .

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

موشک ها اسلحه هستند : $\text{Weapon(x)} \Rightarrow \text{Missle(x)}$

$$\text{Enemy(x,America)} \Rightarrow \text{Hostile(x)}$$

وست آمریکایی است ... American(West).

---

[۱] Nono

[۲] Colonel West

کشور نونو یک دشمن آمریکا است : Enemy(Nono,America)

زنجیره ی مستقیم

ایده ی اساسی این است که ، با واقعیّات و قوانین شروع کنید (استنباط ها) و دایما روش طرز عمل کلّی (GMP) را به کار ببرید تا واقعیّات جدید بتوانند به وجود بیایند .

الگوریتم زنجیره ی مستقیم (به بیان دیگر)

تابع FOL-FC-Ask(KB,α) یک تعویض یا غلط را بر می گرداند

مادامی که new خالی است کارهای زیر را انجام بده

$new \leftarrow \{\}$

برای هر عبارت r در KB موارد زیر را انجام بده

$(p_1 \wedge ... \wedge p_n \Rightarrow q) \leftarrow S\tan dardize - Apart(r)$

برای هر θ در $p_1^{'},...,p_n^{'}$ برای برخی از $(p_1 \wedge ... \wedge p_n)\theta = (p_1^{'} \wedge ... \wedge p_n^{'})\theta$ در KB

$q^{'} \leftarrow Subst(\theta, q)$

در صورتی که $q^{'}$ تغییر نام یک عبارت موجود در KB یا new نبود کارهای زیر را انجام بده

$q^{'}$ را به new اضافه کن

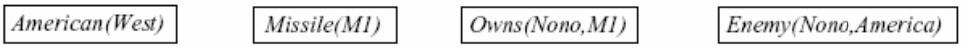
$\phi \leftarrow Unify(q^{'}, \alpha)$

در صورتی که ϕ ، fail نمی باشد ، ϕ رابرگردان

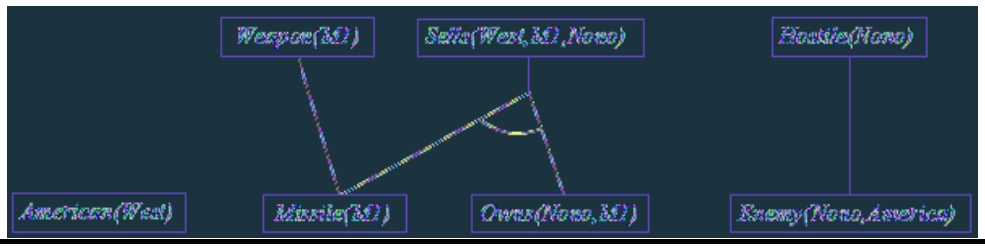
new را به KB اضافه کن

false را برگردان

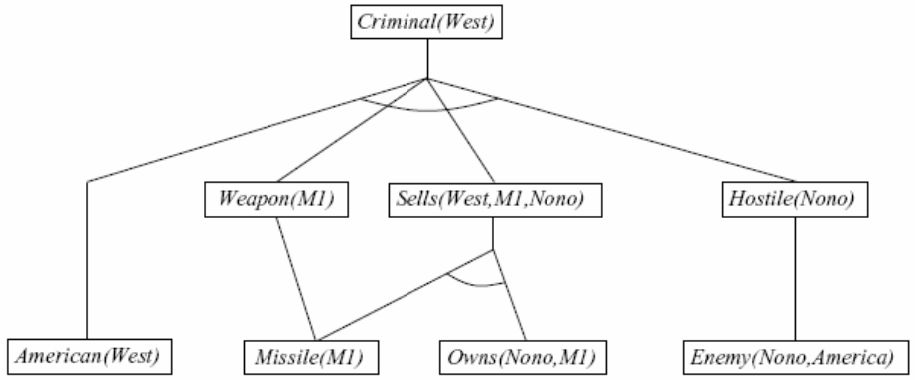
اثبات به روش زنجیره ی مستقیم



گام بعدی :



گام بعدی :

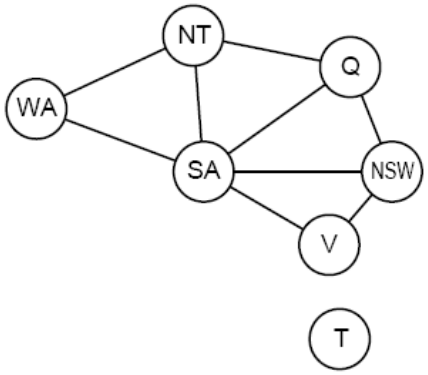


خصوصیّات زنجیره ی مستقیم

زنجیره ی مستقیم ، بی عیب می باشد ، چون که از روش طرز عمل کلّی (GMP) استفاده می کند . برای شرط های محدود ، زنجیره ی مستقیم ، کامل می باشد . خیلی شبیه به جستجوی اوّل – بهترین عمل می کند . این الگوریتم اساسی ، خیلی هم کارآمد نیست ، چون : اوّل ، همه ی یکی کننده های ممکن را پیدا می کند ولی گران است . دوم ، هر قانون ، در هر تکرار ، مجددا چک می شود و سوم ، این که واقعیّاتی که به هدف منجر نمی شوند ، تولید می شوند .

زنجیره ی مستقیم به طور گسترده ای در ***پایگاه داده های استقرایی یا استنتاجی یا قیاسی*** [۱] استفاده می شود . یک سیستم پایگاه داده ی قیاسی ، سیستمی پایگاه داده ای است که می تواند قیاس را براساس قوانین و واقعیّات نگهداری شده پایگاه دانش قیاسی انجام دهد . سیستم های پایگاه دانش قیاسی ، اساسا به قوانین و واقعیّات رسیدگی می کنند ، از یک زبان اعلانی (نظیر پرولوگ) برای مشخص کردن این قوانین و واقعیّات استفاده نمایید و از یک موتور استنتاج که می تواند قوانین و واقعیّات جدید را از آن هایی که ارایه شده اند قیاس نماید استفاده نمایید . [۲]

مثال تطبیق دهنده ی سخت [۳]



[۱] deductive databases

[۲] en.wikipedia.org/wiki/Deductive_database

[۳] hard matching

$$Diff(wa,nt) \wedge Diff(wa,sa) \wedge Diff(nt,q) Diff(nt,sa) \wedge Diff(q,nsw) \wedge Diff(q,sa) \wedge Diff(nsw,v) \wedge Diff(nsw,sa) \wedge Diff(v,sa) \Rightarrow Colorable()$$

Diff(Red,Blue) Diff(Red,Green) Diff(Green,Red)
Diff(Green,Blue) Diff(Blue,Red) Diff(Blue,Green)

Colorable ( ) در صورتی که CSP راه حلی برای CSP ها ی شامل 3SAT در مورد خاص داشته باشد استنتاج می شود ، با وجودی که تطبیق به صورت NP-hard می باشد .

زنجیره ی معکوس

این روش ، به طرف معکوس ، از هدف به طرف واقعیّاتی که باید برای هدف نشان داده شوند حرکت می کند . از روش طرز عمل کلّی ، به صورت معکوس برای تمرکز دادن جستجو روی پیدا کردن شرط هایی که می توانند منجر به هدف شوند استفاده می کند ؛ همچنین از شرط های نامحدود استفاده می نماید و جستجو به روش اوّل – عمق جلو می رود .

الگوریتم زنجیره ی معکوس

تابع FOL-BC-Ask(KB,goals,θ) مجموعه ای از تعویض ها را بر می گرداند

ورودی ها : KB ، که یک پایگاه دانش می باشد

goals ، لیستی از پیوندهای شکل دهنده ی یک صف (θ قبلا به کار گرفته شده است)

θ ، تعویض جاری ، به صورت اوّلیه ، تعویض خالی می باشد { }

متغیّرهای محلی : answers ، که مجموعه ای از تعویض ها می باشد ، و به صورت اوّلیه تهی ({ }) می باشد .

در صورتی که goals تهی است مجموعه ی $\{\theta\}$ را برگردان

$$q^{'} \leftarrow Subst(\theta, First(goals))$$

برای هر عبارت r موجود در KB

در جایی که $S\tan dardize - Apart(r) = (p_1 \wedge ... \wedge p_n \Rightarrow q)$ و $\theta^{'} \leftarrow Unify(q, q^{'})$ برقرار می باشد .

$$new - goals \leftarrow [p_1, ..., p_n \mid \mathrm{Re}st(goals)]$$

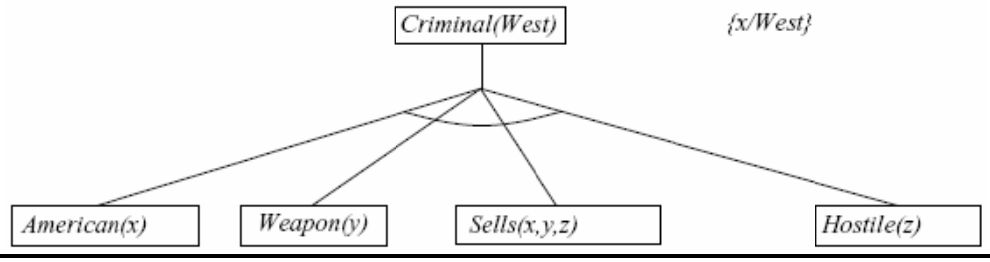
$$answers \leftarrow FOL - BC - Ask(KB, new - goals, Compose(\theta^{'}, \theta)) \cup answers$$

answers را برگردان

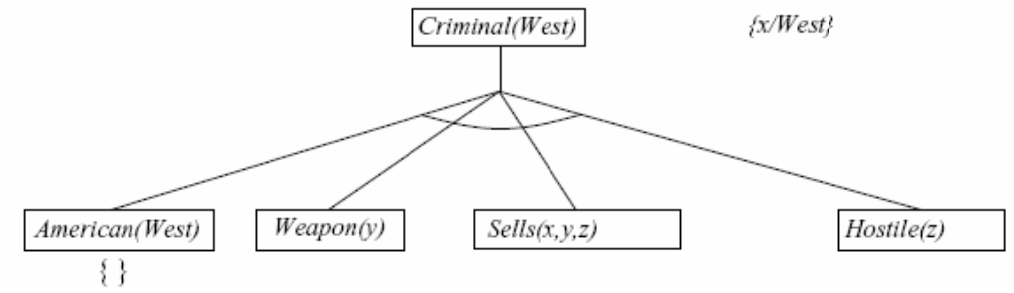
مثال زنجیره ی معکوس

Criminal(West)

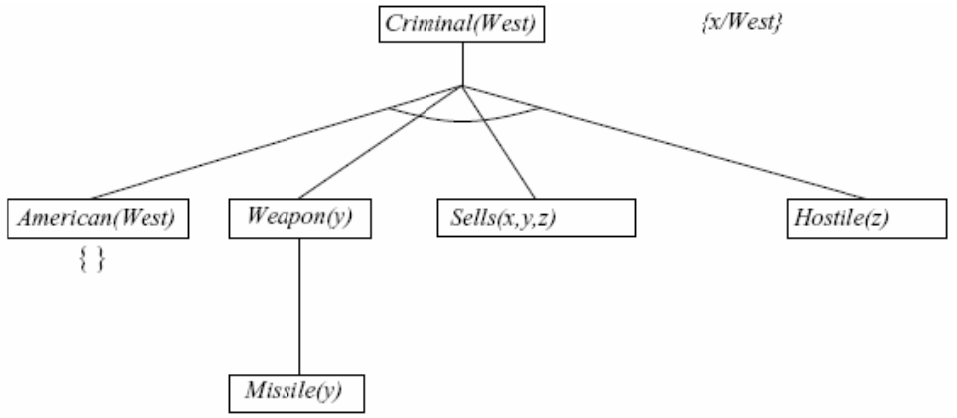
گام بعدی :



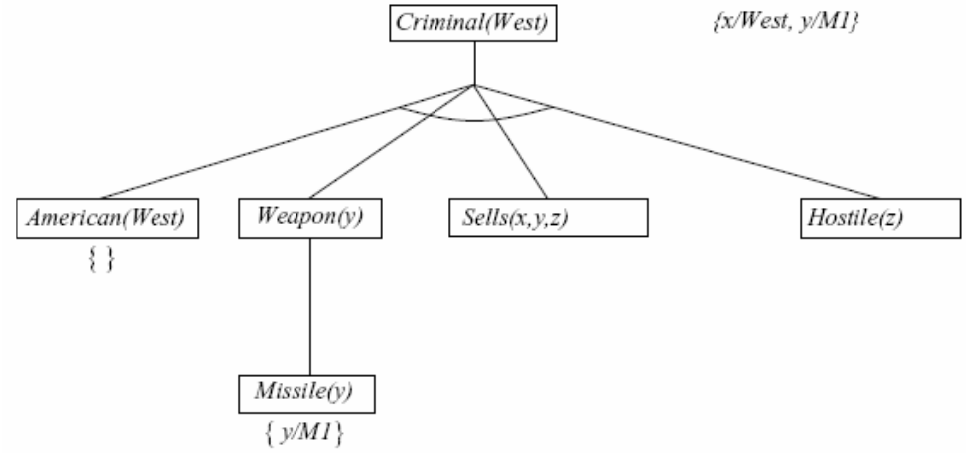
گام بعدی :



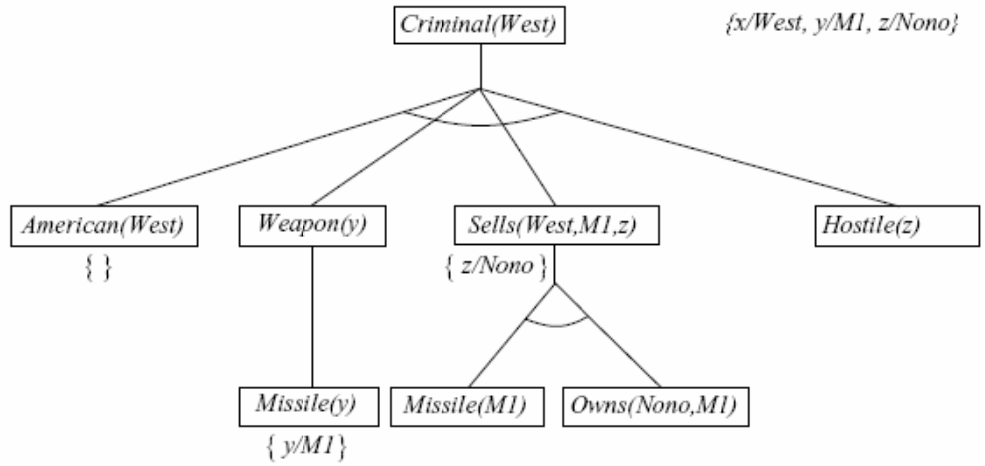
گام بعدی :



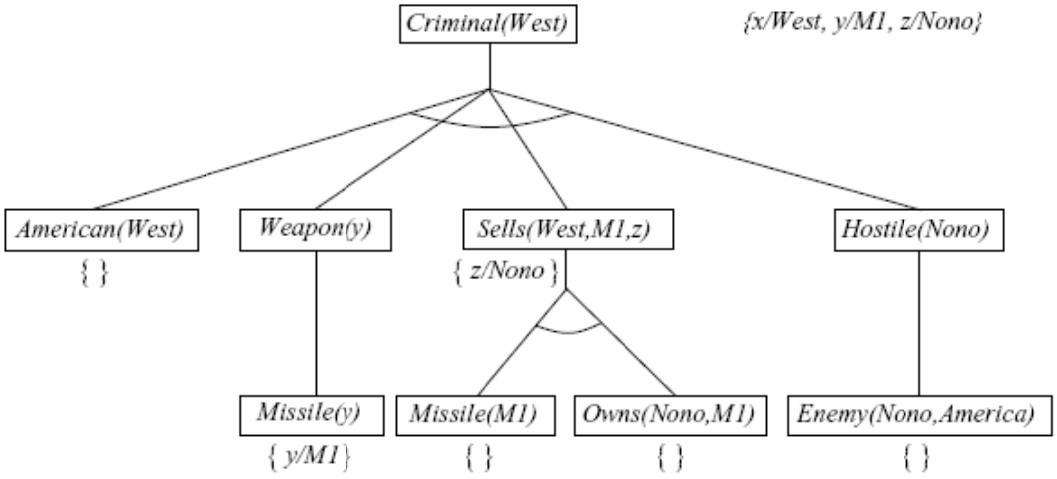
گام بعدی :



گام بعدی :



گام بعدی :



خصوصیّات زنجیره ی معکوس

زنجیره ی معکوس از روش اوّل – عمق استفاده می کند ؛ این به این معنی است که از وضعیت های تکرار شده رنج می برد . همچنین این روش کامل نمی باشد و می تواند برای سیستم های برمبنای پرس و جو [۱] خیلی موثر باشد در ضمن ، این روش برای برنامه نویسی منطقی [۲] به طور گسترده ای استفاده می شود .

تحلیل : خلاصه

تحلیل را در منطق گزاره ای به یاد بیاورید : $(A \vee C) \wedge (\neg A \vee B) \Rightarrow (B \vee C)$ ، تحلیل در منطق مرتبه ی اوّل به طریقی مشابه عمل می کند . *توجه کنید که لازم است عبارات به فرم CNF باشند .*

تبدیل به صورت نرمال CNF – هر کس که همه ی حیوانات را دوست می دارد کسی او را دوست می دارد :

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

۱. استلزام [۳] ها را حذف کنید

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

۲. $\neg$ را به درون حرکت دهید :

---

[۱] query – based systems

[۲] logic programming

[3] implication

در کامپیوتر ، عملی منطقی با ساختار
IF-THEN
اگر A,B درست باشند ، آن گاه تابع AND این دو هم درست است

$\neg\forall x, p \equiv \exists x \neg p, \neg\exists x, p \equiv \forall x \neg p$:

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

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

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

۳. متغیّرهای استاندارد : هر کمّیّت ، باید از یک متغیّر جدا استفاده نماید ؛

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

۴. عبارت های وجودی را اسکلمایز نمایید

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

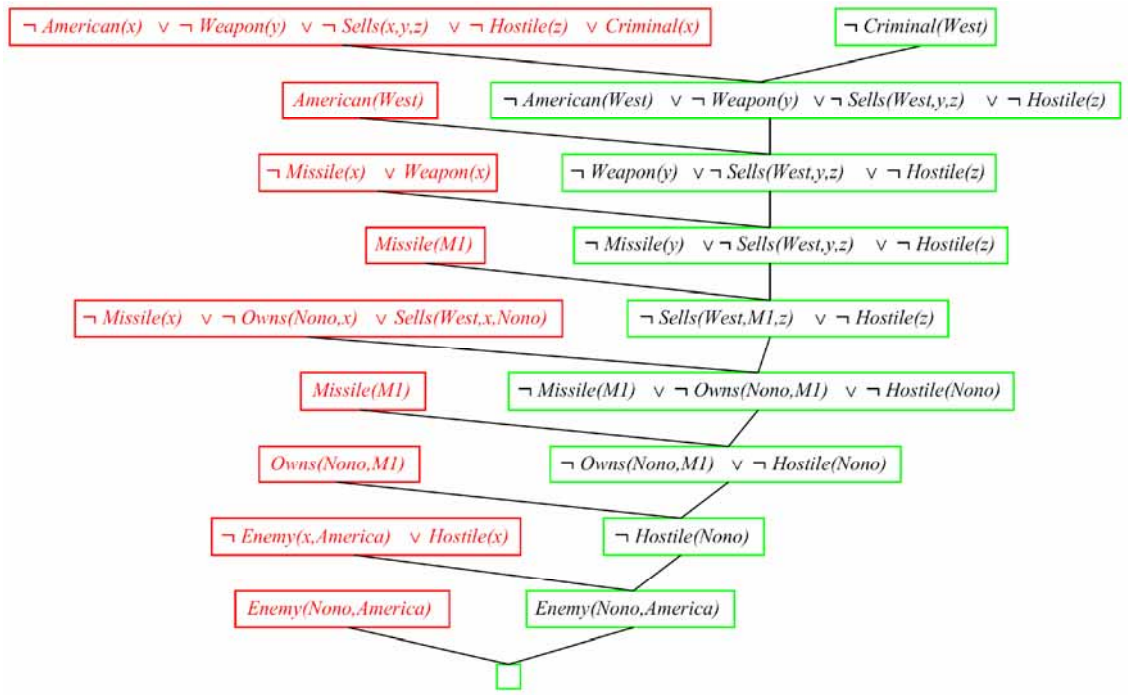
۵. سورهای عمومی را رها نمایید :

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

۶. $\wedge$ را روی $\vee$ توزیع نمایید :

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

اثبات تحلیل : شرط های نامحدود


¬ American(x) ∨ ¬ Weapon(y) ∨ ¬ Sells(x,y,z) ∨ ¬ Hostile(z) ∨ Criminal(x)
¬ Criminal(West)
American(West)
¬ American(West) ∨ ¬ Weapon(y) ∨ ¬ Sells(West,y,z) ∨ ¬ Hostile(z)
¬ Missile(x) ∨ Weapon(x)
¬ Weapon(y) ∨ ¬ Sells(West,y,z) ∨ ¬ Hostile(z)
Missile(M1)
¬ Missile(y) ∨ ¬ Sells(West,y,z) ∨ ¬ Hostile(z)
¬ Missile(x) ∨ ¬ Owns(Nono,x) ∨ Sells(West,x,Nono)
¬ Sells(West,M1,z) ∨ ¬ Hostile(z)
Missile(M1)
¬ Missile(M1) ∨ ¬ Owns(Nono,M1) ∨ ¬ Hostile(Nono)
Owns(Nono,M1)
¬ Owns(Nono,M1) ∨ ¬ Hostile(Nono)
¬ Enemy(x,America) ∨ Hostile(x)
¬ Hostile(Nono)
Enemy(Nono,America)
Enemy(Nono,America)

مترجم: سهراب جلوه گر

ویرایش دوّم ، بهار ۱۳۸۸

فصل سیزدهم

نامعلومی [۱]

(عدم قطعیّت)

[۱] uncertainty یا ابهام یا عدم قطعیّت

مترجم: سهراب جلوه گر
ویرایش دوّم ، بهار ۱۳۸۸

ریوس مطالب

- نامعلومی
- احتمال [۱]
- ظاهر و معناها
- استدلال (استنتاج)
- استقلال [۲] و قانون بیز [۳]

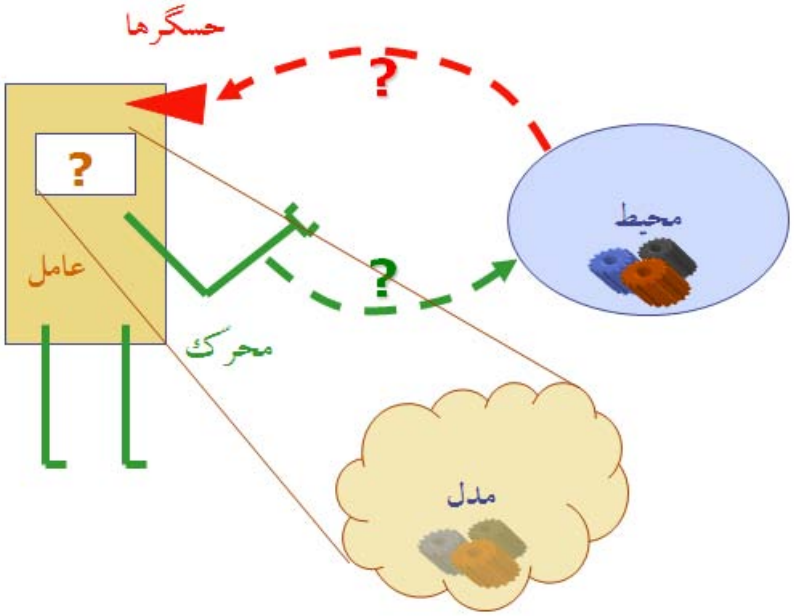
[۱] Probability
[۲] Independence
[۳] Bayes' Rule

مترجم: سهراب جلوه گر
ویرایش دوّم ، بهار ۱۳۸۸

نامعلومی

در تعدادی از محیط های جالب عامل ها ، نا معلوم بودن یا عدم قطعیّت دارای یک نقش زیاد می باشد ؛ به عبارت دیگر ؛ عملکردها ممکن است دارای اثرات نامعلومی باشند ؛ مثل ، پرتاب یک پیکان به هدف . عامل ها ممکن است وضعیّت درست جهان را ندانند . در ضمن ارتباط های میان واقعیّت ها ممکن است به صورت معلوم نباشند . مثلا گاهی اوقات ، در زمانی که هوا ابری می باشد ، باران می بارد ؛ به عبارت دیگر ، هر وقت که هوا ابری باشد ، باران نمی بارد ! . عامل های هوشمند باید نامعلوم بودن یا ابهام را بررسی نمایند .

عامل نامعلوم [۱]



انواع نامعلومی

[۱] uncertain agent

مترجم: سهراب جلوه گر
ویرایش دوّم ، بهار ۱۳۸۸

نامعلومی در دانش قبلی

به عنوان مثال ، بعضی از دلایل بیماری ناشناخته اند و در دانش یک عامل دستیار پزشک بیان نشده اند .

نامعلومی در عملکردها

مثلا ، کارهایی که باید برای انجام آن ها پیش شرط هایی برقرار باشند ؛ شاید هم این پیش شرط ها طولانی هم باشند ؛ به عنوان مثال ، برای رانندگی کردن با اتومبیل خودم در صبح باید اتومبیل من در طول شب دزدیده نشده باشد ؛ تایر یا تایرهای آن پنچر نباشند ؛ باید در مخزن (باک) آن ، گاز وجود داشته باشد ؛ باتری آن از کار نیفتاده باشد ؛ باید موتور آن روشن شود ؛ کلیدهای آن را گم نکرده باشم ؛ کامیون مسیر رانندگی را مسدود نکرده باشد ؛ نباید به طور ناگهانی ، من کور یا فلج شوم و ... . البتّه نمی توانیم همه ی موارد را لیست کنیم .

نامعلومی در ادراک

مثلا ممکن است حسگرها اطّلاعات کامل یا دقیق را در مورد جهان برنگردانند ؛ مثلا ممکن است یک ربات مسیر درست را پیدا نکند :