بازنمایی دانش

بازنمایی دانش (نمایش دانش)

در ادامه، تکنیک‌های مختلف در کدکردن دانش در یک سیستم خبره را مورد بحث، قرارمی‌دهیم .

ارائه دانش = کد کردن دانش

تفاوت برنامه‌های متداول با سیستم‌های خبره از دید دانش

- برنامه‌های متداول روی داده‌ها عمل پردازش انجام می‌دهند (داده ها را پردازش‌می‌کنند)
- سیستم‌های خبره روی دانش عمل پرداش انجام‌می‌دهند (دانش را پردازش می‌کنند)

دلایل نمایش دانش در سیستم‌های خبره

۱. پوسته سیستم های خبره طوری طراحی می شوند که برای روش خاصی از نمایش دانش مثل قواعد یا منطق مناسبند .

۲. روش ارائه دانش بر روی سرعت ، توسعه و کارایی و نگهداری سیستم موثر است.

Domain Expert : اشاره به فرد خبره و متخصص انسانی دارد .یعنی فردی که می تواند یک مساله را به گونه ای‌حل نماید که دیگران قادر به آن نیستند . منظور از Domain این است که یک محدوده خاصی داشته باشد مثل مکانیک ، مهندسی ، پزشکی و...

| Expert |= Expert – Non Expert = knowledge

نکته: تفاوت Expert و Non Expert در دانش است ؛ به عبارت دیگر، تفاوت یک فرد خبره و یک فرد عادی تنها در داشتن دانش است .

پرسش: دانش چیست ؟

دانش به فهم و دانایی اشاره‌می‌کند . از نظر فلسفی، دانش مفهومی است که همه می‌دانند ولی قادر به تعریف آن نیستند.

چیزی که واضح است این است که هرچه دانش بیشتر باشد توانایی بیشتر است

Knowledge is Power

توانا بود هرکه دانا بود

و سیستم خبره قوی تر، دارای دانش بیشتری است .

با توجه به دانش، یک تعریف از سیستم خبره به صورت زیر است :

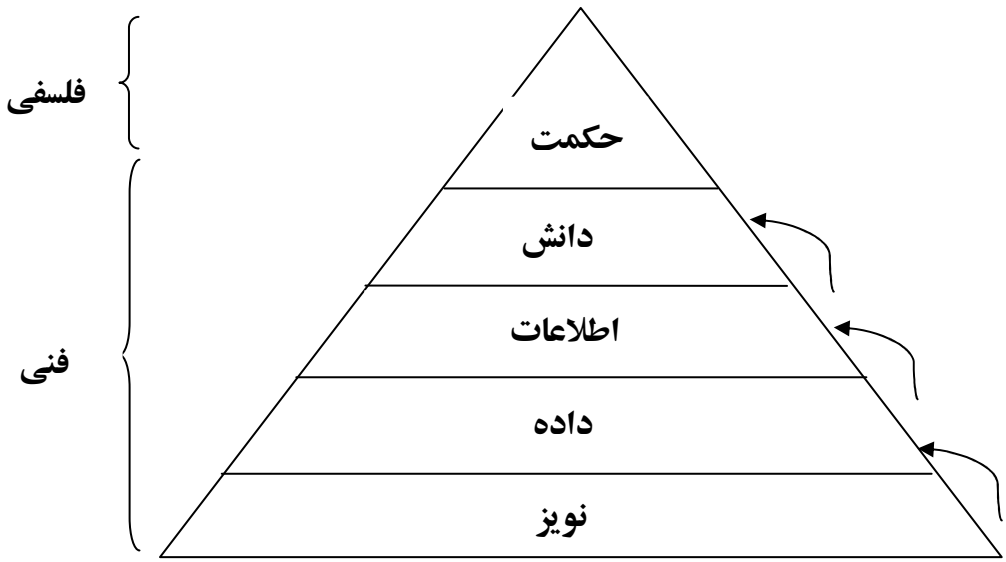
Expert System = knowledge + Reasoning

سیسستم خبره = دانش + استدلال

همانطور که یک برنامه، ترکیبی از الگوریتم‌ها و ساختمان داده هاست .

Program = Data structure + Algorithms

هرم سلسله مراتبی دانش



شکل۵-۱: هرم دانش

- در پایین ترین سطح نویز قرار دارد سپس داده و بعد ازآن اطلاعات و سپس دانش و حکمت قرار دارد.
- داده؛ نویزهای پاک شده است و قسمت‌های غیرمفید حذف شده‌اند.
- اطلاعات یعنی داده‌هایی است که پردازش‌شده است.
- دانش یعنی اطلاعاتی که پردازش‌شده است
- حکمت(خرد) دانشی است که هدف و ارزش دارد و به زندگی و فلسفه آن، مربوط است.

نکته: اگر داده‌ای، هیچ مفهومی نداشته باشد نویز است. اما اگر بدانیم این سمبل‌ها و نمادها برای چه منظوری است داده‌می‌باشد .به طور مثال اگر شخصی به زبانی سخن بگوید که برای شما ناآشناست، سخنان آن شخص برای شما نویز محسوب‌می‌شودچون هیچ مفهومی برای شما ندارد .

نویز : داده‌ای که هیچ مفهومی ندارد .

داده : حداقل می‌دانیم برای چه منظوری استفاده می‌شود.

دانش : فهم در یک موضوع خاص است.

مثال:

فرض کنید یک رشته ۲۴ عددی داریم که از هر چیزی ممکن است تشکیل شده باشد . اگر ندانیم این رشته به چه منظور استفاده می شود نویز است ولی اگر بدانیم برای بیان قیمت ، نمره و ... استفاده شود ، داده است .

برای این که از داده به اطلاعات برسیم به الگوریتم نیاز داریم. پس تبدیل داده به اطلاعات توسط یک الگوریتم صورت می‌گیرد. مثلا اگر رشته ۲۴ عددی را ۸ تا ۸ تا جداکنیم هر ۸ کاراکتر نمایانگر یک کد اسکی است و می‌توانیم به کد اسکی برگردانیم .

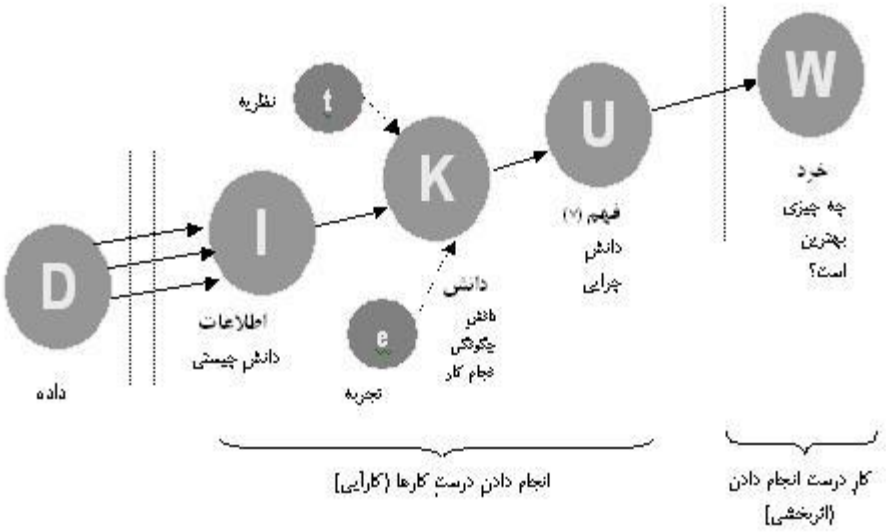
یا تصورکنید این عدد به + Gold تبدیل شود که خود یک اطلاع می باشد. پس الگوریتم مبدل داده به اطلاعات است که + Gold نمایانگر افزایش قیمت طلا باشد . اگر از این اطلاع در جایی استفاده‌شود تبدیل به دانش می شود مثلا "چون قیمت طلا در حال افزایش است پس طلا خریداری شود " یک دانش را ایجاد کرده است.

دانش به کار گیری اطلاعات است

فوق دانش meta Knowledge، دانش چگونگی استفاده از دانش و تجربه است .

نکته: قابل‌توجه‌است که تمامی این تعاریف، نسبی هستند مثلا ممکن است چیزی که در محیطی دانش، درنظرگرفته‌می‌شود در محیط دیگر داده‌باشد .

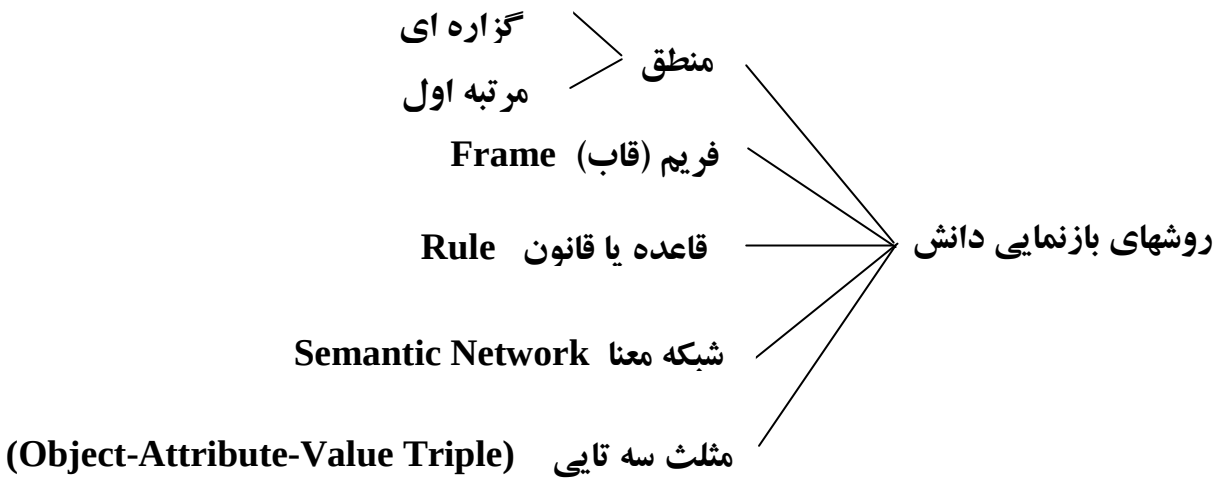
تمرین: شکل ۵-۲ را تفسیر کنید.



شکل۵-۲: روند تبدیل از داده تا خرد

بازنمایی دانش

نمایش دانش یا بازنمایی دانش روشهایی است که برای کد کردن دانش در پایگاه دانش سیستم خبره استفاده می شود. بعد از آنکه دانش از یک فرد خبره در دامنه خوش – متمرکز (well-focused domain) بدست آمد، باستی در یک سیستم خبره کد شود. برای کدکردن دانش نیاز به ساختار بندی دانش است به گونه‌ای که سیستم، قادر به حل مساله به روشی مشابه فرد خبره باشد.



نکته: روش‌های دیگری علاوه بر روش‌های بالا، برای بازنمایی دانش وجود دارد.

در علوم شناختی ، حقایق (fact) بلوک‌های سازنده در سازماندهی دانش انسان است .

حقایق فهم و دانایی ما از یک رویداد یا مساله است که قالبی از دانش توصیفی است .

در هوش مصنوعی و سیستم خبره حقایق و واقعیت‌ها بخشی از قالبها Frames و شبکه های معنا و قوانین است که اغلب به یک گزاره اشاره دارد .

گزاره : عبارتی است درست یا نادرست

مثال :" هوا بارانی است " یک گزاره است که ممکن است درست یا نادرست باشد

روش مثلث سه تایی OAV :

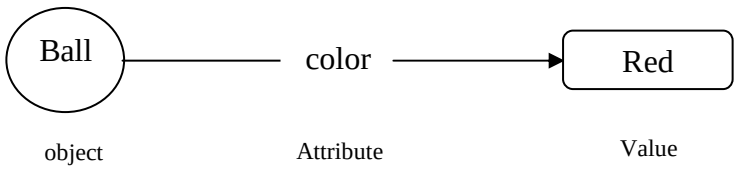
OAV یک نوع گزاره پیچیده تر است که یک عبارت را به سه بخش تقسیم می‌کند .

OAV وظیفه دارد به گزاره ، ساختاریا بخش بندی بدهد این بخش بندی تشکیل شده است از Object و Attribute و Value . مزیت OAV این است که با عبارتهای کمتری می‌توان دانش بیشتری را ارائه‌نمود.

نکته: یک ویژگی OAV این است که امکان مدیریت بخشهای متفاوت گزاره دارد که سبب‌می‌شود OAV جذاب‌تر از بازنمایی گزاره‌ای ساده باشد

The ball's color is red

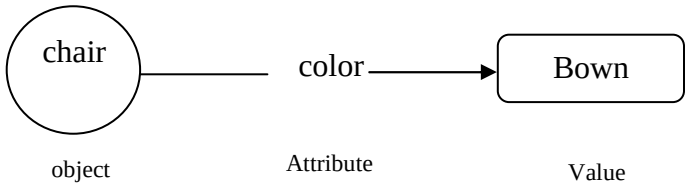
حال در این گزاره سه بخش بالا را معین می کنیم:



شکل۵-۳: سه تایی OVA برای توپ

و یا در گزاره دیگری داریم :

Chair color is brown



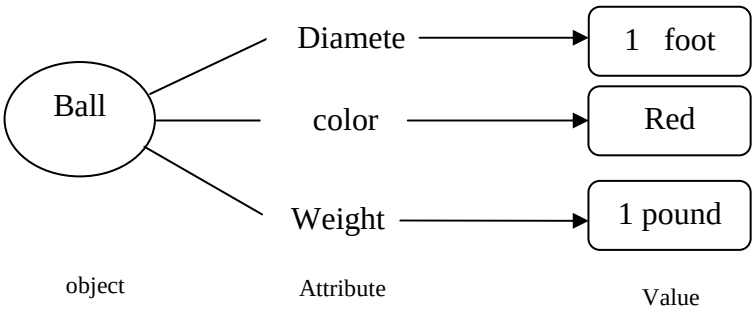
شکل۵-۴: سه تایی OVA برای صندلی

درمورد گزاره It's raining داریم

شکل۵-۵: سه تایی OVA برای آب و هوا

یک شی با چندین ویژگی Object with multiple attribute :

در شکل زیر یک Object با چندین Attribute نمایش‌داده‌شده‌است که نمایانگر این است که یک شی ممکن است چندین ویژگی داشته باشد.



شکل۵-۶: سه تایی OVA با چندین ویژگی

به این دانش ، دانش سه تایی گفته می‌شود زیرا به سه بخش تبدیل می‌شود و مدیریت و سازماندهی آن ساده‌تر است ولی بیانگر یک گزاره است.

نکته ۱: یک Object میتواند مفهومی فیزیکی(ماشین ، توپ و ...) یا انتزاعی (عشق ، دوستی و...) باشد .

نکته ۲: Value می‌تواند یک مقدار داشته باشد (Single value facts) و یا چند مقدار داشته باشد (Multi Value facts)

فشارسنج مثالی است ازsingle value facts که سه مقدار دارد ولی از سه مقدار درحال افزایش ، ثابت ، در حال کاهش در هر لحظه تنها یک مقدار می‌گیرد. یعنی در هر لحظه فشارسنج تنها یکی از مقادیر در حال افزایش یا در حال کاهش و یا ثابت، نشان‌می‌دهد . در فرمها این خاصیت با Radio Boxها نمایش‌داده‌می‌شود.

سطح تحصیلات درجات مختلفی دارد که نشانگر Multi value facts است به طور مثال اگر درجات تحصیلی را به دیپلم ، لیسانس ، فوق لیسانس و دکترا تقسیم کنیم . اگر شخصی دارای مدرک فوق لیسانس باشد مسلما دیپلم و لیسانس هم دارد پس چند Value را در بر می‌گیرد . پس در Multi value facts یک Attribute بیش از یک Value دارد. در فرم ها این خاصیت با Check box ها نمایش داده‌می‌شود.

تمرین: تحقیق کنید علاوه بر روش‌های بانمایی معرفی شده، چه روش‌های دیگری وجود دارد.

انواع دانش

دانش به انواع گوناگونی تقسیم می شود :

دانش رویه ای Procedural knowledge :

دانش رویه ای دانشی است که چگونگی حل یک مساله را توضیح می دهد و به بررسی قوانین ، استراتژی ها ، لیست اهداف و پروسیجرها می‌پردازد.

دانش توصیفی Declarative:

دانش توصیفی، دانشی است که در مورد بخش‌های شناخته‌شده‌ی مساله توضیح‌می‌دهد که شامل یک سری عبارتهای دارای ارزش درست یا نادرست است.

فرا دانش Meta knowledge :

فرا دانش دانشی است که چگونگی استفاده از دانش را توضیح می‌دهد . این دانش، کمک می‌کند چه مسائلی در حل یک مساله مناسب‌تر هستند.

دانش هیورستیک Heuristic Knowledge :

دانش هیورستیک که گاهی قواعد سرانگشتی نامیده می‌شود فرآیند استدلال را هدایت می‌کند . این دانش تجربه‌ای است و فرد خبره از طریق تجربه بدست‌آورده است .

دانش ساختاری Structural knowledge:

دانش ساختاری دانشی است ساختار یک دانش را توصیف می کند . ای‌ن نوع دانش مدل کلی ذهنی یک خبره در مورد یک مساله تشریح می کند که این مدل ذهنی شامل مفاهیم ، زیر مفاهیم و... است .

در جدول ۵-۱ نمونه‌هایی برای انواع دانش آورده شده است :

جدول ۵-۱: انواع دانش

Types of knowledge	
Procedural	Rules قوانین Strategies استراتژی ها Agendas اهداف Procedures رویه ها
Declarative	Concepts مفاهیم Objects اشیا Facts حقایق
Meta	Knowledge about the Other types of knowledge And how to use them دانشی که معین کند از کدام دانش استفاده شود وچگونه از آنها استفاده شود
Heuristic	Rules of thumb قواعد سر انگشتی
Structural	Rule sets مجموعه قواعد Concept Relationship ارتباطات مفهومی Concept to object Relationship مفاهیم ارتباط اشیا

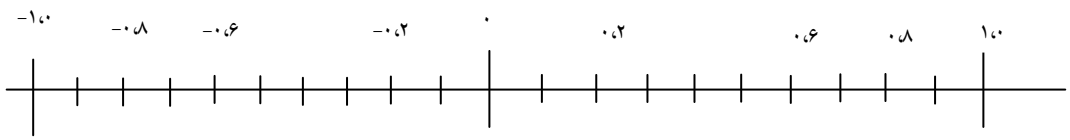
حقایق قطعی Certain Facts و حقایق غیر قطعی Uncertain Facts :

دنیای اطراف ما ، یک دنیای سیاه و سفید نیست ، و رویدادهای اطراف ما تمام درست یا نادرست نیستند. برای درک بهتر حقایق قطعی و غیر قطعی به تعریف ضریب قطعیت نیاز داریم.

ضریب قطعیت Certainty Factor (CF) :

ضریب قطعیت بیانگر درجه باور ما نسبت به یک حقیقت می باشد . اولین بار ضریب قطعیت در سیستم پزشکیMYCIN مورد استفاده قرار گرفت.

شکل ۵-۷ مربوط به CF سیستم MYCIN می‌باشد در این شکل CF مقادیری از ۱،۰ - تا ۱،۰ + دارد.



شکل۵-۷: ضریب قطعیت

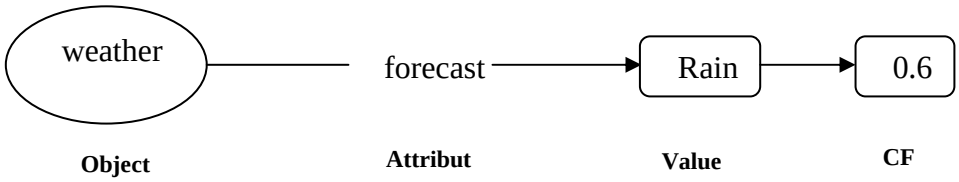
که در این نمودار اعداد نمایانگر مقادیر زیر می باشد :

۱،۰- : کاملا نادرست ۰،۸- : تقریبا نادرست ۰،۶- :احتمالا نا درست

۰،۲- تا ۰،۲ : شناخته شده نیست

۰،۶ : احتمالا درست ۰،۸ : تقریبا درست ۱،۰ : کاملا درست

در مثال زیر همان گزاره It is raining با CF بیان می شود .



شکل۵-۸: سه تایی OVA همراه با ضریب قطعیت

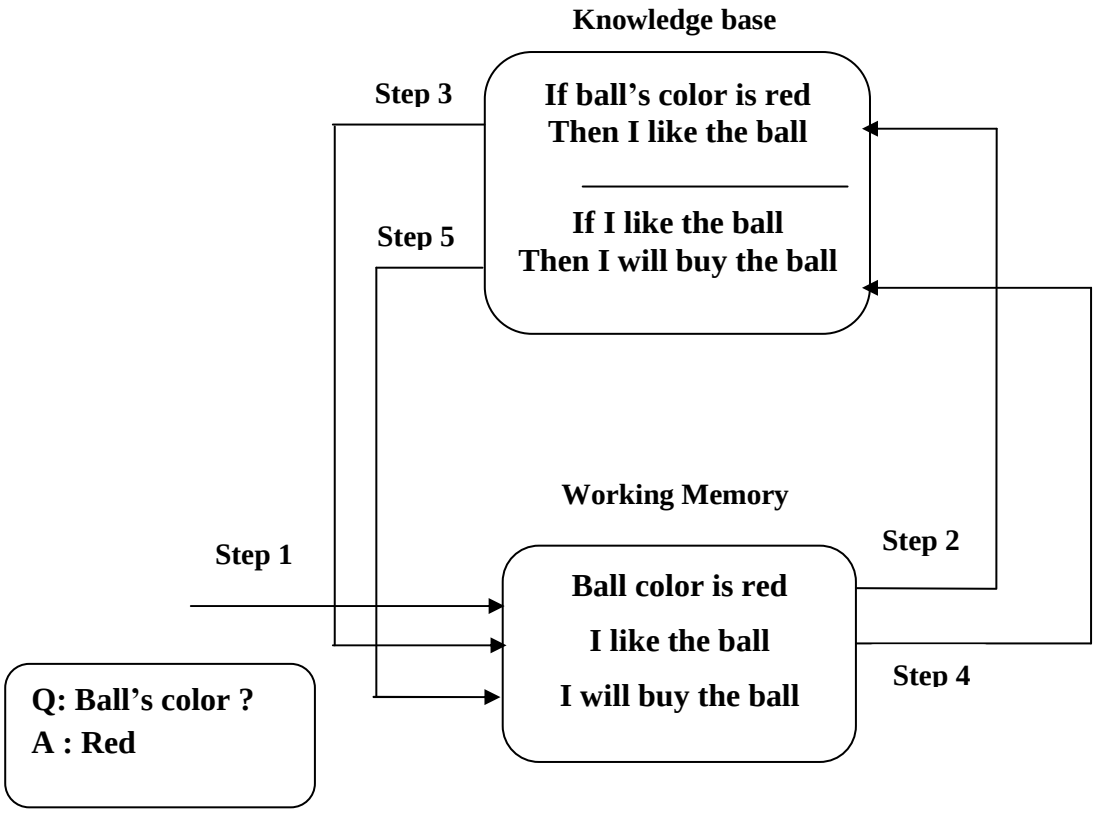
قانون Rule :

قانون ساختاری از دانش است که اطلاعات شناخته شده‌ای را به اطلاعات شناخته‌شده دیگری مرتبط می‌سازد که می‌تواند این ساختار مورد استنتاج و نتیجه‌گیری قرار گیرد .

به طور مثال :

If the ball's color is red
Then I like the ball

اگر توپ قرمز باشد من این توپ را دوست دارم .



شکل۵-۹: روند استنتاج در مثالی ساده

شکل ۵-۹، روند استناج در این مثال نشان می دهد. در سوال می پرسد توپ چه رنگی است ؟ پاسخ می دهد : قرمز

وقتی قرمز به سیستم داده شد قانون شماره ۱ فعال می شود و نتیجه می گیرد توپ رادوست دارد و سپس قسمت پایین فعال می شود و نتیجه می گیرد توپ را خواهد خرید .

انواع قانون

قانون‌های رابطه‌ای Relationship Rules

IF the battery is dead
Then the car will not start

اگر باطری مشکل داشته باشد ماشین روشن نخواهد شد .
در این نوع قانون رابطه ای بین علت و معلول وجود دارد .

قانون های توصیه‌ای Recommendation Rules

IF the car will not start
THEN take a cab

اگر ماشین روشن نمی شود . کاپوت ماشین را بالا بزن .
قانون های توصیه ای حاوی یک پیشنهاد یا توصیه هستند .

قانون‌های دستوری Directive Rules

IF the car will not start
AND the fuel system is ok
THEN check out the electrical system

اگر ماشین روشن نمی شود و سیستم سوخت مشکلی ندارد سیستم برق را بررسی کن.
این قوانین به یک جمله امری ختم می شود .

قانون‌های راهبردی Strategy Rules

IF the car will not start
THEN first check out the fuel system
THEN check out the electrical system

اگر ماشین روشن نمی‌شود ابتدا سیستم سوخت را چک کن و سپس سیستم برق را چک کن .
قوانین راهبردی مراحل انجام کار را بیان می کند .

قانون‌های تجربی Heuristic Rules

IF the car will not start
AND the car is a 1957 Ford
THEN check the float

اگر ماشین روشن نمی شود و مدل ماشین فورد سال ۱۹۵۷ است پس شناور آن را چک کن
قوانین تجربی از روی تجربه های قبلی به انجام کار می پردازد .

در مثال زیر قبل از X علامت سوال می بینید این بدین معنی است که بهتر است در قوانین از متغیرها استفاده شود چون باعث می‌شود قوانین عام تر و کلی تر گردند

IF ?X is Employee
AND ?X Age>65
THEN ?X can reti**re**

Pattern Matching Rules یعنی تطبیق مقدار با متغیر

Meta Rules

Meta Rules قانون یا فرا قانونی است که بیان می کند از چه قانونی استفاده شود.

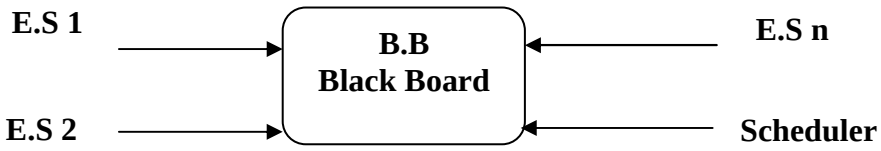
IF the car will not start
AND the electrical System is operating normally
THEN use rules concerning the fuel system

اگر ماشین روشن نمی شود و سیستم برق آن به درستی کار می کند قوانین مربوط به سیستم سوخت را به کار ببر.

Black Board

حل مسائل توزیع شده Distributed problem solving (DPS)

ممکن است در مسائلی نیاز باشد چندین خبره در مورد یک مساله و برای حل آن بیندیشند و به عبارت دیگر چندین خبره با هدف حل نمودن یک مساله بپردازند . چنین مسائلی از نوع مسائل توزیع شده می باشند . و یکی از مکانیزمهای پیاده سازی آنها تخته سیاه Black Board می باشد. بخشی از برنامه مفهوم تخته سیاه را ایجاد می کند که این تخته سیاه وسیله‌ای برای اشتراک گذاری راه حل‌ها می باشد .



شکل۵-۱۰: مفهوم تخته سیاه

در پیاده‌سازی این بخش یک فایل است. به طور مثال یک فایل متنی است و به علت این که سیستم‌های خبره برنامه هستند راه حل‌های مسائل حل شده را داخل این فایل قرار می‌دهند و تمام سیستم های خبره به این فایل دسترسی دارند و می توانند در صورت نیاز از راه حل های دیگران استفاده نمایند. به طور مثال یک پردازش متن چند زبانه یک مساله توزیع شده است ، یعنی یک متن دارای چند زبان باشد و کسی که این متن را تهیه می‌کند می بایست خبرگی در زبان‌های مختلف داشته باشد و یا افراد متفاوتی که در یک مکانیکی کار می کنند هر کدام در بخشی از اتومبیل خبره هستند ، بخش سوخت ، بخش الکتریک ، موتور ، چرخها و ...

مثال:

سیستم خبره عیب یاب کامپیوتر که پس از شناخت عیوب خبره های متفاوتی برای حل مشکل فرا خوانده می شوند ، خبره در زمینه مانیتور ، خبره در زمینه Mother Board و...

مولفه های اصلی سیستم های خبره توزیع شده:

۱- مجموعه ای از سیستم های خبره Commiunity of different Expert System

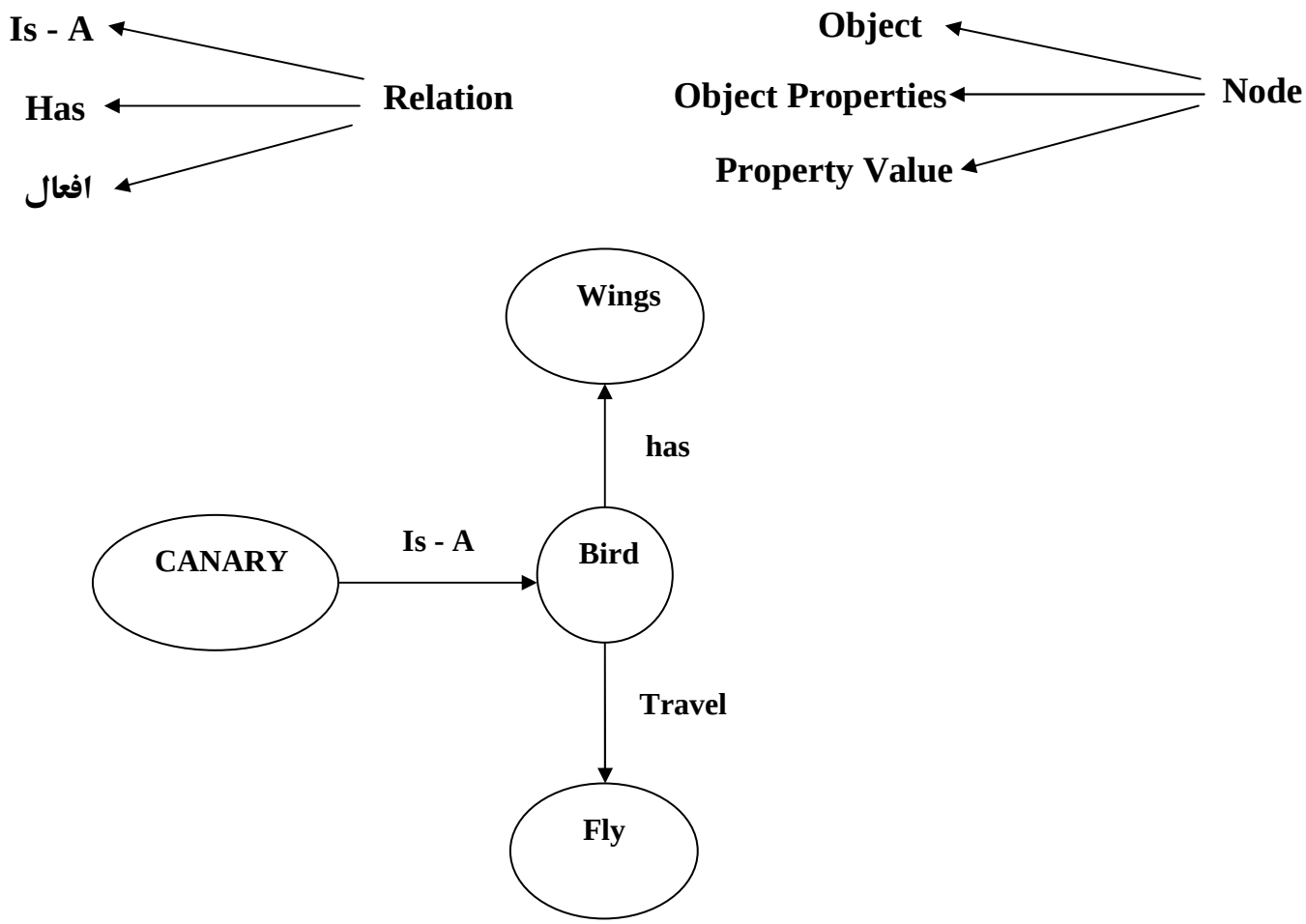
۲- تخته سیاه Black Board

۳- زمان بند Schedular

سیستم تخته سیاه علاوه بر این که کمیته ای از سیستم های خبره است بخشی برای قرار دادن راه حل ها (Black Board) نیز دارند و یک زمانبند نیز برای تقسیم زمان موجود و در اختیار قرار دادن زمان در میان خبره ها دارند

شبکه های معنا Semantic Network

شبکه های معنایی یک روشی برای بازنمایی دانش با استفاده از گراف هستند که از گره ها و یالها تشکیل شده است به طوری که گره ها بیانگر اشیا و یالها نشانگر ارتباط بین اشیا می باشد.



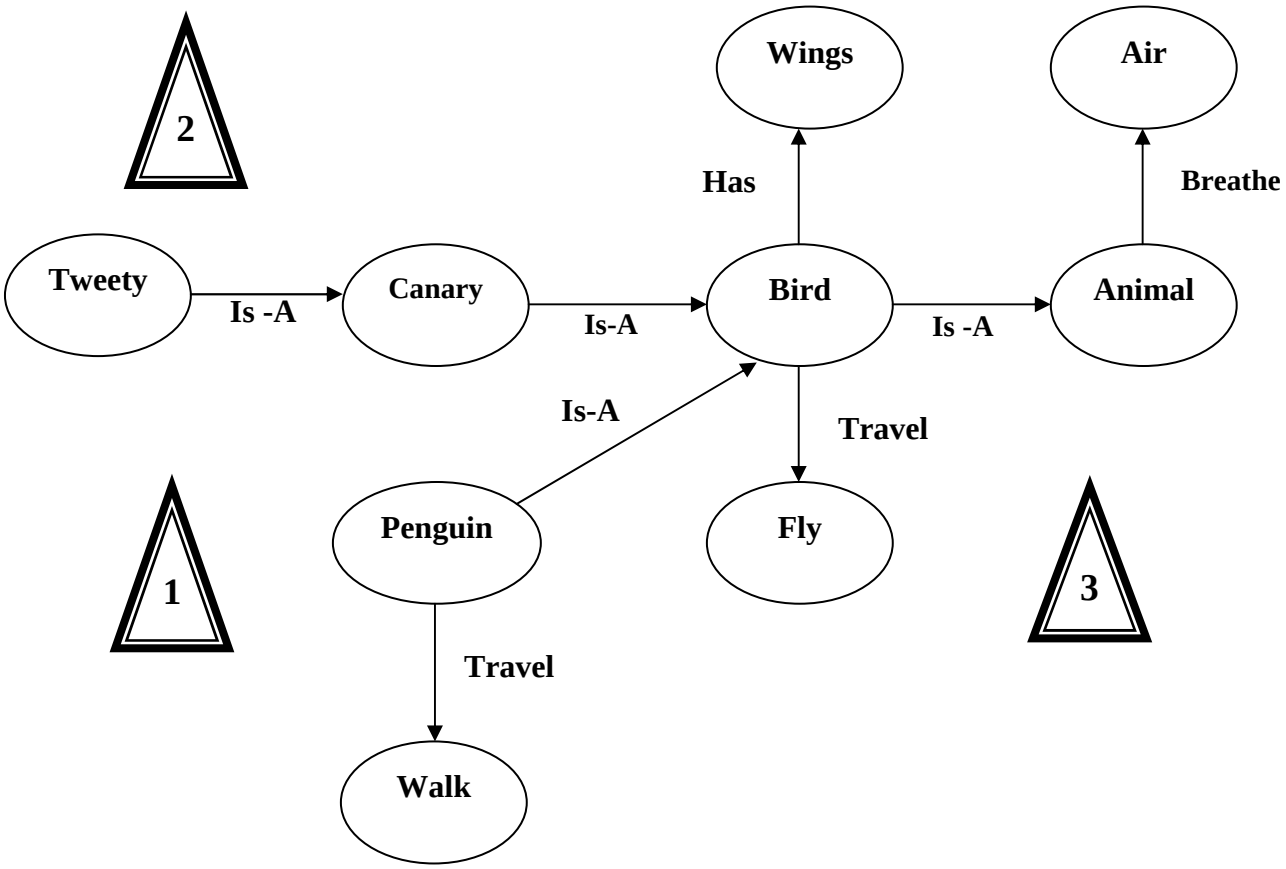
شکل۵-۱۱: شبکه معنا برای پرنده

این مثال در مورد پرنده است. این گراف دارا ی سه صفت داشتن ، جابجا شدن ، و نوع پرنده است. نکته ای که در اینجا قابل مشاهده می باشد این است که جهت در گراف شبکه های معنایی مهم هستند :

- پرنده بال دارد
- پرنده با پرواز جابه جا می شود
- پرنده مورد مطالعه قناری است .

این شبکه می‌تواند توسعه پیدا کند و گسترش با استفاده از سه روش صورت می گیرد.
شبکه معنا با استفاده از ۱- خاص‌تر شدن ،۲- عام‌تر شدن و ۳- مفاهیم مشابه گسسترش می یابد .

مثال



شکل۵-۱۲: گسترش شبکه معنا

در بخش اول(۱) مفاهیم مشابه نمایش داده شده است یعنی قناری و پنگوئن هر دو پرنده هستند.
در بخش دوم (۲) خاص تر نمودن مفاهیم نمایش داده شده قناری یک نوع پرنده است و Tweety یک نمونهخاص از قناری هاست.
در بخش سوم (۳) عام تر شدن مفاهیم نمایش داده شده است پرندگان گونه ای از حیوانات هستند.

وراثت Inheritance

یک ویژگی بسیار مهم در شبکه های معنا وراثت است . وراثت بدین معناست مفهوم یاخاصیتی از گره‌ای به ارث برده شود این ویژگی در شبکه‌های معنا به وسیله $Is - A$ نمایش‌داده‌می‌شود. در مثال بالا تمام ویژگی های Animal را Bird دارد و Animal ها موجوداتی هوازی هستند .

وراثت باعث کاهش حجم پایگاه دانش می شود و سبب می شود هر مفهومی چندین بار تکرار نشود .

در مثال بالا مشاهده می شود سطح و راستای پنگوئن و قناری در گراف یکی است و هر دو با یک یال به پرنده وصل شده است و این نمایانگر این است که بسیاری از مفاهیم در پنگوئن و قناری شبیه هستند.

در پاسخگویی به شبکه های معنا جستجویی روی گراف صورت می گیرد تا پاسخ یافته شود .

مشکل اصلی شبکه های معنا استاندارد نبودن روش بازنمایی آن است که ممکن است سلیقه‌ای عمل شود .

نمونه تکمیل شده شبکه‌های معنا همان نمودارهای کلاس در UML خواهد بود .

روش شبکه‌های معنا جزء روشهای توصیفی است این بدین معنی است که هر چه وجود دارد توصیف می کند در نتیجه زبانهای توصیفی همچون Lisp ,SQL ,Prolog, به خوبی می‌تواند شبکه معنا را کد کند.

مدیریت استثناها Exception Handling

مثال :پنگوئن ی‌ک پرنده است ولی پرواز نمی کندکه این قابلیت توسط exception Handling تصحیح می شود .

ای‌ن مفهوم مشابه مفهوم Override در زبان های شی گرا است .

Override the Inherited Information

فریم یا چهارچوب Frame:

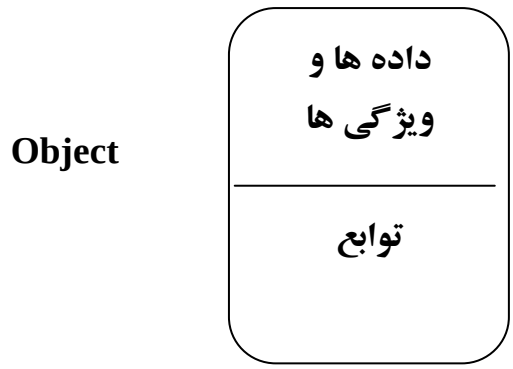
اگر به شبکه های معنا، مفاهیم رویه ها Procedure ، اضافه شود، تشکیل فریم می دهد .

Semantic Network + Procedure = Frame

فریم هم توصیفی است هم رویه ای.

مفهوم فریم ، مشابه مفهوم Object (شیء) است.

Object = Data (Properties) + Procedure

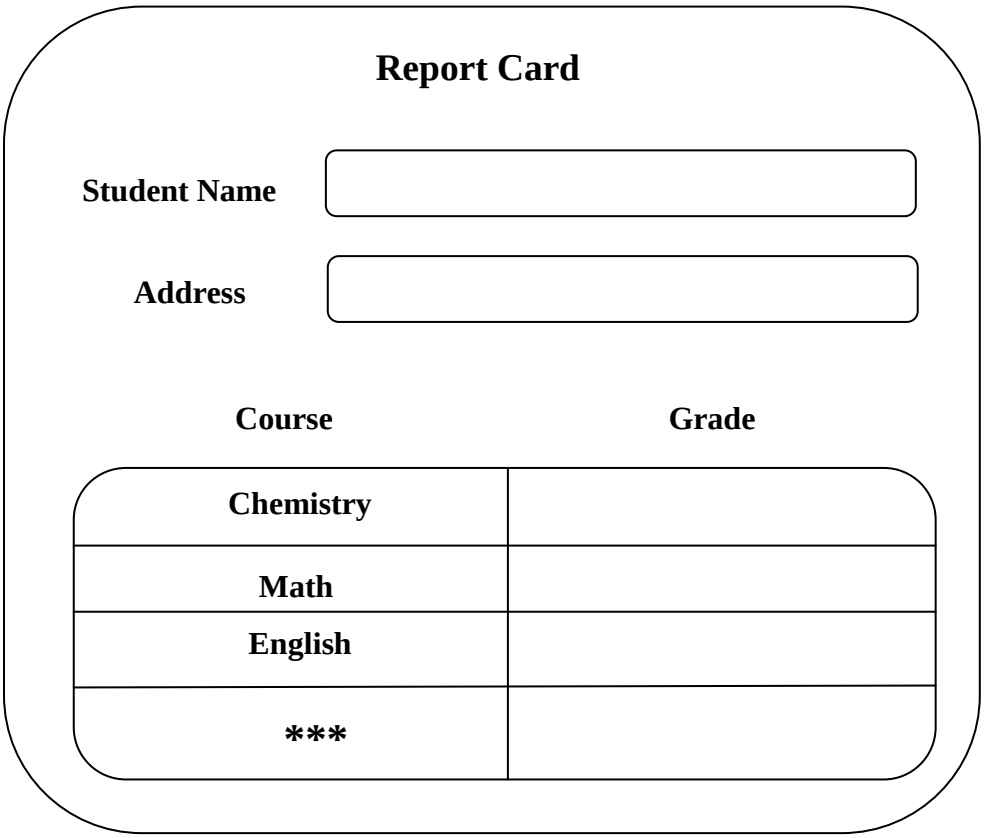


فریم : فریم ساختمان داده ای برای بازنمایی دانش کلیشه ای مشابه مفهوم یا شیء می باشد .

Object در مهندسی نرم افزار، همان فریم در سیستم خبره است .

تفاوت فریم با شبکه‌های معنا: فریم ها علاوه بر مفاهیم، دارای پروسیجرها هم هستند .

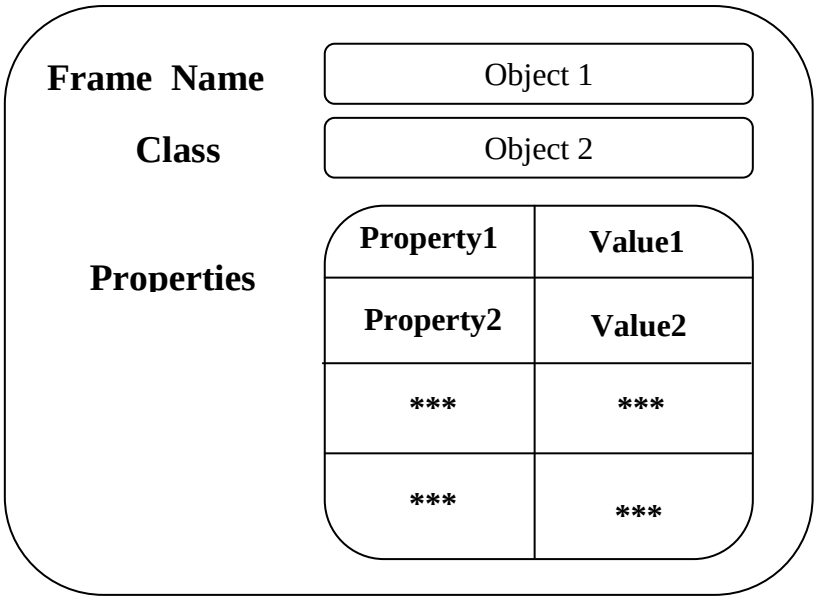
CRC : خیلی از مفاهیم شی را توسط فرم ها و کارت ها نمایش می‌دهند که به این کارت ها CRC گویند. مثال زیر بیانگر کارتی در مورد دانشجوست.



شکل۵-۱۳: نمایش کارت یا فرم

مثال بالا کارنامه ای است که در آن نام و آدرس دانشجو و نام دروس ترم جاری و نمرات آنها را نمایش می دهد .

ساختار کلی قاب : در شکل زیر یک ساختار کلی فریم نمایش داده شده است :



شکل۵-۱۴: ساختار کلی فریم

در یک کارت CRC می بایست نام فریم ، کلاس ، و ویژگی ها و مقادیر آن ویژگی ها آورده شود. فیلد class یک فیلد اختیاری در قاب می باشد که مقدار آن obj2 نام قاب دیگر مرتبط با obj1 است . این رابطه معمولا از نوع Is – A می باشد .

Obj1 Is – A Obj2

Obj1 اطلاعاتی از obj2 به ارث می برد .

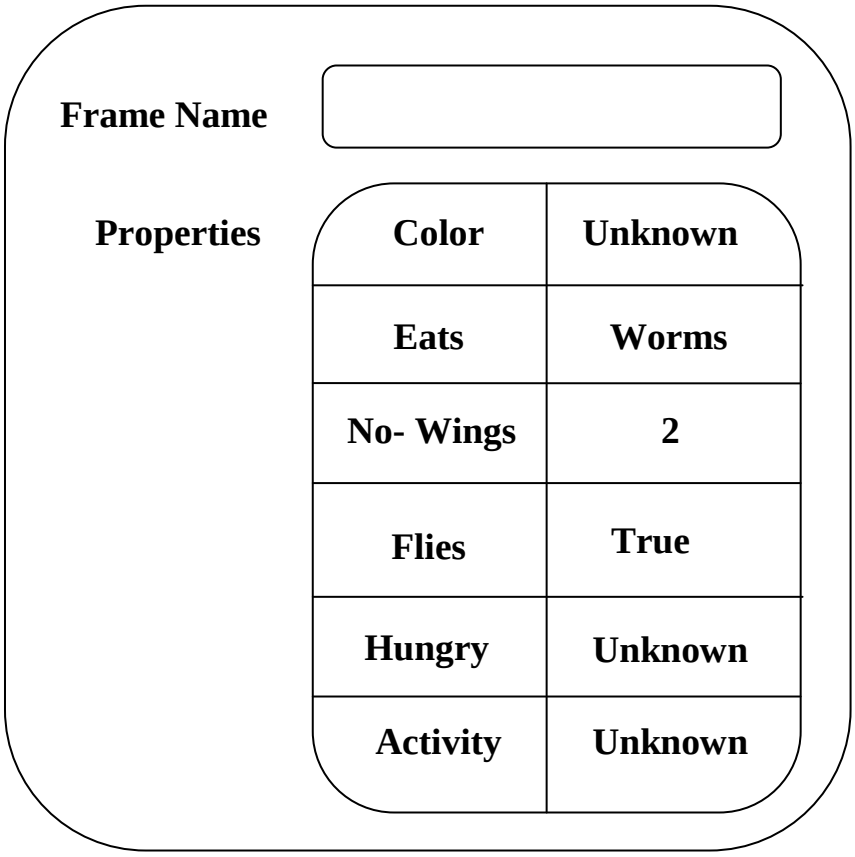
قاب کلاس Class Frame

به طور مثال CRC زیر یک فریم در مورد یک پرنده است .

پس نام فریم پرنده است و ویژگی های آن رنگ ، نوع خوراک ، تعداد بالها ، قابلیت پرواز ، گرسنگی و فعالیت پرنده است .

در این نوع مسائل مشاهده می شود که ویژگی ها دو حالت پویا و غیر پویا (استاتیک و دینامیک) دارند :

- ایستا: رنگ ، تعداد بال
- پویا: گرسنگی و فعالیت



شکل۵-۱۵: فریم پرنده

چون فریم توسعه یافته روش Semantic Network است ویژگی وراثت را حتما دارد .

رفتارهای ارثی Inheriting Behavior

در کنار ارث بری اطلاعات شروع از کلاس ، یک نمونه می تواند رفتارهای مهم را نیز به ارث ببرد . برای این منظور در قاب کلاس یک پروسیجر یا متد، اعمالی که بایستی قاب انجام دهد تعریف می شود مثلا متدی که بگوید اگر پرنده گرسنه باشد چه کاری بایستی انجام شود.

وجوه Facets

Facet کنترلی روی مقادیر ویژگی هاست به بیانی می توان گفت Facet نوعی محدودیت و قیدگذاری است به طور مثال می‌توان مقدار عددی را به رنج خاص محدود نمود، یا تنها قادر به پذیرفتن مقادیری خاص باشد.

Facet به دو نوع تقسیم می شود :

۱. IF Needed در صورت نیاز
۲. IF change در صورت تغییر

اگر Facet ما برای دستور دادن ، به این منظور که یک ویژگی چگونه مقداری بگیردIF Needed است و اگربه منظور این باشد که عملیات مشخصی را در صورت تغییر مقدار تعیین نماید ، یعنی تعیین کند چه کاری انجام دهد اگر مقدار آن تغییر یابد IF Change نامیده‌می‌شود .

IF Needed

If Tweety has less than two wings
THEN Tweety can’t fly

اگر Tweety کمتر از دو بال داشته باشد قادر به پرواز نیست. به صورت Default در نظر گرفته شده است هر پرنده ای قادر به پرواز است حالا اگر در نمونه ای مشاهده شد که کمتر از دو بال دارد پس آن مقدار Can not می شود . در IF needed فقط مقداری عوض می شود به طور مثال اینجا Fly به Not Fly تبدیل شد .

IF Change

IF self : Hungry = True
THEN self : Activity =Eating #self =Eats

اگر ویژگی گرسنگی مثبت باشد، فعالیت صورت‌گرفته، خوردن خواهد بود. در IF Change با توجه به ویژگی خاص عمل خاصی صورت خواهدگرفت یعنی به ازای مثبت یا منفی شدن ویژگی خاصی عمل خاصی صورت خواهد گرفت. Self های قبل از ویژگی ها مشابه کلمه THIS در برنامه‌نویسی شیء‌گراست.

منطق Logic

قدیمی ترین روش بازنمایی دانش در کامپیوتر، منطق است. اما طراحان حرفه‌ای از این روش به ندرت استفاده می‌کنند. روش منطق پایه و اساس بازنمایی دانش است . به طور مثال Prolog زبانی برای بازنمایی منطق است و ارتباط زبان Prolog با سیستم خبره مانند ارتباط زبان اسمبلی با زبانهای برنامه‌سازی است و تمام زبانهای بازنمایی در پایین‌ترین سطح، به منطق می‌رسند همانگونه که زبانهای برنامه‌سازی در پایین‌ترین سطح، به اسمبلی می‌رسند، یعنی مفاهیم پایه را در بر می گیرد .

دو نوع منطق وجود دارد

۱. منطق گزاره ای
۲. منطق مسندی یا مرتبه اول

منطق گزاره‌ای

در منطق گزاره ای هر گزاره با یک نماد یا سمبل بازنمایی می شود و با بررسی جدول‌های درستی به استنتاج می پردازیم.

A= The car will start

در این مثال روشن شدن ماشین با نماد A بازنمایی شده است .

برای استنتاج از جداول درستی استفاده می‌شود. به طور مثال جدول درستی برای عمل عطف $B \wedge A$ به صورت زیر خواهد بود .

A	B	$B \wedge A$
F	**F**	F
F	**T**	F
T	**F**	F
T	**T**	T

جدول درستی عمل فصل $B \vee A$ به صورت زیر خواهد بود

A	B	$B \vee A$
F	**F**	F
F	**T**	T
T	**F**	T
T	**T**	T

و برای هم ارزی داریم $B \equiv A$:

A	B	$B \equiv A$
F	**F**	T
F	**T**	F
T	**F**	F
T	**T**	T

و برای عملگر NOT داریم :

A	NOT A
F	T
T	F

و برای عملگر IMPLIES یا عمگر آنگاه داریم :

$C= A \rightarrow B \vee \ \sim A \equiv B$

A	B	C
F	**F**	T
F	**T**	T
T	**F**	F
T	**T**	T

حساب مسندی

حساب مسندی که گسترش یافته منطق گزاره‌ای است که بازنمایی دقیق تری از دانش امکان‌پذیر می‌نماید. این روش با استفاده از متغیرها و توابع ، پردازش دانش را بهبود می‌بخشد. در منطق مسندی از نماد یا سمبل و ثابت و متغیر و تابع و مسند استفاده می شود .

برای استفاده از Constant آنها را با حروف کوچک نمایش می دهیم مثل اسم افراد .

نمونه ای از مسند:

like (john,mary)

Predicate رابطه بین مفاهیم است در این جا like مسند یا Predicate است و John , mary پارامتر های Predicate هستند و همانطور که مشاهده می کنیدJohn , mary هر دو ثابت هستند زیرا با حروف کوچک شروع شده اند.

متغیر ها را با حروف بزرگ نمایش می‌دهیم

Likes(X,Y)

در این مثال like مسند میباشد و پارامتر های آن متغیر هستند زیرا X,Y را با حروف بزرگ نمایش داده‌ایم .

خروجی تابع یک Object است در حالی که خروجی predicate غلط یا درست False یا True می باشد .

Father(jack) = bob

Mother(judy) = kathy

تمام عملگرهای موجود در منطق گزاره ای قابل استفاده در منطق مسندی هم می باشد. به طور مثال عملگرهایی نظیر آن گاه ، عطف و فصل و ... در منطق مسندی قابل استفاده هستند. با ترکیب Predicate ها با استفاده از عملگر ها قادر خواهیم بود دانش های پیچیده تر را بازنمایی کنیم .

Likes(X,Y) $\wedge$ likes(Z,Y) Implies Not like(X,Z)
OR
$\sim$likes(X,Z)$\rightarrow$Likes(X,Y) $\wedge$ likes(Z,Y)

اگر X,Y یکدیگر را دوست داشته باشند و Z,Y نیز یکدیگر را دوست داشته باشند نمی توان نتیجه گرفت که X,Z نیز یکدیگر را دوست دارند.

مفهوم سور وجودی و سور عمومی:

وقتی سور عمومی به کار می بریم که موردی برای تمامی موارد وجود داشت باشد یا بخواهیم وجود آن را در تمامی موارد بررسی کنیم . سور وجودی زمانی به کار می رود که حداقل یک مورد را بخواهیم بررسی کنیم .

همه mary را دوست دارند.

$\forall$ X likes(X,mary)

حداقل یک نفر وجود دارد که mary را دوست داشته باشد.

$\exists$ likes(X,mary)

نکته: در Prolog تنها سور عمومی وجود دارد و برای مفهوم سور وجودی نیز با ایجاد یک سری محدودیت به سور عمومی تبدیل می کنیم. تبدیل زبان گفتاری و محاوره ای به زبان منطق مرتبه اول کار ساده ای نیست زیرا زبان گفتاری دارای مرتبه n است و مرتبه منطق مسندی، یک است .

Modus Ponens

تا کنون بررسی‌شد که Modus Ponens اطلاعات جدید را از داده مساله اولیه ایجاد می کند. این روش گزینه مناسبی در کاربردهایی است که آموختن و ایجاد اطلاعات از اطلاعات اهمیت دارد با این حال در کاربردهایی نیاز داریم که اطلاعاتی مشخص برای اثبات هدفی جمع آوری شود.

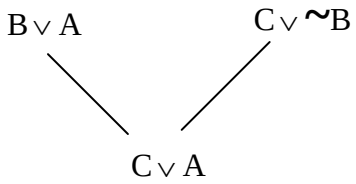
مفهوم Modus Ponens همان قیاس است.

IF A is True
AND A$\rightarrow$B is True
THEN B is True

اگر A درست باشد و A$\rightarrow$B نیز درست باشد در نتیجه B هم درست است.

Resolution

Resolution یک استراتژی استنتاج است که در سیستم های منطقی برای تعیین درستی بیانیه یا حکم (Assertion) استفاده می شود .این روش تلاش می کند تا تئوری یا هدفی مانند گزاره P را با استفاده از مجموعه ای از اصول مرتبط با مساله اثبات نماید. در واقع سعی می کند در عمل ثابت کند که P~ نمی‌تواند درست باشد از این رو اثبات با نقیض یا همان برهان خلف نامیده می‌شود. (proof by Refutation). در روند اثبات با این روش عبارتهای جدید ی ایجاد می شود که Resolvent نامیده می‌شود.



در این مثال $C \vee A$ یک Resolvent است .Resolvent بدست آمده به اصول موجود اضافه می شود و این روند ادامه می یابد و...

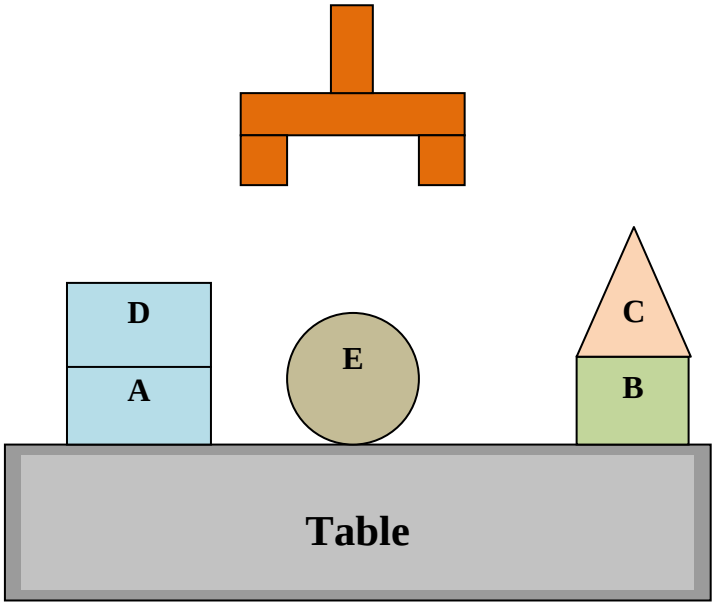
پرسش: روند ایجاد Resolvent ها تا چه زمانی، ادامه‌می‌یابد ؟

تا زمانی که در عبارات تناقض وجود داشته باشد . تناقض زمانی رخ می دهد که دو اصل از نظر منطقی در تضاد با یکدیگر باشند.
B~ و B در مثال بالا. روش Resolution این است که موارد مختلف را گرفته و آنها را ساده نموده تا نتیجه نهایی بدست آید

مقایسه استنتاج Modus Ponens و Resolution

در M.P داده ها از طریق اطلاعات اولیه و فرم آنگاه و جستجو روی این قالب بدست می‌آید. اگر مساله‌ای الزامات شرطی زیادی داشته باشد این فرآیند باید روی تمام اطلاعات ممکن انجام شود در حالیکه Resolution تمرکز روی هدفی که در جهت اثبات آن است ، دارد و در نتیجه تنها الزاماتی که مرتبط با این هدف یا فرض است، در نظر می‌گیرد.

نکته: یکی از کاربردهای منطق در بازنمایی محیط حل مساله برای یک ربات جهت برنامه ریزی یا Planning روبات‌ها می‌باشد . به طور مثال در شکل۵-۱۶، یک روبات، نشان‌داده‌شده‌است.



شکل۵-۱۶: طرح‌ریزی در جابجایی بلوک‌ها

قوانین زیر در مورد ربات بالا در نظر گرفته شده است:

1) Cube(a) cube(b) cube(d) pyramid(c) sphere(e) hand(hand) table(table)
2) On(a,table) on(b,table) on(d,a) on(c,b) on(e,table)
3) Holding(hand,nothing)

این قوانین محیط حل مساله را تشریح می نماید به بیان دیگر یک فریم وضعیت هستند. Cube نماد مکعب‌ها ،pyramid نماد هرم، sphere نماد کره، hand دستگیره یا بازوی ربات،table میز است. تابع On برای قرار گرفتن شی ای روی شیء دیگر استفاده می شود و نشان می‌دهد پارامتر اول روی پارامتر دوم قرار دارد.تابع holding برای نمایش شیء ای که توسط بازوی ربات برداشته شده است به کار می رود. در اینجا بدین معنی است که ربات چیزی را بر نداشته است. در این مثال با تغییر موقعیت هر آیتم توابع و مقادیر تغییر کرده و می‌تواند به برنامه ریزی یا Planning پرداخته و وضعیت را Trace کند.
حال مفهومی تحت عنوان putOn(b,a) تعریف می شود که مفهوم آن این است که b را روی a قرار بده .

$$\text{Hand} - \text{holding}(b) \wedge \text{clear}(a) \rightarrow \text{putOn}(b,a)$$

این قانون بدین معنی است که اگر بازوی روبات، b را برداشته باشد و روی a خالی است، می‌توان b را روی a قرار داد.

تمرین: تحقیق کنید منطق و زبان‌های مبتنی بر منطق، چگونه می‌توانند در برنامه‌ریزی روبات‌ها استفاده شوند.