جزوه درس:

سیستم‌های خبره
Expert systems

کتاب مرجع:

Expert Systems Design and Development

نوشته:

John Durkin

مانی عابدینی

تورج بنی‌رستم

فصل اول – معرفی سیستم های خبره

۱- ماشین هوشمند

خلاصه تاریخچه جستجو برای ماشین هوشمند را در نظر بگیریم:

در اواخر سالهای ۱۷۰۰ و اوایل سالهای ۱۸۰۰، گروهی از آمریکایی‌ها و اروپایی‌ها وسیله‌ای بنام "شطرنج اتوماتیک" ایجاد کردند که توانایی آن را داشت که با یک انسان شطرنج بازی کند.

در سال ۱۸۳۴ چارز بابیج اولین کامپیوتر مکانیکی را بنام "موتور تحلیلی" ایجاد کرد. این ماشین می‌توانست محاسبات ریاضی را انجام داده و خروجی چاپ کند. بابیج توسعه سیستم را تا حد ایجاد یک ماشین قوی که توانایی رقابت شطرنج با انسان را داشته باشد، دیده بود. تا نیمه این قرن ساخت ماشین هوشمند رویایی بود که منوط به پیشرفت تکنولوژی بود. این تکنولوژی با پیدایش کامپیوترهای امروزی به حقیقت پیوست.

کامپیوترهای اولیه، پردازنده‌های سریع داشتند که به آنها این امکان را می‌داد که با داشتن برنامه، عملیات مشخصی را بر طبق الگوریتمی‌مشخص انجام دهند. برنامه‌هایی نوشته شده بود تا معادلات را حل کنند، یا لیستی از داده‌ها را پردازش کنند، یا در پایگاه داده‌ها را برای یافتن اطلاعات مورد نیاز جستجو کنند. اگرچه، به همان خوبی که اطلاعات را پردازش می‌کردند، اما در مورد همان اطلاعاتی توانایی استدلال نداشتند. لذا مسئله استدلال هنوز مختص انسان بود.

نقطه تغییر مسیر زمانی بود که دانشمندان شروع به کد کردن دانش یک مسئله برای کامپیوتر کردند. قوانین، حقایق و ساختارهای یک مسئله به صورت سمبلیک در می‌آمدند، از نمونه های اولیه زبانهای سمبلیک LISP و PROLOG است، که توانایی جستجو روی اطلاعات سمبلیک ارائه شده را دارند.

۲- هوش مصنوعی

در راه رسیدن به ماشین هوشمند و از طریق زبانهای برنامه نویسی سمبلیک، علم هوش مصنوعی شکل گرفت. در سال ۱۹۵۶ گروهی از دانشمندان کامپیوتر در یک کارگاه آموزشی که توسط IBM برگزار شد، شرکت کردند. در این کارگاه در مورد روشی برای آنکه کامپیوتری ایجاد شود که استدلال انسانی را شبیه سازی کند بحث شد. این کنفرانس به عنوان تولد هوش مصنوعی شناخته شد.

تعریف: هوش مصنوعی، دامنه تحقیقاتی است در علم کامپیوتر است که هدف ایجاد کامپیوتری را دنبال می‌کند که بتواند همانند یک انسان استدلال کند.

در معنای خیلی ساده، هوش مصنوعی علم توسعه برنامه‌های کامپیوتری است که چیزی شبیه به هوش انسانی را به نمایش در می‌آورند. این مفهوم هدف را کاملاً مشخص می‌کند بدون آنکه وارد بحث هوشمندی بشویم که کار بسیار سخت می‌شود.

دو معیار هوش انسانی:

- توانایی استدلال
- دانش مرتبط با یک موضوع خاص

از دیدگاه کاربردی، هدف هوش مصنوعی آن بود که کامپیوترها برای انسانها کاراتر شوند:

- ایجاد برنامه‌های کامپیوتری که انسان را در تصمیم‌گیری یاری دهند
- جستجوی هوشمند اطلاعات
- بکارگیری زبان طبیعی در ارتباط با کامپیوتر

۲-۱- رشد هوش مصنوعی

اغلب کارهای جدید صورت گرفته در هوش مصنوعی، جنبه تحقیقات دانشگاهی داشت، مانند ایجاد بازی شطرنج. نمونه‌ها:

برنامه بازی شطرنج ۱۹۵۵ توسط Shanon

برنامه چکرز ۱۹۶۳ توسط Samuel

یکی از بهترین کارهای صورت گرفته در این دوره، ایجاد **حل کننده مسائل عمومی** یا GPS (General Problem Solving) بود. GPS تکنیکی بود که دامنه وسیعی از مسائل را قابل حل می‌کرد. این تکنیک اولین قدم در راه جدا سازی روش‌های حل مسائل از دانش مسائل بود. در GPS، مسئله در غالب state های مختلف بیان می‌شود. برای مثال در بازی شطرنج هر چیدمان مهره‌های شطرنج در صفحه شطرنج یک حالت است. سپس فاصله بین حالت جاری و حالت هدف (مثلاً کیش و مات) محاسبه می‌شود. عملیات مناسب (مانند حرکت مهره در صفحه شطرنج) انتخاب می‌شود تا روی حالت جاری اعمال شود و حالت جدیدی ایجاد کند که امیدواریم به حالت نهایی نزدیک تر شده باشد. این قدمها آنقدر ادامه پیدا می‌کند تا به حالت هدف برسیم.

مشکلات GPS:

- بدست آوردن فاصله بین حالتها و بدست آوردن حرکت مناسب برای مسائل پیچیده مشکل است.
- برای مسائل پیچیده حافظه و سرعت پردازنده کامپیوتر نمی‌تواند جوابگو باشد.

لذا به سرعت فهمیدند که، این تکنیک (GPS) برای مسائل پیچیده مناسب نیست.

در سال ۱۹۷۰ این مفهوم قالب شد که اصولاً هوش مصنوعی در حل مسائل جهان واقعی ناتوان است.

۲-۲- تولد مجدد هوش مصنوعی

از سال ۱۹۷۱ تا این دوره جدید، اوایل ۱۹۸۰ دوره افول هوش مصنوعی بود. پیشرفت تکنولوژی موجب شد تا مجدد هوش مصنوعی ظهوری فعال داشته باشد. اولین جهش در حرکت هوش مصنوعی در برنامه DENDERAL ایجاد شد. این برنامه از سال ۱۹۶۵ مقدماتش در دانشگاه استندفورد به سفارش NASA آغاز شد.

هدف این پروژه آن بود که ناسا بتواند فضا پیمای بدون سرنشینی به ماه بفرستد و توسط یک برنامه کامپیوتری در آن بتواند خاک ماه را آنالیز شیمایی کند و با بدست آوردن داده‌های طیف نگار جرمی‌خاک نوع ساختار مولکولی آن را تشخیص دهد.

روش سنتی برای تشخیص ساختار مولکولی "تولید و تست" بود، ساختارهای مولکولی ممکن ابتدا ایجاد می‌شوند و بعد تست می‌شوند که آیا می‌توانند با داده‌های طیف نگار جرمی مطابقت داشته باشند یا خیر. مشکل آنجا بود که در ابتدا میلیون‌ها ساختار ممکن بود ایجاد شود تا بررسی گردد آیا ساختار مناسبی هست یا خیر. لذا تیم تحقیق بدنبال روشی جهت کنترل تعداد تولید ساختارهای ممکن افتاد.

در این بین تیم تحقیقاتی متوجه شدند که شیمیدان‌ها ماهر و زیرک، از یک روش ابتکاری کمک می‌گیرند و همان ابتدا تعداد زیادی از ساختارها را سریعاً حذف می‌کنند. در نتیجه با استخراج آن دانش و اعمال آنها برنامه‌ای کامپیوتر ایجاد شد که همانند یک فرد خبره عمل می‌کرد. اولین برنامه‌ای بود که موفقیتش مرهون دانش مرتبط با مسئله بود، نه تکنیک جستجوی پیچیده.

کار انجام شده دانشمندان را بر این داشت که رفتار هوشمند آنقدری که به دانش استدلالی، وابسته است به تکنیک های استدلال متکی نیست.

لذا گفتند: "در دانش قدرت نهفته است" . "In the Knowledge lies the power"

از آنجا بود که مفهوم سیستم‌های دانش پایه یا سیستم‌های خبره ظهور کرد.

۳- سیستم‌های خبره

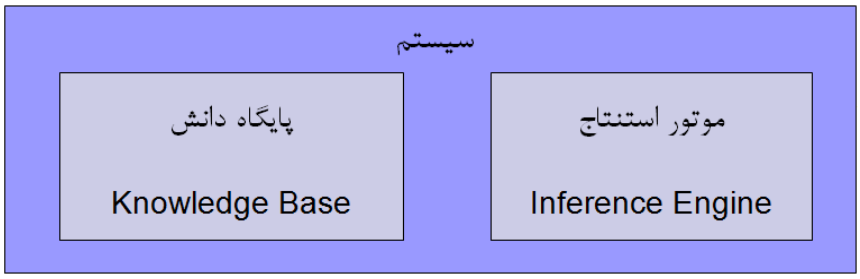
تعریف سیستم‌های خبره: یک برنامه کامپیوتری است و طوری طراحی شده که توانایی یک فرد خبره در حل مسئله مدل می‌کند. دو بخش اصلی در مدل سیستم باید در نظر گرفت:

- دانش فرد خبره
- استدلال

لذا دو ماژول در سیستم دیده می‌شود:

پایگاه دانش

موتور استنتاج



شکل (۱) مدلی ابتدایی از سیستم خبره

۳-۱- سیستم‌های خبره- پایگاه دانش

پایگاه دانش شامل دانش بسیار خاص مربوط به دامنه مسئله است که توسط فرد خبره ارائه می‌شود. شامل حقایق، قوانین، مفاهیم و روابط می‌باشد. برای مثال، ممکن است شامل قوانینی باشد که توسط پزشک متخصص جهت تشخیص بیماریهای خونی ارائه می‌شود، یا دانش برنامه‌ریزی سبد سهام باشد که توسط مشاور سرمایه‌گذاری ارائه می‌شود.

نحوه کد کردن دانش و وارد کردن آن به پایگاه دانش مربوط به ارائه دانش (Knowledge Representation) می‌شود که در ادامه بحث خواهد شد.

۳-۲-سیستم‌های خبره- موتور استنتاج

موتور استنتاج، پردازنده دانش است که مدلی از روش استنتاج فرد می‌باشد.

موتور استنتاج بر اساس اطلاعات فراهم شده برای یک مسئله کار خود را آغاز می‌کند و بر اساس دانش ذخیره شده در پایگاه دانش، یکسری نتایج و پیشنهادات را ارائه خواهد کرد. نحوه طراحی و پیاده‌سازی اینگونه موتورها در بحث تکنیک های استنتاج مورد بررسی قرار خواهد گرفت.

۳-۳- چرا سیستم‌های خبره

افراد خبره منابع ارزشمند یک سازمان هستند. آنها می‌توانند ایده‌های خلاق و ارزشمندی ارائه کنند، مسائل سخت را حل کنند، یا وظایف روزمره را بصورت موثر انجام دهند. همکاری آنها می‌تواند بطور موثری در سودمندی سازمان موثر شود.

اما چرا می‌خواهیم فرد خبره را در یک سیستم خبره مدل کنیم؟

جدول (۱) مقایسه مزایای فرد خبره نسبت به سیستم خبره

فاکتور	فرد خبره	سیستم خبره
زمان در دسترس	روز کاری	همیشه
جغرافیای فعالیت	محلی	در هر کجای ممکن
امنیت	غیر قابل جایگزینی	قابل جایگزینی
فنا پذیری	بله	خیر
کارایی	متغییر	ثابت
سرعت	متغییر	ثابت و معمولاً سریع تر
هزینه	زیاد	قابل تهیه

مانند هر ماشینی، یک سیستم خبره می‌تواند بعد از روز کاری فرد خبره به طور پیوسته و بی وقفه به فعالیت ادامه دهد. مانند هر برنامه کامپیوتری، به ارزانی می‌توانیم سیستم خبره را کپی کرده و به هر کجایی که کمبود فرد خبره احساس می‌شود ارسال کنیم. می‌توان سیستم خبره را به محیط‌های خطرناک فرستاد بدون آنکه نگران آن باشیم که منبع اصلی دانش را به خطر انداخته ایم.

فرد خبره فناپذیر است. بر اثر مرگ، بازنشستگی، تغییر شغل و غیره سازمان از مزایای دانش فرد خبره محروم خواهد شد. اما اگر بتواند در یک سیستم خبره دانش فرد خبره را کسب کند، می‌تواند بطور پیوسته و بدون محدودیت از آن دانش بهره مند شود. همچنین از این سیستم جهت آموزش کارشناسان جدید می‌توان بهره برد.

یک سیستم خبره نتایج پایدارتری نسبت به فرد خبره ارائه می‌کند. ممکن است مشکلات شخصی در کارایی فرد خبره تاثیر گذار باشد. در شرایط بحران ممکن است فرد خبره تحت تاثیر استرس و کمبود زمان بخشی از دانش مهم را از خاطر ببرد. اما سیستم خبره احساساتی نمی‌شود و همیشه کارایی آن

یکسان است. سرعت حل مسئله در فرد خبره متاثر است فاکتورهای زیادی است. در مقام قیاس، سیستم خبره همیشه با سرعت یکنواختی و در برخی موارد با سرعت خیلی بهتری نسبت به فرد خبره می‌تواند مسائل را حل کند. افراد خبره معمولاً گران قیمت هستند، درخواست حقوق بالا و سرویسهای مجانی زیادی دارند. معمولاً افراد خبره کمیاب هستند.

در مقابل سیستم های خبره نسبتا گران هستند. هزینه ایجاد یک سیستم ممکن است در بسیاری موارد بالا باشد اما این هزینه پس از اجرای سیستم به سرعت جبران می‌شود.

۴- جایگزینی افراد خبره

حالتی که یک سیستم خبره برای جایگزینی یک انسان ایجاد می‌شود عواقب نامطلوبی را نیز در بردارد. این مسئله همان تصویری ناخوشایندی را می‌سازد که انسان‌ها سالها پیش با مشاهده پیشرفت انقلاب صنعتی(جایگزینی ماشین به جای انسان) با آن مواجه شدند. هرچند که امکان ایجاد چنین تصوری وجود دارد اما در عمل استفاده از سیستم خبره به جای انسان تاکنون تاثیر نامطلوب نداشته است.

چند دلیل عمده ایجاد سیستم‌های خبره برای جایگزینی انسان عبارتند از:

- قابل دسترس بودن تجربه در زمان و مکان‌های مختلف
- مکانیزه کردن یک کار روزمره که انجام آن نیاز به فرد خبره دارد
- فرد خبره بازنشسته می‌شود یا محل را ترک می‌کند
- فرد خبره گران قیمت است
- خبرگی در محیط‌های خطرناک مورد نیاز است

برای شرح اینکه چگونه برخی سازمانها از سیستم‌های خبره جهت جایگزینی افراد خبره استفاده کرده‌اند مثال‌های زیر آورده می‌شود:

۴-۱-مشاور حفاری

در حفاری‌های شرکت Elf زمانیکه مته حفاری با مشکل برخورد می‌کرد، باید چندین روز و حتی هفته حفاری را متوقف می‌کردند تا فرد خبره به سایت حفاری برسد. هزینه هر روز ۱۰۰/۰۰۰ دلار بود.

علل مشکل تعداد مشخص و ساده‌ای هستند ولی به هرحال فرد خبره بر اساس تجربیات و سنگها و گل ولایی که در طول حفاری بدست آمده، علت مشکل را تشخیص می‌داد. سیستم خبره‌ای نوشته شد که کار فرد خبره را انجام می‌داد و نیاز ضروری به حضور فرد خبره در سایت نبود.

۴-۲- مشاور اجاق خوراک‌پزی

شرکت Campbell دارای اجاق خوراک‌پزی بزرگ و گران قیمتی در کارخانجات مختلف است که در سراسر جهان توزیع شده‌اند. در مواجه با مشکل معمولاً خود پرسنل اقدام می‌کنند ولی در برخی شرایط مجبور می‌شوند منتظر فرد خبره بمانند که ضرر بسیار زیادی به شرکت تحمیل می‌شود. آقای Aldo Cimino فرد متخصصی است که در شرکت با حدود ۴۴ سال سابقه کار می‌کرد و اغلب مشکلات اساسی کارخانجات منوط به کمک ایشان بود. با نزدیک شدن دوره بازنشستگی ایشان شرکت به فکر ذخیره دانش ایشان به فرم یک سیستم خبره افتاد تا هم در رفع نقص اجاقها و هم در آموزش پرسنل جدید کمک بگیرد.

جزوه درس:

سیستم‌های خبره

Expert systems

کتاب مرجع:

Expert Systems Design and Development

نوشته:

John Durkin

مانی عابدینی

تورج بنی‌رستم

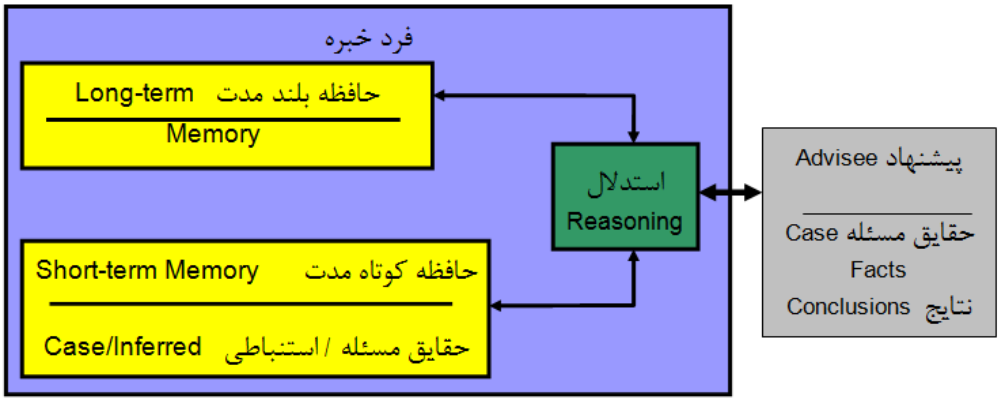
فصل دوم
ویژگی‌های سیستم‌های خبره

۱- ساختار سیستم خبره (Expert System Structure)

یک فرد را زمانی خبره به مسئله‌ای می‌گوییم که او دانش خاصی در مورد آن مسئله داشته باشد. در فیلد سیستم‌های خبره، به اینگونه از دانش Domain Knowledge (DK) یا دانش محیط کاربرد می‌گوییم. کلمه «محیط کاربرد» اشاره دارد که دانش مربوط به یک مسئله مشخص و محدود شده است.

یک فرد خبره دانش محیط کاربرد (DK) خود را در حافظه بلند مدت(Long-Term Memory (LTM خود نگهداری می‌کند. در زمان ارائه پیشنهاد به کسی، فرد خبره ابتدا حقایق (facts) مربوط به مسئله را جمع آوری می‌کند (Case Facts)، و در حافظه کوتاه مدت Short-Term Memory (STM) خود نگهداری می‌کند.

فرد خبره بر مورد مسئله یا مشکل با ترکیب حقایق درون STM و دانش درون LTM استدلال می‌کند. با بکارگیری این فرآیند، فرد خبره به استنباط‌های جدیدی و در نهایت به نتیجه (راه حل) مسئله خواهد رسید.



شکل (۱) نمایش روش استدلال در یک فرد خبره

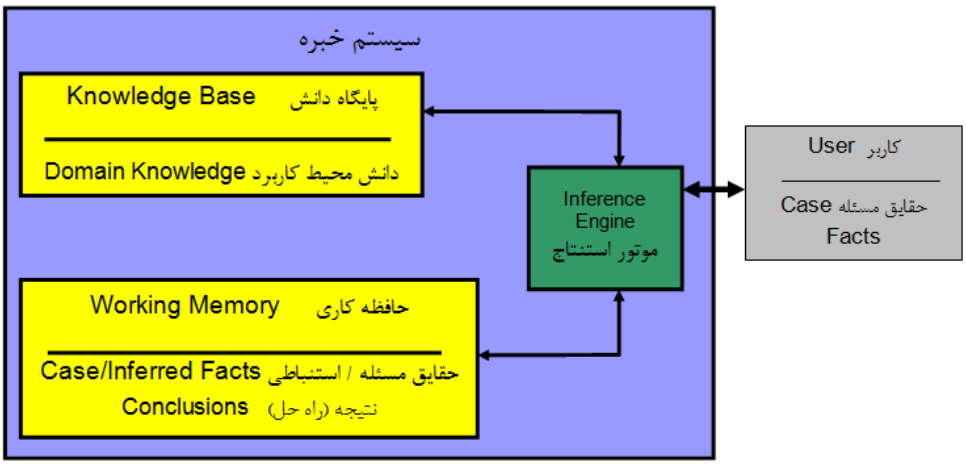
برای فهم بیشتر؛ فرض کنید در ماشین شما مشکلی رخ داده‌است. یک عمل منطقی آن است که وسیله را به یک مکانیک ماشین (فرد خبره) نشان دهیم.

در طی سالیان و تجربیات مختلف مجموعه دانش لازم جهت تشخیص عیب ماشین در حافظه بلند مدت LTM مکانیک نقش بسته است. فرض کنید به مکانیک بگویید که ماشین روشن نمی‌شود، مکانیک اطلاعاتی که شما می‌دهید را در حافظه کوتاه مدت STM خود نگهداری می‌کند و شروع به استدلال بر اساس آنها می‌کند.

با اطلاعات ارائه شده توسط شما و دانش محیط کاربرد DK، مکانیک ممکن است به این نتیجه برسد که مشکل در سیستم برق ماشین است و این یافته خود را در STM قرار داده و به استدلال در مورد مسئله ادامه دهد. در ادامه بررسی ممکن است تست باتری ماشین نشان دهد که مشکل به سبب خرابی باتری است.

در مثال مکانیک، نشان دادیم که نه تنها مکانیک مشکل را رفع می‌کند بلکه فرآیند استدلالی پشت آن را نیز نشان می‌دهد.

سیستم‌های خبره مسائل را با فرآیندی خیلی مشابه با روش فرد خبره حل می‌کنند و از ساختار زیر تشکیل شده‌اند.



شکل (۲) نمایش روش استدلال در یک سیستم خبره

۲- پایگاه دانش (Knowledge Base)

یک سیستم خبره دانش محیط کاربرد یک فرد خبره را در ماژولی بنام پایگاه دانش نگهداری می‌کند. پایگاه دانش همان LTM در مدل فرد خبره است.

تعریف: پایگاه دانش بخشی از یک سیستم خبره است که شامل دانش محیط کاربرد است.

وظیفه یک مهندس دانش Knowledge Engineer، آن است که دانش را از فرد خبره بدست آورده و در پایگاه دانش کد کند. تکنیکهای مختلفی برای این فرآیند وجود دارد که در بخش‌های بعدی اشاره می‌شود.

یکی از روشهایی‌که می‌توان دانش را در یک سیستم خبره ارائه کرد، با کمک قوانین (Rules) است. یک قانون از ساختار اگر/ آنگاه If/Then پیروی می‌کند و بطور منطقی اطلاعاتی که در قسمت اگر می‌آیند، با اطلاعاتی که در قسمت آنگاه ظاهر می‌شوند مرتبط هستند.

- برای مثال:

RULE ۱
IF The car will not start
THEN The problem may be in the electrical system

RULE ۲
IF The Problem may be in the electrical system
AND The Battery voltage is below ۱۰ volts
THEN The fault is a bad battery

۳- حافظه کاری (Working Memory)

حافظه کاری شامل حقایق در مورد مسئله است که در طی اجرا بدست آمده است.

تعریف: حافظه کاری، بخشی از یک سیستم خبره است که حقایق مسئله را که در طی اجرا بدست آمده است، در خود دارد .

سیستم خبره اطلاعات بدست آمده در طی اجرا را که در حافظه کاری ذخیره می‌شود با قوانین تطبیق می‌دهد تا حقایق جدید استنتاج شده و آنها را در حافظه کاری قرار دهد. ممکن است به نتایجی نیز برسد که قبلاً در حافظه کاری وجود داشته است. برخی سیستم‌های خبره توانایی بهره‌گیری از پایگاه‌داده‌های خارجی، صفحه‌های گسترده و حتی سنسورها را دارند. که آن اطلاعات به حافظه کاری منتقل شده و در جریان فرآیند استنتاج قرار می‌گیرد.

۴- موتور استنتاج (Inference Engine)

همانطور که گفته شد، سیستم خبره فرآیند استدلال انسانی را با ماژولی بنام موتور استنتاج مدل می‌کند.

تعریف: موتور استنتاج، پردازنده‌ای در یک سیستم خبره است که حقایق موجود در حافظه کاری را با دانش محیط کاربرد موجود در پایگاه دانش تطبیق می‌دهد تا در مورد مسئله به نتیجه برسد.

موتور استنتاج با حقایقی که در حافظه کاری و پایگاه دانش قرار دارد کار می‌کند تا به اطلاعاتی جدید برسد. بدین صورت که در مجموعه قوانین بدنبال قسمت اگر قوانینی می‌گردد که با اطلاعات موجود در حافظه کاری منطبق باشند. زمانیکه یک نمونه تطبیق شده از قوانین

پیدا شد، نتیجه قانون را به مجموعه حقایق موجود در حافظه کاری اضافه می‌کند «دانسته جدید»، تا جستجو برای یافتن قوانین تطبیق پذیر جدید ادامه پیدا کند.

برای مثال، دو قانون قبلی را در نظر گرفته و پرسش و پاسخ سیستم خبره و کاربر را ببینید:

STEP ۱

EXPERT SYSTEM: Does the car not start?
USER: TRUE

کاربر این واقعیت را به حافظه کاری اضافه می‌کند، از آنجا که این حقیقت قانون ۱ را پشتیبانی می‌کند، نتیجه آن به حافظه کاری اضافه می‌شود.

USER ASSERTS: The car Will not start.
SYSTEM ASSERTS: The problem may be in the electrical system

STEP ۲

EXPERT SYSTEM: Is the battery voltage below ۱۰ volts
USER: TRUE

حالا حافظه کاری شامل اطلاعات جدیدی است که قانون ۲ را پشتیبانی می‌کند، لذا سیستم خبره نتیجه آن قانون را به حافظه کاری اضافه می‌کند.

USER ASSERTS: The battery voltage is below ۱۰ volts
SYSTEM ASSERTS: The fault is a bad battery

در این مرحله کار اتمام می‌یابد زیرا قانون دیگری برای بررسی وجود ندارد.

۵- سهولت توضیح (Explanation Facility)

مشخصه بارز سیستم‌های خبره توانایی آنها در توضیح فرآیند استدلالی‌شان است. در سیستم خبره ماژولی قرار دارد که وظیفه‌اش توضیح روند استدلال است. با بکارگیری این ماژول، یک سیستم خبره می‌تواند برای کاربر توضیح دهد که *چرا Why* چنین سوالی از کاربر می‌کند و *چگونه How* به نتیجه نهایی‌رسیده‌است.

بکارگیری این ماژول «سهولت توضیح» برای توسعه دهنده سیستم این مزیت را دارد که خطاهای ممکن در دانش سیستم را تشخیص بدهد و برای کاربر نیز این مزیت را دارد که روش استدلالی سیستم را بطور شفاف ببیند.

۶- توضیح چگونگی

علاوه بر نتیجه نهایی، هم فرد خبره و هم سیستم خبره می‌توانند توضیح دهند که چگونه به نتیجه رسیده‌اند. این توانایی برای یک سیستم خبره بسیار مهم است.

بر خلاف یک برنامه متعارف که روی مسائل کاملاً تعریف شده کار می‌کند. یک سیستم خبره روی مسائلی کار می‌کند که ساختارمند نیستند. لذا دستاورد سیستم خبره زیر سوال خواهد رفت، به همین منظور باید ابزاری برای پشتیبانی نتایج بدست آمده ارائه شود و از آن نتیجه دفاع کند.

برای مثال:

EXPERT: The battery is bad.
USER: HOW
EXPERT: از آنجایی که ماشین روشن نمی‌شد، به این نتیجه رسیدم که در سیستم برق ماشین اشکالی وجود دارد، مشاهده شد که ولتاژ باطری از ۱۰ ولت کمتر است ، پس به این نتیجه رسیدم که باطری خراب است.

این گونه توضیح دادن، در واقع دنبال کردن فرآیند استدلالی فرد خبره است. سیستم‌های خبره نیز به دستور HOW بطریق مشابه دیده شده پاسخ می‌دهند. در حقیقت عقب گردی از روی قوانینی که منتج به نتیجه نهایی‌شده‌اند خواهد کرد. این مسیر طی شده از روی قوانین تا نتیجه نهایی، مسیر استدلالی سیستم را نشان می‌دهند. در نتیجه کاربر به نتیجه ارائه شده سیستم بیشتر اعتماد خواهد کرد زمانیکه منطق پشت توصیه سیستم خبره را ببیند.

۷- توضیح چرا

یک سیستم خبره همچنین می‌تواند توضیح دهد چرا چنین سوالی از کاربر می‌پرسد.
در شرایطی که با فرد خبره سوال و جواب می‌کنیم، ممکن است فرد از فرد خبره سوال کند که چرا چنین سوالی از او می‌کند، پاسخ فرد خبره باعث اعتماد بیشتر فرد به فرآیند استدلالی می‌شود و به این نتیجه می‌رسد که دانش فرد خبره کامل و قابل دفاع است.
برای مثال:

EXPERT: Will the car not start?
USER: WHY
EXPERT: اگر بدانم که ماشین روشن نمی‌شود، معمولاً به این نتیجه می‌رسم که در سیستم برق ماشین مشکلی وجود دارد.

۸- رابط کاربری (Interface)

تعامل بین یک سیستم خبره و یک کاربر باید به فرم خیلی طبیعی باشد، همانند گفتگوی بین انسانها. برای این منظور در مورد رابط کاربری سیستم خبره باید توجه لازم انجام گیرد. یکی از نیازهای اصلی در رابط کاربری نحوه سوال پرسیدن است. برای آنکه اطلاعات قابل اعتماد از کاربر بدست آید، باید به نحوه طراحی سوالات دقت کافی بشود. برای همین منظور ممکن است مجبور به بکارگیری منوها، گرافیک، نمودار و دیگر ابزار تعاملی با کاربر شویم.
حتی ممکن است ابزاری جهت تغییر و مشاهده اطلاعات درون حافظه کاری فراهم سازیم. زمانیکه کاربر بخواهد پاسخ سوالات قبلی را تغییر دهد این ابزار کمک خواهند کرد.

۹- ویژگیهای یک سیستم خبره:

ویژگی‌های یک سیستم خبره شامل:

۱- جدایی دانش از کنترل
۲- داشتن دانش خبره
۳- تخصص متمرکز
۴- استدلال با سمبلها
۵- استدلال بصورت ابتکاری
۶- توانایی ارائه استدلال نادقیق
۷- تنها به مسائل قابل حل محدود می‌شود
۸- با پیچیدگی معقول قابل توسعه است
۹- می‌تواند اشتباه کند

۹-۱- جدایی دانش از کنترل

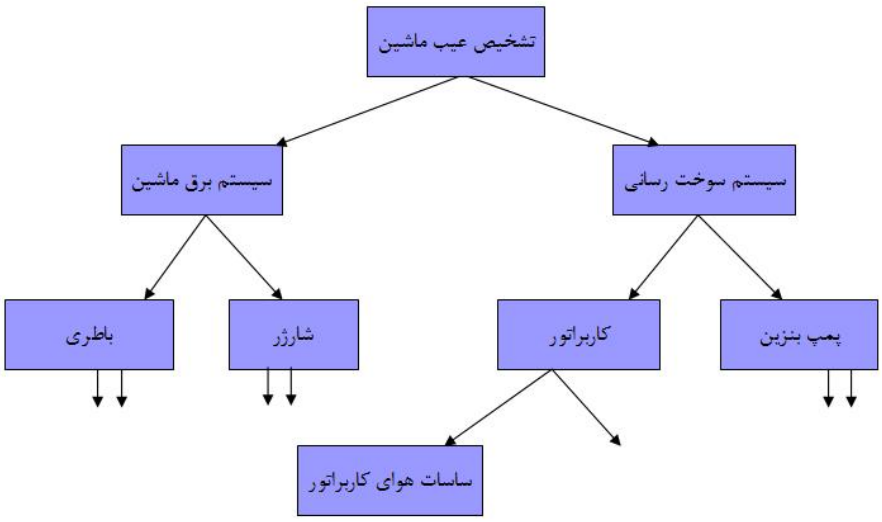
همانطور که در شکلهای قبلی نشان داده شد، پایگاه دانش از موتور استنتاج مجزا ست. این ویژگی وجه تمایز سیستم خبره از نرم افزارهای متعارف است و مزایایی به سیستم خبره می‌دهد. در برنامه‌های متعارف که دانش و کنترل باهم عجین هستند، تغییرات برنامه هم در دانش و هم در فرآیند کنترل برنامه تاثیر خواهد گذاشت. از طرفی بررسی برنامه و فهم دانش و روش استدلالی برنامه مشکل می‌شود.
جداسازی دانش از کنترل، نگهداشت و توسعه سیستم را آسانتر می‌سازد. می‌توان به راحتی قانونی را تغییر داد یا قانون جدیدی را اضافه کرد. اگر لازم باشد روش استدلالی تغییر کند کافی است الگوریتم استنتاج تغییر کند و به دانش کاری نخواهیم داشت.

۹-۲- داشتن دانش خبره

یکی از ویژگی‌های یک سیستم خبره آن است که تخصص یک فرد خبره را درونی می‌سازد. تخصص منبع با ارزشی است که تعدادی مشخص از افراد آن را دارند. افرادی که می‌توانند با موفقیت مسائلی را حل کنند که بقیه در حل آن عاجز هستند. بدست آوردن این سیستم خبره و کد کردن آن خبرگی بسیار ارزشمند خواهد بود.

۹-۳- تخصص متمرکز

اغلب افراد خبره در شاخه کوچکی از تخصصشان توانا هستند. اما در خارج آن محدوده توانایی کمتری دارند. همانند انسان یک سیستم خبره نیز در دامنه خاصی از مسئله تواناست. مثلاً از سیستمی‌که برای تشخیص عیب ماشین طراحی می‌شود، انتظار توانایی حل مسائل مالی نمی‌رود.
بهترین نتایج ارائه شده در سیستم‌های خبره‌ای بوده که برای دامنه مشخص و محدودی از مشکلات ایجاد شده‌اند. سیستم‌های خبره‌ای که دامنه وسیعی را هدف گرفته‌اند موفقیت چندانی نداشته‌اند.
مثال: برای ایجاد سیستم تشخیص عیب ماشین، یک سیستم جامع می‌توان ایجاد کرد اما مدیریت آن بسیار مشکل خواهد شد. لذا منطقاً سعی می‌کنیم که آن را به زیر سیستم‌های قابل مدیریت تقسیم کنیم.



شکل (۳) روند استدلال یک سیستم خبره

۹-۴- استدلال با سمبلها

سیستم‌های خبره دانش را به فرم سمبلیک ارائه می‌کنند. می‌توانیم با کمک سمبلها دامنه وسیعی از گونه‌های مختلف دانش را ارائه کنیم مانند حقایق، مفاهیم، قوانین. بخشی از هوش مصنوعی روی نحوه ارائه دانش متمرکز شده‌است. یکی از روشهای ارائه دانش با کمک سمبل‌هاست.

برای مثال:

علی تب دارد.

مردمی‌که تب دارند باید مقداری آسپرین بخورند.

Fever (Ali) ↔ علی تب دارد

با کمک ارائه دانش با زبان سمبلها، سیستم خبره می‌تواند مسایل را حل کند.

برای مثال:

ادعا	علی تب دارد	fever (Ali)
قانون	اگر مردم تب داشته باشند آنگاه باید مقداری آسپرین بخورند	X takes (aspirin) → fever(X)
نتیجه	علی باید مقداری آسپرین بخورد	Ali takes Asprin

منطق گزاره‌ای، از عملگر استدلال Implies یا → برای نمایش ساختار یک قانون کمک می‌گیرد.

از متغییر X برای آنکه عناصر مختلف بتوانند در قانون صدق کنند کمک می‌گیریم. وهمانطور که دیده می‌شود به جواب می‌رسیم.

۹-۵- استدلال به صورت ابتکاری

اغلب افراد خبره از خبرگیشان کمک می‌گیرند و مسائل را به صورت ابتکاری حل می‌کنند.

مثلاً ممکن است یک فرد خبره بگوید:

همیشه اول سیستم برق ماشین را چک می‌کنم.

مردم به ندرت در تابستان سرما می‌خورند.

اگر به سرطان مشکوک شوم، همیشه تاریخچه خانوادگی را بررسی می‌کنم.

افراد خبره از ابتکار برای میانبری به حل مسئله نگاه می‌کنند. با بکار گیری این خاصیت در سیستم خبره آن را از رویه‌های خشک برنامه‌های کاربردی متعارف متمایز می‌کند.

در برنامه‌های متعارف یک الگوریتم وجود دارد که اعمال خاصی را به ترتیب انجام می‌دهد. این الگوریتم همیشه عملیاتی را به ترتیب مشخص انجام می‌دهد. در حقیقت در برنامه‌های متعارف علاقه روی محاسبات عددی است.

در استدلال ابتکاری، بر اساس اطلاعات فراهم شده نتیجه گیری می‌کند، سمت و سوی مشخصی را دنبال نمی‌کند. بطور مثال برای تشخیص کاهش جریان در لوله‌ها می‌توان از روش ابتکاری زیر بهره برد:

ابتکار: لوله‌های قدیمی هنگام کاهش جریان عبوری می‌لرزند.

استدلال ابتکاری: اگر لوله‌ها می‌لرزند و لوله‌ها قدیمی‌هستند آنگاه به کاهش جریان عبوری باید مشکوک شد.

سیستم خبره می‌تواند از این قانون برای تشخیص کاهش جریان عبوری کمک بگیرد.
این قانون اصلاً تضمین نمی‌کند که حتماً فشار کاهش یافته‌است، بلکه جریان استنتاج را به سمت بررسی فشار جریان عبوری هدایت می‌کند. در حقیقت نقطه شروع بررسی مناسبی را برای سیستم خبره انتخاب می‌کند.
اگر این روش موفق نبود می‌توان از همان روشهای متعارف برای حل مسئله کمک گرفت.

۹-۶- توانایی ارائه استدلال نادقیق

سیستم‌های خبره در کاربردهایی که نیازمند استدلال نادقیق هستند بسیار خوب جواب داده‌اند. جایی که اطلاعات نامشخص، مبهم یا حتی در دسترس نباشد و یا دانش محیط کاربرد بصورت پایه‌ای شامل عوامل نادقیق باشد.
مثال، پزشک به سبب کمبود وقت در شرایط اورژانس و بدون انجام آزمایشات کافی تصمیم‌گیری مناسب می‌کند.
اطلاعات نادقیق:

ممکن است از حسن یک ساندویچ بخرم.

نتایج تست قلب را ندارم.

موتور ماشین داغ کرده‌است.

دانش نادقیق:

ساندویچ‌های حسن معمولاً خوب است.

اگر تست قلب نداشته باشیم، بیمار درد قفسه سینه داشته باشد، ممکن است ناراحتی قلبی داشته باشد.

مقداری روغن به موتوری که جوش می‌آورد اضافه کنید.

۹-۷- تنها به مسائل قابل حل محدود می‌شود

زمانیکه پروژه یک سیستم خبره آغاز می‌شود، باید مراقب باشید که آیا مسئله اصولاً قابل حل است یا خیر؟
این تصور اشتباه وجود دارد که هوش مصنوعی می‌تواند از پس حل تمام مسائل برآید. اگر فرد خبره‌ای وجود نداشته باشد تا مشکل را حل کند، امید بسیار کمی برای ایجاد سیستم خبره‌ای برای حل مشکل وجود دارد.
سیستم خبره برای مسائلی قابل ارائه‌است که فرد خبره‌ای برای حل آن مسئله وجود داشته باشد. مسائل نو یا مسائل با تغییرات زیاد کاندید خوبی نیستند.

۹-۸- با پیچیدگی معقول قابل توسعه است

مسئله باید پیچیدگی معقول داشته باشد، نه زیاد سخت باشد نه زیاد آسان. اگر مسئله خیلی آسان باشد، شاید صرف هزینه برای حل مسئله با کمک سیستم خبره مقرون به صرفه نباشد. البته برای برخی کاربردهای ساده، ممکن است اقدام به ارائه سیستم خبره بکنیم، چون حجم اطلاعات زیاد است یا حساسیت مسئله ایجاب می‌کند. مانند قیمت گذاری وسیله نقلیه.

مسئله نباید خیلی پیچده باشد، مسائلی که ساعت‌ها وقت کارشناس را می‌گیرد برای حل به کمک سیستم خبره مناسب نیستند. در صورت پیچیدگی باید به زیر مسائل ساده تر شکسته شود.

۹-۹- می‌تواند اشتباه کند

یک کارشناس یا فرد خبره از آنجایی که انسان است ممکن است اشتباه کند، این را می‌دانیم اما باز به فرد خبره اعتماد می‌کنیم. در یک سیستم خبره از آنجا که دانش فرد خبره است که استخراج می‌شود، ممکن است این دانش دچار اشکالاتی باشد. از این لحاظ برنامه‌های متعارف نسبت به سیستم‌های خبره برتری دارند، اما اگر در مقام جایگاه کاربرد در نظر بگیریم، می‌توان گفت این مقایسه نادرست است. برنامه‌های متعارف برای اطلاعات درست و دقیق بکار گرفته می‌شوند مانند پایگاه داده‌ها یا برنامه‌های حسابداری، اگر اطلاعات ناقص باشد، برنامه دچار خطا می‌شود. اما در یک سیستم خبره ممکن است با وجود کمبود اطلاعات نتیجه درست و مناسب توسط سیستم اتخاذ شود.

۱۰- مقایسه سیستم خبره با برنامه‌های کاربردی متعارف

سیستم‌های خبره	برنامه‌های متعارف
سمبلیک	عددی
ابتکاری	الگوریتم
دانش از کنترل مجزاست	اطلاعات و کنترل مجتمع است
اعمال تغییرات آسان است	اعمال تغییرات مشکل است
اطلاعات نا دقیق است	اطلاعات دقیق است
شامل گفتگو با کاربر به همراه توضیحات	رابط کاربری بصورت دستور است
توصیه و توضیحات لازمه ارائه می‌شود	نتیجه نهایی داده می‌شود
راه حل قابل قبول ارائه می‌شود.	راه حل بهینه ارائه می‌شود

۱۱- برنامه‌نویسی در مقابل مهندسی دانش

از زمان ایجاد کامپیوترها، افراد از آنها جهت توسعه برنامه‌هایی به منظور انجام سریعتر محاسبات، دستیابی به اطلاعات یا مدل کردن فرآیندهای پیچیده بهره برده‌اند. از تجربیات بدست آمده از ساخت خیلی از سامانه‌ها، مهندسان برنامه نویس به تکنیک‌ها و روشهای مناسبی جهت طراحی و پیاده‌سازی برنامه برای مسایل مختلف دست یافته‌اند.

اما سیستم‌های خبره هنوز نو هستند لذا به متدولوژی مناسبی جهت توسعه سیستم‌های خبره نرسیده‌ایم.

۱۱-۱- برنامه‌نویسی مرسوم

برنامه‌نویسی مرسوم معمولاً شامل سه بخش طراحی، پیاده‌سازی و رفع اشکال است. برنامه نهایی زمانی ارائه می‌شود که از تمام مراحل فوق با موفقیت گذشته باشد.

نقطه شروع برنامه‌نویسی مرسوم از نیازهای کابردی است (فاز طراحی). از همین جا برنامه‌نویس تصویری از برنامه نهایی‌را در ذهن خود دارد. در مرحله پیاده‌سازی برنامه نویس تنهاست و معمولاً تغییرات روی خصوصیاتی که در فاز طراحی در نظر گرفته شده پذیرفته نیست.
در مرحله رفع نقص، برنامه تست شده و بررسی می‌شود آیا به آن ویژگی‌های مورد انتظار رسیده‌است یا خیر.

۱۱-۲- مهندسی دانش (Knowledge Engineering)

در دنیای برنامه‌نویسی مرسوم «داده» عنصر اساسی است، تمرکز روی داده‌ها است و سعی داریم فرآیندی برای حل آنها پیدا کنیم. اما در سیستم‌های خبره تمرکز روی دانش است. آنها دانش را کسب، دسته‌بندی و بررسی می‌کنند تا به مسئله کاملاً فهمیده شود.
تعریف: مهندسی دانش، فرآیند ساخت یک سیستم خبره است.
برخلاف برنامه‌نویسی مرسوم، ایجاد سیستم خبره فرآیندی تعاملی است. طراحی بخشی از سیستم را ایجاد می‌کند تست کرده لذا در صورت لزوم دانش سیستم را اصلاح می‌کند. این فرآیند تا رسیدن به محصول نهایی ادامه دارد.

۱۲- مراحل ایجاد سیستم خبره

فاز اول- ارزیابی «سنجش» (Assessment) :

در این مرحله امکان پذیری انجام پروژه بررسی می‌شود. از مراحل مهم و اولیه است. در این مرحله نیازهای انجام پروژه مانند تعداد پرسنل و منابع مورد نیاز بدست می‌آید. منابع دانش مورد نیاز مانند افراد خبره و گزارشات لازم دیگر نیز مشخص می‌شود.

فاز دوم- اکتساب دانش (Knowledge Acquisition) :

هدف این فاز استخراج دانش مورد نیاز است. این دانش معمولاً از فرد خبره بدست می‌آید. این فاز شامل انجام جلسات با افراد خبره جهت مشخص شدن ابعاد مسئله، مفاهیم اصلی و روشهای حل مسئله توسط افراد خبره است. این فاز به عنوان گلوگاه ایجاد سیستم خبره است.

تعریف: اکتساب دانش فرآیند کسب، دسته‌بندی و بررسی دانش است.

فاز سوم- طراحی (Design) :

در این فاز ساختار کلی و سازمان دانش سیستم ایجاد می‌شود. ابزار نرم‌افزاری مناسب انتخاب می‌شود. در این فاز نمونه اولیه برنامه ایجاد می‌شود تا به درک مناسبی از صورت مسئله برسیم.

فاز چهارم- تست (Testing):

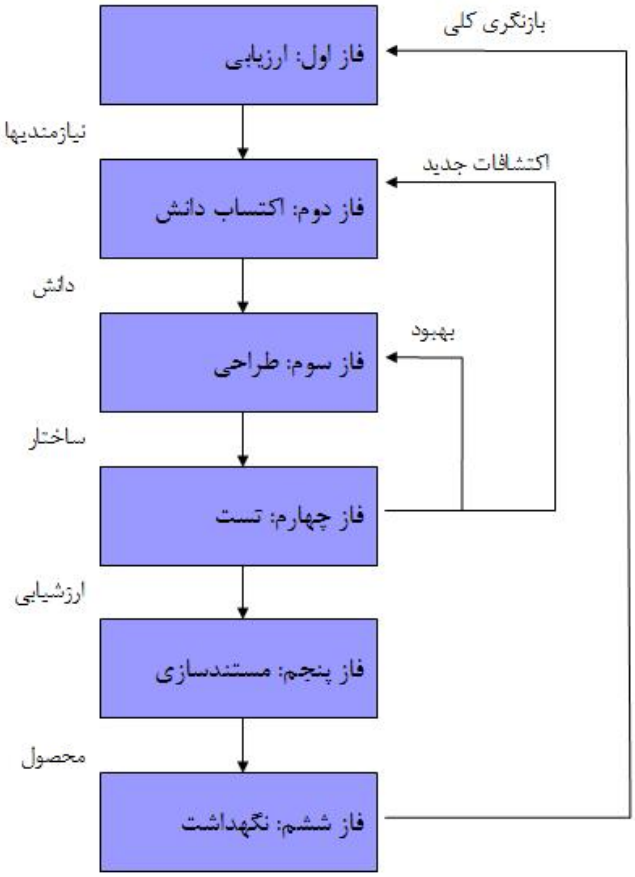
این فاز اگرچه جدا در نظر گرفته شده، اما تا پایان فرآیند تکاملی پروژه در حال انجام است. زمانیکه دانش جدیدی به سیستم اضافه می‌شود برنامه مورد تست جامع قرار می‌گیرد. در طی مرحله تست طراحی هم با فرد خبره جهت توسعه دانش سیستم و هم با کاربر جهت توسعه رابط کاربری همکاری نزدیک دارد.

فاز پنجم- مستندسازی (Documentation):

مستندسازی کارهای صورت گرفته در فرآیند تولید نرم‌افزار یکی از نیازهای اصلی پروژه نرم‌افزاری است. مستندات جهت آموزش، راه‌اندازی سیستم و بیان مشخصات کلی سیستم است.

فاز ششم- نگهداشت (Maintenance):

سیستم بعداز تحویل باید بصورت متناوب و دوره‌ای مورد بررسی قرار گرفته و پشتیبانی شود. همانند یک کودک سیستم خبره بزرگ می‌شود و می‌آموزد. دانش ثابت نیست، توسعه می‌یابد، رشد می‌کند و کامل می‌شود. حتی ممکن است تغییرات بنیادی باشد و ویژگی‌های سیستم را نیز دگرگون کند.



شکل (۴) فازهای پیاده‌سازی یک سیستم خبره

۱۳- برنامه‌نویسی در مقابل تولید سیستم خبره

تفاوتهای اساسی این دو فرآیند شامل موارد زیر است:

تولید سیستم خبره	برنامه‌نویسی مرسوم
تمرکز روی مسئله است	تمرکز روی راه‌حل است
کار گروهی است	برنامه‌نویس به تنهایی کار می‌کند
فرآیند تولید تکراری است.	فرآیند تولید پله‌ای است

۱۴- افراد درگیر در پروژه سیستم خبره

افراد موثر در پروژه تولید یک سیستم خبره شامل، کارشناس محیط کاربرد، مهندس دانش و کاربر نهایی است. که هرکدام در طی فرآیند نقش کلیدی خود را ایفا می‌کنند. مشخصات کلی آنها:

فرد خبره	- دانش کارشناسی دارد - توانایی کافی در حل مسائل را دارد - توانایی ارائه دانش را دارد - می‌تواند زمان بگذارد - خصومتی با انجام پروژه ندارد
مهندس دانش	- توانایی مهندسی دانش را دارد - تونایی برقراری ارتباط قوی را دارد - توانایی‌تطبیق مسئله را به نرم‌افزار را دارد - توانایی برنامه‌نویسی سیستم خبره‌ای دارد
کاربر نهایی	- می‌تواند در تعریف محیط کاربری موثر باشد - می‌تواند در کسب دانش کمک باشد - می‌تواند در توسعه سیستم کمک باشد.

۱۴-۱- کارشناس محیط کاربرد (Domain Expert)

تعریف: کارشناس محیط کاربرد فردی است که توانایی و دانش لازم جهت حل یک مسئله را بصورت بهتری نسبت به بقیه دارد.

وجود محیط کابرد بسیار مهم است، زیرا می‌خواهیم راجع به یک مسئله مشخص و محدود کار کنیم.

وجه تمایز یک فردخبره (کارشناس) با یک فرد ناخبره در دانش است. فرد خبره دانش حل مسئله را دارد. فرد خبره نسبت به فرد ناخبره این توانایی را دارد که اطلاعات مهمتر را طوری کنار هم قرار می‌دهد تا در حل مسئله بطور موثری بکار آیند.

افراد خبره تخصص‌شان را در حل مسائل در طی سالیان زیادی کسب کرده‌اند لذا می‌توانند بطور خلاصه رویه‌ای جهت حل موثر مشکلات ارائه دهند. افراد خبره باید بتوانند دانش خود را به راحتی بیان کنند تا به صورت کد در سیستم خبره درآید.

معمولاً افراد خبره افرادی ارزشمند هستند لذا باید زمان کافی برای همکاری در پروژه از این افراد را بدست آورد. زمان ایجاد یک سیستم خبره معمولاً طولانی و بصورت تکاملی است.

فرد خبره باید همکاری لازمه را در اجرای پروژه داشته باشد. اگر خصومتی در انجام کار نشان دهد پروژه مطمئناً با شکست مواجه خواهد شد. معمولاً افراد خبره نگاه خوبی نسبت به سیستم‌های هوشمند ندارند. لذا باید بسترسازی مناسب انجام گیرد.

۲-۱۴- مهندس دانش (Knowledge Engineer)

تعریف: مهندس دانش، فردی است که طراحی، ساخت و تست یک سیستم خبره را بر عهده دارد.
در نگاه اول، مهندس دانش و برنامه‌نویس نزدیک بهم هستند، چون هر دو کد می‌کنند. اما تفاوتهایی به شرح زیر دارند:
مهارت‌های مهندس دانش: مهندسی دانش، بخشی از ساخت یک سیستم خبره است. در حقیقت نوعی هنر است. زیرا فرآیندی پیچیده دارد و دستورالعمل‌های خیلی مشخصی برای انجام این فرآیند وجود ندارد. اما آنچه مشخص است آن است که مهندس دانش باید بتواند مشخص کند که حل مسئله با سیستم خبره ممکن هست یا خیر؟
ارزیابی امکان انجام پروژه شامل تحقیق روی مشخصات مسئله است. برخی مسائل برای حل به کمک سیستم خبره بسیار مناسب هستند در حالیکه برخی دیگر از مسایل با سیستم خبره به سختی حل می‌شوند. در حقیقت امکان سنجی برای پروژه سیستم خبره از رویه‌های سفت و سخت آنالیز هزینه‌ها و درآمدها که در مهندسی نرم‌افزار متعارف انجام می‌گیرد بهره نمی‌گیرند. معمولاً علاقه بکارگیری تکنولوژی جدید و نگهداشت دانش درون سازمان می‌تواند انگیزه‌ای قوی در جهت تایید انجام پروژه باشد.
در طی فرآیند اکتساب دانش، هدف اصلی مهندس دانش آن است که پرده از دانش فرد خبره بردارد. از طریق مصاحبه‌های مختلف، مهندس دانش بدنبال مفاهیم کلیدی مسئله و راه‌حلهایی که توسط افراد خبره در حل مسئله انجام می‌دهند، می‌گردد.
مهندس دانش باید از مهارت لازم جهت هدایت جلسه مصاحبه بر خوردار باشد تا در نهایت بتواند دانش فرد خبره را بطور موثر و در زمان کوتاه‌تری استخراج کند.
بعد از آن مهندس دانش، مجموعه دانش استخراجی را طوری دسته‌بندی می‌کند که به طور موثر در سیستم خبره قابل اعمال باشد.
مهندس دانش پکیج نرم‌افزاری مناسبی را انتخاب خواهد کرد که توانایی پذیرش دانش استخراجی باشد و شامل فرآیند استنتاج باشد.
مهندس دانش همچنین مسئول کد کردن، تست کردن و تجدید نظر کردن سامانه است تا زمانیکه سامانه به درجه‌ای از بلوغ رسیده باشد که کارایی یک فرد خبره را نشان بدهد. البته بعد از پایان پروژه مهندس دانش همچنان مسئول نگهداشت و توسعه سامانه خواهد بود.
مهارت‌های ارتباط موثر: بخش زیادی از موفقیت مهندس دانش در نحوه بیرون کشیدن دانسته‌های فرد خبره است. همان‌طور که گفته شد فرد خبره تمایل زیادی دارد که دانشش را طوری ارائه دهد که فرآیند حل سریع و کارا باشد پس آنها فرآیند استدلالی را عجیب خواهند کرد. با گرفتن مقادیری اطلاعات اولیه، به سرعت به نتیجه نهایی‌خواهند رسید. برای یک سیستم خبره که می‌خواهد در سطح یک فرد خبره نمایان شود، دانش پنهان در روند فرآیند استدلالی آشکار شود. این فرآیند مشکل را تجزیه دانش Knowledge Decomposition می‌نامیم. که خود مهارت خاص دارد و باید به مهندس دانش آموخته شود.

توانایی انطباق مسئله به نرم‌افزار: تعداد زیادی زبان برنامه‌سازی و ابزار توسعه سیستم‌های خبره (که به آنها پوسته یا Shell گفته می‌شود) وجود دارد تا با آنها بتوان سیستم خبره را ایجاد و توسعه داد. هر یک از آنها ویژگی‌های خاصی در حل دسته‌ای خاص از مسائل دارند. وظیفه مهندس دانش است که نرم‌افزار مناسب را انتخاب کند. مهندس دانش سعی می‌کند تا ارتباطی مناسب بین مسئله و نرم‌افزار برقرار سازد. افراد خبره گونه‌های مختلفی از دانش را بکار می‌گیرند، آنها را با روشهای مختلفی برای خودشان دسته‌بندی و مرتب می‌کنند و از روشهای استدلالی مختلفی بهره می‌برند تا مسئله را حل کنند. بهمین نسبت هم زبانهای مختلف و پوسته‌های مختلف می‌توانند دانش را به روشهای مختلفی ارائه، دسته‌بندی و پردازش کنند. بهمین دلیل برای مهندس دانش مهم است که همخوانی بین نیازمندیهای مسئله و توانایی‌های نرم‌افزار برقرار سازد.

مهارت برنامه‌نویسی مهندس دانش: مسئول اصلی کد کردن دانش استخراج شده از فرد خبره، مهندس دانش است. لذا ایشان باید در برنامه‌نویسی سیستم خبره مهارت لازم را داشته باشد. برای این منظور باید اول بدانیم که چگونه می‌توان دانش را ارائه کرد که در فصل بعد نگاهی به آن خواهیم انداخت. در فصل چهارم پردازش آنها را بررسی خواهیم کرد. در حقیقت یکی‌از اهداف اصلی این درس آن است که این مهارت را به شما ارائه دهد.

۱۴-۳- کاربر نهایی (End User)

کاربر نهایی فردی است که بطور مستقیم با سامانه کار می‌کند. پذیرش نهایی سیستم بقدار زیادی به این بستگی دارد که چقدر از نیازهای کاربر نهایی مرتفع می‌شود. تاریخچه سیستم‌های خبره پر از سیستم‌هایی است که به لحاظ تکنیکی درست ساخته شده بودند اما هرگز بکار گرفته نشدند، زیرا نیازهای کاربر نهایی اصلاً در نظر گرفته نشده بود.

تعریف ویژگی‌های رابط کاربری: سیستم خبره نهایتاً باید ویژگیهای رابط کاربری که کاربر نهایی انتظار دارد را داشته باشد. مشخصاتی مانند: دسترسی سیستم، ورود اطلاعات، توضیحات سیستم، فرم نتایج نهایی و ابزارهای کمکی مورد نیاز و ...

کاربر نهایی مشخص می‌کند که سیستم خبره چگونه راه‌اندازی شود، خواه از طریق منو یا از طریق یک برنامه دیگر، یا اصلاً با روشن شدن کامپیوتر فعال شود.

روشهای مختلفی برای ورود اطلاعات به یک سیستم خبره وجود دارد. باید مشخص شود که کاربر نهایی چه روشی را برای پاسخ به سوالات گزینه‌ای، سوالات چند جوابی یا حتی تایپی مناسب می‌داند و می‌پذیرد.

کاربر نهایی ممکن است به توضیحات کافی برای فرآیند استدلالی سیستم احتیاج داشته باشد، مخصوصلاً زمانیکه بخواهد پاسخ نهایی سیستم را ارزیابی یا چک کند. لذا کاربر نهایی می‌تواند نحوه ارایه توضیحات را مشخص کند.

کاربر نهایی ممکن است به نرم‌افزارهای کمکی احتیاج داشته باشد که برخی احتیاجات جانبی‌اش را رفع کرده و سامانه را پشتیبانی کنند. برای مثال ممکن است بخواهد برنامه قدرت آن را داشته باشد که روی پایگاه داده‌ای خارجی دست برده و اطلاعات آن را تغییر دهد یا فایل با فرمت صفحه گسترده ایجاد کند. طراحی رابط کاربری سیستم می‌تواند کاری پیچیده باشد، حتی نصف بودجه پروژه را مصرف کند. اما امری حیاتی است که موفقیت پروژه در گرو آن است و کاربر نهایی نقش تعیین کننده‌ای در آن دارد. بدون پشتیبانی کاربر نهایی، هرچقدر قدرتمند باشد، در تست آخر قبول نخواهد شد.

کاربر نهایی نقش اساسی در فرآیند آغازین پروژه دارد. اغلب مهندسین دانش تازه کار در ابتدای پروژه مبهوت جنبه‌های ریز مسئله می‌شوند زیرا آنها اول از فرد خبره شروع می‌کنند. در حالیکه فرد خبره همیشه برای توضیح مسئله وارد جزئیات زیاد و گاهی هم غیر ضروری می‌شود که تنها سر درگمی مهندس دانش را در پی دارد، در حالیکه ابتدا باید از کاربر نهایی شروع کرد زیرا او فهم گسترده‌ای از آنچه می‌خواهیم، را ارائه می‌دهد و در مرحله بعد فرد خبره جزئیات لازم را برای پر کردن فضاهای خالی صورت مسئله، ارائه می‌کند.

جزوه درس:

سیستم‌های خبره

Expert systems

کتاب مرجع:

Expert Systems Design and Development

نوشته:

John Durkin

مانی عابدینی

تورج بنی‌رستم

فصل سوم - ارائه دانش
Knowledge Representation

۱- مقدمه

در این فصل تکنیک‌هایی که برای کد کردن دانش در یک سیستم خبره وجود دارد ارائه می‌شود. این تکنیک‌ها با رویه‌هایی‌که در روشهای برنامه‌نویسی متعارف دیده شده متفاوت خواهد بود.

یک برنامه روی داده پردازش می‌کند اما یک سیستم خبره روی دانش، لذا باید دانش را به فرم سمبلیکی ارائه کرد که قابل کار در سیستم خبره باشد.

در این فصل به بررسی برخی روشهای ارایه دانش پر کاربرد اشاره می‌کنیم، ضمن آنکه باید توجه داشته باشیم که هیچ‌کدام از روشهای مطرح به تنهایی نمی‌تواند برای تمام مسائل جواب‌گو باشد.

همان‌طور که قبلاً مفصل بحث شده بود، قدرت سیستم‌های خبره در دانش آنهاست، به قولی «دانش قدرت است».

اما خود دانش چیست؟ دانش مفهومی انتزاعی است که سعی دارد درک فردی را پیرامون موضوعی ضبط کند.

تعریف: دانش، فهم و درکی از یک بستر موضوعی مشخص است.

مثال: بستر موضوع پزشکی.

در حین ساخت سیستم خبره سعی نمی‌کنیم تمام دانش فرد خبره را ضبط کنیم، بلکه تنها دانش فرد خبره مرتبط با یک موضوع تعریف شده از آن بستر کلی را در نظر می‌گیریم. بطور مثال بیماری‌های عفونت خون. لذا می‌گوییم دانش دامنه کاربرد خاص.

تعریف: دامنه، یک بستر موضوع خاص کاملاً تعریف شده.

عنصر اساسی در موفقیت ایجاد یک سیستم خبره، تمرکز روی دامنه تعیین شده است. زمانیکه دامنه وسیع باشد، به دانش روی موضوعات مختلفی نیاز پیدا خواهد شد که کارایی را به شدت تحت تاثیر می‌گذارد.

بعد از آنکه دانش در دامنه کاملاً مشخصی از فرد خبره استخراج شد، باید آن را در سیستم خبره کد کرد. لذا باید ساختاری برای کد کردن دانش داشته باشیم که سیستم خبره بتواند با کمک آن، و براساس آن دانش مسئله را حل کند، همانطوری که فرد خبره می‌توانست حل کند. که بحث ارائه

دانش (Knowledge Representation) است. *تعریف: ارائه دانش، روشی است که برای کد کردن دانشی که در پایگاه دانش یک سیستم خبره بکار گرفته می‌شود.*

۲- گونه‌های مختلف دانش:

روانشاسان علوم شناختی تئوریهایی دارند که نحوه حل مسئله در انسان را بیان می‌کند. اما نوع دانشی که انسان معمولاً با آن کار می‌کند، اینکه چگونه آنها را دسته‌بندی می‌کند و چگونه از آنها بطور موثری برای حل یک مسئله کمک می‌گیرد هنوز مشخص نیست.

از آنجایی که تئوری واحدی برای توضیح دسته‌بندی دانش انسان وجود ندارد و حتی بهترین تکنیک برای ساختاردهی داده در یک برنامه کامپیوتری متعارف را نمی‌توان انتخاب کرد، پس هیچ ساختار ارائه دانش مشخصی را نمی‌توان ایده‌آل دانست.

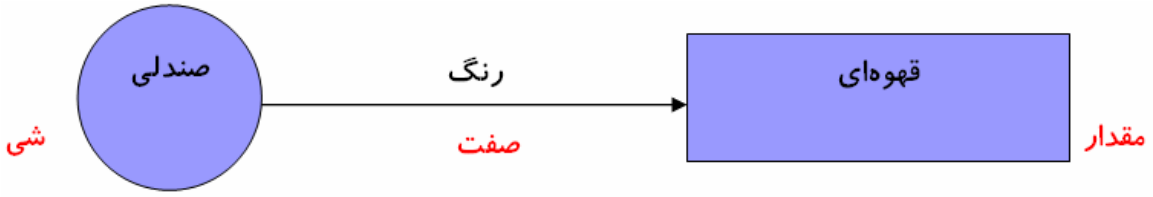
یکی از وظایف مهندس دانش انتخاب بهترین روش برای ارائه دانش با توجه به صورت مسئله است. لذا باید روشهای مختلف ارائه دانش را بطور کامل بشناسیم و بدانیم هرکدام، چه دسته‌ای از دانش را بهتر می‌تواند ارائه دهد.

در ادامه تعدادی از روشهای ارائه دانش را بررسی می‌کنیم که شامل:

۱) سه گانه شی_صفت _مقدار (O-A-V)
۲) قوانین Rules
۳) شبکه‌های معنایی Semantic Networks
۴) قاب‌ها Frames
۵) منطق Logic

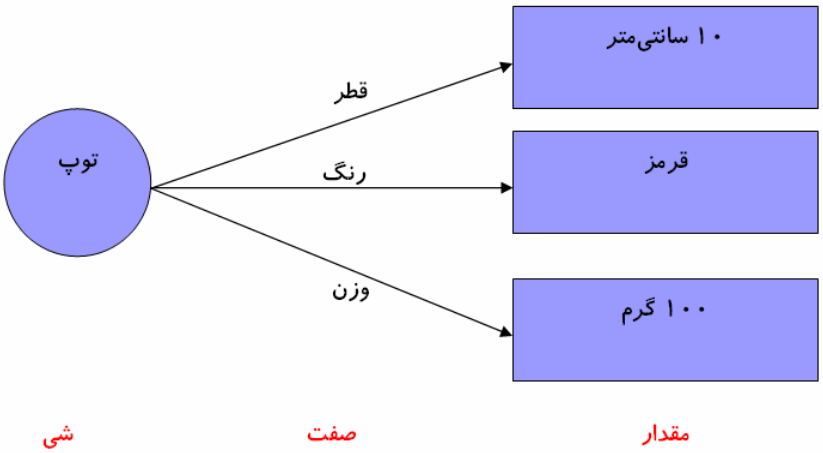
۲-۱- سه‌گانه OAV (Object-Attribute-Value):

در بیان یک واقعیت، می‌توان از رابطه یک شی با یک صفت و یک مقدار بهره گرفت که با کمک اشکال و خطوط آن را نشان می‌دهیم. مثلاً رنگ صندلی قهوه‌ای است را می‌توان به صورت زیر نشان داد:



شکل (۱) مثالی از O-A-V

در رابطه بالا، شی می‌تواند فیزیکی باشد مانند اتومبیل، می‌تواند انتزاعی باشد مانند سپرده. صفت ویژگی آن شی است و مقدار آنچه را که به صفت نسبت داده شده را مشخص می‌کند.
در برخی موارد یک شی ممکن است چندین صفت مهم داشته باشد. در این مواقع براحتی می‌توان چند صفت برای آن تعریف و مقداردهی کرد.
در ادامه خواهیم دید که در شبکه‌های معنایی و فریم‌ها هم از چند صفتی برای بیان بهتر شی بهره می‌گیریم.
برخی صفات بطور منطقی تنها می‌توانند یک مقدار بخود بگیرند، در حالیکه بقیه می‌توانند چند مقداری باشند. که به آنها حقایق تک مقداری یا چند مقداری می‌گوییم. در طی فرآیند طراحی یک سیستم خبره باید تعیین شود که رابطه O-A-V قرار است تک مقداری باشد یا چند مقداری.



شکل (۱) مثالی از O-A-V با چند صفت

فرض کنید از ارائه O-A-V برای بیان جمله‌ای در مورد اندازه فشار که توسط فشار سنجی گرفته شده بخواهیم بهره بگیریم. شی «فشارسنج» است، صفت «فشار». مقادیر ممکن می‌تواند «کاهش»، «ثابت» یا «افزایش» باشد. اگر سیستم خبره به اطلاعات فشارسنج احتیاج داشته باشد.

به صورت زیر سوال خواهد کرد:

سوال: لطفاً اعلام کنید فشار در فشار سنج به چه صورتی است:

کاهش

ثابت

افزایش

پاسخ: کاهش.

بطور منطقی فشار تنها می‌تواند یکی از سه وضعیت را داشته باشد. لذا O-A-V تک مقداری خواهد بود.

حال سطح تحصیلات یک نفر را در نظر بگیرید. شی یک «شخص» است و صفت «سطح تحصیلات». اما یک نفر می‌تواند چندین مدرک داشته باشد. دبیرستان، دانشگاه و حتی در رشته‌های مختلف. لذا رابطه O-A-V چند مقداری خواهد بود.

سوال: لطفاً سطح تحصیلات خود را وارد کنید:

دیپلم

پیش‌دانشگاهی

لیسانس

پاسخ: پیش‌دانشگاهی و دیپلم.

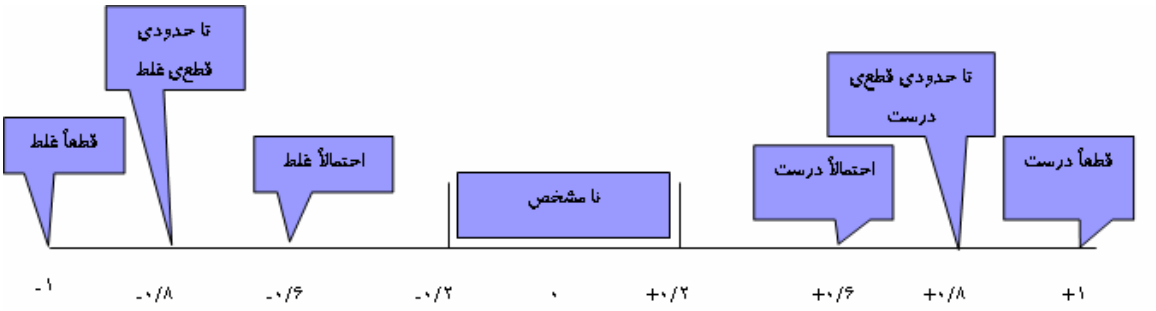
هردوی حقایق تک مقداری یا چند مقداری یک ویژگی مهمی را به اشتراک گذاشته‌اند. زمانیکه کاربر یک مقدار از یک لیست را انتخاب می‌کند، سیستم مقدار آن صفت را در حافظه کاری خود true فرض می‌کند و بقیه را false . مانند کاهش فشار.

حقایق غیر قطعی (Uncertain Facts)

دنیای پیرامون ما، سیاه یا سفید نیست. معمولاً بطور کاملاً دقیق نمی‌دانیم رخدادی درست است یا غلط. در واقع درجه باور داریم. این درجه باور در مکالمه روزمره می‌تواند: شاید، احتمالاً، با احتمال قوی و غیره باشند. برای مثال فردی می‌گوید: «احتمال دارد امروز باران ببارد». انسانها براحتی می‌توانند مفهوم جمله را بفهمند. اما متاسفانه یک سیستم خبره چنین توانایی ندارد.

یک روش متداول برای مدیریت اطلاعات نادقیق بکارگیری «ضریب قطعیت» (Certainty Factor) است. یک CF مقدار عددی است که به یک جمله داده می‌شود و نشان می‌دهد که چه‌قدر به آن جمله اعتقاد داریم. «میزان درستی آن جمله».

شکل سه ضریب قطعیت را برای نگاشت توضیح کیفی جملات به مقدار عددی CF نشان داده شده است.



شکل (۳) نمونه‌ای از نگاشت کیفی

همانطور که دیده می‌شود مقادیر ۱- و ۱+ قطعیت باور را می‌رساند اما به طرف وسط یعنی ۰ که پیش می‌رویم از میزان قطعیت کاسته می‌شود تا جاییکه در بازه‌ای بین ۰/۲- تا ۰/۲+ نامشخص است.

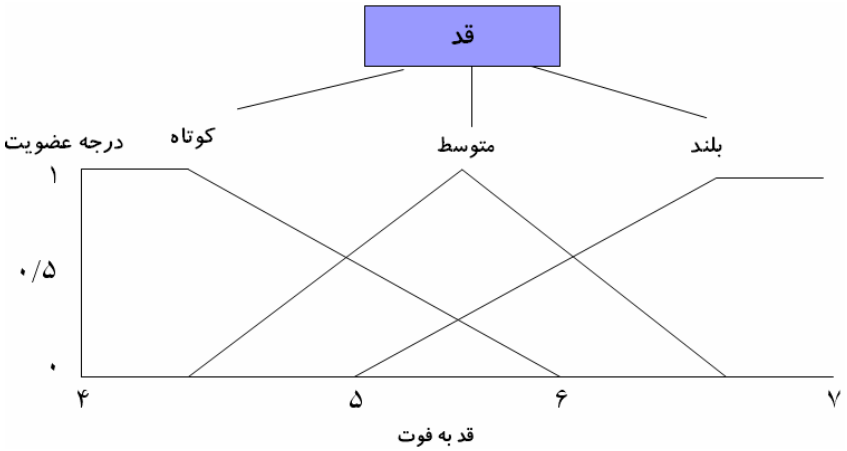
با این ترتیب برای جمله «احتمال دارد امروز باران ببارد» CF را ۰/۶ می‌توان در نظر گرفت.

حقایق فازی (Fuzzy Facts)

جای دیگری که عدم قطعیت در دنیای سیستم‌های خبره می‌شود، آنجاست که بیان نامشخص در ارائه جملات زبان روزمره‌مان دیده می‌شود. بطور مثال «فرد بلند قد است». جمله ابهام دارد زیرا از کلمه بلند استفاده شده است. انسانها با عبارات مبهم در استدلال‌هایشان اصولاً مشکلی ندارند اما در کامپیوتر اینگونه نیست.

خوشبختانه منطق فازی روشهایی هم برای ارائه و هم برای استدلال با عبارات مبهم فراهم ساخته است. عبارات مبهم در مجموعه‌های فازی ارائه می‌شوند، که در حقیقت برداشت کیفی از تفسیر آن عبارت را کمی می‌کند. به این نگاشت از مفهوم به عدد، درجه عضویت می‌گوییم.

در شکل زیر مجموعه‌های فازی را نشان می‌دهد که صفات مختلفی از قد افراد را نشان می‌دهد.



شکل (۴) تابع عضویت در سیستم فازی

درجه عضویت عددی بین ۰ و ۱ است که بیانگر آن است که قد فرد تا چه حدی به هر مجموعه فازی تعلق دارد. برای مثال زمانیکه قد فردی ۵/۵ فوت باشد، فرد به افراد متوسط با درجه عضویت ۱ تعلق دارد در حالیکه همان فرد به گروه کوتاه قد و بلند قد با درجه عضویت ۰/۲۵ تعلق دارد.

علاوه بر ساخت مجموعه‌های فازی، منطق فازی اجازه نوشتن قوانین فازی را نیز می‌دهد. یک قانون فازی شامل مجموعه‌های فازی در هر دو بخش اگر و آنگاه است. برای مثال:

IF the person's height is *tall*
THEN the person's weight is *heavy*

با قراردادن درجه عضویت در قسمت مقدم یک قانون، می‌توان استدلال کرده و درجه عضویت فرد را به مجموعه سنگین وزن‌ها محاسبه کرد. در حقیقت یک قانون فازی نگاشتی از مجموعه‌های فازی به مجموعه‌های فازی است.

۲-۲- قوانین (Rules):

بیان حقایق یا واقعیات برای حل مسئله در سیستم خبره بسیار موثر هستند، زیرا وضعیت جاری را با کمک آن حقایق درک می‌کند. اما برای حل، احتیاج به دانش بیشتری دارد. یکی از ساختارهای پرکاربرد برای بیان این دانش اضافی قانون است.

تعریف: قانون، یک ساختار دانش است که برخی اطلاعات شناخته شده را به اطلاعات دیگری مرتبط می‌سازد که می‌تواند نتیجه‌گیری شده یا استدلال شده باشند.

یک قانون بین اطلاعات داده شده و عملی یک ارتباط ایجاد می‌کند. این عمل می‌تواند اطلاعات جدیدی را ارائه دهد یا بخشی از یک رویه باشد. لذا می‌توان گفت قانون می‌گوید چگونه مسئله را حل کنیم.

ساختار منطقی قانون یک یا چند مقدم را که در قسمت اگر ظاهر شده‌اند به یک یا چند تالی که در قسمت آنگاه آمده‌اند مرتبط می‌سازد. مثال:

IF the ball's color is red
THEN I like the ball

این قانون نشان می‌دهد که اگر رنگ توپ قرمز باشد، می‌توان استنتاج کرد که توپ را دوست دارم.

بطور کلی، یک قانون می‌تواند مقدمهای زیادی داشته باشد که با AND یا OR یا ترکیبی از هردو بهم متصل شده‌اند. در قسمت نتیجه می‌توان یک جمله منطقی داشت یا ترکیبی منطقی از AND.

یک قانون می‌تواند جمله Else نیز داشته باشد که زمانی استنتاج خواهد شد که مقدمها نادرست باشند. مثال صفحه بعد را دقت کنید:

IF Today's time is after 10 am
AND today is a weekday
AND I am at home
OR My boss called and said that I am late for work
THEN I am late for work
ELSE I am not late for work

در یک سیستم خبره قانون_پایه (مبتنی بر قانون)، دانش یک دامنه کاربردی در مجموعه‌ای از قوانین ضبط می‌شود و وارد پایگاه دانش سیستم خواهد شد. سیستم از این قوانین به همراه اطلاعات درون حافظه کاری بهره خواهد گرفت تا مسئله را حل کند. زمانیکه قسمت اگر قانون با اطلاعات درون حافظه کاری منطبق باشد، سیستم عملی را که در قسمت آنگاه آمده است انجام

خواهد داد. در این شرایط به اصطلاح قانون fire شده است و جملات قسمت آنگاه به حافظه کاری اضافه می‌شود. که خود می‌تواند موجب fire شدن قانون‌های دیگری شود.

فرآیند از سوال سیستم که رنگ توپ چه رنگی است آغاز می‌شود. سیستم پاسخ قرمز را می‌گیرد و به حافظه کاری اضافه می‌کند. (قدم اول).

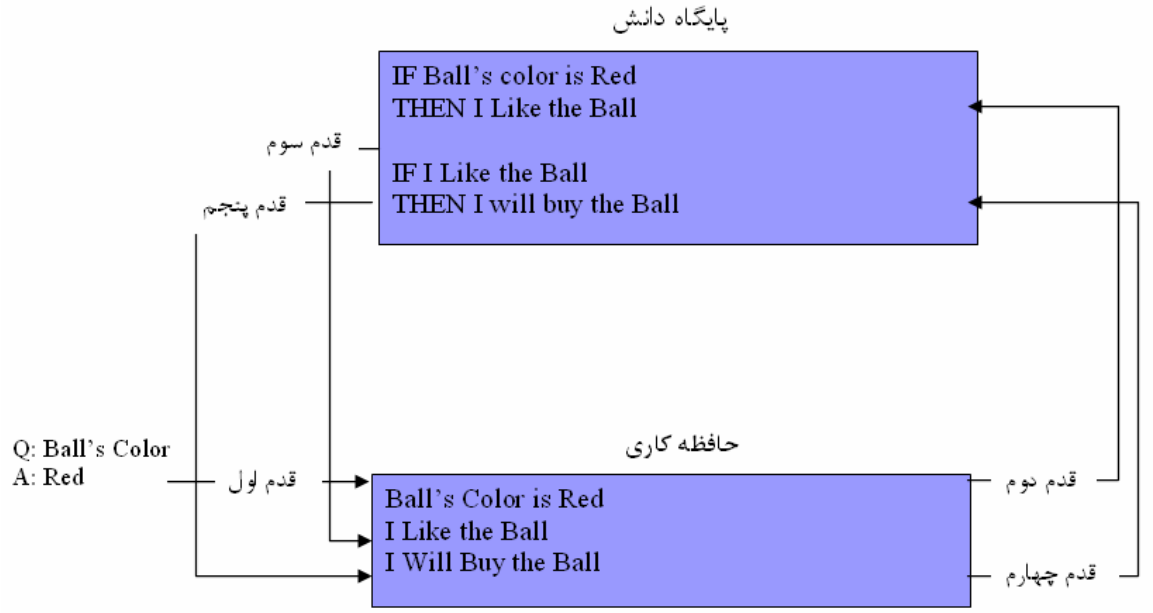
این اطلاعات جدید روی مقدم قانون اول تطبیق دارد (قدم دوم).

این انطباق باعث اجرای قانون شده و تالی آن به حافظه کاری اضافه می‌شود (قدم سوم).

این اطلاعات جدید با مقدم قانون دوم تطبیق دارد (قدم چهارم).

قانون اجرا شده حقیقت «توپ را می‌خرم» را به حافظه کاری اضافه می‌کند (قدم پنجم).

در این مرحله چون قانون دیگری برای اجرا وجود ندارد کار خاتمه پیدا می‌کند.



شکل (۵) فرآیند استدلالی یک سیستم خبره قانون پایه

اجرای یک رویه:

علاوه بر استنباط اطلاعات جدید، یک قانون می‌تواند عملیاتی مانند محاسبات ساده را نیز انجام دهد.

```
IF The Area of the square is needed
THEN AREA = LENGTH*WIDTH
```

زمانی این قانون اجرا می‌شود که مقدم آن به حافظه کاری اضافه شود، و در صورت اجرا مساحت مربع را محاسبه می‌کند. اگر عملیات پیچیده‌تری مد نظر باشد باید یک برنامه خارجی اجرا شود و نتیجه برگشت داده شود.

به مثال صفحه بعد توجه شود:

Rule 1 اجرای یک تابع خارجی

IF The design requires a new box
AND NUMBER= number of items to pack
AND SIZE = Size of item
THEN CALL COMPUTE_BOX_VOLUME
AND SEND NUMBER, SIZE
AND RETURN VOLUME

Rule 2 کسب اطلاعات از یک صفحه گسترده

IF January Sales Needed
THEN OPEN SALES
AND JANUARY_SALES = B7

Rule 3 کسب اطلاعات از پایگاه داده‌ای

IF There is a Plant emergency
AND NAME = Smith
THEN OPEN TELEPHONE
AND FIND NAME, NAME-FIELD
AND TELEPHONE = TELEPHONE-FIELD

انواع قوانین:

قوانین می‌تواند از انواع زیر باشد:

رابطه‌ای

IF باتری خراب است

THEN اتومبیل روشن نخواهد شد

توصیه

IF اتومبیل روشن نشود

THEN یک تاکسی بگیر

جهت دهی

IF اتومبیل روشن نشود

AND سیستم سوخت رسانی مشکلی نداشته باشد

THEN سیستم برق ماشین را چک کن

استراتژی

IF اتومبیل روشن نشود

THEN اول سیستم سوخت رسانی را بررسی کن سپس سیستم برق ماشین را بررسی کن

ابتکاری

IF اتومبیل روشن نشود

AND نوع ماشین فورد ۱۹۵۷ باشد

THEN شناور باک را چک کن

قوانین می‌تواند بر اساس استراتژی که الگوی حل مسئله نیز نامیده می‌شود دسته‌بندی شوند:

مشکلات تفسیری

IF ولتاژ مقاومت R1 از ۲ ولت بیشتر بود

AND ولتاژ جمع‌کننده Q1 از ۱ ولت کمتر بود

THEN بخش مدار تغذیه در رنج نرمال قراردارد

مشکلات تشخیصی

IF نشانه سردرد دارید

AND سابقه حمله سردرد شدید دارید

AND نتیجه سی‌تی اسکن نرمال نیست

THEN اینگونه پیداست که درون جمجمه خونریزی داخلی رخ داده و باید عمل جراحی صورت گیرد

مشکلات طراحی

IF کار فعلی طراحی منبع تغذیه باشد

AND محل منبع تغذیه درون کابینت در نظر گرفته شده باشد

AND فضای کافی برای منبع تغذیه در کابینت باشد

THEN منبع تغذیه را درون کابینت قرار بده

قوانین متغییر:

در برخی کاربردها، ممکن است یک عمل را بروی مجموعه‌ای از اشیاء مشابه اعمال کنیم. می‌توان یک قانون برای هر شی نوشت، این روش کارا نیست و نگهداشت آن مشکل خواهد بود. برای مثال فرض کنیم بخواهیم برای تک تک کارمندان یک اداره قوانین بازنشستگی را بنویسیم. هم زمانبر است و هم اگر زمانی قانون بازنشستگی تغییر کند باید کل مجموعه قوانین تغییر کند. می‌توان برای

این دسته از قوانین از قوانین الگویی کمک گرفت. در این قوانین متغییری قرار دارد که می‌تواند بجای آن جملات منطقی قرار بگیرد:

IF ?X is EMPLOYEE
AND ?X AGE > 65
THEN ?X can Retire

این قانون تمام حافظه کاری را برای دو عنصر مقدم قانون می‌گردد، اگر تطابقی پیدا کرد، نتیجه می‌گیرد که آن فرد می‌تواند بازنشسته شود. بطور مثال اگر در حافظه کاری اطلاعات زیر باشد:

Smith is EMPLOYEE	Jones is EMPLOYEE	Miller is EMPLOYEE
Smith AGE = 67	Jones AGE = 70	Miller AGE = 60

متغییر ?X نام شخصی را به خود می‌گیرد، بررسی می‌شود برای تک تک آنها انجام می‌شود و خروجی زیر خواهد بود.

Smith can retire Jones can retire

قوانین نادقیق:

همانطور که حقایق نادیق داریم، قوانین نادقیق نیز داریم. یک فرد خبره ممکن است قانونی بیان کند که رابطه نادقیقی بین مقدم و تالی آن برقرار باشد. مانند:

IF تورم زیاد باشد

THEN قریب به یقین نرخ سود پایین است

در این مثال قریب به یقین نادقیق است که باید از ضریب اطمینان کمک گرفت:

IF تورم زیاد باشد

THEN نرخ سود بالاست CF=۰/۸

اگر تعیین شود که تورم دقیقاً زیاد است آنگاه نتیجه می‌توان گرفت که نرخ سود با قطعیت ۰/۸ بالاست.

فرا قانون (Meta Rule)

برخی افراد خبره دانشی دارند که حل مسئله را جهت دهی می‌کند. این دسته دانش از آنهایی که تا حال بررسی گردید متفاوت است، زیرا تعیین می‌کند که حل مسئله چگونه بهتر انجام شود.

تعریف: فرا دانش، دانش پیرامون بکارگیری و کنترل دانش محیط کاربرد است.

تعریف: فرا قانون، یک قانون است که توضیح می‌دهد قوانین دیگر چگونه بکار گرفته شوند.

- مثال:

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

مجموعه‌های قانون:

دانش یک فرد خبره معمولاً بصورت مجموعه‌های جدا دسته‌بندی شده است، برخی قوانین برای مشکلی کارآمد هستند در حالیکه برای مشکلی دیگر اصلاً بکار نمی‌آیند. بطور مثال ممکن است یک مکانیک دانش خود را در قالب مجموعه‌های قوانین سیستم سوخت و برق تفکیک کند. زمانیکه مشکل سوخت‌رسانی باشد تنها آن قوانین بکار گرفته می‌شوند. دسته‌بندی مجموعه‌های قوانین به خبرگی فرد کارشناس بستگی دارد.

برخی قوانین برای آنکه تصمیم بگیریم اصولاً مشکل در سیستم برق است یا در سیستم سوخت، اگر در سیستم سوخت بود مشکل در کاربراتور است یا در لوله سوخت و ... مجموعه فرا قوانین سیستم را تشکیل می‌دهند.

یک مزیت این روش در استخراج قوانین آن است که پروسه فکری یک فرد خبره دقیقاً از چنین سیستمی پیروی می‌کند. دومین مزیت آن این است که نگهداشت و اعمال تغییرات در سیستم راحت‌تر خواهد بود. هر ماژول را می‌توان جداگانه ایجاد و تست کرد. مزیت دیگر این روش آن است که با این خط‌مشی، می‌توان هر ماژول را خود سیستمی خبره مجزا فرض کرده و اجازه دهیم تا از ارائه دانش خاص خود و حتی تکنیک استنتاجی خاص خود بهره ببرد.

در مورد مجموعه‌های مجزای قوانین و مزیت آنها بحث شد، اما چگونه این مجموعه‌های باهم تعامل دارند؟

در برخی مسائل پیچیده نیاز به دانش افراد خبره مختلفی داریم. در حقیقت کمیته‌ای از خبرگان جمع شده که هرکدام در زمینه خاص دانش ارزنده‌ای دارند. در نهایت با اعمال قوانین کل کمیته روی هم می‌توان به نتیجه مناسب رسید. مثال استراتژی بازاریابی یک سازمان بزرگ را در نظر بگیرید.

این رویه به توزیع حل مسئله منجر می‌شود. که در سیستم خبره از طریق تعامل بین مجموعه قوانین بدست می‌آید.

تعریف: تخته سیاه، طراحی است که سیستم‌های خبره مختلف اطلاعاتشان را در یک منبع مشترک به اشتراک می‌گذارند.

معماری سیستم‌خبره‌ای که از تخته سیاه بهره می‌گیرد شامل اجزاء اصلی زیر است:

کمیته‌ای از سیستم‌های خبره

تخته سیاه

برنامه‌ریز

هر ماژول سیستم خبره بخشی از مشکل را حل می‌کند، هر کدام اطلاعاتشان را با دیگری به اشتراک می‌گذارند،که از طریق نوشتن در تخته سیاه یا خواندن از آن انجام می‌گیرد.

۲-۳- شبکه‌های معنایی(Semantic Networks):

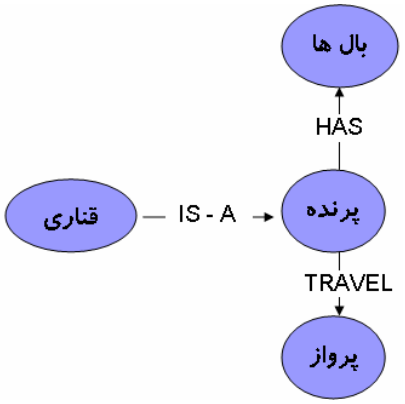
یکی از دست‌یافته‌های جدید علم هوش مصنوعی شبکه‌های معنایی‌است.

تعریف: شبکه‌های معنایی، روشی از ارائه دانش است که از گرافی شامل گره‌ها و کمان‌هایی استفاده می‌کند که گره‌ها بیانگر اشیاء و کمانها بیانگر ارتباطات بین اشیاء هستند.

یک شبکه معنایی، دید گرافیکی از اشیاء مهم، خصوصیات و ارتباطاتشان است. هم گره‌ها و هم کمان‌ها دارای اسمی هستند که کاملاً شی را معرفی می‌کند یا رابطه بین آنها را بیان می‌کند.

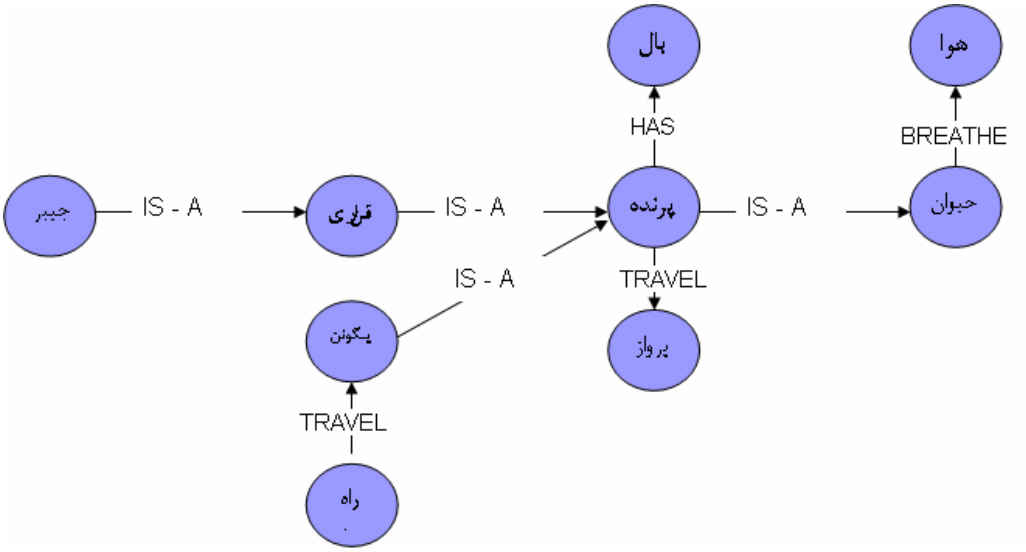
شکل زیر مثالی از یک شبکه معنایی است:

این نمودار شامل دو گره است که بیانگر شی و دو نود بیانگر صفت است. از آنجایی که پرنده دو بال دارد و می‌تواند پرواز کند و از آنجایی‌که قناری گونه‌ای پرنده است، منطقی است که استدلال کنیم قناری بال دارد و می‌تواند پرواز کند.



شکل(۶) نمونه‌ای از یک شبکه معنایی

شبکه را به راحتی می‌توان توسعه داد و اشیاء دیگر به همراه خصوصیاتشان را وارد نمودار کرد.



شکل (۷) توسعه یک شبکه معنایی

رابطه IS-A نوعی رابطه خاص به عام است. قناری گونه‌ای خاص از پرنده است.
وراثت مفهومی اصلی در شبکه‌های معنادار است. در نمودار صفحه قبل، یک جیبر قرمز تمام خصوصیاتی که یک قناری، پرنده و حیوان دارد را خواهد داشت زیرا از طریق رابطه IS-A گونه‌ای از تمام آنهاست و تمام خصوصیاتشان را به ارث می‌برد.
فرض کنید از گره پرنده بپرسیم چگونه پرواز می‌کند؟ گره پرنده روی کمانهای مرتبطش بدنبال کمانی با نام TRAVEL می‌گردد، اطلاعات مرتبط با آن کمان را برمی‌گرداند.
اگر گره‌ای از طریق کمانهای محلی نتوانست پاسخ پیدا کند جستجویش را از طریق کمانهای IS-A ادامه می‌دهد. فرض کنید همین سوال را از جیبر قرمز بپرسیم. چون خود گره دارای چنین کمانی نیست که به پاسخ متنهی شود، سوال به گره قناری داده می‌شود و آن گره نیز به نوبه خود سوال را به گره پرنده می‌دهد.
در نمودار دیده می‌شود که پنگوئن گونه‌ای پرنده است، پس انتظار داریم پرواز کند که غلط است. برای آنکه این گونه مشکلات پیش نیاید باید صفاتی که از نسل بالایی به ارث می‌رسد و غلط است را Over-ride کنیم. بدین منظور بطور مستقیم صفت راه‌رفتن را به آن انتساب می‌کنیم. نقطه ضعف وراثت در شبکه‌های معنایی، می‌تواند اشکالاتی پنهان را ایجاد کند که بررسی آنها خود نیاز به زمان زیاد دارد.

۲-۴- قاب‌ها (Frames):

یک توسعه طبیعی روی شبکه‌های معنایی است. یک شما واحدی است شامل دانش نمونه درمورد مفاهیم و اشیاء و در برگیرنده دانش است. برای مثال در شمای یک پرنده، ممکن است شامل دانشی باشد که نشان دهد پرنده هم پا دارد هم بال و اینکه چگونه غذا تهیه می‌کند. به این شما، قاب گفته می‌شود.

تعریف: یک فریم، ساختمان داده‌ای است برای ارائه دانش کلیشه‌ای از برخی مفاهیم یا اشیا.
زمانیکه به شرایط جدید می‌رسیم یا تغییراتی در بخشی از دید مسئله ایجاد می‌شود، ساختاری بنام فریم از حافظه فراخوانی می‌شود که واقعیات دیده شده در آن با اعمال جزئیات ثبت می‌شود.

طراحی قاب اصلی:

یک قاب در ظاهر مانند خیلی از فرم‌هاست، در هر فرم یکسری فیلد خالی وجود دارد که نام دارند و مقدار درون آنها باید پر شود. مانند فیلد نام دانشجو.

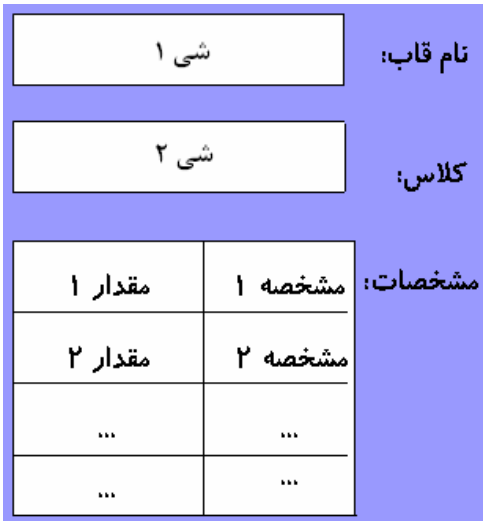
ساختار اصلی یک قاب به شکل زیر است:

مشخصات شامل نام و محلی برای وارد کردن مقدار آن است. مقادیر مشخصات معمولاً از سه گونه داده: دودویی (Boolean)، رشته‌ای(String) یا عددی است. هر فریم فیلد اختیاری کلاس نیز دارد.

در این فیلد می‌توان نام شی دیگر را قرار داد که به این شی ارتباط دارد در حقیقت رابطه IS-A را نشان می‌دهد. بطور مثال اگر در فریم علی، در قسمت فیلد کلاس مقدار انسان قرار دهیم. بیانگر آن است که «علی گونه‌ای از انسان» است.

خصوصیات در فریم همانند رابطه OAV عمل می‌کند.

فیلد کلاس در فریم و مقداردهی آن بین دو فریم همانند بودهای شبکه‌های معنادار اثر خواهد کرد.



شکل (۸) نمونه‌ای از یک قاب

قاب کلاسی (Class Frame)

یک قاب کلاسی، بیانگر خصوصیات کلی مجموعه‌ای از اشیاء است. برای مثال می‌توان یک قاب کلاسی ساخت که اشیائی مانند ماشین‌ها و قایق‌ها و حتی دما و فشار را توصیف کنند. برای مقدار هر مشخصه می‌توان مقدار پیش‌فرض Default در نظر گرفت.

مشخصات در قاب‌ها به دو دسته تقسیم می‌شوند:

- مشخصات ثابت: مشخصه‌ای از یک شی را نشان می‌دهد که مقدارش تغییر نمی‌کند.
- مشخصات متغییر: مشخصه‌ای از یک شی که در طول عملیات سیستم امکان تغییر آن وجود دارد.

در مثال پرنده در قاب زیر، مشخصاتی مانند رنگ، تعداد بال ثابت و مشخصاتی مانند گرسنگی و فعالیت پویا هستند.

مشخصات ارائه شده تقریباً برای تمام پرندگان مشترک است.

مقادیر زمانی که بین پرندگان مشترک باشد در این قالب کلاسی می‌آید در غیر این صورت «*نامشخص*» زده می‌شود. این مقادیر مقدار پیش فرض است، لذا می‌تواند برای هر پرنده تغییر کند.

نام قاب:	پرنده

مشخصات:	
رنگ	نا مشخص
غذا	کرم
تعداد بال	۲ عدد
قدرت پرواز	دارد
گرسنگی	نا مشخص
فعالیت	نا مشخص

شکل (۹) نمونه‌ای از یک قاب کلاس

قاب نمونه (Instance Frame)

همانطور که از یک کلاس Instance گرفته می‌شود و Object ایجاد می‌شود. زمانیکه یک نمونه از قاب کلاسی ایجاد می‌شود هر دوی مشخصات و مقادیر مشخصات را از کلاس به ارث می‌برد. می‌توان مقادیرش را بسته به آن نمونه خاص تغییر داد حتی می‌توان در صورت لزوم به آن نمونه مشخصاتی جدید با مقادیرشان اضافه کرد.

نام قاب:	جیبر قرمز
کلاس:	پرنده

مشخصات:	
رنگ	قرمز
غذا	کرم
تعداد بال	۱ عدد
قدرت پرواز	ندارد
گرسنگی	نا مشخص
فعالیت	نا مشخص
محل زندگی	قفس

شکل (۱۰) نمونه‌ای از یک قاب

وراثت در قاب‌ها:

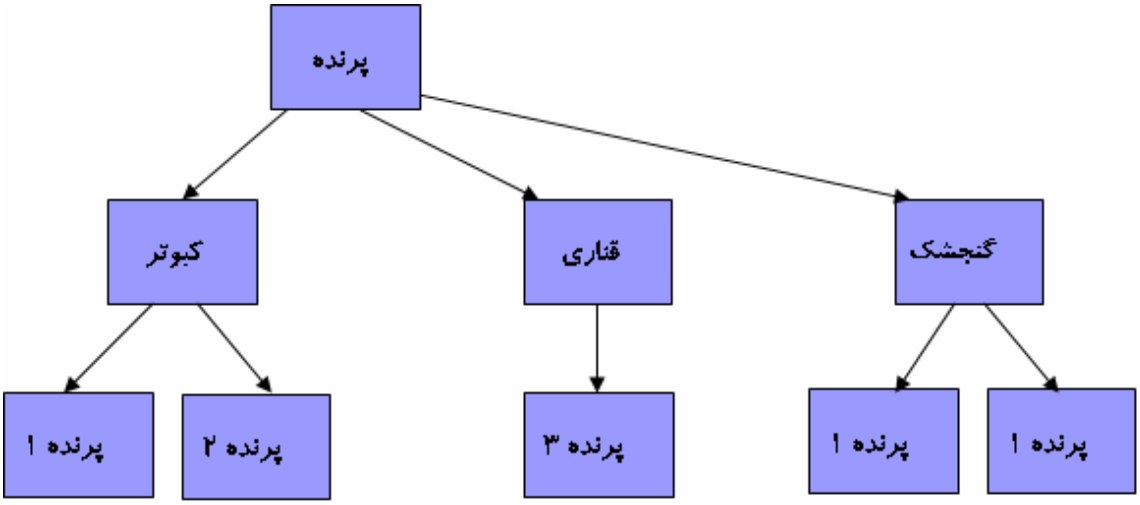
همانطور که گره‌ها در شبکه‌های معنایی‌از گره دیگر اطلاعات به ارث می‌برند، قاب نمونه از قاب کلاسی نیز اطلاعات به ارث می‌برد.

همین طور می‌توان در قاب کلاسی رویه‌هایی نیز قرار داد که در هنگام نمونه‌گیری به قاب نمونه به ارث برسد. بطور مثال در کلاس پرنده اگر متدی برای آنکه پرنده بداند در موقع گرسنگی چه کاری انجام دهد اضافه کنیم، این متد به قاب جیبر قرمز نیز به ارث می‌رسد.

ساختار سلسله مراتبی:

علاوه بر ساخت یک کلاس تکی و نمونه‌های مرتب با آن، می‌توان ساختار سلسله مراتبی برای قاب‌ها که ممکن است پیچیده هم باشند ایجاد کرد.

این ساختار سلسله مراتبی مفهوم یک پرنده را در سطوح مختلف انتزاعی دسته‌بندی می‌کند. قاب سطح بالا شامل اطلاعات مشترک بین همه پرندگان و قاب‌های سطح متوسط (زیر کلاسها) شامل اطلاعات خاص‌تری برای هر گروه خاص هستند.



شکل (۱۱) نمونه‌ای از ساختار سلسله مراتبی در قاب‌ها

بُعد (Facet)

سیستم‌های بر اساس قاب، ویژگی بنام فَست نیز دارند که قدرت کنترل روی مقادیر خصوصیات را می‌دهند. یک روش برای بکارگیری بعد تعریف قید روی یک مقدار خصوصیت است. برای مثال، می‌توان مقدار یک خاصیت را به لحاظ عددی محدود به بازه‌ای خاص کرد. یا اگر رشته‌ای است به نمونه‌های مشخصی محدود کرد. حتی نوع مقداری که برای فیلدی پذیرفته می‌شود را می‌توان مشخص کرد: رشته‌ای، عددی یا بولین.

یک روش دیگر آن است که از یک فَست استفاده کنیم و مشخص کنیم که یک خصوصیت چگونه می‌تواند مقدار بگیرد و یا اگر مقدار تغییر کرد چه عملی انجام دهد. که به ترتیب IF-Needed facet و IF-Changed Facet می‌نامیم.

If-Needed facet

برخی مواقع پیش می‌آید که می‌خواهیم برای آنکه مقدار مشخصه‌ای تعیین گردد رویه یا متدی اجرا شود. این رویه شامل محاسبات یا دسترسی به پایگاه داده خارجی است. یا حتی ممکن است وابسته به مقادیر دیگر خصوصیات باشد که بطور پویا درحال تغییر هستند. این رویه را با زبان برنامه‌نویسی رویه‌ای نوشته و به IF-Needed facet یک خصوصیت می‌چسبانیم.

بطور مثال فرض کنید رویه‌ای داشته باشیم:

```
IF      Bird:No_Wings <2
THEN    Bird:Flies= False

IF      Bird:No_Wings = 2
THEN    Bird:Flies= True
```

If-Changed facet

می‌توان فکتی ایجاد کرد و به یک خصوصیت مرتبط کرد تا در صورت تغییر مقدار آن، رویه‌ای اجرا شود. از این روش زمانی بهره می‌گیریم که تغییر در مقدار باعث ایجاد تغییر در فریمی دیگر شود یا حتی مقدار یک فیلدی دیگر در خود همان فریم باید مجدد محاسبه گردد.

بطور مثال فرض کنید رویه‌ای داشته باشیم که اگر پرنده گرسنه شد فعالیت پرنده را به خوردن تغییر دهد:

```
IF      Bird:Hungry = True
THEN    Bird:Activity= Eating
```

۲-۵- منطق (Logic)

یکی از روشهای ارائه دانش که خیلی قدیمی نیز است، منطق می‌باشد. منطق گزاره‌ای Propositional Logic و منطق اسنادی Predicate Logic روشهایی خیلی معروف هستند. هر دو تکنیک از سمبل برای ارائه دانش بهره می‌گیرند و از اپراتورها که به سمبل‌ها اعمال می‌کنند برای ایجاد استنتاج منطقی بهره می‌گیرند.

منطق گزاره‌ای: Propositional Logic

در منطق گزاره‌ای سعی داریم درستی جملات را تحقیق کنیم. به هر گزاره می‌توان یک سمبل متغییری نسبت داد مثال:

A = The Car will start

روی سمبلها می‌توان عملگرهای منطقی اعمال کرد و با هم ترکیب کرد. مانند: AND, OR, NOT
منطق گزاره‌ای روشی برای بیان حقایق و قوانین به فرم سمبلیک ارائه می‌دهد و روی آنها از طریق عملگرهای منطقی کار می‌کند. این روشی برای کار کردن روی جملاتی که مشخصاً درست یا غلط هستند بکارگرفته می‌شود. اما برای برخی جملات پیچیده معمولاً خیلی سریع نمی‌توان به یک نتیجه درست یا غلط رسید، لذا مجبور هستیم آن جمله را به جملات کوچکتر تفکیک کنیم. (محاسبات اسناد یا محاسبات گزاره‌ای)

محاسبات اسنادی (Predicate Calculus)

محاسبات اسنادی یا منطق اسنادی، توسعه‌ای روی منطق گزاره‌ای است که بیان بهتری برای ارائه دانش دارد. بجای آنکه تمام یک گزاره را با یک سمبل نشان دهیم -مثلاً رنگ توپ قرمز است=A - ، این منطق این اجازه را می‌دهد که به صورت رابطه‌ای تفکیک شده درآوریم: Color(ball, red) که مزیت آن در بکارگیری توابع و متغییرهاست.

در این منطق همانند منطق گزاره‌ای از سمبل‌ها کمک می‌گیریم که می‌توانند ثابت، گزاره، متغییر یا تابع باشد. روی این سمبلها می‌توان عملگرهای منطق گزاره‌ای را اعمال کرد.

ثابتها: برای نامدهی به اشیاء مشخص یا مشخصات خاص در مسئله بکار گرفته می‌شود. ثابت‌ها با حروف کوچک نوشته می‌شوند. مانند: ali, temperature

Predicate: در این منطق، یک fact یا گزاره به دو بخش تقسیم می‌شود. یک predicate و آرگومان. یک آرگومان بیانگر شی است و predicate یک یک ادعا روی آن شی است.

مانند: brother(ali,hassan)

به کلمه اول predicate می‌گوییم که رابطه‌ای بین دو آرگومان را تعریف می‌کند.

متغییرها: برای بیان کلاس کلی از اشیاء یا خصوصیات بکار گرفته می‌شود. با حروف بزرگ شروع می‌شوند. مانند: brother(X,Y)

متغییرها می‌توانند مقدار بگیرند، مثلاً X=ali و Y=hassan

توابع: از سمبلها برای نشان دادن یک تابع کمک می‌گیریم. تابع یک نگاشت از مجموعه‌ای از المانها به مجموعه دیگر است. مانند: father(jack)=bob

توابع را می‌توان داخل predicate ها بکار گرفت.

مانند: friends(father(jack),father(judy)) = friends(bob, Martin)

عملگرها: در این منطق همان عملگرهایی که در منطق گزاره‌ای بود بکار گرفته می‌شود.

مانند: $\text{brother}(X,Y) \wedge \text{brother}(Z,Y) \wedge \neg(X \equiv Z) \rightarrow \text{brother}(X,Z)$

سور عمومی: از سور عمومی برای نشان دادن اینکه عبارت به ازای تمام مقادیر ممکن برای متغییر تعیین شده درست خواهد بود، استفاده می‌شود.

مانند: $\forall X$ likes(X, flower)

سور وجودی: از سور وجودی برای نشان دادن اینکه عبارت به ازای برخی (حداقل یک مقدار) مقادیر ممکن برای متغییر تعیین شده درست خواهد بود، استفاده می‌شود.

مانند: $\exists X$ likes(X, flower)

استنتاج با کمک منطق:

نشان دادیم که چگونه از منطق برای ارائه دانش می‌توان بهره برد. اما باید بتوانیم روی این دانش استدلال کنیم. این فرآیند استدلالی روی حقایق موجود و رسیدن به نتیجه را استنتاج Inference می‌نامیم.

یک راه برای استنتاج استفاده از قانون حذف استنتاج است که اشاره دارد که اگر گزاره A درست باشد و $A \rightarrow B$ نیز درست باشد، آنوقت B درست است:

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

مثال کنترل ربات:

یک ربات را در نظر بگیرید که وظیفه‌اش آن است که بلاک‌هایی را جابجا کند. باید با منطق اسنادی به آن کمک کنیم دنیای پیرامون خود را بشناسد.

1-Cube(a) , cube(b) , cube(d) , pyramid(c) , sphere(e) , HAND(hand) , table(table1)
2-On(a,table1), on(b,table1) , on(d,a) , on(c,b)
3-holding(hand,nothing)

برای آنکه نشان دهیم شی را روی شی دیگر قرار داده: put_on(b,a)

برای آنکه بتواند جابجا کند: $\text{hand\_holding(b)} \wedge \text{clear(a)} \rightarrow \text{put\_on(b,a)}$

اگر بخواهیم این قدرت را برای تمام اشیاء داشته باشیم:

$\forall X \exists Y$ $\text{hand\_holding(X)} \wedge \text{clear(Y)} \rightarrow \text{put\_on(X,Y)}$

اما چگونه تشخیص دهد که شیء آزاد است:

$\forall X (\neg \exists Y \text{ on(Y,X)} \rightarrow \text{clear(X)})$

با اعمال قانون حذف استنتاج و داشتن اطلاعات زیر

On (a, table1), on(b,table1) , on(d,a) , on(c,b)

و قانون $\forall X (\neg \exists Y \text{ on(Y,X)} \rightarrow \text{clear(X)})$

می‌توان به اطلاعات جدید زیر رسید:

clear(c) , clear(e) , clear(e)

این مثال نشان داد که چگونه از استنتاج می‌توان برای استخراج اطلاعات جدید کمک گرفت (چه بلوک‌هایی آزاد هستند تا روی آنها بلوک دیگری قرار داد).

جزوه درس:

سیستم‌های خبره

Expert systems

کتاب مرجع:

Expert Systems Design and Development

نوشته:

John Durkin

مانی عابدینی

تورج بنی‌رستم

فصل چهارم - تکنیک‌های استنتاج
Inference Techniques

۱- مقدمه

در فصل قبل نشان داده شد که چگونه می‌توان دانش را به یک سیستم خبره ارائه کرد. در این فصل ارائه می‌گردد که چگونه سیستم می‌تواند با این دانش استدلال کرده و یک مسئله را حل کند. فرآیند استنتاج خود شامل تکنیک استنتاج است که تکنیک‌های مختلف نحوه ترکیب دانش درون پایگاه دانش و اطلاعات و حقایق درون حافظه کاری را با هم نشان می‌دهند.

۲- استدلال (Reasoning)

تعریف: استدلال فرآیند کار کردن با دانش، حقایق و استراتژی حل مسائل تا رسیدن به نتیجه است.
انواع مختلف استدلال وجود دارد:

- استدلال استنباطی (از معلول به علّت - از کل به جزء)
- استدلال استقرایی (از جزء به کل)
- استدلال انتزاعی
- استدلال قیاسی
- استدلال عقل سلیم
- استدلال غیر یک‌نواخت

در ادامه در خصوص هر یک از موارد بیان شده، توضیحاتی ارائه می‌گردد.

۲-۱- استدلال استنباطی (Deductive Reasoning)

انسان از استدلال استنباطی برای استنباط اطلاعات جدید از اطلاعات منطقاً مرتبط قبلی، بهره می‌گیرد. برای مثال یک کارآگاه جنایی، از مدارک صحنه جنایت و یک زنجیره از فرضیات شروع به تحقیق کرده (استنباط) و او را به تشخیص مجرم هدایت می‌کند.
استدلال استنباطی از حقایق یا بدیهیات و دانش عمومی مرتبط در قالب قوانین یا استنتاج‌های منطقی بهره می‌گیرد. فرآیند با مقایسه بدیهیات (یا حقایق) و مجموعه‌ای از استنتاج‌ها آغاز شده و به حقیقتی جدید ختم می‌شود. برای مثال:

حقایق: اگر در زیر باران ایستاده باشم، خیس خواهم شد.
حقیقت: من زیر باران ایستاده‌ام
نتیجه: من خیس خواهم شد.

این گونه استدلال به سبب منطقی مورد توجه است و یکی‌از تکنیک‌های پرکاربرد در حل مسائل است. قانون «حذف استنتاج» بر این پایه استوار است:

If A is true and if A implies B is true, then B is true

۲-۲- استدلال استقرایی (Inductive Reasoning)

از مجموعه‌ای محدود از حقایق به نتیجه کلی می‌رسیم. «از طریق فرآیند عمومی سازی» مثال:

فرض: میمون‌ها در باغ وحش ارم موز می‌خورند.

فرض: میمون‌ها در باغ وحش کیش موز می‌خورند.

نتیجه: در حالت کلی، تمام میمونها موز می‌خورند.

برای یک مجموعه از اشیاء $X=\{a,b,c,d,\ldots\}$: اگر خصوصیت P برای a درست باشد و اگر خصوصیت P برای b درست باشد و اگر خصوصیت P برای c درست باشد و ... آنگاه P برای تمام X ها درست است.

۲-۳- استدلال انتزاعی (Abductive Reasoning)

استنباط Deductive در حقیقت استنتاجی است که از حقایق و قوانین استنتاجی موجود بدست آمده و بطور منطقی درست است. Abduction گونه‌ای Deduction است که استنتاج محتمل را ممکن می‌سازد. محتمل به این معنی که نتیجه از روی اطلاعات موجود بدست می‌آید اما ممکن است غلط باشد. مثال:

اگر B درست باشد و A نتیجه بدهد B آنگاه A درست است؟

قانون استنتاج: اگر باران ببارد زمین خیس می‌شود.

حقیقت: زمین خیس است.

نتیجه: باران می‌بارد؟!

با داشتن تنها اطلاعاتی که از زمین خیس داریم، یک استنتاج ممکن این است که باران می‌بارد، اما ممکن است اینگونه هم نباشد و زمین به سبب دیگری خیس شده باشد. مثلاً باغبان علفها را آب داده است.

۲-۴- استدلال قیاسی (Analogical Reasoning)

انسان یک مدل مغزی برای برخی مفاهیم که از تجربیاتش بدست می‌آید، می‌سازد. او این مدل را از طریق استدلال قیاسی بکار می‌گیرد تا وضعیت اشیاء را درک کند. بین دو موجودیت قیاس انجام می‌دهد، بدنبال تشابه‌ها و تفاوت‌ها می‌گردد. مثال:

قاب ببر

گروه خاص از : حیوانات

تعداد پا: ۴

غذا: گوشت

محل زندگی: هند و جنوب آسیا

رنگ: زرد مایل به قهوه‌ای با خطوط موازی

انسان یک مدل مغزی برای برخی مفاهیم که از تجربیاتش بدست می‌آید، می‌سازد. او این مدل را از طریق استدلال قیاسی بکار می‌گیرد تا وضعیت اشیاء را درک کند. بین دو موجودیت قیاس انجام می‌دهد، به دنبال تشابه‌ها و تفاوت‌ها می‌گردد.

بطور مثال شیر خیلی مانند ببر است و خیلی مشترکات با ببر دارد: هر دو گوشت خوارند و در هند زندگی می‌کنند. تفاوتهایی نیز دارند برای مثال رنگشان فرق دارد و در نقاط مختلفی زندگی می‌کنند. با این دیدگاه با کمک استدلال قیاسی می‌توان درکی از مفهوم جدید ایجاد کرد و بر اساس تفاوت‌ها و شباهت‌ها آن مفهوم جدید با مفاهیم قبلی آن را شناخت.

۲-۵- استدلال عقل سلیم (Common-Sense Reasoning)

انسان از گونه‌ای استدلال برای افزایش سرعت استدلال بهره می‌گیرد. استدلال عقل سلیم بیشتر روی قضاوت صحیح استوار است تا منطق. مثال: شل بودن تسمه فن رادیاتور معمولاً موجب ایجاد صدای عجیب می‌شود.

یک مکانیک به اینگونه از استدلالها شاید براساس تجربیات سالهای زیاد برسد. و به محض شنیدن صدای ناهنجار تشخیص بدهد که تسمه شل است. این گونه قوانین با گونه دانش که به روش اکتشافی مسئله را حل می‌کند متفاوت است. در روش اکتشافی در حقیقت جستجویی برای یافتن راه‌حل استوار تضمینی وجود ندارد که جهتی که انتخاب شده است جهتی است که حتماً به جواب برسد. همانند جستجوی Best-first Search عمل می‌کند.

۲-۶- استدلال غیر یکنواخت (Non-Monotonic Reasoning)

برای بسیاری از شرایط، بر اساس اطلاعاتی عمل می‌کنیم که ثابت هستند، منظور این است که در طی فرآیند حل مسئله حالت برخی حقایق ثابت می‌ماند. اینگونه استنتاج را یکنواخت می‌گوییم.

اما در برخی از مسائل حالت حقایق تغییر می‌کند. مثال:

اگر باد بوزد، آنگاه پرچم تکان خواهد خورد.

در این شرایط اگر باد بوزد استدلال می‌کنیم که پرچم در حال تکان خوردن است، اما اگر وزش قطع شود، انتظار داریم که دیگر پرچم تکان نخورد. در حقیقت شرایط برای وزش باد تغییر خواهد کرد. لذا استدلال همیشه ثابت نیست(non monotonic reasoning). در این شرایط باید سیستمی باشد که همواره بررسی کند که چه چیزی باعث ایجاد حقیقتی شده و اگر آن شرایط تغییر پذیرند در صورت تغییر باید چه استدلالهایی‌مجدد گرفته شود.

۳- استنتاج (Inference)

تعریف: استنتاج، فرآیند بکار گرفته شده در سیستم خبره است که اطلاعات جدید را از اطلاعات قبلی بدست می‌آورد.

در این بخش به فرآیندهای انجام شده در موتور استنتاج در یک سیستم خبره نگاه خواهیم کرد و بررسی می‌کنیم که چگونه موتور استنتاج می‌فهمد که چه سوالی را از کاربر بپرسد، چگونه در پایگاه دانش حرکت کند، چگونه یک قانون از مجموعه قوانین انتخاب کند و Fire کند و چگونه اطلاعات استدلال شده بر جستجو تاثیر گذارند. در پایان دو دسته تکنیک استنتاجی پیش رو و پس رو را بررسی خواهیم کرد.

۳-۱- حذف استنتاج (Modus Ponens)

تعریف: حذف استنتاج، قانون منطقی است که ادعا می‌کند اگر بدانیم A درست است و از A می‌توان B را نتیجه گرفت، آنوقت B نیز درست است.

با حذف استنتاج از بدیهیات و حقایق به حقایق جدید می‌رسیم. مثال:

$E1 \rightarrow E2$ اگر دمای بدن>۳۵ باشد آنگاه تب دارد

$E2 \rightarrow E3$ اگر بیمار تب داشته باشد آنگاه پیشنهاد می‌شود ۲ آسپرین بخورد

اگر در یکی از بیماران شرایط طوری باشد که: شامل «دما بیشتر از ۳۵ باشد» = E^1

آنگاه می‌توان استدلال کرد که : بیمار تب دارد = E^2

آنگاه می‌توان استدلال کرد که : پیشنهاد می‌شود ۲ عدد آسپرین بخورد = E^3

۳-۲- حل یا رفع (Resolution)

با کمک حذف استنتاج می‌توان از حقایق قبلی به حقایق جدید رسید و این عمل آنقدر ادامه پیدا می‌کند تا تمام اطلاعات ممکن جدید را بتوان بدست آورد. اما در برنامه‌های کامپیوتری بدنبال یک هدف هستیم، و می‌خواهیم واقعیتی را اثبات کنیم.

تعریف: رفع استراتژی استنتاجی است که در سیستم‌های منطقی بکار گرفته شده و صحت یک ادعا را تعیین می‌کند.

روش رفع سعی می‌کند ثابت کند با توجه به حقایق موجود P درست است و همچنین سعی می‌کند ثابت کند که $\neg P$ نمی‌تواند درست باشد. با هر دو روش مجموعه‌ای از حقایق جدید و تئوری‌های نقض بدست می‌آید.

فرض کنید دو ترکیب زیر را داریم:

$\neg B \vee C$ و $A \vee B$ با ترکیب آنها: $(A \vee B) \wedge (\neg B \vee C) = A \vee C$ از طرف دیگر به مجموعه راه‌حل و دانسته‌ها سعی می‌کنیم نتایجی که از نقیض Goal هم بدست می‌آید اضافه کنیم تا در نهایت به

یک تناقض برسیم و بر این اساس ثابت کنیم که عکس Goal نمی‌تواند درست باشد پس Goal درست است.

برای مثال درمان تب بیمار، فرض کنیم می‌خواهیم ثابت کنیم بیمار باید ۲ عدد آسپرین بخورد لذا، ابتدا فرم استنتاج را به فرم OR درآورده خواهیم داشت:

¬E1 ∨ E2

¬E2 ∨ E3

E1

¬E3

با ترکیب دو رابطه اول و دوم خواهیم داشت: $\neg E^1 \vee E^3$

لذا مجموعه دانسته‌ها خواهد شد:

¬E1 ∨ E2

¬E2 ∨ E3

E1

¬E3

¬E1∨ E3

با ترکیب دو رابطه سوم و پنجم خواهیم داشت: E^3

لذا مجموعه دانسته‌ها خواهد شد:

¬E1 ∨ E2

¬E2 ∨ E3

E1

¬E3

¬E1∨ E3

E3

باتوجه به مجموعه دانسته‌ها به این نتیجه رسیدیم که E^3 درست است اما قبلاً فرض کرده بودیم که E^3 غلط است. لذا تناقض خواهیم داشت. پس فرض اول درست نبوده لذا E^3 درست است و پیشنهاد می‌شود که ۲ عدد آسپرین به مریض داده شود.

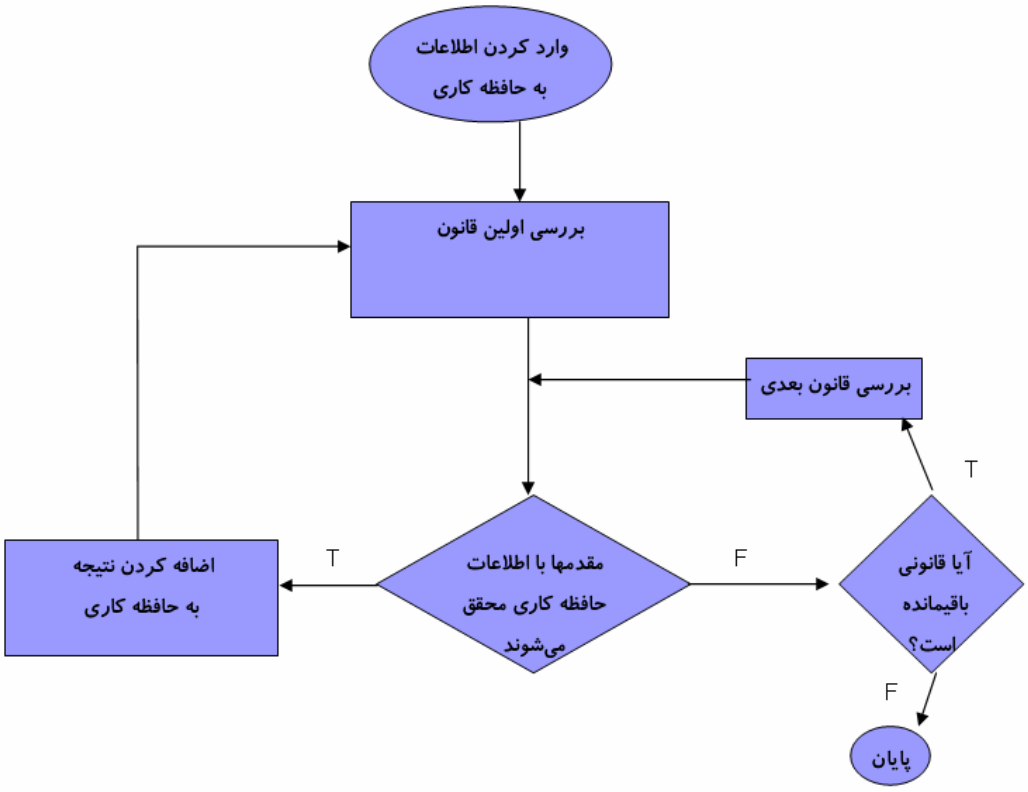
در روش حذف استنتاج، هر چه دانسته‌ها بیشتر باشد، و مجموعه قوانین گسترده، حرکت به سمت دانسته‌های جدید دامنه وسیعی از حقایق جدید ایجاد می‌کند که مدیریت آنها مشکل می‌شود. اما در روش رفع، چون هدف‌دار حرکت می‌کنیم تنها آن قوانین استنتاجی که مرتبط با اثبات قضیه است در نظر گرفته می‌شود.

۴- زنجیره پیش‌رو (Forward-Chaining)

فرآیند یافتن راه‌حل بعضی مسائل از جمع‌آوری اطلاعات شروع می‌شود. روی اطلاعات استدلال صورت گرفته تا به نتایج منطقی برسیم. این شمای حل مسئله که جستجوی داده‌محور است زنجیره پیش‌رو نامیده می‌شود.

تعریف: زنجیره-پیش‌رو، استراتژی استنتاجی است که با مجموعه‌ای معلوم از حقایق شروع کرده و حقایق جدید باکمک قوانینی که مقدم آنها توسط آن حقایق قبلی محقق می‌شود، بدست می‌آید.

این رویه تا رسیدن به حالت هدف یا تا بررسی تمام قوانینی که مقدم آنها توسط حقایق(شناخته شده یا بدست آمده) محقق می‌شود، ادامه پیدا می‌کند.



شکل (۱) نمایش زنجیره پیشرو

فرض کنید بیماری به پزشکی مراجعه می‌کند، مجموعه قوانین پزشک شامل موارد زیر است:

Rule 1

IF بیمار گلو درد دارد

AND مضنون به عفونت باکتریایی است

THEN مطمئناً بیمار گلودرد میکروبی دارد

Rule 2

IF دمای بدن بیمار بیش از ۴۰ درجه باشد

THEN بیمار تب دارد

Rule 3

IF بیمار بیش از یک ماه مریض است

AND بیمار تب دارد

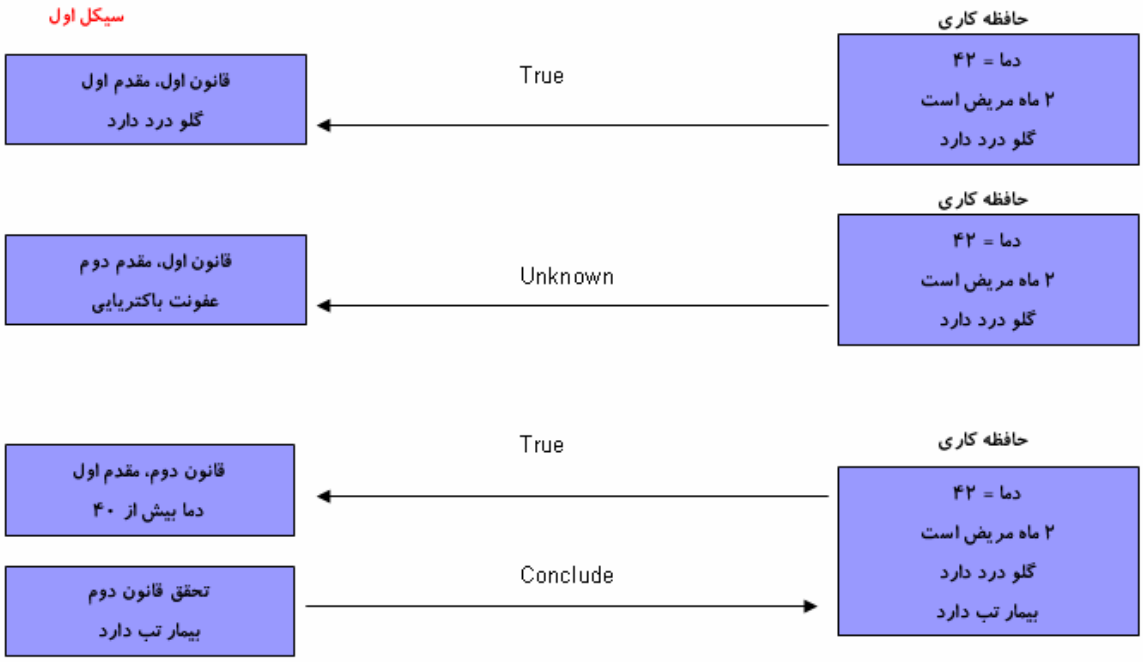
THEN بیمار مضنون به داشتن عفونت باکتریایی است

از طرفی دانسته‌های زیر در حافظه کاری اضافه می‌شود:

دمای بدن بیمار ۴۲ است.

بیمار به مدت دوماه بیمار است.

بیمار درد در ناحیه گلو دارد.



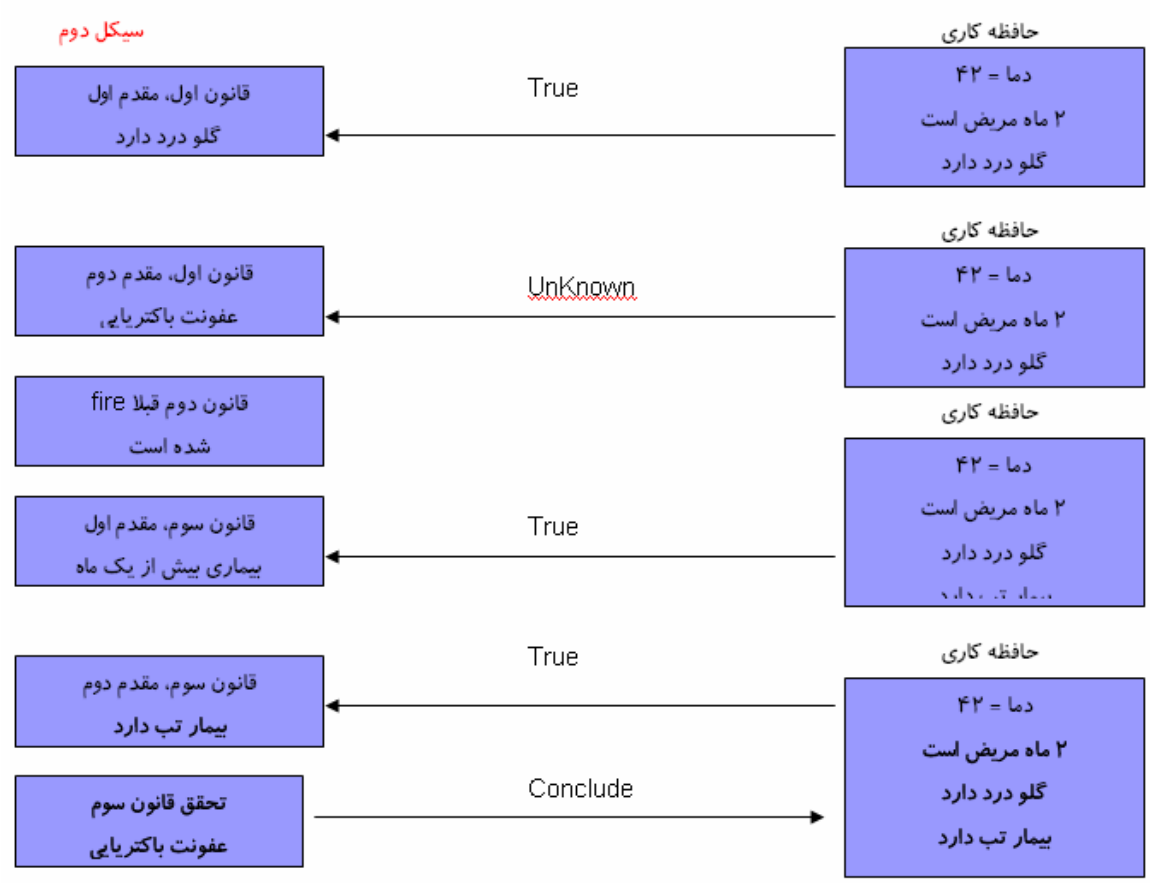
شکل (۲) گام اول در استفاده از زنجیره پیشرو

با کمک نمودار های شکل ۲ تا ۴ که روش زنجیره پیش‌رو را نشان می‌داد، سه حقیقت زیر قابل استنتاج است:

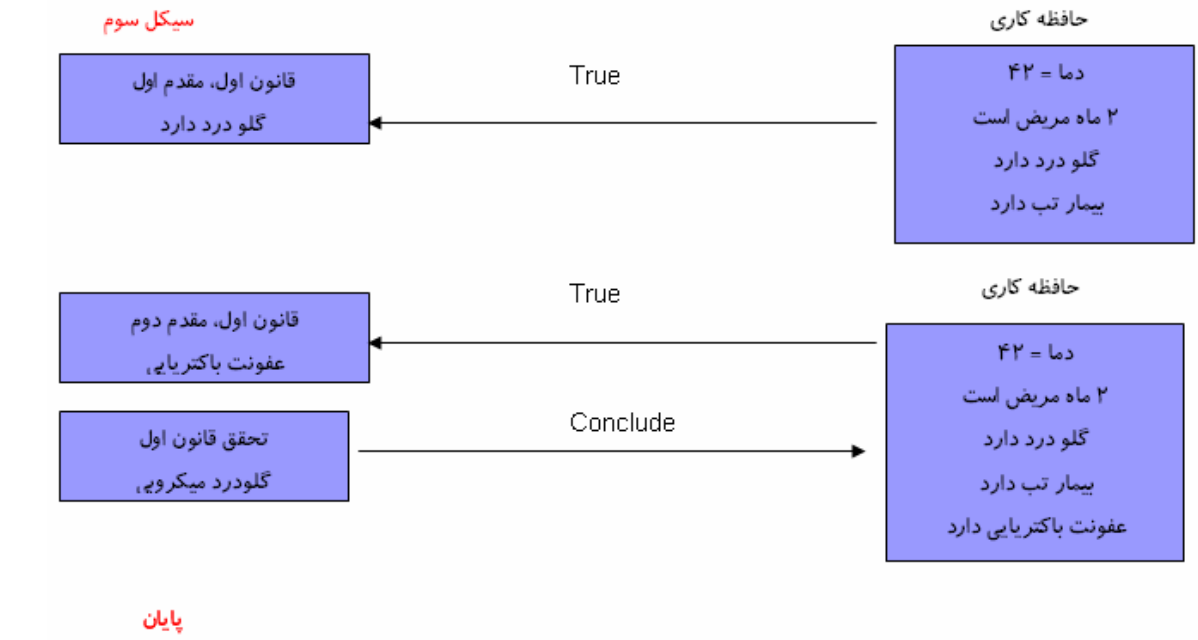
بیمار تب دارد.

بیمار مضنون به بیماری باکتریایی‌است.

مطمئن هستیم که بیمار گلودرد میکروبی دارد.



شکل (۳) گام دوم در استفاده از زنجیره پیشرو



شکل (۴) گام سوم در استفاده از زنجیره پیشرو

در برخی کاربردها این روش بسیار مفید است اما در مسائلی که اطلاعات بدرد نخور زیاد وجود دارد و ممکن است زیاد تولید است کارایی ندارد. مثال:

Rule 4

IF بیمار تب داشته باشد

THEN بیمار باید در خانه بماند

Rule 5

IF بیمار باید در خانه بماند

THEN بیمار می‌تواند کتاب بخواند

با اعمال قوانین فوق به این نتیجه هم می‌رسیم که بیمار باید کتاب بخواند که می‌توان اطلاعاتی بدرد نخور قلمداد کرد.

حل تداخل‌ها:

در مثال قبل دیدیم که قوانین به ترتیب بررسی شده و اولین قانونی که مقدمهایش محقق شدند، نتیجه گیری شده و به مجموعه حقایق اضافه کرده و سیکل بعدی مجدد شروع می‌شود.

در حل برخی مسائل ممکن است به تداخل برسیم، بدین معنا که دو قانون مختلف محقق شده و هدف با دو پاسخ متفاوت بدست آید. برای حل این تداخل، دو استراتژی وجود دارد:

- قوانین به ترتیب اجرا شده و اولین قانونی که fire می‌شود، بعدی دیگر در نظر گرفته نمی‌شود.
- به قوانین درجه اهمیت می‌دهیم (با عددی که به قانون نسبت می‌دهیم وزن دارشان می‌کنیم). لذا در مجموعه قوانینی که می‌توانند محقق شوند قانونی fire می‌شود که وزن قانونش بیشتر است.

تعریف: حل تداخل، استراتژی است که برای انتخاب توالی قانون اجرا شده بکار گرفته می‌شود زمانیکه بیش از یک قانون می‌توانند fire شوند.

لذا در این گونه سیستم‌ها سه فرآیند تشخیص-حل- اقدام وجود دارد:

- تشخیص: مقدم تمام قوانین با حقایق درون حافظه کاری تطبیق داده شده و تمام قوانینی که می‌توانند محقق شوند مشخص می‌شوند.
- حل: اگر بیش از یک قانون اجرا شود، یکی از قوانین بسته به استراتژی مشخص انتخاب می‌شود.
- اقدام: قانون اجرا شده و نتیجه آن به حقایق حافظه کاری افزوده می‌شود.

استراتژی‌های انتخاب قوانین معمولاً شامل موارد زیر هستند:

- اولین قانونی که با حافظه کاری منطبق شود.
- قانونی که بالاترین اولویت را داشته باشد.

- قانون خیلی خاص: قانون خاص‌تر نسبت به قانون کلی‌تر ارجحیت دارد، قانونی خاص‌تر است که مقدم‌هایش بیشتر باشند. ایده اصلی آن است که هر چه قانون از اطلاعات بیشتری استفاده کند، پس دقیق‌تر است.
- قوانینی که به اطلاعاتی اشاره دارد که جدیداً به حافظه کاری اضافه شده: این روش روی اطلاعات جدیدتر جهت‌گیری دارد.
- قانونی که قبلاً اجرا شده، اجرا نمی‌شود. که در تمام سیستم‌ها حتماً وجود دارد و از ایجاد LOOP جلوگیری می‌کند.
- تمام قوانین را اجرا می‌کند اما در خطوط مجزای استدلالی: به این مفهوم که تمام قوانین موازی را اجرا کرده و نتایج آنها را در حافظه‌های کاری موازی اضافه می‌کند تا به هدف‌های متفاوت و گاهاً جایگزین برسد.

۵- زنجیره پس‌رو (Backward-Chaining)

در برخی از مسائل برای حل، از یک تئوری شروع می‌کنیم و سعی می‌کنیم آن را با جمع‌آوری اطلاعات کافی ثابت کنیم. بطور مثال دکتر به نوع بخصوصی از بیماری مشکوک می‌شود و سعی می‌کند علائم آن بیماری را در بیمار جستجو کند. به این جستجوی هدف‌پایه (هدف محور) زنجیره پس‌رو می‌گوییم.

تعریف: زنجیره پس‌رو، استراتژی استنتاجی است که سعی دارد با جمع‌آوری اطلاعات مرتبط، تئوری را اثبات کند.

سیستم زنجیره پس‌رو از یک هدف شروع کرده و سعی دارد آن را ثابت کند. برای این منظور از حافظه کاری شروع می‌کند و نگاه می‌کند که آیا می‌تواند هدف را ثابت کند. اگر ثابت نشده بود، جستجو را آغاز می‌کند. تمام قوانینی که هدف در قست آنگاه آنها قرار دارد- Goal-Rule - را در نظر گرفته، بررسی می‌کند که آیا مقدم آن قوانین در حافظه کاری وجود دارد یا خیر.

اگر مقدمی باشد که در حافظه کاری نباشد، آنوقت خود به عنوان یک هدف جدید یا به تعبیر دیگر زیر هدف در نظر گرفته می‌شود. که خود ممکن است توسط قوانین دیگر اثبات شود. این فرآیند آنقدر ادامه پیدا می‌کند تا مقدمی پیدا شود که توسط قانونی اثبات نمی‌شود که به آن اطلاعات ابتدایی Primitive گفته می‌شود.

تعریف: اطلاعات ابتدایی، مقدم قانونی است که در قسمت آنگاه هیچ قانون دیگری نباشد.

زمانیکه Primitive بدست آمد، سیستم از کاربر برای گرفتن مقدار آن سوال می‌کند، از اطلاعات گرفته شده و عکس رویه که تاکنون طی شده برای اثبات زیر هدف‌ها و هدف اصلی بهره گرفته می‌شود.

فرض کنید بیماری به پزشکی مراجعه می‌کند، پس از توضیحات بیمار پزشک به گلو درد میکروبی مشکوک می‌شود، برای اثبات آن فرض کنید مجموعه قوانین پزشک شامل موارد زیر باشد:

Rule 1

IF نشانه‌هایی از عفونت گلو دارد

AND نشانه‌هایی است که در ناحیه گلو streptococcus داریم

THEN بیمار گلودرد میکروبی دارد

Rule 2

IF اگر گلوی بیمار قرمز است

THEN نشانه‌هایی از عفونت گلو دارد

Streptococcus Bacteria

This scanning electron micrograph shows disease-causing Streptococcus bacteria, commonly found in the human mouth, throat, respiratory tract, bloodstream, and wounds. Often airborne in hospitals, schools, and other public places, Streptococcus bacteria are responsible for infections such as strep throat, scarlet fever, and some types of pneumonia.

شکل (۵) توضیحاتی در خصوص باکتریStreptococcus

۵-۱- دستور العمل هدف (Goal Agenda)

در زنجیره پس‌رو باید حداقل یک هدف داشته باشیم و سعی می‌کنیم تا آن را اثبات کنیم.

در برخی سیستم‌ها مجموعه‌ای از اهداف مد نظر است، که به آن Goal Agenda گفته می‌شود.

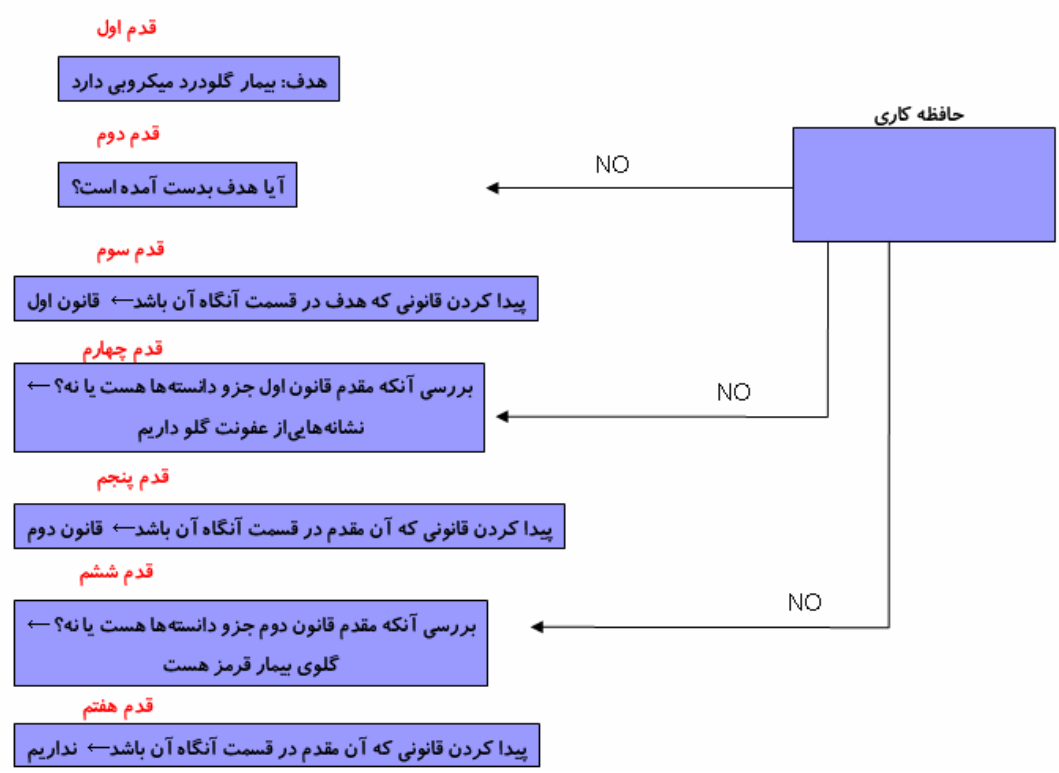
تعریف: دستورالعمل هدف، مجموعه‌ای از اعداف است که باید به ترتیب گفته شده محقق شوند.

مثال:

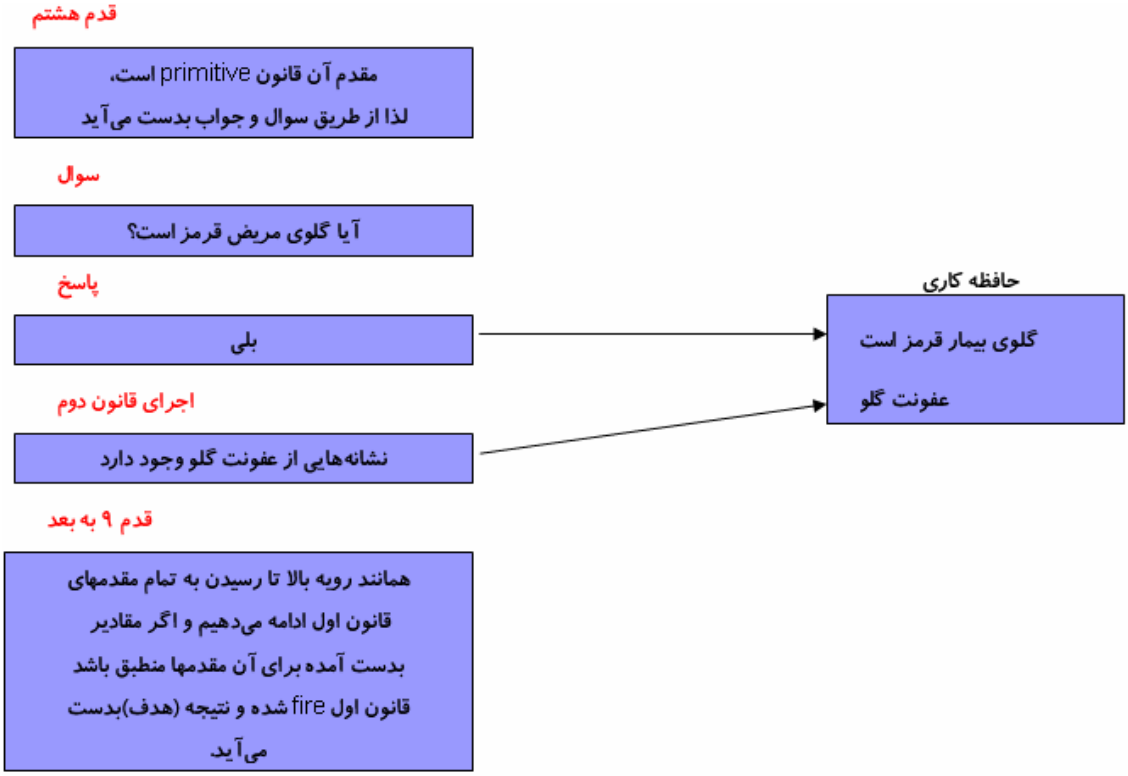
۱- هدف ۱

۲- هدف ۲

۳- هدف ۳



شکل (۶) روش اعمال زنجیره استدلال پسرو مرحله اول



شکل (۷) روش اعمال زنجیره استدلال پسرو مرحله دوم

۵-۲- مزایا و معایب زنجیره پیش‌رو

الف- مزایا: مهمترین مزیت این روش آن است که در مسائلی که بطور طبیعی مسئله از تعدادی اطلاعات جمع‌آوری شده آغاز شده و منتظر می‌مانیم ببینیم چه از آنها می‌توان نتیجه کرد، بسیار خوب عمل می‌کند. از مجموعه‌ای کوچک از حقایق مجموعه‌ای زیاد اطلاعات ایجاد می‌کند.

برای برخی از مسائل مانند: برنامه‌ریزی، مونیتورینگ، کنترل و تفسیر بسیار مناسب است.

ب- معایب: مهمترین عیب زنجیره پیش‌رو آن است که نمی‌تواند تشخیص دهد که برخی اطلاعات مهمتر از برخی دیگر هستند. سیستم سوالات بسیاری می‌کند در حالیکه تعداد کمی از آن سوالات می‌تواند به راه حل منجر شود.

۵-۳- مزایا و معایب زنجیره پس‌رو

الف- مزایا: یکی از مهمترین مزایای این روش آن است که در مسائلی که از یک فرضیه آغاز می‌شوند و بعد بررسی می‌کنند که آیا درست یا خیر، بسیار خوب عمل می‌کند.

در این روش براساس هدف داده شده کار شروع می‌شود. سوالات مرتبط با هدف و اثبات آن است. در این روش تنها بخشی از پایگاه دانش جستجو می‌شود که مرتبط با هدف و اثبات آن باشد. برای گونه‌ای مسائل مانند: تشخیص پزشکی، تجویز پزشکی و رفع نقص بسیار مناسب است.

ب- معایب: پایه‌ای ترین عیب این تکنیک آن است که سیستم تنها یک مسیر استنتاجی داده شده را پی می‌گیرد حتی اگر باید به مسیری دیگر سویچ کند، این کار را تا رسیدن به انتهای آن خط نمی‌تواند تغییر دهد. ضریب اطمینان و فرا-قوانین می‌تواند به حل مسئله کمک شایانی بکنند.

جزوه درس:

سیستم‌های خبره
Expert systems

کتاب مرجع:
Expert Systems Design and Development

نوشته:
John Durkin

مانی عابدینی
تورج بنی‌رستم

فصل پنجم – اکتساب دانش
Knowledge Acquisition

۱- مقدمه

از آنجا که دانش بخش مهم و اساسی سیستم خبره است، نحوه جمع‌آوری و درستی آنها مهم می‌شود. اکتساب دانش ذاتاً عملی پیچیده است.

مهندس دانش باید با فرد خبره تعامل داشته باشد تا دانش مسئله را کسب کرده، سازماندهی کرده و بررسی کند.

ادامه این بخش به بررسی فرآیند اکتساب دانش می‌پردازد و خطوط راهنمایی جهت حرکت از طریق یک تکنیک اکتساب دانش موفق را ارائه می‌نماید.

۲- اکتساب دانش

هدف اکتساب دانش، گردآوری بدنه دانش مسئله مورد نظر و کد کردن آن در سیستم خبره است. منابع این کار می‌تواند، کتابها، گزارشات یا رکوردهای پایگاه دانش باشد. اما مهمترین منبع حاکم، فرد خبره دامنه مسئله است. اکتساب دانش از مفهوم کلی‌تر استخراج دانش Knowledge Elicitation متفاوت است.

استخراج دانش از فرد خبره شامل جلساتی زمانگیر و ملال‌آور بین مهندس دانش و فرد خبره است. این جلسات ممکن است شامل مباحثات دوجانبه باشد. به این گونه از جلسات مصاحبه گفته می‌شود. روش دیگر که بنام case study یا بررسی موردی است، سعی دارد تا دانش فرد خبره را از طریق روش حلش در نمونه‌های مختلف بدست آید.

با هر روشی که باشد مهم آن است که دانش حل مساله بدست آید.

ممکن است اینگونه برداشت شود که فرآیند استخراج دانش سر راست و ساده است. اما اینگونه نیست. در حقیقت سخت‌ترین بخش فرآیند ایجاد یک سیستم خبره است.

۳- انواع دانش

روشهای مختلفی برای استخراج دانش وجود دارد، هر روش گونه‌ای خاص از دانش را می‌تواند بدست آورد. ابتدا انواع دانش بررسی می‌گردد. جدول (۱) انواع مختلف دانش را ارائه می‌دهد.

جدول (۱) انواع دانش

Procedural Knowledge	Rules, Strategies, Agendas, Procedures
Declarative Knowledge	Concepts, Objects, Facts
Meta Knowledge	Knowledge about the other types of knowledge and how to use them
Heuristic Knowledge	Rules of thumb
Structural Knowledge	Rule sets Concept relationships Concept to object relationships

۴- منابع دانش

منابع اطلاعاتی مختلفی در یک پروژه وجود دارد که شامل موارد زیر است:

- فرد خبره: منبع اصلی در یک پروژه سیستم خبره است، مهم خبره‌گی یکتای آن فرد خبره است که کسب شود. البته به بقیه منابع هم باید توجه داشت.
- کاربر نهایی: یکی دیگر از منابع مهم اطلاعات است. فرد خبره معمولاً از سطح پایین با مسئله برخورد می‌کند، جزئیات مهم را می‌بینند. اما کاربر نهایی از سطح بالا به مسئله نگاه می‌کند و مسائل کلی را می‌بینند. زمانیکه آغاز پروژه است مهم است که نظر کلی کاربر نهایی را پیرامون مسئله جویا شویم. کاربر نهایی در ادامه پروژه برای رفع نواقص و کاستی‌های پروژه نیز همواره باید حضور داشته باشد.
- چندین خبره، معمولاً در مسائل از یک فرد خبره بهره می‌گیریم، حسن آن این است که در دانش استخراج شده تناقضی وجود ندارد و البته کار سریع‌تر انجام می‌شود.
  البته باید توجه داشت که در برخی مسائل مجبوریم دانش فرد خبره اول را جهت تایید به چند خبره دیگر نشان دهیم، یا برای حل زیر مسائل جزیی‌تر که از حیطه دانسته‌های فرد خبره اول خارج است به فرد خبره دیگری رجوع کنیم.
- ادبیات تحقیق: منابع دیگری که ممکن است اطلاعات لازم از آنها استخراج شود، شامل گزارشات، آیین‌نامه‌ها، دستورالعمل‌ها و کتابها باشد. اگر چنین منابعی موجود باشند، حتماً به آنها باید رجوع شده و دید کلی از مسئله بدست آید.

۵- مراحل استخراج دانش

فعالیت‌هایی که در استخراج دانش وجود دارد، یک سیکل طبیعی است که از جمع‌آوری دانش شروع شده، با تفسیر و تحلیل آنها ادامه پیدا کرده و در نهایت، روشهایی برای جمع‌آوری بیشتر دانش طراحی می‌شود.

۵-۱- جمع‌آوری

بدست آوردن دانش از فرد خبره است. به توانایی برقراری ارتباط با دیگران احتیاج دارد و نیازمند آن است که فرد خبره را مجاب به همکاری کنید. در ابتدای پروژه کلیات و مفاهیم اولیه مسئله را کسب می‌کنیم و در جلسات بعدی اطلاعات خاص‌تر بدست می‌آید.

۵-۲- تفسیر

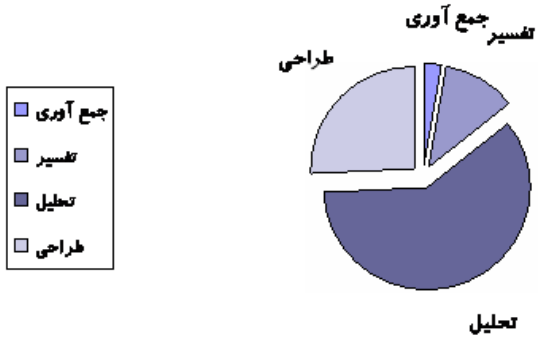
این مرحله شامل مروری روی اطلاعات جمع‌آوری شده و شناسایی دانش کلیدی است. در مراحل اولیه که اطلاعات جمع‌آوری شده کلی‌تر است حاصل این مرحله می‌تواند شامل: هدف مسئله، قیدها و دامنه مسئله باشد. در مراحل بعدی گونه‌های مختلف دانش دیگر مد نظر خواهد بود.

۵-۳- تحلیل

از تحقیقات انجام گرفته در مرحله تفسیر، بخش‌های اطلاعات کلیدی مشخص می‌شود و استراتژی حل مسئله و سازمان دانش تا حدودی معیین می‌شود. در مراحل اولیه مفاهیم مهم توسط فرد خبره ارائه می‌شود. ارتباط مفاهیم و اینکه چگونه آنها را برای حل مسئله بهم مرتبط کنیم، را باید در این مرحله بدست آوریم. در مراحل بعدی همین کار را با جزئیات بیشتر ادامه می‌دهیم. ساختار ارائه دانش در این مرحله انتخاب می‌شود.

۵-۴- طراحی

از طی مراحل قبلی می‌توان طرحی جهت ادامه کار و کسب اطلاعات بیشتر ارائه کرد که در این مرحله صورت می‌گیرد. فرآیند اکتساب دانش شرط خاتمه ندارد. هرچه دانش بیشتری استخراج شده و به سیستم خبره اضافه شود، سیستم کامل‌تر می‌شود. نمودار زیر زمان لازم جهت انجام هریک از فرآیندهای اکتساب دانش را نشان می‌دهد.



شکل (۱) زمانبندی مراحل اکتساب دانش

۶- مشکلات اکتساب دانش

به لحاظ روانشناسی: مطالعات روانشناسی نشان می‌دهد که انسان بطور عقلی در خیلی از فعالیت‌هایش آگاه نیست. حتی از نحوه فرآیند مغزی خود در حل مسئله مطلع نیست. از طرفی برخی گفته‌اند که انسان نمی‌تواند دانش خود را منتقل کند، نه به این خاطر که نمی‌توانند بیان کنند، بلکه به این سبب که دقیقاً نمی‌دانند چه دانشی را در فعالیت‌هایشان بکار می‌گیرند.

حتی اگر بتواند دانش خود را منتقل کند، لزوماً وابستگی بین آنچه بیان شده با رفتار مغزی وجود ندارد. به تعبیری انسان کارهایی را به روشهای مختلفی انجام می‌دهد و اساس هر روش کارایی آن روش است.

اثبات شده که حافظه کوتاه مدت انسان گنجایش کمی دارد، لذا اطلاعاتی معدودی در آن قرار داده می‌شود، مشکلات حافظه کوتاه مدت انسان منشا اشکلات منطقی است که زمانیکه انسان می‌خواهد فرآیند استدلالی حل مسائل قدیمی‌را توضیح دهد، رخ می‌دهد.

برخی از مشکلات و سختی‌های روانشناختی اکتساب دانش به شرح زیر است:

- فرد خبره از دانشی که استفاده می‌کند بی‌خبر است.
- فرد خبره توانایی بیان دانش خود در قالب جملات را ندارد.
- فرد خبره ممکن است دانش غیر مرتبط ارائه دهد.
- فرد خبره ممکن است دانش ناقص ارائه دهد.
- فرد خبره ممکن است دانش ناسازگار ارائه دهد.

از دانش داشته، باخبر نیست: اغلب افراد خبره دانش خود را از طریق تجربه بدست می‌آورند. بعد از مدتی فرد خبره از عمق دانشی که استفاده می‌کند باخبر نیست، فرد خبره دانش خود را به فرم فشرده در می‌آورد و در هنگام حل مسئله به سرعت و کارایی بالا عمل کند. این گونه دانش، دانش سایه، نامیده می‌شود و بیشتر از آنکه بر اساس اصول اولیه باشد بر پایه تجربه است.

دانش بیهوده ارایه می‌کند: یک جلسه یک ساعته مصاحبه با فرد خبره به راحتی ۱۰ صفحه متن خواهد شد. ابتدای پروژه چون هنوز در مورد پروژه آگاهی لازم نداریم، باید فرض کنیم که تمام اطلاعات داده شده توسط فرد خبره مهم است، مگر آنکه از فرد خبره بپرسیم و او مشخص کند که چه اطلاعاتی اهمیت ندارد. زمانیکه به دامنه مساله مسلط شدیم، کم کم خواهیم توانست خود بخشی از اطلاعات را حذف کنیم، چون کمکی به حل مسئله نمی‌کنند.

ارائه دانش ناکامل: در بیان دانش گاهی فرد خبره از مطالبی که برای خودش کاملاً هضم شده است به راحتی عبور می‌کند. در حقیقت بدنه دانش ناقص خواهد بود، مهندس دانش باید در طی فرآیند تحلیل این نواقص را مشخص کرده و دانش را کامل کند.

فراهم کردن دانش غلط: ممکن است فرد خبره از دانش لازم برخوردار نباشد یا ممکن است یک اشتباه کوچک رخ داده باشد تا در استراتژی حل مسئله که همیشه به خوبی جواب می‌داده خلل ایجاد شود.

همه آنچه را که فرد خبره می‌گوید باور نکنید.

سعی کنید در مورد صحت دانش فرد خبره بررسی کنید. البته این مرحله اتوماتیک در زمان تست سیستم خبره انجام خواهد شد.

ارائه دانش ناسازگار: در برخی مواقع تناقض‌هایی در دانش استخراج شده وجود دارد که حتی مهندس دانش هم متوجه آن نمی‌شود. مثلاً مکانیک می‌گوید: اول همیشه باطری ماشین را بررسی می‌کنم. اما در ادامه ممکن است او بگوید، اول باید بررسی کرد که سیمی اتصالی ندارد. در بیان اهمیت موضوعات، معمولاً ناسازگاری رخ می‌دهد.

۷- مصاحبه

اغلب روشهای استخراج دانش را می‌توان به دو گروه مستقیم و غیر مستقیم تقسیم‌بندی کرد. در روش مستقیم، از روشهای مصاحبه و case study بطور مشخص بهره گرفته می‌شود و تعامل نزدیک با فرد خبره جهت کسب دانشش وجود دارد. در روش غیر مستقیم از روش پرسشنامه بطور مشخص بهره می‌گیریم.

روش مصاحبه از رایج‌ترین روشهاست لذا به بررسی آن خواهیم پرداخت.

مرحله استخراج دانش فرآیندی است که به همکاری گروهی نیازمند است، اگر یکی از اعضای تیم همکاری مناسب نداشته باشد احتمال شکست پروژه بالا خواهد رفت.

در مورد سازمان و صورت مسئله آگاهی لازم کسب شود: قبل از اولین ملاقات باید راجع به سازمان و مشکلات آن تحقیق شود،که معمولاً سازمانها در غالب گزارشاتی خود مشکلاتشان را دسته بندی کرده و آماده دارند. لازم به تحقیقی عمیق نیست، یک نگاه کلی می‌تواند به شما این امکان را بدهد که پروژه را بطور کلی تعریف کنید و با اصطلاحات فرد خبره نیز آشنا شوید. باید بدنبال توانایی‌های بالقوه سیستم خبره در حل مشکلات سازمان باشید. مشکلاتی که قابل دسترس‌تر هستند به عنوان اهداف اصلی در نظر گرفته شده و برای اولین جلسه ملاقات آماده می‌شویم.

اولین ملاقات تیم: همکاری بالای اعضای تیم در طول پروژه به اولین برخورد آنها با پروژه بستگی دارد. مدیریت اولین نشست بطور چشم‌گیری در افزایش یا کاهش موفقیت آینده پروژه موثر خواهد بود. برای توضیحات بیشتر می‌توان به موارد زیر اشاره کرد:

- زدودن ترس:
  - سعی کنید این مفهوم را منتقل کنید که سیستم خبره یک جایگزین نیست و تنها کمکی جهت بالا رفتن کالایی فرد خبره خواهد بود.
  - باورهای غلط را اصلاح کنید و سعی کنید توضیح دهید که با آمدن سیستم خبره چه تغییراتی ایجاد خواهد شد و چه تغییراتی ایجاد نخواهد شد.
- رفع شک و شبه:
  - نگاهی اجمالی و مختصر از سیستم خبره ارائه دهید.
  - مروری روی سیستم‌های خبره موفق در همین زمینه کاری ارایه کنید.
  - تجربیان موفق متعدد در سیستم‌های خبره را معرفی کنید.
- تعریف اهداف منطقی:
  - در مورد سیستم زیادی خیال پردازی نکنید تا افراد تشویق نابجا نشوند.
  - توانایی‌ها و محدودیتهای سیستم را توضیح دهید.
- ترویج تمایل به تغییر:
  - توضیح دهید که همکاری هر فرد گروه چه میزان می‌تواند در توسعه آینده و قبول این تکنولوژی موثر باشد.
  - توضیح دهید که با معرفی سیستم خبره برخی ناسازگاری‌ها رخ خواهد داد که مصمئناً موقت خواهد بود.
- ارائه مفهوم تلاش مورد انتظار:
  - همکاری مورد انتظار از هر فرد گروه را توضیح دهید.
  - توضیحاتی مکتوب از وظایف هر فرد پروژه به مدیران ارایه کنید.
- مکتوب کردن اهمیت درگیر شدن در پروژه:
  - اعضای تیم را آگاه سازید که آنها بخش مهم پروژه هستند و بدون تلاش آنها پروژه شانس کمی در موفقیت دارد.

۷-۱- توصیه‌هایی برای جلسه اول

هدف اصلی اولین جلسه ایجاد هماهنگی لازم بین اعضای تیم و بالابردن اعتبار موفقیت پروژه است.

- برای جلسه آماده شده باشید: در مورد مساله تحقیق لازم صورت داده باشید و در مورد سازمان آگاهی لازم را کسب کرده باشید.
- هر عضو گروه را تشویق کنید تا پروژه را از دیدگاه خودش ارائه کند.
- هر گونه مقاوت در برابر پروژه را تشخیص داده و برای آن برنامه ریزی کنید.
- خلاصه‌ای از سیستم‌های خبره ارائه کنید.

- جهت‌گیری آتی پروژه را مشخص کنید.
- از جلسه نسخه ضبط شده داشته باشید تا بعداً آن را مرور کنید.

۷-۲- آماده‌سازی برای مصاحبه

موفقیت در مصاحبه تا حد زیادی به آماده‌سازی مقدمات آن بستگی دارد. در این مرحله موضوعاتی که باید اشاره شود و برنامه‌ریزی و ابزارآلات لازم و روشهای ضبط جلسات باید بطور دقیق مورد بررسی قرار گیرند.

- تهیه دستور جلسه: یک مصاحبه کننده خوب کسی است که می‌داند هدفش چیست و چگونه به آن برسد. دستور جلسه به شما این کمک را می‌کند که مراحل رسیدن به هدف را مشخص کرده و لیستی از موارد مهمی که، در مورد آنها باید بحث شود مشخص گردد. دستور جلسه به اعضای دیگر تیم کمک می‌کند تا بفهمند که چه هدفی را در طی مصاحبه دنبال می‌کنید و این شانس را به آنها می‌دهید تا خود را از قبل آماده کنند، در نهایت شانس موفقیت جلسه بالا می‌رود.
- دیر نکنید: دیر کردن نه تنها چهره نامناسبی از شما در دیگر اعضای تیم ایجاد می‌کند بلکه، نشان می‌دهد که برای وقت آنها اهمیت قائل نیستید و شاید پروژه را جدی نگرفته‌اید. در نظر داشته باشید که اغلب فعالیت شما به تنهایی و بدون نظارت اعضای دیگر انجام خواهد گرفت لذا برداشت آنها تنها در جلسات مصاحبه شکل می‌گیرد. سعی نکنید نگاهی منفی از خود در این جلسات ایجاد کنید. مصاحبه باید بطور طبیعی‌در محل استقرار فرد خبره انجام گیرد، علاوه بر نشان دادن نوعی احترام به مصاحبه شونده، خود به نوعی آشنایی با محیط کاری و مشکلات و نحوه حل آنها نیز هست. در حقیقت هر فعالیتی وابسته به محیط فعالیت است، اگر سعی کنید فعالیت را خارج از محیطش مورد بررسی قرار دهید در حقیقت معنایش را از آن گرفته‌اید. زیرا فرد خبره در آن محیط است که می‌تواند در مورد مسائل تصمیم گیری کند و اجازه جمع‌آوری اطلاعات را به شما می‌دهد.
- لیست ابزار لازم: باید ابزار و مایحتاج لازم در طول مصاحبه را لیست کنید، مشتمل بر گزارشات، اوراق و ... و قبل از مصاحبه به اعضای تیم داده باشید. معمولاً از اطلاعات بحث شده در جلسات قبلی بحث آغاز می‌شود، لذا روند مصاحبه را مستند کرده و به مصاحبه شونده نیز یک نسخه از آن را بدهید. باید تمام اطلاعات جلسات مصاحبه مستند سازی شود، لذا از نکات بحث شده، پاسخ سوالات و نکات مبهم یادداشت بردارید. مطمئناً می‌دانید که دفترچه با صفحات لازم، چند خودکار یا مداد و مارکرهای مختلف رنگی برای جلسه مصاحبه الزامی است.
- ضبط صدای جلسه: یادداشت‌برداری محدودیتهایی دارد و معمولاً کامل نیست، برخی اطلاعات فراموش شده و از قلم می‌افتد. مثلاً فرد خبره «احتمالاً» را مطرح می‌کند که شما یادداشت نمی‌کنید و در نهایت در قوانین CF لازم در نظر گرفته نمی‌شود. در نظر داشته باشید که ضبط

کردن جلسات به شما این امکان را می‌دهد که بعداً با مرور آن، نکات جا افتاده را پیدا کنید. البته ممکن بکار گیری تکنیک ضبط صدای جلسه مصاحبه کننده را وسوسه کند تا دیگر یادداشت برداری نکند، که اینهم غلط است. زیرا ممکن است فرد خبره در طی جلسه روی نکاتی تاکید داشته باشد که در صدای ضبط شده این تاکید واضح نباشد، لذا هر دو روش باید در کنار هم قرار گیرد. سعی کنید هر جلسه را جداگانه و در فایل مشخصی نگهداری کنید تا در صورت لزوم جستجوی آن با مشکل مواجه نشود. همین طور در نظر داشته باشید که کیفیت ابزار ضبط کننده خوب باشد چون امکان برگزاری مجدد جلسه خیلی کم است.

- ضبط تصویر جلسه: برخی مصاحبه کننده‌ها از ابزار ضبط تصویری برای ضبط محتوی جلسه بهره می‌گیرند. این بستگی به آن دارد که چقدر اطلاعات دیداری در کارایی پروژه و تجزیه و تحلیل‌های آتی می‌تواند موثر باشد. اگر در پروژه‌ای فرد خبره فعالیت‌های فیزیکی دارد، مطمئناً ضبط تصویر فرد خبره لازم است. البته باید در نظر داشت که افراد معمولاً در برابر دوربین راحت نیستند و ممکن است به گونه‌ای رفتار حقیقی خود را نشان ندهند و حتی در ارائه دانش خلل ایجاد شود. لذا تنها در صورتی از ابزار تصویری بهره بگیرید که به گونه‌ای دیگر نمی‌توانید اطلاعات را ذخیره کنید.

۷-۳- نکات مهم در آماده‌سازی جهت مصاحبه

موفقیت یک جلسه مصاحبه بسته به تلاش انجام گرفته در زمان آماده سازی دارد. رعایت نکات زیر در فرآیند آماده‌سازی می‌تواند شانس موفقیت را افزایش دهد:

- تعیین هدف مشخص و دستور جلسه
- مشخص کردن شرکت کنندگان در جلسه
- به شرکت کنندگان اطلاعات زیر داده شود:
  - اهداف
  - گزارشات و اسناد
  - مباحث اصلی
- مشخص کردن زمانبندی جلسه مصاحبه:
  - به شرکت کنندگاه از یک هفته قبل اطلاع دهید.
  - طول جلسه مصاحبه حدود یک ساعت در نظر گرفته شود.
  - هرگز دیر سر جلسه نرسید.
- در نظر گرفتن مکان مصاحبه
- ضبط مصاحبه:

- برای هر جلسه یک نوار مجزا در نظر گرفته شده و روی نوار مشخصات تاریخ، مکان، شرکت‌کنندگان و شمای کلی جلسه را یادداشت کنید.
- موضوعات مهم جلسه را یادداشت برداری کنید.
- تنها زمانی از ضبط تصویری جلسه بهره بگیرید که واقعاً لازم باشد.

- آوردن ابزار و وسایل لازم به جلسه:
  - دستور جلسه
  - گزارشات و اسناد و غیره
  - مستندات پروژه
  - موارد جانبی مانند:
    - دستگاه ضبط صدا
    - نوار و باطری اضافی
    - دفترچه یادداشت
    - مارکر، خودکار و مداد.
  - اگر کمی بخواهید جلسه برای مصاحبه شونده جذاب تر باشد، مقداری میوه و شیرینی و ...

۷-۴- آغاز جلسه مصاحبه

جلسه باید طوری آغاز شود که شرکت‌کنندگان احساس راحتی کنند، لذا ابتدا می‌توان با سوالات حاشیه‌ای، میزان اهمیتتان به شخص مصاحبه شونده را نشان دهید »مثلاً سفرتان چطوری بود». با پاسخ به این سوال مصاحبه شونده احساس آرامش بیشتری با محیط پیدا خواهد کرد و آماده پاسخ گویی به سوالات جدی‌تر می‌شود. سعی کنید سوالات ابتدایی به هیچ وجه به پروژه مرتبط نباشند، چون ممکن است برگرداندن بحث به مسیر برنامه‌ریزی شده مشکل شود.

دستور جلسه را مرور کنید: البته قبلاً این دستور جلسه را به مصاحبه شونده نیز داده‌اید لذا مروری سریع کافی است. به سوالات و سو تفاهم‌های احتمالی برای شروع مصاحبه پاسخ گویید.

۷-۵- انجام مصاحبه

دنبال کردن دستور جلسه: برای رسیدن به هدف تعیین شده رعایت این امر ضروری است. معمولاً فرد خبره مسیر صحبت را به سمتی می‌برد که علاقه شخصی‌اش را دنبال کند، که هنر مصاحبه کننده آن است که مسیر بحث را کنترل کند. البته باید توجه داشت که مقید بودن زیاد به دستور جلسه، ممکن است جلوی کشف برخی جنبه‌های ناشناخته پروژه را نیز بگیرد. استخراج دانش اقدامی است در جهت کسب دانش مرتبط با پروژه.

شکست در رسیدن به اهداف تعیین شده نیست، بلکه شکست در نرسیدن به تعدادی دانش است.
نوع سوالات: سوالات می‌تواند برای کسب اطلاعات باشد یا برای جهت دهی به بحث، پس باید از انواع و کاربردهای هر گروه آگاه باشید.

- سوالات مستقیم: اگر به دنبال اطلاعات خاصی هستید، می‌توانید از سوالاتی همانند: معنی ... چیست؟ آیا .... درست است؟ ارزش ... در چیست؟ بهره ببرید.
- بطور مثال: فشار غیر نرمال پمپ را چگونه تعبیر می‌کنید؟
- پاسخ سوالات مستقیم معمولاً صریح و کوتاه است، حتی در حد بلی یا خیر. از سوالات مستقیم معمولاً در ابتدای پروژه استفاده نمی‌شود، بلکه بیشتر در ادامه پروژه رایج است.
- سوالات غیر مستقیم: اینگونه سوالات اکتشافی هستند و به فرد خبره این امکان را می‌دهند تا به روش خودش به آنها پاسخ گوید. با سوالات غیر مستقیم سعی داریم تا مفاهیم اصلی و دانش رویه حاکم را که توسط فرد خبره بکار گرفته می‌شود کشف کنیم. گونه سوالات: چه مباحثی را در نظر می‌گیرید اگر .... ؟ چگونه تعیین می‌کنید که .... ؟ بدنبال چه می‌گردید وقتی که .... ؟

برای مثال: برای آنکه مشخص کنید نیاز به تعویض کردن پمپ است چه مباحثی را در نظر می‌گیرید؟

پاسخ می‌تواند: اگر فشار پمپ کم باشد، یا دمایش بالا یا عمرش زیاد باشد باید آن را جابجا کرد.

از پاسخ داده شده مفاهیم مهم مانند فشار پمپ، دما و عمر بدست می‌آید. در مرحله بعد می‌توان از سوالات مستقیم برای فهم بیشتر این مفاهیم بهره برد.

- سوالات تحقیقی: اگر احساس می‌کنید فرد خبره نکاتی را جا انداخته یا به واسطه فرآیند ذهنی و فشرده کردن دانش، از روی نکاتی پریده است، با سوالاتی مانند: لطفاً ... را توضیح دهید؟ اگر امکان دارد بیشتر در مورد ... بگویید؟، اطلاعات بیشتری بگیرید.
- سوالات بیدرنگ: زمانیکه بخواهیم جریان بحث را عوض کنیم از سوالاتی مانند: می‌شود در مورد ... بحث کنیم؟ اگر امکان داشته باشد برگردیم به .... ؟

۷-۶- طراحی درست سوالات

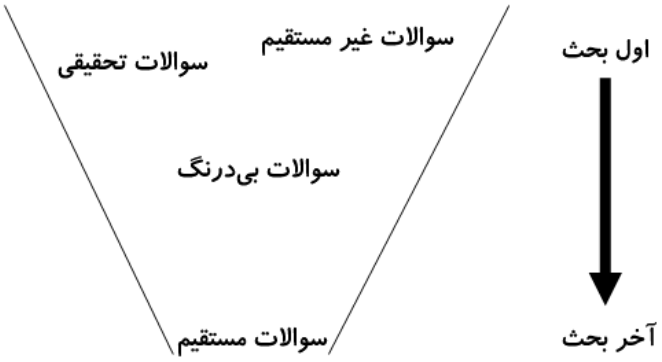
زمانیکه هدف از سوال مشخص شد، طریقه بیان آن نیز مهم می‌شود، برای بیان سوالات مناسب، خوب است نگاهی به سوالات نامناسب بیاندازیم که به طور گسترده و روزمره از آنها استفاده می‌کنیم:

- از سوالات چند بخشی دوری کنید: در یک سوال بیش از یک مبحث پرسیده نشود مثلاً: چرا هم دما و فشار پمپ .... ؟ اینگونه سوالات فرد خبره را گیج می‌کند. زیرا آنوقت فرد خبره نمی‌داند که شما بدنبال فشار پمپ هستید؟ یا دمای پمپ؟ یا رابطه بین آنها. حتی ممکن است در هنگام پاسخ دادن، فرد خبره مباحث مربوط به فشار پمپ را فراموش کند.
- از پرسش سوالات جهت‌دار بپرهیزید: در سوالات نظرات شما نباید دخیل باشد و جهت گیری اولیه به فرد خبره بدهد. بطور مثال: آیا اول فشار پمپ را چک نمی‌کنید؟
- از سوالات منفی بپرهیزید: اینگونه سوالات گیج کننده هستند و تفسیر آنها ممکن است فرد خبره را دچار مشکل کند. کلاً از سوالاتی که چندین نفی در آنها بکار رفته بپرهیزید. مثلاً : آیا غلط است اگر بگوییم «فشار آب بالا نیست پس اشکال ندارد».
- از پرسش سوالات خیلی کلی بپرهیزید: اینگونه سوالات شامل کلمه «همیشه» هستند. بهتر است در سوال پرسیده شود هر چند وقت .... ؟
- از بکارگیری واژه‌های غریب بپرهیزید: برخی اصطلاحات تخصصی مرتبط و آشنا، برای فرد خبره کاملاً غریب است. لذا ممکن است ایجاد نوعی سو تفاهم کند. مثلاً: آیا شما همیشه از استنتاج پیش رو در حل مسئله بهره می‌گیرید؟
- از بکارگیری «اصطلاحات ذهنی هر فرد» بپرهیزید: منظور از این اصطلاحات آن دسته‌ای است که هر فرد تعبیر خودش را از آن اصطلاح دارد. مثلاً: اگر پمپ واقعاً داغ باشد، آن را سریع خاموش می‌کنید؟

هر فردی از مفهوم خیلی داغ و سریع برداشتی متفاوت دارد.

۷-۷- ترتیب سوالات

اگر از تکنیک ترتیب قیفی[۱] در جهت ترتیب‌دهی به سوالات کمک بگیرید کارایی بیشتری بدست خواهید آورد. در این روش ابتدای بحث باید سوالات کلی مطرح شود و در طی بحث به سمت سوالات تخصصی‌تر حرکت کنیم.



شکل (۲) تکنیک ترتیب قیفی

[۱]. Funnel Sequence

این روش از کل به جز رسیدن است. می‌تواند در پایان بحث، بحث جدیدی را شروع کرده و مجدد روش فوق را اعمال کنید.

۷-۸- گوش دادن فعّال

در طول مصاحبه باید بیاموزید که شنونده فعّالی باشید تا آنکه فقط سوال کنید و پاسخ‌ها را ضبط کنید. برای این منظور ممکن است گاهی مجبور شوید پاسخ فرد خبره را تفسیر کنید تا به درک درست برسید. باید میزان اهمیت پاسخ هر بحث را نیز بدست آورید. گاهی ممکن است مجبور شوید خلاصه‌ای از آنچه که فهمیده‌اید را برای فرد خبره بازگویید تا برداشتتان تایید شود.

۷-۹- پایان مصاحبه

نتیجه تحقیقات روانشاسی نشان می‌دهد که افراد اغلب ابتدا و انتهای یک جلسه را بخوبی بخاطر می‌سپارند. اگر اختتام جلسه به خوبی انجام شود، فرد خبره از آن جلسه همیشه به خوبی یاد خواهد کرد.

در انتهای جلسه مروری روی نکات بدست آمده و نظرات کارشناسان داشته باشید. اشاره‌ای به سوالات مهمی که در طی جلسه بوجود آمد بکنید. در مورد دستور جلسه بعدی و در کارهایی که اعضا باید تا جلسه بعدی انجام دهند بحث کنید.

در پایان جلسه از همکاری اعضا و دادن زمان به شما تشکر کنید.

۸- بررسی موردی

به سبب برخی مشکلات و سختی‌های روش مصاحبه، اکثر مهندسان دانش روش بررسی موردی، را ترجیح می‌دهند. یک مورد، مشکل واقعی است که قبلاً حل شده و هم راه‌حل و هم قدمهای انجام گرفته در حل آن را داریم. دو خط‌مشی اصلی وجود دارد:

- خط‌مشی مرور خاطرات: از فرد خبره می‌خواهیم که بررسی کرده توضیح دهد که در مورد قبلی چگونه صورت مسئله را حل کرده است.
- خط‌مشی مشاهده: در این روش از فرد خبره می‌خواهیم در حین مشاهده شما، توضیح دهد که چگونه مسئله را حل می‌کند.

انواع موارد: در هر دو خط‌مشی مطرح شده فوق، دو نوع اصلی مورد را باید در نظر گرفت:

- مورد آشنا
- مورد غیر متعارف

در مورد آشنا، یا مورد معمول، که برای فرد خبره کاملاً شناخته شده است. بررسی این موارد دانش معمول فرد خبره در حل مسائل را آشکار می‌کند و در ابتدای پروژه که کلیات و دانش عمومی باید استخراج شود زیاد بکار گرفته می‌شود. اما موارد نامتعارف، زیاد رایج نبوده و در حل آن نیاز به

نوعی ابتکار احساس می‌شود. از این جهت است که فرد خبره را مجبور به تفکر و بررسی بیشتر می‌کند و جنبه‌های پنهان دانش فرد خبره آشکار می‌شود. زمانیکه نیاز به بهبود دانش استخراج شده احساس شود از اینگونه موارد استفاده می‌کنیم.

پروتکل: مجموعه‌ای از اطلاعات جمع‌آوری شده از کارهای انجام گرفته مجزا روی یک صورت مسئله را پروتکل می‌نامیم. یک پروتکل رکوردی است که رفتار فردی را در حل مسئله دنبال می‌کند. یک پروتکل شامل مجموعه‌ای از یادداشت‌ها، ضبط صدا، نوار ویدیویی یک جلسه است.

۸-۱- خط‌مشی موردی با مرور خاطرات

هدف این روش آن است که از مرور موارد موفق قبلی دانش استخراج گردد. بعد از آنکه مورد انتخاب شد از فرد خبره بخواهید تا مسئله را حل کند. در وضعیت فرد خبره هدف محور حرکت می‌کند. ابتدا اطلاعات جمع‌آوری شده را مرور می‌کند و بعد سعی می‌کند نشان دهد که چگونه از آنها برای حل مسئله کمک می‌گیرد. نکات مهمی که باید در نظر بگیرید:

- اغلب افراد خبره از یک رویه کاملاً واضح برای حل بهره می‌گیرند.
- دقت کنید که فرد خبره چگونه اطلاعات جمع‌آوری کرده و از آنها برای حل چگونه کمک می‌گیرد.
- چگونه فرد خبره راه‌حل‌ها را دسته‌بندی کرده و شناسایی می‌کند.
- برخی مفاهیم مهم را نیز بیان می‌کند که باید به آنها حتماً توجه شود.

۸-۲- مزایا و معایب روش مرور خاطرات

- مزایا: دیدی بهتر نسبت به حل مسئله در مقایسه با روش رایج مصاحبه، خواهد داد. در این روش چون بیشتر تاکید روی اطلاعات لازم جهت حل مسئله قبلی است. فرد خبره کمتر با خود فرآیند استنتاجی درگیر می‌شود و از طرفی شرایطی غیر طبیعی ایجاد نمی‌کنیم تا فرد خبره آن را حل کند و نتایج عجیب نیز نخواهیم گرفت. از طرفی در این فرآیند معمولاً به مفاهیم کلیدی بهتر اشاره می‌شود.
- معایب: مهمترین عیب آن است که چون بر اساس مرور خاطرات است، ممکن است زیاد کامل نباشد چون فرد خبره برخی جزئیات را از خاطر برده است. چون این روش اصولاً گونه‌ای فرآیند مرور است، لذا طبیعتاً دیدی سطح بالا از مسئله را شامل خواهد شد. پس جزئیات زیادی بدست نخواهد آمد.

۸-۳- مزایا و معایب روش موارد آشنا

- مزایا: مهمترین نقطه قوت این روش آن است که بینش فرد خبره در حل مسئله متعارف، بدست می‌آید.

- معایب: از آنجا که موارد نوعی بررسی می‌شود، فرد خبره از خبرگی سطح بالا برای حل مسئله بهره می‌گیرد، حتی ممکن است از قدمهایی مبتنی بر احساس خود نیز کمک بگیرد. با ترکیب این روش با روش مرور خاطرات شاید به شفاف‌تر شدن موضوع کمک کند.

۸-۴- مزایا و معایب روش مورد غیر متعارف

- مزایا: دانش پایه فرد خبره به کار گرفته می‌شود. لذا مشکل فشرده سازی دانش حل خواهد شد و نکات پنهان و دانش نهفته که در مراحل قبل بدست نیامده بود، در این مرحله آشکار خواهد شد.
- معایب: یکی از بزرگترین مشکلات این روش، انتخاب مورد غیر متعارف است. باید موردی انتخاب شود که نه تنها قابل حل باشد بلکه امکان پذیر نیز باشد. لذا نیاز به نوعی بصیرت و چیرگی در دامنه مسئله احساس می‌شود، پس توصیه می‌شود از این روش در اوایل پروژه استفاده نشود.

۹- تحلیل دانش

با جمع‌آوری دانش، باید آن دانش تحلیل و تفسیر شود. اولین قدم آن است که رونوشت جلسات تهیه شود. باید این رونوشت‌ها بررسی شده و بخش‌هایی کلیدی دانش بدست آمده مشخص شود که چگونه بهم مرتبط هستند.

برای تفسیر متن جلسات، مفاهیم کلیدی مشخص می‌شود. بطور کلی بدست آوردن دانش توصیفی ساده‌تر از بدست آوردن دانش رویه‌ای است. ممکن است استراتژی حل در یک جلسه بطور کامل مشخص نشود. در حین فرآیند تفسیر، برخی مباحث مهم جدید و قابل بررسی نیز مشخص می‌شود. برای تحلیل متن جلسات، بعد از استخراج مفاهیم کلیدی باید آنها را مرتب‌سازی کرد. باید رابطه بین بخش‌های مختلف دانش بدست آید.

- ضبط دانش: قدم اول در تحلیل دانش، مستند کردن تمام تکه‌های اطلاعاتی و بخش‌های مشابه دیگر که در طول فرآیند بدست آمده است، می‌باشد. این فرهنگ لغات دانش، بخشی از مستندسازی پروژه است که بخش‌هایی برای گونه‌های مختلف دانش: مفاهیم، اشیاء، قوانین و ... بطور مجزا دارد. برای هر بخش دانش اضافه شده مرجع بدست آمده آن نیز باید معرفی شود.
- ارتباط دانش: قدم بعدی مرتب کردن و ایجاد ارتباط بین این بخش‌های مختلف دانش است. البته کار بسیار سختی است چون باید همانطور که فرد خبره بین این بخش‌ها ارتباط ایجاد می‌کند، شما نیز ارتباط ایجاد کنید، البته اگر اشتباهی رخ دهد در مرور آنها با فرد خبره متوجه آنها شده و رفع خواهد شد.

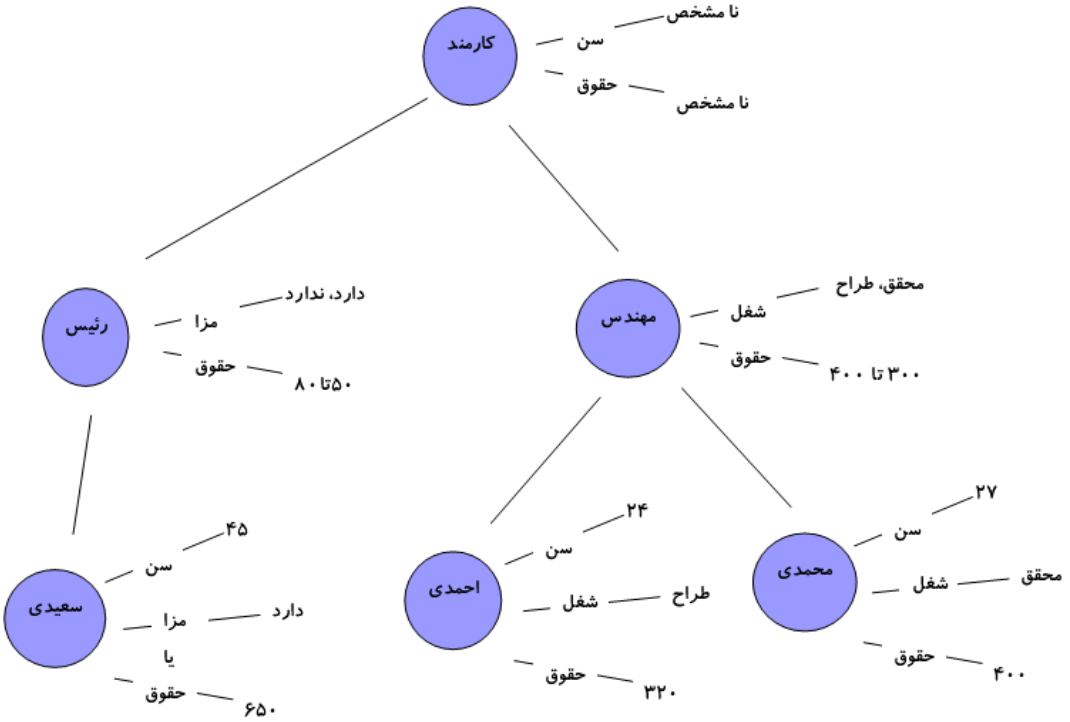
- بررسی دانش: بعد از جمع‌آوری دانش، مرتب کردن و ایجاد ارتباط بین بخش‌های مختلف آن، باید نتایج بررسی شوند. باید آنچه آموخته‌اید و آنچه که باید بیاموزید مشخص شود. ساختار مفهومی که استراتژی حل بر پایه آن شکل می‌گیرد و شبکه قوانین بر اساس آن گسترش می‌یابد در این مرحله کاملاً آشکار می‌شود.

در حقیقت نتیجه این فرآیند طراحی مراحل اکتساب دانش بعدی است. فرد خبره حتماً خروجی این فرآیند را باید تایید کند. برای بیان تکه‌های دانش و مفاهیم استخراجی و روابط بین آنها از ساختارهای دانش بهره می‌گیریم که در ادامه آنها را بررسی می‌کنیم.

۱۰- نقشه شناخت (Cognitive Map)

یکی از روشهای رایج در نمایش گرافیکی روابط طبیعی بین مفاهیم و اشیا، نقشه شناخت است. گراف شامل نودها و لبه‌هایی است که نودها را بهم مرتبط می‌سازد. هر نود می‌تواند شی انتزائی یا حقیقی را بیان کند. اشیا انتزاعی مانند پرنده، کامپیوتر یا ماشین‌ها درک مفهومی ما را از این موارد بیان می‌کنند. اما اشیا حقیقی مانند یک پرنده مشخص، یا هر نمونه مشخصی از موارد ذکر شده فوق می‌باشند. در نودهای پایین این دیاگرام، اجتماع مشخصات نودهای بالاتر قرار دارد.

فرد خبره با بررسی دیاگرام، می‌تواند تغییرات لازم را مشخص کند و حتی شاید مواردی جدید نیز مشخص کند که قبلاً مشخص نبوده، مثلاً تعداد سنوات شغلی هر کارمند.



شکل (۳) نمونه‌ای از یک نقشه شناخت

۱۱- شبکه‌های استنتاجی (Inference Networks)

شبکه‌های استنتاجی ارائه‌ای گرافیکی از قوانین سیستم را در بردارند. با داشتن مقدم و تالی قوانین که در قالب گره‌ها ظاهر می‌شوند و رابطه بین آنها با اتصال بین گره‌ها نشان داده می‌شود.
تالی برخی قوانین پشتیبان مقدم قانون دیگر هستند، لذا بین آنها می‌توان ارتباط ایجاد کرد. که در شکل (۴) به وضوح مشخص است.
با داشتن این دیاگرام، مدیریت و اعمال تغییرات روی قوانین راحت‌تر خواهد بود. تاثیرات بین قانون بر یکدیگر به راحتی دیده می‌شود.

- مثال:

Rule ۱
IF فشارسنج بارومتری خراب باشد
AND وضعیت باد نشان‌دهنده باران باشد
AND دما متوسط باشد
THEN پیش‌بینی وضعیت هوا بارانی است

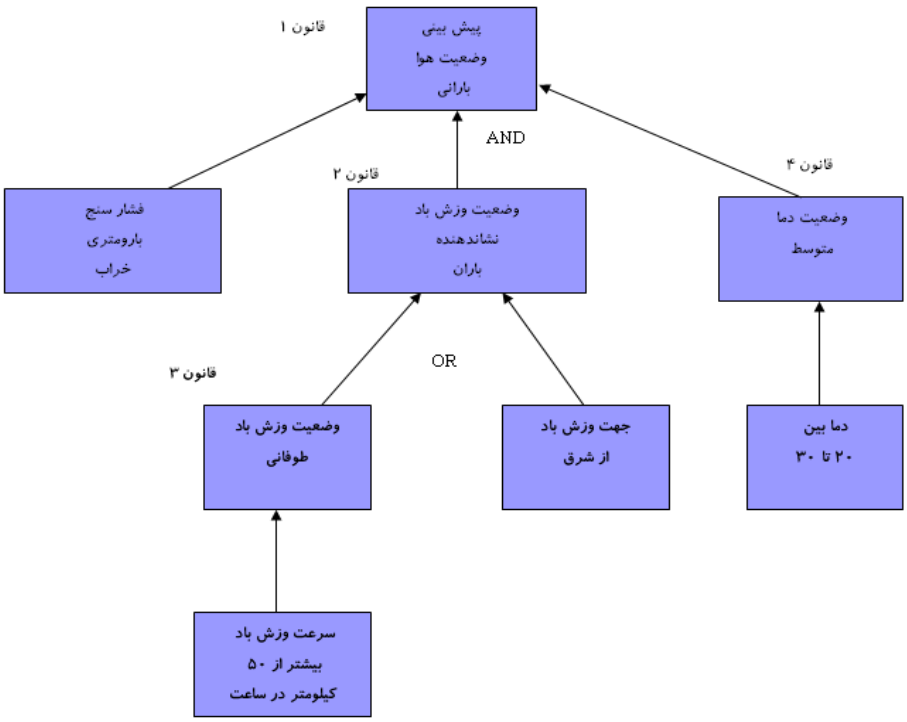
Rule ۲
IF وضعیت باد طوفانی باشد
OR باد از سمت شرق بوزد
THEN وضعیت باد نشان‌دهنده باران است

Rule ۳
IF سرعت باد > ۰۵ کیلومتر در ساعت باشد
THEN وضعیت باد طوفانی است

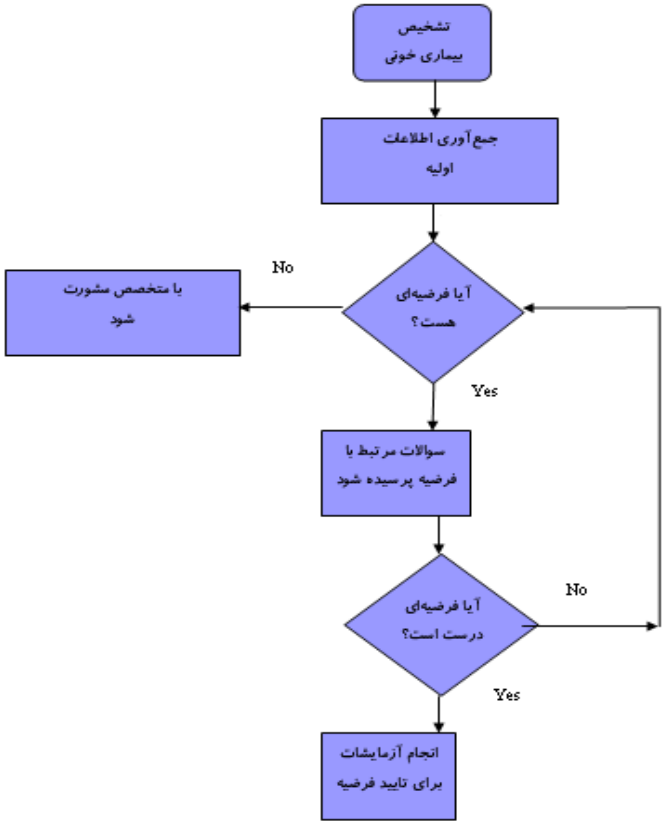
Rule ۴
IF دما بین ۲۰ تا ۳۰ درجه باشد
THEN وضعیت دما متوسط است

۱۲- فلوچارت (Flowchart)

یک فلوچارت، ترتیب قدمهای انجام یک عمل خاص را نشان می‌دهد. یک تکنیک استاندارد در برنامه‌نویسی متعارف است که از نمایش گرافیکی جهت نشان دادن توالی عملیات بهره می‌برد. بلوک‌ها بیانگر عملیات خاص، تصمیم‌گیری‌هاست. لینک بین بلاکها ترتیب اجرایی آنها را نشان می‌دهد.



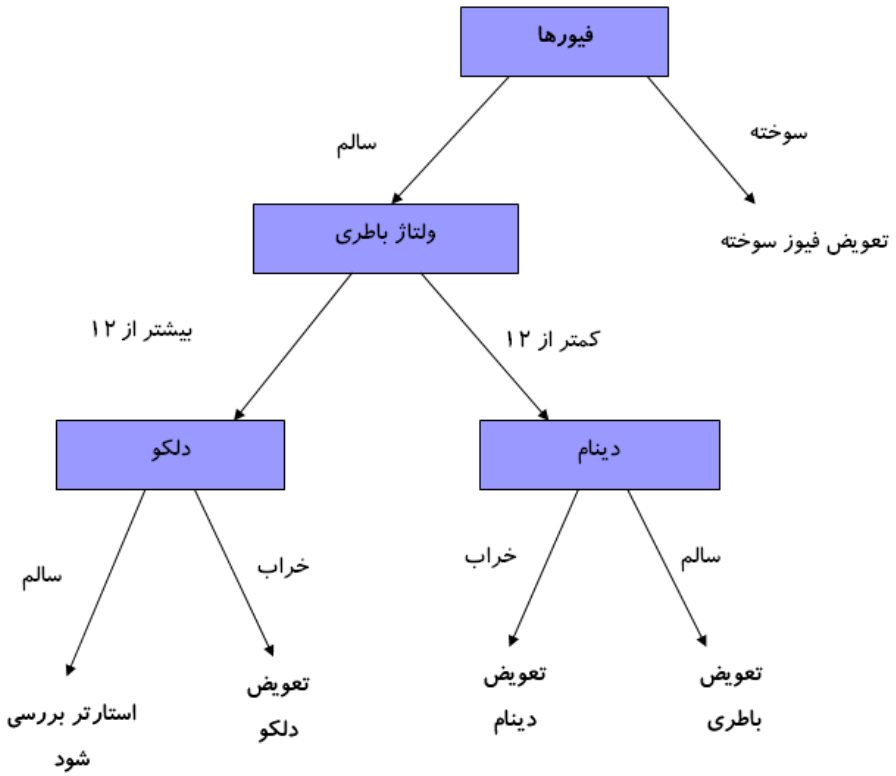
شکل (۴) نمونه‌ای از شبکه استنتاجی



شکل (۵) نمونه‌ای از فلوچارت

۱۳- درخت تصمیم‌گیری (Decision Tree)

یک درخت تصمیم‌گیری نمایش گرافیکی از فضای جستجوی یک مسئله است. درخت ترکیبی از گره و یال است که یال‌ها، گره‌ها را بهم مرتبط می‌سازد. هر گره بیانگر یک موضوع است که در مورد آن تصمیم‌گیری می‌شود و یال‌ها بیانگر مقادیر ممکن می‌باشد. با بکارگیری اطلاعات مرتبط با مسئله، می‌توان در طول درخت حرکت کرده و راه‌حل مسئله را ارائه کرد.



شکل (۶) نمونه‌ای از درخت تصمیم‌گیری

جزوه درس:

سیستم‌های خبره
Expert systems

کتاب مرجع:
Expert Systems Design and Development

نوشته:
John Durkin

مانی عابدینی
تورج بنی‌رستم

فصل ششم- استنتاج فازی
Fuzzy Inference

۱- چرا منطق فازی

- منطق فازی به راحتی قابل فهم است. مفاهیم ریاضی که در استنباط فازی استفاده می‌شود خیلی ساده است. سادگی زیاد و منطبق بودن آن با دنیای حقیقی از مزایای منطق فازی می‌باشد.
- منطق فازی بسیار انعطاف پذیر است. روی هر سیستمی می‌توان منطق فازی را اضافه کرد بدون آنکه مجبور باشیم آن سیستم را از پایه بسازیم.
- منطق فازی توانایی مقابله با داده‌های نادقیق را دارد.
- یک سیستم فازی را به راحتی روی تجربیات افراد خبره می‌توان بنا کرد.
- منطق فازی را با سیستم‌های کنترلی رایج می‌توان تلفیق نمود.
- پایه منطق فازی بر اساس پایه زبان طبیعی انسان است.
- منطق فازی ابزاری مناسب برای کار کردن تحت شرایط عدم قعطیت و شرایط غیر خطی است.

۲- مواردی که از منطق فازی استفاده نمی‌شود

منطق فازی دوای هر دردی نیست.

منطق فازی روشی مناسب برای نگاشتی از یک فضای ورودی به فضای خروجی است. اگر در صورت مسئله‌ای خاص که مد نظر است این نگاشت مناسب برقرار نبود، باید روش دیگری جستجو گردد، منطق فازی مناسب آن مساله نمی‌باشد.

اگر راه‌حل ساده‌تری قبلاً وجود دارد از آن راه حل استفاده می‌شود.

منطق فازی، کد کردن تجربیات و علم ناخودآگاه است. هنگام پیاده‌سازی از دانش ناخوداگاه خود بهره استفاده می‌شود، در نهایت سیستم چگونه به خوبی جواب می‌دهد.

۳- بررسی یک مثال

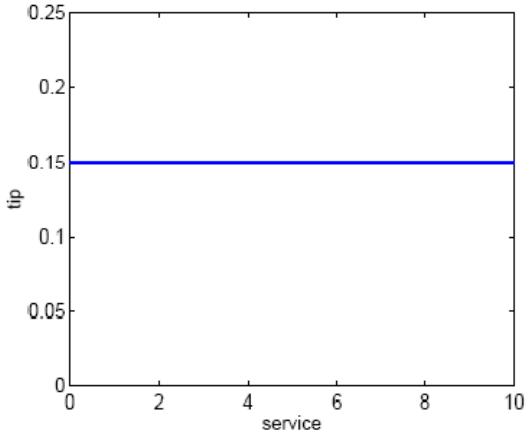
یک مثال برای فهم بهتر منطق فازی با منطق خطی ارایه می‌گردد.

صورت مساله شامل روشی برای محاسبه میزان انعامی است که به پیشخدمت یک رستوران داده می‌شود. عدد ورودی ۰ تا ۱۰ نشان دهنده میزان کیفیت سرویس ارایه شده در رستوران است. ۱۰ یعنی کیفیت عالی.

۱

روش غیر فازی (خطی): فرض می‌شود همیشه انعام ۱۵٪ میزان هزینه غذا در نظر گرفته می‌شود.

Tip = ۰.۱۵

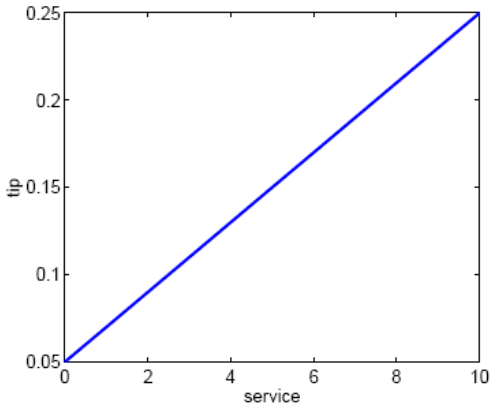


شکل (۱) میزان ثابت برای محاسبه انعام

در این روش میزان کیفیت سرویس ارایه شده، در نظر گرفته نمی‌شود.
روش بهتر شاید آن باشد که از کیفیت بد (۰) تا عالی (۱۰با مقدار ۵٪ تا ۲۵٪ بصورت خطی محاسبه شود.

tip=(۰/۲۰÷۱۰)*service+۰/۰۵

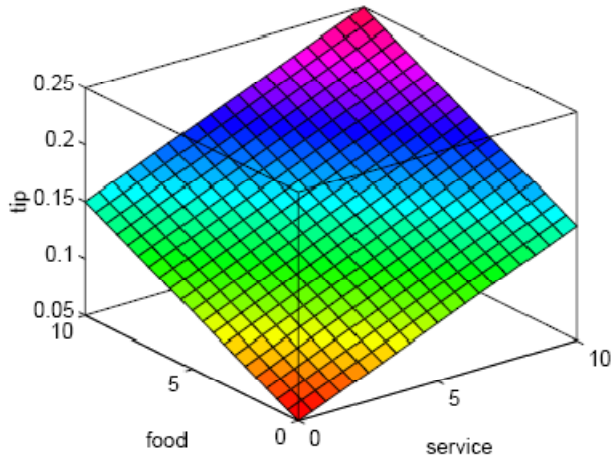
این روش تا حد قابل قبولی جواب گو است.



شکل (۲) روش خطی برای محاسبه انعام

اگر مساله توسعه داده شود یعنی پارامتر دیگری مثل کیفیت غذای رستوران نیز در محاسبه انعام نقش داشته باشد، آنگاه دو ورودی از ۰ تا ۱۰ در نظر گرفته می‌شود که به ترتیب نشان‌دهنده کیفیت سرویس و کیفیت غذای ارایه شده می‌باشند.
می‌توان فرمول قبلی را توسعه داد:

tip = (۰/۲۰÷۲۰)*(service+food)+۰/۰۵

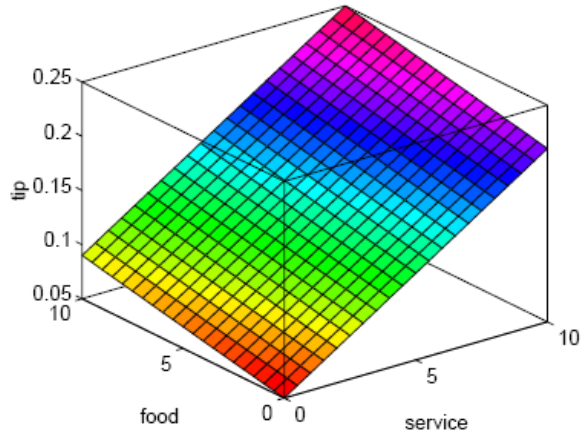


شکل (۳) استفاده از دو پارامتر برای محاسبه انعام

اما واقعیت این است که هر دو فاکتور به یک نسبت تاثیر گذار نیستند. مثلاً سرویس مهتر از کیفیت است (برای انعام به گارسون). پس اگر فرض شود که سرویس ۸۰٪ در میزان انعام موثر است و کیفیت غذا ۲۰٪ آنگاه:

servRatio=۰/۸

tip=servRatio*(۰/۲۰÷۱۰*service+۰.۰۵) + (۱-servRatio)*(۰/۲۰÷۱۰*food+۰/۰۵)

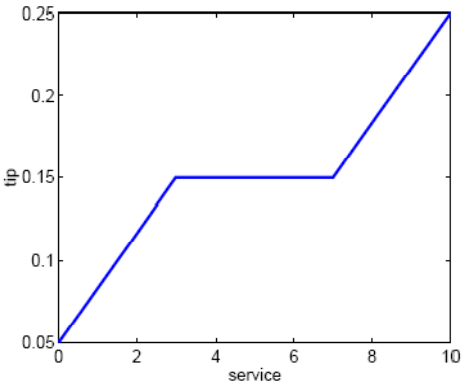


شکل (۴) استفاده از دو فاکتور برای محاسبه انعام

شکل تابع نگاشت کاملاً خطی است، اما در واقع اینگونه محاسبه نمی‌گردد. در اکثر مواقع حدود ۱۵٪ انعام در نظر گرفته می‌شود و تنها زمانیکه سرویس خیلی خوب یا خیلی بد باشد به صورت تصاعدی نرخ انعام تغییر می‌کند. پس روش خطی دیگر جواب‌گوی مساله نیست.

در حالت یک متغییری می‌توان تابع را به صورت زیر تعریف نمود:

```
if service<۳,
    tip=(۰.۱۰/۳)*service+۰.۰۵;
elseif service<۷,
    tip=۰.۱۵;
elseif service<=۱۰,
    tip=(۰.۱۰/۳)*(service ۷)+۰.۱۵;
end
```

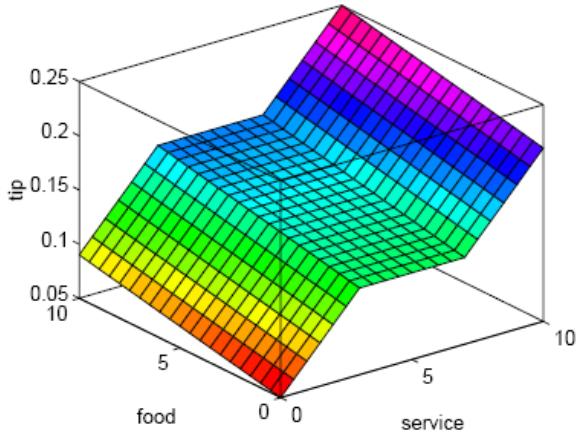


شکل (۵) استفاده از یک متغییر برای محاسبه انعام

اگر متغییر دوم نیز وارد شود تابع به صورت زیر خواهد شد:

```
servRatio=۰.۸;
if service<۳
    tip=((۰.۱۰/۳)*service+۰.۰۵)*servRatio + (۱-servRatio)*(۰.۲۰/۱۰*food+۰.۰۵)
elseif service<۷
    tip=(۰.۱۵)*servRatio + (۱-servRatio)*(۰.۲۰/۱۰*food+۰.۰۵)
else
    tip=((۰.۱۰/۳)*(service ۷)+۰.۱۵)*servRatio + (۱-servRatio)*(۰.۲۰/۱۰*food+۰.۰۵)
end
```

مسئله حل شد، اما برای یک مسئله ساده، یک تابع پیچیده بدست آمد. ریسک وجود خطا در توابع پیچیده بالاتر می‌رود و خطا زدایی اینگونه توابع با مشکل همراه است. از طرفی خود تابع کد شده، به تنهایی نمی‌تواند گویای رفتارش باشد، مستندسازی لازم است تا مفهوم و هدف تابع به نفر بعدی منتقل شود. اعمال تغییرات در تابع ارائه شده به سختی صورت می‌گیرد.



شکل (۶) استفاده از دو متغییر برای محاسبه انعام

۴- مشی فازی

فرض می‌شود فاکتور اصلی (سرویس) به صورت قانونمند معرفی می‌گردد:

۱. If service is poor, then tip is cheap
۲. If service is good, then tip is average
۳. If service is excellent, then tip is generous

ترتیب قوانین اهمیت ندارد.

اگر فاکتور کیفیت غذا نیز تاثیر گذار باشد می‌توان قوانین زیر را اضافه نمود:

۴. If food is rancid, then tip is cheap
۵. If food is delicious, then tip is generous

با ترکیب قوانین فوق مجموعه قوانین زیر به دست می‌آید:

۱. If service is poor or the food is rancid, then tip is cheap
۲. If service is good, then tip is average
۳. If service is excellent or food is delicious, then tip is generous

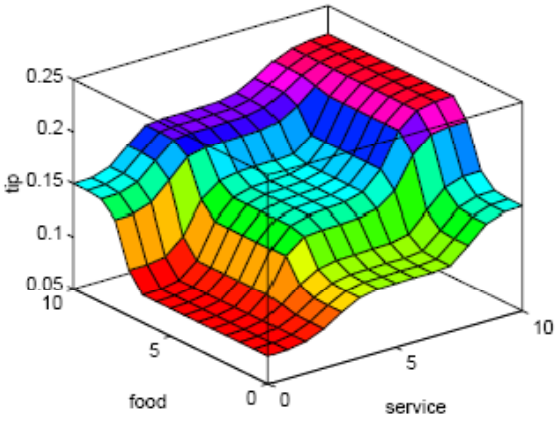
سه قانون ارائه شده، قلب سیستم فازی را تشکیل می‌دهند. منتها باید به متغییرهای زبانی[۱] که استفاده کردیم معنی ریاضی داده شود. یعنی باید معیین گردد منظور از average چیست.

اگر به روش قبلی مساله حل می‌شد:

[۱]. Linguistic variable

```
% Establish constants
    lowTip=۰.۰۵; averTip=۰.۱۵; highTip=۰.۲۵;
    tipRange=highTip-lowTip;
    badService=۰; okayService=۳;
    goodService=۷; greatService=۱۰ ;
    serviceRange=greatService-badService;
    badFood=۰; greatFood=۱۰;
    foodRange=greatFood-badFood;


% If service is poor or food is rancid, tip is cheap
if service<okayService
    tip=(((averTip-lowTip)/(okayService-badService))    *service+lowTip)*servRatio    +(۱-
servRatio)
            *(tipRange/foodRange*food+lowTip)
% If service is good, tip is average
    elseif service<goodService
            tip=averTip*servRatio + (۱ servRatio)* (tipRange/foodRange*food+lowTip)
% If service is excellent or food is delicious, tip is generous
    else
            tip=(((highTip    averTip)/(greatService    goodService))*    (service-
                    goodService)+averTip)    *servRatio    +    (۱-servRatio)    *
            (tipRange/foodRange*food+lowTip)
end
```

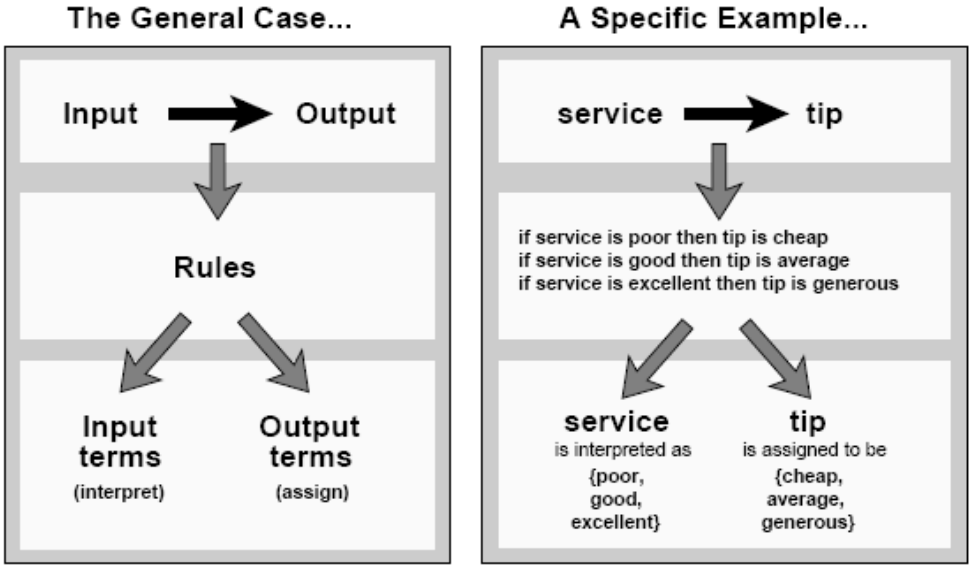


شکل (۷) حل مساله محاسبه انعام با استفاده از فازی

با بررسی کد ارائه شده و حذف کدهای پیچیده، قوانین فازی مجدد خود را نشان می‌دهند.

```
% If service is poor or food is rancid, tip is cheap
% If service is good, tip is average
% If service is excellent or food is delicious, tip is generous
```

در سیستم فازی هیچ گونه پیچیده‌گی دیده نمی‌شود، همه مجموعه دانش در حد دستور یک کد پیچیده (که با زبان سطح بالایی و در عین حال قابل فهم توسط ماشین)، قابل ارایه است.
نکته اصلی در منطق فازی، نگاشت فضای ورودی به فضای خروجی است و ابزار این نگاشت قوانین اگر-آنگاه است که **قانون** نامیده می‌شود.
تمام قوانین به صورت موازی بررسی می‌شوند و ترتیب آنها مهم نیست.
قوانین شامل متغییرها و صفاتی هستند که آن متغییرها را تعریف می‌کنند.



شکل (۸) عملکرد روش فازی

بطور خلاصه: استنتاج فازی روشی برای تفسیر مقادیر بردار ورودی است که بر اساس مجموعه‌ای از قوانین، مقادیری برای بردار خروجی مقداردهی می‌کند.
در ادامه تعریف بردارهای ورودی و خروجی و فرآیند استنتاج ارائه می‌شود.

۵- مجموعه‌های فازی

یک مجموعه فازی مجموعه‌ای است که مرزهای کاملاً تعریف شده و واضحی ندارد. در مجموعه فازی هر عضو با درجه عضویت نسبی به مجموعه تعلق دارد.
در یک مجموعه کلاسیک هر عضو یک به مجموعه تعلق دارد یا خیر. مثلاً مجموعه روزهای هفته شامل: شنبه، یکشنبه، دوشنبه و ... است. مقادیری مانند کتاب، قلم به مجموعه روزهای هفته متعلق نیستند.
پس یک عنصر یا توسط مجموعه‌ای تایید می‌شود، یا تکذیب می‌شود.

حال مجموعه آخر هفته را در نظر گرفته می‌شود. جمعه مسلماً آخر هفته است، اما پنج شنبه، هم جزو روزهای آخر هفته به حساب می‌آید و هم نمی‌آید. یعنی این عضو را می‌توان بطور کامل از مجموعه آخر هفته حذف کرد و نه می‌توان بطور کامل عضو آن مجموعه دانست.

منطق مجموعه کلاسیک، اینگونه برداشت‌ها را نمی‌پذیرد. در حالیکه زندگی انسان پر است از اینگونه برداشتها و تفسیرهای نسبی است.

اما منطق فازی، بر اساس جمله زیر بنا شده:

در منطق فازی، درستی هر جمله و عبارت را با درجه می‌توان نشان داد.

ابزار فازی می‌تواند برای سوال بله/ خیر، جوابی نه کاملاً بله و نه کاملاً خیر ارائه دهد. این همان رفتاری است که انسانها نیز بطور روزمره از خود بروز می‌دهند.

منطق فازی، توسعه‌ای روی منطق بولین است. در منطق بولین اگر مقدار true را با ۱، و false با ۰ نشان داده می‌شود، در منطق فازی مقادیر میانی را می‌توان ارایه کرد.

مثال:

سوال: آیا جمعه یک روز آخر هفته است؟ پاسخ: ۱ (بله، درست).

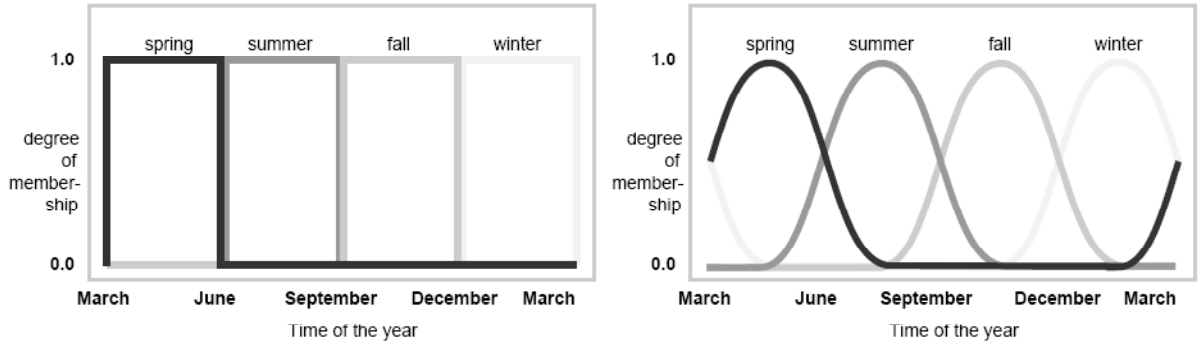
سوال: آیا یکشنبه یک روز آخر هفته است؟ پاسخ: ۰ (خیر، غلط).

سوال: آیا پنج‌شنبه یک روز آخر هفته است؟ پاسخ: ۰/۸

در حقیقت در منطق فازی، یک عضو می‌تواند به مجموعه‌ای هم تعلق داشته باشد، هم نداشته باشد. بطور نسبی (با درجه عضویت مشخص) عضو باشد.

یک مثال دیگر برای مجموعه‌های فازی: فصل‌های یک سال.

در نیمکره شمالی، درست زمانیکه قطب شمال بیشترین تمایلش را به خورشید پیدا می‌کند، بطور رسمی تابستان نامیده می‌شود. تنها در طول سال این واقعه یکبار رخ می‌دهد. بر این اساس در تقویم، روزهای سال به فصل‌های مختلف تقسیم بندی شده و می‌توان مجموعه زیر را در نظر گرفت:



شکل (۹) مجموعه فصل‌های مختلف در قالب مجموعه‌های فازی

۸

۶- تابع عضویت (Membership function)

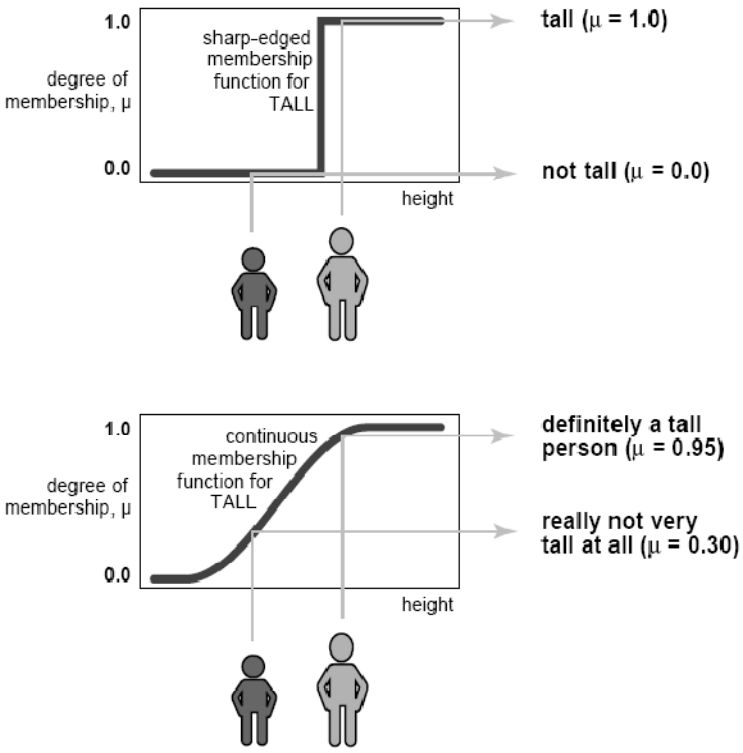
یک تابع عضویت در حقیقت، منحنی است که نشان می‌دهد، یک نقطه در فضای ورودی چگونه به مقدار عضویت در خروجی نگاشت می‌شود. این مقدار عضویت عددی بین ۰ تا ۱ است.

به فضای ورودی *universe of discourse* هم اطلاق می‌شود.

یک مثال متعارف در بحث تابع عضویت، مجموعه افراد بلند قد است. طول افراد بلند قد معمولاً از ۱/۸۰ متر تا ۲/۵۰ متر است. اگر به روش منطق کلاسیک مجموعه بلند قدها تعریف شود، باید عددی مشخص گردد، مثلاً افراد بالای ۱/۹۰ متر را بلند قد می‌باشند.

μ درجه عضویت را نشان می‌دهد. که عددی بین ۰ تا ۱ است.

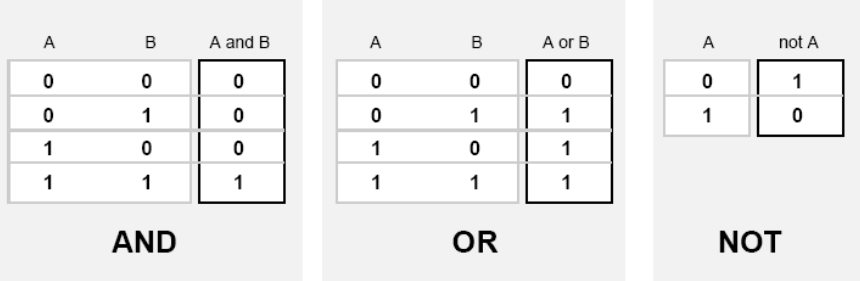
در تابع عضویت فازی، منحنی عضویت وجود دارد که از افرادی که اصلاً بلند قد نیستند تا افراد کاملاً بلند قد را شامل می‌شود.



شکل (۱۰) تابع عضویت فازی برای یک مجموعه

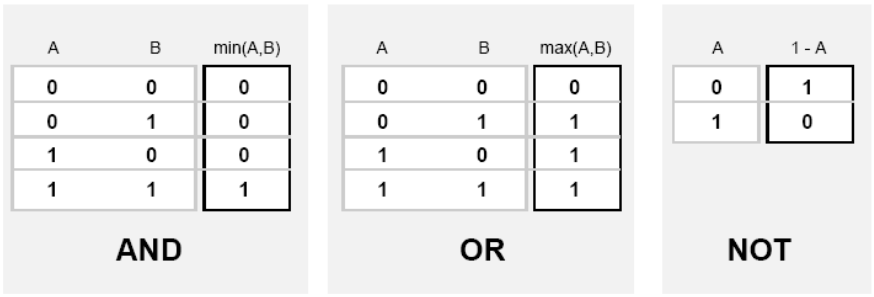
۷- عملیات منطقی

عملیات منطقی در منطق بولین به صورت زیر است:



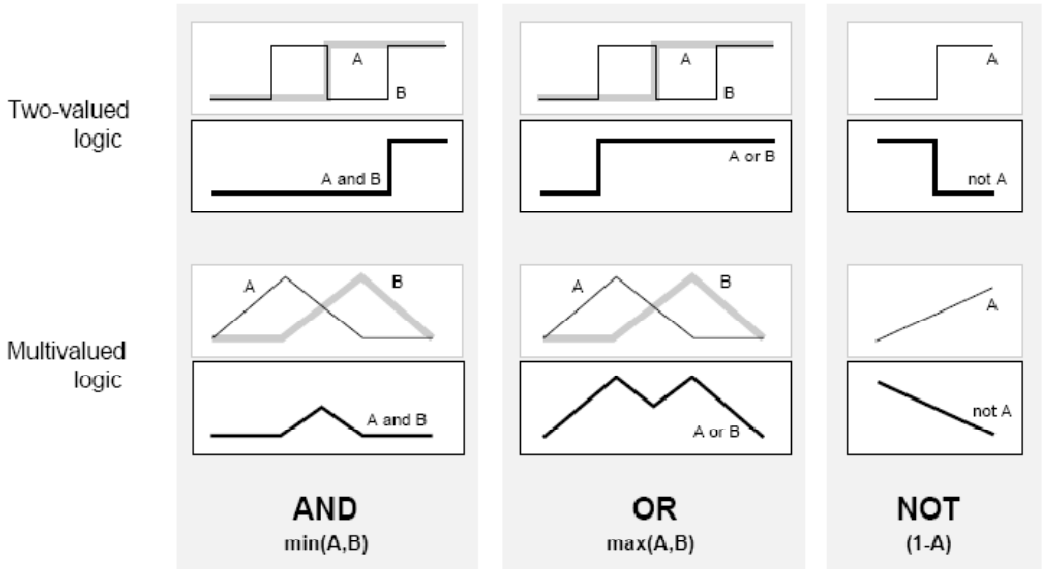
شکل (۱۱) عملیات منطقی در بولین

برای منطق فازی که توسعه‌ای روی منطق بولین است، باید عملیات منطقی طوری تعریف شود که با منطق بولین سازگاری داشته باشد، یعنی And در فازی برای مقدار ۰ و ۰ باید همچنان ۰ باشد. بر این اساس می‌توان بجای And از تابع Min و بجای OR تابع Max استفاده کرد.



شکل (۱۲) عملیات منطقی در فازی

مقایسه بین انجام عملیات در دو منطق فازی و بولین:



شکل (۱۳) مقایسه بین انجام عملیات در دو منطق فازی و بولین

۸- قوانین اگر- آنگاه

عملگرهای فازی و مجموعه‌های فازی لغات زبان فازی است، جملات فازی بر اساس قوانین فازی با الگوی زیر ساخته می‌شود که جملات شرطی هستند که یک نتیجه را بدست می‌دهند.

if x is A then y is B

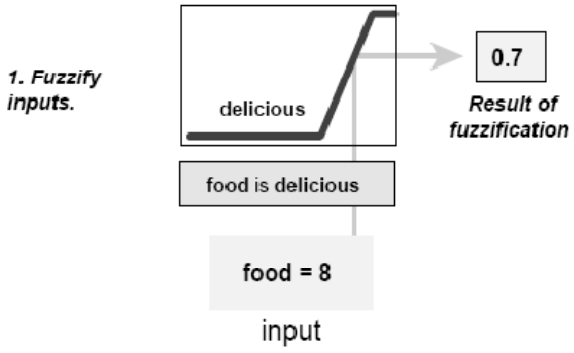
مانند:

If service is good then tip is average

ورودی، مقدار عددی است برای متغییر (مثلاً Service) و خروجی یک مجموعه فازی کامل است. (مثلاً Average) مجموعه فازی خروجی باید بعداً غیرفازی شود و یک مقدار عددی به عنوان خروجی ارائه خواهد شد.

۹- فرآیند استنتاج فازی تا خروجی

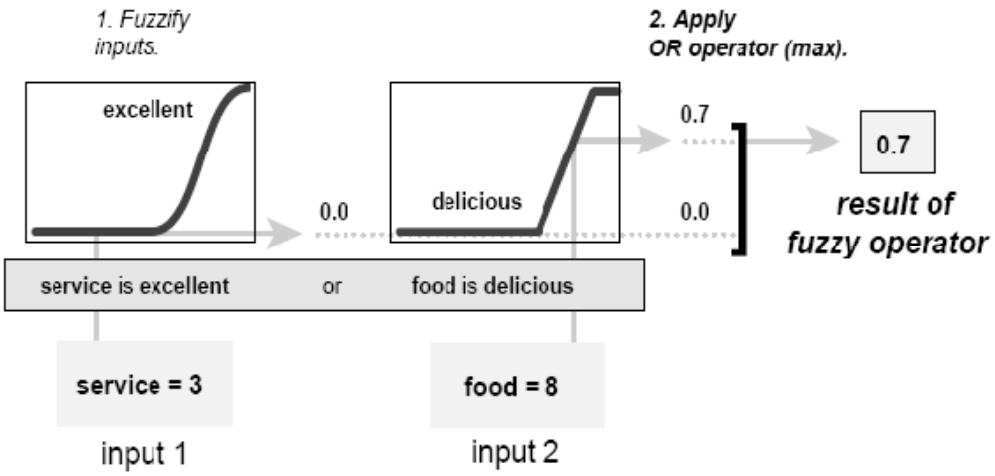
قدم اول، **فازی سازی**: تعیین درجه عضویت ورودی‌ها به مجموعه‌های فازی با کمک توابع عضویت فازی.



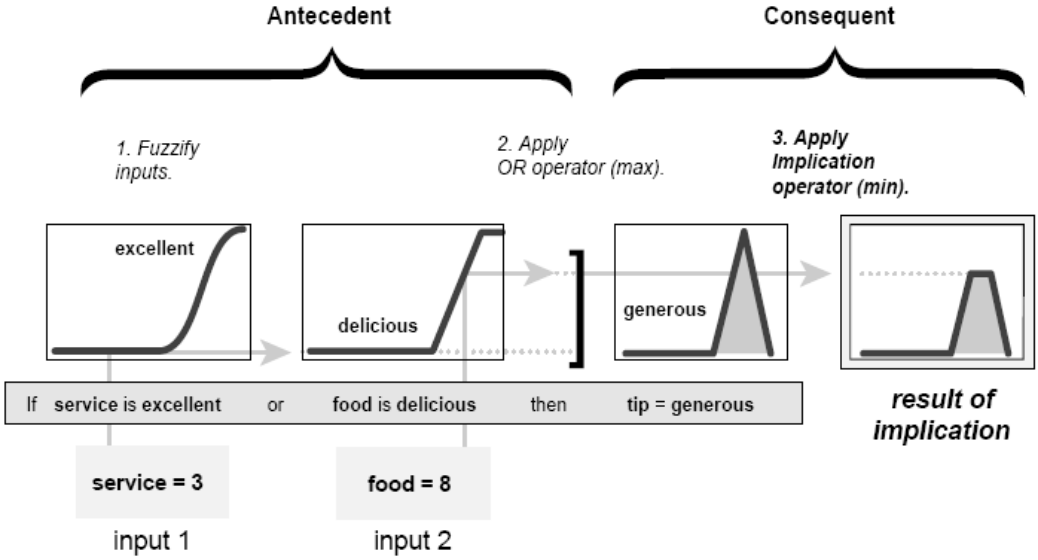
شکل (۱۴) فازی‌سازی

قدم دوم، اعمال عملگرهای فازی: هر قانون فازی از چند مقدم تشکیل شده که هرکدام در مرحله قبل مقدار عضویتشان به مجموعه‌های فازی مشخص می‌شود. اما باید یک عملگر فازی روی تمام آنها اعمال شود تا یک عدد که نمایانگر نتیجه کلی تمام مقدمها (ورودیها) است بدست آید.

قدم سوم، **اعمال استلزام**: خروجی یک مجموعه فازی است که با یک تابع عضویت ارائه می‌شود. با توجه به عدد بدست آمده از مقدمها این مجموعه فازی تغییر پیدا کرده و به عنوان خروجی ارایه می‌شود. از تابع min برای برش مجموعه فازی خروجی (هنگام استلزام) بهره می‌گیریم:



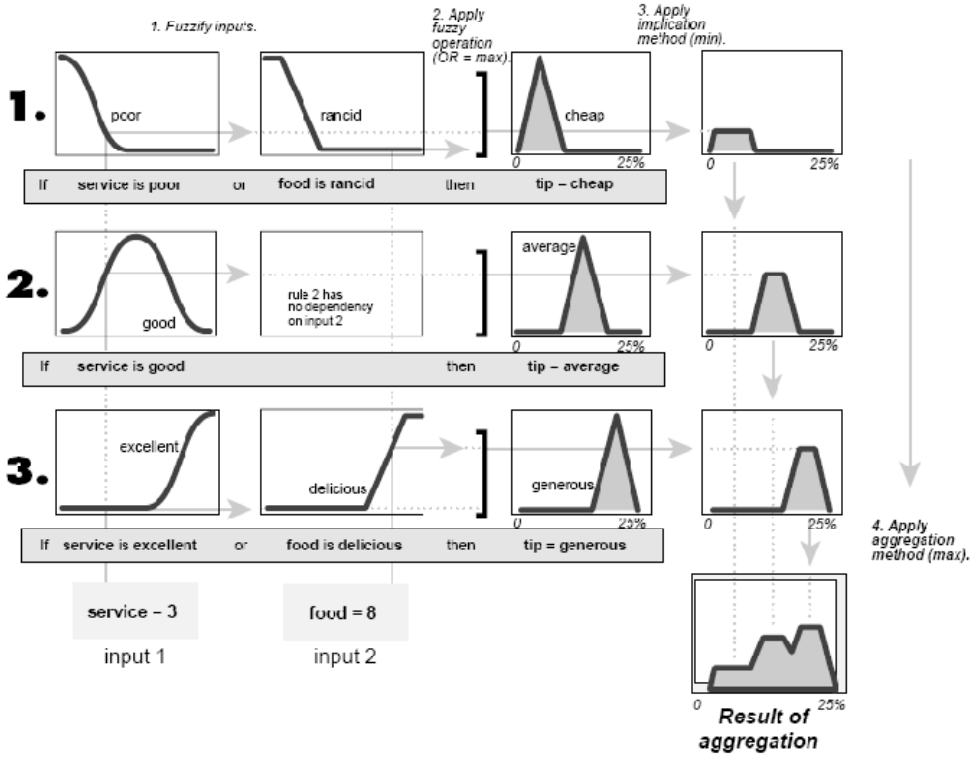
شکل (۱۵) اعمال عملگرهای فازی



شکل (۱۶) اعمال استلزام

قدم چهارم، **جمع کردن تمام خروجی‌ها**: از آنجا که تصمیم‌گیری باید بر اساس تست تمام قوانین FIS بصورت موازی، صورت گیرد، باید نتیجه تمام قوانین که به روش بالا تک تک محاسبه شده‌اند به گونه‌ای باهم جمع شده و مجموعه فازی خروجی واحدی ایجاد شود.

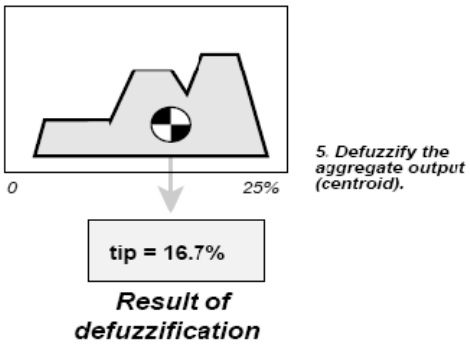
معمولاً برای اجتماع مجموعه‌های فازی خروجی از تابع max بهره می‌گیرند.



شکل (۱۷) جمع کردن تمامی متغییرها

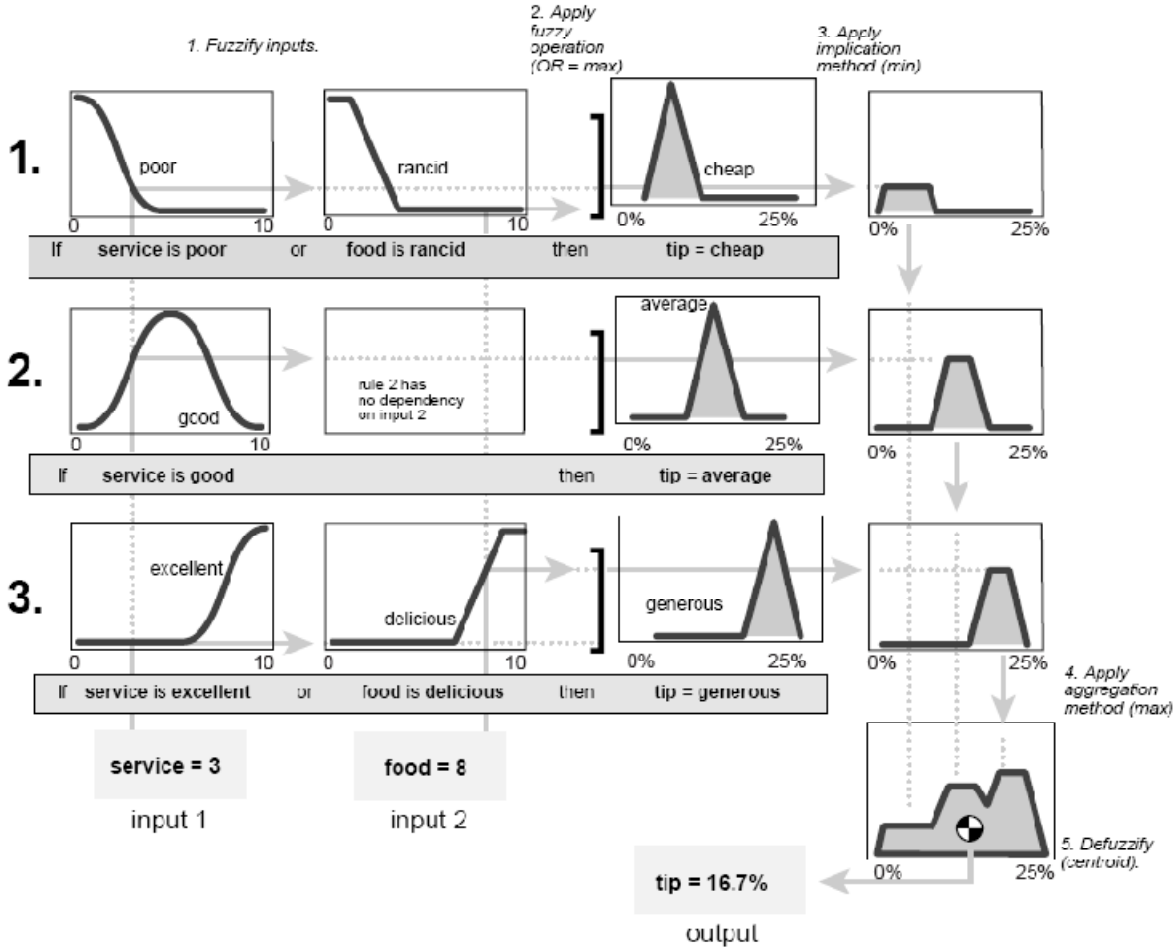
قدم پنجم، **غیر فازی‌سازی**: ورودی فرآیند غیرفازی‌سازی، یک مجموعه فازی است (اجتماع مجموعه‌های فازی خروجی) که باید یک عدد خروجی را نتیجه دهد.

هرچه فازی‌سازی در طی فرآیندهای استنتاج کمک بود، اما برای مرحله اعلام خروجی، یک مجموعه فازی گویا نیست. لذا از روشهای مختلفی برای نگاشت یک مجموعه فازی به یک عدد استفاده می‌شود که مرکز ثقل یکی از روشهای رایج است.



شکل (۱۸) غیر فازی‌ساز

۱۳



شکل (۱۹) عملکرد یک سیستم فازی