Pravidlové znalostní systémy

22. října 2024

Pravidlové (produkční) systémy

Většina znalostních systémů je založena na pravidlech, popř. kombinuje pravidla s jiným způsobem reprezentace.

Název je odvozen od produkčních pravidel, která jsou v těchto systémech využívána.

Pravidlové systémy se od klasických logických systémů odlišují nemonotónním uvažováním a možností zpracování neurčitosti.

Neurčitost se může vyskytnout jednak v předpokladech pravidla, jednak se může týkat pravidla jako celku. Jedním z nejobtížnějších úkolů při studiu neurčitosti je identifikovat jednotlivé typy neurčitosti a najít, jak správně změřit množství neurčitosti každého typu v jednotlivých teoriích neurčitosti (G. J. Klir, 2008).

Nemonotónní usuzování

Oproti klasickému monotónnímu usuzování, které nepředpokládá změny v již nastavených faktech, nemonotónní usuzování tyto změny umožňuje a podporuje.

Nemonotónní usuzování tedy umožňuje úpravy faktů a korekce závěrů během procesu usuzování (viz dále).

Nemonotonní usuzování je takový způsob inference, kdy dříve učiněný závěr může být zpochybněn ve světle nové informace.

Klasickým příkladem je formule „každý pták létá“. Závěr, který můžeme učinit na základě této formule o leteckých schopnostech libovolného ptáka, bude ale zpochybněn, přidáme-li dodatečnou formuli (znalost), že tučnák nelétá.

Zpracování neurčitosti

Neurčitost je ve znalostních systémech a jejich znalostních bázích vyjadřována obvykle numerickými parametry, které se v různých systémech nazývají různě, např. váhy, míry, stupně důvěry, faktory jistoty. Tyto numerické parametry se přiřazují jednotlivým pravidlům, obvykle nabývají hodnot z intervalu $\langle 0,1 \rangle$ nebo $\langle -1,1 \rangle$, což znamená, že se neurčitost vyjadřuje pomocí jediného čísla.

V poslední době se však začínají prosazovat přístupy, v nichž je neurčitost vyjadřována dvojicí čísel, která může být interpretována např. jako interval hodnot).

A konečně existují takové systémy, kde se pracuje s kvalitativně vyjádřenými neurčitostmi, které jsou však obtížně kvantifikovatelné a tím pádem obtížně vyjádřitelné (reprezentovatelné).

Produkční pravidla

- Produkční pravidla se skládají z levé a pravé části
- Levá část se nazývá antecedent, část vzorů, …
  - obsahuje podmínky, které musejí být splněny, aby mohlo být pravidlo aplikováno,
  - může obsahovat logické operátory AND, OR, …,
  - negace (NOT) značí, že danému vzoru nesmí žádný fakt (z báze faktů) odpovídat.
- Pravá strana je tzv. konsekvent, závěr, důsledek, …
  - obvykle provádí změny v bázi faktů,
  - může obsahovat AND, větvení, …

Tvary pravidel

Pravidla (rules) mohou mít například takovéto tvary:

IF předpoklad THEN závěr
IF situace THEN akce
IF podmínka THEN závěr AND akce
IF podmínka THEN důsledek1 ELSE důsledek2

V předpokladové části (antecedentu) se mohou vyskytnout spojky AND a OR, v důsledkové části (konsekventu) se může vyskytnout spojka AND. Součástí pravidla může být také tzv. kontext, ve kterém má být pravidlo uvažováno.

Jiný způsob zápisu pravidla if E then H : $E \rightarrow H$

(E – evidence, H – hypothesis)

Pozn.: Pravidlo $E \rightarrow H$ neznamená totéž, co implikace $E \Rightarrow H$.

Inference v pravidlových systémech

Inference je založena na pravidlu *modus ponens*:

$$\frac{E, E \rightarrow H}{H}$$

To znamená, že jestliže platí předpoklad E a pravidlo $E \rightarrow H$, pak platí závěr H. Modus ponens představuje přímé usuzování.

Nepřímé usuzování pak je dáno pravidlem *modus tollens*:

$$\frac{\neg H, E \rightarrow H}{\neg E}$$

Řešení problému spočívá v nalezení řady inferencí (*inference chain*), které tvoří cestu od definice problému k jeho řešení.

Základní strategie procesu usuzování

Usuzování řízené daty (dopředné řetězení, forward chaining):

Začíná se všemi známými daty a postupuje k závěru. Je vhodné pro problémy zahrnující syntézu (navrhování, konfigurace, plánování, rozvrhování, …).

Usuzování řízené cíli (zpětné řetězení, backward chaining):

Vybírá možný závěr a pokouší se dokázat jeho platnost hledáním dat, která jej podporují. Je vhodné pro diagnostické problémy, které mají malý počet cílových hypotéz.

Základní struktury v pravidlových systémech

Systém porovnávání se vzory (pattern-matching system):

Závěry pravidel jsou obecnější a můžeme je chápat jako kolekce faktů, které mohou nebo nemusí korespondovat se vzory popsanými v předpokladech jiných pravidel.

Inferenční síť (inference network):

Inferenční síť lze charakterizovat jako síťový graf tvořený uzly a hranami. Uzly odpovídají faktům a hrany reprezentují příslušná pravidla. Závěry pravidel jsou fakta, která korespondují s předpoklady jiných pravidel.

Bázi znalostí potom můžeme vizualizovat např. jako síťovou strukturu propojených pravidel a faktů.

Systém porovnávání se vzory

Vztahy mezi fakty a pravidly se ustavují až při běhu na základě úspěšného porovnání faktů se vzory, které se nacházejí v levých částech pravidel. V případě shody všech vzorů v levé části pravidla s fakty v bázi faktů se mohou provést akce v pravé části pravidla (např. to může být zápis faktu do báze faktů nebo zrušení faktu v bázi faktů).

Systémy založené na porovnání se vzorem se vyznačují vysokou flexibilitou a schopností řešit problémy. Jsou spíše aplikovatelné v doménách, kde počet možných řešení je vysoký nebo neomezený, jako je navrhování, plánování a syntéza. V těchto doménách nejsou předdefinovány vztahy mezi fakty a pravidly.

V takových systémech se na druhé straně hůře zajišťuje podpora rozhodování za neurčitosti. V rozsáhlých aplikacích hrozí snížení efektivnosti při vyhledávání aplikovatelných pravidel.

Inferenční síť

Inferenční síť je jeden ze způsobů implementace produkčních systémů. Tato síť je graf, jehož uzly odpovídají jednotlivým faktům (datům) a orientované hrany odpovídají pravidlům.

Inferenční sítě jsou užitečné pro domény, kde počet možných řešení je limitován, jako jsou např. klasifikační nebo diagnostické problémy. Takové systémy jsou však méně flexibilní.

Inferenční sítě se snáze implementují a snáze se v nich zajišťuje vysvětlování.

Prohledávání sítě – dopředné (dopředným řetězením)
– zpětné (zpětným řetězením)

Dopředné (přímé) řetězení

Systém, využívající dopředného řetězení (někdy také přímého řetězení), porovnává aktuální bázi faktů s levými stranami pravidel. Takto vybírá použitelná pravidla, prostřednictvím nichž postupuje k závěrům.

Takový přístup je zejména vhodný pro řešení plánovacích, rozvrhových a konfiguračních problémů.

Základní kroky dopředného řetězení

1. **Porovnání (matching):**

   Pravidla ze znalostní báze jsou porovnávána se známými fakty, aby se zjistilo, u kterých pravidel jsou splněny předpoklady.

2. **Řešení konfliktu (conflict resolution):**

   Z množiny pravidel se splněnými předpoklady se vybírá pravidlo podle priority a v případě více pravidel se stejnou prioritou podle vhodně zvolené strategie.

3. **Provedení (execution):**

   Provede se pravidlo vybrané v předchozím kroku. Důsledkem provedení pravidla může být přidání nového faktu do báze faktů, odstranění faktu z báze faktů, přidání pravidla do báze znalostí ap.

Obvykle je přitom uplatňována podmínka, že pravidlo může být aktivováno pouze jednou se stejnou množinou faktů.

Algoritmus dopředného řetězení

1) Prosté porovnávání levých stran produkčních pravidel s bází faktů a získání množiny odpovídajících pravidel.

2) Při nalezení více než jednoho vyhovujícího pravidla se řeší, které pravidlo má přednost. K tomu se využívá několika postupů (z nichž základní jsou popsány dále).

3) U pravidla, vybraného v předchozím kroku, je zpracována pravá část a pokračuje se výběrem dalšího pravidla.

Možnosti řešení konfliktů

- **Prohledávání do hloubky** – preference pravidel, jejichž vzory odpovídají novějším položkám v bázi faktů
- **Prohledávání do šířky** – preferována jsou pravidla, jejichž vzory odpovídají starším položkám v bázi faktů
- **Preference složitosti** – vybírána jsou pravidla s největším počtem odpovídajících vzorů
- **Preference jednoduchosti** – preferována jsou produkční pravidla s obecnější vzorovou částí
- Preference **podle pořadí uvedení** daného pravidla
- Preference **podle přiřazené priority**…

Možnosti lze formulovat i jinak:

Strategie hledání do hloubky (depth strategy)

– preferována jsou pravidla používající aktuálnější data (data, která se v bázi faktů vyskytují kratší dobu)

Strategie hledání do šířky (breath strategy)

– preferována jsou pravidla používající starší data

Strategie složitosti, resp. specifičnosti (complexity strategy)

– preferována jsou speciálnější pravidla (pravidla mající více podmínek)

Strategie jednoduchosti (simplicity strategy)

– preferována jsou jednodušší pravidla

Vhodné aplikace pro dopředné řetězení

1. Monitorování a diagnostika řídicích systémů pro řízení procesů v reálném čase, kde data jsou kontinuálně získávána a měněna a kde existuje málo předem určených vztahů mezi vstupními daty a závěry. V těchto aplikacích se z důvodu potřeby rychlé odezvy používá inferenční síť.

2. Problémy zahrnující syntézu (navrhování, konfigurace, plánování, rozvrhování, …) – v těchto aplikacích existuje mnoho potenciálních řešení a pravidla proto musejí vyjadřovat znalosti jako obecné vzory. Přesné vztahy (inferenční řetězce) tudíž nemohou být předem určeny a musejí být použity systémy porovnávání se vzorem.

Zpětné řetězení

Znalostní systém postupující metodou zpětného řetězení nejprve vybere možný závěr a poté se na základě získaných faktů (od uživatele, z měřicích přístrojů…) a báze znalostí snaží potvrdit vybranou hypotézu. Z toho vyplývá, že se procházejí pravidla v bázi znalostí od závěrů k předpokladům.

Takové systémy jsou vhodné zejména pro diagnostické a klasifikačních úlohy.

Algoritmus zpětného řetězení

1. Utvoř zásobník a naplň jej všemi koncovými cíli.
2. Shromáždi všechna pravidla schopná splnit cíl na vrcholu zásobníku. Je-li zásobník prázdný, pak konec.
3. Zkoumej postupně všechna pravidla z předchozího kroku.
   a) Jsou-li všechny předpoklady splněny, pak odvoď závěr (proveď pravidlo). Jestliže zkoumaný cíl byl koncový, pak jej odstraň ze zásobníku a vrať se na krok 2. Jestliže to byl podcíl (dílčí cíl), odstraň jej ze zásobníku a vrať se ke zpracování předchozího pravidla, které bylo dočasně odloženo.
   b) Jestliže fakty nalezené v bázi faktů nesplňují předpoklady pravidla, je zkoumání pravidla ukončeno.

c) Jestliže pro některý parametr předpokladu chybí hodnota v bázi faktů, zjišťuje se, zda existuje pravidlo, z něhož by mohla být tato hodnota odvozena. Pokud ano, parametr se vloží do zásobníku jako podcíl, zkoumané pravidlo se dočasně odloží a přejde se na krok 2. V opačném případě se tato hodnota zjistí od uživatele a pokračuje se v kroku 3a) zkoumáním dalšího předpokladu.

4. Jestliže pomocí žádného ze zkoumaných pravidel nebylo možné odvodit hodnotu důsledku, pak daný cíl zůstává neurčen. Odstraní se ze zásobníku a pokračuje se krokem 2.

Vhodné aplikace pro zpětné řetězení

Zpětné řetězení je vhodnější pro aplikace mající mnohem více vstupů než možných závěrů.

Dobrou aplikací pro zpětné řetězení je **diagnostika**, kde člověk komunikuje se znalostním systémem a zadává data pomocí klávesnice. Většina diagnostických systémů byla implementována inferenční sítí, protože vztahy mezi fakty jsou obvykle dobře známy.

Ideální pro zpětné řetězení jsou rovněž **klasifikační úlohy**. Tento typ aplikace může být implementován buď inferenční sítí nebo pomocí vzorů v závislosti na složitosti dat.

Rezoluce

V úlohách, ve kterých se znalostní systém pokouší dokázat určitou hypotézu (obvykle využívající zpětné řetězení) je vhodné používat rezoluční metodu místo pravidla odloučení (modus ponens).

$$(A \vee B) \wedge (\neg B \vee C) \Rightarrow A \vee C$$

Algoritmus Rete (čti ri:ti:)

Dopředný systém porovnávání se vzorem je velmi neefektivní. V každém cyklu se musejí opakovaně porovnávat všechna pravidla se všemi fakty v bázi faktů. Podle odhadu až 90% času práce produkčního systému je věnováno opakovanému porovnávání se vzorem. Přitom po provedení pravidla většina ostatních pravidel v bázi faktů a tudíž také jejich účinků na další pravidla zůstává nezměněna.

Rete je účinný porovnávací algoritmus redukující dobu porovnávání na základě síťové struktury, ve které jsou uloženy informace o ztotožnění podmínek s fakty v bázi faktů.

Síťová struktura pro Rete algoritmus

Uzly sítě: Startovací uzel a dále jeden uzel pro každou podmínku a konjunkci podmínek. Konjunktivní uzly s výstupním stupněm 0 korespondují s pravidly. S každým uzlem je spojena množina faktů, se kterými je podmínka ztotožněna.

Hrany sítě: Jsou označeny vazbami nebo vztahy proměnných vyskytujících se v počátečním uzlu hrany.

Fakt, který je přidáván do báze faktů, je reprezentován příznakem. Tento příznak je zpočátku umístěn ve startovacím uzlu a odtud se pak šíří po síti. Příznak může „projít“ hranou, jestliže jeho argumenty splňují vztah spojený s hranou.
Jestliže v uzlu, odpovídajícím pravidlu, všechny příznaky splňují podmínky pravidla, pak se pravidlo přidá do konfliktní množiny.
Pokud je nějaký fakt z báze faktů odstraněn, pak se jeho příznak podobně „šíří po síti“, přičemž se mažou všechny jeho kopie vyskytující se v uzlech a z konfliktní množiny jsou pak také odstraněna odpovídající pravidla.

Rete algoritmus – postup (provedení):

Rete algoritmus slouží k rychlému výběru pravidla v produkčním systému. Jedná se o propracovanější postup, než je výběr pravidla přístupem dopředného řetězení. V případě dopředného řetězení se po každém provedení pravidla porovnává báze faktů se všemi vzory (vzor je podmínka umístěná v levé části pravidla) každého produkčního pravidla (daného systému). Ovšem vykonáním již zmíněného pravidla se obvykle báze faktů upraví jen minimálně, tudíž dochází k mnoha redundantním operacím.

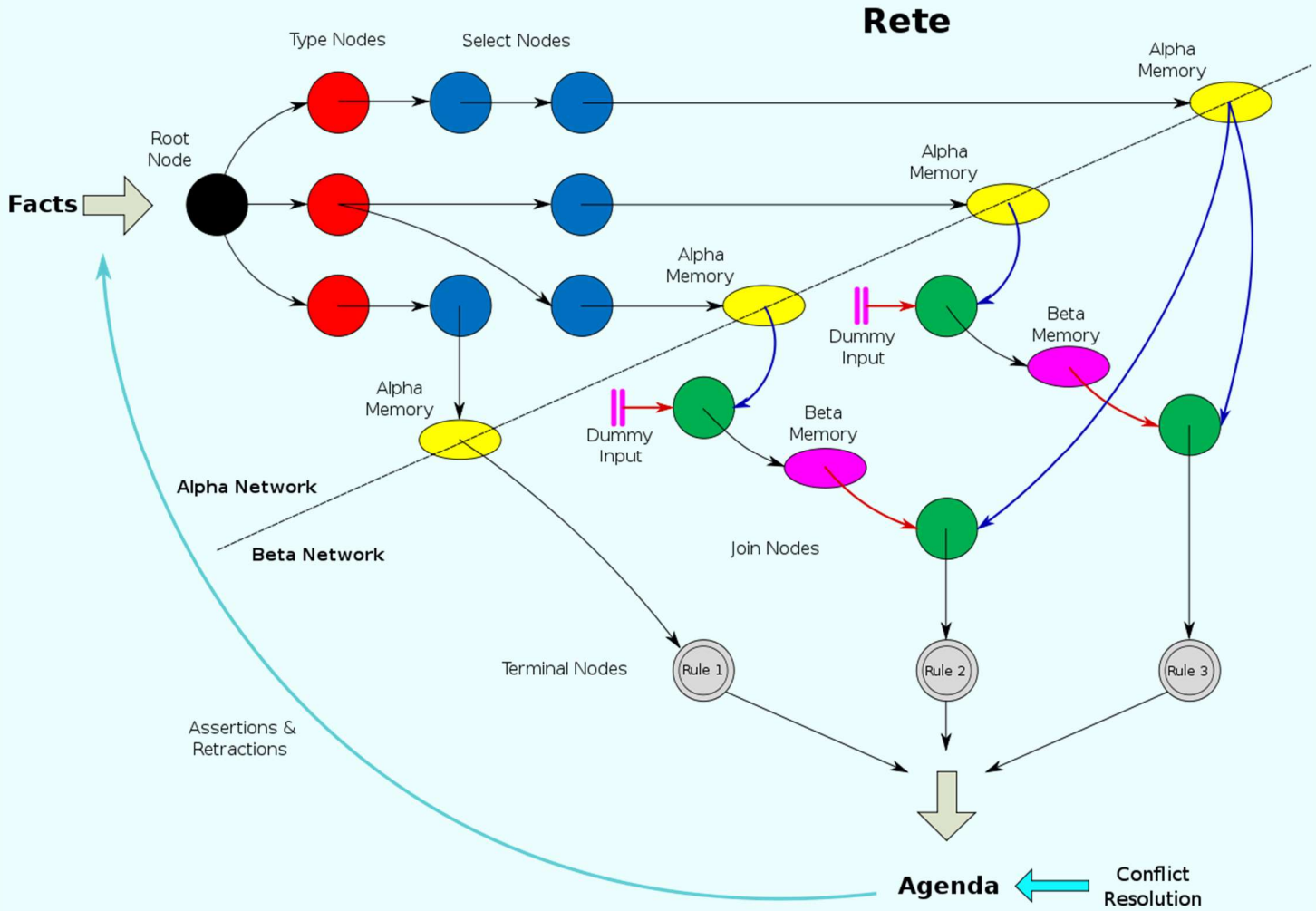
Pro zamezení právě zmíněného jevu Rete algoritmus využívá síťovou architekturu (rete je latinský výraz pro síť) skládající se z hran a uzlů. Každá hrana spojuje dva uzly (počáteční a koncový uzel) a reprezentuje vztahy proměnných počátečního uzlu. Uzly jsou tří typů: startovní uzel, podmínkový uzel (existuje vždy jeden uzel pro jednu podmínku) a uzel pro konjunkci podmínek (pro každou konjunkci se použije jeden uzel). Každý uzel má určitý počet výstupních hran vedoucích do dalších uzlů. Pokud však daný uzel výstupní hrany nemá, pak odpovídá pravidlu, které se (za předpokladu splnění všech podmínek) vykoná.

Nově přidaný fakt do báze faktů je opatřen příznakem (tzv. token). Tento příznak se vkládá do startovního uzlu, odkud se šíří dále do sítě, ale jen těmi hranami, ve kterých jeho argumenty splňují vztah. Pokud je některý z faktů odstraněn, pak se tento jev šíří po síti stejným způsobem.

Jednotlivé uzly Rete sítě se ukládají do alfa a beta pamětí. Do alfa paměti jsou uloženy uzly (označované $A_{i,1}$, $A_{i,2}$, …, $A_{i,n}$) vytvářené pro každý vzor každého pravidla. V beta paměti jsou pak uloženy propojovací uzly (značené $B_{i,1}$, $B_{i,2}$, …, $B_{i,n}$) vytvořené pro alfa uzly každého pravidla. Indexy (i) představují dané pravidlo. Jednotlivé uzly v beta paměti jsou pak vytvářeny následujícím způsobem:

- pro $B_{i,1}$ – levý vstup je z $A_{i,1}$ a pravý vstup je z $A_{i,2}$
- pro $B_{i,j}$, kde $j>1$ – levý vstup je z $B_{i,j-1}$ a pravý vstup je z $A_{i,j+1}$

Příklad Rete algoritmu:



Příklady pravidlových systémů

ART — prázdný ES založený na Lispu, dopředné řetězení, algoritmus Rete

CLIPS — programové prostředí, dopředné řetězení, algoritmus Rete

EXSYS — prázdný expertní systém, dopředné a zpětné řetězení

M.4 — programové prostředí, dopředné a zpětné řetězení, porovnávání se vzorem

ILOG-RULES — programové prostředí, dopředné řetězení, algoritmus Xrete

OPS5 — programové prostředí, dopředné řetězení, algoritmus Rete

Výhody a nevýhody pravidlových systémů

Výhody:

- modularita
- uniformita
- přirozenost

Možné nevýhody a problémy:

- nebezpečí nekonečného řetězení
- přidání nové rozporné znalosti
- modifikace existujících pravidel
- neefektivnost
- neprůhlednost
- pokrytí domény (existují domény vyžadující příliš mnoho pravidel)