Metody odvozování znalostí

19. prosince 2023

Metody odvozování

- „klasické“ logické
- nemonotónní
- generování a testování
- analogie
- defaultní

Východiska a aparáty metod odvozování

- matematická východiska: logika, Prolog
- psychologická východiska: rámce
- biologická východiska: konekcionismus, neuronové sítě
- statistická východiska: kauzální (bayesovské) sítě
- ekonomická východiska: racionální agenti (teorie užitku)

Hlavní komponenty systémů typu tabule

Základní logické metody inference

A	B	$A \Rightarrow B$
0	0	1
0	1	1
1	0	0
1	1	1

- **Dedukce**: logické usuzování, při kterém **musí** závěr plynout z pravidla a předpokladů

$$\frac{A \qquad A \Rightarrow B}{B}$$

(modus ponens)

- **Abdukce**: usuzování z pravidla a pravdivého závěru na předpoklady, které **mohly** tento závěr způsobit (lze použít pro vysvětlení)
- **Indukce**: zobecnění speciálních případů (strojové učení)

13. Metody odvozování

Dedukce je proces usuzování, v němž se od předpokladů (premis) dochází k závěru z těchto premis vyplývajícího. Usuzování je vždy jisté (= prokázaná pravdivost), jde o základní základní postup při dokazování.

Důkazem je konečná posloupnost formulí vedoucí k závěru.

Způsoby dokazování:

1. Formule tvořící výchozí soubor jako pravdivé definujeme => *axiomy*
2. Pravdivost formulí dokážeme pomocí pravidel správného usuzování z tzv. *premis* (předpokladů)

Při použití souboru axiomů musí tento soubor být úplný a bezesporný. Formule, které takto dokazujeme, nazýváme *teorémy* (*věty*).

Indukce spočívá v odvozování obecného poznatku z řady poznatků speciálních. Induktivní metody jsou spojovány s empirickým poznáváním.

Úplná indukce × neúplná indukce

Př.: Mějme tvrzení: Všichni sourozenci pana XY mají krevní skupinu A_2.

Důkaz: Odebereme panu XY i (všem) jeho sourozencům krevní vzorek, provedeme rozbor a jsou-li všechny vzorky skupiny A_2, pak je tvrzení dokázáno.

Nemonotónní usuzování

– předcházející znalosti se mohou revidovat na základě nových poznatků

Klasická logická inference – odvozování důsledků plynoucích z formulí v prostředí, které je statické – popisují jej axiomy:

1. $X \subseteq Cn(X)$
2. $X \subseteq Y \Rightarrow Cn(X) \subseteq Cn(Y)$
3. $Cn(Cn(X)) = Cn(X)$
4. $Cn(X) = \bigcup Cn(Y)$,

kde Y jsou konečné podmnožiny množiny X a kde Cn(X) je množina všech důsledků množiny formulí X.

Pro nemonotónní usuzování neplatí axiom 2, tedy přidáme-li k X další formuli, přestanou platit úvodní závěry. Toho lze využít pro vyjádření neurčitosti typu výjimka z obecného pravidla:

všichni ptáci létají, ale tučňák nelétá

Generování a testování

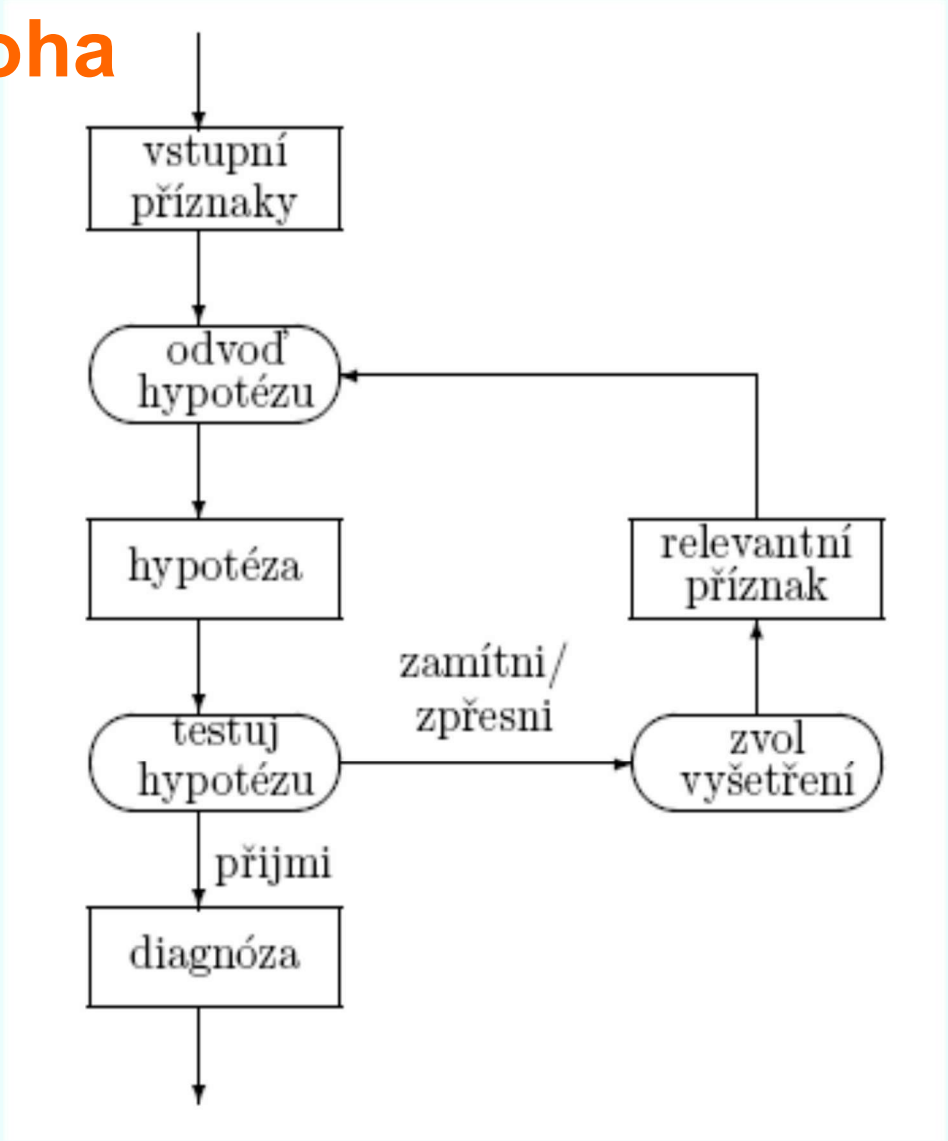
– opakovaně se generuje možné řešení

V daném okamžiku běhu systému mohou být splněny podmínky více pravidel, přičemž levé strany každého z nich mohou být nasyceny (v pracovní paměti existuje objekt, který vyhovuje podmínkám pravidla) více způsoby. Dvojice skládající se z pravidla a jeho nasycení se nazývá instance.

Základní odvozovací cyklus:

- **porovnání** (match) – vytvoření rozhodovací množiny, která obsahuje všechna v dané chvíli aplikovatelná pravidla,
- **rozhodnutí sporu** (conflict resolution) – výběr právě jedné instance (a tedy právě jednoho pravidla) z rozhodovací množiny,
- **úkon** (act) – provedení akcí pravé strany vybrané instance; tyto akce typicky vytvářejí, modifikují a ruší objekty v pracovní paměti.

Diagnostická úloha



Diagnostická úloha řeší:

- kde vzít počáteční hypotézu
- jak zvolit vyšetření, které má prověřit zkoumanou hypotézu
- co způsobí změnu hypotézy
- jak zjistit, že vyšetřování skončilo

Inference v diagnostických systémech

Prohledávání báze znalostí: výběr relevantních znalostí v dané fázi konzultace:

- **exhaustivní** – nalezení všech znalostí umožňujících odvodit daný závěr; typické pro kompozicionální systémy pracující s neurčitostí
- **neexhaustivní** – nalezení prvního pravidla, které umožní odvodit závěr; typické pro systémy bez neurčitosti

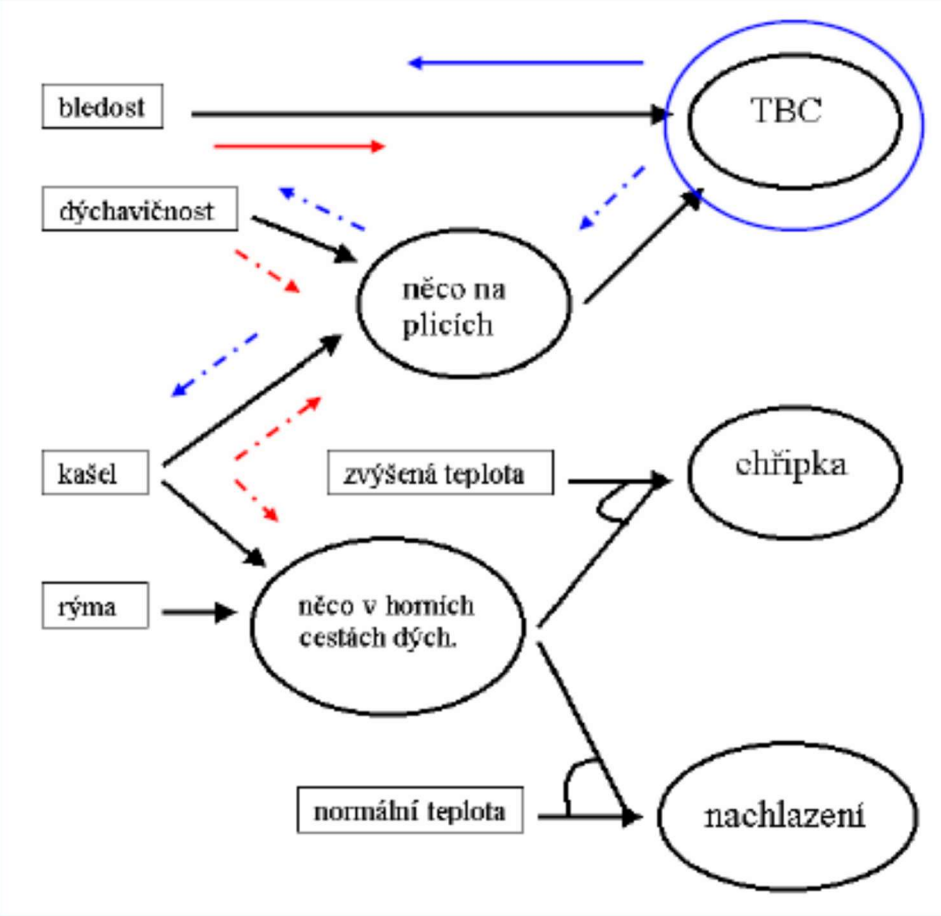
Aplikace pravidel: aplikace jednoho pravidla vychází z dedukce

- **exhaustivní** – skládání dílčích příspěvků všech pravidel vedoucích k témuž závěru; typické pro systémy pracující s neurčitostí
- **neexhaustivní** – aplikace prvního pravidla, které umožní odvodit závěr; typické pro systémy bez neurčitosti – „skládání“ pravidel má podobu disjunkce

Práce s neurčitostí: odvozování závěrů z neurčitých informací a znalostí

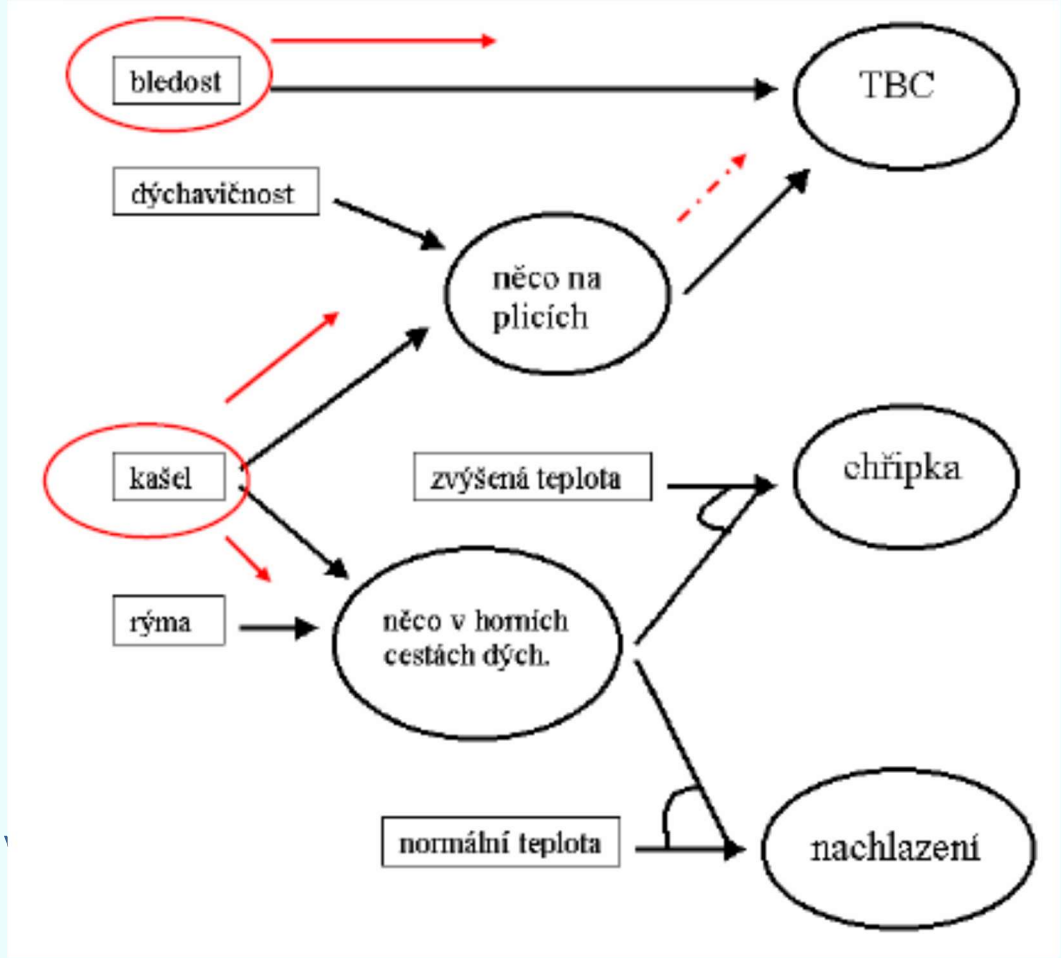
Prohledávání báze znalostí

Zpětné řetězení (backward chaining): vychází z cílů, které chceme odvodit a pokoušíme se nalézt vhodná pravidla:



(lze použít pouze v diagnostických systémech)

Přímé řetězení (forward chaining): vycházíme z faktů, která jsou splněna, a pokoušíme se nalézt aplikovatelná pravidla:



jediný způsob pro plánovací (generativní) systémy; lze ale zčásti využít i v diagnostických systémech

Zpětné nebo přímé řetězení ?

Dle typu úlohy:

- plánovací (generativní) – lze použít jen přímé řetězení pravidel
- diagnostické – lze použít obojí (přímé i/nebo zpětné)

Dle dostupných dat:

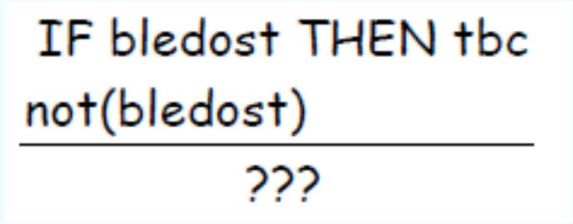
- jestliže zatím nemáme k dispozici žádná fakta a zajímá nás, zda je plněn některý z cílů (vlastně se ptáme, co potřebujeme vědět pro odvození cíle), je vhodnější zpětné řetězení,
- jestliže známe všechna fakta a zajímá nás, co všechno lze z těchto faktů odvodit, je vhodnější přímé řetězení.

Dle struktury báze znalostí:

- pro hlubokou bázi, která má méně cílů, je vhodnější zpětné řetězení,
- pro plochou bázi, která má více cílů, je vhodnější přímé řetězení.

Aplikace pravidel

Aplikace jednoho pravidla vychází z dedukce (na základě platnosti předpokladu lze odvodit platnost závěru). **Pokud není předpoklad pravidla splněn, pravidlo se nebude aplikovat**, protože platí-li pravidlo a neplatí-li předpoklad, nelze nic říci o platnosti závěru.



Aby se „naše“ báze znalostí chovala „rozumněji“, musíme doplnit další pravidla, která pokryjí situace, kdy původní předpoklady platnosti pravidel neplatí, např:

IF not(bledost) THEN not(tbc)

Při úpravách lze postupovat dle de Morganových pravidel.

Příklad (Berka):

Odvozování v systému NEST (1)

Prohledávání báze:

- **režim dialog** - klasické zpětné řetězení
- **režim dotazník** - pracuje se pouze s předem zadanými fakty (simulace přímého řetězení)
- **režim dialog s dotazníkem** - možnost předem zodpovědět některé dotazy, pak následuje zpětné řetězení

Výběr relevantních znalostí v dané fázi konzultace:

- **exhaustivní** v rámci apriorních a kompozicionálních pravidel
- **neexhaustivní** v rámci logických pravidel

Aplikace pravidel (modus ponens):

- **kompozicionální** v rámci apriorních a kompozicionálních pravidel
- **nekompozicionální** v rámci logických pravidel

Odvozování v systému NEST (2)

V síti pravidel se odvozuje všemi dostupnými směry.

Práce s neurčitostí:
(příště)

Vyhodnocení integritních omezení:

Po skončení konzultace se zjišťuje, zda jsou porušeny logické implikace.

$$Ant \Rightarrow Suc \text{ (stupeň)}$$

Míra porušení je definována jako

$IMPL(a,s) = \max(0, \min(1, a-s))$ pro $a > 0$

(Integritní omezení je porušeno pokud Ant je splněn více než Suc)

Další možnosti odvozování znalostí

Agenda je seznam (obvykle zásobník, tedy struktura LIFO) úkolů, které se mají provést. Vytváří se jako vedlejší produkt řešení úlohy. Princip vyřizování agendy umožňuje efektivní zaostřování pozornosti.

Tabule (blackboard) je speciální datová a řídicí struktura, která umožňuje předávat informace mezi jednotlivými částmi systému. Zpočátku tabule představuje pouze sdílenou datovou strukturu. Později je architektura tabule rozšířena o řídicí strukturu, která umožňuje výměnu řídicích informací nebo se podílí na řízení celého systému.
V současné době se tabule používá v systémech distribuované umělé inteligence.

Usuzování založené na analogii

je odvození závěru na základě podobnosti s jinou situací. Používá se např. v rámci tzv. **případového usuzování** (Case-Based Reasoning, CBR).

Pro měření podobnosti (vzdálenosti) se používá metrika:

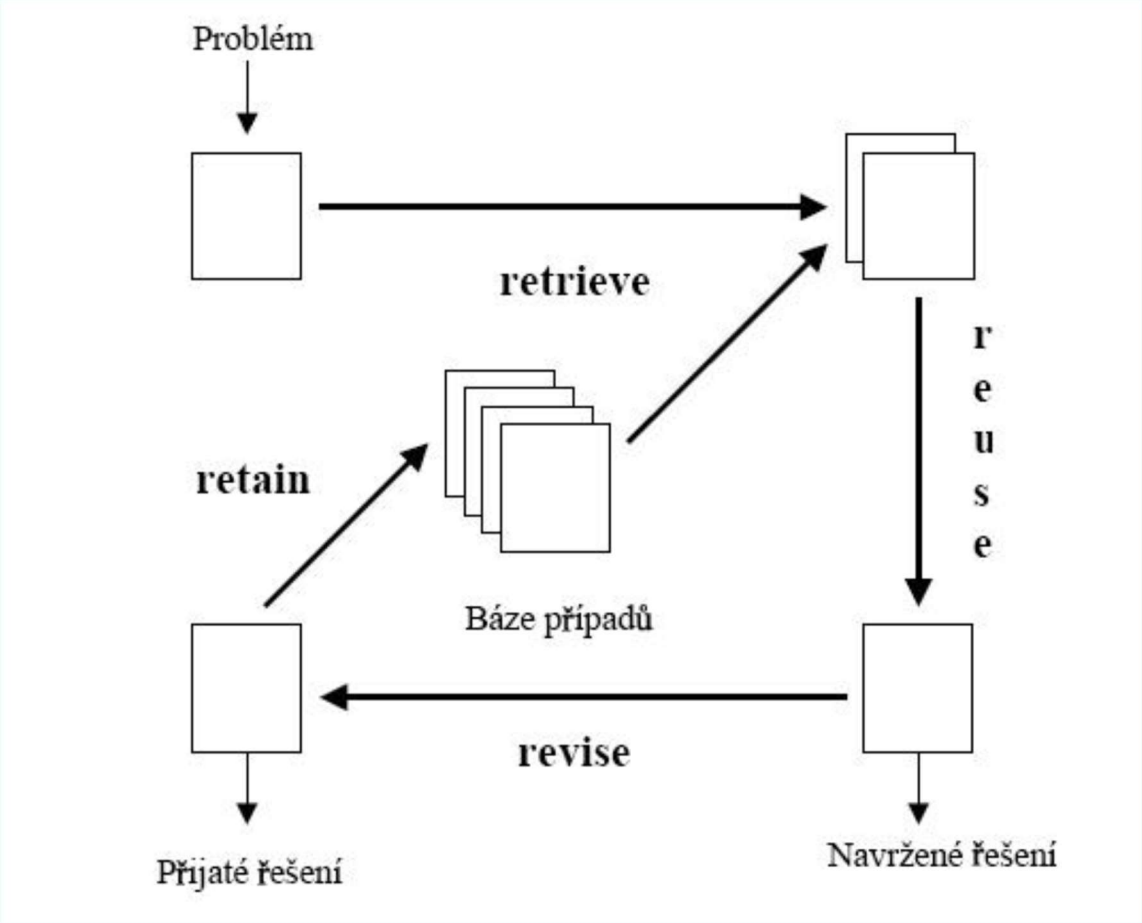
1. $\forall x_1, x_2 \in X;\ d(x_1, x_2) \geq 0$
2. $d(x_1, x_2) = 0 \Leftrightarrow x_1 = x_2$
3. $d(x_1, x_2) = d(x_2, x_1)$
4. $\forall x_1, x_2, x_3 \in X;\ d(x_1, x_2) + d(x_2, x_3) \geq d(x_1, x_3)$

např:

- Eukleidovská metrika – $$d(x_1, x_2) = \sqrt{\sum_{j-1}^{m} (x_{1j} - x_{2j})^2}$$
- „překrytí“ — $$d(x_1, x_2) = \sum_{j=1}^{m} \partial(x_{1j}, x_{2j}), \text{ kde } \partial(x_{1j}, x_{2j}) = \begin{cases} 0 & \text{pro } x_{1j} = x_{2j} \\ 1 & \text{pro } x_{1j} \neq x_{2j} \end{cases}$$

Příklad: Případové usuzování (Case-Based Reasoning – CBR)

– alternativa k usuzování založeném na pravidlech



Inferenční cyklus (Watson, Marir, 1994)

- retrieve (najdi nejpodobnější případy)
- reuse (použij navržené řešení)
- revise (případně reviduj navržené řešení)
- retain (uchovej nové řešení v bázi případů)

Příklad: Případové usuzování v systému NEST

- případy odpovídají konzultacím – obsahují informace o odpovědích na dotazy i o odvozených cílech (vyjádřeno jako váhy příslušných výroků)
- podobnost mezi případem a novou konzultací dána podobností mezi váhami odpovídajících si výroků
- analogicky s pravidlovým usuzováním navrženo kompozicionální a logické (nekompozicionální) odvozování

Kompozicionální případové usuzování

- Podobnost vah výroku

$$pv_k(k_a, k_b) = 1 - |w_a - w_b|$$
$$pv_k \in [-1;1]$$

- Podobnost dvou konzultací

$$pk(k_1, k_2) = \frac{\sum_{a \text{ je dotaz}} \frac{wa}{nva} \sum_{i=1}^{pva} pv(v_{k1i}, v_{k2i})}{\sum_{a \text{ je dotaz}} wa}$$

- Váha cíle

$$w_{cn} = \frac{\sum_i \max(0, pk(k_n, k_i)).w_{ci}}{pkon_+}$$

Logické (nekompozicionální) případové usuzování

- Podobnost vah výroku

$$pv_l(k_1,k_2)=1 \quad \text{jestliže} \quad pv_k(k_1,k_2)>p$$
$$pv_l(k_1,k_2)=-1 \quad \text{jinak}$$

- Podobnost dvou konzultací

$$pk(k_1,k_2)=\frac{\sum_{a\ je\ dotaz}\frac{wa}{nva}\sum_{i=1}^{pva}pv(v_{k1i},v_{k2i})}{\sum_{a\ je\ dotaz}wa}$$

- Váha cíle

$$w_{cn}=\frac{\sum_{i\in M}w_{ci}}{|M|}$$

$$kde \quad M=\begin{cases}\{i\mid i=\arg\max_j[pk(k_n,k_j)]\} & \text{pro } pk(k_n,k_i)>0\\ 0 & \text{jinak}\end{cases}$$

Defaultní metody

Pokud nejsou k dispozici speciální znalosti, uvažuje se na základě obecných znalostí, tedy např.:

Jestliže α(x) platí a β(x) může být konsistentně předpokládáno, pak odvoď γ(x)

Odvozovací pravidlo (Reiter, 1980):

$$\frac{\alpha(x):\beta(x)}{\gamma(x)}$$

Operátor modální logiky (McDermott, Doyle, 1980):

$$(\forall x)\ \alpha(x) \wedge M\beta(x) \Rightarrow \gamma(x)$$

Defaultní logiky definují pojem **rozšíření** (extenze) teorie s defaultními prvky jako analogii k pojmu logického uzávěru (množiny logických důsledků).

Závěr k odvozování (Křemen)

Až dosud předpokládanou oblast aplikace strojového odvozování, v níž se v posledních padesáti letech ve velké míře objevují soubory inherentně vágních znalostí, získaných přirozeným poznáním (v terminologii umělé inteligence se jim říká povrchové či mělké znalosti), je nutno redukovat a znalosti s inherentní vágností zcela oželet, vypustit je. Je přípustné používat pouze znalosti získané metodou exaktních věd (v terminologii umělé inteligence se jim říká hloubkové znalosti), tedy znalosti vyjádřené atributy a matematicky reprezentovanými vztahy mezi nimi.

Entity, které mohou být použity při automatizovaném odvozování, tedy při zachování nulové vnitřní vágnosti interpretace všech jazykových konstrukcí formálního systému, jsou jen a jen tyto:

- matematické entity (např. dokazování teorémů)
- exaktní hry – umělé formální systémy materializované do podoby figur, karet apod. (např. šachy, karetní hry)
- znalosti získané Newtonovou metodou exaktních věd (odvozování ve vědě a technice, hloubkové znalosti v umělé inteligenci, např. automatizovaná optimalizace logických obvodů)

Dalším problémem automatického odvozování je nesamohybnost formální inference. Je nutno vytvořit umělého hybatele. Učebnicové návody na vytvoření hybatele např. metrizací stavového prostoru (tj. prostoru vytyčeného všemi dovolenými pohyby inference), a tak akcentování jisté inference na úkor jiných aplikovatelných v daném bodě stavového prostoru je použitelné jen v ojedinělých případech.
Jen ve spíše vzácných případech lze pro celý stavový prostor vytvořit globálně působícího hybatele. Ve většině reálných případů musí tvůrce systému nalézt vhodný podnět pro pohyb inference pro každý bod stavového prostoru, což je velice obtížná a namáhavá úloha, uvážíme-li, že takových bodů jsou stovky i tisíce.

Poznámka: Při výuce exaktních věd, řekněme fyziky, by již na úvod měl být vysvětlen rozdíl mezi přirozeným a umělým poznáním a tím princip ustavení exaktní vědy – Newtonův zásadní přínos, který je nepostradatelným dílem moderního vzdělání.

Vysvětlování

Důvody pro vysvětlování ve znalostním systému:

- uživatel získá jistotu, že způsob usuzování systému je v zásadě v pořádku a že nabízená řešení jsou akceptovatelná
- tvůrce aplikace se může přesvědčit, že implementované znalosti (a jejich využívání) odpovídají představám experta

U diagnostických expertních systémů typicky nacházíme vysvětlení tří typů:

- **why** – proč systém klade svůj dotaz
- **how** – jak dospěl systém ke svým závěrům
- **what if** – jak by systém reagoval na změnu některé odpovědi

Tři úrovně vysvětlování

- pro řešitele systému či pro specialistu na úrovni experta – jednotlivá vysvětlení budou jednoduchá a budou obsahovat pouze významná fakta, mnohdy v symbolické formě, popisující činnost systému (způsob nalezení vysvětlení, stručně význam vysvětlení)
- pro znalého uživatele plně obeznámeného s funkcí používaného znalostního systému – vysvětlení ve strukturované podobě, obsahující veškerá potřebná fakta a závislosti mezi nimi, avšak bez vysvětlování základních pojmů a vztahů, např. k procesu
- pro náhodného uživatele či naprostého laika – velmi obsáhlá vysvětlení v dobře zpracované podobě (dobře srozumitelné věty apod.) obsahující vyčerpávající vysvětlení všech používaných pojmů, funkcí, výsledků či závěrů odvozování apod.

Funkce vysvětlovacího modulu

Aby se zvýšila důvěra uživatelů v závěry a doporučení znalostního systému, musí systém poskytovat vysvětlení svého uvažování. Obvykle systém vysvětluje (do jisté míry) právě položený dotaz, znalosti relevantní k nějakému tvrzení, právě zkoumanou cílovou hypotézu, právě probíhající odvozování a zdůvodňuje postup systému při odvozování.

Vysvětlení předkládá uživateli ve formě vhodné pro uživatele a v rozsahu, který si uživatel určí.

Vysvětlovací schopnost systému je nezbytná též při ladění a modifikacích báze znalostí.

Způsob komunikace se znalostním systémem pak obvykle vypadá tak, že:

- iniciativa je na straně znalostního systému; ten volí dotazy, které klade uživateli
- uživatel se nespokojí pouze s doporučením systému, nýbrž požaduje ještě jeho vysvětlení či zdůvodnění

Příklad: Vysvětlovací modul aplikace znalostního systému pro výpočet erozní ohroženosti zemědělsky využívaného pozemku poskytuje uživatelům detailní rozbor současného stavu pozemku, potenciálu hospodaření, limitující prvky hospodaření, ale i rozbor navržených variant řešení z hlediska jejich účinnosti, efektivity, dopadů, předpokladů použitelnosti a dalších aspektů pro jejich komplexní porovnání.

13. Metody odvozování