

EXPERTNÉ SYSTÉMY

**KONCEPCIE, PRINCÍPY A ARCHITEKTÚRA,
PROSTRIEDKY REPREZENTÁCIE ZNALOSTÍ
ROZVINUTÉ A NEKLASICKÉ TECHNIKY ODVODZOVANIA,
TVORBA SYSTÉMOV**

Mikuláš Popper
MFF UK

Bratislava 1997-2001

PREDSLOV A ÚVODNÉ POZNÁMKY

Predložený text uvádza čitateľa do jednej z oblastí *umelej inteligencie*¹ - do problematiky **expertných systémov**.² Ponúkaný rozsah výkladu na pozadí vybraných tém tvoriacich konceptuálne pozadie týchto systémov, umožňuje v širších súvislostiach orientáciu v problematike tohto druhu softvéru. Umožňuje vnímať jeho princípy aj praktické uplatňovanie, predovšetkým však umožňuje v širokom rozsahu osvojiť si poznatky nevyhnutné pre tvorbu ES. Hoci texty sú určené poslucháčom univerzitného štúdia informatiky, najmä pre špecializáciu UI, môžu byť nepochybne prínosom aj pre iné štúdijné smery ako aj pre odborníkov z rôznych aplikačných oblastí. Čitateľova orientácia aspoň vo východiskových tāmach informatiky a v uvádzajúcich predmetoch UI zvyšuje čitateľnosť týchto textov.

ES tvoria špecifickú podtriedu *znalostne-intenzívnych* (hovori sa tiež *znalostných*) *systémov*. Tie môžu byť nositeľmi netriviálneho rozsah **explicitne formulovaných (vyjadrených) a symbolovo reprezentovaných znalostí** z aplikačnej oblasti, v ktorých sa uplatňujú. **Expertné systémy** sa od iných *znalostne-intenzívnych systémov* líšia navyiac tým, že sú nositeľmi – stelesnením - **odborných znalostí vyvierajúcich z oblasti, v ktorých majú svoje uplatnenie**.

Hoci predkladaný text v mnohých smeroch rozširuje poňatie ES predkladané vo väčšine porovnateľnej literatúry³, predsa len, okrem iného aj z rozsahových dôvodov, nezahrňuje alebo nerozvádza niektoré témy, ktoré by si pozornosť zaslúžili. Napriek tomu v dostatočnom rozsahu uvádzajú čitateľa do živej a aktuálnej teoretickej aj implementačnej problematiky a zároveň môže motivovať k prípadnému hlbšiemu štúdiu aj vedeckému bádaniu. Pre tých, čo v tej či inej podobe sa budú chcieť, ba dokonca z pracovných dôvodov budú musieť, danou problematikou hlbšie zaoberať, sú tieto texty nepochybne zdrojom poznatkov a východísk ku konkrétnym praktickým aplikáciám.

Je potrebné zdôrazniť, že uvádzaný text nie je príručkou pre čitateľa, ktorý vytvára expertný systém - neposkytuje k tomu konkrétne návody. Pozornosť je v ňom väčšmi zameraná na poznatkové zázemie ES, menovite, popri základných varintov symbolovej reprezentácie znalostí, na rôzne metódy, princípy, prostriedky a techniky tvorby týchto systémov, vrátane prístupov k ich uplatňovaniu. Uvádza, resp. upozorňuje aj novšie a perspektívne vývojové trendy, ktoré majú potenciál praktického uplatnenia pri používaní týchto systémov.

Produkty vznikajúce v oblasti tvorby ES sa dajú členiť na

- ✓ **prázdne expertné systémy** – čo sú programové produkty zodpovedajúce konkrétnemu funkčnému typu ES s vymedzenými vlastnosťami, štruktúrou a súvisiacou formou reprezentácie znalostí, avšak bez štruktúr symbolov reprezentujúcich znalosti – nazýva sa **báza znalostí**⁴ - potrebných pre vymedzené aplikácie,

¹ V skratke UI. Slovenský termín *umelá inteligencia* vznikol prekladom z anglického originálu *artificial intelligence*. Napriek tomu, že sa v našom jazykovom prostredí udomácnil, je vhodné poznamenať, že jestvujú a aj sa uplatňujú synonymické termíny, najmä v podobe *strojová inteligencia*, či *intelektika*. Vzhľadom na špecifický zúženú sémantiku týchto pojmov je ich používanie uprednostňované nemalým počtom odborníkov.

² V skratke ES.

³ Za východisko predkladaného textu sa dá sovojm zameraním považovať učebnica *J. Kelemen a kolektív: Základy umelej inteligencie, Alfa, Bratislava, 1992*, a obsahu knihy *M. Popper, J. Kelemen: Expertné systémy, Alfa, Bratislava, 1988*, ktorého je aktualizáciou a rozvinutím. Druhé z uvedených diel, aj napriek sústavne a rýchlo sa rozvíjajúcim novým poznatkom odboru, dá sa stále považovať za východiskový a orientujúci štúdijný prameň.

⁴ V skratke BZ.

- ✓ **bázy znalostí**, ktoré svojou štruktúrou sú kompatibilné s **prázdny** ES a sú produktom často veľmi náročnej spolupráce dvoch kategórií odborníkov: profesionála z oblasti aplikácie a špecialistu na tvorbu ES a BZ,
- ✓ **problémovo zamerané (dedikované) - ES** – to sú expertné systémy vystrojené BZ, ktoré sú nositeľmi znalostí z danej problémovej oblasti,
- ✓ **vývojové prostredie tvorby ES** – to sú softvérové produkty na tvorbu prázdnych alebo problémovo zameraných systémov.

Finančné náklady na vývoj alebo zakúpenie ES, prípadne aj BZ, obdobne ako na vývoj alebo zakúpenie netriviálneho *vývojového prostredia tvorby ES*, sú značné. V čase spisovania týchto textov sa môžu pohybovať rádovo až v rozmedzí od 5-6 ciferných čísiel v korunovom vyjadrení. Preto poznatky a informácie uvádzané v tomto texte sa dajú považovať za významné aj pre

- formulovanie aplikačných potrieb a požiadaviek na používanie expertného systému a teda pre jeho cenovo a funkčne adekvátnu špecifikáciu,
- tvorbu vlastného prázdneho či aplikačného ES,
- výber a zakúpenie prázdneho alebo aplikačne zameraného (dedikovaného) ES,
- výber komerčne dostupných prázdnych ES alebo ich vývojových prostredí a pod.

Prednášky z predmetov **Expertné systémy I.** a **II.** korešpondujú s týmto textom, je ich východiskom. Avšak pri "živom" výklade sú spravidla viaceré témy podrobnejšie rozvíjané aj rozšírené, prípadne aktualizované. Rozsah, výber a príprava tém prednášok je nezriedka aj výslednicou interakcie s poslucháčmi.

Autor zatiaľ tieto texty nepovažuje za uzatvorené. Vznikajú, doplňujú a modifikujú sa priebežne. Najmä preto, že neustále pribúdajú nové poznatky zverejňované v rozmanitých literárnych prameňoch, alebo sú artikulované na významných vedeckých podujatiach. Takže základné, vlastne už klasické témy, sa priebežne aktualizujú a doplňujú o nové vybrané state. Keďže texty zatiaľ neprechádzajú ani odbornou ani jazykovou recenziou, rozsah potenciálnych nedostatkov, chýb, obsahovo či štylisticky nevydarených formulácií je určite nemalý. Autor s vďakou uvíta všetky upozornenia na akékoľvek nedostatky, žiaduce opravy aj úpravy, ako aj odporúčenia na doplnenie a vylepšenie textov.

1. EXPERTNÉ SYSTÉMY - UVEDENIE DO PROBLEMATIKY

1.1 Konceptuálne hľadiská

Expertné systémy v kontexte umelej inteligencie

Umelá inteligencia (UI) je vedný odbor s vlastným teoretickým zázemím a vlastnou oblasťou praktických aplikácií. **Expertné systémy** tvoria jednu z významných častí UI – majú svoje špecifické teoretické základy a sú významné pre svoju praktickú uplatniteľnosť.

Problematika ES sa zameriava na

- a) tvorbu počítačom realizovaných javov, ktoré dostatočne presvedčivo pripomínajú prirodzený fenomén ľudskej inteligencie pri riešení odborných problémov (a v tesnej súvislosti s tým)
- b) hľadanie hraníc a možností symbolovej reprezentácie poznatkov, vrátane procesov ich nadobúdania, uchovávaní, využívania a rozvíjania.

Témy súvisiace s oboma zložkami tejto problematiky sú nosné. Zaoberáme sa nimi v nerovnakom rozsahu a s líšiacou sa detailnosťou – pozornosť kladieme skôr na druhú z uvedení tém. Na ich uvedenie a priblíženie uvažujeme nasledujúce otázky:

Úvaha o charaktere znalostí a ich uplatnenia pri riešení úloh

- Keď máme *k dispozícii systém parciálnych diferenciálnych rovníc, ktoré reprezentujú fyzikálne znalosti týkajúce sa zmien tlaku, prietoku a teploty média pretekajúceho v potrubí jadrového reaktora, a použijeme metódu numerického riešenia takého systému rovníc*, dá sa považovať programová realizácia zodpovedajúcich algoritmov v niektorom zo všeobecných programovacích jazykov, resp. tých čo sa používajú v umelej inteligencii (Prolog, LISP, FRL [Frame Representation Language], KRL [Knowledge Representation Language], Clips, a iné) za prejav inteligencie?
- Keď máme *k dispozícii rozsiahlu sústavu algebraických rovníc opisujúcich štrukturálne väzby medzi veľkým počtom ekonomických fenoménov a použijeme metódu lineárneho programovania na nájdenie optimálnych parametrov skúmaného systému*, dá sa považovať programová realizácia zodpovedajúcich algoritmov v niektorom zo všeobecných programovacích jazykov, resp. tých čo sa používajú v umelej inteligencii (pozri v predošlom) za prejav inteligencie?
- Keď máme *za úlohu podporiť diagnostický proces, v ktorom je potrebné interpretovať pozorovaniu dostupné prejavy chybného správania skúmaného systému tak, aby sa zistila zodpovedajúca príčina (chybový stav) a použijeme k tomu známe vzťahy pre výpočet podmienených pravdepodobností napr. na základe Bayesovho vzťahu*, dá sa považovať programová realizácia zodpovedajúcich algoritmov v niektorom zo všeobecných programovacích jazykov, resp. tých čo sa používajú v umelej inteligencii (pozri v predošlom) za prejav inteligencie?

Áno? Nie? A prečo?

Dajú sa očakávať záporné odpovede. Aj napriek tomu, že pri riešení spomínaných úloh sa uplatňuje symbolová reprezentácia poznatkov (stelesnené príslušnými matematickými vzťahmi). Zdôvodniteľnosť zápornej odpovede vyplýva najmä z

toho, že zodpovedajúce výpočtové procedúry, potenciálne akokoľvek zložité, *sú implikované formuláciou úlohy, sú deterministické, sú pomerne jednoducho mechanizovateľné (algoritmizovateľné) a kvantitatívne*. Jadrnejšie sa to dá vyjadriť ešte aj nasledujúcim spôsobom

Mechanické, reproduktívne, postupy nie sú považované za prejav inteligencie

AK POSTUP RIEŠENIA PROBLÉMU SPOČÍVA NA **REPRODUKOVANÍ** VOPRED DANÉHO ALGORITMU, KTORÝ VYPLÝVA Z FORMULÁCIE PROBLÉMU A STELESŇUJE NÁVOD NA RIEŠENIE, TAK TAKÝTO POSTUP JE POVAŽOVANÝ ZA **MECHANICKÝ**.

kognitívnymi (poznávacimi) prejavmi myslenia.

Tie sú charakterizovateľné

- **diskrétnymi, symbolovými, kvalitatívnymi,**
- **prehľadávacimi, rozpoznávacimi,**
- **nemonotónnymi, t.j. predpokladajúcimi, skúšajúcimi, nahrádzajúcimi a najmä nezáväznými (t.j. pripúšťajúcim odstupenie od predchádzajúcich postupov) a aj nedeterministickými krokmi.**

Charakteristické rysy inteligentného riešenia problémov

Na rozdiel od mechanických prejavov, iba ak sa riešiacie postupy prejavujú (aspoň zčasti) uvedenými charakteristikami – nezávisle od toho, či ich nositeľom je človek alebo (jeho výtvor) počítač – sme im náchylní pripísať *atribút inteligentného správania*.

Počítačom realizované symbolové procesy, ktoré potenciálne považujeme za nemechanické, teda také čo pripúšťajú možnosť priradiť im atribút inteligencie, sa dajú charakterizovať takto:

RIEŠENIE PROBLÉMU SPOČÍVA NA ALGORITMOCH VYHLADÁVANIA, ODSKÚŠAVANIA A ZREŤAZOVANIA VHODNÝCH RIEŠIACICH KROKOV (OPERÁCIÍ);

Produktívne riešenie problémov

V PRÍPADE NEZDARU SA OD NICH ODSUPUJE A NAHRADZUJÚ SA INÝMI KROKMI A ICH POSTUPNOSŤAMI;

KEĎ PRIEBEH RIEŠENIA NESPOČÍVA NA VOPRED DANOM NÁVODE, KTORÝ BY VYPLÝVAL Z FORMULÁCIE PROBLÉMU, ALE POSTUPNE SA GENERUJÚ A OVERUJÚ JEDNOTLIVÉ KROKY, VTEDY SA RIEŠIACI PROCES POSTUPNE PRODUKUJE.

Pre kognitívne (inteligentné) správanie, na rozdiel od náhodilého či exhaustívneho (vyčerpávajúceho - systematického - prešetrenia všetkých možností) – teda pre produkovanie riešiacich postupov je nevyhnutné

využívať súbory všeobecných a doplňujúcich špecifických

⁵ Spomeňme si na Turingov test inteligencie.

stratégií spočívajúcich na všeobecných a najmä špecifických znalostiach takým spôsobom, že *minimalizujú výpočtovú zložitosť*, ktorá je vlastná riešeným úlohám.

Z tohto pohľadu možno vnímať expertný systém ako

**PROGRAMOVÝ PRODUKT STELESŇUJÚCI ZNALOSTI
UMOŽŇUJÚCE STRATÉGIAMI RIADENÉ EXPLORÁCIE
(PRESKÚVANIE)
PROBLÉMOVÉHO PRIESTORU
ZA ÚČELOM EFEKTÍVNEHO PRODUKOVANIA
SPRÁVNYCH A VYHOVUJÚCICH RIEŠIACICH
POSTUPOV.**

ES ako
progra-mový
produkt

Expertné systémy majú špecifické danosti a charakteristiky. Medzi ne patria:

- **ich aplikácie:**
slúžia na riešenie problémov, z **formulácie** ktorých pre nás **nevyplývajú deterministické a teda algoritmické riešiacie postupy**; v tejto súvislosti hovoríme o
 - **nedobre formulova(tel')ných** – štruktúra(tel')ných a/alebo
 - **podurčených** (nedobre vymedzených, nedostatočne informovaných –informačne insuficientných) problémoch;pri ich riešení sa uplatňujú **produktívne** riešiacie postupy spočívajúce na **kognitívnych stratégiach**, ktoré - hoci dosiahnutie výsledku *nezaručujú* - vedú spravidla k požadovaným výsledkom, aj keď tie prípadne môžu byť zaťažené neurčitost'ami a preto majú aj **nekategorickú či nejednoznačnú povahu**,
- **ich funkčné vlastnosti:**
stelesňujú **situačné, údajmi** (aj nekategorickými, kvalitatívnymi, neurčitými - "fuzzy", nezáväznými, predpokladateľnými) **podmieňované a riadené prehľadávacie procesy** založené na symbolovej reprezentácii **znalostí** z oblasti, v ktorej vznikol riešený problém,
- **vzájomné väzby ich funkčných zložiek (prvkov štruktúry):**
dochádza medzi nimi k situačnej (oportunistickej) výmene a zdielaniu údajov, k situačnému odovzdávaniu/preberaniu riadenia, čo zabezpečuje realizáciu ich funkcií,
- **metodiky ich tvorby, rozvíjania a udržovania:**
je prípustné autonómne (relatívne nezávislé) vytváranie a úpravy jednotlivých funkčných prvkov systému, čiastkové a postupné rozvíjanie v systéme reprezentovaných znalostí a stratégií ich používania,
- **charakteristiky ich používania:**
spravidla v interakcii s používateľom, niekedy však iba v interakcii s prostredím, pričom v prípade potreby poskytuje používateľovi požadované informácie; pri interakcii s používateľom by malo ísť aj o spôsobilosť systému **vyvetľovať a zdôvodňovať** postup riešenia danej úlohy a odvodených výsledkov, prípadne aj o adaptíbilnosť interakcie v závislosti na danostiach či potrebách používateľa.

a niektoré ich
charakteris-
tické rysy

Aj z uvedeného by malo byť zrejmé, že ES sú nesporne netriviálnymi softvérovými dielami. Dá sa na ne nazerať z viacerých zorných uhlov - v závislosti na tom, čo je ťažiskom pozornosti. Kvôli načrtnutiu čo najúplnejšieho mnohorozmerného obrazu – bez rozboru, iba heslovitým zoznamom rozličných zložiek - naznačíme (snáď aj motivujúce) pohľady, s ktorými sa v súvislosti s týmito systémami dá najčastejšie stretnúť:

- **filozofické**
 - myseľ v neživej matérii;
 - rozsah a ohraničenia realizovateľných prejavov a funkcií myslenia;
- Rôznosť pohľadov na ES • **(kognitívno) psychologické** – neživou materiálou realizovateľné
 - kognitívne, psycho-biologické a psycho-lingvistické fenomény vlastné iba vyšším formám živého;
 - mentálne modely sveta;
 - psychologické poňatie prehľadávacích procesov;
 - psychologické schémy "logického" usudzovania;
 - "techniky" kognitívnych procesov, napr. dedukcia, abdukcia, indukcia; dokazovanie sporom, princípy a mechanizmy asociovania, komplementovania (doplňujúce úvahy o opak);
 - hierarchizovanie, zovšeobecňovanie, generalizovanie;
 - detailizovanie, zjemňovanie, špecializovanie;
 - ohodnocovanie, kvalitatívne porovnávanie, uprednostňovanie;
 - fenomenologické (na základe pozorovania dostupných znakov) a kauzálne (príčinno-dôsledkové) usudzovanie (aj vo vzájomných interakciách);
 - pravdepodobnostné usudzovanie, usudzovanie na základe očakávateľnosti a vierohodnosti;
 - zameriavanie a presun pozornosti, plánovanie;
 - usudzovanie na báze cieľov (intencií), postojov a presvedčení;
 - nemonotónne (nezáväzný) usudzovanie - *čo-ak* a *čo-ak-nie* techniky usudzovania;
 - usudzovanie podmienené kontextom a analógiami (preformulovanie problému);
 - usudzovanie na základe "zdravého" rozumu (tiež zemitého rozumu);
 - rozklad problému na podproblémy a zohľadňovanie vzájomných väzieb podproblémov, syntéza čiastkových výsledkov;
 - usudzovanie o a s danými ohraničeniami, šírenie ohraničení v stavovom priestore problému;
 - lokálne (ohraničené) a globálne (celostné) riešiacie postupy;
 - uplatňovanie nekonzistentných a neúplných znalostí;
 - učenie sa (zhromažďovanie skúseností);
- Filozofické hľadiská • **abstraktné a konceptuálne** – uvažovanie o problémoch v kategóriách
 - riešiteľnosti, stavového priestoru riešenia, princípov a procesov prehľadávania stavového priestoru, metód traverzovania grafov;
 - spontánnej aktivácie procesov, produktívnych riešiacich postupov;
 - reprezentácie a využívanie odborných a všeobecných znalostí človeka;
 - uplatňovania povrchových a hĺbkových štruktúr znalostí;
- Kognitívno-psychologické hľadiská

Abstraktné a
konceptuálne
témy

- údajových štruktúr efektívne reprezentujúcich (dané, zistené, či predpokladateľné) fakty, poznatky, znalosti;
- faktami riadených riešiacich procesov;
- vzájomnej konzistentnosti faktov vo vzťahu k odvodeným výsledkom riešenia a vo väzbe k daným ohraničujúcim podmienkam;
- prostriedkov a metód spracúvania neurčitostí;
- procesov realizujúcich kognitívne (mentálne) spôsobilosti človeka (napr. dedukovanie, asociovanie), uplatňovanie logických princípov v mentálnych operáciách a iné,

Formálno-
logické
otázky

- **formálno-logické**
 - modely uzavretého a otvoreného sveta;
 - princípy intenzionálneho a extenzionálneho vymedzovania (definovania) entít;
 - uplatniteľnosť a ohraničenia logických formalizmov, najmä výrokovej a predikátovej logiky, ale aj temporálnych a iných neklasických logík, vrátane viachodnotových;
 - logické schémy odvodzovania (inferencie); logické aspekty dedukcie, indukcie a abdukcie;
 - princípy a aplikovateľnosť dokazovania teorémov;
 - priamy a spätný chod pri interpretácii produkčných pravidiel;
 - rozšírenie produkčných pravidiel o mimologické konštrukty;
 - nemonotónne odvodzovanie;
 - axiomatizácia, konzistentnosť a úplnosť symbolovej reprezentácie znalostí a pod.;

Psycho-
lingvistické
témy

- **(psycho)lingvistické**
 - možnosti komunikovania s ES v prirodzenom jazyku (písmom aj hovorom);
 - porozumenie (bežnému) významu textov, prípadne aj kontextuálne či situácie zmeneným významom;
 - intencie (snahy, úmysly, ciele) komunikujúceho;
 - interakcie s používateľom umožňujúce odstraňovanie alebo aspoň minimalizovanie nejednoznačností a nedorozumení;
 - transfér informácií a znalostí výrazovými prostriedkami prirodzeného jazyka;

Informaticko-
teoretické
hľadiská

- **informaticko-teoretické**
 - mnohorozmerné väzby na logiku a matematiku;
 - konzistentnosť, bezospornosť a spoľahlivosť reprezentovanej množiny poznatkov;
 - všeobecná pamäťová a výpočtová zložitosť odvodzovania a špecifická pamäťová a výpočtová zložitosť vzťahujúca sa k čiastkovým procesom zodpovedajúcim javom korešpondujúcim k uplatňovaným modelom kognitívnych procesov;
 - procedurálne a deklaratívne (situčné) programovanie, evolučné (trénovaním sa zdokonaľujúce) programy (neurónové siete, genetické algoritmy);
 - paralelizmus výpočtových procesov a pod.;

Realizačné
hľadiská

- **informaticko-realizačné**
 - princípy štruktúrovania a programovej realizácie ES;
 - výber programových jazykov, prostriedkov, prostredí a metód programovania;
 - formy symbolovej a subsymbolovej reprezentácie poznatkov a znalostí, ich vyjadrovacia, odvodzovacia a výpočtová účinnosť;
 - explicitné a implicitné prostriedky a formy symbolovej reprezentácie rôznych a principiálne odlišných poznatkov a znalostí;
 - princípy adekvátnej voľby a realizácie riešiacich postupov, stratégií, metód a prostriedkov riešenia problémov, ich evidovanie, revidovanie a nahradzovanie;
 - spôsoby navrhovania a dizajnu systému;
 - formy interakcie s používateľom, grafické a multimédiové rozhrania;
 - interakcie s údajovým a programovým prostredím systému;
 - metodológia programovania ES, najmä vzhľadom na previazanosť, inicializovanie, suspendovanie (prerušenie), obnovenie a ukončenie jednotlivých procesov;
 - riadenie a trasovanie procesov odvodzovania (inferencie), tvorba špecifickými údajmi (situačne) inicializovaných procesov (démony, metódy);
 - riadiace a sledovacie údajové štruktúry, metodológii tvorby a testovania súvisiacich programových systémov a pod.;
- **odborníka aplikačnej oblasti - (spolu)tvorca problémovo zameraného systému**
 - rozsah, možnosti, spôsoby, prostriedky vyjadrovania a reprezentácie znalostí z určitej problémovej oblasti;
 - koexistencia rôznych, aj vzájomne nekonzistentných odborných znalostí a postupov ich uplatňovania;
 - metódy získavania, formulovania, transféru znalostí odborníka do symbolovej podoby a zodpovedajúcej reprezentácie v systéme;
 - kontrola konzistencie a úplnosti reprezentovaných znalostí;
 - overovanie kvality reprezentovaných znalostí a použiteľných metód odvodzovania;
 - aktualizácia, modifikovanie, rozvíjanie a zdokonalovanie systému reprezentovaných znalostí, vrátane metód ich uplatňovania;
- **pragmatické (cieľového používateľa)**
 - poslanie, uplatniteľnosť a opodstatnenosť ES v jednotlivých aplikačných oblastiach;
 - účelnosť a užitočnosť ES v rukách cieľového používateľa v roli prostriedku na riešenie daných problémov zaručujú efektívnosť vynaložených investícií;
 - jednoduchosť, zrozumiteľnosť a pohodlnosť používania systému.

Pohľady
odborníka z
aplikačnej
oblasti

Ppragmatické
hľadiská

S mnohými z týchto hľadísk a s nimi súvisiacich tém sa zaoberáme v nasledujúcich textoch. V niektorých prípadoch iba stručne, inokedy podrobne a aj z viacerých pohľadov. Je to dané cieľmi výučby, prípustným rozsahom prednášok a prijá-

teľným rozsahom týchto textov.

1.2 Poslanie expertných systémov

Ako už bolo uvedené, programové produkty stelesňujúce princípy *umelej inteligencie* majú zabezpečovať také správanie počítača, resp. nim ovládaného zariadenia, ktoré v určitej aplikačnej oblasti dostatočne presvedčivo pripomína prejavy prirodzeného fenoménu ľudskej inteligencie. Aplikačných oblastí tohto druhu je nepreberné množstvo. Patria medzi ne napríklad

Ilustrácie
aplikačných
oblastí UI

- intelektuálne náročné hry (napr. šach, dáma), riešenie rébusov a hlavolamov (napr. kryptoaritmetické úlohy, lúštenie krížoviek) a rad ďalších,
- vykonávanie operácií s formulami výrokovej a predikátovej logiky, dokazovanie teorémov,
- symbolové matematické operácie (napr. algebraické výpočtové procesy, riešenie úloh z geometrie, derivovanie a integrovanie, rozvoj do radov, štandardné aj kvalitatívne riešenie sústavy rovníc a nerovností, kvalitatívne modelovanie a simulácie a iné), riešenie slovne formulovaných výpočtových úloh,
- deduktívne dátábázové systémy a inteligentné dialógové systémy uplatňujúce sa v tradičných informačných, hospodárskych, technických a vedeckých aplikáciách,
- interpretovanie napísaných aj vyslovovaných textov prirodzeného jazyka, preklad textov z jedného do iného prirodzeného jazyka, analýza príbehov a ich generovanie,
- počítačové videnie, rozpoznávanie a porozumenie obrazov a scén,
- kognitívna robotika (inteligentné roboty),
- automatizovaná tvorba programov,
- učenie sa a podpora výučby,
- interpretácia dostupných faktov v procesoch biologického aj technického diagnostikovania (identifikovanie priamemu pozorovaniu nedostupného stavu skúmaných systémov), monitorovanie (priebežná interpretácia dostupných faktov), určovanie spôsobu odstraňovania neželaných stavov (porúch, chorôb),
- zostavovanie plánov, odvodzovanie postupov a riadiacích postupov,
- riešenie konštrukčných úloh, navrhovanie zariadení.

Iba niektoré z vyššie uvedených kategórií úloh riešených metódami, ktorým sa pripisuje atribút "kognitívne" tvoria doménu expertných systémov. Sú to tie, ktorých riešenie spočíva na uplatňovaní **odborných poznatkov z príslušnej problemovej oblasti**.

Neinformované
postupy

Riešenia úloh uvedených typov spočívajú zväčša na vzájomne (aj výrazne) odlišných princípoch a postupoch. UI **naivné – neinformované – postupy**, t.j. buď **exhaustívne** (všetko systematicky a vyčerpávajúco preverujúce) alebo **náhodilé** (slepé), zo svojej pozornosti vylučuje. Nie sú to postupy, ktoré by mali vzory v prirodzenej inteligencii (neboli by prejavom inteligencie ani keby ich uskutočňoval človek), ale aj preto, že pri riešení netriviálnych problémov je ich uplatňovanie neefektívne a často nepoužiteľné v dôsledku neprijateľnej výpočtovej zložitosti.

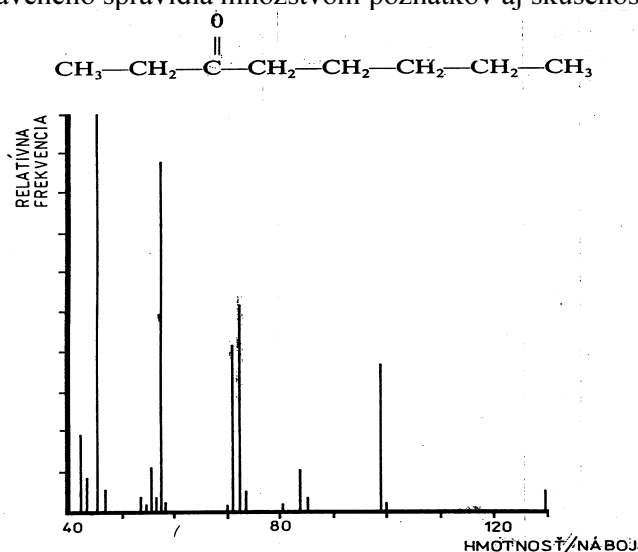
Slabé metódy

Rané obdobie rozvoja UI bolo poznačené vyvíjaním **univerzálnych (všeobec-**

riešenia problémov

ných) riešiacich postupov (mechanizmov). Produkovanie riešiaceho postupu spočívalo na nachádzaní všeobecných operácií, t.j. *elementárnych krokov usudzovania* a ich postupným zreťazovaním s cieľom *identifikovať/odvodiť/nájsť výsledok*. Uplatňovanie všeobecných riešiacich krokov - práve pre ich univerzálnosť - je lákavé aj z hľadiska programovej realizácie. Ale, a to je podstatné, pri riešení netriviálnych problémov sú všeobecné operácie málo účinné - sú slabé⁶. *Všeobecné operácie (usudzovania)* práve kvôli svojej univerzalite nemôžu byť informované o konkrétnej realite ovplyvňujúcej a podmieňujúcej povahu problémov a ich riešení. Práve preto sa riešiace metódy spočívajúce iba na takých operáciách klasifikujú ako **slabé metódy**. Prekonať sa dajú iba použitím *poznatkov z oblasti, v ktorej problém vzniká*. Poznanky sú prostriedkom umožňujúcim uskutočňovať **cielené (špecifické, problémovo-orientované)** riešiace kroky, ktoré zodpovedajú postupom experta (vybaveného spravidla množstvom poznatkov aj skúseností).

a ich prekonávanie



Obr. 1

DENDRAL -
implikácia
nevyhnutnosti
špecifických
poznatkov

Uvedený výrok sa javí až príliš samozrejým. A predsa nebol takým v počiatkových fázach vývoja ES. V tejto súvislosti je účelné spomenúť historicky mimoriadne významnú úlohu, ktorú zohral systém DENDRAL (*Feigenbaum, Buchanan, Lederberg, 1969*). Tento pozoruhodný a úspešný programový produkt sa stal pragmatickým dôkazom pravdivosti vyššie uvedeného výroku. Jeho autori⁷ sa rozhodli vytvoriť systém schopný odvodiť štruktúru chemickej molekuly z údajov, ktoré poskytuje hmotový spektrometer. Vstupom do programu bol sumárny vzorec molekuly (napr. $C_6H_{13}NO_2$) spolu s korešpondujúcim nameraným spektrogramom

⁶ Nepochybne, všeobecný gravitačný zákon, za formuláciu ktorého vďačíme Newtonovi, nemožno ani na okamih spustiť zo zreteľa. Avšak úspešný štart a pristávanie lietadla, popri striktnnej konformite so všeobecnými zákonmi gravitácie, mechaniky a aerodynamiky, nevyhnutne vyžaduje zohľadňovanie ďalších špecifických fyzikálnych a technologických zákonitostí vyplývajúcich aj z vlastností daného lietadla, špecifickej aktuálnej poveternostnej situácie, smeru vetra, viditeľnosti, charakteru štartovacích/pristávacích dráh, aktuálnej premávky na letisku i v jeho okolí, vrátane potenciálneho výskytu vtáctva v danom priestore, ako hoci aj prípadného opotrebovania, či. poškodenia niektorej pozornosť si zasluhujúcej súčiastky. Jednoducho, len s gravitačným a ani s ostatnými všeobecnými fyzikálnymi zákonmi sa nedá vystačiť, hoci je ich nevyhnutné rešpektovať.

⁷ Ed Feigenbaum (pôvodne študent Herberta Simona), Bruce Buchanan (pôvodom filozóf, ktorý sa stal informatikom) a Joshua Lederberg (genetik, nositeľ Nobelovej ceny).

Ilustrácia slabej exhaustívnej metódy

informujúcim o hmotnosti rôznych zložiek molekuly (pozri obr. 1). Program obsahoval množstvo poznatkov o rôznych možných tvaroch a štruktúrach spektrogramov zodpovedajúcich reálne existujúcim chemickým látkam.

Tak napríklad stelesňoval poznatok, že pík (vrchol) v pozícii zodpovedajúcej hmotnosti $m=15$ v hmotovom spektrograme molekuly svedčí o prítomnosti metylovej zložky (CH_3) v tejto molekule.

Princíp pôvodnej naivnej verzie systému odvodzovania štruktúry molekúl z ich sumárneho chemického vzorca a zodpovedajúceho hmotového spektrogramu je možné vyjadriť v nasledujúcej podobe:

- (1) systém vygeneruje všetky možné chemické štruktúry konzistentné so zadaným sumárnym vzorcom,
- (2) každej vygenerovanej chemickej štruktúre priradí (vygenerovaný alebo už pamätaný) tvar zodpovedajúceho hmotového spektrogramu,
- (3) vhodnou metódou nachádzania zhody vyhladá spomedzi priradených spektrogramov taký, ktorý je v najväčšej zhode s práve skúmaným spektrogramom,
- (4) vybraný spektrogram určuje zisťovanú štruktúru molekuly.

Aplikácia tejto metódy narazila však na vážny problém *bariéry výpočtovej zložitosti*. Je to dané tým, že jedinému sumárnemu vzorcu už veľkej molekuly môže zodpovedať až prekvapujúco veľký počet prípustných spektrogramov.

Ilustrácia špecifického poznatku v podobe produkčného pravidla

Autori systému DENDRAL-u konzultovali tento problém u expertov z oblasti analytickej chémie. Zistili, že títo namiesto exhaustívnych postupov uplatňujú postupy cieleň: *Vyhľadávajú pre reálne existujúce podštruktúry molekuly s daným sumárnym vzorcom dobré známe zoskupenia (útvary, vzorce) pík v hmotovom spektrograme*. Také, čo sú typické. Napríklad, keď sa dá predpokladať, že sa v skúmanej molekule vyskytuje *ketónová podskupina* ($\text{C}=\text{O}$), pri analýze spektrogramu, v snahe jej výskyt potvrdiť, použijú nasledujúce pravidlo⁸:

AK v hmotovom spektrograme existujú dve pozície \mathbf{x}_1 a \mathbf{x}_2 , v ktorých sú píky také, že

- (a) $\mathbf{x}_1 + \mathbf{x}_2 = M + 28$ (M je hmotnosť celej molekuly),
- (b) $\mathbf{x}_1 - 28$ je vysoký pík,
- (c) $\mathbf{x}_2 - 28$ je vysoký pík,
- (d) aspoň jeden z \mathbf{x}_1 , \mathbf{x}_2 obsahuje vysoký pík,

TAK molekula obsahuje ketónovu podskupinu.

Rozpoznanie výskytu určitých konkrétnych prvkov štruktúry molekuly výrazne znižuje počet alternatív následnej analýzy a tým sa riešenie problému stáva priechodným, alebo aspoň priechodnejším. Zmenou pôvodného prístupu k tvorbe DENDRAL-u tým, že sa v ňom uplatnili pravidlá (sú reprezentáciou špecifických poznatkov) uvedeného typu vznikol praktický a výkonný systém. Autori to vysvetlili takto:

Poznatkovo intenzívny systém

"Všetky relevantné teoretické znalosti potrebné na vyriešenie predmetných problémov sa pretransformovali z ich všeobecného tvaru [v zlož-

⁸ Pravidlá tohto typu sa nazývajú **produkčné**. Podrobne sa nimi zaoberáme v ďalšom. Podoba pravidla zodpovedá tvaru, v ktorom je uvádzaný v (Russell, Norvig, 1995).

ke generovania zodpovedajúcich hmotových spektrogramov] ("počítačový princíp") do účinných špeciálnych tvarov ("receptov v kuchárskej knižke")." (Feigenbaum et al. 1971)

Význam úspešného DENDRAL-u spočíva(l) v tom, že odhalil

- (1) *možnosť a spôsob spôsob vytvárania poznatkovo-intenzívneho (informovaného) systému uplatňovaním explicite symbolovo reprezentovaných (špecifických) znalostí reprezentovaných adekvátnym počtom expertami používaných špeciálnych účelových pravidiel,*
- (2) *princíp umožňujúci vytvárať programy schopné riešiť problémy, ktorých riešenia pri klasických metódach programovania narážali na medze výpočtovej zložitosti.*

Kľúčová úloha znalostí

DENDRAL sa stal zdrojom poznania:

Efektívne riešiacie postupy takých úloh, ktoré svojou formuláciou riešiaci postup neimplikujú, spočívajú na účelnom používaní potrebného rozsahu všeobecných a/alebo špecifických znalostí z oblasti, v ktorej riešený problém vznikol.

ZNALOSTI sú prostriedkom umožňujúcim STRATÉGIAMI RIADENÉ EXPLORÁCIE PROBLÉMOVÉHO PRIESTORU pri efektívnom produkovani správnych riešiacich postupov.

V období formulovania týchto výrokov vôbec sa nezdali takým samozrejým a prirodzeným ako ich vnímame v súčasnosti, keď sa všeobecne prijímajú nasledujúce dve vymedzenia

Znalostné systémy

PROGRAMOVÉ SYSTÉMY, KTORÉ OPIERAJÚ SVOJU ČINNOSŤ O SYMBOLOVÚ REPREZENTÁCIU ZNALOSTÍ, NAZÝVAME ZNALOSTNÉ SYSTÉMY (NA ZNALOSTIACH ZALOŽENÉ SYSTÉMY)

a tie z nich

Expertné systémy

ČO SÚ STELESNENÍM ODBORNÝCH ZNALOSTÍ Z URČITEJ PROFESNEJ OBLASTI A SLUŽIA NA PODPORU (PRÍPADNE AJ NAHRADENIE) ODBORNÍKOV-EXPERTOV - PRI RIEŠENÍ ODBORNÝCH PROBLÉMOV (TVORIA PODMNOŽINU ZNALOSTNÝCH SYSTÉMOV) ZODPOVEDAJÚ EXPERTNÝM SYSTÉMOM.

Je vôbec možné takéto systémy vytvoriť? Ak áno, za akých okolností?

Pokúsme sa najprv hľadať a nachádzať odpoveď vo filozoficko-informačno-psychologickej rovine.

1.3 *Filozoficko-informačno-psychologické pozadie inteligentných symbolových systémov*

Trocha filozofie a psychológie v informatike

Úvahy o tvorbe programových produktov manifestujúcich *fenomén inteligencie* je vhodné aspoň v krátkosti zasadiť do rámca niektorých východiskových

filozoficko-informačno-psychologických aspektov. Tomu venujeme pozornosť v tomto článku. Ani zďaleka si pritom nekladíme za cieľ poskytnúť ucelený a už vôbec nie vyčerpávajúci pohľad na problematiku. Popri uvedení viacerých pozorností si zasluhujúcich výrokov popredných odborníkov, čo by nepochybne mohlo naznačiť "príchut'" danej témy, je naším prioritným zámerom motivovať čitateľa k samostatnej analýze náročnej problematiky aj na základe uvádzaných hypotéz. Dá sa predpokladať, že by sa potom téma *umlá inteligencia a expertné systémy* mohla vnímať v podstatne bohatších a obohacujúcich hľadiskách než je iba informaticko-realizačný pohľad.

Bežné laické, ale často aj odborné pozorovanie a seba pozorovanie (introsepekcia) úspešných procesov myslenia obchádza otázku **povahy týchto procesov, spôsobov ich uskutočňovania**, ba ani túto otázku neberie na vedomie⁹. Vedeckým skúmaním procesov myslenia sa zaoberá **kognitívna psychológia**. Tá vytvára *pracovné modely procesov myslenia*. Skúma myslenie človeka práve z pohľadov súvisiacich s predmetom nášho záujmu.

Ústredná idea kognitívnej psychológie sa dá formulovať takto:

Myslenie

Myslenie spočíva v interných manipuláciách s MENTÁLNYMI MODELMI sveta, ktoré si ľudia vytvárajú neartikulovateľnými mentálnymi procesmi.

Mentálny model – začlenená sústava poznatkov

Mentálnym modelom sa rozumie (určité nevyhnutné) *zoskupenie znalostí o existencii, príčinách, princípoch a spôsoboch vzniku, priebehu (trvania) a zániku fenoménu/javu, jeho vlastnostiach, štruktúre (zloženia) a správaní, o jeho príčinách, dôsledkoch, pôsobení, implikáciách, o jeho ovplyvniteľnosti, riadení, vyvolaní, znemožnení, o jeho vzťahu a podobnosti k iným záležitostiam, o predvídateľnosti jeho vzniku, zániku a správania.*

Na základe mentálneho modelu má človek očakávania, aspirácie, volí si ciele, tvorí plány, selektívne vyberá hypotézy a informácie, ktoré vyhodnocuje a spracúva, prijíma, vytvára premisy a propozície a aj na ich základe hypotézy dokazuje či zamietá, potvrdzuje alebo sponchybuje.

Mentálny model je fenomén, na základe ktorého v interakcii individua s okolím (s inými individuími, spoločnosťami, so živou aj neživou prírodou, s fyzickými či duchovnými dielami človeka) dochádza k procesom **porozumenia**. Sú to procesy, v ktorých individuum

- (a) prijímané informácie dokáže bezprostredne (významovo a propozíčne) **interpretovať v rozsahu svojho mentálneho modelu** a/alebo
- (b) vďaka prijímaným informáciám **modifikuje svoj mentálny model** (doplňuje, mení, vylučuje niektoré jeho prvky, vzťahy medzi nimi, premisy a propozície), čím im priradzuje (významovú a propozíčnú) interpretáciu. Možno vysloviť¹⁰ tvrdenie

⁹ Nekladie si otázky typu: Prečo nedokážeme súčasne rozmýšľať o viacerých záležitostiach, aký je proces porozumenia významu slov, čo sa odohráva v našej myšli keď násobíme 7 so 4, čo je podstatou procesu zapamätávania a neskoršieho vybavovania zapamätaného, prečo rovnako znejúcu úlohu odborníci z príslušnej profesie riešia rozličnými postupmi, rôznou rýchlosťou a s líšiacou sa úrovňou kvality?

¹⁰ Je to výrok, ktorého význam sa znovu vynorí pri rozbere **vysvetľovacieho modulu** expertného systému.

Porozumenie

Porozumenie niečoho, hoci len čiastočné, predpokladá vytvorenie zodpovedajúceho mentálneho modelu, t.j. určitého vzájomne previazaného (začleneného) súboru súvisiacich poznatkov vytvárajúcich znalosť daného, resp. presvedčenie o danom.

Spôsobilosť čosi **vysvetliť** predpokladá, že danému bolo porozumené. Zároveň úspešné vysvetlenie je potvrdením porozumenia. Proces vysvetľovania sa odohráva najmenej medzi dvoma účastníkmi **spôsobilými porozumieť téme**. To znamená, že príjemca vysvetľovania musí už disponovať určitými poznatkami – **mentálnym modelom**, aby výklad mohol pochopiť.

Model

Model, nezávisle na materiálnej podstate jeho realizácie, by mal reflektovať ŠTRUKTÚRU modelovanej predlohy a musí byť spôsobilý uskutočňovať analogické PROCESY (CHOVANIE) ako nim simulovaná (imitovaná) predloha (vzor-systém).

V ohnisku našej pozornosti je teda principiálna otázka:

Dajú sa vytvoriť modely, ktoré funkčne umožňujú simuláciu myslenia?

Otázka je zrozumiteľná a na prvý pohľad dobre formulovaná. Dokážeme sa však jasne a jednoznačne dohodnúť na významovom obsahu a rozsahu pojmu **MYSLIEŤ**?

Čo označujeme termínom 'myslieť'?

Jednoznačne kladnú odpoveď možno očakávať iba s malou pravdepodobnosťou. Z toho potom plynie, že nájdenie odpovede na vyššie položenú otázku je náročné. Potvrďuje to už niekoľko desaťročí trvajúce a od ukončenia stále vzdialené poznávacie úsilie filozofov aj psychológov. Filozofov práve preto, lebo vo vedných odboroch, ktorým sa priraduje predikát *exaktné*, absentujú jednoznačné prostriedky umožňujúce vyhľadávať odpovede a dokazovať ich korektnosť.

V ďalšom, bez poskytovania dôkazov, vychádza sa z predpokladu, že **pre modely a simuláciu myslenia** nemusí byť podstatným ľudský mozog (prirodzený nositeľ – realizátor myslenia) so svojimi mechanizmami vnímania a produkovania akcií. Ide pritom o hypotézu **symbolizmu**. Podľa nej

Abstrakcia: symbolové operácie

operácie so symbolmi, t.j. ich vytváranie/rušenie, spájanie/rozčleňovanie, štruktúrovanie, menenie, porovnávanie, zaradovanie/preraďovanie/prieraďovanie a podobne, umožňujú realizáciu modelov a sú prostriedkom simulácie procesov myslenia - prirodzene iba vtedy, keď je "svet" vhodne reprezentovaný zodpovedajúcimi symbolmi.

.....**Táto hypotéza** - spočívajúca okrem iného na skutočnosti, že aktivity neurónov v mozgu sa prejavujú procesmi ich *excitácie* a *inhibície* (*excitované: zapnuté-prenášajú signál, inhibované: vypnuté-neprenášajú signál; môžu byť teda iba v nám dobre známých binárnych stavoch*) - implikuje možnosť vytvárať prostriedkami počítača modely myslenia. Hypotéza korešponduje so snahami UI **počítačom realizovať pracovné modely myslenia a jeho simulovanie**.

V nasledujúcom uvedieme niekoľko hypotéz, ktoré prispievajú k rozvíjaniu a

približovaniu predstáv o spôsobilosti počítača *myslieť*, resp. vytvárať v niektorých oblastiach *zdanie myslenia*.

Treba však zdôrazniť, že jestvuje nezanedbateľný počet renomovaných odborníkov, ktorí ich prijímajú s výhradami, prípadne ich aj s implikovanými dôsledkami odmietajú. Myslenie považujú za prejav výlučne ľudského (prípadne ešte živočíšného) mozgu (*wetware*). Pokiaľ ide o hardvér (*dryware*), jeho spôsobilosť myslieť až radikálne vylučujú, hoci pripúšťajú, že zdanie myslenia dokážu vyvolať. [Záujemcom o hlbšie preniknutie do tejto problematiky možno odporučiť rad literárnych prameňov. Spomedzi oponentov strojového myslenia treba siahnúť najmä po dielach autorov Sarl a Penrose. Pozornosť si zasluhuje súhrn náhľadov na túto problematiku ako ju uvádzajú Russell, Norvig, 1995, kapitola 26.]

Ponuka niekoľkých hypotéz

Prvá z nasledujúcich hypotéz je reflexiou materiálnej podstaty myslenia a druhá je vlastne reflexiou existencie psychológie.

Mozog realizuje rekurzívne funkcie

H-1 Materiálna podstata neurofyzilogického substrátu (mozgu), bez toho, aby kládla ohraničenia, poskytuje biofyzikálne základy procesov myslenia tým, že má schopnosť realizovať rekurzívne funkcie - teda to čo sa dá uskutočňovať Turingovým strojom.

Skúmanie myslenia

H-2 Myslenie sa dá skúmať nezávisle od skúmania mozgu, t.j. nezávisle od neurofyziológie.

Nasledujúcich 6 hypotéz o myslení pochádza z "myšlienkovvej dielne" významného kognitívneho psychológa Johnson-Lairda z rokov 1987-89. Ako psychológ hlboko prenikol do teórie informatiky aj praktických otázok programovania. Vďaka tomu jeho psychologicko-informatické hypotézy nepochybne poskytujú množstvo motivujúcich podnetov.

Psychologická teória a TS

H-3 Psychologická teória sa dá považovať za schopnú čosi vysvetliť iba vtedy, keď je ju možné formulovať ako efektívnu procedúru (Turingov stroj - TS).

Psychologická teória a vypočítateľné funkcie

H-4 Psychologické teórie (*pokrývajúce rozsiahle oblasti myslenia*) sa dajú poňmať ako vypočítateľné funkcie. Jestvujú však také aspekty ľudskej mentality, ktoré sa nedajú vystihnúť žiadnou teóriou. Zároveň však nemožno vylúčiť rozšírenie konceptu efektívnej procedúry (TS) tak, aby bol schopný zahrnúť viac než množinu súčasne vypočítateľných funkcií a tým pokryť aj ďalšie mentálne fenomény.

Produkčné pravidlá a možnosť tvorby teórií kognície

H-5 Produkčné pravidlá vytvárajú gramatiku, ktorá je ekvivalentná univerzálnemu TS, pričom sú považované za základňu psychologicky orientovaných teórií vypočítateľnosti, ktoré umožňujú tvorbu špecifických teórií kognície.

Mentálne a výpočtové procesy

H-6 Funkčná organizácia mentálnych procesov môže byť charakterizovateľná pojmom efektívnych procedúr, pretože schopnosť mysle vytvárať pracovné modely zodpovedá výpočtovému procesu.

O možnosti simulovať

H-7 Akceptovanie hypotézy H-6 (funkcionalistický pohľad) implikuje

teórie myslenia

- (a) možnosť vhodnými programami simulovať teórie myslenia
- (b) principiálne aj možnosť nimi (aspoň z časti) realizovať racionálne myslenie.

H-7 hovorí o tom, že myslenie, pokiaľ je v určitom rozsahu stotožniteľné s výpočtovým procesom, dá sa realizovať aj počítačom.

Myslenie a vypočítateľnosť

H-8 Ak Turingové tézy a funkcionálny pohľad sú správne, tak vznikajúce teórie myslenia budú vyjadriteľné v termínoch vypočítateľnosti.

K hypotéze H-8: Problém spočíva v tom, že k danej vypočítateľnej funkcii existuje mnoho efektívnych procedúr, ktoré sa môžu vzájomne až principiálne líšiť. Z toho plynie, že aj keby sme presne poznali funkcie výpočtovo zodpovedajúce jednotlivým mentálnym procesom, aj tak by vznikli problémy s nachádzaním (identifikáciou) procedúr, ktoré sú ich schopné vypočítať (Marr, 1977). Napriek tomu tvorba programových systémov **schopných vytvárať pracovné modely myslenia** je oprávneným predmetom teoretického aj praktického záujmu UI. Opiera sa o zásadné myšlienky o možnostiach využívania počítačov v úlohe **prostriedku modelovania procesov operácií so symbolmi**, ktoré zodpovedajú najcharakteristickejšej ľudskej činnosti - mysleniu. Základy týchto myšlienok vznikli už v priebehu *Dartmouth Summer Conference on Artificial Intelligence* v roku 1956. Následný vývoj vhodných programovacích prostriedkov a experimentovania s nimi vytvorili predpoklady na vyslovenie základnej hypotézy motivujúcej rozvoj UI. Vyslovili ju A. Newell a H.A. Simon v r. 1976 a nazývajú sa **hypotéza fyzikálneho symbolového systému**. [Medzi výrazných oponentov dá sa radiť J. R. Searle. Vyplýva to aj z jeho diela *Minds, Brains and Science*, Harvard University Press, Cambridge, Ma, 1984, český preklad *Mysl, mozek a věda*, Mladá fronta, Praha 1994.] Jej formulácia je nasledovná:

Hypotéza fyzikálneho symbolového systému

H-9 Číslicový počítač splňuje nutné a postačujúce podmienky pre vykonávanie inteligentnej činnosti.

E. A. Feigenbaum, vedúca osobnosť kolektívu tvorcov systému DENDRAL zo Stanfordskej university, obohacuje H-9 tzv. **znalostnou hypotézou**:

Znalostná hypotéza

H-10 Systém (počítač) dokáže vykonávať inteligentnú činnosť na vysokej úrovni kompetencie vďaka znalostiam, ktoré má z oblasti svojich aktivít (znalosti zodpovedajúce oblasti, v ktorej riešený problém vzniká).

Pokračujúci rozvoj poznávania rozvíja a ďalej obohacuje základné myšlienky. Príkladom je B. Smith z Massachusetts Institute of Technology, ktorý v roku 1982 významne rozvinul hypotézu H-10, tým že formuloval nasledovnú hypotézu (nazvime ju **propozičná**):

Propozičná hypotéza

H-11 Každý (určitým mechanizmom) realizovaný inteligentný proces je tvorený štruktúrnymi prvkami, ktoré

- (a) sa vonkajšiemu pozorovateľovi správania systému javia ako *propozičný prejav znalostí*,
- (b) *nezávisle od vonkajšieho (sémantického) hodnotenia zabezpečujú (sú príčinou) správania systému, ktoré je manifestáciou jeho znalostí.*

a jej implikácie

Táto hypotéza osvetľuje významné stránky (mechanickej) inteligencie:

- (a) **Nevyhnutnosť symbolovej reprezentácie znalostí *propozície interpretovateľnými prvkami*** (možno im priradovať pravdivostné hodnoty) **vytvárajúcimi konzistentne interpretovateľné vety určitého jazyka.**
- (b) **Inkorporovanie zodpovedajúcich viet do systému** (mechanickej inteligencie) **je nevyhnutným predpokladom jeho inteligentného správania.**

Uvedené hypotézy rozvíjajú a spresňujú výroky týkajúce sa počítačom realizovaných procesov s atribútom inteligentného správania. Hovoria o tom, že také procesy

NEVYSTAČIA S REALIZÁCIOU IBA VOPRED DANÝCH JEDNOZNAČNÝCH DETERMINISTICKÝCH NÁVODOV.

Fenomé-
n inteligentného
procesu

Schopnosť manifestovať fenomén inteligencie spočíva na

SITUAČNOM UPLATŇOVANÍ (PREHLADÁVACÍCH A REVOKOVATEĽNÝCH, T.J. ODVOLATEĽNÝCH) PROCESOV, KTORÝCH POSTUPNOSŤ NIE JE VOPRED DETERMINISTICKY DANÁ.

a človek

Je to v súlade s poznatkami o myslení človeka. Ten

- *spravidla dokáže svoje špecifické, t.j. problémovo zamerané, a všeobecné znalosti efektívne situačne používať aj v prípade problémov, pre riešenie ktorých nemá vopred daný presný deterministický predpis; dokáže teda vykonať viac, než na čo je "naprogramovaný"; navyše dokáže svoje znalosti rozširovať, prehĺbovať, zdokonaľovať,*
- *keď vôbec nemá k dispozícii predpis na riešenie daného problému, alebo len taký, ktorý je neúplný, nedeterministický, alebo inak nezaručujúci rezultatívnosť, používa rôzne (MENTÁLNE) TECHNIKY VYHLADÁVANIA VYUŽITEĽNÝCH RIEŠIACICH KROKOV K VYTVÁRANIU TAKEJ REVOKOVATEĽNEJ POSTUPNOSTI Z NICH, KTORÉ MU NAPOKON UMOŽNIA DOSPIEŤ K ŽELANÉMU VÝSLEDKU - TEDA POSTUP RIEŠENIA SI VYTVARA, PRODUKUJE.*

Obdobné spôsobilosti sa očakávajú aj od fyzikálneho symbolového systému pokiaľ má patriť do kategórie informačných technológií majúcich oprávnenie na atribút *inteligentná*. V predošlom uvedené hypotézy podporujú presvedčenie, že by to malo byť v určitom (doteraz definitívne nevymedzenom) rozsahu možné:

a počítač

POČÍTAČ BY MAL BYŤ SPÔSOBILÝ VYKONAŤ VIAC, NEŽ MU JE PROGRAMOM PREDPÍSANÉ!

Fyzikálne princípy (pneumatické, hydraulické, reléové, elektrónkové, polovodičové, integrované obvody vysokej hustoty) a funkcie (registre, organizácia ALU, pamäte, vykonateľné kódy) konkrétneho počítača nie sú z tohto pohľadu relevantné.

Podstatným je
realizovateľ-
nosť funkcií

Podstatné sú **funkcie** počítačom vykonávaného **programu**.

Ten istý program môže byť preložený do úplne iného vykonateľného kódu zbiehateľného na konštrukčne inom počítači, ktorý uskutočňuje svoje činnosti (aj

fyzikálne) odlišným spôsobom a napriek tomu realizuje tú istú funkciu (konformné s H-2). Teda pre dokonalé pochopenie činností, ktoré môže vykonávať *fyzikálny symbolový systém*, je podstatné pochopiť funkcie programu a princípy umožňujúce realizáciu želaných funkcií.

Mnohé z týchto princípov sú už známe. Spoznalo sa, že ich programová realizácia vyžaduje uplatňovať ďalší **štýl programovania**. Jednou z podstatných možností je doplnenie (nahradenie) **procedurálneho** programovania **deklaratívnym**. To je téma, ktorej je venovaná väčšina z nasledujúceho výkladu.

1.4 Procedurálne a deklaratívne programovanie

Neuronové (konekcionistické) siete, genetické algoritmy, objektovo-orientovaná a deklaratívna paradigma programovania vyvrelí z úsilia riešiť triedy problémov, ktoré sa stali a stávajú predmetom pozornosti a potreby až v pomerne nedávnom období. Výrazný rozvoj technických prostriedkov (hardvér aj softvér) a vedeckých poznatkov sú predpokladom toho.

Skutočnosť, že formulácie uvažovaných tried problémov neimplikujú zreteľne formulované riešiacie postupy v klasickej algoritmickej podobe, vedie k potrebe rozšíriť predchádzajúce ponímanie prístupov k programovaniu. Dôsledkom je aj vznik vyššie spomínaných paradigiem. V tomto kontexte je dôležité ozrejmiť predovšetkým odlišnosti a vzťah medzi **klasickým procedurálnym** a iným (novším) **deklaratívnym štýlom programovania**.

Princíp klasického **procedurálneho** spôsobu **programovania** je tvorba postupnosti príkazov, pre ktorú platí:

Ak z práve vykonávanej inštrukcie priamo nevyplýva inštrukcia, ktorá sa má následne vykonať, tak sa vykoná tá, ktorá bezprostredne nasleduje v poradí ich usporiadania.

Podstatou uvedeného je veľmi **jednoduchý princíp riadenia** zabezpečujúci zbiehanie procedurálneho programu: Nech **n** je prirodzené číslo označujúce poradie inštrukcie v programe. Nech **m** je celé číslo a NIL zodpovedá prázdnej inštrukcii. Princíp riadenia možno potom formulovať v nasledujúcom **funkčnom vyjadrení**

$$f(n) = \begin{cases} n+m & \text{(v prípade odskoku)} \\ n & \text{(cyklus)} \\ \text{NIL} & \text{(keď sa proces zastaví)} \\ n+1 & \text{(implicitné riadenie)} \end{cases}$$

Jednoduchosť riadiaceho algoritmu má za následok nevyhnutnosť domysliť do všetkých podrobností situácie, do ktorých sa spracúvanie údajov môže dostať, a spôsoby, ako sa v nich má výpočtový proces správať. Vedie to k potrebe tvorby **RIADIACEJ INFRAŠTRUKTÚRY VÝPOČTOVÝCH PROCESOV**. Zodpovedajúce príkazy sú spravidla roztrúsené v tele programov, čo spôsobuje, že riadiaca infraštruktúra je často neprehľadná a ťažko zrozumiteľná dokonca aj samotnému autorovi procedurálneho programu.

Potreba zmrny spôsobu programovania

Expertné systémy sa vytvárajú na riešenie problémov, *pre ktoré nie sú vopred dané deterministické algoritmy*. Nimi realizované riešiacie postupy sú založené iba NA SITUAČNE UPLATŇOVANÝCH PREDPISOCH SPOČÍVAJÚCICH NA POZNATKOCH umožňujúcich *vyhľadávať revokovateľné postupnosti využiteľných riešiacich krokov potenciálne vedúcich k želanému výsledku*. A preto programová realizácia zodpovedajúcich procesov vyvoláva potrebu odlišného štýlu programovania v porovnaní s klasickým procedurálnym štýlom:

Deklaratívny štýl programovania

Tvorbu vhodne organizovaných a štrukturovaných zoskupení symbolových konštruktov reprezentujúcich zákonitosti (pravidlá, vzťahy, väzby, ohraničenia, predpisy, výnimky atď.) danej problémovej oblasti.

Riešiteľnosť

A práve v tom spočíva princíp **deklaratívneho (situačného) štýlu programovania**. **Zdôrazňujú sa nim zákonitosti riešenia jednotlivých druhov problémov, pričom sa do značnej miery ich symbolové vyjadrenie (reprezentácia) separuje od vlastných (univerzálnych) riešiacich procesov.**¹¹

Pri deklaratívnom programovaní **riešiteľnosť problému nezávisí od žiadnej konkrétnej riadiacej stratégie**. **Riešiteľnosť** daného problému je **závislá** iba od (vzájomne nezávislých alebo len málo závislých) **symbolových výrazov reprezentujúcich zákonitosti, ktorým podlieha riešenie problému**.

Vznik jednotlivých situácií sa nemusí predvídať, ale

Kvalita deklaratívneho programu je podmienená kvalitou

- **analýzy predvídateľných situácií (stavov) vyskytujúcich sa v danej triede riešených problémov,**
- **zodpovedajúcich akcií, ktoré sú im priradené, a**
- **symbolovej reprezentácie takých situačno-akčných prvkov.**

Kvalita deklaratívnych programov je len málo, ak vôbec, závislá od vlastných **výpočtových procesov**.

To, či a kedy jednotlivé situácie vzniknú sa nemusí v nich predvídať.

Jednotlivé situácie sú určené zosstavami

- **známych údajov** – tie môžu byť *dodané* (používateľom, prostredím), *získané* (dotazovaním používateľa alebo vonkajšieho prostredia), *odvodené* (použitou metódou odvodzovania, vypočítané, dedené), *predpokladané* a ešte nespochybné (pokiaľ inými dostupnými prostriedkami sa v danom stave nedajú zistiť),
- **chýbajúcich a nedostupných údajov,**
- **údajov korešpondujúcich s obsahom infraštruktúry riešiacich procesov** – zodpovedajú prebiehajúcim stavom riešeného problému.

Zostavy týchto údajov charakterizujú riešený problém aj jeho priebeh.

Kedže zbiehanie riešiaceho programu je podmieňované výskytom situácií a tie sú určované údajmi, hovoríme o situačnom alebo o údajmi riadenom vykonávaní programu.

Pri deklaratívnom programovaní teda stačí, keď sa vhodne reprezentujú

s ich možným výskytom sa musí rátať

¹¹ Univerzálne riešiacie procesy majú nezanedbateľné limity. Ich prekonávanie postupov špecifickými generickými makro-operáciami je predmetom výkladu mnohých ďalších častí týchto textov.

Interpretátor

zákonitosti problémovej oblasti. Následne, pri riešení problémov sa nemusí formulovať ako sa má postupovať, stačí uviesť ČO sa má riešiť.

Postup riešenia je zabezpečovaný nevyhnutným komplementom - **AKO** komponentom – deklaratívnych programov, ktorý sa nazýva **INTERPRETÁTOR**. *Ten je vo svojej podstate vlastne klasickým procedurálnym (a univerzálnym) programom, ktorý koordinuje procesy zabezpečujúce dosahovanie požadovaných cieľov.* Tým, že stelesňuje všeobecné (univerzálne) čiastkové riešiacie procedúry,

automaticky – na základe interpretovania symbolovo vyjadrených zákonitostí problémovej oblasti a z nich plynúcich dôsledkov - vyvoláva a koordinuje ich činnosť a tak zabezpečuje realizáciu procesov potrebných pre riešenie problému.

Vďaka deklaratívnemu programu nie je potrebné vopred deterministicky predpisovať jednotlivé akcie (operácie) ani ich postupnosť. Interpretátor napr. na základe *zisteného rozdielu medzi želaným a skutočným stavom riešenia problému*, vyhľadá, vyberie a zabezpečí vykonanie operácií vtedy, keď ním detekovaná situácia akciu vyžaduje alebo umožňuje.

Zbiehanie deklaratívneho programu

Programová realizácia interpretátora je prostriedkom, ktorý umožňuje vytváranie deklaratívnych programov. Odbremeňuje tvorcu takých programov od nevyhnutnosti predvídať postupnosť vykonávania jednotlivých krokov deklaratívneho programu, teda starosť o riadenie jeho zbiehania a vytvárania zodpovedajúcej infraštruktúry.

Vyhľadávanie a vyberanie riešiacich operácií (riešiacich krokov) je **nedeterministický proces**. Keďže iba niektoré ich postupnosti vedú potenciálne k požadovanému výsledku, nie je však vopred známe ktoré, je nevyhnutné zabezpečiť možnosť od nich **odstúpiť, tým spravidla aj revokovať (odvolať, zrušiť) dovtedy vykonané akcie a následne voliť iný postup.**

Zbiehanie deklaratívneho programu je riadené (vo všeobecnosti) netriviálnou procedúrou. Na jej priblíženie použijeme túto symboliku:

- s_k – *stav (stúcia) riešenia problému*,
- g - *zobrazenie stavu na množinu aplikovateľných (prípustných) operácií $\{o_k\}$,¹² t.j. situačno-akčných pravidiel¹³,*
- f – *funkcia výberu vhodného operátora* (pravidla, povelu, inštrukcie) - ktorá vo všeobecnosti môže byť pomerne netriviálna,
- n – *poradové číslo vybratého operátora*, ktoré pri ukončení činnosti je nahradené príslušným symbolom **NIL**.

Teda riadiaca procedúra zodpovedá **zobrazeniu a následnej aplikácii výberovej funkcie**. Možno to vyjadriť v nasledujúcej podobe

Realizácia výberovej funkcie

$$f(g(s_k)) = f(\{o_k\}) = \begin{cases} n \\ \text{NIL} \end{cases}$$

12 Množina aplikovateľných operácií implikuje vznik nedeterminizmu !

13 V stúciach, pre ktoré absentujú potrebné poznatky, táto množina môže byť aj prázdna. Taký prípad vyžaduje prostriedok ošetrenia vzniknutého stavu.

Výberová funkcia f, v závislosti na dômyselnosti jej realizácie, môže na jednej strane zodpovedať triviálnym, na druhej strane až veľmi zložitým a sofistikovaným, **heuristikami** podmieneným, procesom. Nasledujú niektoré možnosti:

- mechanický výber najmenšieho, či najväčšieho **n**,
- náhodilý výber operácie (akcie),
- výber operácie (akcie), ktorá
 - ✓ bola posledne prerušená, teda nedokončená,
 - ✓ je buď zo všetkých prípustných najvšeobecnejšia alebo najšpecializovanejšia,
 - ✓ posledne vykonanú najlepšie dopĺňa, alebo je práve jej opakom,
 - ✓ sa posledne vykonanej a iba čiastočne úspešnej najviac podobá,
 - ✓ má pre uskutočnenie k dispozícii najviac údajov alebo najlepšie údaje (spoľahlivosť, presnosť, kategorickosť, špecifickosť, senzitivnosť, diskriminačná účinnosť a pod.),
- výber operácie, ktorá
 - ✓ je buď najčastejšie alebo najzriedkavejšie používaná,
 - ✓ spôsobí aktiváciu buď najväčšieho alebo najmenšieho počtu naviazujúcich akcií,
 - ✓ spôsobí aktiváciu najlacnejších (napr. v zmysle výpočtovej zložitosti, nárokov na doplnenie chýbajúcich údajov a pod.) naviazujúcich akcií,
 - ✓ má potenciál získať najviac nových alebo najdiferencujúcejších informácií,
 - ✓ sa v analogickej situácii najčastejšie osvedčovala,
 - ✓ vzhľadom na dané kritéria, pokiaľ ich splňuje, zabezpečuje najrýchlejšie dosiahnutie cieľového stavu,

a ďalšie.

Realizácia výberovej funkcie má zabezpečiť buď jednoznačný výber jedinej z uvedených alternatív, alebo výber jedinej **makrooperácie** tvorenej vhodnou deterministickou kombináciou zodpovedajúcich elementárnych operácií. V prvom prípade ide o pomerne dobre zvládnuté procesy elementárnej úrovne, t.j. takej, ktorá má *mechanickú povahu*. V druhom prípade ide o zatiaľ nedostatočne preskúmané a spoznané riešiacie procesy vyššej úrovne, t.j. také čo stelesňujú analógie kognitívnych procesov odborníkov.

Hoci samotný proces výberu operácií by mal byť určitou aproximáciou kognitívne efektívneho algoritmu, vzhľadom na absentujúci cieleňý návod expertnými systémami riešených problémov, je taký **proces nedeterministický, iba nezáväzne skúma aplikovateľnosť a úspešnosť vybraných operácií!**

Nedeterminizmus, ako charakteristický rys problémov riešených ES, spravidla implikuje mnoho riešiacich krokov, v ktorých nastáva aspoň jedna z nasledujúcich situácií

- **z formulácie problému ani z predchádzajúceho priebehu jeho riešenia nie je zrejmé, ktorý krok sa má v ďalšom vykonať,**
- **výber ďalšieho postupu riešenia je podmienený výsledkom testovania podmienky, o ktorej však nie je možné s určitosťou rozhodnúť, či je alebo nie je splnená.**

Pri alternatívnom prekleňovaní nedeterminizmov sa uplatňujú **heuristické pravidlá**. Sú to akési nezdôvodnené, skúsenosťou nadobúdané návody na rozhodovanie v nedeterministických situáciách riešiaceho postupu. Hoci sú často

Nedeter-
minizmus

Heuristické
pravidlá

úspešné, predsa len nie sú spoľahlivou zárukou nájdenia správneho postupu, môžu viesť aj k nezdaru. Ťažkosť spočíva v tom, že

heuristika neumožňuje vopred rozpoznať, či jej použitie je vhodné; nie je vyjadriteľná v tvare kritéria použiteľnosti,
preto nie je návodom k **deterministickému** riešiacemu postupu.

1.5 Paradigma deklaratívneho programovania a produktívne riešiace postupy

Na základe doterajších úvah sa o aplikovateľnosti procedurálneho, resp. deklaratívneho programovania, dá vysloviť nasledovné

MUSÍ a MÔŽE

Keď zákonitosti platné pre riešenie problému umožňujú hovoriť o tom, ČO A KEDY MUSÍ nastat', vtedy je výhodný štýl PROCEDURÁLNEHO PROGRAMOVANIA – vtedy dokážeme k daným problémom vytvoriť jednoznačný a efektívny riešiaci proces.

Keď však zákonitosti platné pre riešenie problému poznáme len natoľko, že dokážeme iba predvídať ČO A KEDY MÔŽE nastat', vtedy je potrebné vytvárať také riešiace prostriedky, ktoré umožňujú potenciálne odvodenie postupu riešenia. A to je to prípad, keď je potrebné orientovať pozornosť na deklaratívne programovanie.

Paradigma deklaratívneho programovania

Paradigma deklaratívneho programovania kladie dôraz na zákonitosti (pravidelnosti), ktorým podliehajú problémy danej aplikačnej oblasti, zákonitosti vyjadruje prostriedkami formalizmu, ktoré sú propozičnými prejavmi znalostí, a tie považuje za nezávislé od riešiacich procesov, v ktorých sa uplatňujú.

- **Deklaratívny program je tvorený identifikovateľnými štruktúrovanými prvkami, ktoré explicitne symbolovo reprezentujú deklarácie zákonitostí platné pre určitú triedu problémov.**
- **Postup jeho zbiehania (riešenia problému) vyplýva z dôsledkov špecifikovaných (logických) formúl reprezentujúcich zodpovedajúce zákonitosti.**
- **Tento postup sa generuje na základe riadenej - formálnej logike neodporujúcej inferencie, t.j. interpretácie príslušných formúl - vhodným (dodaným, vytvoreným) interpretátorom.**

Tvorba deklaratívneho programu podlieha

- **syntaktickým pravidlám zvoleného reprezentačného formalizmu a**
- **korešpondujúcim princípom interpretácie formúl a riadeného priebehu interpretácie.**

Odčlenenie riešiacich procesov - a teda súvisiacich riadiacich konštruktov - od vlastného deklaratívneho programu výrazne uľahčuje

- tvorbu jednotlivých prvkov programu,
- porozumenie ich úlohám a
- účinkom ich použitia.

Deklaratívny program a jeho interpretátor vo "vlastnej réžii" zabezpečujú odvodenie riešiaceho postupu a generujú ho.

Súhrn.

Zhrnutie

Produktívne riešiace postupy

- prehľadávanie ✓ vyžadujú prehľadávanie, odhadovanie, voľbu, skúšanie, či inak nedeterministické vyberanie niektorej z alternatívnych možností riešenia problému,
- nezáväznosť ✓ sú pri výbere flexibilné a na výber sa nefixujú: ak sa v nadväzujúcom riešení postupe
 - narazí na protirečenie či konflikt,
 - zistí, že nevedie k požadovanému výsledku,
 - odhalí možnosť iného, výhodnejšieho postupu,
 tak sa od časti alebo celého prebiehajúceho postupu odstúpi, revidujú sa vybraté (odhadnuté, zvolené, skusmé, či inak podmienené, alebo kontextuálne závislé) operácie/údaje a pokračuje sa iným, vzhlľadom na dosiahnutý stav vhodnejším variantným postupom,
- vratnosť ✓ sú vratné – čo ich principiálne odlišuje od nevratných *reproduktívnych* postupov uskutočňovaných procedurálnymi programami,
 ✓ *nemajú metriku*, čo znemožňuje hodnotenie vzdialenosti od požadovaného výsledku alebo zisťovanie odchýlky od správneho "smeru" postupu k riešeniu - *súvisí to s absentujúcou možnosťou analytického opisu riešacieho postupu, a teda absenciou algoritmu, ktorý by jednoznačne viedol k správne výsledku.*

1.6 Princípy prehľadávacích procesov

Vnímanie
riešiacich
procesov

Prehľadávacie procesy sa dajú percipovať z rôznych pohľadov. Zdôrazňované aspekty najčastejšie hovoria o (riešiacich) postupoch v terminológii:

- (a) **prehľadávania stavového priestoru problému,**
- (b) **procesov splňovania ohraničujúcich podmienok,**
- (c) **logickej inferencie.**

Ide o vzájomne sa nevylučujúce hľadiská, ktoré sústreďujú pozornosť na líšiace sa prístupy k reprezentovaniu a riešeniu problémov, pričom každý z nich má svoje oprávnenie aj poslanie. Zo syntaktických hľadísk sú vo všeobecnosti vzájomne zameniteľné.

Stavový
priestor

Predstava **stavového priestoru problému** vedie k úvahám o **grafe**, ktorého **vrcholy** zodpovedajú **stavom** riešenia problému a **hrany** zodpovedajú **operáciám** meniacim stav na iný: *produktívne riešenie problému má potom povahu prehľadávania možných ciest medzi vrcholom zodpovedajúcim počiatočnému stavu riešenia problému a niektorým z vrcholov zodpovedajúcich podmienkám cieľového stavu.*

Splňovanie
ohraničení

Predstava **procesov splňovania ohraničení (ohraničujúcich podmienok)** vníma riešenie problému ako **postupné splňovanie explicitne zadaných ohraničujúcich podmienok**: *V riešení postupe sa postulovaná (predpokladaná) počiatočná množina objektov, o ktorých sa predpokladá, že môžu vyhovovať daným ohraničeniam, postupne zužuje v závislosti na súlade (konzistentnosti) ich vlastností s danými a odvoditeľnými ohraničeniami¹⁴ až potenciálne k množine,*

¹⁴ Odvoditeľné ohraničenia explicitne alebo implicitne vyplývajú z reprezentovaných zákonitostí problémovej oblasti.

obsahujúcej objekt(y) najlepšie vyhovujúci(e) všetkým ohraničujúcim podmienkám - čo je cieľový stav.

Logická inferencia

Predstava **logickej inferencie** vníma riešenie problému ako logické rezolvovanie: *Predpokladá (postuluje) určitý logický systém, ktorý umožňuje z daných axióm a už odvodených tvrdení odvodzovať ďalšie. Postup riešenia spočíva v prehľadávaní a vyberaní použiteľných axióm a tvrdení a z vybratých logickou inferenciou produkuje nové tvrdenia, kým sa nenájde odpoveď na otázku zodpovedajúcu riešenému problému – cieľový stav.*

1.7 Formálna logika a jej postavenie pri tvorbe ES

Nezastupiteľnosť formálnej logiky

Interpretácia deklaratívneho programu súvisí s realizáciou univerzálnych procesov uplatňovania reprezentovaných poznatkov. Zodpovedajúci proces musí spočívať (byť "ukotvený") na pevných teoreticky podložených princípoch. A v tom majú formálne logiky nezastupiteľné postavenie. Súvisiacej problematike je venovaný tento článok.

Z pohľadu formálnej logiky sústava znalostí **Z** (axiómy a teorémy) reprezentovaná príslušnými symbolovými prostriedkami má v konkrétnych prípadoch umožniť nájdenie všetkých pravdivých viet (klauzúl), ktoré z nej vyplývajú. To znamená, že keď sa v rámci **Z** hľadá odpoveď na konkrétnu otázku **Q**, je nevyhnutné, aby sa dalo zistiť, či **Z** umožňuje túto odpoveď určiť (odvodiť). V symbolovom vyjadrení píšeme

$$\mathbf{Z} \Rightarrow \mathbf{Q},$$

kde \Rightarrow je symbolom pre 'určuje' (odpoveď na otázku **Q**).

Na rozdiel od bežných databáz obsahujúcich iba konkrétne údaje, zisťovanie (odvodzovanie) odpovede na otázku **Q** vo všeobecnosti vyžaduje navyše k prehľadávaniu sústavy explicitných poznatkov v **Z** hľadať aj iba **implicitne** obsiahnuté odpovede. Preto k **Z** musia jestvovať inferenčné pravidlá odvodzovania hľadanej odpovede. Tie musia umožniť napríklad na základe pravidiel typu

Inferenčné pravidlá

$$\forall \mathbf{x}(\mathbf{a}(\mathbf{x}) \rightarrow \mathbf{c}(\mathbf{x})) \tag{1.1}$$

na základe platnosti predikátu **a(K)** - t.j. z pravdivosti tvrdenia '**K** má vlastnosť **a**' - odvodiť zo **Z** pre individuovú konštantu **K** odpoveď na otázku **Q**, teda platnosť **c(K)**, čo znamená, že **K** má aj vlastnosť **c**. Symbolovo píšeme

$$\mathbf{a}(\mathbf{K}) \rightarrow \mathbf{c}(\mathbf{K}) \tag{1.2}$$

kde \rightarrow je symbolom pre *odvoditeľné*. Odvodenie odpovede **c(K)** zabezpečujú pravidlá inferencie aplikované na pravidla typu (1) obsiahnuté v **Z** a na *fakt (poznatok) a(K)*.

Kým vzťah (1.1) je **deklarovaním určitého poznatku** z danej problémovej oblasti, vzťah (1.2) vyjadruje **procedurálnu stránku použitia daného poznatku**.

Logika vytvára základňu - **syntaktických** - mechanizmov algoritmického exhaustívneho (zo sémantického hľadiska nedeterministického) **prehľadávania**

Logika	<p>všetkých dôsledkov, ktoré vyplývajú z množiny splnených predpokladov. Jej pravidlá sú permisívne, určujú dôsledky priamo odvoditeľné z množiny predpokladov, umožňujú usudzovať o možnostiach, ktoré však nemusia nastať alebo nie sú plauzibilné (vierohodné) ani relevantné (významné), neurčujú však tie dôsledky, ktoré sa musia odvodiť, ani poradie ich odvodzovania.</p>
jej pravidlá sú permisívne	<p>Táto skutočnosť má veľmi vážne implikácie. Dá sa to ilustrovať na dokazovaní teorém: Jazyk prvorádovej predikátovej logiky (v skratenom zápise JPL) vedie k problematike nachádzania účinných procedúr dokazovania teorém (substituovanie, unifikácia a rezolvovanie) na základe daných axióm a daných či odvodených teorém. Neredukuje exhaustívne prehľadávacie procesy na taký zvládnuteľný rozsah, ktorý by adekvátne a účinne imitoval efektívne formulovanie problému (otázku, ktorá by sa mala zodpovedať) ani riešiace postupy skúseného odborníka. Naviac dokazovanie teorém má exponenciálnu výpočtovú zložitosť. Hoci sa dôsledky toho prejavujú iba v pesimálnych prípadoch, skutočnú výpočtovú zložitosť riešenia konkrétneho problému, ani jej priemerný odhad, nevieme predvídať. Musí sa teda rátať aj s najhoršou alternatívou, čo z praktických hľadísk je neprijateľné</p>
Výpočtová zložitosť	<p>Všimnime si ďalšie ťažkosti.</p> <p>V priebehu riešenia konkrétnych problémov môže sa meniť pravdivosť tvrdení o realite (napr. kvôli dynamickým zmenám v nej). Preto je nevyhnutné sústavne sledovať bezospornosť všetkých daných a odvodených tvrdení a podľa potreby ich aktualizáciou zabezpečovať konzistentnosť riešiaceho postupu. Avšak mechanizmy logickej inferencie sú kumulatívne (monotónne): pri odvodzovaní nových tvrdení všetky v predošlom odvodené zostávajú v platnosti, nie sú aktualizované. Reálne systémy usudzovania nie sú také, môžu porušovať pravdivosť už odvodených výrokov - sú nemonotónne.</p>
Problémy s monotónnosťou	<p>Pri riešení reálnych problémov, v počiatočom stave riešenia spravidla nie sú k dispozícii všetky potenciálne použiteľné údaje. Tie sa v priebehu riešenia postupne, podľa potreby, získavajú, zhromažďujú. Príliv nových údajov môže viesť k situácii, ktorá vyžaduje (dramatickú) preformuláciu pôvodného problému a tým riešenie úplne nového problému. Zodpovedajúce procesy preformulovania nie sú založené na formalizovateľných postupoch, majú predovšetkým empirickú povahu. Aj to svedčí proti názorom zužujúcim a stotožňujúcim uvažovanie len na mechanizmy logickej dedukcie.</p>
Lokálne versus globálne	<p>Prostriedky inferencie v JPL vyberajú niektoré z možných alternatívnych riešiacich postupov len vzhľadom na riešenie práve aktuálnej čiastkovej úlohy nezávisle od riešenia ostatných podproblémov, či celkového cieľa. Realizujú teda len tzv. lokálne riešiace procesy. Absentuje v nich možnosť uskutočniť globálne riešiace postupy (makrokroky), t.j. spôsobilosť z určitého nadhľadu syntetizovať čiastkové úlohy. Napríklad využívať určitý plánovací mechanizmus analyzujúci stavový priestor problému na výber akcie, ktorý umožňuje súčasne rozhodovať medzi väčším počtom alternatívnych postupov (výbere hypotéz).</p>
Fuzzy kvantifikátory	<p>Riešenie praktických problémov nezriedka vedie k potrebe reprezentovať, sémanticky interpretovať a procedurálne využívať aj iné ako bežné logické kvantifikátory, napríklad NAJMENEJ, PRÁVE, NAJVIAC, MÁLO, MNOHO, SPRAVIDLA, ČASTO, ZRIEDKA a mnoho iných. Podobne je často žiaduce reprezentovať nemálo výnimiek (napr. "Každý (?) vták lieta.", "Každý (?) kov je kujný."). Prostriedky JPL neposkytujú k tomu priamočiare možnosti ani na</p>

deklaratívnej ani na procedurálnej úrovni. (Kým prvé tri z uvedených kvantifikátorov sú jednoznačné a dajú sa vyjadriť prostredníctvom univerzálneho a existenčného kvantifikátora, ostatné sú problematickejšie, lebo, okrem iného, sú zvyčajne kontextovo závislé).

Psychológia tvrdí (nie sú známe protiargumenty), že: **inferenčné pravidlá JPL nie sú totožné s prirodzeným (spontánnym) usudzovaním človeka**. Ten si vytvára, osvojuje, preberá systémy pravidiel a mechanizmov ich používania, ktoré sa odlišujú od formálnych pravidiel logiky, napr. jednoduchý princíp substituovania pravdivostných hodnôt do premís (napr. *hradlovanie, komplementovanie, asociovanie*, ako o tom pojednáme v ďalšom) a sledovanie dôsledkov, ktoré z toho vyplývajú.

JPL umožňuje pri reprezentácii znalostí vynechávať množstvo špecifikácií skutočností, ktoré spôsobujú neúplnosť reprezentácie. **Tým sa prednosť JPL, ktorá spočíva v jeho všeobecnosti, stáva zároveň jeho nedostatkom**. Východisko sa hľadá a nachádza v používaní **jednoduchšieho, či slabšieho formalizmu, resp. jeho obohatenie mimologickými konštruktami**.

Potreba
mimologických
konštruktov

Schodnosť tohto postupu vyplýva z existencie viacerých prostriedkov na tvorbu znalostných (expertných) systémov. Každý z nich je výsledkom snahy hľadať **kompromis medzi požiadavkami logickej koherentnosti a prijateľnej výpočtovej zložitosti, ako aj medzi vlastnosťami symbolickej (syntaktickej) a mentálnej (sémantickej) logiky**.

Hľadajú a nachádzajú sa neštandardné inferenčné procesy (a prirodzene zodpovedajúce formalizmy reprezentácie znalostí), ktoré dopĺňujú štandardné vhodnými riešiacimi schémami pre **situačné uprednostňované a selektívne odvodzovanie iba cielenej podmnožiny logických dôsledkov**. Prostriedkom toho sú napríklad

- ✓ **neklasické logiky** (modálne, viachodnotové, t.j. stupňovanie "pravdivosti", resp. "nepravdivosti" až po spojité (prípadne nelineárne) kontinuum medzi pravdou a nepravdou - fuzzy logika),
- ✓ **nezáväzné používanie očakávatel'ných (predpokldatel'ných) údajov** nahradzujúcich chýbajúce,
- ✓ **udržovanie konzistentnosti** odvodených faktov - **nemonotónna logická inferencia** (odvodzovanie),
- ✓ **oceňovanie "nákladov"** za vyhodnocovanie podmienok (získavanie údajov) a vykonanie operácií (ich informačný prínos),
- ✓ **sémantické odlíšenie** poslania operácií (pravidiel), či častí stavového priestoru riešenia problému (napr. rozlišovanie hlavných a vedľajších inferenčných schém, rozlišovanie fenomenologických a kauzálnych závislostí, rešpektovanie ohraničení a pod.),
- ✓ **špecifikovanie, generovanie a používanie podporných riadiacich údajových štruktúr a čiastkových (generických makro-) procesov** umožňujúcich obohacovať prostriedky odvodzovania, ktoré vyplývajú z vlastností JPL, a iné.

Vo všeobecnosti jestvujú dve **alternatívy používania mimologických konštruktov** a možnosť ich kombinovania:

(1) rozšírenie systému logických klauzúl o riadiace údajové štruktúry, ktoré umožňujú explicitne vyjadrovať situačne podmienené riadiace akcie a

(2) *rozšírenie funkčných spôsobilostí interpretátora logických formúl o situačne riadené spúšťanie procedurálnych riešiacich mechanizmov.*

1.7 Poznatky a znalosti

Na záver tejto uvádzajúcej kapitoly venujme pozornosť ešte významu, v ktorom používame kľúčové pojmy súvisiace s fenoménom (nazvaným) expertný systém: **poznatok** a **znalosť**. Sú to pojmy, ktorých význam sa nedá jednoducho definovať, najmä vtedy nie, keby sa mali zohľadniť všetky súvisiace aspekty. Pre potreby výkladu v nadväzujúcich kapitolach (*a kvôli vytvoreniu neprázdneho prieniku prvkov 'mentálneho modelu' čitateľa a autora*) uvádzame vymedzenia, ktoré sú nepochybne primeranou aproximáciou týchto fenoménov, pričom ich považujeme za otvorené, teda nenárokujúce si definitívnosť.

Poznatky

Poznatky sú reprodukciou určitej vymedzenej časti objektívneho sveta a v ňom platných zákonitostí.

Je pre ne charakteristické, že

sú transformáciou rozptýlených (aj nejasných) predstáv a tušení do takej komunikateľnej formy, v ktorej sa z nich zachováva to, čo možno oznámiť iným ako ustálený základ racionálneho konania.

Vznikajú ako **produkt myšlienkovvej činnosti najmä v súvislosti s pracovnými a spoločenskými aktivitami.**

Znalosti

Znalosti

sú meniteľné a doplniteľné štruktúry vzájomne previazaných poznatkov, ktoré sú podmieňované ich uplatniteľnosťou v interakcii so svetom:

- ✓ **znalosť čohosi znamená vlastniť tomu zodpovedajúcu reprezentáciu v podobe dostatočne verného a presného kognitívneho (mentálneho) modelu, vrátane spôsobilosti vykonávať s tým, čo je reprezentované rôzne kognitívne (mentálne) operácie,**
- ✓ **na základe a v rozsahu týchto operácií dokáže človek (potenciálne aj počítač) predvídať a predpovedať to, čo sa musí alebo môže v reálnom svete odohrať**

súvisia s konceptualizáciou entít (objektov, javov, ich vlastností) a v nich, či s nimi prebiehajúcich procesov:

- ✓ **konceptualizácia spočíva - okrem iného - v schopnosti pomenúvať, opisovať a definovať entity a ich vzájomné vzťahy, ako aj kategorizovať ich, t.j. zaraďovať ich do rôznych tried, medzi ktorými sú špecifikovateľné vzájomné vzťahy**
- ✓ **súčasťou konceptualizácie je aj formovanie výrokov o entitách, skladanie a vyhodnocovanie (interpretovanie) výrokov a v tom spočívajúce odvodzovanie dôsledkov, ktoré z nich vyplývajú.**

Deskripcie
(opisy)

Termín znalosť zahrňuje a kombinuje

- **deskripcie (opisy, definície) identifikujúce a rozlišujúce entity aj ich triedy** - sú vyjadrované **vetami určitého jazyka**, ktorých prvky sú tvorené *primi-*

Relácie
(vzťahy)

tívnymi konceptami,

- **relácie**, ktoré vyjadrujú **vzťahy** (závislosti, podmienenosti) entít - zodpovedajú **definičným, asociatívnym (empirickým), príčinným (kauzálnym), taxonomickým, štrukturálnym, kontextovým, priestorovým, časovým, funkčným, úlohovým, ohraničujúcim, modelovým a riadiacim/regulačným** väzbám (súvislostiam),

Procedúry
(predpisy)

- **procedúry (preskripcie)**, ktoré **postupnosťami operácií** v priebehu riešenia problémov (situačne) **vymedzujú používanie a interpretovanie deskripcií a relácií**

Hoci pre bežné odborné aj laické riešenie problémov táto abstraktná kategorizácia nebýva prínosom, v prípade ich komunikovania (výklad, článok, učebnica, symbolová reprezentácia atď.) ich utriedenie, štrukturovanie, zdôvodňovanie a pod. sa stáva užitočným až nevyhnutným. A je tomu tak aj pri reprezentácii znalostí v počítači.

V snahe obísť prípadné nejasnosti, neurčitosti, nedorozumenia v nadväzujúcich textoch, uvádzame význam, v ktorom používame niektoré z vyššie uvedených pojmov:

- **Definícia** - presné a úplné vymedzenie/určenie objektov, javov, procesov.
- **Asociácia** - obvyklá, pozorovateľná, či pravdepodobná, nie však nevyhnutná a zdôvodnene(niteľ)ý vzťah(y), spojenie(a) medzi dvomi alebo viacerými entitami (napr. ich súčasný alebo následný výskyt).
- **Kauzalita (príčinnosť)** - známe zdôvodniteľné súvislosti medzi dvomi alebo viacerými entitami, ktoré sú vo vzťahu príčiny a následku.
- **Taxonómia** - zodpovedá usporiadaným vzťahom typu všeobecnejší-špeciálnejší (generalizačno-špecializačné vzťahy), ale aj nadriadený-podriadený, predchádzajúci-následnícky (napr. vývoj v čase) umožňujúce organizovanie/usporiadanie entít.
- **Štruktúra** - vzájomný vzťah prvkov celku, t.j. ich zloženie a organizácia, ako aj vzťah prvkov k celku.
- **Kontext** - okolnosti, súvislosti, za ktorých sa uplatňujú určité špecifické vzťahy medzi entitami podmieňujúce ich líšiace sa postavenie či úlohy, teda aj rôzne prípustné významy (pori tiež *úlohy*).
- **Priestorové vzťahy** - v tomto prípade sa spravidla jedná o pozičné, rozmerové a smerové vzťahy medzi entitami (susedstvo, vnútro-vonkajšok, veľkosť, vzájomné vzdialenosti/poloha/orientácia v priestore, tvar, smer pohybu atď.).
- **Časové (temporálne) vzťahy** - spravidla sa uvažujú súvislosti medzi entitami, ktoré vzhľadom na ich výskyt alebo zmeny v určitých časových okamihoch/intervaloch podmieňujú okamžité, súčasné a časovo následné výskyt, či zmeny iných entít - umožňujú uvažovať o javoch minulých, súčasných a budúcich.
- **Funkčné vzťahy** - správanie, poslanie, či použitie entít.
- **Úlohy** - známe či predpokladané polymorfizmy (polyfunkčnosti) entít (tá ista entita môže mať rôzne (situačne podmienené) úlohy a poslanie s odlišnými kvalitami, vlastnosťami - (pozri tiež *kontext*)).
- **Ohraničenia** - predpisy a zákony (spoločenské a prírodné, vrátane vzťahov k fyzikálnym, chemickým, biologickým, ekonomickým konštantám) vzťahujúce sa k entitám a ich vzájomným funkčným, štrukturálnym, ekonomickým, sociálnym a iným väzbám (podmienenostiam).

- **Modelové väzby** - známe či predpokladané dynamické súvislosti určitým spôsobom vzájomne previazaných a ovplyvňujúcich sa entít, ktoré reprezentujú štruktúru a správanie sa modelovanej predlohy.
- **Regulačné/riadiace vzťahy** - špecifické závislosti, na základe ktorých je možné ovplyvňovať správanie sa entít.

Termy a asercie

Uvedené typy poznatkov a znalostí môžu byť v bázach znalostí - v závislosti na konceptualizácii reprezentácie uvažovaného výseku sveta – môžu vystupovať ako prvok deskripcií, teda v roli **termu**, alebo ako prvok preskripcií, teda v roli **asercie**. V súlade s uvažovanou rolou príslušného kvanta znalostí je potrebné voliť odlišný formalizmus symbolovej reprezentácie. Sú témou nasledujúcich kapitol.

term

Túto úvodnú kapitolu, po prečítaní ktorej predpokladáme už existenciu aspoň hrubého a neúplného mentálneho modelu ES v mysli čitateľa, uzatvárame vymedzením dvoch vyššie uvedených pojmov rozlišujúcich poznatky a znalostí z hľadiska ich postavenia v znalostných systémoch: **term** a **asercia**.

asercia

Termy zodpovedajú opisom/definovaniu entít, opisom/definovaniu ich vlastností a prislúchajúcich (skutočných či predpokladaných) hodnôt, ich (kontextuálne podmieneným) úlohám, zaradeniu a vzťahom, ich štruktúram a prípustnej organizácii (vzťahy časť-celok), hierarchickým klasifikáciam (taxonomizácii), t.j. účinnému vyjadreniu ich generalizačno-špecializačným, (podmieneným) príčinnno-dôsledkovým závislostiam a úlohám, čo všetko možno s výhodou situačne použiť pri efektívnom využívaní asertívnych znalostí a umožňuje odvodzovať konkrétne implicitné poznatky a fakty.

Asercie sú tvrdenia o entitách, o ich konkrétnych stavoch, vzťahoch a vlastnostiach. Zodpovedajú propozíciam, ktoré umožňujú explicitne odvodzovať konkrétne poznatky, t.j. skutočnosti súvisiace s riešeným problémom. V deklaratívnej podobe majú najčastejšie tvar izolovaných produkčných ("AK-TAK", "KEĎ-POTOM") pravidiel reprezentujúcich vzťahy a súvislosti medzi entitami. Môžu však mať aj tvar analytických alebo iných matematických závislostí reprezentujúcich ohraničenia, funkcie a (v znalostných systémoch najmä *kvalitatívne*) modely. Dajú sa tiež považovať buď za jedinečné preskripcie, alebo také čo parametrizujú všeobecné procedúry odvodzovania (najčastejší prípad).

2. SYMBOLOVÁ REPREZENTÁCIA POZNATKOV A ZNALOSTÍ

2.1 Prostriedky symbolovej reprezentácie

Analógie medzi myslením a strojovým odvodzovaním na úrovni operácii so symbolmi.

Hypotézy uvedené v predošlej kapitole, najmä koncept *mentálneho modelu*¹⁵, sú východiskom realistického predpokladu o uvažovaní človeka. **Sú to v mysli uskutočňované procesy (operácie, manipulácie) s entitami mentálneho modelu, teda s mentálnou reprezentáciou týchto entít.** Existencia príslušných entít v mysli/pamäti človeka je nevyhnutnou podmienkou uskutočňovania takýchto procesov a teda myslenia.

Podstata procesov *odvodzovania/usudzovania* uskutočňovaných počítačom je však tiež proces operácií (manipulácií) s entitami v jeho aritmeticko-logickej jednotke. Nevyhnutnou podmienkou ich uskutočňovania je existencia symbolovej reprezentácie príslušných entít v pamäti počítača.

Z tejto evidentnej analógie vyplýva pozornosť, ktorá sa v súvislosti s **UI** a tvorbou **ES**, venuje *symbolovej reprezentácii poznatkov a znalostí*. Menovite

¹⁵ Koncept mentálneho modelu nespochybnuje ani Searle, hoci je jedným z výrazných oponentov fenoménu *strojového myslenia*.

	☞ aspektom, ktoré sú spoločné pre myslenie a spracúvanie symbolov, ☞ spoznávaniu takých procesov vytvárania a spracúvania symbolových štruktúr, ktoré v prípade, keď ich vykonáva človek, nazývame myslením.
Symboly	<p>Symbol je jednoznačne identifikovateľný a manipulovateľný fenomén, ktorý môže označovať (zastupovať, reprezentovať) iné entity. Základným elementárnym (nerozložiteľným) symbolom hovoríme symbolové primitíva. Umožňujú vytváranie zložitejších symbolových štruktúr. A práve v súvislosti s takou tvorbou sa dá hovoriť nielen o procesoch vytvárania, rušenia, spájania, rozčleňovania, rozdeľovania, ale aj vzájomného prepájania, viazania či iného menenia takýchto štruktúr.</p> <p>Z hľadiska symbolovej reprezentácie znalostí je veľmi významné rozoznávať a vo vzájomných súvislostiach uvažovať o vlastnostiach reprezentačných prostriedkov, o ich</p> <ul style="list-style-type: none"> ▪ vyjadrovacej účinnosti (expresivite), t.j. ČO A V AKOM ROZSAHU umožňujú REPREZENTOVAŤ, ▪ odvodzovacej účinnosti (použitelnosti), t.j. ČO A AKO umožňujú ODVO-DZOVAŤ, ▪ výpočtovej účinnosti (efektivitve) t.j. o VÝPOČTOVEJ A PAMÄŤOVEJ ZLOŽITOSTI ODVODZOVANIA. <p>Predmet reprezentácie je to, čo sa v určitom rozsahu má a môže reprezentovať. <i>Ten istý predmet reprezentácie sa dá zastúpiť (reprezentovať) rôznymi symbolovými prostriedkami.</i> Preto je potrebné rozhodnúť čím reprezentovať, t.j. voliť symbol reprezentovaného. Navyiac, dôležitý je aj prostriedok reprezentácie, t.j. prostredníctvom čoho sa reprezentuje tak, aby bolo možné zachytiť všetky - z istých hľadísk podstatné - črty predmetu reprezentácie. To všetko súvisí s expresivitou symbolovej reprezentácie. Zvýraznime, že ide o rozhodovanie</p> <ul style="list-style-type: none"> ▪ čím reprezentovať, t.j. voliť symbol reprezentovaného, ▪ prostredníctvom čoho reprezentovať, t.j. voliť prostriedok reprezentácie tak, aby bolo možné zachytiť všetky - z istých hľadísk podstatné - črty predmetu reprezentácie. <p>Príslušné rozhodnutia ovplyvňujú expresivitu symbolovej reprezentácie. Úlohu expresivity osvetlíme na dvoch príkladoch.</p> <p>V prvom prípade uvažujeme problém identifikovania stavu (diagnózy) určitého technického zariadenia, povedzme televízora. Na riešenie problému je potrebné reprezentovať jeho <i>štruktúru, funkcie a správanie (prejav)</i>. Tým je špecifikované, čo sa má reprezentovať aj rozsah reprezentovaného (nespomíname napr. fyzikálne zákony podmieňujúce funkcie zariadenia). V tabuľke na nasledujúcej strane sú uvedené ilustračné možnosti toho, čím a prostredníctvom čoho reprezentovať. Ďalšia tabuľka uvádza jednoduchší a priezračnejší príklad - je reprezentovanie kvantít - ilustrujúci a ozrejmujúci význam pojmu expresivita.</p> <p>Voľba symbolov (čím) a prostriedkov reprezentácie (prostredníctvom čoho) podmieňuje použitelnosť reprezentanta, teda procesy, ktoré umožňujú produkovať riešiace postupy, t.j. to ČO a AKO sa dá odvodzovať. V tejto súvislosti hovoríme o odvodzovacej účinnosti.</p> <p>Význam pojmu <i>prostriedok reprezentácie</i> ilustrujeme na prípade reprezentácie <i>celočíselných množstiev</i> (druhá z uvedených tabuliek). Tabuľka zreteľne poukazuje na výrazné rozdiely odvodzovacej účinnosti v závislosti na voľbe prostriedku reprezentácie.</p> <p>Slovná reprezentácia je nesporne prostriedkom zaznamenávania a prenosu informácií o množstvách. Keby sa však mala vykonať operácia súčinu medzi výrazmi <i>štyristosedemdesiatdeväť</i> a <i>deväťstosedemdesiatštyri</i>, vznikli by určité ťažkosti. To isté možno tvrdiť o prípade, v ktorom by sa spomínané množstvá mali reprezentovať na báze Morzeovej abecedy. Ani jeden z týchto prostriedkov reprezentácie neposkytuje totiž účinné prostriedky na uskutočňovanie príslušných výpočtových operácií. O málo výhodnejšie by bolo používanie vhodných zoskupení sybolov <i>III...</i>, aspoň z hľadiska operácie súčtu. Binárna reprezentácia množstiev, najmä s ohľadom na rôzne možnosti pravidiel kódovania a nimi podmienených operácií, poskytuje (predovšetkým pre elektronické zariadenia) rozsiahle možnosti produkovať zodpovedajúce výpočtové procesy. Keď sa napr. kvantily 479 a 974 vyjadria v tvare</p>
Účinnosť symbolovej reprezentácie	
ČO, ČÍM A PROSTREDNÍCTVOM ČOHO reprezentovať	
Symbol reprezentovaného a prostriedok reprezentácie	
Expresivita	

IIIOIII a IIIIOIII,

operácia násobenia, ako je to dobre známe, sa vykoná zodpovedajúcimi posuvmi a sčítovaním. V prípade BCD kódu nadobúdajú uvedené čísla podobu

Odvodzovacia účinnosť a čitateľ ľahko zistí pravidlá, na základe ktorých vznikne proces násobenia. O prednostiach dekadického číselnej sústavy (najmä vzhľadom na to, že väčšina ľudstva je dobre "naprogramovaná" na zodpovedajúce operácie, hoci konštruktér zodpovedajúcich elektronických obvodov zrejme sleduje iné hľadiská) sa vari

<p>ČO A V AKOM ROZSAHU SA MÁ REPREZENTOVAŤ</p>	<p>ŠTRUKTÚRA, FUNKCIE A SPRÁVANIE (POZOROVATELNÉ PREJAVY) SYSTÉMU (nie je potrebná reprezentácia fyzikálnych zákonov podmieňujúcich princípy správania)</p>
<p>ČÍM REPREZENTOVAŤ SYMBOLY REPREZENTOVANÉHO</p>	<p>PROSTREDNÍCTVOM ČOHO PROSTRIEDKY REPREZENTÁCIE</p>
<p>fotografie vnútornej konštrukcie a záber obrazovky (a) bloková schéma elektrického zapojenia (konštrukcie) s uvedením náležitých hodnôt fyzikálnych veličín, (b) podrobná schéma elektrického zapojenia s požadovanými hodnotami jednotlivých parametrov</p>	<p>bitmapa zodpovedajúca fotografiám ikonické symboly zaužívaných geometrických útvarov: trojuholníky, štvoruholníky, úsečky, kružnice, špirálovité krivky..., (prípadne aj vo farbe) s priradenými znakmi/symbolmi (čísla, označenia fyzikálnych jednotiek); v pamäti reprezentované napr. kódmi ikonických útvarov alebo bitmapami</p>
<p>slovný opis prvkov systému, ich vzájomných väzieb a funkcií korešpondujúcich so schémou systému formalizovaný (symbolový) prepis schémy či slovného</p>	<p>výrazové prostriedky prirodzeného jazyka podliehajúce zodpovedajúcim gramatickým pravidlám symbolové prostriedky a pravidlá zvoleného</p>

opisu daného zariadenia	formalizmu (výroková/predikátova logika, produkčné pravidlá, rámce)
-------------------------	---

ČO A V AKOM ROZSAHU SA MÁ REPREZENTOVAŤ	CELOČÍSELNÉ MNOŽSTVÁ
ČÍM REPREZENTOVAŤ SYMBOLY REPREZENTOVANÉHO	PROSTREDNÍCTVOM ČOHO PROSTRIEDKY REPREZENTÁCIE
slovné formulácie	prirodzený jazyk
bodkami a čiarkami	Morzeova abeceda
zskupeniami znakov ///...	päťprvkové zskupovanie znakov(### //)
kombináciami symbolov O a I	binárna číselná sústava, alebo BCD kód
arabskými číslicami	dekadická pozičná číselná sústava
arabskými číslicami v kombinácii s písmenami	hexadecimálna pozičná číselná sústava
vymedzenými písmenami	nepozičná sústava rímskych čísiel

netreba zmieňovať.

Obdobne sa dá hovoriť o hexadecimálnej (pre laika nie veľmi známej) číselnej sústave.

Napokon sa prichodí zmieniť o reprezentácii celočíselných množstiev rímskymi číslami. Porovnaním vzájomne sa odlišujúcich reprezentácií nasledujúcich čísiel

$$479 = IIII IIII = CDLXXIX$$

$$974 = IIII O O IIII = CMLXXIV$$

je zrejmé, že vzhľadom na expresivitu sú si ekvivalentné, významne sa však líši ich odvodzovacia účinnosť. Porovnaním aritmetických operácií sa ľahko zistia zvýšené nároky na odvodzovanie aj výpočtovú zložitosť.

Výpočtová účinnosť¹⁶ je označením výpočtovej zložitosti, resp. efektivity procedúr vykonávaných (situačne) so vzájomne previazanými REPREZENTANTAMI REPREZENTOVANÉHO. Volba reprezentanta dokáže aj z hľadiska efektivity výrazne ovplyvniť použiteľnosť navrhnutého systému. Preto je žiaduce voliť prostriedky reprezentácie, ktoré sú vyhovujúce aj z pohľadu výpočtovej účinnosti.

Výpočtová účinnosť

Predmetom záujmu pri skúmaní prostriedkov reprezentácie poznatkov a znalostí sú vhodné symboly a symbolové štruktúry. A práve pri ich voľbe je veľmi dôležité ozrejniť si uvedené tri charakteristiky a ich optimálne vyváženie.

Vyjadriteľnosť a použiteľnosť poznatkov/znalostí sú dôležitými vlastnosťami všetkých prostriedkov reprezentácie. Každý z takých prostriedkov disponuje určitým systémom symbolových štruktúr, ktoré sú nositeľmi spôsobilosti čosi vyjadriť a použiť.

Pri uplatnení hociktorého programovacieho jazyka, použiteľné štruktúry reprezentácie sú determinované prípustnými syntaktickými konštrukciami jazyka ako aj prípustnými spôsobami organizovania údajov pre procesy, ktoré sú realizovateľné programami vytvorenými v danom jazyku. To je skutočnosť, ktorá aj v súvislosti s rozvojom metód UI, ale aj z mnohých iných (aj teoreticky motivovaných) dôvodov viedli a vedú k vzniku

- programovacích jazykov obsahujúcich a rozširujúcich predchádzajúce (napr. LISP, Prolog, FRL, KRL, C++, SmallTalk a iné),
- nových metód programovania (napr. volanie procedúr cieľom alebo operáciami, ktoré sú schopné vykonať, namiesto volania procedúr menom, situačná aktivácia procedúr nazývaných démon, objektovo-orientované metódy a programovanie, multiagentové systémy a množstvo iných),
- prostredí na vytváranie aplikácií (uviesť treba predovšetkým vývojové prostredia tvorby

¹⁶ Čitateľ, povedzme vynásobením dvoch čísiel (napr. vyššie uvedенých) reprezentovaných dekadicky, binárne, rímskymi číslami, sa môže pragmaticky presvedčiť o líšiacom sa počte nevyhnutných elementárnych operácií vynútených použitou reprezentáciou a ňou implikovanými prostriedkami odvodzovania. Aj v prípade realizácie zodpovedajúcich výpočtových procesov počítačom narazíme na takéto rozdielne počty operácií.

problémovo zameraných ES).

Sú prostriedkom zvyšovania expresivity reprezentácie a použiteľnosti reprezentovaného.

Kvôli ilustrácii porovnajme produkčné pravidlá a prvorádový predikátový počet:

Vyjadrovacia účinnosť v oboch prípadoch je daná aserciami typu **AK-TAK** či **KEDĎ-POTOM**. Navyše, v prípade predikátového počtu dajú sa reprezentovať aj *termy s vlastnosťami* (unárnymi aj n-árnymi) entít.

V súvislosti s **použiteľnosťou**, t.j. **odvodzovacou účinnosťou**, v prípade produkčných pravidiel možno hovoriť o odvodzovaní na základe modus ponens, modus tollens, techník prehľadávania **A-ALEBO** grafu a v prípade prvorádového predikátového počtu o dokazovanie teorém na základe substituovania, unifikácií a rezolvovania.

Všimnime si, že teoreticky veľmi dobre prepracovaný prostriedok reprezentácie poznatkov/znalostí akým je jazyk prvorádovej predikátovej logiky (skrátene značíme *JPL*), svojimi pravidlami zostavovania správne utvorených formúl určuje prípustné reprezentačné štruktúry a tým vyjadriteľnosť poznatkov - expresivitu reprezentácie - aj termového typu. Použiteľnosť poznatkov (*proceduralita tohto formalizmu, jeho odvodzovacia účinnosť*), t.j. ako sa dá to či ono urobiť, vedie k problematike nachádzania účinných procedúr dokazovania teorém (substituovanie, unifikácia a rezolvovanie) na základe daných axióm a daných či odvodených teorém.

Treba si tiež všimnúť aj ohraničenia JPL vo vzťahu k *expresivite aj odvodzovacej účinnosti*. Napríklad v súvislosti s vyjadrovacou účinnosťou vznikajú ťažkosti pri priamočiaram a jednoznačnom reprezentovaní vzťahov (v prirodzenom jazyku bežne používaných) ako napr. "vzájomne", "respektíve" (**vyjadrenie vzťahu medzi prvkami z dvoch alebo viacerých zoznamov**), "spoločne", "musí", "môže" a pod. Nie je tomu inak keď je potrebné vyjadriť vierohodnosť predikátov, či nekategoricky kvalifikovať platnosť proposície výrazmi ako "zväčša", "takmer", "spravidla", "často", "zriedka", "súčasne", "následne", ako aj iných, pre uvažovanie človeka veľmi významných kvalifikátorov.¹⁷

Ilustrácia

Prvorádový predikátový počet v úlohe reprezentačného formalizmu

2.2 Znalosti: začlenené poznatky

¹⁷ Ako už bolo spomenuté, to je dôvod pre obohacovanie pôvodného formalizmu **JPL** mimologickými konštrukciami a pre používanie (nie bezproblémovo) prostriedkov neklasických alebo logík vyššieho rádu.

Znalosti sú začlenené, vzájomne previazané poznatkz

Ak hovoríme o *znalostiach*, nemáme na mysli navzájom *izolované poznatky*, ale aj ich **vzájomné súvislosti, previazanosť**. Preto pri reprezentácii znalostí je významná požiadavka

ZAČLENITEĽNOSTI POZNATKOV

Z toho plynie požiadavka na prostriedky reprezentácie poznatkov:

mali by poskytovať možnosti na začleňovanie poznatkov.

Sústavu symbolovo reprezentovaných začlenených poznatkov nazývame **báza znalostí (symbolovo BZ)**.

Nasledujúce články tejto kapitoly stručne uvádzajú najčastejšie sa vyskytujúce reprezentačné prostriedky, s ktorými sa stretávame v **UI**. Hoci sa medzi ne radí aj **JPL**, samostatne sa ním nezaobráme¹⁸. Na tomto mieste sa však žiada uviesť, že v prípade prostriedkov typu **JPL** začleniteľnosť poznatkov vyplýva z **implicitných** vzájomných súvislostí medzi formulami v dôsledku jednotnej interpretácie rovnakých symbolov.

Príklad: Všetky formuly, v ktorých sa vyskytuje predikát reprezentovaný povedzme symbolom tvoreným reťazcom **OTEC**, trebárs v atóme **OTEC(x,y)**, tento predikát reprezentuje ten istý vzťah medzi ľubovoľnými premennými **x** a **y**, ako je to aj vo výraze

STARÝ_OTEC(x,z) \equiv OTEC(x,y) & OTEC(y,z).

Pokiaľ sa v procese odvodzovania symboly premenných nahradzujú menami indivíduí, každá identická premenná sa substitúje tou istou indivíduovou konštantou. A tak prostredníctvom rovnakých symbolov sú implicitne zabezpečené vzájomné súvisiace poznatky a tým je zabezpečená ich vzájomná začleniteľnosť.

Naviac, čitateľ, ako sa dá očakávať, rozpozna, že **JPL**, resp. jeho rigorózne princípy sa dajú identifikovať v pozadí ostatných reprezentačných prostriedkov, hoci interpretácia zodpovedajúcich deklaratívnych programov nespočíva na princípoch rezolvencie. O týchto reprezentačných prostriedkoch pojednávame v nasledujúcich článkoch.

2.2.1 Reprezentácia znalostí produkčnými pravidlami

Produkčné pravidlá

Produkčné pravidlo je

➤ **propozičný výrok typu AK-TAK, KEĎ-POTOM**

- vyjadrením poznatku *asertívneho* typu
- určitej dávky (*kvanta*) *asertívnych znalostí*.

Uplatňovanie produkčných pravidiel je motivané tým, že sa jedná o teoreticky dobre zdôvodnený a prebádaný formalizmus, ktorý vznikol v oblasti *výpočtových výskumov logiky*. V matematickej informatike, v podobe *prepisovacieho systému*, je používaný ako jeden z modelov výpočtov. Svoje uplatnenie nachádza aj v oblasti *kognitívnej psychológie pri výskumoch myslenia človeka* (Newell-Simon, Johnson-Laird).

Štandardná štruktúra produkčného pravidla je zložená z dvoch častí:

- ◆ z **ľavej strany (LS)**, ktorá sa volá aj **predpokladová** či **situačná**, tiež **AK-časť**, v logike sa hovorí **antecedent**, v anglickej odbornej literatúre sa často stretávame s výrazom *left [hand] side (L[H]S)*
- ◆ z **pravej strany (PS)**, ktorá sa volá aj **dôsledková** či **akčná**, tiež **TAK-časť**, v logike sa hovorí

a ich štruktúra

Ľavá strana

¹⁸ Predpokladá sa aspoň čiastočná čitateľová znalosť **JPL**.

Pravá strana	<p>konzekvent, v anglickej odbornej literatúre sa často stretávame s výrazom <i>right [hand] side (R[H]S)</i>.</p>
Štruktúra predpokladovej časti	<p>Predpokladová časť (propozičný výraz AK-časti pravidla) je tvorená formulou v syntakticky vopred stanovenej podobe, ktorá sa skladá z jednej alebo viacerých podmienok (atómov). Ak sú podmienky v požadovanej štruktúre splnené, tak predpoklad je splnený, čo implikuje aj platnosť dôsledkovej časti pravidla \Rightarrow oprávnenosť vyvodiť určitý záver (platnosť, resp. neplatnosť referencovaného faktu) alebo vykonať istú akciu.</p> <p>Formula tvoriaca predpokladovú časť, môže byť vytvorená kombináciou rôznych podmienok vzájomne viazaných logickými operátormi.</p> <p>Podmienka, tvoriaca atóm, je elementárnym prvkom <i>predpokladu</i>.</p> <p>Aj <i>akčná (dôsledková) časť</i> pravidla môže byť tvorená kombináciou viacerých zložiek. (Podrobne sa o tom pojednáva v 4. kapitole venovanej reprezentácii asertívnych znalostí.)</p> <p>Formalizmus produkčných pravidiel je najfrekvencovanejším prostriedkom reprezentácie znalostí pretože</p>
Prednosti formalizmu produkčných pravidiel	<ul style="list-style-type: none"> ◆ umožňujú prirodzeným spôsobom vyjadrovať poznatky asertívneho typu (<i>vyjadriteľnosť</i>), ◆ ich interpretácia umožňuje prirodzeným spôsobom imitovať postupnosti úvah (usudzovaní) človeka (<i>použiteľnosť</i>), ◆ vysvetľovanie a zdôvodňovanie riešiacich postupností (vysvetľovacím mechanizmom expertného systému¹⁹) je pomerne priamočiare a teda aj jednoduché, ◆ umožňujú postupnú modulárnu tvorbu a upravovanie báz znalostí, ◆ dá sa použiť ako nositeľ mimologických reprezentačných konštrukcií: <ul style="list-style-type: none"> ▪ najmä pri reprezentovaní rôznych druhov neurčitostí, ▪ situačne sa uplatňujúcich riadiacich príznakov (semafórov, návěstí, priorit a pod.). <p>Symbolovo píšeme:</p>
	$s; p \rightarrow d; a,$
	<p>kde</p> <ul style="list-style-type: none"> ◆ $p \rightarrow d$ sa nazýva jadrom produkčného pravidla - vyjadruje propozičný poznatok tvrdiaci, že splnenie (logická pravdivosť) predpokladu p implikuje platnosť, dôsledku d; ◆ symbol s sa nazýva semafór alebo návěstie pravidla – je to vlastne <i>riadiaca podmienka</i>, ktorá situačne pripúšťa alebo zabraňuje interpretovať (použiť) pravidlo; ◆ symbol a zodpovedá (riadiacim) akciám, ktoré sa <i>situačne</i> vykonajú po vyhodnotení pravidla. (Podrobnosti najmä v 4. kapitole.) <p>Medzi nedostatky produkčných pravidiel (<i>produkčných systémov</i>) vzhľadom na ich odvodzovaciu účinnosť sa radí</p> <ul style="list-style-type: none"> ◆ ich spôsobilosť reprezentovať relatívne <i>izolované poznatky</i>, čo bez dodatočných opatrení, najmä reprezentovania poznatkov termového typu, vedie spravidla len k <i>lokálne ponímaným riešiacim postupom</i> bez priamočiarych možností realizovať riešenie z určitého globalizujúceho pohľadu; je to dané aj tým, že v prípade produkčných pravidiel, tak ako v prípade JPL, začleniteľnosť poznatkov vyplýva iba z implicitných súvislostí plynúcich z jednotnej interpretácie rovnakých symbolov, ◆ produkčný systém bez vhodného štruktúrovania a prípadných riadiacich konštruktov (napr. semaforů alebo mimologické konštrukty reprezentujúce neurčitosti) vedie k riešiacim procesom, ktoré nie je možné považovať za adekvátne usudzovacích procesov človeka.
Jadro produkčného pravidla	
Semafor	
Akčná časť	
Nedostatky reprezentácie	

¹⁹ Avšak pre mnohé aplikácie vysvetľovanie produkované iba na báze znalostí reprezentovaných produkčnými pravidlami nie je postačujúce.

2.2.2 Reprezentácia znalostí asociatívnou sieťou

Asocionistický prístup k reprezentácii

Problematika **začleniteľnosti** poznatkov zvyrazňuje **asocionistické predstavy**. Tie vo vývoji reprezentačných prostriedkov viedli k vzniku **asociatívnych (sémantických) sietí**. Motiváciou bola snaha o prekonávanie ťažkostí s izolovanou reprezentáciou poznatkov, ktoré sa prejavujú v prípade formalizmov **JPL** a produkčných systémov. Išlo teda o snahu nájsť formalizmus umožňujúci reprezentovať **konceptuálne jednotky - TERMY**.

Konceptuálnou jednotkou sa rozumie *čosi, čo možno opísať, čo má nejaké vlastnosti, o čom môžeme čosi vedieť a čosi predpokladať, o čom môžeme mať informácie*.

Konceptuálna jednotka

Asociatívne siete v **úlohe reprezentačného prostriedku** mali umožniť združovanie všetkých informácií a poznatkov o danej konceptuálnej entite tak, aby boli čo najbezprostrednejšie **sprístupiteľné z miesta, kde je symbolová reprezentácia tejto entity uložená**.

Myšlienka spočíva v nájdení účelnej a efektívnej organizácie údajov reprezentujúcich znalosti typu **termov** zodpovedajúcich

Združovanie informácií a poznatkov

- ◆ **konceptom** (pojmom, entitám), s ktorými sú združené nie len opisy, ale implicitne (to vplýva na konštrukciu funkcií interpretátora) aj **spôsoby ako s nimi zachádzať**,
- ◆ možnosti vyjadrovať len **čiasťočnú znalosť**, ktorá sa dá postupne **doplňovať**,
- ◆ metódam umožňujúcim **porovnávať daný pojem s ostatnými v sieti** a tým **spoznať kontext**, v ktorom sa vyskytuje.

Asociatívne siete sa dajú vo všeobecnosti charakterizovať takto:

- ◆ **konkrétne** (individuové) aj **všeobecné pojmy** sú **reprezentované samostatnými vrcholmi** v sieti,
- ◆ **elementárne poznatky** sú reprezentované **podgrafmi siete**, ktoré s každým javovým vrcholom obsahujú **všetky jeho povinné atribúty** (vlastnosti),
- ◆ **každému pojmu zodpovedá najviac jeden vrchol** v sieti a **všetky s ním súvisiace informácie sa dajú vyčítať z jeho väzieb s okolitými vrcholmi**, menovite ide najmä o väzby na
 - **všeobecnejšie pojmy ktoré ho obsahovo vymedzujú,**
 - **na špeciálnejšie pojmy - až po individua, ktoré sú jeho inštanciami (prípady),**
 - **javy, na ktorých sa daný pojem zúčastňuje.**

Vymedzenie asociatívnej siete

Prednosť týchto sietí spočívajú v

- ◆ možnosti vytvárať **taxonómie entít - vzťahy generalizácia-spezializácia**,
- ◆ úspornosti reprezentácie - **zahrňovanie**, bez potreby opakovať vlastnosti získavané z nadtriedy,
- ◆ reprezentácii vzťahov **exkluzivity** - vzájomného vylučovania sa entít tej istej nadtriedy.

Praktická symbolová reprezentácia asociatívnych sietí sa dá realizovať prostriedkami vhodného **JPL** napríklad s predikátmi typu

JE_PODMNOŽINOU (*trieda1, trieda2*),

JE_PRVKOM (*trieda, individuum*),

JE_ČASŤOU (*trieda1:individuum_a, trieda2:individuum_b*)

JE_ČASŤOU (*individuum1, individuum2*),

JE_VLASTNÍKOM (*trieda1:vlastník, trieda2:vlastnené, čas:začiatok, čas:koniec*),

JE_PRÍPADOM (*vlastnenie:vlastníctvo_i*),

VLASTNÍCTVO_i (vlastník:Ivan, vlastnený:Dunčo, čas:1991, čas:1992).

Prednosti
asociatívnych
sietí:
taxonómie,
zahrňovanie,

Medzi výrazné nedostatky asociatívnych sietí je nevyhnutné radiť skutočnosť, že tieto explicitne neurčujú procesy produkujúce postupy usudzovania. Bez toho, aby boli obohatené príslušným produkčným systémom, alebo zodpovedajúcimi procesmi v rámci interpretátora (čo by nebol adekvátny prístup), nie sú prostriedkom generovania príslušných procesov.

exkluzivita

Lineárny zápis
siete

Nedostatky

2.2.3 Procedurálna reprezentácia poznatkov

Rozdielne
postavenie
>procedurality<

Na úvod tohto článku je potrebné ozrejmiť dôležitý rozdiel medzi *proceduralitou stelesnenou interpretátorom* (procedurálnym programom) deklaratívneho programu a *procedurálnou reprezentáciou znalostí, ktoré tvoria súčasť deklaratívneho programu*.

Procedúry
deklaratívneho
programu

Deklaratívny program môže obsahovať množstvo vzájomne sa líšiacich partikulárnych procedúr reprezentujúcich proceduralne poznatky, t.j. aj opakovane použiteľné deterministické postupy, návody použitia. Ich spúšťanie je však situácie podmienené, pre sekvencie ich potenciálneho použitia sa deterministicky nepredpisujú.

Procedúry
procedurálneho
programu

Procedurálny program sa odlišuje tým, že spravidla viacnásobne používa všeobecné parametrizovateľné procedúry na báze vopred deterministicky predpísaných postupností. Dá sa to ilustrovať aj na interpretátore, ktorý stelesňuje napr. spravidla v jednom behu opakovane používané procedúry ako je hľadanie komplementárnych literálov, rezolvovanie, modus ponens atď. Uplatňovanie takých procedúr je determinované syntaxou prostriedkov reprezentácie poznatkov.

Keď teda v tomto článku hovoríme o proceduralnej reprezentácii poznatkov, máme na mysli

špecifické operacionistické znalosti - také, ktoré vykonávajú špecifické problémovo zamerané operácie.

Vznik tohto druhu reprezentácie bol motivovaný zdôrazňovaním **použiteľnosti** poznatkov prostredníctvom súboru **procedúr**, ktorými sa reprezentujú **partikulárne** (parametrizovateľné) **poznatky majúce povahu určitého procesu**.

Jednou z najvýznamnejších charakteristík tohto spôsobu reprezentácie je

Operacionis-
tické znalosti

volanie procedúr cieľmi, t.j. na základe ich funkcie, či ich obsahu.

Ide o odlišný princíp v porovnaní s volaním programu menom! Ilustrujeme to na príklade.

Uvažujme triviálny prípad: identifikácia mnohostenu. Konkrétne, v danej situácii treba zistiť, či skúmaný objekt je kockou. Symbolova podoba formulácie cieľa úlohy by mohol mať nasledujúci tvar

Cieľ: **KOCKA** (objekt:W)

Volanie procedúr cieľmi

Takáto formulácia požiadavky vedie k zisťovaniu toho, či objekt **W** vyhovuje požiadavkám predikátu **KOCKA**, alebo ináč, či je za reťazcom "Cieľ:" uvedená atomárna **propozícia** pravdivá. Z druhej časti predošlej vety vyplýva, že ide vlastne o **zodpovedanie otázky: Je objekt W kockou?** Odpoveď sa má získať na základe nasledovných symbolovo vyjadrených faktov

Daná úloha, či otázka

SÚČASŤ (objekt:W, stena:A) & **ŠTVOREC** (stena:A)
SÚČASŤ (objekt:W, stena:B) & **ŠTVOREC** (stena:B)
SÚČASŤ (objekt:W, stena:C) & **ŠTVOREC** (stena:C)
SÚČASŤ (objekt:W, stena:D) & **ŠTVOREC** (stena:D)

PROCEDÚRA_1 (zisťuje počet stien mnohostenu)

...

SÚČASŤ(objekt:x, stena:y) (propozícia, test)

...

POČET_STIEN (objekt:x) := **POČET_STIEN** (objekt:x) + 1 (počítadlo a pmätanie výsledku)

...

...

ne procedúry (sú procedúry cieľom).

Dostupné fakty

PROCEDÚRA_2 (zisťuje tvary stien)

...

POČET_STIEN (objekt:x) = 6 (propozícia)

...

ŠTVOREC (stena:y) (propozícia)

...

KOCKA (objekt:x) (cieľová propozícia)

...

Cieľom volané procedúry

Špecifikovaný cieľ (**KOCKA** (objekt:W)) riešenia problému vedie k tomu, že interpretátor (po substitúcii) identifikuje použiteľnosť druhej z uvedených procedúr (zodpovedajúci literál sa nachádza v tele **PROCEDÚRA_2**). Pri zbiehaní tejto procedúry sa narazí na propozíciu **POČET_STIEN** (objekt:W)=6. Jej vyhodnotenie sa stáva podcieľom. Interpretátor hľadá procedúru, v ktorej sa vyskytuje premenná **POČET_STIEN** (objekt:W) na ľavej strane priradovacieho príkazu. V danom prípade identifikuje **PROCEDÚRA_1** a preto ju inicializuje.

Je zrejmé, že interpretátor musí stelesňovať **prostriedky substituovania, unifikácie a príslušného prehľadávania procedúr**. Na základe stanoveného cieľa, resp. aktuálneho podcieľa, **prehľadáva** procedurálne reprezentované poznatky a **vyberá** z nich tie, ktoré môžu svojím účinkom zabezpečiť dosiahnutie výsledku. Tým realizuje postupné, **vopred presne nedeterminované reťazenie** partikulárnych procedúr a teda **produkuje** riešiacu postupnosť.

Odvodzovací proces nahrádzajúci slepé prehľadávanie procedúr, ktoré sú súčasťou deklaratívneho programu, cieľnými postupmi sa zabezpečuje zodpovedajúcimi prostriedkami. K ním patria stavové a riadiace údajové štruktúry, ktoré slúžia **informovanému rozhodovaniu** v odvodzovacom procese. Ide napríklad o štruktúry, ktoré

- ✓ zjednodušujú identifikáciu procedúr potenciálne zabezpečujúce dosiahnutie práve aktuálnych (pod)cieľov,

a priebeh ich volaní

- ✓ registrujú aktuálny stav jednotlivých procedúr, t.j. ich stav aktivovania, zamietnutia, prerušenia, opustenia²⁰ a pod.
- ✓ zaznamenávajú situačné použitie jednotlivých procedúr a ich úspešnosť v jednotlivých situáciách atď.

Medzi nevýhody procedurálnej reprezentácie poznatkov radíme stratu možností postupného, ľahko modifikovateľného a nezávislého reprezentovania jednotlivých poznatkov, potenciálne ťažkosti so zrozumiteľnosťou deklaratívneho programu (môže sa v ňom vyskytovať neprehľadné rekurzívne volanie procedúr) a nebezpečenstvo, že sa doň zavedú neľahko identifikovateľné nekonzistentnosti.

O začleniteľnosti procedurálne reprezentovaných poznatkov možno vysloviť nasledovné:

- ✓ V prípade bežných programovacích jazykov sa spravidla jedná o **explicitné** odvolávanie na procedúry ich menami. V takom prípade súvislosť medzi dvomi zložkami programu vyplýva z výskytu mena procedúry v roli argumentu operácie volania na jednom mieste a na druhom mieste v roli identifikátora určitej sekvencie kódov.
- ✓ V prípade deklaratívnych programov sa vlastne tiež jedná o explicitné vzťahy. Rozdiel je iba v tom, že tie sú špecifikované symbolovými konštruktami, ktoré v prípade odvolávania tvoria **zložku propozície (podmienky)** a v prípade volaného prvok **prirad'ovacieho príkazu** v tele určitého programu.

Informované rozhodovanie

Nevýhody procedurálnej reprezentácie

Začleniteľnosť procedurálne reprezentovaných poznatkov

2.2.4 Reprezentácia znalostí rámcami

MOTIVÁCIA RÁMCOVEJ REPREZENTÁCIE ZNA-

Motiváciu vzniku tejto formy reprezentácie **znalostí termového typu** je potrebné vidieť v snahách motivovaných hľadiskami

²⁰ Význam týchto pojmov sa bližšie osvetľuje v neskorších častiach týchto textov.

LOSTÍ

- ♦ **INFORMATCKÝMI: účelná a účinná syntéza atribútov vyjadri-
tel'nosti, použiteľ'nosti, efektivity a začleniteľ'nosti.**
- ♦ **PSYCHOLOGICKÝMI: reprezentácia PREDSTÁV A OČAKÁVANÍ (mentálny
model) súvisiacich s určitou entitou a procesov ich zlad'ovania so
skutočnosťou.**

Ilustračný
príklad

Majme ilustračný príklad na osvetlenie podstaty reprezentačného formalizmu rámcového typu. Uvažujeme v ňom prípad osoby, ktorá pre ňu v nie dostatočne známej budove má nájsť zasadaciu miestnosť, v ktorej sa má zúčastniť rokovania:

1. **Nachádzajúc sa osamotene (nemá sa koho opýtať) v priestrannej hale (chodbe) budovy, v ktorej je rad číselne (nie funkčne, t.j. nápismi alebo ikonami) označených dverí, medzi ktorými sa dajú predpokladať aj tie, čo vedú do cieľovej miestnosti.**
2. **Očakáva, že po ich otvorení uvidí určité charakteristiky - atribúty miestnosti:** povedzme štyri istým spôsobom upravené steny vyzdobené obrazmi, kdesi v stenách okná a prípadne ďalšie dvere, podlahu vyloženú parketami čiastočne pokrytú kobercom, strop so svetidlom, dlhší konferenčný stôl obklopený pohodlnejšími stoličkami, nevylučuje kávu, či osviežujúce nápoje na stole, prítomnosť tabule, premietacieho plátna, spätného projektora a pod.
3. Dá sa predpokladať, že v myslí má **pripravenú predstavu, určité očakávanie**, t.j. akýsi (zrejme všeobecný a pomerne flexibilný) **rámec** (v psychologizujúcom poňatí mentálny model) zodpovedajúci **konceptu** primeranej zasadacej miestnosti.
4. V závislosti od toho, čo po otvorení dverí naozaj uvidí, **pôvodné predpoklady/očakávania sa budú naplnia, alebo sú v rozpore s realitou**. Ak by otvorené dvere viedli do záhrady, k schodom na povalu, či do pivnice, dôjde k nezhode reality s danou predstavou. Ak by za otvorenými dverami nebodaj zazrela šíry oceán, nezhoda by mohla dosiahnuť stupeň šoku. V prípade, že by išlo o kancelársku miestnosť, predsieň, klasickú učebňu, čítareň a pod. môže situačne dôjsť k **čiasťočnej zhode s rámcovou predstavou** miestnosti.
5. Iba ak by to bola naozaj typická zasadačka alebo miestnosť, ktorá podľa všeobecných predstáv by mohla slúžiť ako zasadačka, došlo by k dostatočnej zhode **medzi očakávanými a skutočnými vlastnosťami reality**, pričom sa **modifikujú a konkretizujú hlavné charakteristické predstavy** o očakávaných vlastnostiach miestnosti (napr. tvar a pozícia okien, druh, kvalita, rozmery a umiestnenie nábytku, veľkosť a kvalita koberca a pod.) a to nezávisle od toho, či je alebo nie je v miestnosti tabuľa, spätný projektor s premietacím plátnom, či na stole je nejaké občerstvenie, alebo či miestnosť má ešte ďalšie dvere.

Tento príklad hovorí o určitých predstavách/očakávaniach a naznačuje proces ich konfrontácie a zlad'ovania s realitou. Treba pritom zvýrazniť, že niektoré z očakávaní, aj keď sú modifikovateľné, sú považované za **nevyhnutné podmienky zhody** (ak by namiesto stola/stolov boli v miestnosti posteľ, kuchynské zariadenie, rad vaní, či bazén, nebola by splnená nevyhnutná podmienka), iné tvoria iba fakultatívne podmienky (absencia koberca, občerstvenia, tabule, ...).

Čo sú teda rámce ako prostriedky reprezentácie ?

Rámce sú údajové štruktúry, ktoré integrujú prednosti

- ♦ **asociatívnych sietí, vrátane možností reprezentovať vlastnosti entít a hodnôt, ktoré sú pre ne povinné, prípustné, očakávateľné (predpokladateľné), vrátane spôsobov ich nadobúdania - výhodné na reprezentovanie termových znalostí,**
- ♦ **reprezentácie asertívnych poznatkov v podobe produkčných pravidiel, resp. formúl logických formalizmov (vrátane JPL),**
- ♦ **procedurálnej reprezentácie znalostí, teda akciami, cieľmi, či menom volaných procedúr.**

O čosi podrobnejšie sa dajú rámce špecifikovať nasledujúcim prehľadom charakteristík:

RÁMCE

- ✓ sú svojim **menom identifikovateľné údajové štruktúry**, ktoré sú vzájomne prepojitelné (môžu na seba odkazovať, odvolávať),
- ✓ ich **štruktúra** obsahuje (aj hierarchicky) **usporiadanú postupnosť rôznych pomenovaných a typovo charakterizovateľných údajových položiek**, nazýva-ných **rubrika**, ktoré sa jednak v

Nevyhnutné a
nepovinné
očakávania

Špecifikácia rámcov	rozsahu rámca môžu na seba odvolávať a jednak sú referencovateľné aj z iných rámcov alebo referencujú iné rámce či ich rubriky (menom rubriky a menom rámca, v ktorom sa nachádzajú),
Charakteristiky rámcov	<ul style="list-style-type: none"> ✓ rubriky zodpovedajú vlastnostiam reprezentovanej entity, sú to rezervo-vané pamäťové miesta, v ktorých buď už sú požadované údaje, alebo sa tieto do nich, na základe pozorovania skutočností, postupne vkladajú ako ich hodnoty v súlade so špecifikovaným typom rubriky, ✓ (dátový) typ rubriky (ich šablóna) určuje požiadavky, resp. ohraničenia vzťahujúce sa k povoleným hodnotám rubriky (napr. kardinalitu, t.j. či je prípustná iba jediná alebo aj viacnásobná hodnota, prípustné vlastnosti hodnoty - napr. celočíselná, reálna, či komplexná hodnota kvantity, pravdivostná (propozičná) hodnota, reťazcová hodnota, meno procedúry, telo procedúry, produkčné pravidlo atď.), ✓ (aj dynamicky meniteľnými) predpismi prioritného spôsobu nadobúdania hodnôt jednotlivých rubriky resp. prioritného prehľadávania a vyhodnocovania obsahu rubriky, ✓ povinnými (obligatórnymi) rubrikami s povinnými hodnotami (nevyhnutnými vlastnosťami reprezentovanej entity, napr. <i>minimálny počet stien v miestnosti s mnohouholníkovou podlahou: 3, počet okien v obývacej miestnosti ≥ 1</i>, [okno je reprezentovateľné vlastným rámcom]), fakultatívnymi rubrikami s voliteľnými hodnotami (zodpovedajúce možným, avšak nepovinným vlastnostiam entity, napr. <i>rozmer steny, farba steny, prítomnosť okna v stene, rozmery miestnosti</i> [rubrika môže obsahovať aj situačne aktivovanú procedúru, ktorá vypočíta túto hodnotu z rozmerov stien]) a iné,
Identifikovateľná, vnútorne štrukturovaná sústava údajových položiek	<ul style="list-style-type: none"> ✓ obsahom (hodnotou) rubriky, ktorý môže byť vopred definovaný a fixný, v priebehu odvodzovania získaný fixný, alebo dočasný a modifikovateľný (dynamický), náhradný (zástupný, predpokladateľný, 'default') nahradzujúci nepoznanú, len očakávateľnú, domnelú skutočnosť,
Rubriky	<ul style="list-style-type: none"> ✓ hodnotami rubriky (spravidla obligatórnymi) sa špecifikujú <ul style="list-style-type: none"> ▪ zovšeobecnia reprezentovanej entity, ich nadtriedy, ktoré sú tiež reprezentované rámcami (napr. miestnosť je prvkom triedy <i>uzavretý priestor</i>), ▪ špecializácie, t.j. buď ich podtriedy alebo individuové prípady, ktoré sú tiež reprezentované rámcami (napr. miestnosť má podtriedy: obývačka, spálňa, sála, trieda, hala, a iné, pričom zasadačka č. 5 na konkrétnej adrese je individuová entita), ▪ súčasti reprezentovanej entity ako odkazy na zodpovedajúce rámce (stena, okno, podlaha, spätný projektor, ...), ▪ príčiny a dôsledky reprezentovanej entity odkazujúce na príslušné rámce, ▪ procedúry (identifikátor alebo telo procedúry môže byť obsiahnutý v rubrike),
a ich dátový typ	<ul style="list-style-type: none"> ✓ s rubrikami je možné spájať aj explicitnú alebo implicitnú podmienku spustenia im zodpovedajúcich procedúr pri nadobudnutí, modifikovaní, zrušení ich hodnoty (napr. vyhľadanie najbližšej voľnej stoličky, pozdraviť prítomných, ak sa už nachádzajú v miestnosti, keď je potrebné vypočítať plochu podlahy, zistenie rozmerov a ich uplatnenie v zodpovedajúcej výpočtovej procedúre).
spôsob nadobúdania a prehľadávania hodnôt	
povinné a nepovinné hodnoty	
fixné a dynamické hodnoty	
špecifikovanie zovšeobecnení,	

špecializácii a

častí entít

kauzalít,
procedúr
hodnotami
rubrik

procesy
priradené
rubrikám

2.3 Prostriedky reprezentácie znalostí

Prístup k
tvorbe ES

Všetky spomenuté prostriedky možno realizovať na základe vlastnej koncepcie v niektorom z bežných programovacích jazykov. Prirodzene všetky majú vzhľadom na problematiku reprezentácie svoje výhody aj nedostatky. V aplikáciách metód umelej inteligencie pôvodne bolo možné pomerne často sa stretávať s programovacími jazykmi ako je **LISP** alebo **PROLOG**. V súčasnosti sa však skôr stretávame s objektovo-orientovanými programovacími jazykmi, napr. **C++** alebo **SMALLTALK**.

Otázka optimálneho výberu programovacieho jazyka tvorí mnohodomennú problematiku. Každý možno hodnotiť z viacerých hľadísk a tie sú spravidla protichodné. Naviac, pri hodnotení sa uplatňujú aj požiadavky aplikácií. Komplexný rozbor celej problematiky, vzhľadom na nevyhnutné predpoklady, rozsah a náročnosť presahuje zámer týchto textov. Preto len značne všeobecná úvaha: výhodné je použiť jazyk, ktorý je výkonný pri uskutočňovaní rozmanitých reťazcových operácií (operácie so znakmi) a poskytuje dostatok možností na tvorbu rozmanitých údajovo-riadiacich štruktúr, ktoré sú nevyhnutné pre efektívne riadenie interpretácie deklaratívneho programu.

Keď sa jedná o tvorbu netriviálnych reprezentačných prostriedkov, realizácia zodpovedajúcich produktov je intelektuálne, rozsahom a prácnosťou zvyčajne veľmi náročná. V snahe racionalizovať príslušné programátorské činnosti, boli vytvorené viaceré špeciálne programovacie jazyky, napr. **KRL** (Knowledge Representation Language), **FRL** (Frame Representation Language), **OWL** (One World Language) a iné. Tieto jazyky mali predovšetkým teoretický význam, v praxi nenašli príliš veľké uplatnenie.

Odporúčanie v
prvom
priblížení

V súčasnosti sa uplatňujú predovšetkým **vývojové prostredia** tvorby expertných systémov. Sú to programové prostriedky, ktoré umožňujú vytvárať

- **deklaratívny program**, čo v kontexte expertných systémov sa nazýva **báza znalostí**,
 - **interpretátor deklaratívneho programu**, čo v kontexte ES sa nazýva **inferenčný mechanizmus**,
 - **používateľské rozhrania** - v kontexte ES nazývané **komunikačný modul**,
- a iné.

Príkladmi takýchto (komerčne dostupných) prostredí sú napr. Egeria, M4, Leonardo, ART-IM, KEE, Kappa, NexpertObject, RTworks, G2 a rastúci počet ďalších. V novších verziách, či variantoch týchto prostredí sa uplatňujú princípy objektovo-orientovaných programovacích technológií.

Vývojové
prostredia –
súčasný trend

Hoci rôzne prostredia poskytujú rozmanité možnosti, ktorými sa dá vyhovieť požiadavkám *expresivity, odvodzovacej účinnosti, výpočtovej účinnosti a začleniteľnosti*, aj tie majú svoje vzájomne sa líšiace prednosti a ohraničenia. Sú tiež nositeľmi rôznych implementačných náhodností, ktorých neželaný dopad sa môže prejaviť až po dlhšom používaní. Vzhľadom na ich vysokú nadobúdajúcu cenu (cena niektorých vývojových prostredí sa pohybuje rádovo na úrovni miliónov

SK), pred ich zakúpením je potrebné veľmi starostlivo zvažovať a preskúšavať, či vyhovujú požiadavkám aplikácií. Ich vhodnosť sa spravidla preukáže až pri konkrétnom a opakovanom použití. Preto je výhodné pred ich potenciálnym zakúpením zabezpečiť ich skúšobné používanie.

Tak ako pri programovacích jazykoch, ani pri vývojových prostrediach nie je zatiaľ možné poskytnúť dostatočne všeobecné návody na ich optimálny výber. Rozhodne však treba uprednostňovať prostredia, ktoré vzhľadom na aplikácie poskytujú čo najpružnejšie možnosti reprezentácie a používania znalostí. Menovite také, čo umožňujú

- ✓ prístup k tomu istému poznatku z rôznych aspektov,
- ✓ realizáciu rôznych reprezentačných formalizmov,
- ✓ vytvárať rozmanité problémovo zamerané makro-operácie.

Najmä pokiaľ ide o makro-operácie (stretne sa s nimi ešte v ďalších kapitolách) je dôležité, aby ich bolo možné vytvárať na základe vhodnej konceptualizácie riešiacich postupov danej aplikačnej oblasti bez toho, aby ich tvorba príliš podliehala vlastnostiam daného programového prostredia.

3. REPRESENTÁCIA ZNALOSTÍ TERMOVÉHO TYPU

Vymedzeni e problému	<p>Každý konkrétny problém je špecifikovaný a vymedzený množinou faktov, cieľov a (prípadne aj explicitne formulovanými) ohraničeniami. Nezávisle na kvalite formulácie problému (jednoznačnosť, úplnosť, vymedzenie prípustných operácií, implikáciu riešiaceho postupu, rozsah explicitných a implicitných ohraničujúcich podmienok atď.) je počiatočný (východzí) stav riešenia daný množinou údajov zodpovedajúcich formulácii. V priebehu riešenia problému spravidla pribúdajú nové informácie – sú produkované ako dôsledok interakcií s okolím a ako záznamy významných fenoménov vznikajúcich v priebehu riešiaceho postupu, ktoré sa prípadne môžu uplatniť v nasledujúcich krokoch postupu. Zhromažďované <i>aj vyjavujúce (emergujúce)</i> údaje umožňujú</p>
Počiatočný a	<p>➤ jednak úplnejšie a presnejšie opisovať problém, prostredie (svet), v ktorom sa rieši, a požiadavky splňovania ohraničujúcich podmienok,</p>
priebežný opis problému	<p>➤ zároveň umožňujú charakterizovať (opisovať)</p> <ul style="list-style-type: none"> ▪ priebeh riešenia, ▪ použité operácie, ▪ uplatnené predpoklady ("čo-ak") a náhradne údaje, vrátane ich postavenia v procese riešenia (napr. ich aktuálnosť, nerozhodnuteľnosť, zamietnutie), ▪ efektívnosť a úspešnosť riešiacich postupov, ▪ následnosť niektorých ďalších krokov. <p>Kým údaje zodpovedajúce opisu problému (teda známe fakty) vytvárajú v terminológii ES to čo sa nazýva báza faktov (symbolovo BF), charakteristiky riešiacich postupov sú zaznamenávané v údajových štruktúrach tvoriacich ich infraštruktúru (sú predmetom pozornosti v nasledujúcej kapitole).</p>
Báza faktov	<p>Báza faktov – ako dynamická údajová štruktúra - tvorí spolu s bázou znalostí (BZ) a inferenčným mechanizmom (IM) jadro architektúry expertných systémov. V závislosti na konkrétne zvolenej architektúre je BF²¹ buď iba komplementárnou súčasťou (relatívne) statickej BZ, alebo je to jej samostatný dynamický prvok. V oboch prípadoch sa však jej obsah, pokiaľ používateľ neaktivuje program archivácie, po ukončení riešenia problému zaniká. Nezávisle na skutočnom spôsobe implementácie BF, bez toho, aby to ovplyvňovalo obsah nadväzujúceho výkladu, v nasledujúcom budeme predpokladať, že BF je samostatná zložka ES.</p>
a báza znalostí	<p>Po tomto uvedení zameriame pozornosť najprv na prostriedky reprezentácie <i>znalostí termového typu</i> a následne <i>znalostí asertívneho typu (v rozsahu produkčných pravidiel)</i>, teda na</p>

²¹ V niektorých literárnych prameňoch sa termín *báza faktov* nahrádza výrazom *báza údajov (databáza)*. V kontexte expertných systémov uprednostňujeme termín *báza faktov*. Dôvodom pre to je odlišná povaha **BF** v porovnaní s **DB**. Kým *databázou spravidla* rozumieme rozsiahle zoskupenia **rovnorodo štruktúrovaných údajov**, nie je to prípad zoskupovania faktov, ktorými je determinovaný problém riešený **ES**. V tomto prípade postupne zhromažďované údaje spravidla nemajú ani uniformnú povahu, ani začlenenie a štruktúru. Ich obsahová rôznorodosť a formálna mnohorakosť motivuje k odlišnému pomenovaniu.

klúčové prostriedky tvorby **BZ**²². Tým sa dostávame k významným otázkam implementačnej úrovne a v súvislosti s tým sa výklad týka aj obsahového naplňovania **BF**.

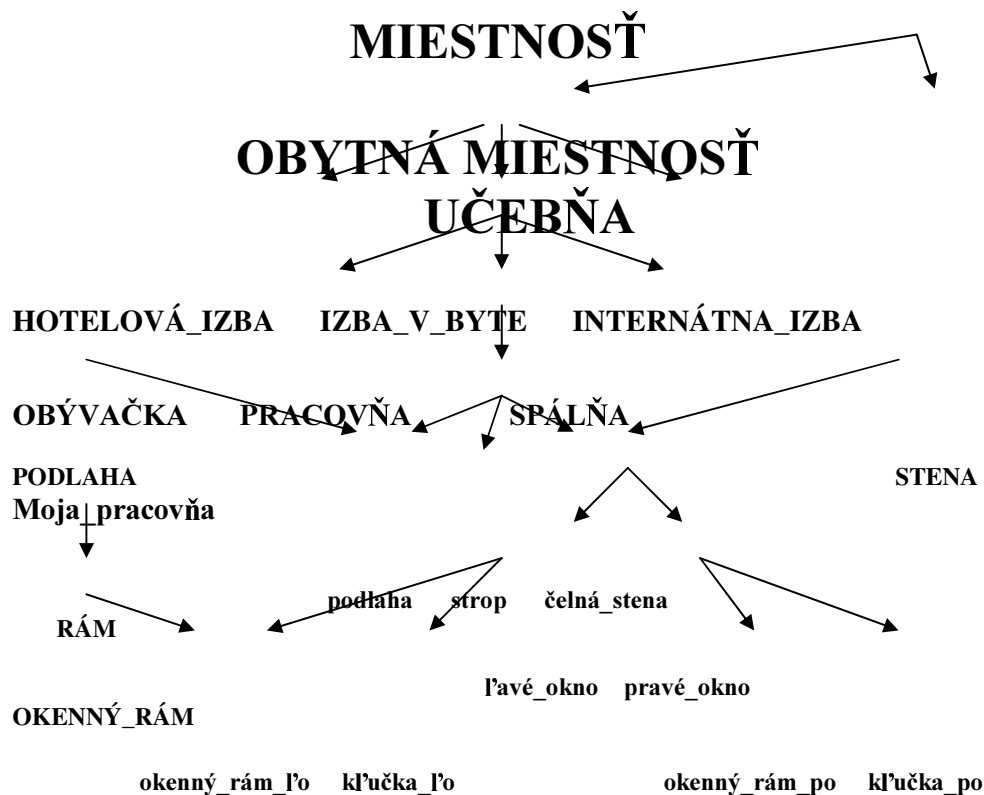
3.1 Základné pojmy

Term ako pojem, koncept, jav, entita, objekt	<p>Termami opisujeme, definujeme, vymedzujeme pojmy, koncepty, javy, entity, objekty. Sú to východiskové výrazy, ktoré z hľadiska rozboru zodpovedajúcich reprezentačných formalizmov sa dajú považovať za <i>synonymá</i>. Preto ich v rozsahu nasledujúcich textov nebudeme rozlišovať. Kvôli jednoduchosti najčastejšie budeme používať termín objekt ako zastupujúci výraz nahradzujúci aj ostatné.</p> <p>Údajová štruktúra, ktorú použijeme na symbolovú reprezentáciu <i>objektu</i>, bude v predošlej kapitole zavedená štruktúra RÁMEC. Je <i>to identifikovateľná údajová štruktúra zodpovedajúca určitému kvantu (predovšetkým) opisných znalostí</i>.</p> <p>Rámcom je reprezentovaný každý z nasledovných nami uvažovaných typov objektov:</p>
Rámec – reprezentačná štruktúra	<ul style="list-style-type: none"> ➤ objekt typu trieda (označíme T-OBJEKT, resp. T-RÁMEC) - zodpovedá kategórii (množine) objektov, ktoré majú (z istého pohľadu) spoločné určujúce vlastnosti; prvkami triedy sú buď jej podtriedy, tiež T-OBJEKTY, alebo individuové objekty, ➤ individuový objekt (označíme I-OBJEKT, resp. I-RÁMEC)²³ - zodpovedá jedinečnému, svojim identifikátorom jednoznačne odlišiteľnému objektu; ktorý je vždy prvkom aspoň jedného T-OBJEKTu: I-OBJEKT ∈ T-OBJEKT;
Objekt-trieda	<p>I-OBJEKT sa môže členiť na svoje individuové prvky – súčasti, t.j. na S-OBJEKTY,</p> <ul style="list-style-type: none"> ➤ objekt typu súčasť – tiež subobjekt (označíme S-OBJEKT, resp. S-RÁMEC) – má vždy duálne postavenie: <ol style="list-style-type: none"> (1) je jednak individuovým, teda I-OBJEKTom patriacim aspoň do jednej triedy, teda je prvkom aspoň jedného T-OBJEKTu, a zároveň (2) je sám časťou (zložkou) individuového objektu.
Objekt-individuum	<p>Každý (T-,I-,S-)objekt môže byť súčasne prvkom viacerých tried. Na obr. 3.1 je uvedený príklad, ktorý je ilustráciou možných vzťahov medzi reprezentovanými objektami.</p> <p>Na označovanie jednotlivých typov identifikátorov budeme používať typograficky odlišené identifikátory:</p>
Subobjekt	<ul style="list-style-type: none"> ▪ len veľké písmená - T-OBJEKT, ▪ veľké začiatkové písmeno, ostatné malé - I-OBJEKTom, ▪ len malé písmená - S-OBJEKT.

²² Uplatňujú sa pritom *objektovo-orientované* pohľady, ktoré v mnohom aproximujú vlastnosti vývojového prostredia *NexpertObject*.

²³ Príklady individuových objektov (vždy jesthuje ako jedinečný, s iným nestotožiteľný): Vojtech Zamarovský (autor knihy Grécky zázrak); Octavia BA-3344-AB; nízkonapäťový kábel 2990 E52853 počítača v mojej pracovni.

Typografické
odlíšenie



Obr. 3.1

Symbolový (lineárny) prepis 'hierarchickej siete' z obr. 3.1 sa dá vyjadriť napríklad v nasledujúcej podobe:

- kľučka_po{(je_časťou: pravé_okno)};
- okenný_rám_po{(je_časťou: pravé_okno)};
- kľučka_ľo{(je_časťou: ľavé_okno)};
- okenný_rám_ľo{(je_časťou: ľavé_okno),
(je_prvkom: OKENNÝ_RÁM)};
- OKENNÝ_RÁM{(je_podtriedou: RÁM)};
- ľavé_okno{(je_časťou: čelná_stena)};
- pravé_okno{(je_časťou: čelná_stena)};
- čelná_stena{(je_časťou: Moja_pracovňa),
(je_podtriedou: STENA)};
- strop{(je_časťou: Moja_pracovňa)};
- podlaha{(je_časťou: Moja_pracovňa),(je_prvkom: PODLAHA)};
- Moja_pracovňa{(je_prvkom: PRACOVŇA)};
- OBÝVAČKA{(je_podtriedou: IZBA_V_BYTE)};
- SPÁLŇA{(je_podtriedou: IZBA_V_BYTE)};
- PRACOVŇA{(je_podtriedou: IZBA_V_BYTE)};
- HOTELOVÁ_IZBA{(je_podtriedou: OBYTNÁ_MIESTNOSŤ)};
- IZBA_V_BYTE{(je_podtriedou: OBYTNÁ_MIESTNOSŤ)};
- INTERNÁTNA_IZBA{(je_podtriedou: OBYTNÁ_MIESTNOSŤ)};
- OBYTNÁ_MIESTNOSŤ{(je_podtriedou: MIESTNOSŤ)};
- UČEBŇA{(je_podtriedou: MIESTNOSŤ)};

Takýto lineárny zápis hierarchického acyklického grafu sa dá zovšeobecniť napr. do nasledujúcej podoby:

```

IDENTIFIKÁTOR_RÁMCA {
(identifikátor_rubriky: hodnota_rubriky),
(identifikátor_rubriky: hodnota_rubriky),
...
}

```

V generickom (aj keď nie úplnom) BNF vyjadrení píšeme

```

RÁMEC ::= IDENTIFIKÁTOR_RÁMCA {TELO_RÁMCA}
IDENTIFIKÁTOR_RÁMCA ::= REŤAZEC|Ret'azec|ret'azec
TELO_RÁMCA ::= (telo_rubriky) | {TELO_RÁMCA},(telo_rubriky)
telo_rubriky ::= identifikátor_rubriky: hodnota_rubriky
identifikátor_rubriky ::= ret'azec
hodnota_rubriky ::= hodnota | hodnota_rubriky,hodnota

```

Keď sa počiatočná štruktúra siete rámcov, t.j. počty rámcov a väzby medzi nimi nemenia, menia sa iba počty rubriek (zvažovaných vlastností) a konkretizujú sa ich hodnoty, hovoríme *o statickej reprezentácii, o modeli statických vzájomných väzieb (začlenení) vopred daných objektov reprezentovaného sveta.*

Lineárny zápis
rámca

Často je však potrebné riešiť aj problémy, pre ktoré nie sú vopred dané a spoznané všetky objekty s ktorými je potrebné uvažovať, ani ich vzájomné vzťahy. Typicky sa to týka **úloh generatívnych a syntetizujúcich** (napr. keď sa zostavuje plán určitej činnosti, zostavuje reprezentačné družstvo, či keď sa konštruuje nový doposiaľ nejestvujúci a teda ani nereprezentovaný objekt). Vtedy je nevyhnutné rátať s tým, že objekty môžu

- ◆ sa vyskytovať v rôznych časovo meniacich reláciách (vzťahoch),
- ◆ v priebehu riešenia problému vznikajú, modifikujú a rušia.

To vyžaduje **dynamickú reprezentáciu**, t.j. **reprezentáciu meniacich sa stavov modelovaného sveta**. V nej je prípustné napríklad preradiť začlenenie I-OBJEKTu z jedného do iného T-OBJEKTu, alebo vytvorenie, modifikovanie, zrušenie niektorého z objektov. To sú procesy, z ktorých prirodzene vyplýva potenciálne rad mnohých ďalších nevyhnutných zmien.

Rámec
definovaný v
tvare BNF
zápisu

Statická a
dynamická
reprezentácia

3.2 Vlastnosti objektov

Objekty, popri tom, že sa **vzájomne začleňujú**, sú špecifikované svojimi **vlastnosťami**.

Vlastnosť, napríklad *hmotnosť, farba, veľkosť, intenzita, rýchlosť, vek, trvanie, byť študentom, diplomantom, informatikom, zamestnancom, súrodencom, záujemcom, pacientom, majiteľom atď.* sa môže týkať množstva záležitostí a tým teda sa vzťahovať k mnohým (I-, T-, S-) OBJEKTOM.

Fenomén **vlastnosť** je tiež **konceptom**, teda je aj objektom. A v tomto zmysle má vlastnosť všeobecnú a *genericкую* povahu.

je aj konceptom

Generická vlastnosť nie je nositeľom žiadnej hodnoty, má uniformné poslanie v rozsahu celej BZ.

Generická vlastnosť

Vlastnosť je v štruktúre rámca reprezentovaná rubrikou. Rubrika je jej nositeľom a zároveň aj **rezerváciou pamäťového miesta**, do ktorého sa **kladá hodnota** danej vlastnosti²⁴.

Vlastnosť má svoje pomenovanie (identifikátor) a aj vlastné (**meta**)vlastnosti.

Metavlastnosť

Metavlastnosť

- ♦ **vymedzuje druh, rozsah a tvar/formu hodnoty vlastnosti** (rozlišujeme vlastnosť a jej hodnotu), **je teda akousi šablónou pre hodnotu** (v angličtine sa hovorí *template, facet*),
- ♦ **može byť aj určitým predpisom situačne podmieneného spôsobu získavania hodnoty a prípadných následných opatrení vyžadovaných nadobudnutou alebo modifikovanou hodnotou.**

Z uvedeného je zrejmé, že samotná metavlastnosť môže byť tvorená komplexnou štruktúrou pozostávajúcou z viacerých položiek. Najčastejšie sa stretávame s nasledujúcimi:

- ♦ **KD: kardinalita hodnoty vlastnosti**
- ♦ **DT: dátový typ hodnoty vlastnosti**
- ♦ **NH: náhradná (default) hodnota**
- ♦ **NV: spôsob nadobúdania vlastnosti**
- ♦ **AP: spôsob nadobúdania hodnoty vlastnosti**
- ♦ **AZ: akcie, ktoré sa musia vykonať pri zmene hodnoty vlastnosti**

Tak ako *vlastnosť* je v štruktúre rámca reprezentovaná *rubrikou* aj *metavlastnosť* je v tejto štruktúre reprezentovaná svojim protažskom – **metarubrikou**.

Prvky metavlastnosti

Metarubriky

- ♦ umožňujú pridávať k reprezentácií deklaratívnych znalostí **implicitne používané procedurálne zložky** – zodpovedajúce *situačne vyvolávaným akciám*,
- ♦ sú **dediteľné**, preto je výhodné špecifikovať ich v čo **najvšeobecnejšom** postavení v rámci hierarchickej siete.

Metarubrika

KONKRÉTNÁ VLASTNOSŤ vzniká priradením *generickej vlastnosti* (T-I,S-) OBJEKTU. Tým povinne nadobúda aj kardinalitu a dátový typ zodpovedajúcej metavlastnosti, ale nie spôsob nadobúdania vlastnosti a jej hodnoty, ani akcie vykonávané pri zmene hodnoty vlastnosti.

Kardinalita vlastnosti (KD) určuje prípustný počet hodnôt vlastnosti: **jediná – niekoľko**. Takýto predpis prispieva ku konzistentnosti bázy faktov a zároveň umožňuje flexibilnú reprezentáciu reality. Ide vlastne o vyjadrenie toho, že napr. vlastnosti akými sú 'váha', 'výška', 'rodné číslo', 'stav_vypínača', 'farba_tekutiny' atď. pripúšťajú iba jedinú hodnotu, kým vlastnosti ako 'dni_týždňa', 'prednášky_z_UI', 'mená_rodíčov', 'mená_súrodencov', 'záujmy', 'predpísané_lieky', ako

²⁴ Vlastnosť s priradenou hodnotou, ktorá však nie je iba predpokladaná, je považovaná v týchto textoch za **fakt**.

aj 'je_podtriedou', 'je_prvkom', 'je_čast'ou' atď. sú vlastnosti pripúšťajúce viacnásobné hodnoty.

Dátový typ (DT) charakterizuje a ohraničuje hodnoty, ktoré vlastnosť smie nadobúdať, napr.:

Konkrétna
vlastnosť

- ◆ **prirodené, celé, reálne**, alebo **komplexné číslo** prípadne s určitým ohraničením prípustných hodnôt, napr. intervalom (potrebne napr. v prípade hodnôt rovinného uhla, funkcie sinus, telesnej teploty, pravdepodobnosti, ako aj prípustných diskretných hodnôt ističov elektrických obvodov alebo obsahu na trhu dostupných fliaš,
- ◆ **číselný interval**, považovaný za údaj kardinality **1**
- ◆ **hodnota času** - v zmysle usporiadanej postupnosti zoskupených číslíc s priradenou časovou jednotkou, pričom pre jednotlivé jednotky môžu platiť líšiac sa pravidlá periodicity,
- ◆ **dátum** - podobné vlastnosti, ako v predošlom (prípadne zohľadňujúc medzinárodné odlišnosti foriem zápisu dátumov ako aj možných rozdielností v uplatňovaných letopočtoch),
- ◆ **súradnice** - n-rozmerný vektor údajov, pričom v závislosti na uplatňovaných konvenciách sa buď jedná o kardinalitu **1**, t.j. súradnice sa chápu ako jediný údaj, alebo o obligatórnu kardinalitu **n**,
- ◆ **pravdivostná (propozičná) hodnota** - boolovská alebo viachodnotová,
- ◆ **reťazec** buď bez ohraničení, alebo ohraničený generický *vzorom* reťazca, či prípustnou množinou predefinovaných povolených výrazov (napr. názvov farieb, dní v týždni, vyučovacích predmetov a pod.)

Kardinalita
vlastnosti

Keď sa vlastnosti priradí **dátový typ**, ten je platný pre celú BZ !

Dátový typ

Medzi *vlastnosťami* je vhodné vyčleniť jednu s osobitným postavením. Nazvime ju **GLOBAL**. Jej osobitosť spočíva v jej úlohe: na rozdiel od individuálnej vlastnosti reprezentovanej entity **INTEGRUJE HODNOTY** *iných jej vlastností*, t.j. **vzťahuje sa na reprezentovanú entitu ako na celok**. Jej hodnota je výsledkom **celkového (výsledného) hodnotenia objektu**²⁵.

Na hodnotu rubriky GLOBAL sa možno odvolávať najmä v súvislosti s rozhodovaním vo vetviacich miestach riešacieho procesu.

Výnimočné postavenie (úloha) tejto vlastnosti je daná aj tým, že sa na ňu nevzťahuje požiadavka univerzálnej platnosti jej dátového typu!

Metavlastnosť vlastnosti má prirodzene svoj prot'ajšok v metarubrike. Vyjadríme to aj v rozšírenom predchádzajúcom BNF zápise:

```
telo_rubriky ::= [metarubrika]* identifikátor_rubriky: hodnota_rubriky
metarubrika ::= (identifikátor: hodnta) | metarubrika, (identifikátor: hodnta)
identifikátor ::= KD | DT | NH | NV | AP | AZ
```

Znižovanie nárokov na pamäťové miesta, na prácnosť tvorby deklaratívneho programu (BZ) a povaha produktívneho (situačného) riešenia problémov vyžaduje používanie prostriedkov umožňujúcich zabezpečovať súvisiace požiadavky s čo najväčšou efektívnosťou. To je kontext, v ktorom sa v nasledujúcich častiach týchto textov zaoberáme vlastnosťami objektov, spôsobami akými nadobúdajú hodnotu, dôsledkami, ktoré môžu vyplynúť zo zmien týchto hodnôt, dedením meta-vlastností, vlastností a ich hodnôt.

²⁵ Majme napr. proces (hľadania odpovede na otázku), v ktorom je potrebné *identifikovať* skúmaný objekt, povedzme tigra. Ak sa na základe určitých charakteristických hodnôt vlastností (správanie alebo spôsob života, spôsob získavania potravy, počet nôh, pokrytie povrchu tela a jeho zafarbenie, tvar a rozmiestnenie očí, vydávané zvuky a i.), ktoré sú reprezentované rámcami TIGER a PUMA, porovnajú vlastnosti pozorovaného zvierat'a, tak v závislosti na ich zhode s realitou, a nezávisle napr. na mieste pozorovania, veľkosti, počte mláďat a pod., hodnota vlastnosti **GLOBAL** hovorí o tom, či sa podarilo rozpoznať - prípadne aj s akou virohodnosťou tigra.

Vlastnosť
zvaná
GLOBÁL

Rozšírený BNF
zápis

3.3 Hodnota vlastností a spôsoby ich nadobúdania

3.3.1 Špecifické a iniciálne hodnoty vlastností

Zavedieme najprv dve špecifické hodnoty: **NEZISTENÁ**, **NEZNÁMA**. Ich význam uvádza nasledujúca tabuľka

HODNOTA	NEZISTENÁ	NEZNÁMA
VÝZNAM HODNOTY	absencia hodnoty vlastnosti , ktorá ešte nebola zisťovaná – je reprezentovaná určitým symbolom (hoci aj prázdny znakom)	napriek už zisťovanej hodnote vlastnosti , (bez ohľadu na príčinu) nepodarilo sa ju zistiť

Iniciálny stav

Keď sa problém začína riešiť, t.j. keď sa *inicializuje* činnosť ES spolu so zodpovedajúcou BZ, obsahy/hodnoty niektorých rubriík rámcov už môžu byť známe, väčšina rubriík však hodnoty nemá. Je to prirodzené, veď v iniciálnom stave riešenia neštandardné hodnoty vlastností špecifikujúce realitu nemohli byť ešte zisťované. Preto teda prevážna väčšina hodnôt rubriík je interpretovaná v zmysle *hodnoty NEZISTENÁ*.

Špecifita oboch vyššie vymedzených hodnôt spočíva v tom, že ich výskyt a zmena sa dá použiť na situačné vyvolanie zodpovedajúcich osobitných akcií (pozri odsek 3.3.3 a 3.3.4). Hodnoty vlastností, ktoré sú známe už na začiatku riešenia problému (sú preddefinované, resp.

vyplývajú zo zadania úlohy), majú atribút **iniciálne**. Zodpovedajú všeobecne platným poznatkom a ohraničeniam danej problémovej oblasti. Sú vyjadrené konkrétnymi hodnotami vlastností v rubrikách rámcov napr. (napätie_spotrebiča: 220V), (tvar_ozubeného_kolesa: čelný_šikmý), (dorozumievací_jazyk: anglický), (povolená_maximálna_rýchlosť: 130 km/hod), (hodnota_ističa: 10 A), (nosnosť_mosta: 3.5t) a pod.

Z hľadiska úsporného využívania pamäte nie je výhodné vkladať známe iniciálne hodnoty do všetkých rubriek referencujúcich príslušnú vlastnosť. Nie je napríklad ani účelné, ani a funkčne v líšiaciach sa reprezentáciách elektrospotrebičov na 220 V uvádzať túto hodnotu dovtedy, kým sa nestanú predmetom pozornosti. **Iniciálne hodnoty stačí teda uchovávať iba raz v jedinom pamäťovom mieste a v čase vykonávaní programu, keď sa referencuje konkrétna rubrika s takou hodnotou, pôvodný symbol NEZISTENÁ sa v takej rubrike nahradí zodpovedajúcou iniciálnou hodnotou.**

Iniciálne hodnoty

Takto, až v čase behu programu získaná hodnota rubriky sa zvykne nazývať **vykonávaná iniciálna hodnota** (v angličtine - "run-time" value).

V *metarubrikách* vlastností sa fakultatívne môže uvádzať tzv. **náhradná** (tiež *očakávaná* či *predpokladateľná* – *angl. default*) **hodnota** vlastností. Je to tiež preddefinovaná hodnota, ktorá sa nezáväzne uplatňuje iba keď by mala vlastnosť nadobudnúť hodnotu **NEZNÁMA**.

Náhradná hodnota

3.3.2 Spôsoby nadobúdania hodnôt vlastností

AP: ak_potrebné

Položka **AP (ak-potrebná)** v *metarubrikách* vlastností svojim obsahom **určuje prioritu spôsobov získavania hodnôt vlastností**. Hovorí sa tiež o tzv. **priorite zdrojov**. Ide o fakultatívne uvedenie rôznych spôsobov, ktorými referencované vlastnosti môžu v čase činnosti ES situačne v konkrétnych prípadoch nadobúdať svoje hodnoty. Uplatňujú sa rôzne spôsoby určovania priorít - najčastejšie usporiadaním jednotlivých možností (zdrojov) do postupnosti zodpovedajúcej zvoleným prioritám, alebo tým, že nezávisle od ich usporiadania sa im priradia kvantitatívne hodnoty (váha) určujúce priority.

Obsahom *metarubriky AP* je zoznam akcií, ktoré sa majú vykonať na určenie hodnoty príslušnej vlastnosti vtedy, keď je táto potrebná. Akciami môžu byť napríklad:

- ◆ **odvodenie hodnoty** na základe zodpovedajúcich produkčných pravidiel alebo špecifikovaných výpočtových procesov,
- ◆ **načítanie hodnoty z externého zdroja údajov** (povedzme z databázy či špecifikovaného súboru, alebo tiež z vyrovnávacej pamäti zodpovedajúceho on-line zariadenia),
- ◆ **zdedenie (prevzatie) hodnoty** z iného rámca,
- ◆ **použitie náhradného údajá,**
- ◆ **predloženie otázky používateľovi a načítanie jeho odpovede**

Odvodenie hodnoty

a prípadne iné.

Načítanie hodnoty

Pokiaľ usporiadanie spôsobov získavania údajov určuje ich prioritu, jednotlivé akcie sa vykonávajú postupne v uvedenom poradí a po prvej úspešnej akcii sa už v ďalších nepokračuje (McCarthyho algoritmus).

Dednie hodnoty

Aj pri uvádzaní *metarubriky AP* je potrebné zohľadňovať efektívnosť využívania pamäte. V súvislosti s tým sa uplatňujú stratégie prípustnej špecifikácie účinnosti obsahu **AP** na (1) prázdny²⁶ expertný systém, (2) konkrétnu bázu znalostí, (3) generickú vlastnosť, (4) konkrétnu vlastnosť T-rámca, (5) konkrétnu vlastnosť I-rámca.

Náhradný údaj
Dotazovanie

²⁶ Expertný systém bez bázy znalostí sa označuje termínom **prázdny expertný systém**.

Uvedené usporiadanie sa uplatňuje tak, že pokiaľ stratégia nie je definovaná na danej úrovni, uplatní sa stratégia na úrovni vyššej. Preto by vždy mala byť definovaná stratégia aspoň na úrovni ES, čo by bola úroveň "default-u". Je prípustná aj výhodná koexistencia všetkých uvedených špecifikácií (stratégií). Výhodnosť spočíva v úspore pamäti aj programátorskej práce: **pokiaľ vyhovuje všeobecnejší predpis, nie je ho potrebné opakovane uvádzať**.

3.3.3 Účinok AP pri referencovaní nezistenej hodnoty vlastnosti

Metóda a démon

V kontexte tejto podkapitoly uvedieme najprv ďalší pojem - **metóda**. Označuje sa ňou situačne, t.j. **iba v prípade potreby aktivovaná procedúra**. Používa sa synonymické pomenovanie *metódy* - **démon**.

Vhodným ilustračným príkladom **metódy** je procedúra, ktorá sa uplatňuje keď metarubrika **AP** obsahuje prioritne usporiadané spôsoby nadobúdania práve referencovanej hodnoty a tá je **NEZISTENÁ**. Nech obsah **AP** je *odvodenie, dedenie, ..., dotazovanie*. Potom *metóda* by mala stelesniť nasledovný algoritmus:

(1) **AK** je referencovaná hodnota rubriky s AP vo svojej metarubrike,

TAK skok na 5

INAK skok na 2

(2) **AK** zodpovedajúca rubrika najbližšieho nadradeného **OBJEKTU** má vo svojej metarubrike AP,

TAK prevezmi (zded') jej obsah, použi ho ako vlastnú AP a skok na 5

(3) **AK** hodnota rubriky je dôsledkovou časťou niektorého produkčného pravidla,

TAK (spätným reťazením – kap. 4) odvod' príslušnú hodnotu,

AK hodnota bola úspešne odvodená,

TAK skok na 7

INAK skok na 4

(4) **AK** hodnotu rubriky možno získať dedením,

TAK ju podľa určenej stratégie (v ďalšom) získaj zhora alebo zdola,

AK hodnota bola úspešne získaná,

TAK skok na 7

INAK použi ako by bol vlastný štandardný predpis (platný v **BZ**, resp. **ES**) a skok na 5

(5) Vykonať postupnosť akcií z AP počnúc prvou a pokračuj dovtedy, kým sa nezistí hodnota rubrik

AK sa nepodarilo hodnotu rubriky určiť,

TAK skok na 6,

INAK skok na 7

(6) Opýtaj sa na hodnotu používateľa

(7) Koniec

Z algoritmu je zrejmé, že v činnosti zodpovedajúceho *démona* zohráva významnú úlohu

Dynamické zmeny priorít

usporiadanie obsahu metarubriky **AP**. Zvýšená pružnosť uplatňovania spôsobov nadobúdania hodnoty vlastnosti sa dá dosiahnuť tým, že v procese riešenia sú **dynamicky meniteľné priority akcií v AP**, ale aj tým, že produkčným pravidlám použiteľným na odvodzovanie hľadanej hodnoty sú priradené priority obdobne ako aj stratégiám dedenia (v nadväzujúcom) a odvodeniu hodnoty na základe špecifikovanej procedúry (napr. v akčnej zložke produkčného pravidla, či v **AZ** metarubrike).

Najfrekvencovanejšie sa uplatňujú zmeny priorít *odvodenia a dedenia*. Umožňujú to **rubriky priorít odvodzovania (inferencie) a dedenia** v metarubrike.

Zavedieme dohovor: Na rubriku rámca budeme odkazovať identifikátorom tvoreným z dvoch častí oddelených bodkou. Zložka pred bodkou bude zodpovedať identifikátoru rámca - vo vyššie zavedenom zmysle - a za bodkou identifikátoru rubriky. Ilustrácie:

'Rámec_1.rubrika_7', 'R.v', 'XXX.yyy'; všeobecne ID_rámca-ID_rubriky.

Je výhodné, keď v metarubrike sa aj rubriky *priority odvodzovania* aj *priority dedenia* skladajú z dvoch položiek: **priamej a nepriamej hodnoty priority odvodenie, resp. dedenia**. V oboch prípadoch je *priama hodnota priority* spravidla určená prirodzeným alebo celým číslom. Prázdny ES špecifikuje narastajúcu prioritu s narastajúcou alebo klesajúcou hodnotou. Opäť v oboch prípadoch rovnako sa *nepriama hodnota priority* určuje **odkazom** na rubriku určeného (pracovného) rámca, povedzme **R.odvod**, resp. **R.ded**, ako nositeľky hodnoty priority odvodenia, resp. dedenia.

Princíp zodpovedajúcich **metód** môže byť nasledovný:

- ◆ Ak **R.odvod** (*nepriama hodnota priority odvodenia*), resp. **R.ded** (*nepriama hodnota priority dedenia*) má hodnotu, tak tá sa stáva hodnotou priority.
- ◆ Ak **R.odvod**, resp. **R.ded** neboli definované, alebo nemajú hodnotu - v tom prípade sa ani nezisťuje, použije sa *priama hodnota priority odvodenia/dedenia*.
- ◆ Ak ani položka *priama hodnota priority odvodenia/dedenia* nie je deklarovaná, alebo nemá obsah, tak sa použije štandardne preddefinovaná hodnota pre BZ alebo systém.

Význam *nepriamej hodnoty priority odvodzovania, resp. dedenia* spočíva v tom, že v priebehu činnosti ES je jednoduché situačne **dynamicky** meniť obsah rubriky rámca, ktorá je nositeľkou hodnoty priority. Využívanie tejto danosti umožňuje meniť štandardne preddefinované postupy získavania hodnoty vlastností (ako aj postupy, ktoré sú dôsledkom ich prípadných zmien). Nastavenie obsahov položky *priamej hodnoty priority odvodzovania, resp. dedenia* umožňuje **staticky** prestaviť aplikačne nevyhovujúce štandardné priority. V oboch prípadoch sa jedná o adaptovanie BZ, resp. ES podľa potrieb konkrétnej aplikačnej oblasti.

Priama a nepriama hodnota priority odvodzovania a dedenia

3.3.4 AZ - dôsledky vyplývajúce zo zmeny hodnoty vlastnosti

AZ:
ak_zmenená

V priebehu činnosti ES môže situačne dochádzať k zmene hodnoty niektorej rubriky - vyplýva to z dynamiky reality, alebo z úvah typu '**čo by sa stalo ak**' (podrobnejšie v ďalšom). Taká zmena môže opäť situačne vyvolať potrebu rôznych navádzujúcich akcií. Obslúženie zodpovedajúcich javov sa dá zabezpečiť tiež procedúrou typu *démon*, ktorý sa vždy spustí pri zmene hodnoty rubriky.

Postupnosť akcií, ktorých vykonávanie zabezpečí, je podmienená typom zmeny hodnoty rubriky. Rozlišujeme nasledujúce typy zmien:

- ☞ **NEZISTENÁ na ZNÁMA** (dodanie údajov²⁷) - vtedy AZ kontroluje
 - *potenciálny nesúlad hodnoty s jej dátovým typom* (na prípadný nesúlad upozorní a umožní opravu),
 - *konzistentnosť hodnoty s už jestvujúcimi faktami* (detekcia prípadnej nekonzistentnosti vedie tiež k upozorňujúcej správe a možnosti úpravy vzájomne nekonzistentných údajov).
- ☞ **NEZISTENÁ na NEZNÁMA** - vtedy AZ môže situačne zabezpečiť napr.
 - *výber a voľbu inej metódy či postupu riešenia problému,*
 - *nahradenie neznámej (nezistite_nej) hodnoty vlastnosti výberom vhodnej predpokladateľnej, avšak nezáväznej hodnoty,*
 - prípadne aj iné činnosti podľa inície tvorca systému.
- ☞ **ZNÁMEJ** (zistenej alebo náhradnej) **hodnoty na inú ZNÁMU hodnotu**, vtedy AZ
 - vykoná najprv akcie ako v prípade '**NEZISTENÁ na známu**²⁸,
 - propaguje dôsledky vyplývajúce z detekovanej zmeny do ostatných súvisiacich rubriek, BF a prípadne aj iných (riadiacich) údajových štruktúr (napr. môže zabezpečiť úpravy iných hodnôt, keby zmenenou hodnotou v dôsledku určitej väzby by medzi nimi vznikla nekonzistentnosť, najmä v prípade *predpokladateľných náhradných údajov*).

Vykonanie zoznamu AZ akcií sa dá tiež podmieniť splnením špecifikovanej explicitnej alebo implicitnej podmienky. Pritom platí:

špecifikované akcie sa vykonajú v danom poradí hneď po detekovanej zmene, a pokiaľ sú podmienené, tak iba ak je splnená zodpovedajúca podmienka.

Pre dedenie AZ, t.j. pri dedení metarubriky je prípustná špecifikácia (pozri 3.4.2 a následné odseky) toho, že pri zmene hodnoty rubriky na NEZNÁMA, zmena nepodlieha AZ-akciám.

Pre tvorca BZ je veľmi dôležité mať na pamäti, že AZ-akcie sú prostriedkom udržiavania vzájomnej konzistentnosti faktov pri ich zmenách.

Príklad: Keď sa detekuje zmenená hodnota rubriky *teplota* v rámci, ktorý tvorí súčasť reprezentácie uzavretej nádoby obsahujúcej isté médium, potom sa musí bezprostredne zmeniť aj hodnota rubriky *tlak*.

Pri navrhovaní BZ nespúšťať zo zreteľa, že v priebehu odvodzovacej činnosti ES, **AZ-akcie môžu vyvolať rozsiahle sekvencie procesov !**

Prostriedok
udržiavania
konzistentnosti
BF

²⁷ Jestvujú systémy, ktoré rozlišujú zmenu hodnoty rubriky z NEZISTENÁ na inú od ostatných možných zmien. Tomu zodpovedá špecializácia démona AZ. Hovorí sa mu prípadne **AK-DODANÁ** (akronym AD), v angl. literatúre IF-ADDED.

²⁸ V prípade, že novou hodnotou sa stane NEZNÁMA, treba uplatniť akcie z '**NEZISTENÁ na NEZNÁMA**'.

3.4 Dedenie a zodpovedajúce stratégie

Hierarchická reprezentácia

Hierarchickým štruktúrovaním rámcov možno reprezentovať viaceré druhy vzťahov medzi reprezentovanými objektami. Najčastejšie sa stretávame s reláciou *všeobecnejšieho k špecializovanejšiemu*. Môže však ísť aj o vzťahy *predchodcu k následníkovi*, *nadradeného k podradenému*, *príčiny a dôsledku*, alebo *celku k časti*.

Zo syntaktického hľadiska (zväčša) tými istými štruktúrami sa reprezentujú sémanticky odlišné záležitosti. **Sémantická odlišnosť reprezentovaných vzťahov sa zabezpečuje ich odlišnou procedurálnou interpretáciou.**

Okrem iného aj spôsobom dedenia (prenosu, prisvojenia) rôznych typov údajov medzi hierarchicky štruktúrovanou reprezentáciou entít. Atribúty (t.j. vlastností) a vlastnosti týchto vlastností (metavlastností) nadradeného (v zmysle hierarchicky vyššie zaradeného) OBJEKTU sa tým potenciálne stávajú atribútmi podriadených (T-,I-,S-) OBJEKTov. Podriadené objekty, pokiaľ majú viac nadradených, môžu získavať atribúty zo všetkých z nich.

Hierarchické štruktúrovanie má iba vtedy zmysel, keď podriadené, t.j. špecifickejšie objekty, majú v porovnaní s nadriadenými niektoré atribúty navyše alebo iné.

Toto štruktúrovanie sa poníma v tom zmysle, že spoločné atribúty podriadených objektov "sú akoby vybraté pred zátvorku" a umiestnené v nadradenom objekte. Hierarchizovaná štrukturalizácia objektov reprezentujúcich termové znalosti, okrem toho, že je reflexiou **hierarchických taxonómií a generalizačno-špecializačných závislostí**, je výhodná aj z *úsporných a spoľahlivostných dôvodov*:

Zmysel hierarchickej reprezentácie

znižuje nároky na pamäť aj rozsah programátorských činností a zároveň zvyšuje konzistentnosť a tým aj spoľahlivosť vytvorenej BZ.

Táto prednosť sa uplatňuje tým, že systém zabezpečí medzi nadradenými a podriadenými objektami **dedenie**

☞ **vlastností**, t.j. *dedenie rubriek v zmysle identifikovateľných pamäťových miest*,

☞ **metavlastností**, t.j. *metarubriek, ktoré sú nositeľmi informácií o im zodpovedajúcich vlastnostiach, spôsobe, ktorým nadobúdajú hodnoty, a dôsledkoch, ktoré vyplývajú z ich zmien (AP a AZ)*,

☞ **hodnoty vlastností**, t.j. *obsahu rubriek*.

Dedenie vlastností

metavlastností

hodnôt vlastností

3.4.1 Dedenie hodnôt vlastností

Dedenie hodnôt vlastností

V procese odvodzovania zohrávajú kľúčovú úlohu *produkčné pravidlá*! V mnohých aplikáciách ES sa dá vystačiť len s nimi. V množstve iných by to však bolo neefektívne a z praktického hľadiska ani nie možné. Keď **symboly vystupujúce v zložkách produkčných pravidiel** (semafór, podmienky predpokladu jadra, dôsledok a akcie v akčnej časti) sú **vlastne odkazovaním na hodnotu vlastnosti entít (rubriek rámcov)**, symbolová reprezentácia znalostí, jej *začleniteľnosť* a *využititeľnosť* sa výrazne a z mnohých aspektov podstatným spôsobom obohacuje a zefektívňuje. Množstvo **poznatkov** sa tak dá reprezentovať prirodzene, úsporne, prehľadne a zrozumiteľne bez potreby ich **vyjadrenia a reprezentovania v asertívnej podobe, t.j.**

produkčnými pravidlami.

Symbol identifikujúci (odkazujúci, referencujúci) vlastnosť reprezentovanej entity (povedzme **R.r**) môže vystupovať v úlohe²⁹

- **dôsledku jadra produkčného pravidla:** vtedy je hodnota vlastnosti určovaná práve pravidlom ($p \rightarrow R.r$), vrátane aj tých, ktoré sa v nadväznosti s nim aktivujú (spätným reťazením - pozri v ďalšom),
- **podmienky predpokladu jadra produkčného pravidla:** podmienka (povedzme tvaru $R.r > 5$) sa proпозиčne vyhodnocuje v závislosti na hodnote vlastnosti,
- **podmienky v semafore produkčného pravidla,**
- **podmienky akcie alebo v príkaze akčnej časti produkčného pravidla** (napr. v roli prvku výrazu na na pravej strane priradovacieho príkazu: $R.a := R.r * R.b$).

Rôznosť úloh rubrik dediacich hodnoty

Teda keď hodnota rubriky **R.r** je **NEZISTENÁ**, tak sa postupuje podľa **AP** zo zodpovedajúcej metarubriky. **AP** môže byť buď lokálna, t.j. definovaná špecificky pre **R.r**, dedená (pozri v ďalšom), alebo štandardne platná pre BZ alebo ES. Keď **AP** obsahuje dedenie hodnoty vlastnosti, vtedy, v závislosti na práve platných prioritách, môže dôjsť k dedeniu príslušnej hodnoty.

Ak k dedeniu dôjde, štandardne sa uplatňuje nasledovný smer postupu dedenia:

T-OBJEKT (nadtrieda)

T-OBJEKT (podtrieda)

I-OBJEKT

S-OBJEKT

Platí pritom pravidlo:

k dedeniu hodnôt - na rozdiel od dedenia vlastností - **dochádza iba keď sa hodnota referencuje!**

Štandardný postup dedenia

Keď sa pri dedení hodnoty z nadradeného rámca zistí, že tento v príslušnej rubrike má hodnotu **NEZISTENÁ**, môžu nastať dva alternatívne prípady:

- spustí sa proces získavania príslušnej hodnoty pre nadradený rámec,
- vyberie sa ďalší nadradený rámec rovnakej úrovne nadradenosti a po vyčerpaní všetkých takých možností (podrobnosti v ďalšom) pokračuje sa podľa obsahu **AP**.

Na triviálnom príklade demonštrujeme priebeh dedenia hodnoty.

Majme T-OBJEKT s názvom *ŠTUDENT_MFF*. Nech zodpovedajúci rámec obsahuje rubriku *znalosť_cudzích_rečí*. Pokiaľ by platilo, že každý poslucháč MFF ovláda angličtinu, tak hodnotou tejto rubriky by bola (aj) *angličtina*. Generický rámec zodpovedajúci uvažovanému T-OBJEKTu by mohol mať napr. nasledujúcu podobu

ŠTUDENT_MFF { ..., (prospech:), ([AP] špecializácia:), (ročník:), (znalosť_cudzích_rečí: angličtina), ... }

Hodnota sa dedí iba keď sa na ňu odkazuje

Nech rámec s identifikátorom *Peter* je I-OBJEKT, ktorý je prvkom T-OBJEKTu *ŠTUDENT_MFF*, a nech jeho podoba pred referencovaním vlastnosti *znalosť_cudzích_rečí* zodpovedá rámcová štruktúra z ľavej strany. Rámcová štruktúra z pravej strany vznikne keď sa po uplatnení produkčných pravidiel, ktoré slúžia na rozhodovanie o vyslaní Petra na zahraničné štúdium v anglicky hovoriacej krajine uplatnilo zodpovedajúce pravidlo.

Peter{ (je_prvkom: ŠTUDENT_MFF)	Peter{ (je_prvkom: ŠTUDENT_MFF)
...	...
(znalosť_cudzích_rečí: NEZISTENÝ)	(znalosť_cudzích_rečí: angličtina)
...}	...}

²⁹ Prvý z uvedených prípadov nemá bezprostredný súvis s dedením. Zaoberáme sa s ním podrobne až v kontexte produkčných pravidiel.

Nech je to napr. pravidlo (v predpoklade je uvedená iba podmienka, ktorá je v tomto príklade predmetom nášho záujmu):

AK

... & (Peter.znalost'_cudzich_reči=angličtina) & ...

TAK

(Peter.kvalifikácia_pre_zahraničné_štúdium: vyhovujúca)

Interpretácia pravidla by vyžadovala poznať hodnotu rubriky *Peter.znalost'_cudzich_reči*. Keďže vo východnom rámci hodnotou rubriky je NEZISTENÁ, spustí sa AP démon. Rubrika nemá špecifikovanú vlastnú AP, preto sa uplatní stratégia dedenia hodnoty z korešpondujúcej rubriky nadradeného rámca. Výsledkom je vyššie uvedený rámec z pravej strany.

Tento triviálny a extrémne jednoduchý príklad slúži len na ilustráciu toho, že **hodnota vlastnosti sa dedí iba pri referencovaní**.

Hoci dedenie hodnoty vlastnosti z nadradeného (všeobecnejšieho) rámca, t.j. **zhora-nadol** je veľmi užitočná štandardná stratégia, nie je vždy žiadúca. Napr. keď by *angličtina* bola hodnotou vlastnosti *znalost'_cudzich_reči* v rámci ŠTU-DENT_MFF, zrejme by nebolo užitočné túto hodnotu automaticky dediť do všetkých I-OBJEKTov tejto triedy. Problematikou dedenia sa podrobnejšie zaoberajú aj nasledujúce články.

Dedenie zhora nadol

3.4.2 Dedenie vlastností

Dedenie vlastnosti

Pre štandardnú stratégiu dedenia **vlastností** platí spravidla nasledujúci postup:

Vlastnosť sa dedí z nadradeného rámca do podradeného rámca, nie však z I-OBJEKTu do S-OBJEKTu, len čo

☞ **I-OBJEKT alebo T-OBJEKT je priradený nadradenému T-OBJEKT-u s definovanými**

vlastnosťami,
☞ **sa nadradenému T-OBJEKTu priradí nová vlastnosť.**

Uplatňovaný postup

Aplikuje sa však pritom pravidlo:

Ak rámec, do ktorého sa dedí, obsahuje už vlastnosť/rubriku, ktorá je predmetom dedenia z nadradeného rámca (jedná sa o totožné identifikátory, tak sauvedená stratégia sa neuplatní.

a uplatňované ohraničenie

Dedenie už jestvujúcej vlastnosti je jednak zbytočné a jednak môže spôsobiť stratu informácie o dôvode jej pôvodného definovania a o potenciálnych dôsledkoch, ktoré by mohli z takej zmeny vyplývať (týka sa to najmä dynamických zmien väzieb medzi rámcami, o čom sa pojednáva v nasledujúcom).

Keďže uvedená štandardná stratégia dedenia nemusí vyhovovať všetkým aplikáciám, je žiaduce poskytnúť tvorcom BZ prostriedky, ktoré mu umožnia stratégiu podľa potreby modifikovať.

Účinok štandardného postupu možno demonštrovať na elementárnom príklade:

Nech T-OBJEKT s identifikátorom **ŠTUDENT** nemá spočiatku definované žiadne vlastnosti.

Máme teda rámec

ŠTUDENT {}.

Nech rámec s názvom Peter je jeho podriadený I-OBJEKT

Peter {...(je_prvkom: ŠTUDENT)... }

Pokiaľ tento rámec nebude mať definované žiadne vlastnosti, nebude ich mať ani po začlenení do nadradeného rámca **ŠTUDENT**. Akonáhle sa však v rámci **ŠTUDENT** definuje vlastnosť, pridáme povedzme (*prospech*), rámec nadobudne tvar

ŠTUDENT {(prospech):}

a táto vlastnosť sa následne dedení do rámca Peter, čím nadobudne tvar

Peter {...(je_prvkom: ŠTUDENT), (prospech):...}

3.4.3 Dedenie metavlastností

Dedenie metavlastností

Analogicky k stratégii dedenia *vlastností* jestvujú aj stratégie dedenia *metavlastností*. Štandardne platí zásada:

Metarubriky (metavlastnosti) sa dedia iba zhora nadol a iba vtedy, keď je to potrebné, t.j. keď sa zisťuje ešte nezistená hodnota referencovanej rubriky a k tomu je to potrebné.

Uplatňovaný postup

Príklad: Majme (nadradený) rámec reprezentujúci T-OBJEKT, v ktorom pre vlastnosť *znalosť_cudzích_rečí* je definovaná metarubrika a v nej iba **AP**

ŠTUDENT_UK {...,([AP: dedenie, vyhľadanie, predpoklad: angličtina] znalosť_cudzích_rečí: NEZISTENÝ),...}

Nech jemu podriadený rámec má pred dedením uvažovanej metavlastnosti nasledovnú podobu

**ŠTUDENT_MFF {...,(je_podtriedou: ŠTUDENT_UK),...
..., (znalosť_cudzích_rečí: NEZISTENÝ),...}**

a rámec reprezentujúci I-OBJEKT, ktorý je jeho inštanciou, má tvar

**Peter {(je_prvkom: ŠTUDENT_MFF),...
..., (znalosť_cudzích_rečí: NEZISTENÝ),...}**

Akonáhle sa vyskytne referencovanie rubriky *Peter.znalosť_cudzích_rečí* dôjde k dedeniu zodpovedajúcej metarubriky a rámec *Peter* sa upraví takto

**Peter {(je_prvkom: ŠTUDENT_MFF),...
...,([AP: dedenie, vyhľadanie, predpoklad: angličtina] znalosť_cudzích_rečí: NEZISTENÝ),...}**

Postup, ktorý sa uplatňuje pri dedení metarubrik demonštrujeme na príklade **AP/AZ**:

- ☞ Ak v metarubrike danej rubriky jestvuje AP/AZ, tak sa uplatňuje ich obsah.
- ☞ Inak sa AP/AZ dedí z niektorého nadradeného rámca.
- ☞ Ak žiadny z nadradených rámcov nemá AP/AZ deklarované, alebo keď dedenie je zakázané, tak sa postupuje podľa štandardných AP/AZ *generickej vlastnosti* (napr. pre priority zdrojov: odvodenie, dedenie zhora, dedenie zdola).
- ☞ Ak sa ani takto hodnotu nepodarí zistiť, tak sa predkladá používateľovi dotaz.

Štandardný postup dedenia metarubrik sleduje postupnosť

Postup dedenia
AP a AZ metód

NADTRIEDA → PODTRIEDA → I-OBJEKT → S-OBJEKT.

Tak ako v prípadoch dedenia **hodnôt** a **vlastností**, aj pri dedení **metavlastností** sa štandardný postup môže globálne (pre BZ, resp. ES) zmeniť, alebo sa mení postup lokálne iba pre konkrétne rubriky.

V metarubrikách **nadradených rámcov** je prípustné

- ☞ vyznačiť **súkromné (chránené)**, t.j. **nedediteľné akcie** (metavlastnosť nadradeného rámca)
- ☞ a tiež určiť, že na vyššej (nadradenej) úrovni sa vykonávajú len súkromné akcie, kým dediteľné (verejné) slúžia iba na dedenie (vyňatie pred zátvorku).

Metarubriky pripúšťajú tiež špecifikovanie možnosti **spájať lokálne** a **zdedené AP/AZ**. Príkladom je povedzme *postupnosť lokálne špecifikovaných akcií, za ktorými nasleduje postupnosť zdedených akcií a napokon postupnosť ďalších lokálne špecifikovaných akcií*.

Možnosť
modifikácií
štandardných
metód dedenia

Súkromné, t.j.
nedediteľné
akcie

3.4.4 Stratégie dedenia

Stratégie
dedenia

V závislosti na typoch riešených problémov z rôznych aplikačných oblastí sa môžu na stratégie dedenia (metavlastností, vlastností a ich hodnôt) klásť líšiace sa požiadavky. Preto by tvorcovia prostredí pre tvorbu ES mali poskytovať dostatočne všeobecné štandardné stratégie dedenia tak, aby boli vyhovujúce čo najširšiemu spektru aplikácií. Spravidla však, hoci sú dômyselne navrhnuté, nevyhovujú rôznym špecifickým požiadavkám. Preto by prostredie tvorby ES malo umožňovať

jednoduché prispôsobovanie štandardných stratégií konkrétnym potrebám jednotlivých kategórií aplikácií alebo v rozsahu BZ individuálnym špecifickým požiadavkám.

Štandardné stratégie dedenia vlastností, hodnôt vlastností, metavlastností a špecifickej vlastnosti **GLOBAL** z T-OBJEKTu do T-OBJEKTu, z T-OBJEKTu do I-OBJEKTu a z I-OBJEKTu do S-OBJEKTu prehľadne uvádza nasledujúca tabuľka (zodpovedá pomerne často uplatňovanému spôsobu dedenia)

Vlastnosť zvaná
GLOBAL

	T-OBJ → T-OBJ	T-OBJ → I-OBJ	I-OBJ → S-OBJ
vlastnosť	hneď	hneď	nededí sa, v špecifických prípadoch prípustné
hodnota	keď je potrebné	keď je potrebné	keď je potrebné
metavlastnosť	keď je potrebné	keď je potrebné	keď je potrebné
GLOBAL	nededí sa	nededí sa	nededí sa

Na jednoduchom príklade sa dá demonštrovať, že stratégia zohľadňujúca iba dedenie "zhoranadol" a nepovoľujúca dedenie z I-OBJEKTu do S-OBJEKTu, nie vždy vyhovuje. Napríklad, keď máme knihu napísanú v určitom jazyku, zrejme sú aj jej jednotlivé kapitoly (S-OBJEKTy) napísané v tomto jazyku, ba aj jej jednotlivé strany a odseky (prirodzene, v špeciálnej literatúre jestvujú výnimky). Preto by malo byť umožnené dediť známu **hodnotu** vlastnosti "jazyk" knihy, kapitoly, či jej konkrétnej strany oboma smermi - **nadol aj nahor**. Dedenie **vlastností** z I-OBJEKTu do S-OBJEKTu je užitočné iba vtedy, keď časti objektu majú rovnaké vlastnosti ako celý objekt. (Dá sa to ilustrovať na nasledujúcom príklade: Majme kábel tvorený viacerými druhmi vodičov. Evidentne by nemalo zmysel dediť hodnotu izolácie kábla do izoláčnych vlastností jednotlivých vodičov.)

Dedenie zdola
nahor

Vývojové prostredia tvorby ES by mali umožňovať tvorcovi BZ prehľadným a jednoduchým spôsobom špecifikovať globálne aj špecifické stratégie dedení.

3.4.5 Vlastnosť **GLOBAL** a súvisiace dedenie

Poslanie
vlastností
GLOBAL

Predstavme si reprezentované (T-,I-,S-)OBJEKTy ako napr. ŠTUDENT, Peter, AUTO, Octávia_BA_1122_BC, STOLIČKA, CHOROBA, TBC_PLÚC, ORGOVÁN, Naša_tohoročná_dovolenka a pod. Bezohľadu na druh objektu, zodpovedajúci *term* je špecifikáciou vlastností reprezentovaného.

Jednou z veľmi frekventovaných činností v procese riešenia problémov je **rozpoznávanie (identifikácia)** možnosti úplne alebo čiastočne stotožniť (na symbolovej úrovni) niektorý(é) z reprezentovaných objektov s reálne pozorovaným alebo skúmaným, či s takým, ktorý je obsiahnutý v mentálnom modeli používateľa ES (pozri v predošlom uvedený príklad s 'tigrom'). K tomu je prirodzene potrebné zisťovať fakty, t.j. hodnoty vlastností, alebo určité hodnoty predpokladať. Zhromažďovanie faktov a predpokladov vedie k naplňovaniu rubrik obsahom. To však je len nutnou, ale nie postačujúcou podmienkou k následnej identifikácii objektu. Je ešte potrebné

rozhodnúť koľko a ktoré hodnoty vlastnosti skúmaného fenoménu musia byť prítomné,

resp. neprítomné (a tiež v akej racionálnej postupnosti je ich potrebné zisťovať či predpokladať), **aby bolo možné pripustiť alebo zamietnúť zhodu reprezentovaného s realitou,**

t.j. či je ktosi/čosi ŠTUDENTom, Petrom, AUTOm, Octáviou_BA_1122_BC, STOLIČKOU, CHOROBOU, TBC_PLÚC, ORGOVÁNOM, Našou_tohoročnou_dovolenkou a pod.

Rozhodovanie je založené na zisťovaní splnenia určitých **kritérií**, ktoré sú vlastne **propozície**. Tvoria prvky **asertívnych znalostí**, ktoré sú reprezentované *produkčnými pravidlami*. Tým, že vlastnosť **GLOBAL** vystupuje v úlohe *dôsledku jadra pravidla*, vytvára sa jedna (častá) možnosť nadobúdania jej hodnoty. *V predpoklade jadra sú spravidla referencované povinné a fakultatívne vlastnosti uvažovaného objektu.*

Z rozmanitosti predpokladov pravidiel reprezentujúcich kritéria pre rozpoznávanie rôznych objektov by mali byť zrejmé dôvody, pre ktoré štandardné stratégie dedenia spravidla nepripúšťajú dedenie *hodnoty* rubriky **GLOBAL**. Avšak podľa konkrétnych požiadaviek aplikácií, ktoré majú zvyčajne iba ohraničené poslanie pre určité typy hierarchizovateľných závislostí, je opäť žiaduce pripustiť modifikácie štandardnej stratégie.

Hodnota vlastnosti GLOBAL má povahu propozície hodnoty

Hodnota rubriky GLOBAL jednotlivých rámcov sa smie a veľmi často sa referencuje v predpoklade produkčných pravidiel. To je práve jeden z významných spôsobov využívania tejto hodnoty na rozhodovanie o následnom riešiacom postupe.

Nasledujúci príklad ilustruje výhodu modifikovania štandardnej stratégie dedenia povolením dediť *zdola-nahor*. Týka sa prípadu, keď je potrebné rozhodnúť, či je použiteľné firemné auto. Nech určitá firma je vlastníkom povedzme troch áut. Reprezentujeme túto skutočnosť takto

FIREMNÉ_AUTO{(je_podtriedou: AUTO),...,([DT: **pojazdné**, **nepojazdné**] **GLOBAL: NEZISTENÉ**),...}

Firemné_auto_i {(je_prvkom: FIREMNÉ_AUTO),...,([DT: **pojazdné**, **nepojazdné**] **GLOBAL: NEZISTENÉ**),...} kde i nahrádza hodnotu 1,2,3.

Časté referencovanie

Ak aspoň jeden z uvedených I-OBJEKTov nadobne v rubrike **GLOBAL** hodnotu **pojazdný**, je zrejmé, že firma má k dispozícii **pojazdné auto**. Tento výrok je *propozíciou*, ktorú možno pomerne priamočiaro reprezentovať *produkčným pravidlom*. Jeho interpretácia umožňuje rozhodnúť o použiteľnosti firemného auta. *V predpoklade jadra* takého pravidla by sa vyskytovala podmienka **FIREMNÉ_AUTO.GLOBAL=pojazdné**. Vyhodnotenie podmienky prirodzene vyžaduje zistiť hodnotu rubriky **FIREMNÉ_AUTO.GLOBAL**. Dá sa to viacerými spôsobmi. Jednou z nich je uplatnenie vhodne zvolenej stratégie dedenia *zdola-nahor*.

Taká stratégia má dobré teoretické podložie:

Ak hodnota rubriky GLOBAL v nadradenom (všeobecnejšom) objekte má propozíciu hodnotu PRAVDA, tak aspoň jeden z podradených (špecifickejších) objektov by mal mať vo svojej rubrike GLOBAL tiež propozíciu hodnotu PRAVDA. A naopak (komplementárny výrok): Ak hodnota rubriky GLOBAL aspoň v jednom podradenom (špecifickejšom) objekte má propozíciu hodnotu PRAVDA, tak hodnota rubriky GLOBAL v nadradenom (všeobecnejšom) objekte má propozíciu hodnotu PRAVDA.

Pribípa tejto stratégie sa pomerne často uplatňuje pri tvorbe BZ.

V nami uvažovanom prípade adekvátom propozície hodnoty **PRAVDA** ("**pozitívna**" hodnota) je **pojazdné** (je **PRAVDA**, že je **pojazdné**). Uplatnenie nastaviteľnej stratégie dedenia hodnoty rubriky **GLOBAL** (*pojazdnosť auta*) *zdola-nahor* zabezpečuje, že "pojazdnosť" aspoň jedného z konkrétnych áut priradí túto hodnotu do rubriky **GLOBAL** aj v nadradenom rámci. "**Negatív-na**" hodnota (*nepojazdné*) sa dedí iba vtedy, keď ju nadobne rubrika **GLOBAL** všetkých podriadených objektov.

Dá sa vysloviť určité zovšeobecnenie. Lokálne ohraničená stratégia dedenia hodnoty rubriky **GLOBAL propozíciu** typu je často výhodná, keď podlieha nasledovným (nie však výlučne týmto) pravidlám:

- ak nadradený rámec obsahuje v rubrike **GLOBAL** negatívnu hodnotu, tak túto hodnotu dedia zodpovedajúce rubriky všetkých podradených rámcov,
- ak nadradený rámec obsahuje v rubrike **GLOBAL** pozitívnu hodnotu, tak sa pripúšťa v

Stratégia dedenia hodnoty rubriky GLOBAL

- **podriadených rámcoch zisťovanie hodnoty GLOBAL, nie však jej dedenie, ak aspoň jeden podriadený rámec obsahuje v rubrike GLOBAL pozitívnu hodnotu, tak sa táto hodnota dedí zdola-nahor,**
- **ak všetky podriadené rámce obsahujú v rubrike GLOBAL negatívnu hodnotu, tak sa táto hodnota dedí zdola-nahor.**

Nezávisle od lokálnej alebo globálnej špecifikácie tejto stratégie dedenia hodnoty rubriky **GLOBAL**, stratégiu je nevyhnutné doplniť ešte o prípad, keď **GLOBAL** má hodnotu **NEZNÁMA**. O postavení a účinku tejto hodnoty je pojednané v súvislosti s tabuľkami pravdivostných hodnôt - pozri kapitolu 4.

Pre dedenie vlastností **GLOBAL** a jej zodpovedajúcej metavlastnosti **neplatia** štandardné stratégie. Plynie to z povahy tejto rubriky. Prípustnosť dedenia hodnoty má byť nastaviteľná.

Uplatnenie stratégie dedenia hodnoty GLOBAL

3.4.6 Stratégie viacnásobného dedenia

Viacnásobné dedenie

Dedenie z viacerých nadradených rámcov zodpovedá prirodzeným vzťahom v realite. Ten istý objekt môže mať niekoľko nadradených OBJEKTOV rôzneho typu³⁰. Z toho vyplýva, že cieľ dedenia - rubrika určitého rámca - môže mať viacero zdrojov, z ktorých môže dediť vlastnosť, hodnotu, metavlastnosť a z nej metódu. Aby nedošlo k **nedeterminizmu (konfliktu)**, musia sa určit stratégie uprednostňovania výberu nadradených/podradených rámcov.

Do úvahy prichádzajú štyri základné stratégie viacnásobného dedenia, ktoré vyplývajú z možnosti kombinovať

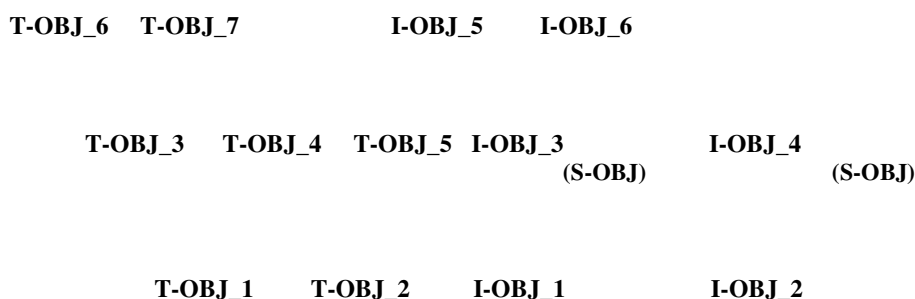
Uprednostňovanie	pri prehľadávaní do
T-OBJEKTY pred I-OBJEKTIAMI	šírky
	hĺbky
I-OBJEKTY pred T-OBJEKTIAMI	šírky
	hĺbky

Ilustračný príklad

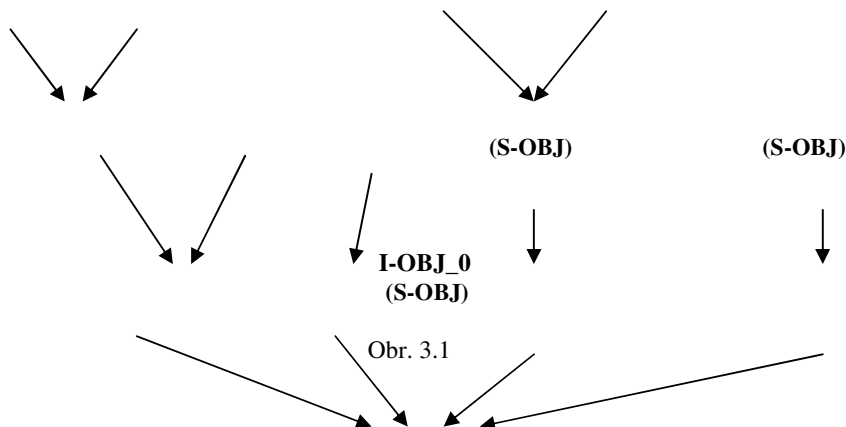
Účinok jednotlivých stratégií možno demonštrovať na podgrafe z obr. 3.1. (Dedenie z I-OBJEKTu do I-OBJEKTu je potrebné ponímať tak, že dediaci objekt je vlastne S-OBJEKT, ktorý je však zároveň aj I-OBJEKTom vzhľadom na jeho nadradený T-OBJEKT.) Dedí sa do **I-OBJ_0**.

Podľa jednotlivých stratégií dostávame nasledovné postupnosti pre dedenie

- (a) T-OBJ_1, T-OBJ_2, I-OBJ_1, I-OBJ_2, T-OBJ_3, T-OBJ_4, T-OBJ_5, I-OBJ_3, I-OBJ_4, T-OBJ_6, T-OBJ_7, I-OBJ_5, I-OBJ_6
- (b) T-OBJ_1, T-OBJ_3, T-OBJ_6, T-OBJ_7, T-OBJ_4, T-OBJ_2, T-OBJ_5, I-OBJ_1, I-OBJ_3, I-OBJ_5, I-OBJ_6, I-OBJ_2, I-OBJ_4
- (c) I-OBJ_1, I-OBJ_2, T-OBJ_1, T-OBJ_2, I-OBJ_3, I-OBJ_4, T-OBJ_3, T-OBJ_4, T-OBJ_5, I-OBJ_5, I-OBJ_6, T-OBJ_6, T-OBJ_7
- (d) I-OBJ_1, I-OBJ_3, I-OBJ_5, I-OBJ_6, I-OBJ_2, I-OBJ_4, T-OBJ_1, T-OBJ_3, T-OBJ_6, T-OBJ_7, T-OBJ_4, T-OBJ_2, T-OBJ_5



³⁰ Príklad: 'Peter' je individuum, ktorý môže súčasne patriť do viacerých tried, povedzme MUŽ, ŠTUDENT, BRAT, HUDOBNÍK, ŠPORTOVEC atď. Z toho plynie možnosť viacnásobného dedenia z nadradených rámcov. Obdobne 'TBC_PLÚC' je podtriedou tried 'PLÚCNE_OCHORENIE' a 'TUBERKULÓZNE_OCHORENIE' a preto získava vlastnosti oboch nadradených entít.



Pri dedení zdola-nahor sa uplatňujú rovnaké stratégie a postup dedenia je analogický ako pri dedení zhora-nadol.

Uvedené základné stratégie dedenia sa dajú kombinovať s ďalšími metódami. Napríklad vo väzbách **nadradených/podradených rámcov rovnakej úrovne sa uprednostňujú rámce, ktorých rubrika má najvyššiu prioritu**

3.5 Dynamické štruktúry rámcov a ich väzieb

Statická a dynamická realita

Doteraz uvedené prostriedky reprezentácie termových znalostí predpokladali svet (realitu), v ktorom všetky objekty a ich vzájomné väzby sú vopred dané a známe. Jednalo sa teda o svet **statický**, v ktorom sa mohli dynamicky meniť iba priority získavania hodnôt v jednotlivých rubrikách. Problémy riešené pomocou ES sa však môžu týkať aj **dynamickej** skutočnosti - takej, v ktorej

- vznikajú, zanikajú a menia väzby medzi objektami,
- vznikajú a zanikajú objekty a ich triedy.

Najväčšie problémy, ktoré súvisia s reprezentáciou dynamického spočívajú v tom, že v čase zostavovania BZ nemôžu jeho tvorcovia

úplne **predvídať všetky** (vznikajúce, meniace sa a zanikajúce) **objekty, ich vlastnosti a ich väzby, s ktorými v čase používania ES bude potrebné narábať**.

S tým tesne súvisí aj problematika (podrobnejšie sa ňou zaoberáme ešte v závere 4. kapitoly)

- ◆ referencovania externých zdrojov údajov a programov, ktoré vopred, v čase tvorby BZ, nie sú známe,
- ◆ vytvárania produkčných pravidiel) ktoré by vo svojich štruktúrach (najmä podmienkach a dôsledkoch) mali mať možnosť referencovať aj potenciálne v budúcnosti vznikajúce, alebo inak začlenené objekty.

Rozvinuté vývojové prostredia a nimi vytvárané ES preto

okrem vopred definovanej - a teda kompilovanej - BZ, pripúšťajú aj v čase činnosti systému vytváranie referencovateľných dynamických objektov a ich väzieb ("runtime" objekty, väzby), ich modifikovanie aj rušenie v súlade s potrebami, ktoré systém v priebehu svojej činnosti situačne detekuje.

Tvorca BZ potom vytvára reprezentáciu len tých častí reality, ktoré sa **vždy môžu použiť**. **Objekty/vzťahy, ktorých existencia je iba situačne podmienená, sa generujú iba v prípade potreby.** (Výrazné zníženie nárokov na pamäť!)

Dynamické (T-,I-,S-)objekty a väzby sa vytvárajú **povelmi podporného jazyka**, ktoré poskytujú vývojové prostredia. Dynamicky vytvorené/zrušené objekty a väzby sa vo všeobecnosti môžu vyskytovať vo všetkých zložkách produkčných pravidiel, ako aj v akciách, ktoré sa týkajú **AP a AZ**.

Potreba prostriedkov umožňujúcich odkazovanie na dynamické

objekty

Novovytvorený dynamický objekt - rámec - môže svoje vlastnosti a metavlastnosti nadobudnúť dedením, keď sa začlení medzi už existujúce rámce. Hoci by mohol byť iniciovaný aj bez vlastností a nadradených či podradených objektov, je otázne, či by to malo prakticky význam. Dynamické väzby – ich **dynamické vytváranie aj rušenie** – sú prípustné aj v rámci vzájomných vzťahov medzi statickými a dynamickými (T-,I-,S-)objektami.

Môže byť výhodné, keď tvorca BZ dokáže zabezpečiť také **dynamické zmeny siete rámcov, ktoré využívajú možnosti rôznych ciest dedenia**. Majme ilustračný príklad, v ktorom uvažujeme študentov špecializácie matematickej analýzy (MA) a umelej inteligencie (UI):

Povelový jazyk tvorby dynamických objektov

ŠTUDENT_MA{ (je_podtriedou: ŠTUDENT_MFF), ..., ([] prospech_matematická_analýza:), ([] prospech_diferenciálne_rovnice:), ([] algebra:), ... }	ŠTUDENT_UI{ (je_podtriedou: ŠTUDENT_MFF), ..., ([] prospech_umeľá_inteligencia:), ([] prospech_expertné_systémy:), ([] algebra:), ... }
Karol{ (je_prvkom: ŠTUDENT_MA), ..., ([] prospech_matematická_analýza: výborný), ([] prospech_diferenciálne_rovnice: veľmi_dobrý), ([] algebra: veľmi_dobrý), ... }	Peter{ (je_prvkom: ŠTUDENT_MA), ..., ([] prospech_matematická_analýza: výborný), ([] prospech_diferenciálne_rovnice: veľmi_dobrý), ([] algebra: výborný), ... }
Paľo{ (je_prvkom: ŠTUDENT_UI), ..., ([] prospech_umeľá_inteligencia: veľmi_dobrý), ([] prospech_expertné_systémy: výborný), ([] algebra: veľmi_dobrý), ... }	Míro{ (je_prvkom: ŠTUDENT_UI), ..., ([] prospech_umeľá_inteligencia: výborný), ([] prospech_expertné_systémy: veľmi_dobrý), ([] algebra: výborný), ... }

Ak by sa v tejto situácii zrušilo zaradenie Petra z T-OBJEKTu ŠTUDENT_MA napr. príkazom tvaru

Ilustrácia

Zruš_hodnotu_rubriky: Peter.je_prvkom: ŠTUDENT_MA

a následne príkazom

Vlož_hodnotu_rubriky: Peter.je_prvkom: ŠTUDENT_UI

nahradí sa pôvodná hodnota ŠTUDENT_MA novou hodnotou, čo znamená preradenie príslušného rámca do inej hierarchickej väzby v rozsahu daného systému rámcov. Následne, v závislosti na práve platnej stratégii dedenia, vznikne (v nami uvažovanom prípade) rámec

Peter

{(je_prvkom: ŠTUDENT_UI),...,(*[] prospech_matematická_analýza: výborný), (*[] prospech_diferenciálne_rovnice: veľmi_dobrý), (*[] algebra: výborný), ([] prospech_umeľá_inteligencia: NEZISTENÝ), ([] prospech_expertné_systémy: NEZISTENÝ), ([] algebra: NEZISTENÝ),}

Ako je vidno, pôvodné rubriky a ich hodnoty, ktoré platili pred preradením, zostávajú

zachované, nie sú však už aktuálne a je to (v danom ilustračnom príklade) vyznačené symbolom '*'; t.j. **nie sú referencovateľné Peter.algebra** po preradení odkazuje na novododenú rubriku rovnakého mena a jej hodnota bude NEZISTENÝ.

V prípade preradenia objektu z jednej do inej triedy sa spravidla najprv zruší pôvodná väzba a následne sa vytvorí nová. Dynamické zmeny zaradenia objektu v priebehu činnosti ES, t.j. zmeny priradení vlastností, spravidla nemenia hodnotu ničoho, čo už bolo na základe dedenia v predošlom získané. To je veľmi dôležité opatrenie. Jeho potreba vyplýva z potenciálneho odstúpenia od prebiehajúceho riešenia a teda z potreby revokovať nové rubriky, a ich hodnoty, a vrátiť sa k niektorému z predošlých krokov riešenia.

Následné udalosti v priebehu odvodzovania ovplyvňujú vždy práve aktuálna štruktúra rámca a obsah jeho rubriek.

Pre dedenie vytvorených dynamických objektov/väzieb platia nasledovné pravidlá:

- **vlastnosti (rubriky) sa dedia ihneď, v predošlom zdedené vlastnosti zostávajú zachované,**
- **hodnota vlastnosti sa dedí iba keď je to potrebné a keď stratégia dedenia to povoľuje, dedením sa pritom nadobúda hodnota, ktorá zodpovedá platným väzbám v okamihu referencovania hodnoty,**
- **metarubriky sa dedia len keď je to potrebné, t.j. keď sa referencuje hodnota vlastnosti a tá je NEZISTENÁ (dynamické objekty nemajú lokálne definované metarubriky), pričom sa uplatňujú tie stratégie dedenia, ktoré sú platné v okamihu keď k dedeniu dochádza.**

Treba si uvedomiť, že uchovávanie dynamických objektov a ich väzieb v BZ vo všeobecnosti nemá význam. Veď tú istú BZ môžu používať viacerí a dokonca aj ten istý jedinec ju zvyčajne používa na riešenie rôznych problémov. To je dôvod, pre ktorý životnosť všetkých dynamických objektov je daná aktuálnou dobou riešenia daného problému. Po ukončení práce so systémom sa všetky dynamické objekty a väzby automaticky rušia. V prípade, že sa predpokladá opakovaná (pokračujúca) práca so systémom, v ktorom boli vytvorené dynamické entity, je potrebné danú BZ a BF archivovať.

Pravidlá dedenia v dynamických objektoch

Uzatvárame: Reprezentácia znalostí **termového typu** vytvára určitý druh **dynamicky meniteľného priestoru problému (akýsi "terén")** tvorený

- **objektami aplikačnej (problémovej) oblasti so špecifickými vlastnosťami a predpismi (metódami) na**
 - **získavanie ich hodnôt,**
 - **vykonávanie akcií vyplývajúcich z ich zmien,**
- **vzájomnými (viacrozmernými) vzťahmi medzi objektami implikujúcimi rôznorodé explicitne nevyjadrené podmienosti,**

ktorý umožňuje v rámci dynamického ovplyvňovania vlastnej štruktúry **produkovať v jeho rozsahu účelné (inteligentné) správanie.**

Už aj čiastočne vytvorený systém reprezentácie termových a asertívnych znalostí umožňuje v určitom rozsahu generovať účelné správanie ES. Reálna použiteľnosť BZ je prirodzene podmienená dovŕšovaním rozsahu a úplnosťou reprezentovaných znalostí, zdokonaľovaním reprezentácie a odstraňovaním, resp. ošetrovaním, protirečivostí (nekonzistentností), ktoré sú v ňom obsiahnuté. V princípe sa však jednotlivé časti BZ môžu vytvárať, dopĺňovať a modifikovať postupne bez vplyvu na funkcie inferenčného mechanizmu.

Vytváranie BZ je evolučný proces, reprezentáciu znalostí termového typu možno uskutočňovať postupne.

Kapitulu uzatvárame triviálnym výrokom: FORMALIZMUS REPREZENTÁCIE TERMOVÝCH AJ ASERTÍVNYCH ZNALOSTÍ MUSÍ BYŤ KOMPATIBILNÝ S TÝM, KTORÝ INFERENČNÝ MECHANIZMUS PREDPOKLADÁ.

Reprezentácia znalostí termového typu vymedzuje aj priestor riešenia problému

4. Reprezentácia asertívnych znalostí

(Produkčné pravidlá a procedúry)

.Druhy symbolovej reprezentácie asertívnych poznatkov

V systémoch založených na znalostiach - ES sú ich špecializáciou - najčastejším formalizmom reprezentácie znalostí **asertívneho** typu sú **produkčné pravidlá**. Tie tvoria základňu pre procesy vedúce k správaniu systému, ktoré sa jeho pozorovateľovi javí ako **propozičný prejav** znalostí.³¹

Vývoj znalostných a expertných systémov vedie však k rastúcemu používaniu aj iných formalizmov reprezentácie asertívnych znalostí. Príkladmi toho sú reprezentácia znalostí

- analytickými **modelmi - najmä kvalitatívnymi**,
- formou **rovníc a nerovníc** vyjadrujúcich **vzťahy** implikujúce závislosti medzi vlastnosťami entít a **ohraničenia** vzťahujúce sa na ich hodnoty,
- týkajúcich sa **vzájomných vzťahov** entít **symbolovo reprezentovaných vektormi alebo bitmapami**.

Sú to formalizmy, ktoré z hľadiska *expresivity, odvodzovacej a výpočtovej účinnosti* majú mnohé výhody - možno ich vnímať aj ako nositeľov implicitných produkčných pravidiel³².

V tejto a nadväzujúcich kapitolách, pokiaľ neuvedieme inak, ohraničíme pozornosť len na produkčné pravidlá.

Pre rané typy ES bolo príznačné, že im zodpovedajúce BZ obsahovali len asertívne znalosti reprezentované výlučne produkčnými pravidlami. Korešpondovalo to s hypotézou, podľa ktorej tieto pravidlá sú *považované za základňu psychologicky orientovaných teórií vypočítateľnosti*³³.

ES aj s takto zjednodušenou reprezentáciou znalostí dosahovali pozitívne výsledky, ktoré upútali pozornosť najmä v súvislosti s riešením jednoduchých a nerozsiahlych problémov.³⁴ Zvyšovaná náročnosť a/alebo rozsah riešených problémov však jednoznačne odhalil, že uplatňovanie iba asertívnych znalostí (v podobe ich explicitného vyjadrovania) bez reprezentácie znalostí termového typu nezabezpečí efektívnosť a teda ani akceptovateľnosť odvodzovacích procesov.

Dá sa o tom presvedčiť napríklad porovnaním reprezentácie **hierarchicky začlenených entít prostredníctvom produkčných pravidiel a hierarchiou rámcovej reprezentácie**. Expresivita aj odvodzovacia účinnosť je výrazne vyššia pri uplatnení rámcovej reprezentácie termov.

Objekty majú mnohé vlastnosti, väzby a úlohy, ktorých reprezentácia iba v a-sertívnej podobe - produkčnými pravidlami - je neprípustne neefektívna a pri *lokálnom* vyhodnocovaní (propozičnej interpretácii) predpokladu nie sú ani dobre zohľadniteľné.

Asercie nie sú vždy výhodné

pozadie odvodzovacím - usudzovacím procesom. v ES sú zodpovedajúce procesy realizované **inferenčným mechanizmom (IM)**. Ten **stelesňuje súhrn základných univerzálnych odvodzovacích operácií** (označovaných symbolom Cn v logike), ten teda je **producentom účelného (aj keď nemonotónneho) "pohybu"** v spomenutom **"teréne"** - ako sa to dá vyjadriť metaforou, čo zodpovedá **účelnému správaniu**, ktoré je považované za prejav **typu racionálnej inteligencie**. **Produkčné pravidlá** - deklarácie asertívnych znalostí - tvoria akoby "palivo" takého

³¹ Smithova hypotéza inteligentného správania.

³² Dá sa to jednoducho ilustrovať. Majme napr. hierarchickú väzbu "meď je kov" a vlastnosť "kov je elektrický/tepelný vodič". Takáto reprezentačná štruktúra implikuje produkčné pravidlo "AK je meď kov, TAK je elektrický/tepelný vodič". Iným príkladom je napr. symbolový zápis Ohmovho zákona $U=R \cdot I$ - z neho plynie pravidlo: "AK zvýšime/znížime hodnotu intenzity pri nezmenenej hodnote odporu, TAK sa zvýši/zníži hodnota napätia!". Je to téma, ktorá je predmetom pozornosti v kapitolách venovaných problematike rozvinutých metód inferencie.

³³ Johnson-Laird.

³⁴ Javili sa potvrdením Johnson-Lairdovej hypotézy.

správania. Ich **propozičná interpretácia zodpovedá základným prostriedkom usudzovania.**

Vo všeobecnosti platí, že **explicitná reprezentácia znalostí termového typu**, čo je prostriedkom **začleňovania a hierarchizácie poznatkov**, ako aj **ich uplatňovania v rôznych úlohách aj z rôznych pohľadov**, výrazne obohacuje odvodzovaciu účinnosť produkčných pravidiel.

BZ, transformujú sa do úspornejšej a výpočtovo efektívnejšej symbolovej podoby.

Uvedené ilustračné príklady pravidiel sú špecifické tým, že ich predpoklady sú jednoznačne vyhodnotiteľné: ak sú semafory funkčné, tak aspoň jedna z ich lámip svieti, hoci len prerušovane.³⁵ Preto sa ich predpoklad dá jednoznačne kategoricky propozične vyhodnotiť. To ich kvalifikuje na **kategorické pravidlá**. Nimi sa zaoberáme najprv.

Úloha
inferenčného
mechanizmu

Hnacia sila:
produkčné
pravidlá

Väzba medzi asertívnymi a termovými znalosťami spočíva v tom, že podmienky predpokladu pravidiel a dôsledok pravidiel sú odvolávky na hodnoty rubriek rámcov.

Jadro
produkčného
pravidla

Ilustrácie

Produkčné pravidlo je **kategorické** vtedy a len vtedy, keď je jeho

- **predpoklad p** na základe dostupných údajov jednoznačne propozične vyhodnotiteľný, t.j. *podmienkam p*; vystupujúcim v predpoklade je možno jednoznačne priradiť kategorickú propozičnú hodnotu - v našom prípade uvažované hodnoty **NEZISTENÉ, PRAVDA, NEPRAVDA a NEZNÁME**,
- **dôsledok d** je jednoznačne propozične vyhodnotiteľný na základe propozičnej hodnoty, ktorú nadobudol predpoklad; ak by hodnota dôsledku bola odvoditeľná vyhodnotením viacerých odlišných predpokladov, ktorých vyhodnotenie by viedlo líšiacim sa propozičným hodnotám, tak buď prirodzene jestvuje alebo dohovorom sa stanovuje deterministicky predpis určenia jeho výslednej kategorickej hodnoty.

³⁵ Predpokladáme teda, že vieme vždy spoľahlivo zistiť, či lampa svieti, alebo nesvieti, hoci je známe, že napríklad pri určitom sklone dopadu slnečných lúčov nie je jednoduché kategorické rozhodovanie.

Kategorické
produkčné
pravidlá

AK sa približuješ ku križovatke, cez ktorú prechod je riadený semaforom A je rozsvietená iba jediná lampa semafóra A farba svietiacej lampy semafóra je <i>zelená</i> , A nemáš v ceste prekážku, TAK pokračuj v jazde. (P 1.1)
AK sa približuješ ku križovatke, cez ktorú prechod je riadený semaforom A je rozsvietená iba jediná lampa semafóra A farba svietiacej lampy semafóra je <i>červená</i> , TAK pred križovatkou zastav. (P 1.2)
AK sa približuješ ku križovatke, cez ktorú prechod je riadený semaforom A je rozsvietená iba jediná lampa semafóra A farba svietiacej lampy semafóra je <i>oranžová</i> , TAK priprav sa na zastavenie. (P 1.3)
AK sú rozsvietené práve dve lampy semafóra A farba jednej svietiacej lampy semafóra je <i>oranžová</i> A farba druhej svietiacej lampy semafóra je <i>červená</i> TAK AK stojíš na križovatke TAK priprav sa na pokračovanie v jazde. INAK AK sa približuješ ku križovatke A nemáš v ceste prekážku TAK pokračuj v jazde. (P 1.4)

Tab. 1

4.1 Štruktúra (jadra) produkčných pravidiel a ich interpretácia

Predpokladová časť	Predpokladová časť jadra produkčného pravidla je tvorená podmienkami, ktoré sú vzájomne viazané logickými operátormi. Symbolovo píšeme
Predpoklad a podmienky	$P \equiv P_1, P_2, \dots, P_n$ kde vo všeobecnosti namiesto čiarky možno dosadiť vhodný operátor (logickú spojku). V nadväzujúcom, pokiaľ to nevedieme inak, uvažujeme dôsledky propozičného typu podmienky a vzájomne viazané iba operátorom konjunkcie (A, resp. AND operátor, symbolovo tiež &, ^). Je to prípustné, lebo ľubovoľná formula tvoriaca predpoklad jadra pravidla sa dá transformovať do dizjunktívnej normálnej formy . A tak možno pravidlo, ktorého predpoklad je tvorený <i>k</i> dizjunktami, rozložiť na <i>k</i> samostatných pravidiel s totožným dôsledkom. Predpoklad takých pravidiel je tým tvorený iba z konjunkcií podmienok. Propozičná hodnota predpokladu je jednoznačne určená propozičnými hodnotami podmienok, ktoré ho vytvárajú. Každý dizjunkt tvorený konjunkciami v ňom sa vyskytujúcich podmienok, ktorý po vyhodnotení nadobudne propozičnú hodnotu PRAVDA , spôsobuje v takom prípade priradenie tejto hodnoty aj dôsledku.
Dizjunktívna normálna forma	

Nasledujúca tabuľka propozičných hodnôt konjunkcie dvoch podmienok p_i a p_j je triviálnym zovšeobecnením svojej boolovskej obdoby:

$p_i \wedge p_j$	NEZISTENÉ	PRAVDA	NEPRAVDA	NEZNÁME
NEZISTENÉ	NEZISTENÉ	NEZISTENÉ	NEPRAVDA	NEZISTENÉ
PRAVDA	NEZISTENÉ	PRAVDA	NEPRAVDA	NEZNÁME
NEPRAVDA	NEPRAVDA	NEPRAVDA	NEPRAVDA	NEPRAVDA
NEZNÁME	NEZISTENÉ	NEZNÁME	NEPRAVDA	NEZNÁME

Zovšeobecnenie tabuľky na n podmienok je triviálne. Pri jej procedurálnej implementácii v inferenčnom mechanizme je však dôležité vytvoriť procedúru s minimálnym počtom nevyhnutných operácií (minimálnou výpočtovou zložitou).

4.2 Podmienky v predpoklade pravidla

Podmienky v predpoklade	Podmienka (aj negovaná) - ako propozičný výraz - môže byť tvorená rozmanitými symbolovými štruktúrami:
Druhy podmienok	<ul style="list-style-type: none"> ➤ Najjednoduchší prípad je prostý identifikátor rubriky rámca, povedzme <i>R.r</i>, ktorej <i>dátový typ</i> je propozičná hodnota (PRAVDA, NEPRAVDA, NEZNÁMA). ➤ Identifikátor propozične vyhodnocovanej funkcie, ktorá vráti hodnoty typu PRAVDA, NEPRAVDA, NEZNÁMA. ➤ Numerická relácia – operand1 operátor operand2, v ktorej <ul style="list-style-type: none"> • operand1 je alternatívne <ul style="list-style-type: none"> – identifikátor rubriky rámca, pre hodnotu ktorej platí <i>dátový typ numericky</i>, – numerický výraz, v ktorom sa vyskytuje aspoň jediný identifikátor rubriky rámca s <i>dátovým typom numerickým</i>, – individuálny identifikátor numerickej funkcie alebo ako prvok numerického výrazu, pričom z jej argumentov je aspoň jeden identifikátor rubriky rámca, pričom funkcia vráti numerickú hodnotu, • operand2 je alternatívne <ul style="list-style-type: none"> – konštanta, – alebo niektorá z alternatív ako v prípade operand1, • operátor je alternatívne $>$, $=$, $<$, \neq, \geq, \leq. ➤ Intervalová relácia - operand1 operátor operand2, v ktorej <ul style="list-style-type: none"> • operand1 je alternatívne <ul style="list-style-type: none"> – identifikátor rubriky rámca, pre hodnotu ktorej platí <i>dátový typ numericky</i>, – identifikátor rubriky rámca, pre hodnotu ktorej platí <i>dátový typ numerický interval (vyjadrený dvojicou hodnôt)</i>, – numerický výraz, v ktorom sa vyskytuje aspoň jediný identifikátor rubriky rámca s <i>dátovým typom numerický</i>, – identifikátor numerickej funkcie, ktorého aspoň jeden argument je tvorený identifikátorom rubriky rámca s numerickým oborom individuálnych, resp. dvojíc hodnôt, alebo dvojicu týchto hodnôt, – numerický výraz, v ktorom sa vyskytuje identifikátor numerickej funkcie, ktorého aspoň jeden argument je tvorený identifikátorom rubriky rámca, s
Propozične vyhodnocované funkcie	
Numerické relácie	
Intervalové relácie	

- numerickým oborom individuálnych, resp. dvojíc hodnôt, alebo dvojicu týchto hodnôt,
- operand2 je alternatívne
 - **konštanta**, tvorená dvojicou hodnôt reprezentujúcich interval,
 - **alebo niektorý z prípadov operand1, pokiaľ jej hodnota zodpovedá intervalu,**
- operátor je alternatívne
 - $>$ (dolná_hranica_1 je menšia ako dolná_hranica_2 a zároveň horná_hranica_1 je väčšia ako horná_hranica_2),
 - $=$ (rovnosť oboch hraníc),
 - $<$ (analogicky prípadu $>$),
 - \neq (keď intervaly majú prázdny prienik),
 - \geq (keď intervaly majú jednu spoločnú hraničnú hodnotu a druhá hraničná hodnota intervalu_2 leží v intervale_1),
 - \leq (analogický s predošlým),
 - alebo symboly s významom
 - \in **je prvkom** (hodnota leží vo vnútri intervalu),
 - \cup **hraničí s intervalom** (intervaly majú spoločnú iba jedinú z hraničných hodnôt),
 - \Rightarrow **nasleduje** (keď hraničí a dolná hranica intervalu_1 je rovná hornej hranici intervalu_2, alebo platí relácia \neq , pričom tiež platí, že dolná hranica intervalu_1 $>$ horná_hranica_intervalu_2),
 - \Leftarrow **predchádza** (analogicky ako v predošlom).
- **Množinová relácia - operand1 operátor operand2**, v ktorej
 - operand1 je alternatívne
 - **identifikátor rubriky rámca**, pre hodnotu ktorej platí *dátový typ množina*,
 - **výraz s množinovými operáciami**, v ktorom sa vyskytuje aspoň jediný identifikátor rubriky rámca s *dátovým typom množina*,
 - operand2 je alternatívne
 - **množina**,
 - alebo niektorý z prípadov operand1,
 - operátor je niektorá z relácií (vlastná, nevlastná) **podmnožina**, resp. **nadmnožina**, **rovnosť**, resp. **nerovnosť** množín, **prvok množiny**.
(Analogicky pre zoskupenia "bag" a zoznamy.)
- **Reťazcová relácia - operand1 operátor operand2**, v ktorej
 - operand1 je alternatívne
 - **identifikátor rubriky rámca**, pre hodnotu ktorej platí *dátový typ reťazec*,
 - **výraz s reťazcovými operáciami**, v ktorom sa vyskytuje aspoň jediný identifikátor rubriky rámca s *dátovým typom reťazec*,
 - operand2 je alternatívne
 - **reťazec**,
 - alebo niektorý z prípadov operand1,
 - operátor je niektorá z relácií **je prvkom dret'azca**, **zahrňuje reťazec**, **rovnosť (zhoda)_reťazcov**, **nerovnosť_reťazcov**.
- **Propozičná (logická) relácia - operand1 operátor operand2**, v ktorej
 - operand1 je alternatívne
 - **identifikátor rubriky rámca**, pre hodnotu ktorej platí *propozičný dátový typ*,
 - **výraz s logickými operáciami**, v ktorom sa vyskytuje aspoň jediný identifikátor rubriky rámca s *dátovým typom propozícia*,
 - operand2 je alternatívne
 - **propozičná konštanta**,
 - alebo niektorý z prípadov operand1,
 - operátor je relácia **rovnosti**, resp. **nerovnosti**.

Množinová relácia

Reťazcová relácia

Poznámka: Špecifickým je prípad podmienky propozičného typu keď oba operandy nadobudnú hodnotu NEZNÁMA. V závislosti na aplikáciách je potrebné rozhodnúť, či sa vyhodnotením takej podmienky má jej priradiť hodnota NEZNÁMA alebo PRAVDA.

Propozičná relácia

4.3 Priority vyhodnocovania podmienok

Priority podmienok V niektorých vývojových prostrediach sa využíva možnosť priradiť jednotlivým pravidlám a podmienkam v ich predpoklade statickú alebo dynamickú **prioritu vyhodnocovania**. Týmto prioritami je možné ovplyvňovať postupnosť vyhodnocovania podmienok a pravidiel.

V aplikačných oblastiach, v ktorých je žiaduce zohľadňovať dostupnosť, náročnosť, cenu, spoľahlivosť, diskriminačný účinok údajov, vedľajšie účinky (napr. nevyhnutnosť zastaviť prevádzku skúmaného systému, jeho potenciálnu deštrukciu) a nebezpečenstvo atď. ich získavania, môže to mať veľký význam. Prioritami jednotlivých pravidiel a podmienok sa dá situačne ovplyvňovať postupnosť získavania príslušných údajov. Navyiac, prioritami je tiež možné aspoň z časti imitovať predpokladaný sled uvažovania človeka za rôznych okolností.

Na ilustráciu majme jednoduchý príklad dvoch pravidiel, v ktorých sú priority vyhodnocovania podmienok uvedené v podobe referencovania metarubriky *statickej a dynamickej hodnoty priority*, značíme [SP:], resp. [DP:]. (Dynamicky meniteľné hodnoty priority sa dajú zabezpečiť príslušnými hodnotami metarubriky atribútov v rámcoch.) Za predpokladu, že čím je vyššia uvedená hodnota podmienky, tým je vyššia priorita jej vyhodnocovania, je možné vysledovať postupnosť vyhodnocovania podmienok a pravidiel.

Pravidlo 2.1:

**Priame a
nepriame
hodnoty
priorít**

AK

NAJMENEJ³⁶ 3(Auto.zvuk_prevodovky = kovový [SP:30],
Auto.radenie_rýchlostnej_páky = obťažné [SP:50],
Auto.únik_oleja_z_prevodovky = áno [SP:100],
Auto.úlomky_kovu_v_prevodovke = áno [SP:1]
Auto.rozbeh = nerovnomerný [SP:20])

A

Auto.používateľ **je prvkom množiny** (vodič_začiatočník, vodič_amatér, technický_laik) [SP:40]

³⁶ Množinové kvantifikátory **NAJMENEJ**, **NAJVIAČ** a **PRÁVE** sú predmetom výkladu v článku 4.7

A **NAJMENEJ 1** (Cesta.ciel' = mimo_mesta_pobytu [SP:200],
Cesta.vzdialenosť ≥ 50 km [SP:200],
Počasie_nepriaznivé = PRAVDA [SP:200])
TAK Auto.použitie := neprípustné

Pravidlo 2.2:

AK Auto.startuje_dobre = PRAVDA [SP:90]
A Auto.chod_motora = prirodzený [SP:80])
A Auto.stav_pneumatik = dobrý [SP:200]
A Auto.stav_osvetlovacích_zariadení = dobrý [SP:90]
A Auto.stav_signalizačných_zariadení = dobrý [SP:50]
TAK Auto.použitie := prípustné

V tomto príklade (ktorý sa nesnaží verne reflektovať realitu) hodnotami priorit sa vyjadruje potenciálny postup vyhodnocovania podmienok predovšetkým vzhľadom na obťažnosť získavania potrebných údajov a aj predpokladateľnú sekvenciu uvažovania. Treba si všimnúť, že zisťovanie úlomkov kovu v prevodovke má najnižšiu prioritu, keďže zisťovanie tohto faktu je náročné a nákladné. Zároveň máme dve pravidlá, ktoré, pokiaľ ich predpoklady po vyhodnotení nadobudnú propozičnú hodnotu **PRAVDA**, vedú k protichodným dôsledkom. Na tomto mieste prenechávame na úvahu čitateľa spôsob riešenia potenciálne konfliktnej situácie.

4.4 Semafór pravidla

Semafor
Г

Semafor pravidla je (spravidla jednoduchá) podmienka, ktorej funkciou je pripustiť - v prípade jej splnenia, alebo zabrániť - v prípade nesplnenia, interpretovaniu pravidla. Využitie semaforov môže byť mnohostranné. Nasledujú najčastejšie prípady ich uplatňovania.

Ošetrovanie
nekonzisten-
tostí

1. **Tvorba BZ zahrňujúca reprezentáciu aj nekonzistentných (protichodných) asercií (poznatkov, predpisov, stanovísk rôznych odborných škôl): Ide o reflexiu reality, v ktorej sa nedá vyhnúť situačnej nekonzistentnosti.**

Súčasné použitie protirečivých pravidiel by viedlo buď k patologickým javom v činnosti IM alebo k nekorektným výsledkom odvodzovania. Neželaná situácia sa ošetrí tým, že

- **tvorca BZ** semaforom vyznačí pravidlá patriace do konzistentnej "školy" - teda patriace do jednej z kategórií obsahujúcej vzájomne neprotirečivé asercie,
- **používateľ BZ** na základe systémom predložených informácií o jednotlivých "školách" volí pre daný beh ES niektorú z nich - jeho rozhodnutie sa prejaví tým, že semafor pripustia interpretáciu iba jednej kategórii reprezentovaných pravidiel, čo pripustí ich interpretáciu a súčasne zabráni interpretáciu iných.

2. **Dynamické situačné povoľovanie/zabraňovanie interpretácie špecifických pravidiel:**

Semafor je prostriedkom

- umožňujúcim uprednostniť určitú stratégiu odvodzovania,
- sústredenie pozornosti iba na určité časti BZ a vynechanie z pozornosti iné,
- reagovanie na dynamické zmeny stavu skúmanej reality - povedzme monitorovanie niektorých ich stavov³⁷.

3. **Situačne zabrániť použitiu pravidiel, ktorých interpretácia vzhľadom na zistené/odvodené fakty stratila zmysel³⁸:**

Zmysel uplatňovania semafora tohto typu sa dá ilustrovať na medicínskom príklade: Ak ktosi bol očkovaním imunizovaný proti určitým ochoreniam, tak nemá význam interpretovať pravidlá, ktoré sa týchto ochorení týkajú.

Vynechanie
neapliko-
vateľných
pravidiel

4.5 Dôsledková časť pravidla

Dôsledk-
ová časť
pravidla

V rozsahu predkladanej koncepcie reprezentácie znalostí jadrom produkčného pravidla v kontexte znalostí termového typu je dôsledková časť *jadra* tvorená **identifikátorom rubriky rámca**. Rubrike sa v závislosti na výsledku vyhodnotenia predpokladu **priraduje hodnota v súlade s jej dátovým typom**. Prírodzene to nevylučuje uplatňovanie aj viacerých riadiacich účinkov súvisiacich s priradením takej hodnoty.

³⁷ Jednoduchým príkladom umožňujúcim ilustrovať uplatňovanie semaforov sú pravidlá z odseku 4.1: Inferenčný mechanizmus imitujúci správanie vodiča blížiaceho sa autom k *dopravnému semaforu* by mal mať povolené interpretovať *semaforom produkčného pravidla* len pravidlá P 1.1, P 1.2 a P 1.3. Pravidlo P 1.4 v tejto situácii nemá význam. Keby sa následne uplatnilo pravidlo P 1.2, následne má zmysel uplatniť iba pravidlo P 1.4 - z uvedených iba jeho interpretácia by sa mala semaforom pripustiť. **Takéto účinné odvodzovacie postupy sa dajú zabezpečiť dynamickým prestavovaním hodnôt, na ktoré sa semafor pravidiel odvolávajú.** V danej situácii by sa to dalo zabezpečiť akčnou časťou pravidla P 1.2.

³⁸ Príklad: Pri medicínskom diagnostikovaní z faktu, že pacient bol očkovaný proti určitým infekčným ochoreniam sa dá oprávnenne usúdiť, že imunizáciou vylúčené chorobné jednotky netreba uvažovať.

Pokiaľ hodnota dôsledku je determinovaná iba jediným predpokladom, operácia priradenia hodnoty je triviálna. Vo všeobecnosti však, ako už bolo v inej súvislosti uvedené,

je prípustné, aby rovnaký identifikátor rubriky rámca sa jvyskytoval v roli dôsledku viacerých svojimi predpokladmi sa líšiacich pravidiel.

Tým sa reflektuje to, že **rovnaký cieľ/hypotéza sa dá dosiahnúť/potvrdiť rôznymi spôsobmi.** Rozoberieme tento prípad podrobnejšie. Majme nasledovný systém pravidiel

Rôzne predpoklady môžu viesť k totožnému dôsledku

$$\begin{aligned} p^{(1)} &\rightarrow d \\ p^{(2)} &\rightarrow d \\ \dots & \\ p^{(k)} &\rightarrow d \end{aligned}$$

v ktorom sú $p^{(i)}$ vzájomne sa líšiace predpoklady³⁹ pravidla, ktoré implikujú rovnaký dôsledok. Predpoklady sú podľa dohovoru tvorené konjunkciou podmienok

$$p_j^{(i)}, 1 \leq j \leq n_i, 1 \leq i \leq k.$$

Uvedená sústava pravidiel je ekvivalentom zápisu

$$p^{(1)} \vee p^{(2)} \vee \dots \vee p^{(k)} \rightarrow d,$$

ktorý je v **dizjunktívnej normálnej forme (DNF)**. Z hľadiska reprezentácie asertívnych znalostí v tvare DNF je prípustné rozčlenenie príslušnej formuly do samostatných pravidiel počtom zodpovedajúcich počtu dizjunktov. Má to rad predností.

Každé samostatné pravidlo môže mať osve

- **dynamicky meniteľnú prioritu,**
- **semafór situačne ovplyvňujúci jeho použiteľnosť,**
 - akčnú časť umožňujúcu situačne reagovať na výsledok vyhodnotenia individuálnych predpokladov,
- **postavenie (úlohu) pri *aproximatívnom* (kvantitatívnom aj kvalitatívnom) odvodzovaní (pozri *aproximatívna inferencia*),**
- **spôsob vysvetľovania/zdôvodňovania vyhodnotenia a priradenia *propozičnej hodnoty dôsledku* (pozri *vysvetľovací mechanizmus*),**

čo môže zjednodušiť vytváranie BZ, prispieť k ľahšiemu porozumeniu jej obsahu a priblížiť správanie sa odvodzovacieho procesu k imitovanej predlohe. Okrem toho zjednodušuje tvorbu riadiacej infraštruktúry inferenčného mechanizmu a sledovanie jej obsahu.

Štandardné princípy priradenia *propozičnej hodnoty* predpokladu $p^{(i)}$ na základe vyhodnotenia podmienok $p_j^{(i)}$ dôsledku d na základe vyhodnotenia predpokladov $p^{(i)}$ v prehľadnej podobe uvádza nasledujúca tabuľka.

Výhodnosť samostatnej interpretácie dizjunktov

Princípy vyhodnoco

³⁹ Na rozdiel od podmienok vzájomne odlišných v predpoklade indexom, odlišenie predpokladov vyznačujeme 'horným indexom' - prirodzené číslo v zátvorke.

vania
predpok-
ladu a
dôsledku

$p^{(i)}$	d
ak aspoň jediná podmienka $p_j^{(i)}$ nadobudla hodnotu NEPRAVDA, tak predpoklad $p^{(i)}$ nadobúda tú istú hodnotu	ak aspoň jediný predpoklad $p^{(i)}$ nadobudol hodnotu PRAVDA, tak dôsledok nadobúda svoju pozitívnu hodnotu podľa svojho dátového typu
ak žiadna podmienka $p_j^{(i)}$ nenadobudla hodnotu NEPRAVDA, ale aspoň jedna nadobudla hodnotu NEZNÁMA, tak predpoklad $p^{(i)}$ nadobúda hodnotu NEZNÁMA	ak <u>žiadny</u> predpoklad $p^{(i)}$ nenadobudol hodnotu PRAVDA, ale aspoň jeden nadobudol hodnotu NEZNÁMY, tak dôsledok nadobudne hodnotu NEZNÁMY
ak všetky podmienky $p_j^{(i)}$ nadobudli hodnotu PRAVDA, tak aj predpoklad $p^{(i)}$ nadobúda túto hodnotu	ak všetky predpoklady $p^{(i)}$ nadobudli hodnotu NEPRAVDA, tak aj dôsledok nadobudne negatívnu hodnotu podľa svojho dátového typu

Uvedené princípy sú **podkladom** pre tvorbu procedúr vyhodnocovania predpokladov pravidiel a prirad'ovania (veľmi často len proпозиčnej) hodnoty dôsledku. Kvôli minimalizácii rozsahu odvodzovania sa takéto procedúry realizujú tak, že bez snahy nahradzovať hodnoty NEZISTENÉ pokúšajú sa najprv na základe už dostupných faktov detekovať hoci len

- **jedinú nesplnenú podmienku** v predpoklade pravidla a tým **predpoklad falzifikovať** (priradiť mu hodnotu NEPRAVDA),
- **jediný splnený predpoklad** (nadobúda hodnotu PRAVDA) z tých, ktoré majú **totožný dôsledok**.

V špecifických situáciách sa môže stať, že uvedený princíp - z hľadiska formálnej logiky korektný - nereflektuje implicitné poznatky. Ilustruje to nasledujúci prípad:

Tvorca BZ získava od odborníka danej problémovej oblasti platný poznatok, ktorý s jeho odobrením korektné reprezentuje v podobe pravidla

$$p^{(1)} \vee p^{(2)} \rightarrow d,$$

teda pravidla v DNF tvare. Nie je však zriedkavosťou, keď odborníkovi z aplikačnej oblasti je natoľko samozrejme, že zároveň platí aj povedzme

$$(p^{(1)} \rightarrow d) \& (d \rightarrow p^{(1)}),$$

teda vzťah ekvivalencie $p^{(1)}$ a d , že ani ho nenapadne takú skutočnosť zvlášť uvažovať a vysloviť. Spoluvorcovi BZ, ktorému je spravidla samozrejmosťou rozlišovať medzi **implikáciou** a **ekvivalenciou**, vôbec nemusí napadnúť preverovať, o ktorý z týchto logických vzťahov ide. Následne sa zistí, že dôsledok d bol - v rozpore so vzťahmi reality - pozitívne odvodený na základe priradenia proпозиčnej hodnoty PRAVDA predpokladu $p^{(2)}$, hoci $p^{(1)}$ bol falzifikovaný.

Vzniká teda problém, ktorý je nevyhnutné ošetriť⁴⁰.

Ošetrovanie takých a podobných prípadov sa dá zabezpečiť niekedy na syntaktickej úrovni logického formalizmu, napríklad

- (a) *rozšírením predpokladov pravidiel o všetky negácie skutočností, ktoré by mohli pritrečovať platnosti odvodzovaného dôsledku* - z praktických aspektov je to zväčša neefektívne⁴¹

O možnej
nekonzistentnosti

⁴⁰ Problémy tohto druhu nie sú vždy tak evidentné ako v uvažovanom jednoduchom prípade. Možno si predstaviť obdobné nevysslovované závislosti medzi entitami, ktoré sa dostávajú do vzájomného vzťahu prostredníctvom celej sekvencie produkčných pravidiel. Vtedy je detekcia príčiny nekonzistentných výsledkov a jej odstránenie náročnejšia.

⁴¹ Často to spôsobuje stratu zrozumiteľnosti obsahu BZ, ťažkosti v činnosti vysvetlovacieho mechanizmu ES a napokon vedie aj k neprimeraným a neprirodzeným nárokom pri získavaní a reprezentácii znalostí a pri prípadnom modifikovaní obsahu BZ.

(b) *vhodným použitím akcií v akčnej časti pravidiel (pozri nasledujúci odsek 4.2.2) a podmienok tvoriacich semafore pravidiel*⁴².

Najmä vzhľadom na neistoty (nespol'ahlivosť) získavania potrebných údajov, inou možnosťou je obohatenie štandardných prostriedkov formálnej logiky prostriedkami, ktoré umožňujú **váhovať vierohodnosť splnenia podmienok, predpokladov a tým aj dôsledkov pravidiel**. To je však už problematika *nekategoričných pravidiel a aproximatívnej inferencie*, ktoré sú predmetom pozornosti v ďalšom.

Okrem týchto hľadísk je žiaduce umožniť používateľovi systému modifikovanie štandardného spôsobu priradovania hodnoty dôsledku, ktorý je odvoditeľný na základe viacerých predpokladov $p^{(i)}$. V závislosti na povahe jednotlivých aplikácií malo by byť možné špecifikovať:

- **požiadavku exhaustívneho vyhodnotenia všetkých pravidiel s totožným dôsledkom**, t.j. požiadavku vyhodnocovania ešte nevyhodnotených predpokladov $p^{(i)}$, aj keď sa už dôsledku d priradila (pozitívna) propozičná hodnota,
- **prioritu vyhodnocovania jednotlivých pravidiel s totožnou dôsledkovou časťou**,
- **spôsob vykonávania akčnej časti vyhodnotených pravidiel**.

Výhodné
uplatnenie
akcií a
semafórov

Tieto metódy vyžadujú komentár.

Štandardne sa vyhodnocovanie predpokladov preruší po detekovaní prvého predpokladu, ktorý nadobudol po vyhodnotení propozičnú hodnotu PRAVDA ("McCarthy-ovský" princíp). Jestvujú však aplikácie, v ktorých je výhodné z pohľadu používateľa zistiť všetky fakty (argumenty), ktoré potvrdzujú/dokazujú platnosť dôsledku aj keď z formálno-logického pohľadu to už nie je potrebné. Vtedy je výhodné pripustiť možnosť lokálne alebo globálne (v danej BZ) nastaviť zmenu štandardnej stratégie na exhaustívne vyhodnocovanie. Lokálne zmeny by sa mali dať priamo voliť pri špecifikovaní pravidla, alebo by mali byť situačne ovplyvniteľné prostredníctvom akcií akčnej časti pravidiel (prostredníctvom semaforov), ako aj akciami z **AP**, **AZ**.

Individuálne priradenie hodnoty priority každému pravidlu umožňuje špecifikáciu očakávanej postupnosti vyhodnocovania predpokladov pravidiel s totožným dôsledkom. Proces vyhodnocovania týchto priorít by mal mať obdobné vlastnosti ako v **AP**, keď sa umožňuje získavať hodnotu na základe *statickej a dynamickej priority dedenia, resp. odvodzovania*. To znamená, že aj k pravidlu by sa mali dať priradiť nasledovné dva údaje

Metódy
vyhodnocovania

- **statická priorita: celé číslo**,
- **dynamická priorita: $R.p$** .

Poznámka: Stretávame sa spravidla s tým, že **$R.p$** vyplýva z priority odvodzovania rubriky rámca **GLOBAL**, ktorá zodpovedá dôsledkovej časti pravidla - pozri v predošlom "odvodenie" v metarubrike priorít využívaných **AP** metódami.

Spôsob využívania priorít:

AK pravidlo má špecifikovanú dynamickú prioritu,

TAK

*AK údaj $R.p$ je známy,
TAK sa použije,*

INAK

*AK je špecifikovaná statická priorita,
TAK sa použije jej hodnota,
INAK sa použije štandardná hodnota.*

AK niekoľko pravidiel s totožnou dôsledkovou časťou má
rovnakú – akokoľvek nadobudnutú - prioritu,

⁴² O prednostiach tejto alternatívy v porovnaní s predošlou sa dá presvedčiť na pravidlách P 2.1 a P 2.2 z odseku 4.1.4. Ak by sa predpoklad prvého v negovanej podobe začlenil ako konjunkt do predpokladu druhého, konfliktný prípad je jednoducho riešený. Vo všeobecnosti by však počet skutočností, ktoré by mohli platnosť dôsledku porušiť, môže byť tak veľký, že uplatňovanie tohoto prístupu by bolo mimoriadne nepraktické. Preto je výhodnejším uplatniť v akčnej časti pravidla P 2.2 akciu, ktorá by falzifikovala potvrdenie dôsledku ako produkt interpretácie pravidla P 2.1. Išlo by o **prípád nemonotónnej inferencie**, o ktorej ešte bude reč. Ak by pravidlo P 2.1 ešte nebolo vyhodnotené, tak by akcia akčnej časti pravidla P 2.2 mala nastaviť semafor pravidla P 2.1 tak, aby sa zabránilo jeho interpretácii.

TAK *prvé v poradí sa vyhodnotí pravidlo, ktorého predpoklad obsahuje podmienku s najvyššou prioritou.*

Statická a dynamická prioritá pravidiel

V prípade, že ani takto nie je možné vybrať jediné pravidlo, možno postupovať ľubovoľne, najčastejšie v poradí usporiadania pravidiel, ktoré sa stali aktuálnymi.

Poznámka: Poradie vyhodnocovania pravidiel môže mať v mnohých situáciách veľký dopad na proces inferencie. Vyplyva to z uplatňovaných riadiacích akcií potenciálne produkovaných v procese vyhodnocovania predpokladov pravidiel. Ako je už z doterajšieho výkladu zrejmé, sprievodným prejavom vyhodnocovania môže byť iniciovanie množstva naväzujúcich akcií a procesov v ľšiacom sa poradí. Z nich značné účinky môžu mať povolené aj nepovolené akcie akčnej časti pravidiel.

Štandardne sa vykonanie akčnej časti viaže k propozičnej hodnote predpokladu, t.j. vykonajú sa len akcie tých pravidiel, ktorých predpoklad nadobudol hodnotu PRAVDA. Pružná voliteľnosť stratégií by mali umožňovať globálne alebo lokálne a aj situačné potlačenie takejto štandardnej možnosti – pozri v nasledujúcom.

Metóda využívania priorit

4.6 Akčná časť produkčných pravidiel

Akčná časť je

Akčná časť konzekventu pravidla je **fakultatívna**. Pokiaľ je však definovaná, môže byť tvorená rozsiahlou postupnosťou akcií od jednoduchých povelov a priradení hodnôt rubrikám rámcov, semaforom pravidiel, resp. riadiacim údajovým štruktúram, až po volanie rozsiahlych procedúr (tie, okrem iného, môžu v konečnom dôsledku zabezpečiť tie isté činnosti ako ostatné spomenuté akcie).

Ilustráciou spúšťania rôznych možných akcií sú napr. nasledovné typy povelov: **prirad', prestav, zruš, rob (volanie procedúry), použi_stratégiu, vytvor_objekt, zruš_objekt, ukáž, vyhl'adaj, zapíš, export, import, vlož_BZ, odlož_BZ** a iné.

Povely povelového jazyka

Akcie sa vykonávajú **bezprostredne po interpretácii jadra daného produkčného pravidla** (aj keď v prípade viacnásobných predpokladov sa nedá definitívne určiť hodnota dôsledku). Spravidla sa pritom uplatňuje *štandardná stratégia*, ktorá pripúšťa vykonanie akcií iba v prípade, že predpoklad pravidla nadobudol po interpretácii propozičnú hodnotu PRAVDA. Podľa aplikačných požiadaviek

má sa však umožniť modifikácia tejto stratégie.

Nie je obťažné si predstaviť, že práve vtedy, keď predpoklad pravidla **nenadobudne** propozičnú hodnotu PRAVDA môže byť žiaduce vykonať tomu zodpovedajúce akcie. Môže ísť pritom o akcie, ktoré sa líšia v závislosti od toho, či hodnota je NEPRAVDA alebo NEZNÁMA. Preto je často účelné spomenutú *štandardnú stratégiu* modifikovať. Jednou z takých možností je podmieňovanie vykonania jednotlivých akcií vlastnou **spúšťacou podmienkou**, ktorá môže, ale nemusí, zahrňovať aj propozičnú hodnotu vyhodnotenia predpokladu pravidla. Spúšťacie podmienky sa môžu odvolávať napr. na obsah bázy faktov a obsah riadiacich údajových štruktúr, t.j. infraštruktúry inferenčného mechanizmu.

Spúšťacia
podmienka
akcií
akčnej
časti
pravidla

Príkladom jednoduchých spúšťacích podmienok referencujúcich propozičnú hodnotu predpokladu svojho pravidla sú jednoduché návestia a ich kombinácie. Môžu byť tvorené povedzme symbolmi P(ravda), N(epravda), X(=neznáme), ktoré sú priradené jednotlivým akciám. Ilustráciou sú konštrukty ako *P:akcia_1, N:akcia_2, P/X:akcia_3, X:akcia_4, P/N:akcia_5, X/N:akcia_6* a pod. Akcie sa potom vykonávajú selektívne v závislosti od propozičnej hodnoty, ktorú nadobudol predpoklad. Keby to bola povedzme hodnota NEZNÁMA, z akcií uvedených v ilustrácii by to boli iba tretia, štvrtá a šiesta. Spúšťacie mechanizmy tohto typu sú prostriedkom umožňujúcim situačné špecifikovanie "smeru" odvodzovania v závislosti na interpretácii predpokladu pravidla.

Pokiaľ v určitej aplikácii použitie uvedených spúšťacích podmienok nepostačuje na cieleň spúšťanie akcií a tým na vyvolanie zodpovedajúceho stavu riešenia problému, vtedy sa môžu použiť aj spúšťacie podmienky iného typu, prípadne aj v kombinácii s doteraz uvažovanými. Spravidla sa jedná o jednoduché podmienkové konštrukty referencujúce najmä obsah **riadiacich údajových (infra)štruktúr**, z ktorých vyplýva jednak aktuálny stav riešenia problému, prípadne aj jeho prehistória (pozri v ďalšom).

V princípe má ísť o jednoduché podmienkové konštrukty, ktorých vyhodnocovanie by nemalo spúšťať vlastný ("vedľajší") inferenčný proces lebo by mohol vyvolať (najmä pre používateľa systému ťažko pochopiteľný) neželaný rozsah a zameranie odvodzovania. Jedným zo spôsobov potlačenia toho javu je nahradenie - **dočasné, pracovné, iba pre aktuálnu potrebu vyhodnocovania spúšťacej podmienky** - práve referencovanej hodnoty rubriky rámca, ktorá je v tomto okamihu **NEZISTENÁ** a iba odvoditeľná, hodnotou **NEZNÁMA**.

Akcie akčnej časti dôsledku pravidla (popri tých, ktoré sú priradené rubrikám rámcov) sú vhodnými prostriedkami jednak zmysluplného ovplyvňovania odvodzovacieho procesu a najmä udržovania konzistentnosti bázy faktov. Významnou je aj možnosť zabezpečiť týmto spôsobom mechanizmy *nemonotónnej inferencie*.

Využívanie akcií zjednodušuje aj postupnú tvorbu a modifikovanie BZ. Akciami je možné - namiesto pridávania riadiacich podmienok do predpokladu produkčných pravidiel - situačne a z lokálneho pohľadu, t.j. pri tvorbe toho-ktorého pravidla (a rámca), zabezpečiť konzistentnosť bázy faktov. Akcie umožňujú obísť potrebu reprezentáciu *všetkých možných predvídateľných okolností, ktoré by mohli potenciálne svedčiť v neprospech dôsledku pravidla*, v jeho predpoklade. Tým aj nevyhnutnosť ich exhaustívneho vyhodnocovania pri interpretácii zodpovedajúceho pravidla.

O potrebe
nahradíť
nezistenú
hodnotu
NEZNÁMA

Je to v aj súlade s fenoménom inteligentných procesov. Tie sú cieleň, nepožadujú vyhodnocovanie všetkých možných údajov, najmä tých čo zodpovedajú zriedkavým a výnimočným okolnostiam. Sú zväčša zamerané na čo "najlacnejšie" potvrdzovanie dôsledkov. Na druhej strane, sú (majú byť) vždy pripravené v prípade detekcie skutočnosti vylučujúcej už odvodený dôsledok, odstúpiť od neho. To zodpovedá fenoménu *nemonotónneho usudzovania*, ktorý sa dá implementačne zabezpečiť aj akciami produkčných pravidiel.

4.7 Komplexné podmienky

Viacere vývojové prostredia tvorby ES poskytujú prostriedky tvorby zložených podmienok, či nepriamych referencií objektov a ich vlastnosti v podmienkách predpokladu pravidiel, prípadne aj v iných úlohách (napr. semafor, akčná časť, **AZ** rubriky). Motiváciou toho je prednosť vyjadrovacej⁴³ aj odvodzovacej účinnosti takých konštruktov. A tie z nich, ktoré umožňujú vytvárať odkazy na objekty (a ich vlastnosti), ktoré v čase tvorby BZ v princípe nemôžu byť známe, tvoria silný a nevyhnutný referenčný prostriedok rozvinutých ES.

Predmetom pozornosti tejto (pod)kapitoly sú naznačené konštrukty.

4.7.1 Kvantifikované množiny podmienky

Kvantifikovaná množina podmienok v predpoklade pravidiel

Majme množinu vzájomne sa líšiacich (elementárnych) podmienok p_i . Nech je jej kardinalita N . **Pokiaľ pre určité rozhodnutie stačí, keď ľubovoľná neprázdna podmnožina kardinality M , $1 \leq M \leq N$, elementárnych podmienok p_i nadobudne proпозиčnú hodnotu PRAVDA, potom hovoríme o kvantifikovanej množine podmienok.**

Kvantifikovaným množinám podmienok zodpovedajú aj zápisy so známymi kvantifikátormi

$$\forall (p_1, p_2, \dots, p_n), \text{ resp. } \exists (p_1, p_2, \dots, p_n).$$

UNIVERZÁLNE KVANTIFIKOVANÁ množina podmienok zodpovedá prípadu, v ktorom $M = N$, t.j. vyžaduje sa splnenie všetkých N elementárnych podmienok k tomu, aby kvantifikovaná (zložená) podmienka nadobudla proпозиčnú hodnotu PRAVDA.

EXISTENČNE KVANTIFIKOVANÁ množina podmienok zodpovedá prípadu, v ktorom $M = 1$, t.j. na priradenie proпозиčnej hodnoty PRAVDA kvantifikovanej množine stačí splnenie ľubovoľnej z podmienok p_i .

Univerzálne

V aserciách vzťahujúcich sa k množstvu praktických aplikácií je vzhľadom na expresivitu a odvodzovaciu účinnosť výhodné používať aj ďalšie – zovšeobecňujúce – typy kvantifikátorov. Pri tvorbe ES sa dá stretnúť najmä s kvantifikátormi typu

Existenčne kvantifikovaná množina

NAJMENEJ M , NAJVIAC M a PRÁVE M z (p_1, p_2, \dots, p_n) podmienok.

Sémantika uvedených kvantifikátorov vyplýva z ich vzťahu k univerzálnemu (píšeme **PRE_VŠETKY**), resp. existenčnému (píšeme **EXISTUJE**) kvantifikátoru predikátovej logiky prvého rádu. Platí

**PRE_VŠETKY = PRÁVE N z N , resp. PRE_VŠETKY = NAJMENEJ N z N ,
EXISTUJE = NAJMENEJ 1 z N .**

Zovšeobecná kvantifikácia

Na základe týchto vzťahov možno odvodiť sémantiku vyššie uvedených zovšeobecných kvantifikátorov. Sémantika kvantifikátora **NAJMENEJ**⁴⁴ M z N (v súlade s intuitívnym významom slova **najmenej**) je možné špecifikovať takto:

- (1) v n -prvkovej množine podmienok sa uvažujú všetky m -tice,
- (2) tie sa proposisične vyhodnotia ako univerzálne kvantifikované, a
- (3) ak aspoň jedna z nich nadobudla proposisičnú hodnotu PRAVDA, tak aj pôvodný výraz kvantifikovaný kvantifikátorom **NAJMENEJ M z N** nadobudne túto hodnotu.

Analogicky k vzťahu medzi univerzálnym a existenčným kvantifikátorom v predikátovej logike 1.

⁴³ Prednosťou je aj prirodzený spôsob reprezentácie častých asercií aplikačných expertov.

⁴⁴ Niektorí autori hovoria v tomto prípade o **Booleovskej prahovej funkcii**.

rádu (negáciu zapíšeme v tvare NON)

$\text{NON [PRE_VŠETKY}(p_1, p_2, \dots, p_n)] = \text{EXISTUJE}(\text{NON } p_1, \text{NON } p_2, \dots, \text{NON } p_n)$

možno písať

$\text{NAJVIAČ } M(p_1, p_2, \dots, p_n) = \text{NAJMENEJ } (N-M)[(\text{NON } p_1), (\text{NON } p_2), \dots, (\text{NON } p_n)]$

Sémantika
kvantifi-
kátora
NAJMENEJ

$\text{PRÁVE } M(p_1, p_2, \dots, p_n) = [\text{NAJMENEJ } M(p_1, p_2, \dots, p_n) \ \& \ \text{NAJVIAČ } M(p_1, p_2, \dots, p_n)]$.

Vzhľadom na trojhodnotové propozičné vyhodnocovanie takto kvantifikovaných množín elementárnych podmienok platí:

- výraz **NAJMENEJ** $M(p_1, p_2, \dots, p_n)$ nadobudne pravdivostnú hodnotu
 - **PRAVDA** vtedy a len vtedy, keď z uvedených N podmienok počet tých, ktoré nadobudnú hodnotu **PRAVDA** je rovný alebo väčší než M ,
 - **NEPRAVDA** vtedy, keď počet podmienok s hodnotou **NEPRAVDA** je väčší $N-M$,
 - **NEZNÁMA** vo všetkých ostatných prípadoch,
- výraz **NAJVIAČ** $M(p_1, p_2, \dots, p_n)$ nadobudne pravdivostnú hodnotu
 - **PRAVDA** vtedy a len vtedy, keď z uvedených N podmienok počet tých, ktoré nadobudnú hodnotu **NEPRAVDA** je aspoň $N-M$,
 - **NEPRAVDA** vtedy keď počet podmienok s hodnotou **PRAVDA** je väčší než M ,
 - **NEZNÁMA** vo všetkých ostatných prípadoch,
- výraz **PRÁVE** $M(p_1, p_2, \dots, p_n)$ nadobudne pravdivostnú hodnotu
 - **PRAVDA** vtedy a len vtedy, keď z uvedených N podmienok počet tých, ktoré nadobudnú hodnotu **PRAVDA** je práve M a všetky zostávajúce podmienky majú hodnotu **NEPRAVDA**,
 - **NEPRAVDA** vtedy keď počet podmienok s hodnotou
 - ✓ **PRAVDA** je väčší než M ,
 - ✓ **NEPRAVDA** je väčší než $N-M$
 - **NEZNÁMA** vo všetkých ostatných prípadoch.

Sémantika
kvantifikácie
NAJVIAČ
a
PRÁVE

Interpre-
tácia
kvanti-
fikátora

NAJMENEJ

Poznámka: Nie je zriedkavé, keď vo viacerých pravidlách sa referencuje tá istá množina podmienok (p_1, p_2, \dots, p_n) s rovnakými alebo líšiacimi sa kvantifikátormi. Vtedy je výhodné priradiť takej množine identifikátor a s ním ju pamätať. Množina sa dá následne referencovať, hoci aj s odlišným kvantifikátorom, v predpokladoch rôznych pravidiel. Prednosti nepriameho odkazovania na kvantifikované množiny elementárnych podmienok sa prejavujú okrem iného v kontexte s kvalitatívnymi metódami aproximatívneho odvodzovania (pozri v ďalšom).

NAJVIAČ

4.7.2 Nepriame referencovanie entít

Výkonné (rozvinuté) vývojové prostredia a prostredníctvom nich vytvorené ES, najmä tie, ktoré umožňujú v čase zbiehania generovať objekty a ich triedy, poskytujú prostriedky na

nepriame, implicitné, referencovanie rubriík individuových objektov alebo ich tried.

Nepriame,
implicitné

Sú to konštrukty, ktoré sa dajú uplatniť

a ich
uplatnenie

- ✓ v podmienkách predpokladu pravidiel,
- ✓ v akciách AP, AZ a akčnej časti pravidiel,
- ✓ vo funkciách alebo procedúrach,
- ✓

pri nepriamom referencovaní rubriík objektov/tried, prípadne aj iných zdrojov údajov. V čase vytvárania konštruktov, t.j. pri tvorbe BZ, umožňujú

- odkazovať na entity, ktoré vtedy ešte nie sú presne známe,
 - nahradzovať individuálne odkazy generickými, t.j. nahradzovať potenciálne veľké množstvo konkrétnych referencovaní zovšeobecneným (hromadným) odkazom,
 - vyhľadávať entity, o ktorých nemôže byť vopred známe, či vyhovujú špecifikovaným požiadavkam, podmienkam,
 - vyhľadávať fakty korešpondujúce vlastnostiam entít.

Jedná sa teda o možnosť vytvárania

symbolových konštruktov nahradzujúcich skutočné identifikátory rubriík rámcov reprezentujúcich (T-, I-)objekty,

ktoré sa dajú vnímať a používať ako prostriedok **NEPRIAMEHO ODKAZOVANIA**. Svojou odvodzovacou účinnosťou a možnosťami sú to mimoriadne významné prostriedky výrazne obohacujúce a zefektívňujúce tvorbu BZ aj ich uplatňovanie. Efektívne sú aj pri prehľadávaní potrebných údajov.

Na ilustrovanie a ozrejmienie problematiky majme jednoduchý príklad: Nech sa ES uplatňuje pri výbere vhodných adeptov do určitej bezpečnostnej služby. Adepti musia spĺňať nasledujúce kritéria: muži vysokí minimálne 175 cm s intelligenčným kvocientom (IQ) nie horším než 115 bodov.

Produkčnými pravidlami je principiálne možné identifikovať jedincov spĺňajúcich uvedené podmienky, avšak doteraz uvedené prostriedky tvorby podmienok v predpoklade jadra produkčných pravidiel umožňujú referencovať iba **jestvujúce rubriky už jestvujúcich rámcov**.

Ako však referencovať všetkých uchádzačov o zamestnanie registrovaných na úrade práce? Ako zostrojiť vhodné **generické produkčné pravidlo**, ktoré nemôže byť vopred informované o aktuálnej a sústavne sa meniacej situácii na jednotlivých úradoch práce?

Keby bolo prípustné odvolávať sa len na konkrétne individua, by bolo potrebné vytvoriť toľko produkčných pravidiel koľko je uchádzačov. Čas a námaha, ktorú by taká činnosť vyžadovala by bola zrejme nezrovnateľne vyššia ako prosté manuálne prehľadanie kartotéky. Nevhodnosť takého prístupu sa prejaví ešte väčšmi pri predstave potreby pravidelnej a frekventovanej aktualizácie sústavy zodpovedajúcich produkčných pravidiel.

Očividným by sa pritom stalo, že vytvárané pravidlá by mali rovnakú podobu, lišili by sa iba odlišnými referenciami názvov I-objektov. Už porovnanie s klasickými procedurálnymi programami určite vedie k presvedčeniu, že by sa jednalo z viacerých aspektov o krajne neefektívny postup. Vzniká teda prirodzená otázka:

Jestvuje možnosť nahradit' také pravidlá jediným zovšeobecneným (generickým), v ktorom namiesto referencovania konkrétnych indivíduí bude vystupovať obdoba určitej premennej?

Pozitívna odpoveď spočívajúca na uplatňovaní **nepriameho, implicitného referencovania** entít sa rozvádza v nasledujúcich článkoch.

4.7.3 Technika nachádzania zhody

Technika

nachádzania zhody – TNZ

Technika nachádzania zhody (symbolovo **TNZ**, v *anglicky písanej literatúre* *pattern matching*), je jedným zo spôsobov **implicitného referencovania** hodnôt rubrik rámcov.

Konkrétnou formou TNZ (z viacerých možných) sú kombinácie **univerzálne** alebo **existenčne kvantifikovaných podmienok** v predpoklade pravidiel, resp. **výrazov v akciách, funkciách, alebo procedúrach**.

V nadväzujúcom výklade uplatníme nasledujúci dohovor:

- ☞ **univerzálne kvantifikovaný** T-objekt zapíšeme v tvare **{meno_rámca}**, kde zátvorky { } symbolizujú univerzálnu kvantifikáciu,
- ☞ **existenčne kvantifikovaný** T-objekt zapíšeme v tvare **<meno_rámca>**, kde zátvorky < > symbolizujú existenčnú kvantifikáciu.

Univerzálna kvantifikácia

Účinok podmienky s UNIVERZÁLNE KVANTIFIKOVANÝM objektom je nasledovný:

Podmienka sa postupne testuje s každým I-objektom patriacim do triedy univerzálne kvantifikovaného T-objektu. Generuje sa pritom lokálny zoznam objektov vyhovujúcich danej podmienke a keď sa

a

existenčná kvantifikácia RÁMCA

- **nenatrafí na objekt, ktorý podmienku nesplňuje, celá kvantifikovaná podmienka je splnená - priradí sa jej propozičná hodnota PRAVDA,**
- **detekuje aspoň jeden prípad objektu, ktorý nevyhovuje špecifikovanej podmienke, kvantifikovaná podmienka je nesplnená - priradí sa jej hodnota NEPRAVDA a lokálny zoznam zanikne,**
- **nenatrafí na žiadny objekt, ktorý podmienke nevyhovuje, ale aspoň na jediný, o platnosti ktorého sa nedá rozhodnúť, tak kvantifikovaná podmienka je nerozhodnuteľná - priradí sa jej hodnota NEZNÁMA a lokálny zoznam zanikne.**

Interpretácia univerzálnej kvantifikácie

Vo všeobecnosti je predpoklad pravidla tvorený viacerými podmienkami a všetky smú byť kvantifikované.

Ak vyhodnocovanie niektorej z podmienok spôsobilo vytvorenie lokálneho zoznamu objektov, tak ten sa zachová až do úplného vyhodnotenia celého predpokladu.

Lokálny zoznam

To má podstatný význam, lebo účinok nadväzujúcej, či nadväzujúcich kvantifikovaných podmienok sa môže vzťahovať práve na taký zoznam.

Ukážeme si to na vyššie načrtnutom príklade. Uvažujme pritom množinu uchádzačov o zamestnanie reprezentovaných hierarchizovanou štruktúrou rámcov:

UCHÁDZAČ_O_ZAMESTNANIE {([] vek:), ([] výška:), ([] pohlavie:), ([] iq:)}

ZÁUJEMCA_O_STRÁŽNU_SLUŽBU {(je_podtriedou: UCHÁDZAČ_O_ZAMESTNANIE), ([] vek:), ([] výška:), ([] pohlavie:), ([] iq:)}

ZÁUJEMCA_O_PRÁCU_NA_SMENY {(je_podtriedou: UCHÁDZAČ_O_ZAMESTNANIE), ([] vek:), ([] výška:), ([] pohlavie:), ([] iq:)}

Milan {(je_prvkom: ZÁUJEMCA_O_PRÁCU_NA_SMENY), (vek: 23), (výška: 189), (pohlavie: mužské), (iq: 108)}

Fero {(je_prvkom: ZÁUJEMCA_O_PRÁCU_NA_SMENY), (vek: 22), (výška: 174), (pohlavie: mužské), (iq: 118)}

Milka {(je_prvkom: ZÁUJEMCA_O_PRÁCU_NA_SMENY), (vek: 23), (výška: 162), (pohlavie: ženské), (iq: 115)}

Ilustračný príklad

Helena {(je_prvkom: ZÁUJEMCA_O_PRÁCU_NA_SMENY), (vek: 25),

(výška: 159), (pohlavie: ženské), (iq: 114)}
Danka {(je_prvkom: ZÁUJEMCA_O_STRÁŽNU_SLUŽBU), (vek: 21),
(výška: 170), (pohlavie: ženské), (iq: 117)}

Valika {(je_prvkom: ZÁUJEMCA_O_STRÁŽNU_SLUŽBU), (vek: 22),
(výška: 172), (pohlavie: ženské), (iq: 115)}

Martin {(je_prvkom: ZÁUJEMCA_O_STRÁŽNU_SLUŽBU), (vek: 23),
(výška: 189), (pohlavie: mužské), (iq: 107)}

Peter {(je_prvkom: ZÁUJEMCA_O_STRÁŽNU_SLUŽBU), (vek: 24),
(výška: 179), (pohlavie: mužské), (iq: 118)}

V tomto prípade aplikácia **univerzálne kvantifikovanej podmienky** v tvare

{UCHÁDZAČ_O_ZAMESTNANIE}.iq ≥ 100,

spôsobí prehľadanie všetkých I-OBJEKTov uvedenej triedy, pričom, ako zrejme, pre všetky je podmienka splnená. Produktom aplikácie podmienky je vytvorenie nasledovného lokálneho zoznamu: (*Milan, Fero, Milka, Helena, Danka, Valika, Martin, Peter*).

Modifikovaná kvantifikovaná podmienka v tvare

{UCHÁDZAČ_O_ZAMESTNANIE}.iq ≥ 115

nie je splnená, lebo už prvý objekt jej nevyhovuje - preto ani nevytvorí žiadny lokálny zoznam.

Pri **EXISTENČNE KVANTIFIKOVANEJ PODMIENKE** uplatňuje sa nasledujúci spôsob vyhodnocovania:

Podmienka sa postupne testuje s každým I-objektom patriacim do triedy existenčne kvantifikovaného T-objektu. Generuje sa pritom lokálny zoznam objektov vyhovujúcich danej podmienke a keď sa

- **natrafi na objekt, ktorý podmienku spĺňa, ten sa zaradi do lokálneho zoznamu; pokračuje sa však ďalej tak, aby všetky objekty spĺňajúce podmienku sa zaradili do lokálneho zoznamu - neprázdny lokálny zoznam implikuje priradenie propozičnej hodnoty PRAVDA existenčne kvantifikovanej podmienke,**

- **prázdny lokálny zoznam indikuje, že mohli nastať dva alternatívne prípady**

- ✓ **bol detekovaný aspoň jeden objekt, ktorý v príslušnej rubrike má hodnotu NEZNÁMA, čo spôsobí, že kvantifikovaná podmienka nadobudne hodnotu NEZNÁMA,**

- ✓ **žiadny objekt nevyhovuje špecifikovanej podmienke, čo spôsobí, že kvantifikovaná podmienka je nesplnená, nadobudne hodnotu NEPRAVDA.**

Neprázdny lokálny zoznam vyhovujúcich objektov sa zachová až do úplného ukončenia všetkých činností súvisiacich s práve vyhodnocovanou zložkou produkčného pravidla.

V uvažovanom prípade existenčne kvantifikovaná podmienka tvaru

<UCHÁDZAČ_O_ZAMESTNANIE>.iq ≥ 115

je splnená – vytvorí lokálny zoznam (*Fero, Milka, Danka, Valika, Peter*).

Doteraz spomínaná kvantifikácia T-OBJEKTov sa uplatňovala vyhodnocovaním referencovaných vlastností všetkých I-OBJEKTov patriacich do danej triedy – nezávisle od úrovne T-OBJEKTU v danej taxonomii. Je však prípustné kvantifikovať aj I-OBJEKTY. Účinok kvantifikácie sa štandardne uplatní na všetky bezprostredné (o cestu dĺžky 1 vzdialené) S-OBJEKTY.

Lokálne zoznamy objektov, ktoré vznikli z identifikátorov tých, ktorých rubriky vyhovujú

Účinok
univerzálnej
kvantifikácie

- kvantifikovanej podmienke, sú využitelné v**
- **sekvencii podmienok v predpoklade**
 - **a v akčnej časti daného pravidla.**

Interpre-
tácia exis-
tenčnej
kvantifikácie

Po ukončení vyhodnocovania pravidla, pokiaľ sa vhodnou akciou - zvyčajne v akčnej časti pravidla - nezabezpečí zachovanie lokálneho zoznamu, tento zoznam zanikne.

V prípade sekvencie dvoch podmienok tvaru

**<UCHÁDZAČ_O_ZAMESTNANIE>.iq ≥ 115 &
& {UCHÁDZAČ_O_ZAMESTNANIE}.pohlavie=mužské,**

**<UCHÁDZAČ_O_ZAMESTNANIE>.iq ≥ 115 &
& <UCHÁDZAČ_O_ZAMESTNANIE>.pohlavie = mužské**

sa na lokálny zoznam vytvorený prvou podmienkou (*Fero, Milka, Danko, Valika, Peter*) aplikuje nasledujúca podmienka. V prvej kombinácii podmienok predpoklad nie je splnený, lebo sa v zozname sa vyskytujú objekty nevyhovujúce druhej podmienke. Preto zaniká aj lokálny zoznam. V druhom prípade je však predpoklad splnený, lebo druhá podmienka je splnená. Výsledkom je nový zoznam (*Fero, Peter*).

Keby sa predpoklad v druhom prípade rozšíril o podmienku

<UCHÁDZAČ_O_ZAMESTNANIE>.výška ≥ 175

celý predpoklad by bol splnený a zároveň by sa vytvoril nový lokálny zoznam tvaru (*Peter*)

Po týchto ilustračných príkladoch uveďme rad zovšeobecňujúcich princípov.

Účinok
uplatnenia
existenčnej
kvantifikácie

☞ **Prvá kvantifikovaná podmienka implicitne referuje I-objekty zodpovedajúcej triedy, resp. S-objekty zodpovedajúceho I-objektu, avšak nadväzujúce podmienky predpokladu už referujú objekty z vytvoreného lokálneho zoznamu, pokiaľ vznikol.**

☞ **Ak všetky podmienky predpokladu**

- ✓ **sú rovnako kvantifikované, poradie podmienok v predpoklade je KOMUTOVATELNÉ a výsledná prozopzičná hodnota predpokladu ako aj výsledný obsah lokálneho zoznamu nie je závislý na poradí podmienok.**
- ✓ **nie sú rovnako kvantifikované, ich poradie v predpoklade nie je komutovateľné a výsledok jeho vyhodnotenia je závislý na poradí podmienok.**

☞ **Podmienky v predpoklade je prípustné aj viacnásobne kvantifikovať.**

Možno teda vytvoriť implicitné referencie typu

**<MENO_RÁMCA>, {MENO_RÁMCA},
<<MENO_RÁMCA>>, {{MENO_RÁMCA}},
<<<MENO_RÁMCA>>>, {{{MENO_RÁMCA}}} atď.**

Uplatnenie
lokálnych
zoznamov

Spôsob vyhodnotenia a účinok kvantifikovaných podmienok je rovnaký ako v predošlom. Avšak **KAŽDÁ NÁSOBNOSŤ KVANTIFIKOVANIA VYTVARA SVOJ VLASTNÝ LOKÁLNY ZOZNAM. Tie ale nie sú závislé na poradí viacnásobne kvantifikovaných podmienok, ani na počte kvantifikácií.**

Takže v uvažovanom príklade výsledok vyhodnotenia predpokladu v tvare

**<<UCHÁDZAČ_O_ZAMESTNANIE>>.iq ≥ 115
& <UCHÁDZAČ_O_ZAMESTNANIE>.pohlavie = ženské
& <<<UCHÁDZAČ_O_ZAMESTNANIE>>>.vek ≤ 23
& <<ZÁUJEMCA_O_STRÁŽNU_SLUŽBU>>.pohlavie=mužské,**

ako sa možno presvedčiť, bude prozopzičná hodnota PRAVDA. V priebehu jeho vyhodnocovaní sa

vytvoria nasledovné tri lokálne zoznamy

<UCHÁDZAČ_O_ZAMESTNANIE >.pohlavie = ženské: (Milka, Helena, Danka, Valika),

<<UCHÁDZAČ_O_ZAMESTNANIE>>.iq ≥ 115
&<<ZÁUJEMCA_O_STRÁŽNU_SLUŽBU>>.pohlavie=mužské: (Peter),

<<<UCHÁDZAČ_O_ZAMESTNANIE>>>.vek ≤ 23: (Milan, Fero, Milka, Danka, Valika, Martin).

- ☞ **Kvantifikované výrazy, teda výrazy s nepriamym referencovaním objektov, sa môžu použiť aj v akčnej časti pravidiel.**

Ilustrácia:

A1:
VytvorObjekt
<PRIJATÍ>: <<UCHÁDZAČ_O_ZAMESTNANIE>>.iq ≥ 115

A2:
<UCHÁDZAČ_O_ZAMESTNANIE>.rbr7:=<UCHÁDZAČ_O_ZAMESTNANIE>.Rbr7+10.

Akcie sa vykonávajú iba na zodpovedajúcej úrovni násobnosti kvantifikácie a štandardne iba keď príslušný predpoklad pravidla bol splnený. Kvantifikované výrazy v akciách sa nevyhodnocujú – používajú zodpovedajúce lokálne zoznamy vytvorené pri vyhodnocovaní predpokladu. Preto ani nezáleží na druhu použitého kvantifikátora.

Kvantifikovaný priradovací príkaz v A2 spôsobuje, že nešpecifikovaná hodnota rubriky rbr7 sa zvýši o 10 v každom I-objekte zo zoznamu, ktorý vznikol z triedy UCHÁDZAČ_O_ZAMESTNANIE na základe vyhodnotenia jedenkrát kvantifikovanej podmienky, teda iba u objektov zo zoznamu (Milka, Helena, Danka, Valika).

- ☞ **Zoznamy vytvorené TNZ sa môžu použiť aj v štruktúrach AP a AZ.** Použitie je analogické ako v akčnej časti pravidiel.
- ☞ **Lokálne zoznamy vznikajúce v dôsledku TNZ existujú iba dočasne,** počas vyhodnocovania pravidla, potom zanikajú. **Možno ich však odovzdať externým procedúram alebo údajovým štruktúram. Možno z nich tiež vytvárať nové dynamické objekty** (pozri akciu A1 vyššie). Akonáhle je **lokálny zoznam vytvorený, stáva sa nezávislou údajovou štruktúrou.** Ak sa počas jeho existencie, alebo po jeho uchovaní, dynamicky zruší alebo pridá objekt do triedy, z ktorej zoznam vznikol, nespôsobí to zmenu jeho obsahu

Viacnásobná kvantifikácia

Ako je zrejmé, TNZ umožňuje vykonávať viaceré významné procesy:

- ☞ **operovať so zoznamami objektov namiesto postupného manipulovania jednotlivými objektami,**
- ☞ **testovať splnenie podmienky v množinách rozmanitých objektov, čo – vzhľadom na vlastnosti objektov – slúži vytváraniu zoznamov tých z nich, o ktorých sa zisilo, že vyhovujú daným kritériám,**
- ☞ **dynamicky narábať s množinami objektov, ktoré nie sú známe v čase vytvárania BZ.**

Uvedené účinky aplikácie kvantifikovaných podmienok sú jednou z možností. To isté sa dá povedať aj o použitom symbolizme. Týka sa to aj všetkých ostatných prostriedkov nepriameho referencovania uvádzaných v tejto kapitole.

4.7.4 Nepriame referencovanie a interpretácia reťazcov

Metóda nepriameho referencovania zodpovedá **nepriamemu adresovaniu**. Vyžaduje použitie vhodných údajových štruktúr, ktoré - aj s dynamicky meniteľným obsahom - sú nositeľmi reálnych adries, t.j. názvov objektov, ktoré sa stávajú predmetom záujmu. **Priamym odkazovaním na obsah podporných štruktúr sa následne nepriamo získava potrebná referencia**. Podpornou štruktúrou môže byť aj vhodne navrhnutý rámec, resp. systém rámcov.

Dohovor: V nasledujúcom budeme uvažovať iba podpornú štruktúru v podobe rámca. Na znak toho, že sa jedná o nepriame odkazovanie, t.j. že nerefereujeme rubriku podporného rámca, ale objekt zodpovedajúci jej hodnote, budeme písať

`\POMOCNÝ_RÁMEC.pomocná_rubrika\reálna_rubrika`.

To je ďalší spôsob, ktorým možno obísť potrebu priameho referencovania objektov, najmä v prípade, keď v čase písania príslušného pravidla, alebo iného konštruktú, ani nie je presne známe, o ktorý objekt v čase inferenčného procesu pôjde.

Ak teda vytvoríme podmienku v tvare

\OSOBA.meno\.výška ≥ 170cm,

interpretér pravidla v čase vyhodnocovania najprv situačne nahradí reťazec **\OSOBA.meno** skutočným, práve aktuálnym menom objektu (je obsiahnutý v referencovanej rubrike referencovaného rámca). Ak by obsahom **OSOBA.meno** bolo **Marek**, tak vznikne podmienka v tvare

Marek. výška ≥ 170cm,

teda testovalo by sa, či daná osoba spĺňa požiadavku telesnej výšky

Nezriedka sa vyskytujú prípady, keď nepriame odkazy majú čosi spoločné. V našom prípade by sme mohli napríklad spresniť, že **Marek** je študentom odboru UI. Teda v rámci poslucháčov uvedeného štúdiijného odboru je predmetom záujmu **Marek**, čo v kontexte nepriameho odkazovania vedie k nasedujúcemu konštruktú

"ŠTUDENT_UI:\Osoba.meno\.výška ≥ 170cm.

Reťazec v úvodzovkách sa nazýva **koreň** spoločnej referencie. Jeho účinok je zrejmy z výsledku interpretácie

ŠTUDENT_UI:Marek. výška ≥ 170cm.

Uvedený spôsob nepriameho referencovania môže sa použiť aj v akčnej časti pravidla, napr.

A1:

**"ŠTUDENT_UI:\OSOBA.meno\.rbr7 :=
:= "ŠTUDENT_UI:\OSOBA.meno\.rbr7 + 10.**

Prostriedkom ďalšieho zvyšovania pružnosti vytvárania a využívania BZ je kombinovanie nepriameho referencovania a TNZ.

Uplatňuje sa v súvislosti so vstupno-výstupnými operáciami, či už vo väzbe na cieľového používateľa systému, alebo vzhľadom na externé softverové prostredie. Spôsob uvažovanej interpretácie oboch metód vyžaduje aby nepriame referencovanie bolo **vnorené** do TNZ, nikdy naopak.

Napríklad kvantifikovanú podmienku tvaru

<"ŠTUDENT_:\RÁMEC.rubrika\>.výška ≥ 170cm

v prípade, že **RÁMEC.rubrika** obsahuje ako svoju hodnotu reťazec **MFF** (resp. **UI**) spôsobí najprv nahradenie predošlej podmienky tvarom

<ŠTUDENT_MFF>.výška ≥ 170cm

resp.

<ŠTUDENT_UI>.výška ≥ 170cm.

4.7.5 Technika tromfov

Ďalší spôsob nepriameho referencovania objektov, s ktorým sa možno stretnúť vo vývojových prostrediach je metóda, ktorú nazveme **technika tromfov** (voľný preklad anglického výrazu "wild-card"). Uplatňovaný symol **?**, resp. **??**, nahradzuje **ČOKOLVEK**, akoby *tromfom nahradzoval ľubovoľ-*

nú kartu, v danom prípade **REŤAZEC**. V závislosti na používaných výrazových prostriedkoch (prípustných konštruktoch) vývojových prostredí môže mať rôznu podobu. Pre jednoduchosť, nie však na úkor princípu, zachováme doterajšiu notáciu.

Keď je napríklad potrebné testovať ľubovoľné objekty majúce rovnakú vlastnosť nezávisle od toho, do ktorej triedy patria, použije sa symbol nahradzujúci ich meno, napr. líspovský symbol **?**, povedzme v tvare

? .výška ≥ 170cm.

Účinok tohto konštruktu spočíva v prehľadaní všetkých objektov, ktoré majú atribút **výška**, ich testovanie vzhľadom na danú podmienku a vytvorenie zoznamu mien vyhovujúcich objektov.

Použitie podmienky v tvare

?.? ≥ 170cm

by viedlo k nájdeniu všetkých objektov, ktoré majú ľubovoľný atribút s hodnotou väčšou ako **170cm** a prirodzene aj k vytvoreniu zoznamu vyhovujúcich

MIEN_RÁMCOV.a_zodpovedajúcich_atribútov.

Podmienky tvaru

? .výška = ?

resp.

? .výška = ??

nájdu všetky objekty, ktoré majú atribút výška a to nezávisle od ich hodnoty, hoci aj **NEZISTENÁ** alebo **NEZNÁMA**. V prvom z uvedených prípadov musí byť kardinalita hodnoty **1**, kým v druhom na počte hodnôt nezáleží.

A ešte jedna dvojica príkladov ilustrujúca rozmanité možnosti použitia symbolu **?** (prirodzene dala by sa v závislosti na syntaktických pravidlách konkrétneho vývojového prostredia zapísať aj v inej podobe):

Rámec.atr = "pokazený je ? valec motora"

Rámec.atr = "pokazený je ?? valec motora".

V oboch prípadoch sa obsah rubriky testuje voči reťazcu na pravej strane podmienky. V prvom prípade, testu vyhovuje reťazec obsahujúci v pozícii výskytu **?** práve jediný údaj (číslo alebo reťazec), kým v druhom prípade, na mieste určenom symbolom **??** sa povoľuje výskyt ľubovoľného počtu údajov, vrátane prázdneho. Uplatnenie uvedených náhradných (*tromfových*) symbolov v akčnej časti pravidiel, obdobne ako v predošlých prípadoch, je tiež možné.