# Získavanie znalostí

## pre znalostné systémy

Marián Mach

Košice

1997

Ing. Marián Mach, CSc.
Katedra kybernetiky a umelej inteligencie
Fakulta elektrotechniky a informatiky
Technická Univerzita v Košiciach
machm@ccsun.tuke.sk

Kniha bola vydaná s finančnou podporou Fakulty elektrotechniky a informatiky TU Košice.

Edícia vedeckých spisov Fakulty elektrotechniky a informatiky TU Košice.



ISBN 80-88786-67-3

Rule: "Scylla"

*If the knowledge is*
*easy to formalize*
*then the application*
*is trivial.*

Rule: "Charybdis"

*If the application is*
*interesting*
*then the knowledge is*
*hard to formalize.*

Richard Forsyth

# Predslov

Kvalita a efektivita riešenia úloh pomocou znalostných systémov je daná kvalitou a kvantitou znalostí využívaných pre riešenie úloh. Preto získavanie znalostí je jedným z najdôležitejších procesov pri tvorbe znalostných systémov. Toto tvrdenie sa s dôslednou pravidelnosťou opakuje temer v každej publikácii zaoberajúcej sa systémami ktoré pre svoju činnosť využívajú znalosti.

V rozpore s týmto tvrdením však záujemca iba sporadicky natrafí na publikáciu ktorá by sa tejto problematike (keď už nie úplne tak aspoň čiastočne) venovala. Výsledkom je často bezradnosť či neúspech pri prvých pokusoch aplikovať znalostné systémy pre riešenie praktických problémov, ústiace až do straty záujmu o znalostné technológie. Pritom v oblasti znalostných systémov existuje mnoho metód ako úspešne a efektívne (aj keď nie ľahko) získavať znalosti.

Cieľom knihy, ktorú práve držíte v ruke, je zaplniť túto medzeru a poskytnúť informácie, ktoré by inak bolo potrebné hľadať roztrúsené v cudzojazyčnej literatúre. Kniha je organizovaná nasledovne. Úvod podáva prehľad jednotlivých smerov a prístupov k získavaniu znalostí spolu s ich stručnými charakteristikami. V nadväznosti na to jednotlivé kapitoly sa venujú týmto smerom podrobnejšie, pričom každá kapitola je venovaná nejakému smeru spomenutému v úvodnom prehľade. Niektoré kapitoly sú doplnené jedným alebo viacerými ilustračnými príkladmi, ozrejmujúcimi predkladanú problematiku.

Jednotlivé prístupy sú ilustrované na konkrétnych metódach získavania znalostí, pričom tento výber som sa snažil robiť tak, aby do knihy boli zaradení typickí reprezentanti dobre charakterizujúci hlavné prúdy v oblasti získavania znalostí. Domnievam sa, že takéto spracovanie umožní čitateľovi orientovať sa v oblastí získavania znalostí a zároveň môže slúžiť vážnejším záujemcom ako východzí bod pre ďalšie štúdium. Kniha sa nesnaží byť encyklopédiou metód a v žiadnom prípade si nerobí nárok na úplnosť.

*MACH, M.: ZÍSKAVANIE ZNALOSTÍ PRE ZNALOSTNÉ SYSTÉMY*

Vďaka organizácii knihy nie je nevyhnutné jej sekvenčné čítanie, pretože jednotlivé kapitoly sú navzájom nezávislé. Čitateľ môže čítanie začať ktoroukoľvek kapitolou bez obavy, že neznalosť predchádzajúcich častí mu znemožní jej pochopenie. Napriek tomu odporúčam ako prvý prečítať úvod, ktorého znalosť napomôže zasadiť obsah každej kapitoly do celkového kontextu pomerne širokej oblasti získavania znalostí.

Kniha je určená všetkým tým, ktorí sa chcú dozvedieť niečo o problematike získavania znalostí, bez ohľadu na to, či ich záujem je podmienený ich profesionálnym zameraním, odporúčaním známych alebo jednoducho zvedavosťou. Pre zvládnutie obsahu knihy postačujú základné znalosti o spôsoboch reprezentácie znalostí a ich využívaní v znalostných systémoch. Výnimku tvoria podkapitoly 5.3 a 6.1, ktoré pre svoje plné pochopenie vyžadujú znalosti o dopredných neurónových sieťach a metódach ich učenia.

Chcel by som poďakovať všetkým mojim študentom, ktorí absolvovali predmet *Projektovanie znalostných systémov* ponúkaný v rámci graduálneho štúdia na Technickej Univerzite v Košiciach. Ich záujem o problematiku bol hnacím motorom mojej práce. Moja zvláštna vďaka patrí tým z nich, ktorí mi svojimi projektmi a diplomovými prácami umožnili hlbšie preniknúť do oblasti získavania znalostí a spoznať skutočnosti, ktoré mi dovtedy boli skryté.

Košice, Apríl 1997 Marián Mach

# Obsah

# Úvod

Metódy používané pre získavanie znalostí je možné deliť do mnohých skupín podľa rozličných hľadísk. Ak uvažujeme delenie podľa cieľa, získame delenie do dvoch základných tried:

- získavanie nových znalostí,
- zjemňovanie existujúcich znalostí.

Prvá skupina sa zameriava na rozširovanie množstva získaných znalostí. Hlavným cieľom je kvantitatívne budovanie báz znalostí (čo samozrejme nevylučuje vysokú kvalitu získavaných znalostí). Pre aplikáciu zodpovedajúcich metód nie je nutná existencia znalostí v explicitnej podobe – báza znalostí môže byť prázdna.

Druhá skupina metód je orientovaná na vylepšovanie (zahŕňajúce ako opravu tak aj zjemňovanie) už získaných znalostí. Ich cieľom je kvalitatívne budovanie báz znalostí (ktoré môže byť sprevádzané aj kvantitatívnymi zmenami). Tieto metódy sa používajú pre rôzne úpravy už existujúcich znalostných báz, pričom môžu zahŕňať ako úpravu znalostí, nachádzajúcich sa v bázach znalostí, tak aj dopĺňanie báz ďalšími znalosťami, ktoré sa v nich nenachádzali.

Samotné získavanie znalostí a ich vkladanie do bázy znalostí môže mať rozličnú podobu. Vo všeobecnosti sa jedná o explicitné vyjadrenie tých znalostí, ktoré sú obsiahnuté v nejakom znalostnom zdroji. Potenciálny znalostný zdroj môže byť predstavovaný expertom, ale tak isto to môžu byť znalosti získavané napr. z databáz, kníh alebo z okolia, empirické dáta a texty v ktorých sú obsiahnuté fakty o oblasti expertízy. Pod explicitným vyjadrením znalostí sa najčastejšie chápe ich reprezentácia pomocou nejakého mechanizmu pre reprezentáciu znalostí (napr. produkčné pravidlá, rámce, predikátová logika ap.) alebo neformálneho či formálneho reprezentačného jazyka.

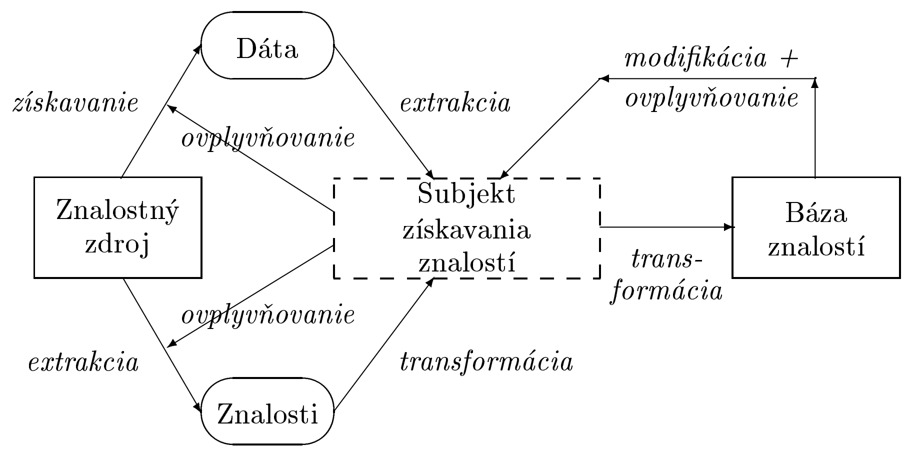
Samotný proces získavania znalostí je možné rozdeliť do niekoľkých úloh:

1. detekcia neúplnosti znalostí (teda nutnosť pokračovať v ďalšom získavaní),

2. získavanie znalostí zo znalostného zdroja (ich explicitné vyjadrenie),

3. transformácia získaných znalostí do vhodného tvaru.

Schematické znázornenie všeobecnej podoby tohto procesu v tvare blokového diagramu je na obr. 0.1.




Obr. 0.1: Proces získavania znalostí

Celý proces budovania bázy znalostí je riadený nejakým subjektom, ktorý hrá centrálnu úlohu v tomto procese. Tento subjekt môže reprezentovať jednu alebo aj viac osôb, pričom tieto osoby môžu byť ako doménovými expertmi tak aj znalostnými inžiniermi.

Znalostný zdroj produkuje znalosti o predmetnej oblasti a/alebo nejaké dáta charakterizujúce časť tejto oblasti. Procesy extrakcie znalostí a získavania dát sú riadené subjektom, ktorý tieto procesy vhodným spôsobom usmerňuje (ak znalostný zdroj vie aktívne produkovať znalosti a/alebo dáta) alebo sám realizuje (v prípade pasívneho znalostného zdroja).

Extrahované znalosti môžu byť priamo vložené do bázy znalostí. Toto vkladanie môže byť doplnené transformáciou znalostí na požadovaný tvar.

Dáta slúžia ako východisko pre extrakciu znalostí obsiahnutých v týchto dátach.

Pri zjemňovaní znalostí, obsiahnutých v báze znalostí, sa znalosti získané zo znalostného zdroja využívajú na modifikáciu obsahu bázy znalostí. Pri získavaní nových znalostí obsah bázy znalostí môže byť využitý pre riadenie využívania znalostného zdroja.

V závislosti od toho, kto rieši tieto jednotlivé úlohy sú možné viaceré modely procesu získavania znalostí:

- klasický model so znalostným inžinierom,
- priamy prenos znalostí,
- strojové učenie.

Historicky najstarším modelom je klasický spôsob získavania znalostí. Pri ňom sú znalosti získavané prostredníctvom znalostného inžiniera, ktorý robí spojovací medzičlánok medzi bázou znalostí a znalostným zdrojom. Znalostný inžinier získava znalosti o danej problémovej oblasti na základe konzultácií s expertom resp. z iných znalostných zdrojov a tieto znalosti vkladá do bázy znalostí. Slabým miestom v tomto spôsobe získavania znalostí je znalostný inžinier, ktorý je vlastne v spracovávanej oblasti laikom. Znalostný inžinier sa vlastne sám učí, pričom učenie pre človeka nie je ľahký proces – zvyčajne je zdĺhavé a nie veľmi efektívne. Iným problematickým článkom je často expert, ktorý nie je vždy schopný vyjadrovať svoje znalosti, používané pre riešenie úloh v danej problémovej oblasti.

Snahy vylúčiť obtiažne konzultácie s expertom viedli k použitiu metód strojového učenia. Najpoužívanejším sa stalo induktívne učenie, pri ktorom sa znalosti indukujú z príkladov, reprezentujúcich typické situácie. Tieto príklady môžu byť priamo odoberané z okolia alebo môžu byť formulované expertom v procese exemplifikácie (pre experta je obvykle ďaleko jednoduchšie vyjadrovať typické príklady, reprezentujúce opisy prípadov a ich riešenia, ako svoje vlastné znalosti, používané pri riešení úloh).

Perspektívnou oblasťou využitia metód strojového učenia je získavanie znalostí priamo z textov v prirodzenom jazyku. V tomto prípade je nutné čítať texty (knihy, články, manuály ap.) a porozumieť im (samotnému textu, grafom, obrázkom). Zložitosť tohto prístupu spočíva nielen v nutnosti spracovávať texty v prirodzenom jazyku, ale aj v nevyhnutnosti súčasne vytvárať model nejakej problémovej oblasti.

Pokusy úplne vylúčiť znalostného inžiniera z procesu získavania znalostí viedli k ďalšiemu typu plnenia bázy znalostí, umožňujúcemu expertovi priamo vkladať svoje znalosti do bázy znalostí (tzv. priamy prenos znalostí). Síce je možné priamo vytvárať bázu znalostí v príslušnom editore v dohodnutom formáte, avšak pre experta minimálne oboznámeného s problematikou programovania je najvýhodnejším tvarom komunikácie medzi ním a systémom komunikácia v prirodzenom jazyku.

Preto pri takomto spôsobe získavania znalostí je nevyhnutný inteligentný modul priameho získavania znalostí, ktorý neostáva iba pasívnym príjemcom znalostí, ale aktívne pomáha expertovi tieto znalosti vyjadrovať, navádza ho a zároveň kontroluje pri získavaní nových znalostí. Tento inteligentný editor musí mať rozsiahle dialógové schopnosti a značné znalosti o štruktúre bázy znalostí.

Všetky prezentované metódy majú svoje výhody a nevýhody. Niektoré je vhodnejšie použiť na začiatku procesu budovania bázy znalostí, iné naopak zase kladú značné nároky na znalosti znalostného inžiniera a teda ich miesto je v neskorších fázach získavania znalostí. Niektoré metódy sú orientované na znalosti všeobecne, iné sú použiteľné iba pre úzky okruh znalostí. Niektoré sa zameriavajú na experta, iné vystačia s doplnkovými znalostnými zdrojmi. Sila prezentovaných metód je práve v ich vhodnej vzájomnej kombinácii, zodpovedajúcej charakteru spracovávanej predmetnej oblasti a dostupnosti jednotlivých znalostných zdrojov.

Väčšina metód rutinne používaná v praxi je orientovaná na získavanie informácií znalostným inžinierom od experta. Tieto metódy sú dopĺňané induktívnymi metódami strojového učenia, keď expert demonštruje niektoré postupy alebo pojmy na konkrétnych príkladoch. Aplikácia priameho prenosu znalostí od experta do bázy znalostí je skôr raritou ako bežnou praxou.

# Časť I

# Prvotné získavanie znalostí

# Kapitola 1

# Priame metódy získavania znalostí

Priame metódy sú založené na vedomých znalostiach experta. Znalostný inžinier sa ho vlastne priamo pýta, čo vie o skúmanej oblasti. Tieto metódy predpokladajú u experta

- dostupnosť znalostí pre introspekciu (expert vie, aké znalosti používa pri riešení úloh),
- schopnosť artikulovať svoje znalosti (dokáže svoje znalosti zdeliť znalostnému inžinierovi).

Všetky priame metódy získavania znalostí možno rozdeliť do niekoľkých skupín podľa rozličných kritérií (napr. podľa ich miesta v procese získavania znalostí, typu získavaných znalostí, podobnosti jednotlivých techník ap.). Ak sa za kritérium zvolí to, ktorá časť predmetnej oblasti je pokrytá danou metódou, tak je možné rozlišovať nasledujúce triedy metód:

**rozhovory** — pokrývajú celú predmetnú oblasť plošným spôsobom, nie sú obmedzené iba na niektorú jej časť,

**prípadové štúdie** — analýzy vybraných konkrétnych prípadov, ktoré reprezentujú iba obmedzenú množinu rezov predmetnej oblasti,

**pozorovania** — sledovanie experta pri riešení konkrétnych prípadov, záber rovnaký ako u prípadových štúdií,

**získavanie pojmov** — metódy sa zameriavajú na vybudovanie pojmovej štruktúry predmetnej oblasti,

**špeciálne techniky** — sú viazané na nejaký konkrétny typ znalostí, pokrývajú iba nejakú vymedzenú časť predmetnej oblasti.

## 1.1 Rozhovory

Vo všeobecnosti majú retrospektívny charakter, pretože sú postavené na expertových znalostiach získaných minulou praxou. Majú podobu viac či menej riadenej konverzácie medzi expertom a znalostným inžinierom.

### Neštruktúrovaný rozhovor

Neštruktúrovaný rozhovor má tvar voľného interview v ktorom ani obsah ani sekvenčnosť rozhovoru nie je určená vopred. Podobá sa obyčajnej konverzácii, kde jeden účastník rozhovoru sa zaujíma o nejaký predmet, o ktorom druhý účastník rozhovoru niečo vie. Samotný neštruktúrovaný rozhovor pozostáva z kladenia otázok, oceňovania získaných odpovedí a podľa tohto ocenenia buď z kladenia ďalších doplňujúcich otázok alebo prechodu na nasledujúcu tému. Otázky majú všeobecný charakter, odpovede majú stručný, širší charakter (detaily sa odkladajú na neskoršie diskusie).

Cieľom je získať kompletný prehľad problémovej oblasti - jedná sa vlastne o identifikáciu pojmov, relevantných pre danú úlohu, a o identifikáciu možných podúloh a charakteristík práce experta pri riešení úloh. Toto všetko sa deje bez prílišného pohrúženia sa do problémovej oblasti.

Pretože sa nevyžadujú ani nepredpokladajú doménové znalosti na strane znalostného inžiniera, tento typ rozhovoru sa používa pre získavanie základných faktických znalostí a širokého prehľadu o doméne záujmu hlavne v počiatočných fázach získavania znalostí. Na druhej strane vďaka neexistencii štruktúry rozhovoru sa od znalostného inžiniera vyžadujú dobré konverzačné schopnosti, určitý tréning a schopnosti usmerňovať takýto rozhovor.

### Štruktúrovaný rozhovor

Od neštruktúrovaného rozhovoru sa líši rozsahom v akom je tento rozhovor viazaný nejakým vopred určeným formátom. Tento sa môže pohybovať od vysoko štruktúrovaného v ktorom obsah aj poradie udalostí sú dané až po semi-štruktúrovaný (nápovedný) v ktorom je obsah daný ale poradie sa môže meniť.

Výhodou takéhoto rozhovoru je existencia štruktúry, ktorá umožňuje dosiahnuť systematickejší spôsob pokrytia predmetnej oblasti. Naviac táto štruktúra robí rozhovor pohodlnejším nielen pre znalostného inžiniera, ale aj pre experta pretože požadovaná informácia je definovaná explicitnejšie.

Hlavným účelom je získať hlbší pohľad na štruktúru objektov v danej oblasti a vzťahov medzi nimi. Podobá sa teda viac na vypočúvanie ako na bežnú konverzáciu. Informácie sú vyžadované v oveľa väčšej hĺbke a špecifičnosti ako pri neštruktúrovanom rozhovore. Obsah rozhovoru je daný vopred pripravenou agendou, ktorá neobsahuje vopred pripravené špecifické otázky ale skôr sa podobá zoznamu námetov, ktoré je potrebné podrobne preskúmať. Od experta sa už neočakáva všeobecný popis ale sústredenie sa na znalosti ktoré sú viazané na špecifické udalosti.

Spracovaniu každého námetu môže predchádzať krátky úvod znalostného inžiniera, ktorý sumarizuje svoje momentálne chápanie daného námetu. Nie vždy sa podarí uzavrieť diskusiu na jeden námet počas jedného rozhovoru. Môže sa stať, že diskusia sa stane príliš zložitou pre momentálne chápanie znalostného inžiniera, prípadne sa v nej vyskytne príliš veľa neznámych termínov, rozširujúcich slovník znalostného inžiniera. Vtedy je vhodnejšie diskusiu na daný námet prerušiť a vrátiť sa k nej v niektorom z ďalších rozhovorov.

Existuje niekoľko variantov tohto typu rozhovoru. Jednou z nich je napr. sústredený rozhovor v ktorom obsah alebo námet interview sa sústreďuje okolo špecifického typu informácií (napr. ciele, pojmy, diagramy).

Použitie tohoto typu rozhovoru vyžaduje viac času na prípravu štruktúry rozhovoru. Od znalostného inžiniera sú vyžadované dobré znalosti z oblasti (získané napr. analýzou neštruktúrovaného rozhovoru) a to ako pre prípravu rozhovoru tak aj pre jeho analýzu.

### Sondy

Používajú sa v spojitosti s rozhovormi s expertmi. Sú to krátke zásahy do konverzácie, umožňujúce znalostnému inžinierovi túto konverzáciu ovplyvňovať. Existuje päť typov sond:

**Aditívna sonda** — Povzbudzuje experta k poskytnutiu ďalších informácií, naznačuje mu (verbálne alebo neverbálne) že to, čo doposiaľ povedal, znalostný inžinier pochopil. Sú to výzvy typu “Pokračujte”, “Povedzte mi o tom viac”, “Nezastavujte sa” ap. Tento typ sond sa používa napr. v rozhovore, keď sa vyskytne nový námet. Vtedy znalostný inžinier hľadá základné pochopenie námetu a je vďačný za každú ďalšiu informáciu.

**Reflexívna sonda** — Povzbudzuje experta (nepríkazovým spôsobom) pohrúžiť sa hlbšie do námetu, ktorý je práve diskutovaný. Používa sa vtedy, keď expert prestal hovoriť a nevyzerá, že by sa chystal ešte pokračovať v danom námete. Vtedy znalostný inžinier urobí syntézu expertovej predchádzajúcej odpovede a postaví ju ako otázku.

**Direktívna sonda** — Bez navrhovania určitého obsahu špecifikuje smer, ktorým by sa mala uberať ďalšia konverzácia s expertom. Napr. ak expert začne poskytovať príliš špecifické odpovede, potom tento typ sondy môže presunúť dôraz na všeobecnejšiu rovinu (neštruktúrovaný rozhovor) alebo prejsť z jedného námetu na iný (štruktúrovaný rozhovor). Príkladom takejto sondy je “Prečo je to (tento prípad)?”.

**Sonda zmeny módu** — Umožňuje znalostnému inžinierovi prechod na iné hľadisko (z abstraktného popisu na špecifickú funkčnosť objektov, z teoretického pohľadu na metodologický, ap.). Používa sa v štruktúrovanom rozhovore, keď znalostný inžinier je už široko oboznámený s oblasťou, takže je schopný zmysluplne vykonávať tieto prechody.

**Definičná sonda** — Vyžaduje od experta, aby vysvetlil význam či zmysel určitého pojmu. Kvôli plynulosti expertovho vysvetľovania sa zvyčajne tieto sondy odkladajú až na začiatok nasledujúceho rozhovoru. To však môže mať za následok značné sťaženie analýzy rozhovoru.

### Spätné učenie

Je to nielen technika získavania nových ale aj verifikácie už získaných znalostí. V tejto technike expert vysvetľuje niečo znalostnému inžinierovi, ktorý sa to potom snaží spätne vysvetliť expertovi. Procedúra pokračuje pokiaľ expert nie je spokojný s vysvetlením znalostného inžiniera. Môže byť kombinovaná s podobnou technikou, keď expert vypracuje prednášku pre znalostného inžiniera.

Znalostný inžinier hrá úlohu experta a vysvetľuje niektoré pojmy alebo riešenie úlohy v nejakých určitých podmienkach. Pokus vyjadriť získané znalosti z pozície experta môže odkryť v znalostiach znalostného inžiniera dvojznačné alebo chybné pochopenie niektorých znalostí, ktoré si znalostný inžinier neuvedomuje.

Táto technika súčasne umožňuje expertovi aby bol schopný kontrolovať aktuálny stav znalostnej bázy.

### Dvadsať otázok

Pri tejto metóde znalostný inžinier volí (v duchu) nejakú konkrétnu situáciu, diagnózu, poruchu, riešenie problému alebo stav. Expert sa snaží uhádnuť, ktorý špecifický pojem má znalostný inžinier na mysli. Pritom mu môže klásť otázky, na ktoré získava odpoveď iba v tvare "áno/nie".

Prostredníctvom expertových otázok znalostný inžinier môže získať informácie o sekvenciách riešiacich krokov, stratégiách, kontrastoch a používaných kategóriách.

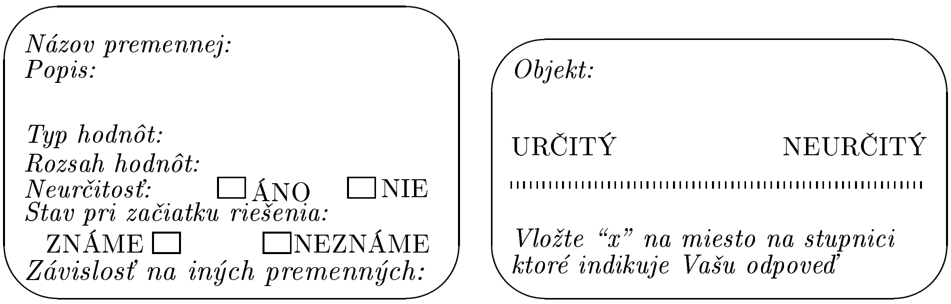
Úspech tejto techniky závisí na voľbe počiatočných pojmov. Zároveň od znalostného inžiniera sa vyžaduje úroveň znalostí dostatočne vysoká na to, aby vedel správne zodpovedať expertove otázky. Možným rozšírením metódy je využitie ďalšieho experta, ktorý hrá úlohu znalostného inžiniera.

### Dotazníky

Je to značne efektívny spôsob získavania znalostí. Nie je taký náročný na čas ako rôzne druhy rozhovorov (pritom sa uvažuje samozrejme náročnosť zo strany experta). Expert môže vyplniť dotazník v pohodlnej a uvoľnenej atmosfére vtedy, keď mu to najviac vyhovuje. Na druhej strane sa nehodí pre získavanie vopred nepredvídaných znalostí.

Dotazníky sú vhodné pre zisťovanie objektov v danej oblasti, v odkrytí vzájomných vzťahov medzi nimi a v určení neurčitosti jednotlivých odkrytých fragmentov znalostí. Z formálnej stránky to nie je zoznam otázok, ktorý sa bežne rozumie pod dotazníkom, ale otvorené otázky na pojmy, hodnoty, atribúty a relácie. Tieto otázky môžu mať rôznorodú formu, napr. expert môže byť tiež požiadaný načrtnúť diagram pre znázornenie relácie.

Môžu by realizované vo forme kartičiek, pričom každá kartička sa týka iba jedného objektu (alebo viacerých, ak ide o definovanie vzájomného vzťahu medzi nimi). Obr. 1.1 ilustruje tvar dotazníkovej kartičky pre zistenie vlastností nejakej premennej a pre zistenie neurčitosti nejakej znalosti.




Obr. 1.1: Príklad dotazníkových kartičiek

## 1.2 Prípadové štúdie

Je to skupina metód sústreďujúcich sa na jednotlivé prípady. Výber vhodných reprezentatívnych prípadov je kritickým faktorom pre dosiahnutie úspechu. Prípady sú vo všeobecnosti vyberané pretože sú zaujímavé, zriedkavé, nerutinné, ľahko zapamätateľné, zábavné alebo obtiažne – a teda existuje jasná tendencia zabúdať na rutinné prípady, ktoré vďaka svojej všednosti neutkvejú v pamäti experta. Na druhej strane, hypotetické prípady ktoré sú expertovi neznáme alebo sú mu cudzie nie sú vhodným prostriedkom pre ľahké získavanie expertových znalostí.

### Analýza prípadových štúdií

Analýza prípadových štúdií sa zameriava na špecifické skúsenosti experta, ktoré nadobudol pri riešení konkrétnych úloh. Je založená na predpoklade, že rozpamätávanie sa na nejaký konkrétny prípad umožní sprístupnenie aj takých informácií, ktoré by boli inak opomenuté (napr. počas nejakej formy rozhovoru medzi expertom a znalostným inžinierom). Táto analýza často obsahuje simulácie v ktorých znalostný inžinier postupuje spolu s expertom krok za krokom v hľadaní riešenia.

Metóda má niekoľko nevýhod. Keďže spočíva na expertovom vybavovaní si, ako v minulosti riešil určité úlohy, môže nastať situácia že analyzovaný prípad od experta vyžaduje veľmi staré spomienky. Toto vybavovanie môže časom podliehať deformáciám (vďaka chybám pamäti) a dodatočnej racionalizácii, kedy sa expert snaží o ich rekonštrukciu (ale na základe svojich momentálnych znalostí). Expert pri svojom vysvetľovaní môže použiť termíny, ktoré sú neznáme znalostnému inžinierovi. Ten môže žiadať ich vysvetlenie a tak prerušovať súvislý tok komentáru.

### Simulácia podľa scenára

V tejto technike znalostný inžinier poskytne náčrt prvotnej situácie a potom expert prechádza riešením prípadu daným načrtnutou situáciou. Toto riešenie môže byť alebo reprodukované (ak expert takýto prípad už riešil v minulosti) alebo konštruované (ak je tento prípad pre neho nový). Počas svojho riešenia sa zameriava na relevantné termíny a pojmy, a popisuje svoje riešiace kroky.

Je spojovacím článkom medzi štruktúrovaným rozhovorom a myslením nahlas. Vyžaduje od experta verbálne prechádzať riešiacimi procesmi, používanými k uskutočňovaniu jeho práce. Je to však viac interview než správa o vykonávaní práce.

Najväčším obmedzením metódy je to, že vyžaduje od znalostného inžiniera pomerne rozsiahle znalosti pre zostavenie počiatočnej situácie a jej opísanie expertovi. Preto sa niekedy používa dvojica expertov, kde jeden zostaví počiatočný popis problému a druhý sa snaží tento problém vyriešiť.

Znalostný inžinier potrebuje jednoduché opisy práce experta pri riešení problémov, vyhýbajúce sa prílišnej komplexnosti. Tento jednoduchý popis potom transformuje do základných metód, ktoré má k dispozícii a ktoré môže využiť pri implementácii prvotných návrhov. Expert musí vysvetliť čo najpodrobnejšie a zároveň čo najpochopiteľnejšie, čo vlastne robí a ako to robí pri svojom spôsobe riešenia úloh. Znalostný inžinier potom tento proces práce experta premieta do modelu procesu riešenia úloh.

Niekedy je obtiažne použiť túto metódu, pretože expert nie je zvyknutý vykladať svoje postupy spôsobom "... a teraz konštruujem hypotézu, ktorú mienim testovať voči podmienke ...".

### Metóda minimálneho scenára

Expertovi je predložený scenár alebo problém, v ktorom niektoré kritické informácie sú vynechané (pritom samotné riešenie problému nemusí byť vynechanou časťou). V priebehu riešenia je expert nútený pokúsiť sa tieto chýbajúce údaje doplniť (napr. doplniť chýbajúce symptómy ak je konfrontovaný iba s čiastočným zoznamom symptómov, výsledkov testov a nejakou diagnózou).

Metóda vyžaduje znalosti pre zostavenie takéhoto čiastočného scenára. Opäť je možné použiť dvojicu expertov.

### Dynamické spätné učenie

V dynamickej variante metódy spätného učenia znalostný inžinier vykonáva simuláciu expertovej práce. Toto nie je možné použiť skôr, až keď znalostný inžinier už dostatočne chápe riešenie úloh v danej oblasti.

Úloha, riešená znalostným inžinierom, by mala byť realistická simulácia, ale bez príliš veľkého počtu problémov a ťažkostí. Expert musí byť počas tejto simulácie k dispozícii, aby aspoň spočiatku navádzal znalostného inžiniera na správny postup. Znalostný inžinier je vlastne v pozícii experta pri myslení nahlas (toto je dôležité, pretože ináč by znalostný inžinier mohol odvodiť správne závery a vykonávať správne akcie na základe chybných predpokladov).

Počas simulovaného riešenia úlohy znalostný inžinier môže zároveň hrať rolu budúceho znalostného systému, stále sa sám seba pýtajúc, ako znalostný systém môže emulovať tú ktorú procedúru. Teda nielen že získané znalosti sa overujú na reálnom príklade, ale aj znalostný inžinier nadobúda jasnejšiu predstavu o implementácii týchto znalostí.

Expert je na prijímajúcej strane (teraz znalosti neposkytuje ale naopak hrá úlohu ich príjemcu), čo je podobné ako keby interagoval s prototypom znalostného systému.

### Verbálny off-line protokol

Oddeľuje sa fáza riešenia úlohy expertom od fázy jej komentovania. Najprv expert úlohu vyrieši bez toho aby bol počas riešenia rušený požiadavkami znalostného inžiniera. Toto jeho riešenie je nejakým spôsobom archivované. Retrospektívne sa vytvára komentár k postupu riešenia úlohy.

Expertovi sa pri komentovaní poskytne určitý stimulačný materiál (napr. video alebo audio záznam) o vykonávanom riešení danej úlohy. Expert iba komentuje svoju predchádzajúcu činnosť alebo činnosť niekoho iného. Je možné takýmto postupom komentovať aj chybné postupy a analyzovať chyby.

Pri komentovaní tým istým expertom, ktorý úlohu riešil, sa predpokladá, že informácie o riešení sú ešte stále aktívne v jeho krátkodobej pamäti. Pri dlhšej prestávke medzi oboma fázami hrozí expertova snaha o dodatočnú racionalizáciu jednotlivých riešiacich krokov.

## 1.3 Pozorovania

Patria sem metódy spojené s riešením špecifických prípadov, pričom tieto metódy prebiehajú paralelne s týmto riešením. Znalostný inžinier sa tak stáva účastníkom (pravda iba pasívnym) procesu riešenia úlohy.

### Pozorovanie

Expert vykonáva svoju prácu v aktuálnom prostredí alebo v prostredí tomuto aktuálnemu čo najpodobnejšom. Znalostný inžinier ho v tichosti pozoruje pri práci. Nie je to všeobecne aplikovateľná metóda – sú situácie keď detailné pozorovanie práce experta je problematické alebo nemožné.

Pozorovanie môže byť použité pre identifikáciu tých stratégií riešení problémov ktoré nie sú expertovi vedome dostupné, pre štúdium motorických schopností a automatických procedúr a pre verifikáciu expertovho popisu čo vlastne robí.

V podstate existuje päť typov úloh ktoré môžu byť cieľom pozorovania. Sú to:

- známe úlohy – jedná sa o úlohy expertovi známe (bežne ich rieši vo svojej praxi). Tieto úlohy sú vhodné pre generovanie prvotnej verzie bázy znalostí, pretože reprezentujú prevažnú väčšinu práce experta.
- simulované známe úlohy – ak úlohy, ktoré sú expertovi známe nemožno vykonávať napr. z toho dôvodu že ich vykonávanie je finančne náročné alebo hazardné, je možné ich simulovať použitím archívnych údajov.
- úlohy s ohraničenou informáciou – sú predkladané expertovi iba s časťou informácií ktoré sú typicky potrebné pre riešenie úlohy. Toto umožňuje pozorovať stratégie, ktoré by inak neboli objavené.
- úlohy s ohraničeným spracovaním – limitujú expertove možnosti vzhľadom na poskytnutý čas na riešenie alebo prijateľný typ riešenia.
- ťažké prípady – skutočne zložité (a zvyčajne iba zriedka sa vyskytujúce) úlohy. Umožňujú odhaliť skutočnú expertízu, pretože neumožňujú expertovi využiť rutinné postupy riešenia.

Táto technika obchádza nutnosť akýchkoľvek zásahov do činnosti experta (či už konverzáciou alebo zasahovaním do samotného riešenia úlohy), čím poskytuje expertovi možnosť riešiť úlohu takým spôsobom a v takom prostredí, na aké je zvyknutý. Jej nevýhodou je však to, že interpretácia údajov získaných z pozorovania môže byť zložitá. Znalostný inžinier preto už musí mať dobré znalosti o riešení úloh z predmetnej oblasti.

### Verbálny on-line protokol

Expert je požiadaný vykonávať svoju prácu. Jediný rozdiel oproti pozorovaniu je v tom, že súčasne pritom poskytuje verbálny report – slovne komentuje svoj postup riešenia problému (niekedy sa metóda označuje ako myslenie nahlas). Je to vhodná doplnková metóda, ktorá môže byť použitá ako základ pre štruktúrovaný rozhovor.

Podľa toho, koľkí experti sú použití, existujú dva základné druhy verbálnych reportov:

**samoreport** — expert súčasne rieši problém a poskytuje informácie o jednotlivých krokoch riešenia,

**tieňový report** — jeden expert rieši úlohu a druhý komentuje jeho postup riešenia.

Samoreport niekedy môže prekážať expertovi riešiť problém optimálnym spôsobom. Nemôže byť použitý ani v prípadoch, keď samotné riešenie problému zahŕňa verbálnu komunikáciu experta s okolím alebo keď na experta sú kladené vysoké kognitívne požiadavky.

Naproti tomu tieňová forma nekladie žiadne obmedzenia na riešiaceho experta. Na druhej strane predpokladom jej použitia je, že znalosti využívané oboma expertmi sa zhodujú v dostatočnej miere (teda druhý expert nekonštruuje vlastný postup riešenia ale vysvetľuje spôsob riešenia prvého experta).

Komplexnosť expertovej práce niekedy neumožňuje použitie metódy na celú jeho činnosť, ale tá musí byť segmentovaná do zvládnuteľných porcií. Toto delenie môže byť získané napr. z aplikácie myslenia nahlas na celú úlohu s úmyslom objaviť jej prirodzený rozklad.

Expert nie je vždy schopný vyjadriť svoje znalosti úplne. Preto zvyčajne je potrebný väčší počet opakovaní, s podstatnou duplicitou získaných znalostí. Kritickým je poskytovanie znalostí, ktoré sa expertovi zdajú samozrejmými, alebo ktoré stoja za expertovými rutinnými návykmi. Niekedy má expert sklon poskytovať knižné znalosti a nie tie, na základe ktorých daný problém skutočne rieši. Môže byť rozdiel medzi jeho direktívami, ako riešiť úlohu, a skutočným spôsobom jej riešenia.

Technika samoreportovania môže expertovi vadiť, pretože nemusí byť zvyknutý na podobný spôsob činnosti. Preto je niekedy vhodné vyskúšať niekoľko takýchto sedení, aby si zvykol na komentovanie svojej činnosti. Inou cestou je použitie niektorej modifikácie tejto techniky:

**Metóda kritickej odozvy** — Expert komentuje iba niektoré dôležité (kritické) časti úlohy. Jeho komentár musí začínať náčrtom situácie a postupu, ako sa dopracoval k danej časti riešenia.

**Metóda periodickej odozvy** — Používa sa vtedy, ak úloha je veľmi komplexná alebo expert má potiaže vykonávať úlohu a súčasne ju komentovať. Je stanovený určitý časový interval. Keď uplynie vymedzený čas, expert preruší svoju prácu a komentuje, čo za uplynulú periódu robil a čo sa chystá urobiť. Je potrebné zabezpečiť náväznosť jednotlivých komentárov. Voľba intervalu závisí na komplexnosti úlohy, s kratšími intervalmi pre komplexnejšie úlohy a naopak.

**Metóda prerušení** — Expert rieši úlohu bez vytvárania verbálneho protokolu až do chvíle, kým nie je požiadaný znalostným inžinierom o komentovanie svojej činnosti. To sa deje v momente, keď ten má problémy porozumieť aktuálne vykonávanému kroku riešenia.

## 1.4 Získavanie pojmov

Táto skupina metód je veľmi užitočná v počiatočných fázach návrhu, napr. v štádiu definovania parametrov úlohy. Vedie ku klasifikácii objektov danej predmetnej oblasti a k odhaleniu jej štruktúry. Toto následne napomáha lepšiemu pochopeniu spracovávanej oblasti, ale zvyčajne to nie je priamo relevantné pre implementáciu (štruktúra kategórií môže byť použitá ako základ objektovej štruktúry v báze znalostí, ak tá je implementovaná objektovým spôsobom).

### Priame získavanie pojmov

Metóda slúži pre získanie zoznamu pojmov, dôležitých pre riešenie problémov v nejakej doméne. Je to vlastne celá rodina metód majúcich veľmi podobné vlastnosti. Tieto metódy majú zvyčajne podobu štruktúrovaných rozhovorov. V rámci tejto skupiny možno rozpoznať techniky:

**zoznam pojmov** — expert je požiadaný vymenovať všetky pojmy, ktoré sú kritické,

**zoznam krokov** — je špecifickejší v tom, že expert je teraz požiadaný vymenovať všetky kroky, ktoré používa pri riešení úloh,

**zoznam kapitol** — expert vymenuje názvy kapitol a podkapitol hypotetickej knihy o predmetnej oblasti. Tieto názvy sú považované za reprezentatívne pojmy.

### Laddering

Metóda je zameraná na zisťovanie hierarchického usporiadania medzi pojmami, pričom do vytváranej hierarchie je možné zaraďovať ako už známe pojmy tak aj pojmy nové.

Použitie tejto metódy zahŕňa systematické generovanie nadradených pojmov, podradených pojmov a atribútov pomocou expertových odpovedí na špecifické otázky. Napríklad otázka typu "prečo" vedie k nadradenému pojmu, otázky typu "ako" a "čo" zase k podradenému pojmu. Expanzia pojmového grafu na rovnakej úrovni môže by dosiahnutá požiadaním experta o vymenovanie alternatívnych príkladov k už generovaným pojmom.

Pretože výsledkom metódy je taxonómia pojmov, táto metóda nie je vhodná pre problémové oblasti ktoré nie sú hierarchické.

### Triádové porovnávanie

Slúži pre získavanie nových pojmov na základe množiny už známych pojmov, pričom nové pojmy nejakým spôsobom charakterizujú tieto známe pojmy. Je to pomerne často používaná technika v spojitosti s nepriamymi technikami získavania znalostí (napr. s repertoárovou sieťou pre získavanie dimenzií - viď kap. 2).

Vychádza sa z existencie množiny $n$ pojmov. Znalostný inžinier vytvára z týchto pojmov trojice. Od cieľa aplikácie metódy závisí počet generovaných trojíc (ktorých je možné vytvoriť až $n(n-1)(n-2)/6$). Vytvorené trojice sú predkladané expertovi.

Expert v každej trojici zvolí ten pojem ktorý sa najviac odlišuje od ostatných dvoch (voľba hľadiska pre realizáciu tohto delenia je ponechaná na expertovi – ten ho volí v závislosti od svojho chápania predmetnej oblasti). Potom musí charakterizovať ostávajúce dva podobné pojmy tak, aby ich odlíšil od zvoleného pojmu. Táto charakteristika má typicky tvar jednej alebo viacerých vlastností ktoré sú dôležité pre rozlišovanie pojmov predmetnej oblasti (a súčasne sami sú pojmami používanými v danej doméne).

### Klasifikácia

Základom tejto metódy je triedenie, keď expert triedi objekty do skupín na základe ich príbuznosti, pričom nie je obmedzovaný počtom skupín alebo počtom objektov v skupine. Často sa od neho vyžaduje pomenovať jednotlivé skupiny po skončení triedenia. Obyčajne označí aj kritérium, ktoré použil pri zaraďovaní objektov do skupín.

Jednou z realizácií tejto metódy môže byť, že objekty, vymenované expertom, sa reprezentujú na kartičkách, pričom na každej kartičke je znázornený iba jeden objekt. Expert je potom požiadaný, aby tieto kartičky vhodne zoradil, alebo nejakým spôsobom usporiadal. Toto možno zopakovať niekoľkokrát. Zvyčajne stačí použiť jedného experta – získané výsledky bývajú dostatočne robustné (zisťuje sa či medzi objektami je nejaký vzťah alebo nie).

Existuje niekoľko verzií triediacej procedúry:

- expert opakuje triedenie s požiadavkou, aby sa aspoň jedna skupina nejakým spôsobom zmenila,
- počet skupín a počet objektov v skupine je ohraničený znalostným inžinierom,
- expert triedi najprv do dvoch skupín, potom do troch, štyroch atď.

Keď už boli identifikované význačné kategórie, presúva znalostný inžinier svoj záujem k definícii etalónov – objektov, ktoré reprezentujú typických členov jednotlivých kategórií. Mal by nasledovať štruktúrovaný rozhovor, aby boli získané informácie, viažúce sa k jednotlivým kategóriám. Zisťuje sa, ktoré atribúty objektov je potrebné uvažovať pre príslušnosť k tej-ktorej kategórii a ktoré atribúty slúžia pre vylúčenie príslušnosti objektu k danej kategórii.

## 1.5 Špeciálne techniky

Metódy tejto skupiny nie sú všeobecne použiteľné ale ich aplikovateľnosť je limitovaná vhodnou štruktúrou skúmanej oblasti a charakterom úloh, ktoré expert zvyčajne rieši. Často sú tieto metódy viazané na nejaký konkrétny spôsob reprezentácie znalostí a ich využitia pre riešenie problémov.

Ak však môžu byť aplikovateľné, ich použitie vlastne predstavuje výhodnú "skratku", umožňujúcu efektívnejšie zostaviť bázu znalostí ako použitím všeobecnejších metód. Na druhej strane ich nevhodné použitie vedie k získavaniu znalostí, ktorými sa expert v skutočnosti neriadi.

### Analýza rozhodnutí

Skúma sa proces prijímania riešení expertom. Metóda sa zameriava na vyhodnocovanie existujúcich alternatív a kvantitatívnu predikciu. Najprv je potrebné preveriť, či expert vôbec nejaké rozhodnutia prijíma, alebo či koná podľa nejakého predpisu. Potom sa analyzuje situácia, keď expert dané rozhodnutie prijíma:

- kedy sa to deje,
- aké sú hodnoty určitých parametrov,
- ako sú viaceré faktory navzájom vážené a kombinované,
- aké sú pravdepodobné následky alternatívnych rozhodnutí.

Popis takejto situácie môže byť formulovaný ako zoznam podstatných atribútov a pravdepodobné hodnoty týchto atribútov (tieto hodnoty môžu byť špecifické pre dané rozhodnutie, alebo môžu nastať pri viacerých možných rozhodnutiach).

Je potrebné zistiť, ktoré fakty sú relevantné a determinujúce v dobe, keď sa rozhodnutie prijíma. Situácia potom bude charakterizovaná poľom atribútov s korešpondujúcimi hodnotami. Prijímanie riešenia je vlastne výberom rozhodnutia z množiny aplikovateľných rozhodnutí. Pre každé takéto rozhodnutie existujú možné následky, každý z nich bude pritom charakterizovaný nejakou pravdepodobnosťou.

Získané znalosti sú silne subjektívneho charakteru. Znalostný inžinier je postavený medzi subjektivitou výpovedí experta a objektívnosťou, vyžadovanou formalizmami reprezentácie znalostí. Musí hľadať pravidlá, ktoré by odpovedali chovaniu experta. Pre modelovanie samotného rozhodnutia je možné:

- využiť nejaký štandardný prostriedok pre výber jednej alternatívy (príkladom je Bayesov teorém) a prvotné nastavenie pravdepodobností (získané napr. priamo od experta) modifikovať tak, aby výsledné rozhodnutie bolo zhodné s expertovým,

- použiť získané údaje bez modifikácie, aj keď niekedy rozhodnutie získané využitím nejakej štandardnej rozhodovacej metódy je iné ako to ktoré zvolil expert (typicky sa využíva regresný prístup v úlohách kde riešením je kvantitatívny odhad na jednorozmernej škále).

### Rozklad cieľov

Od experta sa vyžaduje aby riešenie úloh opísal analogicky k reťazeniu pravidiel. Znalostný inžinier stanoví cieľ a expert popisuje, aké úlohy je potrebné vykonať alebo ktoré podmienky je potrebné splniť, aby daný cieľ bol dosiahnutý. Celý problém sa takto postupne rozkladá na podproblémy, až na úroveň jednoduchých, ľahko realizovateľných problémov. Metóda je vhodná najmä pre systémy (a samozrejme aj úlohy), ktoré umožňujú spätné reťazenie pravidiel.

Pretože pre experta toto môže byť značne cudzia technika uvažovania, existuje veľké riziko získania umelej informácie, keď expertom zdelené znalosti nie sú presne tie, ktoré aktuálne používa pri riešení úloh (ale expert ich skonštruoval keď sa snažil použiť nanútený spôsob uvažovania). Silným predpokladom pre analýzu znalostí, získaných pomocou tejto techniky, je dobré porozumenie znalostného inžiniera úlohe. Metóda môže byť výhodne využitá v súvislosti so simuláciou podľa scenára a myslením nahlas.

### Systematické spájanie symptómov a diagnóz

Táto metóda je vhodná najmä pre triedu diagnostických úloh a môže byť použitá v prípade, že sú k dispozícii zoznamy všetkých možných diagnóz a všetkých ich symptómov. Expert je požiadaný opísať vzťahy medzi diagnózami a symptómami, pričom je možné systematicky preberať všetky možné kombinácie existujúcich porúch voči všetkým možným symptómom. Zvyčajným rozšírením metódy je, že expert zároveň uvádza informácie o možných riešeniach alebo akciách, ktoré môžu byť vykonané v týchto hypotetických podmienkach.

Účelom metódy je vybudovať štruktúru vzťahov medzi diagnózou, symptómami a riešením priamo pod expertovým vedením. Predpokladom je opäť dobré porozumenie znalostného inžiniera spracovávanej oblasti.

# Kapitola 2

# Nepriame metódy získavania znalostí

Nie vždy má expert prístup k detailom svojich znalostí alebo k svojim mentálnym procesom. Často chápe komplexné vzťahy a prichádza k záverom bez toho, aby exaktne vedel, ako to robí. V týchto prípadoch je nutné použiť niektorú z nepriamych metód.

Pri nich expert nevyjadruje svoje znalosti priamo, ale namiesto toho v pre neho prijateľnej forme. Znalostný inžinier potom z takejto reprezentácie môže získať to, čo expert musí vedieť, aby reagoval takým spôsobom, akým reaguje.

Väčšina nepriamych techník získavania znalostí je založená na existencii množiny pojmov z predmetnej oblasti a pre svoju činnosť vyžaduje nejaké odhady miery príbuznosti alebo podobnosti medzi doménovými pojmami (či už vyjadrené numericky, usporiadaním alebo prostredníctvom vzťahu k nejakému inému pojmu).

Z týchto odhadov sú potom odvádzané konečné reprezentácie (napr. štruktúra objektov a konceptov, používaných v danej oblasti). Pretože nie je možné spracovávať všetky pojmy vyskytujúce sa v predmetnej oblasti, je dôležitý ich výber - nejaká reprezentatívna množina.

Z uvedeného vyplýva že pri nepriamych metódach sa vždy jedná o kombináciu získavania prvotných dát od experta (ktoré je jednoduchšie ako priame získanie toho čo daná metóda produkuje – znalostí) a spracovania týchto dát.

Nepriame techniky majú určité obmedzujúce predpoklady o možnej forme reprezentácie objektov. Preto o tom, ktorá forma sa najlepšie hodí pre daný prípad, je potrebné rozhodnúť na základe interview s expertom.

## 2.1 Získavanie dát

Táto skupina metód sa zameriava na počiatočné získavanie odhadov podobnosti medzi zadanými pojmami. Tieto odhady majú typicky podobu matice podobnosti medzi pojmami. Ak riadky a stĺpce matice reprezentujú tie isté pojmy, potom matica obsahuje priamo podobnosti medzi dvojicami pojmov. V prípade, že riadky a stĺpce reprezentujú rôzne doménové pojmy, jedna skupina pojmov slúži na vyjadrenie vlastností inej skupiny. Matica potom obsahuje miery platnosti týchto vlastností pre skupinu pojmov, čím nepriamo vyjadruje podobnosť medzi nimi. Obe reprezentácie sú komplementárne - dáta reprezentované v jednej forme môžu byť transformované do druhej formy.

Maticová reprezentácia umožňuje kombinovať dáta viacerých expertov (vý7sledné dáta budú mať zmysel iba ak medzi týmito expertmi existuje dostatočná zhoda).

Úspech získavania vhodných odhadov podobnosti závisí od toho, či tieto odhady boli získavané vo vhodnom kontexte (dva pojmy sú si rôzne podobné pri rôznych hľadiskách), ktorý bol rovnaký pre všetky odhady, a či tieto odhady sú stabilné (teda či expert jasne interpretuje ohodnocované pojmy alebo či sa jeho interpretácia časom vyvíja).

### Párové porovnávanie

Je to jednoduchá metóda pre priame získavanie odhadov podobnosti medzi dvojicami pojmov. Táto miera podobnosti je:

- symetrická (pojem $P_A$ je podobný pojmu $P_B$ v rovnakej miere ako je pojem $P_B$ podobný pojmu $P_A$),
- diskrétna (teda sú definované stupne podobnosti vyjadrujúce možné podobnosti medzi pojmami),
- nezáporná (0 vyjadruje zhodnosť a čím väčšia hodnota tým menšia podobnosť).

Podmienka diskrétnosti nie je nutnou (miera podobnosti môže byť reprezentovaná aj napr. reálnymi číslami). Pretože však ľudské uvažovanie nie je vždy konzistentné pri práci s veľkým množstvom možností, použitie diskrétnej miery umožňuje redukovať počet prípustných možností.

Expertovi sú predkladané jednotlivé dvojice pojmov na ohodnotenie ich podobnosti. Takýmto spôsobom expert ohodnotí pre $n$ spracovávaných pojmov až $n(n-1)/2$ rôznych dvojíc. Vzhľadom na počet možných dvojíc je proces ohodnocovania časovo značne náročný. Preto sa používa iba pre malé počty pojmov.

Príklad výslednej (pol)matice podobnosti je v tab. 2.1.

| | koza | krava | ovca | prasa | kôň | pes | králik |
|---|---|---|---|---|---|---|---|
| koza | | 8 | 3 | 8 | 7 | 5 | 5 |
| krava | | | 8 | 3 | 8 | 14 | 11 |
| ovca | | | | 7 | 9 | 7 | 4 |
| prasa | | | | | 10 | 13 | 10 |
| kôň | | | | | | 11 | 12 |
| pes | | | | | | | 5 |
| králik | | | | | | | |

Tabuľka 2.1: Matica podobností dvojíc pojmov

V prípade že pojmy netvoria homogénnu množinu (teda nie pre každú dvojicu pojmov je možné určiť ich podobnosť), expert pre každý pojem najprv určí ktoré z ostatných pojmov sú s ním príbuzné. Ohodnocovanie podobnosti sa potom robí iba medzi týmito príbuznými pojmami. Na záver tým párom, ktoré nemajú priradenú podobnosť, sa táto určí ako nejaká veľká hodnota, reprezentujúca maximálnu nepodobnosť.

Expert pre generovanie stupňa podobnosti používa nejaké svoje vnútorné hľadisko, ktoré tvorí kontext celej metódy. Aby sa tento kontext počas celého získavania dát nemenil, je možné použiť postup, keď podobnosť každého páru pojmov je určovaná vo vzťahu k nejakému selektovanému páru pojmov, tvoriacemu referenčný štandard (napr. najviac alebo najmenej podobná dvojica).

### Repertoárová sieť

Táto metóda je založená na *teórii osobných konštruktov* vytvorenej Georgeom Kellym v 1955. Pôvodne bola vyvinutá pre použitie v klinickej psychoterapii. Je založená na tom, že indivíduum je schopné vyjadriť svoju štruktúru pojmov, na základe ktorých klasifikuje svoje okolie a nadväzne predvída budúce javy a riadi svoje jednanie. Túto svoju konceptuálnu štruktúru vonkajšieho sveta vie vyjadriť ako sériu bipolárnych vlastností, ktoré autor metódy nazýva *konštruktami.*

Jedinec chápe svet podľa rôznych systémov konštruktov. Preto rôzne indivíduá sú schopné rovnaké udalosti interpretovať rôzne. Kelly u svojich pacientov predpokladal, že ich problémy vysporiadať sa s okolitým svetom sú pravdepodobne spôsobené nejakou nekonzistenciou v ich sieti konštruktov. Vyvinul techniku, ktorá umožnila individuálnemu pacientovi verbalizovať svoj vlastný systém konštruktov, čo následne umožňovalo tento systém overovať a prípadne aj ovplyvňovať.

Každý konštrukt má individuálny význam. Tento je reprezentovaný dvoma opačnými pólmi, ktoré vyjadrujú medzné hodnoty konštruktu (budeme ich označovať podľa polohy ako *ľavý* a *pravý* pól). Konštrukt vlastne reprezentuje ordinálny atribút s dvomi možnými hodnotami (viď kap. 3). Každý konštrukt má obmedzený rozsah použitia, je aplikovateľný iba na obmedzenú množinu pojmov.

Miera príslušnosti nejakého konštruktu k pojmu leží medzi pólmi konštruktu. V najjednoduchšom prípade to môže byť iba jeden z pólov. V zložitejších situáciách je táto miera stupňovitá (krajné hodnoty sa zhodujú s pólmi). Zvyčajne počet stupňov je nepárny a teda existuje stredná hodnota s významom "rovnaký vplyv oboch pólov".

Repertoárová sieť reprezentuje model expertovej predstavy o predmetnej oblasti. Je to vlastne matica v ktorej pojmom zodpovedajú stĺpce a konštruktom riadky. Jednotlivé polia matice vyjadrujú mieru príslušnosti konštruktu k pojmu.

Jeden konštrukt vlastne reprezentuje ohodnotenie pojmov pozdĺž nejakej dimenzie. Pre každý zadaný konštrukt sa určia oba póly a všetky elementy sa ohodnotia celočíselnou váhou z intervalu $< 1, N >$ ($N$ je nepárne číslo, typicky býva 5), kde ohodnotenie 1 značí, že pre element úplne platí pravý pól konštruktu, a ohodnotenie $N$ zase vypovedá o úplnej platnosti ľavého pólu konštruktu. Ohodnotenie $(N + 1)/2$ označuje strednú hodnotu, keď rovnako platia oba póly. Ak niektorý element nie je ohodnotený pre nejaký konštrukt, znamená to, že daný konštrukt nie je zmysluplne použiteľný pre daný element.

Použité konštrukty môžu byť definované znalostným inžinierom. Ale keďže teória hovorí o "osobných" konštruktoch, je lepšie prenechať rozhodnutie na experta, ktoré konštrukty použije. Pre to možno použiť napr. triádové porovnávanie. Typicky nie sú vytvárané všetky možné trojice, ale iba tie, ktoré obsahujú také pojmy, ktoré v zadanom zozname pojmov susedia (teda pre $m$ pojmov bude vygenerovaných $m-2$ trojíc).

Príklad repertoárovej siete je uvedený v tab. 2.2.

| | | králik | pes | kôň | prasa | ovca | |
|---|---|---|---|---|---|---|---|
| veľkosť | veľká | 1 | 1 | 5 | 3 | 2 | malá |
| úžitkovosť | veľká | 3 | 1 | 2 | 5 | 3 | malá |
| rýchlosť | veľká | 3 | 4 | 5 | 2 | 2 | malá |

Tabuľka 2.2: Repertoárová sieť

V tabuľke konštrukt *veľkosť* bol inšpirovaný trojicou {králik, pes, kôň}, konštrukt *úžitkovosť* trojicou pojmov {pes, kôň, prasa} a konštrukt *rýchlosť* bol použitý pre rozlišovanie pojmov {kôň, prasa, ovca}. Všetky tri konštrukty majú rovnaké póly (pravý pól "malá" a ľavý pól "veľká"). Usporiadanie pólov (ktorý bude pravý a ktorý zase ľavý) a ich názvy môžu byť ľubovoľné.

Vytvorenú repertoárovú sieť je možné analyzovať niekoľkými spôsobmi. Výsledky analýzy sú predkladané expertovi na posúdenie. Ak ten nie je s nimi spokojný, má možnosť sieť modifikovať (meniť váhy pojmov pre jednotlivé konštrukty, dopĺňať alebo vypúšťať konštrukty, dopĺňať alebo vypúšťať pojmy, zlučovať konštrukty navzájom ap.). Ak je s výsledkami analýzy spokojný, získaná repertoárová sieť môže byť použitá ďalšími metódami.

Najjednoduchšími metódami analýzy je výpočet podobnosti (je to vlastne transformácia na reprezentáciu pomocou matice podobností, ktorej každý prvok priamo udáva podobnosť dvoch objektov). Sú možné dva prípady - podobnosť pojmov a podobnosť konštruktov. V oboch sa podobnosť určuje v percentách a je porovnávaná voči nejakej prahovej hodnote. Pri prekročení prahu je potrebné modifikovať repertoárovú sieť (lebo neobsahuje dostatočnú informáciu pre dostatočné odlíšenie pojmov navzájom vzhľadom na stanovený prah odlíšenia). V oboch prípadoch je potrebné zaviesť metriku v priestore konštruktov a pojmov.

**Podobnosť pojmov** .

Dva pojmy sú si najpodobnejšie (100%) vtedy, keď oba príslušné stĺpce repertoárovej siete obsahujú rovnaké hodnoty. Najmenej podobné sú vtedy (0%), keď pre každý konštrukt v jednom stĺpci je hodnota 1 a v druhom $N$. Podľa tejto úvahy pre podobnosť pojmov $P_k$ a $P_l$ vychádza vzťah

$$100 - \frac{100}{(N-1)PK} \sum_{j=1}^{PK} \mid v_{jk} - v_{jl} \mid \tag{2.1}$$

kde $PK$ je počet všetkých konštruktov, vyskytujúcich sa v sieti a $v_{ij}$ je váha j-teho pojmu vzhľadom na i-ty konštrukt.

Pre náš príklad z tab. 2.2 pre pojmy kôň a prasa platia váhy vzhľadom na jednotlivé konštrukty:

| | | $C_1$ | $C_2$ | $C_3$ |
|---|---|---|---|---|
| $P_3$: | kôň | 5 | 2 | 5 |
| $P_4$: | prasa | 3 | 5 | 2 |

Keďže $N = 5$ a sú známe tri konštrukty, kôň a prasa sa podobajú na 33%.

**Podobnosť konštruktov** .

Na rozdiel od predošlého prípadu teraz vystupuje ešte problém určenia, ktoré póly konštruktov patria k sebe (je možné malú veľkosť spojiť s veľkou rýchlosťou a veľkú veľkosť s malou rýchlosťou, avšak nič nebráni tomu, aby toto spojenie bolo práve opačné).

Teraz najväčšia možná odlišnosť medzi konštruktami sa zmenšila iba na $[(N+1)/2-1] * PP$, kde PP je počet pojmov. Ak totiž pri určitom priradení pólov vychádza odlišnosť medzi konštruktami väčšia, potom stačí zmeniť priradenie konštruktov (u jedného z nich prevrátiť póly) a odlišnosť bude menšia ako udaná hraničná hodnota. Na základe uvedeného pre podobnosť konštruktov $C_k$ a $C_l$ vychádza vzťah

$$100 - \frac{100}{(\frac{N+1}{2}-1)PP} \min\left( \sum_{j=1}^{PP} \mid v_{kj} - v_{lj} \mid, \sum_{j=1}^{PP} \mid v_{kj} - v_{lj}^{r} \mid \right) \tag{2.2}$$

kde horný index $r$ označuje výmenu pólov konštruktu.

Pre príklad z tab. 2.2 pre konštrukty veľkosť a úžitkovosť platia váhy vzhľadom na jednotlivé pojmy:

| | | $P_1$ | $P_2$ | $P_3$ | $P_4$ | $P_5$ |
|---|---|---|---|---|---|---|
| $C_1$ | veľkosť | 1 | 1 | 5 | 3 | 2 |
| $C_2$ | úžitkovosť | 3 | 1 | 2 | 5 | 3 |

Keďže $N = 5$ a je známych päť pojmov, uvedené konštrukty sa podobajú na 20% (pri výpočte bol spojený pól veľká veľkosť s pólom veľká úžitkovosť).

## 2.2 Štrukturálna analýza

Patrí sem skupina metód, ktorých cieľom je získať predstavu o štrukturálnych vzťahoch medzi pojmami. Tieto môžu mať rôznu podobu napr. hierarchické usporiadania, globálna podobnosť či typológia vzťahov medzi dvojicami pojmov. Tým že metódy umožňujú zistiť že medzi nejakou skupinou pojmov existuje nejaký vzťah, reprezentujú odrazový mostík k popisu a charakterizácii tohto vzťahu expertom. Metódy z tejto skupiny smerujú k systematickému zachyteniu všetkých štrukturálnych vzťahov medzi skúmanými pojmami.

### Mnohorozmerné škálovanie

Metóda vychádza z matice podobností dvojíc pojmov a generuje rozloženie týchto pojmov v k-rozmernom priestore. Používa sa iba pre tie dáta, ktoré môžu byť zmysluplne reprezentované v tomto mnohorozmernom priestore. Metóda je vhodná pre odkrytie zhlukov pojmov alebo relácií podobnosti.

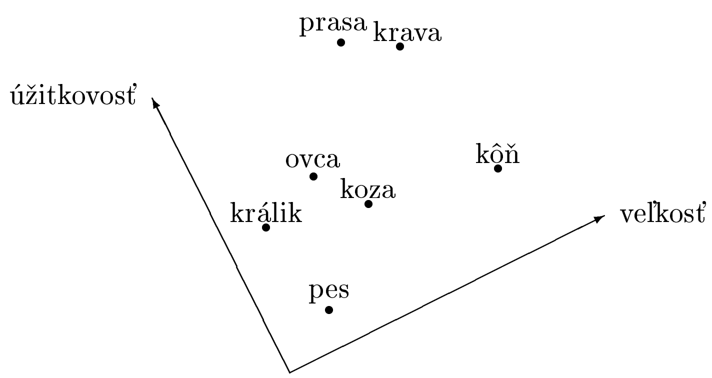
Vytvorená polmatica je vstupom pre hľadanie najlepšieho rozloženia objektov v priestore (pritom rozmernosť priestoru môže byť vopred špecifikovaná znalostným inžinierom) s čo najmenšou odchýlkou od skutočného stavu.

Pre voľbu hodnoty $k$ (dimenzionalita cieľového priestoru) možno využiť nejakú empirickú heuristiku. Jedna taká je založená na predpoklade, že so zvyšujúcou sa hodnotou $k$ chyba rozmiestnenia pojmov v k-rozmernom priestore voči vstupným dátam klesá, pričom sprvoti je tento pokles prudký a neskôr sa zvoľňuje. Preto táto heuristika radí voliť takú hodnotu $k$, za ktorou už zlepšenie by bolo minimálne.

Je možné vytvoriť niekoľko takýchto rozložení, v závislosti na predpísanej dimenzionalite priestoru zobrazenia. Znalostný inžinier z nich potom vyberie najvhodnejšie rozloženie (pri zohľadnení dimenzie použitého priestoru a dosiahnutej chyby).

Pre zvolené riešenie potom expert navrhne polohu osí a ich možné označenie (voľba dimenzionality teda závisí aj od následnej interpretácie). Tieto súradné osi odrážajú vlastnosti pozdĺž ktorých sa pojmy menia (môžu byť podľa týchto vlastností usporiadané). Tieto vlastnosti majú vo všeobecnosti spojitý charakter.

Príkladom použitia metódy môže byť analýza dát z tab. 2.1, ktorá je zobrazená na obr. 2.1 (pre prípad $k = 2$).




Obr. 2.1: Rozloženie pojmov v priestore

Použitie metódy je jednoduché, obtiažnejšie je iba objaviť vhodnú dimenziu, zvoliť a interpretovať jednotlivé osi. Zostavenie rozmiestnenia pojmov v k-rozmernom priestore je rutinnou záležitosťou (samozrejme pre výpočtovú techniku a nie pre znalostného inžiniera s tužkou v ruke).

## Johnsonovo hierarchické zhlukovanie

Podobne ako v mnohorozmernom škálovaní sa vychádza z polmatice podobností. Podobnosť určuje, na ktorej úrovni abstrakcie možno považovať dané pojmy za rovnaké (za členy tej istej kategórie). Je to "výška", v ktorej sa pojmy stávajú členmi tej istej nadradenej kategórie.

Metóda slúži na zoskupovanie pojmov podľa príbuznosti do zhlukov. Redukuje vstupnú množinu dát na hierarchickú množinu zhlukov. Predpokladá sa, že každý pojem je alebo nie je členom nejakého zhluku.

Štartuje sa z polmatice podobností a hľadajú sa pojmy, ktoré sú navzájom najpodobnejšie. Tieto pojmy sa spájajú do jedného zhluku, ktorý v ďalšom bude vystupovať ako jeden objekt. Algoritmus končí vytvorením hierarchickej reprezentácie podobnosti všetkých pojmov.

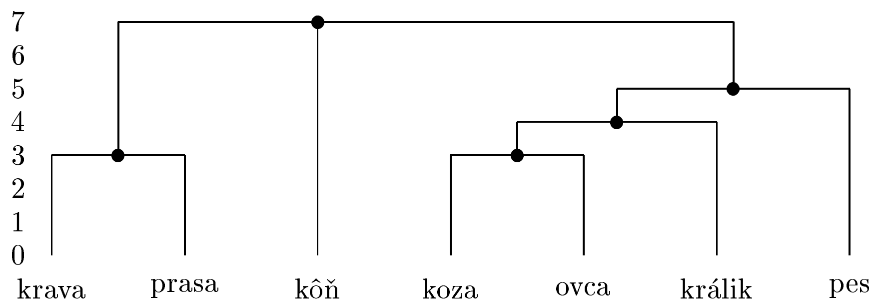
Po vytvorení každého zhluku pojmov sa konštruuje nová polmatica podobností. V nej podobnosti tých pojmov, ktoré v predchádzajúcom kroku neboli spojené do jedného zhluku, ostávajú nezmenené. Podobnosť novovytvoreného zhluku voči ostatným pojmom alebo zhlukom pojmov sa určí ako minimum (možno použiť aj inú operáciu, napr. maximum, priemer, ap. – vždy však iba jednu) podobností každého člena vytvoreného zhluku voči danému objektu.

Tab. 2.3 ukazuje aktuálny stav matice podobnosti po dvoch krokoch, keď boli vytvorené dva zhluky, každý z nich obsahujúci dva pojmy.

| | koza + ovca | krava + prasa | kôň | pes | králik |
|---|---|---|---|---|---|
| koza + ovca | | 7 | 7 | 5 | 4 |
| krava + prasa | | | 8 | 13 | 10 |
| kôň | | | | 11 | 12 |
| pes | | | | | 5 |
| králik | | | | | |

Tabuľka 2.3: Tvorba zhlukov pojmov

Obr. 2.2 zobrazuje výslednú hierarchiu pojmov (vrátane kvantitatívnej miery nepodobnosti).




Obr. 2.2: Hierarchická štruktúra pojmov

Výhodou metódy je to, že znalostnému inžinierovi stačí ceruzka a papier. Nevýhodou je opäť veľký počet vyžadovaných podobností. Naviac výsledok zhlukovania silne závisí od toho, aká operácia sa uvažuje pre výpočet nových podobností. Rôzne spôsoby voľby môžu poskytovať rôzne hierarchie. Tento výber je ľubovoľný a závisí na znalostnom inžinierovi.

Metóda vychádza z predpokladu, že pojmy sú navzájom rozlíšiteľné pomocou diskrétnych vlastností. Z tohto dôvodu môže byť použitá aj pre skupinu heterogénnych pojmov.

## Implikačná analýza

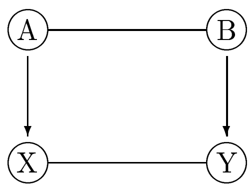
Reprezentuje špeciálny typ analýzy dát v repertoárovej sieti. Táto etapa analýzy repertoárovej siete v pôvodnom chápaní bola cieľom (analyzoval sa vnútorný systém konštruktov pacienta). Pri použití v znalostnom inžinierstve sa stáva iba prostriedkom získania predstavy o vzájomných vzťahoch medzi konštruktami a tým i kontroly znalostí, uložených v repertoárovej sieti.

Metóda odkrýva závislostné vzťahy medzi jednotlivými konštruktami v tvare implikácií. Ak sú dané dva rôzne konštrukty $A - B$ a $X - Y$ (kde $A$, $B$, $X$ a $Y$ sú ich póly), potom medzi nimi môžu existovať nasledovné základné vzťahy:

**Paralelné** — $A$ implikuje $X$ a $B$ implikuje $Y$

Pr.:

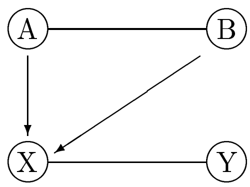
milovaný - nenávidený
príjemný - nepríjemný




**Ortogonálne** — $A$ implikuje $X$ ale $B$ neimplikuje $Y$

Pr.:

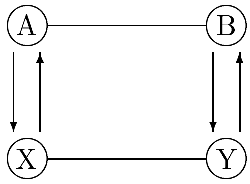
zamestnaný - nezamestnaný
prijíma potravu - neprijíma




**Recipročné** — $A$ implikuje $X$ a $B$ implikuje $Y$, a zároveň to platí aj v opačnom smere ($X$ implikuje $A$ a $Y$ implikuje $B$)

Pr.:

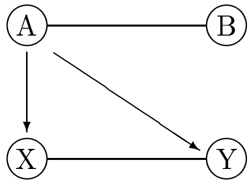
| | | |
|---|---|---|
| nervózny | - | pokojný |
| napätý | - | uvoľnený |



**Dvojznačné** — $A$ implikuje $X$ a zároveň $A$ implikuje $Y$

Pr.:

| | | |
|---|---|---|
| realizmus | - | idealizmus |
| žiadúci | - | nežiadúci |



V prípade recipročného vzťahu ide vlastne o funkčnú ekvivalentnosť konštruktov a je teda zbytočná prítomnosť oboch (možno ich zlúčiť do jedného). V dvojznačnom prípade subjekt má nekompletnú predstavu o rozdiele medzi danými konštruktami v danom kontexte, alebo má jeden pojem pre rôzne konštrukty. Je zrejmé, že to je nežiadúci prípad a je potrebné takúto závislosť z repertoárovej siete odstrániť.

Ak sa v nejakom prípade vyskytnú charakteristické znaky viacerých základných implikačných vzťahov, bude sa uvažovať nepriaznivejší prípad.

Pre dva konštrukty $A - B$ a $X - Y$ je vlastne potrebné preveriť osem implikácií: $A \to X$, $A \to Y$, $B \to X$, $B \to Y$, $X \to A$, $X \to B$, $Y \to A$ a $Y \to B$.

Pre zisťovanie napr. implikácie $A \to X$ je potrebné zistiť, či hodnoty z "$A$" konca prvého konštruktu implikujú hodnoty z "$X$" konca druhého konštruktu. Preto sa skúmajú všetky tie stĺpce repertoárovej siete, kde prvý konštrukt nadobúda hodnotu z konca "$A$" (teda hodnoty z intervalu $< 1, (N+1)/2-1 >$ pre pravý pól konštruktu resp. hodnoty z $< (N+1)/2+1, N >$ pre ľavý pól konštruktu) a porovnávajú sa voči hodnotám druhého konštruktu.

Sila implikácie $A \to X$ sa pre jeden stĺpec určí podľa tab. 2.4 (podľa tabuľky je možné určovať implikáciu iba medzi ľavými stranami konštruktov - v prípade potreby je ešte pred výpočtom sily implikácie potrebné konštrukty vhodne otočiť).

Výsledná sila implikácie sa potom určí ako priemer hodnôt pre jednotlivé stĺpce. Ak táto sila dosiahne zadanú prahovú hodnotu, tak implikácia sa považuje za platnú. V opačnom prípade sa neuvažuje.

| | | X | | | | |
|---|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 | 5 |
| A | 4 | 0.25 | 0.5 | 0.75 | 1 | 1 |
| | 5 | 0 | 0.25 | 0.5 | 0.75 | 1 |

Tabuľka 2.4: Sila implikácie

Jednotlivé implikácie sú predkladané expertovi na posúdenie. Ak expert nesúhlasí s nejakou implikáciou, môže modifikovať repertoárovú sieť, napr. zmeniť ohodnotenie pojmov pre dané konštrukty alebo môže zaviesť nový pojem, ktorý je výnimkou k nevhodnej implikácii.

Pre príklad z tab. 2.2 pre konštrukty rýchlosť a veľkosť platia jednotlivé implikácie:

| | | | | |
|---|---|---|---|---|
| veľká rýchlosť | −> | veľká veľkosť | : | 0.625 |
| veľká rýchlosť | −> | malá veľkosť | : | 0.5 |
| malá rýchlosť | −> | veľká veľkosť | : | 0.625 |
| malá rýchlosť | −> | malá veľkosť | : | 0.875 |
| veľká veľkosť | −> | veľká rýchlosť | : | 1 |
| veľká veľkosť | −> | malá rýchlosť | : | 0 |
| malá veľkosť | −> | veľká rýchlosť | : | 0.583 |
| malá veľkosť | −> | malá rýchlosť | : | 0.583 |

a teda vzťah (napr. pri prahu 0.6) medzi konštruktami rýchlosť a veľkosť môže byť považovaný za dvojznačný.

### Usporiadané stromy

Táto technika predpokladá, že informácie v ľudskej pamäti sú uchovávané v zhlukoch. Stavia na tom, že ľudia pri používaní svojej pamäti vyberajú informácie (ak existuje viacero alternatív) najprv z jedného zhluku a až potom prechádzajú na informácie, ktoré sú uložené v inom zhluku. Vychádza sa pritom z predpokladu, že organizácia pamäti je rôzna pre experta aj pre nováčika v danom odbore.

Expert je požiadaný, aby niekoľkokrát (zvyčajne sa používa 10 - 20 opakovaní) vymenoval pojmy, relevantné ku skúmanej oblasti. Najmä pri väčších počtoch pojmov je expert zaťažovaný oveľa menej ako pri iných spôsoboch získavania potrebnej informácie o pojmoch.

Pre zabezpečenie variability je mu povedané, ktorým pojmom má začať. Tab. 2.5 ilustruje štyri takéto vymenovania siedmich pojmov, každé začínajúce iným pojmom.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1. | pes | králik | ovca | koza | kôň | krava | prasa |
| 2. | prasa | krava | koza | ovca | králik | kôň | pes |
| 3. | kôň | pes | ovca | koza | králik | prasa | krava |
| 4. | pes | králik | krava | prasa | kôň | ovca | koza |

Tabuľka 2.5: Vymenovania pojmov

Získané zoznamy pojmov sú v ďalšom skúmané na pravidelnosť. Hľadajú sa v nich zhluky pojmov, ktoré sa často vyskytujú pri sebe, či už stále v tom istom poradí, v opačnom poradí alebo prehádzane. Z nájdených zhlukov sa potom zostrojuje hierarchická štruktúra.

Niektoré pojmy sa vyskytujú spolu skoro stále, iné menej. Vo všeobecnosti platí, že čím menej krát sa nejaké pojmy vyskytli spolu (alebo dostatočne blízko seba) tým sú menej podobné a teda môžu byť členmi toho istého zhluku iba pri väčšej abstrakcii. Napr. v našom príklade pojmy koza a ovca budú tvoriť bezprostredne jeden zhluk pretože sa stále vyskytujú spolu. Spolu s nimi (teda buď s jedným alebo druhým pojmom) sa v troch prípadoch vyskytol králik. To znamená, že môže vytvárať s nimi spolu zhluk – ale už to nie je bezprostredný zhluk (nevytvára zhluk priamo s nimi ale s ich zhlukom).

Pretože počet vymenovaní je obmedzený, výsledkom analýzy môže byť viacero rovnocenných interpretácií. Jedna takáto štruktúra, zodpovedajúca nášmu ilustračnému príkladu je na obr. 2.2.

Analýza zhlukov môže byť síce realizovaná ručne, ale pre zložitosť úlohy pri veľkom počte pojmov je únavná a citlivá na chyby v dátach (napr. expert pri vymenovaní na nejaký pojem jednoducho zabudne). Výhodnejšia je počítačová analýza, ktorá súčasne umožňuje zisťovať, ako sa mení skúmaná hierarchická štruktúra pri neuvažovaní niektorých vymenovaní objektov.

## 2.3 Odvodzovanie kompilovaných znalostí

Štrukturálne znalosti o pojmoch predmetnej oblasti môžu síce pomôcť znalostnému inžinierovi v lepšom pochopení skúmanej oblasti a tým v ďalšom získavaní znalostí ale nedajú sa priamo využiť ako procedurálne znalosti pre riešenie úloh.

Preto táto skupina metód sa zameriava na získavanie takýchto znalostí. Zvyčajne takto získané znalosti majú tvar produkčných pravidiel ktoré kompilujú zložité závislosti medzi pojmami do tvaru jednoduchých asociácií (a tým vlastne skrývajú ich skutočnú štruktúru). Sú veľmi podobné metódam induktívneho učenia. Jediný rozdiel je ten, že nevychádzajú z typických príkladov.

### Generovanie pravidiel z repertoárovej siete

Repertoárová sieť môže byť využitá pre vygenerovanie znalostí v tvare produkčných pravidiel. Keďže obsahuje dva typy elementov (pojmy a konštrukty), kde jeden typ vždy slúži na vzájomné odlíšenie prvkov druhého typu, tak jeden typ elementov bude predstavovať predpoklady generovaných pravidiel a druhý zase ich závery. Zvyčajne úlohu záverov plnia diferencované pojmy a úlohu predpokladov zase jednotlivé póly konštruktov.

Ak v repertoárovej sieti je každý pojem ohodnotený vzhľadom na každý konštrukt, potom pre každý záver bude existovať rovnaký počet pravidiel, zhodný s počtom všetkých pólov.

Ku každej dvojici pojem–konštrukt je možné vygenerovať dve pravidlá tvaru:

| | | | |
|---|---|---|---|
| AK | ľavá strana konštruktu | POTOM | pojem |
| AK | pravá strana konštruktu | POTOM | pojem |

Takýmto postupom je možné vygenerovať k repertoárovej sieti $2 * PP * PK$ (dvojnásobok súčinu počtu pojmov a počtu konštruktov) pravidiel. Vzniknutá báza znalostí bude iba dvojstupňová (úroveň vlastností a úroveň záverov), jednotlivé vygenerované pravidlá sa nebudú dať reťaziť. Týmto postupom teda nie je možné získať hierarchicky štruktúrované znalostné bázy.

Vygenerované pravidlá nebudú absolútne platné, ale budú mať priradenú nejakú neurčitosť, vyjadrujúcu mieru ich platnosti. Pri stanovovaní tejto neurčitosti je potrebné vziať do úvahy tri rôzne faktory, pričom výsledná váha pravidla sa určí ako súčin hodnôt týchto faktorov, prislúchajúcich danému pravidlu.

1. Spôsob narábania s neurčitosťou v cieľovom systéme (F1)

   Pri použití spôsobu narábania s neurčitosťou zo systému EMYCIN (činiteľ istoty) sa z $n$ neurčitých svedectiev vytvára jedno podľa vzťahu

$$1 - (1 - w_1)(1 - w_2) \ldots (1 - w_n) \tag{2.3}$$

kde $w_i$ je váha i-teho svedectva. Toto skladanie má akumulačný charakter, kde maximálnou hodnotou výsledku skladania je hodnota 1. Na jej dosiahnutie (v rámci určitej malej tolerancie) pri $n$ rovnakých svedectvách stačí, ak tieto svedectvá budú mať hodnoty podľa

$$F1 = w_i = 1 - \sqrt[n]{1 - H_\Theta} \tag{2.4}$$

kde $H_\Theta$ je zvolená hodnota blízka 1 (napr. 0.95).

Napr. pri skladaní piatich svedectiev s váhou 0.55 sa získa výsledná hodnota väčšia než 0.98, prakticky znamenajúca absolútnu platnosť. Teda z hľadiska výslednej neurčitosti nie je rozdiel medzi prípadom, keď svedectvá majú neurčitosť iba 0.6 a prípadom, keď sú absolútne platné (s váhou 1). Vzniká tak akési pásmo necitlivosti, do ktorého môže padnúť výsledná hodnota neurčitosti, ak váha jednotlivých svedectiev nadobudne hodnotu väčšiu ako hodnota podľa vzťahu (2.4). Aby sa práca s neurčitosťou vyhla tomuto pásmu necitlivosti, hodnoty F1 sú maximálne váhy, ktoré môžu pravidlá nadobúdať.

2. Váha pojmu vzhľadom na nejaký konštrukt (F2)

   Ohodnotenie jednotlivých elementov vzhľadom na nejaký konštrukt je číslo z intervalu $< 1, N >$, kde $N$ značí úplný súhlas s ľavým pólom konštruktu a 1 značí úplný nesúhlas s týmto pólom. Toto je potrebné transformovať na interval, vyjadrujúci neurčitosť v cieľovom systéme. V prípade použitia intervalu $< -1, 1 >$ je možné použiť vzťah

$$F2 = \frac{x - \frac{N+1}{2}}{\frac{N-1}{2}} \tag{2.5}$$

   kde $x$ je ohodnotenie pojmu vzhľadom na konštrukt.

3. Dôležitosť konštruktov (F3)

   Dôležitosť konštruktov je možné vyjadriť ako váhu konštruktu ktorá sa zvyčajne hodnotí číslom 0 až $D_{max}$, kde $D_{max}$ je najdôležitejší konštrukt. Ak nejaký konštrukt je ohodnotený hodnotou $D_{max}$, uvažuje sa bez obmedzenia. Ak jeho dôležitosť je nižšia, mal by poklesnúť aj jeho vplyv na konečné rozhodnutie (teda vygenerované pravidlo by malo mať nižšiu váhu). Ak je dôležitosť konštruktu nulová, znamená to, že na konečné rozhodnutie nemá žiadny vplyv (a teda vygenerované pravidlo by malo mať váhu 0). Tomuto zodpovedá

$$F3 = \frac{y}{D_{max}} \tag{2.6}$$

kde $y$ je teraz dôležitosť konštruktu.

Ak sú známe hodnoty všetkých troch faktorov, potom sa váha pravidla (reprezentovaná jeho váhou) určí ako ich súčin.

Uvažujme v príklade z tab. 2.2 dôležitosť 5 (nech to je maximálna dôležitosť) pre konštrukty veľkosť a úžitkovosť a 4 pre konštrukt rýchlosť (teda to nie je až taká podstatná vlastnosť). Keďže v príklade sú tri konštrukty (nech $H_\Theta$ je taká, že $F1 = 0.75$), voľba prasaťa bude podporovaná pravidlami

| | | | | |
|---|---|---|---|---|
| AK | *úžitkovosť* = *veľká* | POTOM | prasa | (0.75) |
| AK | *rýchlosť* = *malá* | POTOM | prasa | (0.3) |

a popieraná pravidlami

| | | | | |
|---|---|---|---|---|
| AK | *úžitkovosť* = *malá* | POTOM | prasa | (-0.75) |
| AK | *rýchlosť* = *veľká* | POTOM | prasa | (-0.3) |

Pravidlá

| | | | | |
|---|---|---|---|---|
| AK | *veľkosť* = *veľká* | POTOM | prasa | (0) |
| AK | *veľkosť* = *malá* | POTOM | prasa | (0) |

sú neutrálne.

# Kapitola 3

# Induktívne učenie

Pri induktívnom učení sa nevychádza priamo z expertových znalostí, ktoré používa pre riešenie problémov z danej problémovej oblasti, ale z konkrétnych prípadov riešenia (bez toho, aby bol známy postup, ako expert došiel k tomuto riešeniu). Znalosti o konkrétnych prípadoch sú reprezentované pomocou *typických príkladov*. Každý takýto typický príklad má dve časti:

1. dátová časť

   obsahuje fakty o danom prípade, ktoré ho popisujú,

2. rozhodovacia časť

   slúži pre klasifikáciu daného prípadu, pre rozhodnutie o činnosti ap. Obsahuje vlastne to, čo je výsledkom expertovej riešiacej činnosti v situácii, popísanej dátovou časťou príkladu.

Vzhľadom na charakter rozhodovacej časti sú metódy induktívneho učenia vhodné iba pre úlohy, pri ktorých je vopred známa množina možných riešení a riešiť takúto úlohu znamená vybrať to riešenie, ktoré sa pre danú úlohu hodí najlepšie (teda je potrebné klasifikovať úlohu do nejakej triedy, pričom každá možná klasifikačná trieda reprezentuje práve jedno riešenie).

Fakty o nejakom riešenom prípade sú vyjadrené pomocou atribútov, ktoré nadobúdajú nejaké hodnoty. Podľa toho, aké hodnoty môže atribút nadobúdať, možno rozlišovať atribúty:

- binárne - dvojprvková množina možných hodnôt (hodnoty “áno” a “nie”),

- nominálne - konečná neusporiadaná množina hodnôt (napr. množina {zem, oheň, voda, vzduch}),
- ordinálne - konečná usporiadaná množina hodnôt (napr. farby usporiadané podľa svojej vlnovej dĺžky),
- reálne - nekonečná množina hodnôt.

Nie každý algoritmus induktívneho učenia môže pracovať s ľubovoľným typom atribútov. My sa budeme v ďalšom zaoberať iba prípadom nominálnych atribútov.

Uveďme ilustračný príklad (v ďalšom bude využívaný pre ilustráciu jednotlivých algoritmov), keď dátová časť typických príkladov bude popísaná pomocou troch atribútov "výška" (výška postavy človeka), "vlasy" (farba jeho vlasov) a "oči" (farba jeho očí). Tieto atribúty môžu nadobúdať hodnoty:

$$\begin{array}{lcl} \text{výška} & = & \{ \text{ vysoký, nízky } \} \\ \text{vlasy} & = & \{ \text{ blond, tmavé, červené } \} \\ \text{oči} & = & \{ \text{ modré, hnedé } \} \end{array}$$

Rozhodovacia časť bude reprezentovaná príslušnosťou do jednej z dvoch tried (trieda "+" alebo trieda "-"), deliacich osoby, opísané pomocou uvedených troch atribútov, na príťažlivé a nepríťažlivé. Tab. 3.1 znázorňuje jednotlivé typické príklady spolu s ich klasifikáciou do vhodnej triedy.

| | výška | vlasy | oči | trieda |
|---|---|---|---|---|
| 1 | nízky | blond | hnedé | − |
| 2 | vysoký | tmavé | hnedé | − |
| 3 | vysoký | blond | modré | + |
| 4 | vysoký | tmavé | modré | − |
| 5 | nízky | tmavé | modré | − |
| 6 | vysoký | červené | modré | + |
| 7 | vysoký | blond | hnedé | − |
| 8 | nízky | blond | modré | + |

Tabuľka 3.1: Ilustračné typické príklady

Cieľom induktívneho učenia je vytvorenie nejakej rozhodovacej štruktúry, pomocou ktorej pre dátovú časť každého príkladu možno nájsť odpovedajúcu rozhodovaciu časť (najčastejšie používaným tvarom tejto rozhodovacej štruktúry sú produkčné pravidlá alebo rozhodovacie stromy).

Najjednoduchší prípad nastáva vtedy, ak sú k dispozícii všetky možné typické príklady. Vtedy je možné vytvoriť rozhodovaciu procedúru, ktorej rozhodovanie bude úplne presné. Častejším a zároveň obtiažnejším prípadom je, ak vo fáze indukovania rozhodovacej štruktúry niektoré prípady nie sú k dispozícii. Rozhodovacia procedúra bude potom síce správne riešiť tie typické príklady, na základe ktorých bola vytvorená (je teda *perfektná*), avšak môže byť neúčinná v prípade tých príkladov, ktoré neboli použité v procese indukcie. Poznamenajme ešte, že existujú aj prístupy vytvárajúce rozhodovacie procedúry, ktoré nemusia správne klasifikovať všetky tie príklady, na základe ktorých boli indukované.

Pre vytvorenie predstavy o úspešnosti rozhodovacej procedúry pre neznáme typické príklady sa zvyčajne používa rozdelenie dostupných typických príkladov do dvoch množín:

1. učiaca (trénovacia) množina

   na základe týchto typických príkladov bude rozhodovacia procedúra vytvorená,

2. testovacia množina

   príklady z tejto množiny sa nezúčastňujú samotnej indukcie ale slúžia pre vyhodnotenie úspešnosti indukovanej rozhodovacej procedúry pri spracovaní neznámych príkladov (teda ich klasifikácii).

Pre rozdelenie príkladov do týchto dvoch množín neexistuje všeobecne platné pravidlo, iba rôzne odporúčania (napr. pre pomer veľkostí učiacej a testovacej množiny 2:1).

V rámci induktívneho učenia je možné použiť metódy, ktoré vznikli mimo rámca znalostných systémov, aj metódy zohľadňujúce špeciálne spôsoby práce znalostných systémov. Prvá trieda metód sa zvyčajne viaže na všeobecne používané spôsoby reprezentácie a využitia znalostí (napr. rozhodovacie stromy). Naproti tomu druhá trieda metód umožňuje generovať aj neurčité znalosti a zohľadňuje nejaký mechanizmus reprezentácie neurčitosti a narábania s touto neurčitosťou.

## 3.1 Tvorba rozhodovacích stromov

Rozhodovací strom je reprezentácia rozhodovacej procedúry pre klasifikáciu typických príkladov do príslušných tried. Každý uzol rozhodovacieho stromu reprezentuje alebo triedu alebo špecifický test, ktorý rozdeľuje priestor príkladov, asociovaných s príslušným uzlom, podľa možného výsledku testu. Každé takéto rozdelenie korešponduje s klasifikačným podproblémom pre daný podpriestor príkladov, ktorý je riešený príslušným podstromom rozhodovacieho stromu.

Formálne pre definíciu rozhodovacieho stromu je potrebné určiť:

- pre listový uzol názov triedy, do ktorej budú klasifikované všetky typické príklady, asociované s daným listovým uzlom (niekedy to je triviálna úloha, inokedy sú s listovým uzlom asociované príklady z viacerých tried),
- pre nelistový uzol atribút, podľa ktorého sa rozhodovanie vetví na jednoduchšie rozhodovacie podstromy pre každú hodnotu zvoleného testovacieho atribútu.

Každý príklad je asociovaný s jedným listovým uzlom. Táto asociácia je definovaná procedúrou, vytvárajúcou cestu od koreňového uzla k uzlu listovému. Hodnota listového uzla potom definuje tú triedu, do ktorej je skúmaný príklad klasifikovaný.

Algoritmy pre vytváranie rozhodovacích stromov sú typicky založené na princípe budovania stromov *zhora nadol*. Pri tomto postupe je pôvodný priestor príkladov delený na jednotlivé podpriestory, charakterizované hodnotami určitého atribútu. Toto delenie prebieha rekurzívne zvyčajne až dovtedy, kým v týchto podpriestoroch nie je splnená podmienka ukončenia delenia priestoru príkladov na podpriestory (napr. sa v nich vyskytujú iba príklady, patriace do jednej triedy – no môže byť definované aj iné kritérium). Prípad, keď už boli vyčerpané všetky atribúty a v niektorom podpriestore sa ešte stále nepodarilo splniť ukončovacie kritérium, signalizuje nedostatočnosť množiny atribútov (resp. protirečivosť v pôvodnej množine príkladov).

Všeobecný postup generovania rozhodovacieho stromu zhora nadol môže byť popísaný podľa nasledujúceho algoritmu:

1. Ak pre každý podpriestor je splnené ukončovacie kritérium, generovanie sa ukončí.

2. Inak

   (a) Zvolí sa taký podpriestor, ktorý obsahuje príklady klasifikované do viacerých tried.

   (b) Pre zvolený podpriestor sa vyberie jeden atribút, ešte nepoužitý pre vytvorenie zvoleného podpriestoru príkladov.

   (c) Zvolený podpriestor príkladov sa rozdelí na ďalšie podpriestory podľa hodnôt zvoleného deliaceho atribútu.

Bod 1 vlastne zodpovedá prípadu, keď generovanie rozhodovacieho stromu je už ukončené. Bod 2 naproti tomu pokračuje v generovaní, keď expanduje jeden listový uzol o jednu úroveň.

Jediný problém je v tom, ako voliť atribúty v kroku 2(b). Pri vhodnej voľbe generovaný rozhodovací strom bude minimálny, zatiaľ čo nevhodná voľba testovacích atribútov môže spôsobiť značné zväčšenie rozmerov generovaného rozhodovacieho stromu.

Jedným z algoritmov vytvárajúcich rozhodovací strom metódou zhora nadol je aj algoritmus ID3 (Inductive Dichotomizer 3), ktorý bol vyvinutý Rossom Quinlainom v 1979 v kontexte reprezentácie znalostí šachistu o hraní šachových koncoviek.

Ukončovacie kritérium pre algoritmus ID3 je, že každý podpriestor obsahuje iba príklady patriace do tej istej triedy. Ak je množina atribútov dostatočná, možno uvedeným postupom zostrojiť rozhodovacie stromy, korektne klasifikujúce príklady z učiacej množiny. Rozhodovací strom generovaný algoritmom je

- *perfektný* — správne klasifikuje všetky príklady z učiacej množiny príkladov,

- *rozmerovo optimálny* — rozhodovacia procedúra je čo najjednoduchšia, s čo najmenším počtom testov hodnôt atribútov.

Algoritmus ID3 je založený na využití teórie informácie pre výber vhodného atribútu pre delenie priestoru príkladov. Pre tento účel využíva pojem entropie. Entropia je použitá v teórii informácií pre meranie množstva informácie. Ak jednotlivé správy $x_1, x_2, \ldots, x_n$ sú možné s pravdepodobnosťami $p(x_1), p(x_2), \ldots, p(x_n)$ pričom pravdepodobnosti vytvárajú úplný súbor pravdepodobností

$$\sum_{j=1}^{n} p(x_j) = 1 \tag{3.1}$$

potom entropiu (neurčitosť) súboru správ $x_1, x_2, \ldots, x_n$ možno vyjadriť ako

$$H = -\sum_{j=1}^{n} p(x_j) log_2(p(x_j)) \tag{3.2}$$

čo je vzťah známy ako Shannonova entropia. Veličinu $H$ možno interpretovať ako množstvo informácie, získanej prijatím jednej zo správ súboru $x_1, \ldots, x_n$. Čím je entropia súboru správ vyššia, tým neurčitejší je obsah budúcej správy a tým väčšie množstvo informácie získame, keď správu prijmeme.

Rozhodovací strom môže byť považovaný za zdroj informácie, ktorý pre nejaký konkrétny príklad generuje správu o klasifikácii daného príkladu do nejakej triedy. Keď uzol stromu obsahuje iba príklady, patriace do tej istej triedy, entropia v tomto uzle je nulová - klasifikačné rozhodnutie pre prípady, prislúchajúce tomuto uzlu, je pevne dané.

Z toho vyplýva, že entropia v listových uzloch je nulová, zatiaľ čo v koreňovom uzle a medziľahlých uzloch má nenulovú hodnotu. Pre vygenerovanie čo najmenšieho rozhodovacieho stromu by teda entropia mala čo najrýchlejšie klesnúť na nulovú hodnotu. Algoritmus ID3 toto nahrádza heuristikou, ktorá sa v každom kroku lokálne snaží aby entropia klesla čo najviac. Pri výbere atribútu pre delenie priestoru príkladov vyberá preto vždy ten atribút, ktorý čo najviac znižuje hodnotu entropie.

Nech uzol $S$ obsahuje $n_1$ príkladov klasifikovaných do triedy $T_1$ a $n_2$ príkladov, patriacich do triedy $T_2$. Potom pravdepodobnosť, že nejaký príklad prislúchajúci uzlu $S$ rozhodovacieho stromu, bude klasifikovaný do triedy $T_1$ resp. $T_2$ je

$$\frac{n_1}{n_1 + n_2} \qquad resp. \qquad \frac{n_2}{n_1 + n_2} \tag{3.3}$$

a teda entropia v danom uzle $S$ (vlastne entropia súboru tých príkladov, ktoré prislúchajú uzlu $S$) bude určená podľa

$$H(S) = -\sum_{j=1}^{2} \frac{n_j}{n_1 + n_2} log_2 \left( \frac{n_j}{n_1 + n_2} \right) \tag{3.4}$$

Nech v uzle $S$ je možné použiť atribút $Atr$ s hodnotami $a_1$ a $a_2$ pre rozdelenie príkladov z uzla $S$ do dvoch uzlov $A_1$ a $A_2$. Ak v uzle $S$ $m_1$ príkladov má hodnotu $a_1$ a $m_2$ príkladov má hodnotu $a_2$ pre atribút $Atr$, potom celková entropia v uzle $S$ s použitím atribútu $Atr$ sa určí ako

$$H(S, Atr) = \sum_{j=1}^{2} \frac{m_j}{m_1 + m_2} H(A_j) \tag{3.5}$$

Takto informácia, získaná použitím atribútu $Atr$ (jeho použitím sme sa priblížili ku konečnej klasifikácii) je označovaná ako informačný zisk

$$I(Atr) = H(S) - H(S, Atr) \tag{3.6}$$

V každej iterácii sa v kroku 2(b) algoritmu pre generovanie rozhodovacieho stromu metódou zhora nadol vyšetrujú všetky ešte nepoužité atribúty a ako testovací sa vyberá ten, ktorý maximalizuje informačný zisk resp. minimalizuje entropiu.

Algoritmus možno jednoducho rozšíriť aj na prípad, keď príklady budú klasifikované do viacerých tried a keď atribúty môžu nadobúdať viac ako dve hodnoty. Ak jednotlivé triedy môžu byť $T_1$, $T_2$, ..., $T_n$, potom vzťah (3.4) bude mať tvar

$$H(S) = -\sum_{j=1}^{n} p(T_j) log_2(p(T_j)) \tag{3.7}$$

kde $p(T_j)$ je pravdepodobnosť toho, že nejaký príklad, patriaci uzlu $S$, bude klasifikovaný do triedy $T_j$. Ak atribút $Atr$ môže nadobúdať hodnoty $a_1$, $a_2$, ..., $a_n$, tak vzťah (3.5) bude mať tvar

$$H(S, Atr) = \sum_{j=1}^{n} p(a_j) H(A_j) \tag{3.8}$$

kde $p(a_j)$ je pravdepodobnosť toho, že nejaký príklad, patriaci uzlu $S$, má hodnotu atribútu $Atr = a_j$.

Pre použitie algoritmu musí platiť podmienka *neprotirečivosti* (nekontradikčnosti) príkladov. Zo spôsobu určovania pravdepodobností vyplýva, že iba dodržanie dvoch dodatočných podmienok *neredundantnosti* príkladov (žiadny príklad sa nesmie v učiacej množine príkladov vyskytovať viackrát) a *vzájomnej nezávislosti* jednotlivých atribútov garantuje nájdenie minimálneho rozhodovacieho stromu.

Pre ilustračný príklad z tab. 3.1 bude pre jednotlivé atribúty:

– výška:

| vysoký | nízky | spolu |
|---|---|---|
| 2+ 3– | 1+ 2– | 3+ 5– |
| H(vysoký)=0.971 | H(nízky)=0.918 | |
| p(vysoký)=0.625 | p(nízky)=0.375 | H(S)=0.954 |
| H(S,výška)=0.951 | | I(výška)=0.003 |

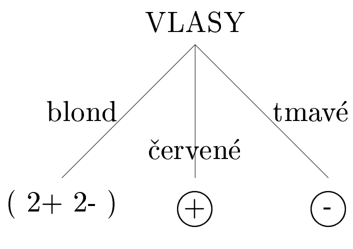
– vlasy:

| blond | červené | tmavé | spolu |
|---|---|---|---|
| 2+ 2– | 1+ 0– | 0+ 3– | 3+ 5– |
| H(blond)=1 | H(červené)=0 | H(tmavé)=0 | |
| p(blond)=0.5 | p(červené)=0.125 | p(tmavé)=0.375 | H(S)=0.954 |
| H(S,vlasy)=0.5 | | | I(vlasy)=0.454 |

– oči:

| modré | hnedé | spolu |
|---|---|---|
| 3+ 2– | 0+ 3– | 3+ 5– |
| H(modré)=0.971 | H(hnedé)=0 | |
| p(modré)=0.625 | p(hnedé)=0.375 | H(S)=0.954 |
| H(S,oči)=0.607 | | I(oči)=0.347 |

Z jednotlivých vypočítaných hodnôt platí, že najväčší informačný zisk sa dosahuje pri výbere atribútu "vlasy". Po výbere tohoto atribútu bude mať rozhodovací strom tvar znázornený na obr. 3.1.




Obr. 3.1: Rozhodovací strom po výbere prvého atribútu

Pre prípady *vlasy* = *tmavé* a *vlasy* = *červené* príslušné podpriestory príkladov obsahujú iba príklady, ktoré patria do jednej triedy (a to do triedy

"-" pre *vlasy* = *tmavé* a do triedy "+" pre *vlasy* = *červené*). Podmnožina príkladov pre *vlasy* = *blond* obsahuje štyri príklady, ktoré sú klasifikované do rôznych tried. Preto pre tento prípad je potrebná voľba ďalšieho atribútu:

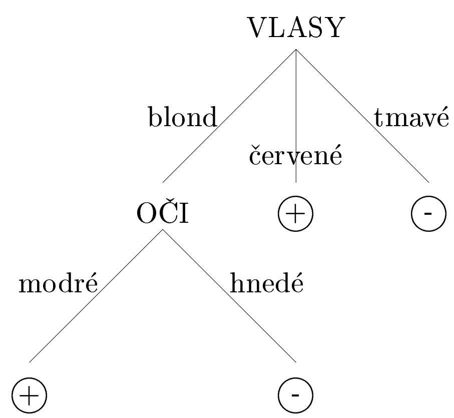
– výška:

| vysoký | nízky | spolu |
|---|---|---|
| 1+ 1– | 1+ 1– | 2+ 2– |
| H(vysoký)=1 | H(nízky)=1 | |
| p(vysoký)=0.5 | p(nízky)=0.5 | H(S)=1 |
| H(S,výška)=1 | | I(výška)=0 |

– oči:

| modré | hnedé | spolu |
|---|---|---|
| 2+ 0– | 0+ 2– | 2+ 2– |
| H(modré)=0 | H(hnedé)=0 | |
| p(modré)=0.5 | p(hnedé)=0.5 | H(S)=1 |
| H(S,oči)=0 | | I(oči)=1 |

Očividne je výhodnejší atribút "oči". Po jeho voľbe vznikne rozhodovací strom v tvare podľa obr. 3.2.




Obr. 3.2: Výsledný rozhodovací strom

Táto podoba rozhodovacieho stromu v každom listovom uzle už obsahuje iba príklady jednej triedy a teda každý listový uzol môže niesť údaj o klasifikácii do určitej triedy. Výsledný rozhodovací strom je značne menší ako strom ktorý by vznikol ak by sa boli použili atribúty v poradí: 1. výška, 2. vlasy a 3. oči.

Rozhodovací strom môže byť jednoducho konvertovaný do súboru produkčných pravidiel. Jedno pravidlo môže reprezentovať jednu možnú cestu od koreňového uzla k uzlu listovému. Pre nejakú triedu je teda možné vytvoriť toľko pravidiel, koľko jej zodpovedá listových uzlov.

Uvedeným postupom je možné rozhodovací strom z obr. 3.2 pretransformovať na 4 produkčné pravidlá, pričom pre triedu "+" dostaneme tieto dve pravidlá:

AK *vlasy* = *blond* A *oči* = *modré* POTOM trieda "+"
AK *vlasy* = *červené* POTOM trieda "+"

## 3.2 Generovanie produkčných pravidiel

Veľa metód patriacich do tejto skupiny využíva fakt, že množina produkčných pravidiel, realizujúcich klasifikáciu do nejakej klasifikačnej triedy $T$ vlastne reprezentuje logický výraz, popisujúci túto triedu (definujúci nutné a postačujúce podmienky pre príslušnosť príkladov do tejto triedy). Tento výraz má tvar disjunkcie

$$\text{podmienka\_pravidla\_1} \vee \text{podmienka\_pravidla\_2} \vee \ldots$$

Každá podmienka nejakým spôsobom obmedzuje, ktoré hodnoty môžu atribúty nadobúdať. Na rozdiel od predchádzajúceho, kde sme možné hodnoty atribútov špecifikovali iba pomocou *selektorov* tvaru *atribút=hodnota*, budeme využívať zložitejšie selektory. Selektor teraz bude tvaru $Atr_i \# R_i$, kde $R_i$ zastupuje disjunkciu hodnôt atribútu $Atr_i$ (jedna hodnota, niekoľko alebo všetky hodnoty, ktoré daný atribút môže nadobúdať) a znak # označuje jeden z operátorov "=" alebo "≠". Ako ilustrácia môžu slúžiť dve ukážky selektorov

$$\begin{array}{lcl} \text{vlasy} & \neq & \text{červené} \\ \text{vlasy} & = & \text{blond ALEBO tmavé} \end{array}$$

Metódy určené pre vytváranie popisu klasifikačných tried v tvare disjunkcie typicky využívajú *princíp pokrývania.* Pri uvedenom zápise selektorov príklad je pokrytý nejakým výrazom práve vtedy, ak pre každý selektor výrazu hodnota atribútu príslušného selektora príkladu patrí do množiny hodnôt atribútu daného selektora výrazu. Inak povedané, každý selektor výrazu v sebe zahŕňa príslušný selektor príkladu. Tak napr. príklad

$$\text{výška} = \text{nízky A vlasy} = \text{blond A oči} = \text{modré}$$

je pokrytý výrazom

$$\text{vlasy} \neq \text{červené A oči} = \text{modré}$$

ale nie je pokrytý výrazom

$$\text{výška} = \text{vysoký A oči} \neq \text{hnedé}$$

Príkladom algoritmu využívajúceho princíp pokrývania je algoritmus AQ11 navrhnutý Michalskim v roku 1969. Umožňuje priamo generovať produkčné pravidlá bez medzistupňa, akým je napr. rozhodovací strom. Pri indukcii používa *generalizáciu* jednotlivých príkladov. Oproti algoritmu ID3 má výhody:

- nepoužíva pravdepodobnosti (odpadá požiadavka vzájomnej nezávislosti jednotlivých typických príkladov,
- nevadí mu redundantnosť príkladov,
- už v princípe inkrementálna povaha algoritmu.

Algoritmus využíva pojem *obálky.* Obálkou $G(e \mid E)$ budeme rozumieť logický výraz, ktorý pokrýva príklad $e$ a zároveň nepokrýva ani jeden príklad z množiny $E$. Podobne obálka $G(e_1 \mid e_2)$ pokrýva príklad $e_1$ a nepokrýva $e_2$, a obálka $G(E_1 \mid E_2)$ pokrýva všetky príklady z množiny $E_1$ a zároveň ani jeden príklad z množiny $E_2$.

Podobne ako ID3 aj AQ11 sa snaží o perfektnú klasifikáciu všetkých príkladov z učiacej množiny. Algoritmus vlastne hľadá popis jednotlivých klasifikačných tried, pričom popis nejakej triedy pokrýva všetky príklady danej triedy a zároveň nepokrýva ani jeden príklad z inej triedy. Spôsob činnosti algoritmu je vlastne rozdelenie problému na podproblémy — nájdenie popisu určitého pozitívneho príkladu (označovaného ako jadro tohto popisu) voči kontrapríkladom, ktoré vystupujú ako ohraničenia výsledného popisu (žiadny z nich nesmie byť pokrytý nájdeným popisom). Tento popis okrem jadra môže súčasne zahŕňať aj iné príklady popisovanej triedy. Toto sa opakuje pre všetky príklady, patriace do danej triedy (resp. dostatočný počet jadier klasifikovaných do popisovanej triedy), až pokiaľ všetky príklady sú pokryté nájdeným popisom (a nie je pokrytý ani jeden negatívny príklad – príklad nepatriaci do danej triedy).

Nech $E_1$ a $E_2$ sú dve navzájom disjunktné množiny typických príkladov, na základe ktorých je potrebné indukciou vytvoriť príslušné produkčné pravidlá, korektne klasifikujúce tieto príklady do správnych tried. Nech všetky príklady z množiny $E_1$ patria do triedy $T_1$ a všetky príklady z množiny $E_2$ patria do triedy $T_2$ .

Algoritmus pre vytvorenie pravidiel pre klasifikáciu do triedy $T_1$ potom pozostáva z niekoľkých krokov:

1. Z množiny $E_1$ sa náhodne vyberie jeden príklad $e$. Vygenerujú sa postupne obálky $G(e \mid e_i)$, $i = 1, 2, \ldots$, kde $e_i$ je kontrapríklad z množiny $E_2$ . Takto sa získajú obálky príkladu $e$ voči všetkým kontrapríkladom z množiny $E_2$ .

   Elementárnu obálku $G(e \mid e_i)$ vygenerujeme nasledovne. Najprv je potrebné nájsť všetky atribúty, ktoré majú rôzne hodnoty v príkladoch $e$ a $e_i$. K týmto atribútom sa pre obálku generuje selektor. Jednotlivé selektory obálky budú mať tvar $Atr_i \neq v_i$, kde $v_i$ je hodnota, ktorú nadobúda atribút $Atr_i$ v kontrapríklade $e_i$. Celkovú obálku $G(e \mid e_i)$ dostaneme ako logický súčet všetkých selektorov

$$G(e \mid e_i) = \bigvee_{i=1}^{n} (Atr_i \neq v_i) \tag{3.9}$$

   kde $n$ je počet tých atribútov, ktoré majú rôzne hodnoty v príkladoch $e$ a $e_i$.

   Počet selektorov je:

   - maximálne počet všetkých atribútov (ak sa príklady líšia hodnotou každého atribútu),
   - minimálne 1 (príklady sa musia líšiť aspoň hodnotou jedného atribútu, aby boli rôzne).

2. Vytvorí sa celková obálka $G(e \mid E_2)$ ako logický súčin všetkých elementárnych obálok $G(e \mid e_i)$

$$G(e \mid E_2) = \bigwedge_{e_i \in E_2} G(e \mid e_i) \tag{3.10}$$

   Pri logickom súčine použijeme absorpčný zákon, čím dostaneme neredundantný logický výraz.

3. Z množiny $E_1$ sa vylúčia všetky príklady, ktoré sú pokryté vygenerovanou obálkou. Ak $E_1$ nie je prázdne, opäť sa pokračuje krokom 1. Ak je množina $E_1$ už prázdna, pokračuje sa krokom 4.

4. Disjunkcia všetkých obálok z kroku 2 tvorí celkovú obálku $G(E_1 \mid E_2)$. Táto celková obálka zároveň predstavuje klasifikačné pravidlá pre priradenie typických príkladov do triedy $T_1$.

$$G(E_1 \mid E_2) = \bigvee_{i=1}^{m} G(e_i \mid E_2) \tag{3.11}$$

kde $m$ je počet rôznych obálok, vygenerovaných v kroku 2.

Podobným postupom sa vygenerujú obálka druhej triedy. Algoritmus je možné použiť aj pre generovanie produkčných pravidiel pre klasifikáciu typických príkladov do viacerých tried. Jedinou podmienkou použitia algoritmu je rozdelenie všetkých typických príkladov do dvoch množín $E_1$ (množina pozitívnych príkladov) a $E_2$ (množina negatívnych príkladov - kontrapríkladov).

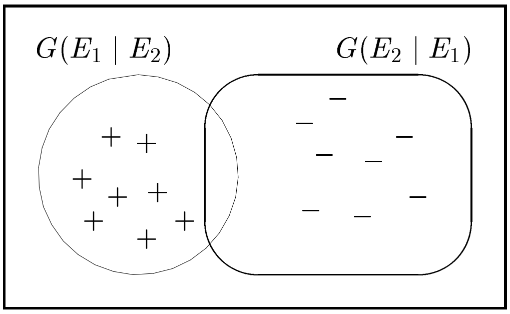
V prípade možnej klasifikácie jednotlivých prípadov do viacerých klasifikačných tried $T_1, T_2, \ldots, T_n$ potom pri generovaní pravidla pre klasifikáciu do triedy $T_i$ budú množinu $E_1$ tvoriť všetky typické príklady klasifikované do triedy $T_i$ a množinu $E_2$ všetky ostatné typické príklady.

Môže sa stať, že tieto obálky sa budú niekde prekrývať (iba v mieste, pre ktoré nebol k dispozícii žiadny typický príklad). Ak v budúcnosti nastane prípad, prislúchajúci miestu prekrytia, bude pozitívne klasifikovaný do viacerých tried (obr. 3.3).

Pre zamedzenie tohto javu sa používa postup, že pri generovaní obálky triedy $T_2$ sa negeneruje voči príkladom z $E_1$ ale voči obálke týchto príkladov. Tým sa dosiahne, že obálky budú disjunktné. Nevýhoda tohoto je, že obálky neskoršie generovaných tried budú obopínať svoje príklady stále tesnejšie a tesnejšie.

Pri generovaní obálky nie všetky príklady sú rovnako významné. Najvýznamnejšie sú "okrajové" prípady, "stredné" nie sú dôležité. Z dôvodu zvýšenia rýchlosti je možné vstupnú množinu príkladov predspracovať s cieľom výberu reprezentatívnych "okrajových" príkladov.

Pre ilustračný príklad z tab. 3.1 bude pri generovaní produkčných pravidiel pre klasifikáciu do triedy "+" množina $E_1$ tvorená typickými príkladmi $e_3$, $e_6$ a $e_8$. Všetky ostatné príklady budú vytvárať množinu $E_2$. Postup generovania obálok pre triedu "+":




Obr. 3.3: Prekrývajúce sa obálky

- *krok 1:* Vyberme príklad $e_3$ ako reprezentanta triedy "+". Potom jednotlivé obálky budú mať tvar:

$$\begin{aligned}
G(e_3 \mid e_1) &= výška \neq nízky \vee \phantom{vlasy \neq tmavé \vee{}} oči \neq hnedé \\
G(e_3 \mid e_2) &= \phantom{výška \neq nízky \vee{}} vlasy \neq tmavé \vee oči \neq hnedé \\
G(e_3 \mid e_4) &= \phantom{výška \neq nízky \vee{}} vlasy \neq tmavé \\
G(e_3 \mid e_5) &= výška \neq nízky \vee vlasy \neq tmavé \\
G(e_3 \mid e_7) &= \phantom{výška \neq nízky \vee vlasy \neq tmavé \vee{}} oči \neq hnedé
\end{aligned}$$

- *krok 2:* Celková obálka príkladu $e_3$ voči množine $E_2$ bude (s využitím absorpčného zákona)

  $G(e_3 \mid E_2) = vlasy \neq tmavé \wedge oči \neq hnedé$

- *krok 3:* Vygenerovaná obálka $G(e_3 \mid E_2)$ pokrýva zároveň aj príklady $e_6$ a $e_8$, preto tieto boli vypustené z množiny $E_1$, ktorá ostala prázdnou.

- *krok 4:* Výsledná obálka ostáva

$$\begin{aligned}
G(E_1 \mid E_2) &= vlasy \neq tmavé \wedge oči \neq hnedé = \\
&= (vlasy = blond \vee červené) \wedge oči = modré
\end{aligned}$$

Výsledkom generovania sú teda dve produkčné pravidlá

AK *vlasy = blond* A *oči = modré* POTOM trieda "+"
AK *vlasy = červené* A *oči = modré* POTOM trieda "+"

Pri generovaní produkčných pravidiel pre klasifikáciu do triedy "-" množina $E_1$ bude tvorená typickými príkladmi $e_1$, $e_2$, $e_4$, $e_5$ a $e_7$. Všetky ostatné príklady budú vytvárať množinu $E_2$. Postup generovania obálok pre triedu "-":

- *krok 1:* Vyberme príklad $e_1$ ako reprezentanta triedy "-". Potom jednotlivé obálky budú mať tvar:

  $$\begin{aligned}
  G(e_1 \mid e_3) &= výška \neq vysoký \lor & & oči \neq modré \\
  G(e_1 \mid e_6) &= výška \neq vysoký \lor vlasy \neq červené \lor & & oči \neq modré \\
  G(e_1 \mid e_8) &= & & oči \neq modré
  \end{aligned}$$

- *krok 2:* Celková obálka príkladu $e_1$ voči množine $E_2$ bude (s využitím absorpčného zákona)

  $G(e_1 \mid E_2) = oči \neq modré$

- *krok 3:* Vygenerovaná obálka $G(e_1 \mid E_2)$ pokrýva zároveň aj príklady $e_2$ a $e_7$, preto tieto boli vypustené z množiny $E_1$, ktorá ešte obsahuje dva príklady.

- *krok 1:* Vyberme príklad $e_4$ ako reprezentanta triedy "-". Potom jednotlivé obálky budú mať tvar:

  $$\begin{aligned}
  G(e_4 \mid e_3) &= & & vlasy \neq blond \\
  G(e_4 \mid e_6) &= & & vlasy \neq červené \\
  G(e_4 \mid e_8) &= výška \neq nízky \lor & & vlasy \neq blond
  \end{aligned}$$

- *krok 2:* Celková obálka príkladu $e_4$ voči množine $E_2$ bude (s využitím absorpčného zákona)

  $G(e_4 \mid E_2) = vlasy \neq blond \land vlasy \neq červené$

- *krok 3:* Vygenerovaná obálka $G(e_4 \mid E_2)$ pokrýva zároveň aj príklad $e_5$, preto tento bol vypustený z množiny $E_1$, ktorá ostala prázdna.

- *krok 4:* Výsledná obálka teraz bude

  $$\begin{aligned}
  G(E_1 \mid E_2) &= oči \neq modré \lor (\ vlasy \neq blond \land vlasy \neq červené\ ) = \\
  &= oči = hnedé \lor vlasy = tmavé
  \end{aligned}$$

Výsledkom generovania sú opäť dve produkčné pravidlá

AK $oči = hnedé$ POTOM trieda "-"
AK $vlasy = tmavé$ POTOM trieda "-"

## 3.3 Kombinačná analýza dát

Kombinačná analýza dát sa zaoberá výskytom kombinácií hodnôt v dátach. Zisťuje početnosti (frekvencie) a zhody (koincidencie) výskytov rôznych kombinácií hodnôt v dátach.

Podstatou použitia kombinačnej analýzy pri vytváraní bázy znalostí je vyhľadávanie vzťahov smerujúcich k zadanej cieľovej kombinácii. Úlohu dát hrajú typické príklady nachádzajúce sa v učiacej množine, cieľovou kombináciou sa chápe nejaká zvolená klasifikačná trieda a vyhľadávané a skúmané kombinácie sú kombinácie hodnôt jednotlivých atribútov.

Každá kombinácia je určená voľbou hodnôt rôznych atribútov. Množinu všetkých hodnôt ktoré môže nadobúdať atribút $Atr_i$ označíme $V(Atr_i)$. Potom kombinácia dĺžky $k$

$$C = (Atr_1 = a_1) \wedge (Atr_2 = a_2) \wedge \ldots \wedge (Atr_k = a_k) \tag{3.12}$$

je určená voľbou hodnôt $a_1 \in V(Atr_1), \ldots, a_k \in V(Atr_k)$ pre rôzne atribúty $Atr_1, \ldots, Atr_k$. Výskytom tejto kombinácie v dátach je každý príklad, v ktorom sa v príslušných stĺpcoch nachádzajú zvolené hodnoty.

Počet možných kombinácií hodnôt všetkých možných dĺžok závisí na počte $n$ všetkých atribútov a na počtoch $N_j = \mid Atr_j \mid$ rôznych hodnôt, ktoré sa v dátach vyskytujú. Celkove je možné zostaviť

$$(N_1 + 1)(N_2 + 1) \ldots (N_n + 1) \tag{3.13}$$

rôznych kombinácií (nie všetky sa však musia v dátach vyskytovať). Počet tých príkladov, v ktorých sa vyskytuje nejaká kombinácia $C$ budeme označovať $\mid C \mid_D$ (index D označuje, že početnosť sa určuje na základe dát). Zaujímavé sú najmä tie kombinácie, ktoré sa vyskytujú u dostatočného počtu príkladov. Dolná medza pre tento počet sa označuje ako požadovaná *fundovanosť*. Kombinácie s početnosťou väčšou alebo rovnou požadovanej fundovanosti sú fundované kombinácie.

Z hľadiska tvorby bázy znalostí sú zaujímavé koincidencie tvaru $C \to T_i$ znamenajúce že výskyt kombinácie $C$ v nejakom príklade je vždy doprevádzaný klasifikáciou tohto príkladu do i-tej klasifikačnej triedy (teda že kombinácia $C$ implikuje triedu $T_i$). Zaujímavý však môže by aj prípad keď niektorý vzťah platí s výnimkami. Vtedy sa získava "empirická implikácia"

$$C \to T_i \quad (p) \tag{3.14}$$

kde $p$ je platnosť implikácie v dátach. Táto platnosť sa zisťuje ako relatívna početnosť tých výskytov kombinácie $C$, ktoré sú sprevádzané klasifikáciou do triedy $T_i$, voči všetkým výskytom kombinácie $C$.

$$p = \frac{| C \wedge T_i |_D}{| C |_D} \tag{3.15}$$

Uvažujú sa teda aj vzťahy, ktorých platnosť nie je absolútna. Dolná medza pre platnosť vzťahov sa zadáva ako požadovaná *platnosť*. Vzťahy s platnosťou väčšou alebo rovnou požadovanej sa nazývajú platné vzťahy (implikácie).

To, ktoré platné empirické implikácie existujúce v dátach budú zaradené do výslednej bázy znalostí, závisí nielen na samotných dátach, ale aj na

- zvolenom inferenčnom mechanizme, ktorý bude používaný s výslednou bázou znalostí (najmä na operácii *op* skladania jednotlivých neurčitých príspevkov do výslednej neurčitosti),
- požiadavke presnosti, s ktorou má byť výsledná báza znalostí v súlade s dátami,
- požadovaných hodnotách presnosti a fundovanosti.

Pre rozhodnutie, ktoré z nájdených platných vzťahov zaradiť do bázy znalostí a ktoré nie, možno použiť analógiu s vytváraním axiomatickej teórie nejakého odboru. Medzi axiómy sú zaraďované čo najjednoduchšie tvrdenia a tieto sú vyberané tak, aby všetky ostatné zákony odboru boli z nich odvoditeľné. Pritom je odstraňovaná redundancia – medzi axiómy nie sú zaraďované zákony odvoditeľné z iných axióm.

Podobne tejto axiomatickej teórii do bázy znalostí budú zaradené iba tie vzťahy, ktorých platnosť v dátach sa "významne" líši od váhy odvodenej inferenčným mechanizmom z váh vzťahov do bázy znalostí už zaradených. Vzťah

$$(Atr_1 = a_1) \wedge \ldots \wedge (Atr_k = a_k) \rightarrow T_i \tag{3.16}$$

budeme považovať v daných dátach za odvoditeľný z nejakej množiny $B$ vzťahov vedúcich k $T_i$, ak platnosť tohto vzťahu v dátach sa významne nelíši od hodnoty odvodenej zložením príspevkov tých pravidiel z $B$, ktorých podmienka je podkombináciou kombinácie $(Atr_1 = a_1) \wedge \ldots \wedge (Atr_k = a_k)$.

Ak nejaký vzťah je zaraďovaný do bázy znalostí, tak jeho váha, s ktorou je zaraďovaný, je určená tak, aby korigovala rozdiel medzi zloženou váhou, určenou operáciou *op* z pravidiel už zaradených v báze znalostí, a platnosťou zaraďovaného vzťahu v dátach.

Pre testovanie významnej odlišnosti je potrebné definovať nejakú funkciu DIF, stanovujúcu či dve hodnoty možno považovať za rovnaké alebo nie. Pre tento účel je možné využiť napr. mechanizmus štatistického testovania hypotéz.

Algoritmus vytvárania bázy znalostí pozostáva z dvoch hlavných fáz:

1. Inicializačná fáza

   Hodnota $d$ bude označovať momentálne skúmanú dĺžku ľavej strany implikácií. Pri inicializácii sa nastaví na hodnotu $d = 0$.

   Určí sa platnosť prázdneho vzťahu $\emptyset \rightarrow T_i$ (jeho podmienka $\emptyset$ je splnená vždy) ako podiel

   $$\frac{\mid T_i \mid_D}{m} \tag{3.17}$$

   kde $m$ označuje celkový počet všetkých príkladov. Ak sa táto hodnota významne líši od hodnoty 0.5, tak prázdny vzťah sa zaradí do bázy znalostí s touto hodnotou. V opačnom prípade sa zaradí s váhou 0.5.

2. Vyhľadávanie vzťahov

   (a) Zväčší sa dĺžka ľavej strany skúmaných implikácií

   $$d = d + 1 \tag{3.18}$$

   Ak hodnota $d$ sa stala väčšou ako maximálne skúmaná dĺžka kombinácií na ľavej strane implikácií, vyhľadávanie pravidiel končí.

   (b) Postupne sa generujú všetky fundované kombinácie dĺžky $d$. Pre každú takúto kombináciu sa vykonáva:

   i. Zisťuje sa platnosť $w_D$ vzťahu

   $$(Atr_1 = a_1) \wedge \ldots \wedge (Atr_d = a_d) \rightarrow T_i \tag{3.19}$$

   ii. Nájdu sa váhy $w(\emptyset)$, $w(Atr_1 = a_1)$, $\ldots$ všetkých vzťahov v báze znalostí, ktorých ľavá strana je podkombináciou momentálne skúmanej kombinácie.

iii. Určí sa zložená platnosť $w_z$ z týchto nájdených váh pomocou operácie $op$ pre skladanie neurčitých príspevkov do výslednej neurčitosti.

iv. Realizuje sa test $DIF(w_D, w_z)$. Ak tieto dve hodnoty sú prehlásené za odlišné, skúmaný vzťah sa zaradí do bázy s váhou $w$, pre ktorú platí

$$w_z \quad op \quad w \quad = \quad w_D \tag{3.20}$$

Pri jednom použití tohto algoritmu sa získajú iba vzťahy vedúce k jednej klasifikačnej triede. Pri potrebe klasifikácie do viacerých tried je potrebné algoritmus využiť opakovane pre každú možnú klasifikačnú triedu.

Báza znalostí získaná popísaným spôsobom je principiálne použiteľná vo všetkých znalostných systémoch, ktoré pracujú s numerickým prístupom ku skladaniu príspevkov pravidiel. Výsledná báza má špecifický tvar bez prítomnosti medziľahlých výrokov.

Pri použití príkladov podľa tab. 3.1 pre klasifikáciu do triedy "+" platnosť prázdneho vzťahu ($\emptyset \rightarrow +$) je 0.375. Ak funkcia DIF bude hodnoty $w_D$ a $w_z$ považovať za rovnaké vtedy, ak hodnota $w_D$ bude v intervale $< 0.9w_z, 1.1w_z >$, tak prázdny vzťah bude zaradený do bázy znalostí s váhou 0.375.

Uvažujme nízke hodnoty požadovanej fundovanosti a platnosti (napr. 0.01), maximálnu dĺžku skúmaných kombinácií rovnú počtu použitých atribútov a spôsob skladania neurčitosti podľa vzťahu

$$x \quad op \quad y = \frac{x+y}{1+xy} \tag{3.21}$$

ktorý predpokladá reprezentáciu neurčitosti z intervalu $< -1, 1 >$, kde hodnota -1 značí absolútnu neplatnosť a hodnota 1 zase absolútnu platnosť.

Pre kombinácie rôznych dĺžok platí (uvažujú sa iba fundované kombinácie):

- $d = 1$

| kombinácia | $w_D$ | $w_Z$ | $w$ |
|---|---|---|---|
| $výška = nízky$ | 0.333 | 0.375 | -0.048 |
| $výška = vysoký$ | 0.4 | 0.375 | – |
| $vlasy = blond$ | 0.5 | 0.375 | 0.154 |
| $vlasy = červené$ | 1.0 | 0.375 | 1.0 |
| $oči = modré$ | 0.6 | 0.375 | 0.29 |

Kombinácie $vlasy = tmavé$ a $oči = hnedé$ neboli uvažované pretože vzťahy založené na týchto kombináciách nedosahujú požadovanú platnosť.

Platnosť vzťahu "$výška = vysoký \rightarrow +$" v dátach sa nelíši významne od váhy prázdneho vzťahu už zaradeného v báze znalostí (ktorý je jediným takým vzťahom v báze, že jeho podmienka je podkombináciou kombinácie $výška = vysoký$) a preto tento vzťah nebol zaradený do bázy znalostí.

- $d = 2$

| kombinácia | $w_D$ | $w_Z$ | $w$ |
|---|---|---|---|
| $výška = nízky \wedge vlasy = blond$ | 0.5 | 0.463 | – |
| $výška = nízky \wedge oči = modré$ | 0.5 | 0.569 | -0.096 |
| $výška = vysoký \wedge vlasy = blond$ | 0.5 | 0.5 | – |
| $výška = vysoký \wedge vlasy = červené$ | 1.0 | 1.0 | – |
| $výška = vysoký \wedge oči = modré$ | 0.666 | 0.6 | 0.111 |
| $vlasy = blond \wedge oči = modré$ | 1.0 | 0.69 | 1.0 |
| $vlasy = červené \wedge oči = modré$ | 1.0 | 1.0 | – |

Kombinácie $výška = nízky \wedge vlasy = červené$ a $vlasy = červené \wedge oči = hnedé$ nie sú fundované. Podobne ako pre $d = 1$ sa neuvažovali tie kombinácie, ktoré vedú na vzťahy s nedostatočnou platnosťou.

- $d = 3$

| kombinácia | $w_D$ | $w_Z$ | $w$ |
|---|---|---|---|
| $výška = nízky \wedge vlasy = blond \wedge oči = modré$ | 1.0 | 1.0 | – |
| $výška = vysoký \wedge vlasy = blond \wedge oči = modré$ | 1.0 | 1.0 | – |
| $výška = vysoký \wedge vlasy = červené \wedge oči = modré$ | 1.0 | 1.0 | – |

Pre tento prípad existujú iba tri platné vzťahy. Pretože ich platnosť v dátach sa nelíši od toho, čo je možné odvodiť z už existujúcej bázy znalostí, ani jeden z nich nebol zaradený do bázy.

Výsledný tvar bázy znalostí (jej časť pre klasifikáciu do triedy "+") teda bude obsahovať množinu ôsmich produkčných pravidiel:

AK $\emptyset$ POTOM "+" (0.375)
AK $výška = nízky$ POTOM "+" (-0.048)
AK $vlasy = blond$ POTOM "+" (0.154)
AK $vlasy = červené$ POTOM "+" (1)

AK $o\check{c}i = modr\acute{e}$ POTOM "+" (0.29)
AK $v\acute{y}\check{s}ka = n\acute{i}zky$ A $o\check{c}i = modr\acute{e}$ POTOM "+" (-0.096)
AK $v\acute{y}\check{s}ka = vysok\acute{y}$ A $o\check{c}i = modr\acute{e}$ POTOM "+" (0.111)
AK $vlasy = blond$ A $o\check{c}i = modr\acute{e}$ POTOM "+" (1)

# Kapitola 4

# Priamy prenos znalostí

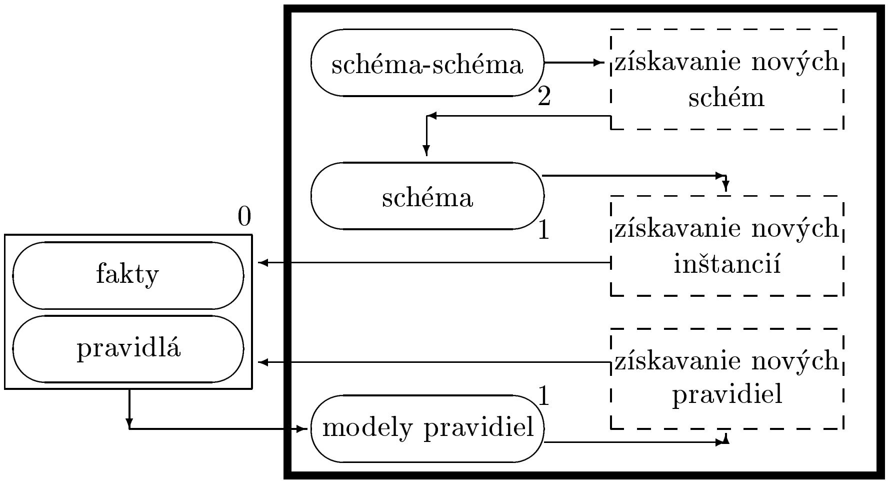
Expert môže sám vytvárať bázu znalostí v príslušnom editore v dohodnutom formáte. Avšak inteligentný modul priameho získavania znalostí od experta by mal:

- "rozumieť" vkladaným znalostiam nielen v syntaktickom zmysle, ale mal by byť schopný odhaliť aj sémantické chyby,
- navádzať experta pri vkladaní nových znalostí, vyhodnocovať, ktorá časť spracovávanej oblasti je slabo pokrytá znalosťami,
- automaticky udržiavať konzistenciu bázy znalostí.

Je zrejmé, že takýto spôsob získavania znalostí vďaka svojej zložitosti nepatrí medzi bežné nástroje používané pri tvorbe znalostných systémov.

## 4.1 Získavanie znalostí v kontexte

Pokusom o takýto typ systému bol systém TEIRESIAS, ktorý slúžil ako modul získavania znalostí pre systém EMYCIN. Jeho základnou charakteristikou je používanie *metaznalostí*. Systém vytvára nielen samotnú bázu znalostí, ale aj znalosti o reprezentácii problémovo špecifických znalostí, uložených v báze znalostí. Na základe svojich metaznalostí o znalostiach v báze znalostí si dokáže systém vytvoriť určité očakávania, ako má nová znalosť vyzerať. To mu poskytuje kontext, voči ktorému môže byť nová znalosť porovnávaná. Obr. 4.1 blokovo znázorňuje proces získavanie nových znalostí.




Obr. 4.1: Získavanie znalostí v systéme TEIRESIAS

Rozlišujú sa dva odlišné typy získavania znalostí:

- získavanie *nových konceptov* (nové atribúty resp. nové hodnoty už známych atribútov), ich popis, špecifiká, vzťahy k už existujúcim konceptom (v hornej časti obrázku),
- získavanie *vzťahov medzi už existujúcimi konceptami* (v dolnej časti obrázku).

Tri bloky, ohraničené prerušovanou čiarou, predstavujú jednotlivé procesy získavania nových znalostí. Oválne bloky zase reprezentujú rozličné typy znalostí, ktoré sú systému k dispozícii. Čísla pri týchto blokoch označujú úroveň znalostí (0 - znalosti o problémovej oblasti, 1 a 2 - metaznalosti).

## Získavanie vzťahov medzi už existujúcimi konceptami

Jedná sa vlastne o získavanie nových znalostí, reprezentovaných pomocou produkčných pravidiel. Toto je založené na skutočnosti, že pravidlá, týkajúce sa toho istého konceptu, zvyčajne vykazujú nejakú pravidelnosť.

Systém si vytvára vnútorné *modely* pravidiel. Tieto sú založené na uniformite jednotlivých podmnožín pravidiel a vyzerajú ako popisy týchto podmnožín. Reprezentujú koncept "typického pravidla" v danej podmnožine.

Jednotlivé modely pravidiel sú vytvárané empirickou generalizáciou pravidiel, patriacich do jednotlivých podmnožín. Proces vytvárania modelov pravidiel je označovaný ako *formovanie konceptov*.

Modely pravidiel (ich štruktúra) pozostávajú zo štyroch častí:

1. príklady

   podmnožina tých pravidiel, ktoré model reprezentuje (spolu s činiteľmi istoty),

2. popis

   charakteristika "typického" člena podmnožiny pravidiel.
   Obsahuje:

   - charakterizovanie predpokladov,
   - charakterizovanie záverov

   (pod charakteristikami sa chápe, ktoré atribúty sa typicky objavujú v predpokladoch resp. záveroch, a typické korelácie medzi atribútmi),

3. všeobecnejší model

   smerník na všeobecnejší model,

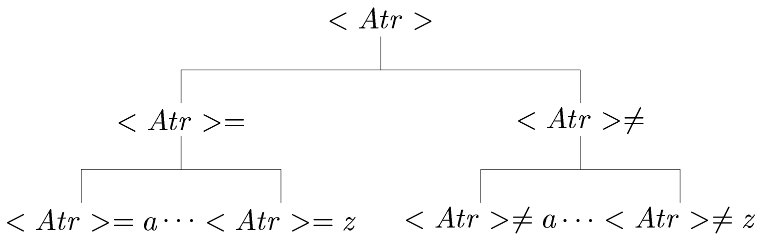
4. špecifickejší model

   smerník na špecifickejší model.

Množina vytváraných modelov je organizovaná do stromových štruktúr majúcich tvar podľa Obr. 4.2.

Jednotlivé modely sú organizované podľa obsahu akčnej časti pravidiel. V koreni každej takejto štruktúry je umiestnený model vytvorený zo všetkých pravidiel, týkajúcich sa určitého atribútu (teda pravidlá majú daný atribút vo svojej záverovej časti, robia záver o hodnote tohoto atribútu). Na nižšej úrovni sú modely pravidiel, majúcich súvis osobitne s potvrdzovaním alebo popieraním hodnoty daného atribútu a na najnižšej úrovni sú modely, týkajúce sa špecifických hodnôt (opäť osobitne pre ich potvrdzovanie resp. popieranie).

Aby mohol byť model vytvorený (automaticky systémom, expert nemá ani potuchy o tom, že nejaké modely sú vytvárané, nieto že by bol týmto vytváraním modelov zaťažovaný), musí existovať aspoň jedno pravidlo. Systém




Obr. 4.2: Stromová štruktúra modelov pravidiel

si takto sám vytvára znalosti o znalostiach, ktoré získal od experta. Vytvára ich na základe jednoduchého štatistického spracovania pravidiel, patriacich k určitému modelu.

*Typický atribút* je v modeloch pravidiel reprezentovaný konštrukciou, ktorá má tvar:

$$( < \text{atribút} > < \text{predikát} >^{+} < \text{cf-suma} > )$$

kde znak "+" indikuje možný viacnásobný výskyt. Takáto konštrukcia je vytváraná pre každý atribút, ktorý sa vyskytuje aspoň v 30% tých pravidiel, pre ktoré je príslušný model vytváraný. Predikát je zaradený iba vtedy, ak sa vyskytuje aspoň v 75% tých pravidiel príslušnej podmnožiny pravidiel, v ktorých sa vyskytuje daný atribút. Samozrejme, je možné použiť aj iné prahové hodnoty. Zmenšenie prahov má vo všeobecnosti za následok zvyšovanie množstva údajov v modeloch pravidiel (tých ktoré popisujú typického člena reprezentovanej množiny pravidiel). Zvyšovanie prahov má za následok opačnú tendenciu.

Hodnota cf-suma je súčet činiteľov istoty tých pravidiel zo skúmanej podmnožiny pravidiel, v ktorých sa daný atribút vyskytuje. Indikuje dôležitosť atribútu pre dokázanie resp. popretie toho parametra, ktorému prináleží vytváraný model. Malá hodnota môže znamenať, že atribút sa vyskytuje iba v malom počte pravidiel a/alebo tie pravidlá, v ktorých sa vyskytuje, sú málo významné. Hodnota cf-suma môže byť takto použitá napríklad pre definovanie nejakej metriky v priestore modelov pravidiel.

Teda konštrukcia ( MORPH SAME NOTSAME 3.83 ) značí, že nejaký atribút MORPH (morfológia) sa nachádza aspoň v 30% pravidiel niektorej z podmnožín pravidiel (súčet činiteľov istoty tých pravidiel, v ktorých sa nachádza, je 3.83) a aspoň v 75% svojho výskytu sa vyskytuje spolu s SAME alebo NOTSAME.

Podobne *typická korelácia* medzi atribútmi je vyjadrená konštrukciou, ktorá má tvar:

$$( \ ( < \text{atribút} > < \text{predikát} >^{+} )^{+} < \text{cf-suma} > )$$

kde druhý atribút (resp. ďalšie atribúty, ak sú uvedené viaceré) sa objavuje spolu so svojou predikátovou funkciou aspoň v 80% prípadoch, keď sa vyskytne prvý atribút so svojou predikátovou funkciou. Príkladom korelácie atribútov je konštrukcia ( (MORPH SAME) (GRAM SAME) 3.83 ) vyjadrujúca koreláciu dvoch atribútov (morfológiu a gramnegativitu).

Nové pravidlo je zadávané expertom v prirodzenom jazyku. Preto musí byť najprv interpretované. Používa sa interpretovanie pravidiel využívajúce princíp *riadenia modelom*, keď modely pravidiel slúžia pre riadenie interpretácie a porozumenia novému pravidlu.

Na základe predbežného spracovania sa vyberá odpovedajúci model a pomocou neho sa pravidlo interpretuje. Jednotlivé interpretácie sa predkladajú expertovi, aby vybral tú, ktorá najlepšie zodpovedá jeho predstavám. Tento proces je označovaný ako *prvé hádanie.*

Po odsúhlasení interpretácie sa nové pravidlo porovnáva s modelom. V prípade, že v pravidle nie je obsiahnuté niečo, čo je v modeli (a teda sa často vyskytuje v iných podobných pravidlách), je expert na to upozornený, s tým, že konečné rozhodnutie je opäť na ňom. Toto je označované ako *druhé hádanie.*

Okrem interpretácie nových znalostí modely pravidiel môžu slúžiť aj pre:

- *abstraktný popis znalosti* (systém vie, čo vie).

  Toto má použitie pri vysvetľovaní. Na základe modelov pravidiel je možné zodpovedať otázky na vyššej úrovni ako iba pomocou trasovania použitia jednotlivých pravidiel.

- *systém vie, čo vie nedostatočne*

  Je to pokus, ako zdeliť systému, kde sú slabosti vytváranej bázy znalostí. Je možné zaviesť metriku modelov (napr. na základe použitých pravidiel a ich činiteľov istoty). Systém potom môže sám žiadať pravidlá, týkajúce sa nejakého modelu, ktorého ohodnotenie je nízke.

### Získavanie nových inštancií konceptov

Systém obsahuje znalosti o použitých reprezentáciách znalostí nultej úrovne (znalostí, týkajúcich sa spracovávanej problémovej oblasti). Tieto pozostávajú z:

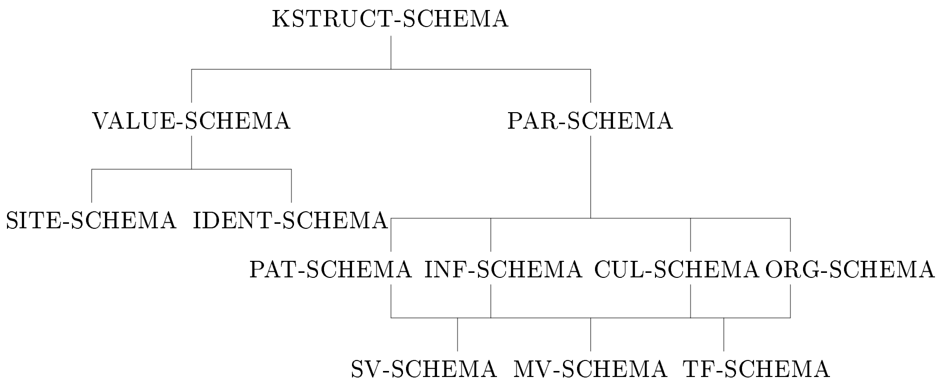
- *hierarchie schém*

  indikuje kategórie jednotlivých reprezentácií a vzťahy medzi nimi,

- *individuálnych schém*

  popisujú štruktúru jednoduchej reprezentácie.

Obr. 4.3 zobrazuje časť hierarchie schém.




Obr. 4.3: Ukážka hierarchie schém

KSTRUCT-SCHEMA (knowledge structure) je prázdna schéma. Na nižšej úrovni sú schémy pre hodnoty a atribúty. Ešte nižšie sú schémy pre atribúty, charakterizované objektom, ku ktorému patria (pacient, infekcia, kultúra a organizmus) a ešte nižšie sú delené podľa možnej hodnoty (single-valued, multiple-valued a true-false). Medzi schémami v hierarchii existujú vzťahy dedičnosti. Je zrejmé, že jedna schéma môže mať viac rodičov.

Vytvorenie nového konceptu znamená vytvorenie novej schémy a jej zaradenie do tejto hierarchie schém (teda určenie väzieb na už existujúce schémy).

Individuálna schéma má vlastnú štruktúru, ktorá sa skladá z piatich častí:

1. štruktúra inštancií schémy

   Určuje, čo obsahuje každá inštancia, vytvorená podľa danej schémy, spolu so spôsobom, ako sa získavajú potrebné informácie (položením otázky, informácia bude známa pri vytváraní inštancie, informáciu je potrebné odvodiť, ap.). Môžu tu byť položky:

   - plný názov inštancie + synonymá
   - vlastnosti, dôležité pre inštanciu (vlastný obsah inštancie)
   - dokumentačné informácie

2. vzťahy s inými dátovými štruktúrami

   obsahuje zoznam tých dátových štruktúr, ktoré musia byť informované o vytvorení novej inštancie,

3. smerníky na už existujúce inštancie danej schémy,

4. organizačné informácie

   smerníky na nadradené a podradené schémy, reprezentuje začlenenie danej schémy do hierarchie schém,

5. dokumentačné informácie

   autor, dátum a popis schémy.

Pre vytvorenie novej inštancie (zaradenie nového pojmu, ktorý je inštanciou nejakej schémy) je dôležité si uvedomiť, že vďaka existujúcej dedičnosti medzi schémami vytváraná inštancia dedí vlastnosti inštancií tých schém, ktoré sú nadradené schéme, ku ktorej sa vytvára daná inštancia. Nová inštancia je vytváraná v dvoch etapách:

1. Najprv je potrebné vyhľadať cestu od danej schémy (ku ktorej sa nová inštancia vytvára) ku koreňu v hierarchii schém. Je to potrebné preto, lebo vďaka dedičnosti medzi schémami údaje pre štruktúru inštancie a dátové štruktúry, ktoré majú byť modifikované, môžu byť obsiahnuté aj v nadradených schémach.

2. Postupuje sa spätne od koreňa k príslušnej schéme. Pre každú schému na tejto ceste prebieha proces vytvárania inštancie, pričom pozostáva vždy z troch krokov:
   (a) Použije sa prvá časť schémy - štruktúra nových inštancií. Po získaní potrebných údajov (od experta, z iných inštancií, ap.) sa všetky tieto zaradia do vytváranej inštancie.
   (b) Aktualizujú sa všetky tie dátové štruktúry, ktoré sú uvedené v druhej časti práve spracovávanej schémy.
   (c) Vytváraná inštancia sa pridá do tretej časti spracovávanej schémy.

# Časť II

# Zjemňovanie a vylepšovanie znalostí

# Kapitola 5

# Vylepšovanie znalostí

Budovanie znalostnej bázy neznamená iba neustále rozširovanie množstva znalostí nepretržitým vkladaním nových znalostí. Niekedy je potrebné sa spätne vrátiť aj k tým znalostiam, ktoré už boli zaradené do bázy znalostí. Tento návrat môže mať rozličnú podobu, napr. podobu opravy alebo zjemňovania (detailizácie). Vo všetkých takýchto prípadoch sa jedná o zvyšovanie celkovej kvality znalostnej bázy.

Z hľadiska vplyvu jednotlivých metód na objem znalostí je možné tieto metódy rozdeliť do troch základných skupín:

- dopĺňanie znalostí,
- redukcia znalostí,
- modifikácia znalostí.

Pri dopĺňaní znalostí sú nové znalosti nejakou formou odvádzané od už získaných znalostí (nejakým spôsobom nadväzujú na ne), ktoré toto doplnkové získavanie ovplyvňujú. Nové znalosti dopĺňajú a rozširujú (či už v zmysle záberu alebo detailnosti) už skôr získané znalosti.

Naproti tomu pri redukcii znalostí sa predpokladá, že v znalostnej báze môžu byť zachytené nie iba globálne platné zákonitosti, ale aj náhodné súvislosti alebo dokonca nepravdivé znalosti. Preto niektoré znalosti sú odstraňované, s cieľom ponechať v báze znalostí iba platné znalosti.

Modifikácia znalostí na rozdiel od predchádzajúcich dvoch skupín zachováva celkový objem znalostí. Jej cieľom je existujúce znalosti spresniť aby lepšie odrážali pomery v predmetnej oblasti.

Vylepšovanie znalostí je možné realizovať podobnými metódami ako prvotné získavanie znalostí (napr. niektorými priamymi metódami umožňujúcimi vhodnú interakciu s expertom). Zaujímavými sú však tie metódy, ktoré sú priamo viazané na existenciu množiny znalostí. Typicky tieto metódy získavajú dodatočné informácie o predmetnej oblasti (napr. množinu testovacích príkladov) a konfrontujú existujúce znalosti s týmito novými informáciami. Následkom tejto konfrontácie je zmena existujúcich znalostí zosúlaďujúca znalosti v báze dát so získanými dodatočnými informáciami.

## 5.1 Dopĺňanie znalostí

Príkladom nutnosti doplnenia bázy znalostí o nové znalosti je situácia, keď chápanie nejakej znalosti expertom je širšie ako jej explicitné vyjadrenie. Vtedy pohľad znalostného inžiniera (determinovaný práve explicitným vyjadrením znalosti) na túto znalosť sa líši od pohľadu experta. V tomto prípade báza znalostí nedostatočne zachytáva expertove znalosti.

Môže sa stať, že expert zdelí nejakú svoju znalosť, pričom pod ňou chápe nielen túto konkrétnu znalosť, ale implicitne aj niekoľko podobných znalostí. Napr. znalostný inžinier od experta získal pravidlo

$$\text{AK } podm_1 \text{ A } podm_2 \text{ A } \ldots \text{ A } podm_n \text{ POTOM záver}$$

Pritom expert implicitne predpokladá, že záver môže byť splnený aj za predpokladu, že niektorá podmienka splnená nie je (v tom prípade je šanca na splnenie záveru menšia, avšak stále existuje). Preto je potrebné doplniť bázu znalostí aj o pravidlá, ktoré niektoré podmienky neuvažujú. Celý postup pozostáva z niekoľkých krokov.

1. Zoradenie podmienok podľa dôležitosti

   Podmienka vystupujúca v pravidle môže byť dvojakého charakteru:

   **dominantná** – bez jej splnenia by záver nemohol platiť,

   **nedominantná** – záver môže byť splnený aj bez jej splnenia.

   Nedominantné podmienky môžu mať rôznu dôležitosť pre splnenie záveru. Je možné im priradiť číslo, vyjadrujúce ich relatívnu dôležitosť. Pri jej určení sa nedominantné podmienky zoradia podľa rastúcej dôležitosti a priradia sa im čísla 1, 2, ... (teda hodnota 1 je priradená najmenej významnej podmienke). Ak sú nejaké podmienky rovnako dôležité, priradí sa im tá istá hodnota.

Toto celkové zoradenie podmienok je možné získať priamo od experta, avšak častejší (a pre experta jednoduchší) prípad je, keď expert porovnáva navzájom dôležitosť dvojíc podmienok, a z týchto dvojíc sa určuje celkové zoradenie a priradenie príslušných hodnôt.

2. Generovanie doplnkových pravidiel

   K pôvodnému pravidlu sa vygeneruje sada doplnkových pravidiel. Tieto budú predstavovať kandidátov pre zaradenie do bázy znalostí. Vygenerujú sa tak, že každá podmienka, ktorá nie je dominantná, môže byť negovaná (teda nemusí byť splnená). Ak teda je $n$ nedominantných podmienok, je potrebné vygenerovať

   $$2^n - 1 \tag{5.1}$$

   doplnkových pravidiel (pôvodné pravidlo nie je nutné generovať ešte raz).

3. Usporiadanie doplnkových pravidiel

   Vygenerované doplnkové pravidlá sa usporiadajú, pričom kritérium tohto usporiadania bude odchýlka od pôvodného pravidla s ohľadom na dôležitosť jednotlivých nedominantných podmienok. Pre každé doplnkové pravidlo sa urobí súčet dôležitostí tých podmienok, ktoré sú voči pôvodnému pravidlu uvedené v negovanej forme. Doplnkové pravidlá budú usporiadané vzostupne podľa týchto súčtov.

4. Určenie hraničného pravidla

   Je potrebné zistiť, ktoré z doplnkových pravidiel je hraničné (ešte akceptovateľné). Všetky doplnkové pravidlá, nasledujúce za týmto hraničným pravidlom, sú zamietnuté. Všetky tie doplnkové pravidlá, ktoré sú pred týmto hraničným pravidlom, sú akceptované a budú vložené do bázy znalostí (vrátane hraničného pravidla).

5. Určenie váh

   Pre akceptované doplnkové pravidlá je potrebné určiť váhy s ktorými budú zaradené do bázy znalostí. Je samozrejme možné požiadať experta, aby určil váhy osobitne pre všetky akceptované doplnkové pravidlá. Je to však značne únavný postup a preto je snaha ho zjednodušiť.

Vychádza sa z predpokladu, že váhy doplnkových pravidiel by mali byť nižšie ako váha pôvodného pravidla. Expertovi stačí určiť váhu hraničného pravidla a váhy ostatných akceptovaných doplnkových pravidiel sa určia z lineárnej závislosti

$$w - \frac{w - w_{hran}}{\sum\limits_{i \in NEG(hran)} d_i} \sum_{i \in NEG} d_i \tag{5.2}$$

kde $w$ je váha pôvodného pravidla, $w_{hran}$ je váha hraničného pravidla (to sa najviac odlišuje od pôvodného pravidla), $\sum_{i \in NEG(hran)}$ je určená suma pre hraničné pravidlo a $\sum_{i \in NEG}$ je určená suma pre práve ohodnocované pravidlo.

Výhodou metódy je, že je jednoducho realizovateľná aj bez prítomnosti znalostného inžiniera, možno ju jednoducho realizovať dialógovým systémom, ktorý kontroluje navrhnutú bázu znalostí a snaží sa v spolupráci s expertom odvodiť čo najviac implicitných pravidiel.

Pre ilustráciu uvažujme nasledovné pravidlo tvaru:

AK $p_1$ A $p_2$ A $p_3$ POTOM z (0.8)

Podľa experta je podmienka $p_3$ dominantná. Pri porovnaní podmienok $p_1$ a $p_2$ expert vyhlásil, že podmienka $p_1$ je dôležitejšia ako podmienka $p_2$. Preto podmienke $p_2$ bude priradená hodnota 1 a podmienke $p_1$ hodnota 2.

Pretože iba dve podmienky nie sú dominantné, budú vygenerované tri doplnkové pravidlá:

dp1: AK $\neg p_1$ A $p_2$ A $p_3$ POTOM z
dp2: AK $p_1$ A $\neg p_2$ A $p_3$ POTOM z
dp3: AK $\neg p_1$ A $\neg p_2$ A $p_3$ POTOM z

pre ktoré sa po určení závažnosti ich odlišnosti voči pôvodnému pravidlu získa usporiadanie $dp2$ ($\sum_{i \in NEG(dp2)} d_i = 1$), $dp1$ ($\sum_{i \in NEG(dp1)} d_i = 2$), a nakoniec $dp3$ ($\sum_{i \in NEG(dp3)} d_i = 3$).

Ak predpokladáme, že expert určil ako hraničné pravidlo $dp1$, potom sa do bázy znalostí budú vkladať pravidlá $dp2$ a $dp1$. Vygenerované pravidlo $dp3$ bude zamietnuté.

Expert môže určiť váhu hraničného pravidla priamo alebo relatívne voči pôvodnému pravidlu. Ak v našom prípade expert určil, že podpora záveru pravidlom $dp2$ je iba 3/4 oproti pôvodnému pravidlu, potom váhy akceptovaných doplnkových pravidiel budú 0.6 ($dp1$) a 0.7 ($dp2$).

## 5.2 Redukcia znalostí

Pri získavaní znalostí metódami induktívneho učenia z typických príkladov môže nastať prípad, že vytvorená rozhodovacia procedúra je "preučená" – reprezentuje nielen závislosti platné v celej predmetnej oblasti, ale aj náhodné vzťahy a súvislosti, zapríčinené voľbou príkladov učiacej množiny. Preučenie môže byť spôsobené aj chybami v príkladoch. Chybná hodnota atribútu alebo príznaku triedy v príklade môže zapríčiniť, že kvôli správnemu "zaradeniu" tohto príkladu sa konštruuje dlhý podstrom. Hrozba preučenia je nevýhoda perfektných rozhodovacích procedúr, ktoré správne klasifikujú všetky príklady z učiacej množiny.

Rozhodovacie stromy, vytvárané podľa príkladov z učiacej množiny, sú zvyčajne voči tejto množine perfektné (bezchybne klasifikujú všetky príklady z tejto množiny), sú optimalizované voči príkladom z tejto množiny. Toto však nič nehovorí o spoľahlivosti klasifikácie tých príkladov, ktoré neboli súčasťou učiacej množiny. Tá môže dosahovať aj pomerne značnú chybu.

Jednou z metód, ako znížiť chybu pri klasifikácii neznámych príkladov, je zjednodušovanie rozhodovacích stromov. Je založené na myšlienke urobiť rozhodovací strom menej spoľahlivý voči príkladom z učiacej množiny (teda už nebude prispôsobený práve pre túto množinu). Tým síce narastie chyba klasifikácie príkladov z učiacej množiny, ale sa zároveň môže zmenšiť chyba klasifikácie neznámych príkladov. Najjednoduchšie to je možno dosiahnuť pomocou orezania pôvodného stromu, nahradením niektorých jeho podstromov listovými uzlami.

Jednotlivé metódy sa líšia navzájom v závislosti od toho, či pracujú s učiacou množinou príkladov (teda s príkladmi na základe ktorých boli rozhodovacie stromy vytvorené) a/alebo s testovacou množinou príkladov.

### Orezávanie cena – komplexnosť

Tento spôsob predstavuje dvojstupňový proces. V prvej fáze sa konštruuje sekvencia stromov $T_0, T_1, \ldots, T_k$, kde $T_0$ je originálny rozhodovací strom a $T_k$ je predstavovaný iba jedným listovým uzlom. Každý $T_{i+1}$ je vytvorený z $T_i$ nahradením jedného podstromu listovým uzlom. V druhej fáze je z tejto sekvencie vyberaný jeden strom ako výsledný zjednodušený rozhodovací strom.

Uvažujme rozhodovací strom $T$ používaný pre klasifikáciu $n$ príkladov z učiacej množiny príkladov, na základe ktorej bol $T$ vygenerovaný. Nech chybne klasifikuje $E$ príkladov. Ak označíme počet jeho listových uzlov ako

$L(T)$, potom možno ako mieru komplexnosti a ceny použiť

$$\frac{E}{n} + \alpha L(T) \tag{5.3}$$

kde $\alpha$ je parameter, vážiaci veľkosť stromu voči jeho presnosti.

Predpokladajme, že podstrom $S$ stromu $T$ nahradíme najlepším možným listovým uzlom (takým, aby výsledná chyba klasifikácie bola čo najmenšia). Vo všeobecnosti nový rozhodovací strom bude chybne klasifikovať o $m$ príkladov viac a bude mať o $L(S) - 1$ menej listových uzlov. Tento nový strom bude mať rovnakú hodnotu miery ceny a komplexnosti ako pôvodný, ak

$$\alpha = \frac{m}{n(L(S) - 1)} \tag{5.4}$$

Pre vytvorenie $T_{i+1}$ z $T_i$ je potrebné preveriť každý nelistový podstrom $S$ stromu $T_i$, aby sa našla minimálna hodnota $\alpha$. Ten podstrom, ktorý túto minimálnu hodnotu dosahuje, je nahradený najlepším listom (tak aby dosahovaná chyba klasifikácie bola minimálna).

Druhá etapa opúšťa model cena–komplexnosť a pokúša sa vybrať jeden $T_i$ z vygenerovanej postupnosti stromov na základe spoľahlivosti voči testovacej množine príkladov. Táto nech obsahuje $n'$ príkladov a nech $T_i$ klasifikuje z nich $E'_i$ chybne. Nech

$$E' = \min(E'_i) \tag{5.5}$$

je minimálny počet chýb, dosiahnutých na testovacej množine príkladov rozhodovacími stromami $T_0, \ldots, T_k$. Potom štandardná chyba sa určí ako

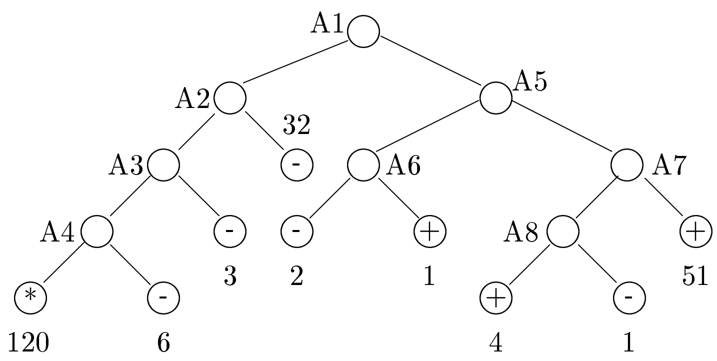
$$SE(E') = \sqrt{\frac{E'(n' - E')}{n'}} \tag{5.6}$$

Konečný vybraný strom je najmenší z tých, ktorých chyba, dosiahnutá na množine testovacích príkladov, spĺňa nerovnosť

$$E'_i \leq E' + SE(E') \tag{5.7}$$

Nevýhodou metódy je nutnosť existencie testovacej množiny príkladov, použitie dvoch rôznych prístupov a veľká výpočtová záťaž (je potrebné generovať značný počet stromov).

Ako príklad tohto postupu počas prvej fázy uveďme originálny perfektný strom $T_0$ podľa Obr. 5.1 (čísla znamenajú počty príkladov, súvisiacich s danými uzlami).




Obr. 5.1: Ilustračný rozhodovací strom - tvar $T_0$

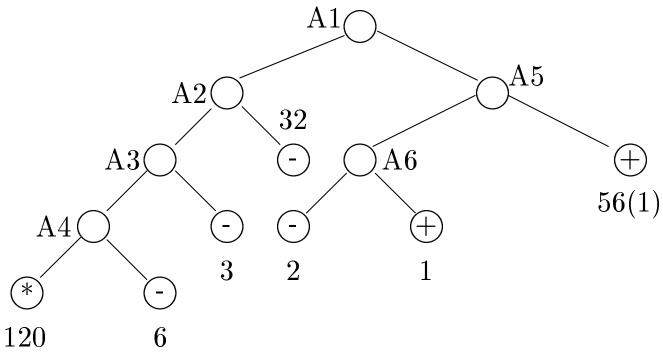
Pre vygenerovanie ďalšieho člena sekvencie $T_1$ je potrebné nahradiť jeden podstrom listovým uzlom. Hodnoty $\alpha$ pre jednotlivé podstromy sú v tabuľke 5.1.

| S | náhrada | m | L(S) | $\alpha$ |
|---|---|---|---|---|
| A1 | * | 100 | 9 | 0.0568 |
| A2 | * | 41 | 4 | 0.0621 |
| A3 | * | 9 | 3 | 0.0205 |
| A4 | * | 6 | 2 | 0.0273 |
| A5 | + | 3 | 5 | 0.0034 |
| A6 | - | 1 | 2 | 0.0045 |
| A7 | + | 1 | 3 | 0.0023 |
| A8 | + | 1 | 2 | 0.0045 |

Tabuľka 5.1:

Keďže hodnota $\alpha$ je najmenšia pre podstrom s koreňom $A_7$, nasledujúci strom v generovanej sekvencii bude mať tvar podľa Obr. 5.2, kde číslo v zátvorke vyjadruje počet chybne klasifikovaných príkladov z učiacej množiny.

Podobným spôsobom sa utvoria ďalšie členy sekvencie, až posledný $T_5$ bude vyjadrený jedným listovým uzlom s triedou "*".




Obr. 5.2: Ilustračný rozhodovací strom - tvar $T_1$

## Redukcia chyby

Namiesto vytvárania sekvencie rozhodovacích stromov a následného výberu poskytuje priamejšiu cestu. Opäť sa predpokladá oddelená testovacia množina príkladov. Príklady tejto množiny sú testované originálnym rozhodovacím stromom. Potom pre každý nelistový podstrom $S$ stromu $T$ sa preskúma zmena počtu chybne klasifikovaných príkladov, ak sa $S$ nahradí najlepším možným listom. Ak nový strom poskytuje rovnaký alebo menší počet chýb a $S$ neobsahuje žiadny podstrom s takouto vlastnosťou, potom $S$ je nahradený daným listom.

Tento proces pokračuje až dovtedy, pokiaľ ľubovoľná ďalšia náhrada zvyšuje počet chybne klasifikovaných príkladov z testovacej množiny.

Pri tomto postupe sa tiež generuje sekvencia stromov, ale už nie je potrebné z nej vyberať výsledný strom osobitným spôsobom (generovanie stromov priamo k nemu smeruje, je to posledný z vygenerovanej sekvencie). Nevýhodou je požiadavka na dokonalosť testovacej množiny (mohlo by sa stať, že tá časť stromu, ktorá nebola zastúpená v testovacej množine, bude odstránená).

## Pesimistické orezávanie

Nech je originálny rozhodovací strom $T$ (nemusí byť perfektný) použitý ku klasifikácii príkladov z učiacej množiny. Nejaký listový uzol potom klasifikuje $k$ z týchto príkladov, pričom $j$ z nich je klasifikovaných chybne. Miera

$j/k$ neposkytuje vhodný odhad chyby pre príklady mimo učiacej množiny. Preto pre odhad počtu chybne klasifikovaných príkladov sa používa hodnota $j + 1/2$.

Nech $S$ je podstrom rozhodovacieho stromu $T$, obsahuje $L(S)$ listov a $\sum j$ a $\sum k$ sú korešpondujúce počty príkladov (chybne klasifikovaných a všetkých) cez všetky listy podstromu $S$. Pesimistický odhad je, že podstrom $S$ bude chybne klasifikovať

$$E' = \sum j + \frac{L(S)}{2} \tag{5.8}$$

prípadov z $\sum k$ príkladov (pri klasifikácii neznámych príkladov). Štandardná chyba sa teraz určí ako

$$SE(E') = \sqrt{\frac{(\sum j + \frac{L(S)}{2})(\sum k - \sum j - \frac{L(S)}{2})}{\sum k}} \tag{5.9}$$

Ak podstrom $S$ je nahradený najlepším listovým uzlom, potom $E$ je počet chybne klasifikovaných príkladov z učiacej množiny. Táto náhrada môže byť uskutočnená, ak

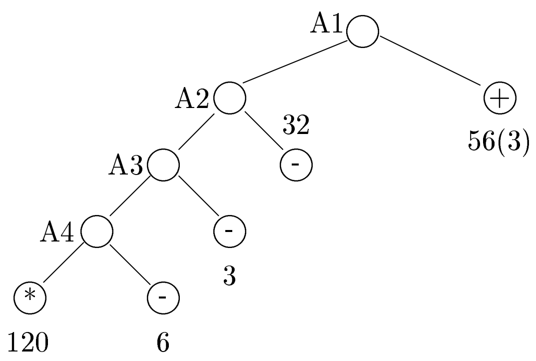
$$E + 0.5 \leq E' + SE(E') \tag{5.10}$$

Ak je viac možností, tak sa realizuje najväčšie orezanie.

Pre príklad originálneho rozhodovacieho stromu podľa Obr. 5.1 dostaneme teraz tabuľku hodnôt 5.2 (skúma sa orezanie iba jedného podstromu, v príklade sa neuvažuje orezanie viacerých podstromov súčasne). Výsledný orezaný strom bude mať tvar podľa 5.3.

| S | A1 | A2 | A3 | A4 | A5 | A6 | A7 | A8 |
|---|---|---|---|---|---|---|---|---|
| $\sum$ k | 220 | 161 | 129 | 126 | 59 | 3 | 56 | 5 |
| $\sum$ j | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| L(S) | 9 | 4 | 3 | 2 | 5 | 2 | 3 | 2 |
| E' | 4.5 | 2 | 1.5 | 1 | 2.5 | 1 | 1.5 | 1 |
| SE | 2.1 | 1.41 | 1.22 | 1 | 1.55 | 0.82 | 1.21 | 0.89 |
| SE+E' | 6.6 | 3.41 | 2.72 | 2 | 4.05 | 1.82 | 2.71 | 1.89 |
| E+0.5 | 100.5 | 41.5 | 9.5 | 6.5 | 3.5 | 1.5 | 1.5 | 1.5 |

Tabuľka 5.2:




Obr. 5.3: Pesimistické orezávanie - výsledný rozhodovací strom

Výhodou metódy je, že nie je potrebná testovacia množina príkladov a že všetky nelistové uzly sú preverené iba raz, aby sa zistilo, či majú byť orezané alebo nie.

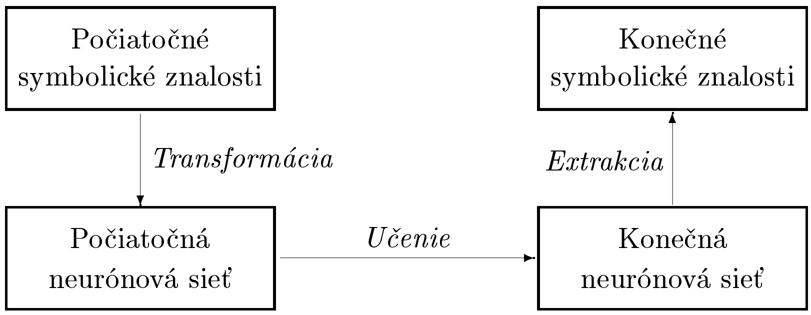
## 5.3 Modifikácia znalostí

Pri tejto skupine metód sa vlastne nejaké znalosti nahrádzajú inými znalosťami. Celkový objem znalostí ostáva nezmenený. Typickým problémom, vyžadujúcim takýto prístup, je jeden z veľkých problémov pri tvorbe znalostných báz, obsahujúcich neurčité znalosti s numericky reprezentovanou neurčitosťou – správne nastavenie numerickej hodnoty tejto neurčitosti. Konkrétna hodnota neurčitosti sa môže aj niekoľkokrát zmeniť. Často sa najprv nastaví iba približne a až neskôr je spresnená tak, aby celkové chovanie bázy znalostí zodpovedalo predstavám experta alebo súboru testovacích príkladov.

Jednou možnosťou ako riešiť takúto úlohu je aj využitie analógie váh produkčných pravidiel a váh hrán neurónových sietí. Základná myšlienka využitia neurónových sietí pre vytváranie znalostných báz je na obr. 5.4.

Vo vytvorenej báze znalostí, obsahujúcej neurčité produkčné pravidlá, sa nejakým spôsobom nastavia inicializačné hodnoty váh pravidiel (vyjadrujúce počiatočný odhad neurčitosti týchto pravidiel). Takéto pravidlá sú potom transformované na neurónovú sieť s inicializovanými váhami hrán.

Na základe trénovacích príkladov sa váhy hrán nastavia pomocou neja-




Obr. 5.4: Využitie neurónových sietí

kého učiaceho algoritmu (sieť sa natrénuje voči príkladom) a výsledná neurónová sieť je spätne transformovaná na produkčné pravidlá, ktoré budú mať svoje váhy nastavené v zhode s trénovacími príkladmi.

Počiatočné znalosti sa budú líšiť od konečných znalostí iba hodnotami neurčitosti jednotlivých pravidiel (samotná štruktúra pravidiel ostane nezmenená).

Pre daný účel je možné použiť štandardné mnohovrstvové neurónové siete, ktoré oproti bežným sieťam budú mať vlastnosti:

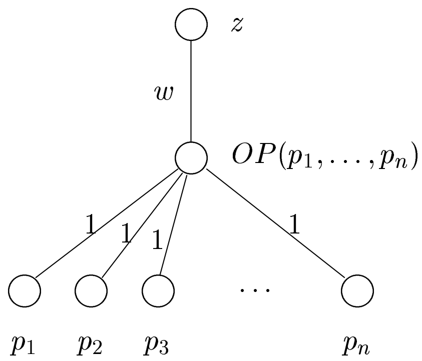
- budú to "riedke" siete - v báze znalostí neexistuje toľko veľa náväzností medzi jednotlivými pravidlami ako je zvyčajne prepojení medzi neurónmi susedných vrstiev,
- neuróny môžu byť prepojené nielen s neurónmi susednej vrstvy, ale aj so vzdialenejšími vrstvami,
- v sieti budú existovať neuróny s rôznymi aktivačnými a výstupnými funkciami,
- neuróny budú mať nezvyčajné aktivačné a výstupné funkcie.

### Transformácia znalostí

Jedným z problémov je transformácia množiny produkčných pravidiel na neurónovú sieť a späť. V báze znalostí sa zvyčajne nachádzajú produkčné pravidlá tvaru

AK $OP(p_1, p_2, \ldots, p_n)$ POTOM z (w)

určujúce silu (resp. neurčitosť) $w$ vzťahu medzi záverom $z$ a kombináciou podmienok. Funkcia $OP$ reprezentuje nejakú logickú funkciu, ktorou sú jednotlivé podmienky viazané navzájom. Takéto pravidlo môže byť transformované na neurónovú sieť tvaru podľa obr. 5.5.




Obr. 5.5: Časť neurónovej siete

Záver a každá podmienka pravidla predstavujú *regulárny* neurón, zatiaľ čo kombinácia podmienok je transformovaná na *operačný* neurón.

Váhy hrán vchádzajúcich do operačného neurónu sú vždy rovné 1 – sú to "tvrdé" váhy (teda tieto váhy nebudú modifikované, ale ostanú nezmenené). Váhy hrán vchádzajúcich do regulárneho neurónu sú vlastne váhy tých pravidiel, ktoré potvrdzujú alebo popierajú záver, reprezentovaný daným neurónom. Toto sú tie váhy, ktoré je potrebné nastaviť na základe dostupných trénovacích príkladov.

Konkrétne aktivačné a výstupné funkcie jednotlivých neurónov závisia od konkrétneho mechanizmu reprezentácie neurčitosti a narábania s ňou počas odvodzovania nových znalostí. Napr. v prípade činiteľov istoty (EMYCIN) výstupné funkcie všetkých neurónov majú tvar lineárnej funkcie. Aktivačná funkcia regulárneho neurónu je v tvare

$$y^+ = 1 - \prod_{i \in P} (1 - x_i w_i) \tag{5.11}$$

$$y^- = -1 + \prod_{i \in N} (1 + x_i w_i) \tag{5.12}$$

$$F = \frac{y^+ + y^-}{1 - \min(|\, y^+ \,|, |\, y^- \,|)} \tag{5.13}$$

kde $P$ je množina potvrdzujúcich a $N$ množina popierajúcich príspevkov.

Aktivačné funkcie operačných neurónov vyjadrujú analógie logických funkcií (napr. AND ako minimálna hodnota vstupov).

Vďaka týmto nezvyčajným funkciám generované siete sa trochu líšia od klasických neurónových sietí. Pre zohľadnenie tejto odlišnosti sa nazývajú *expertnými* sieťami.

### Učenie expertných sietí

Pre učenie expertných sietí je možné použiť napr. štandardný algoritmus využívajúci pohyb v smere záporného gradientu chyby, napr. algoritmus spätného šírenia chyby resp. niektorú jeho modifikáciu. Nevýhodou je nutnosť presne poznať derivácie aktivačných a výstupných funkcií.

Inou možnosťou je použitie algoritmov, nevyžadujúcich presnú hodnotu derivácie napr. algoritmus *R-propagation*, vyžadujúci iba znamienka derivácií.

Osobitným problémom pri učení je definičný obor jednotlivých váh. Pokiaľ u neurónových sietí je $(-\infty, \infty)$, v bázach znalostí ich hraničné hodnoty závisia od spôsobu reprezentácie neurčitosti. V prípade činiteľov istoty to je interval $< -1, 1 >$. Preto ak niektorá váha v procese učenia presiahne hranice tohto intervalu, je potrebné ju zmeniť. Je možné použiť metódu faktoru zmenšenia $\delta$:

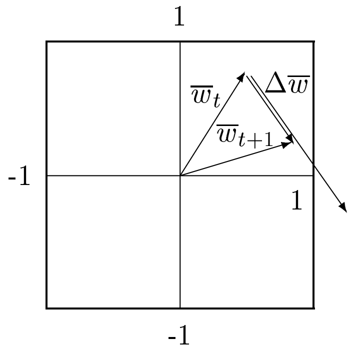
$$\Delta\overline{w}' = \delta\, \Delta\overline{w} \qquad 0 < \delta < 1 \tag{5.14}$$

kde pôvodná zmena váh je určená v danom kroku podľa použitého učiaceho algoritmu. Tento prípad je schematicky znázornený na obr. 5.6.

Pri konkrétnej realizácii je potrebné stanoviť nejakú vhodnú stratégiu pre určovanie hodnoty $\delta$.

Inou možnosťou je priame použitie metódy orezania

$$w_i = \begin{cases} 1 & w_i + \Delta w_i > 1 \\ -1 & w_i + \Delta w_i < -1 \\ w_i + \Delta w_i & inak \end{cases} \tag{5.15}$$




Obr. 5.6: Faktor zmenšenia

Obe metódy zavádzajú do procesu učenia dodatočnú chybu, ktorá sa môže prejaviť v predĺžení procesu učenia alebo aj až v značnej degradácii výkonnosti algoritmu, vedúcej k jeho praktickej nepoužiteľnosti.

# Kapitola 6

# Transformácia znalostí

Transformácia znalostí vlastne znamená zmenu reprezentácie znalostí. Dôvodom tejto zmeny môžu byť rôzne. Príkladmi sú požiadavky použitého inferenčného mechanizmu na spôsob reprezentácie znalostí (napr. použitie inej metódy vyjadrenia a práce s neurčitými znalosťami) či charakter predmetnej oblasti (napr. práca s množinami ekvivalentných podmienok). Táto transformácia z hľadiska obsahu znalostí môže znamenať dva druhy zmien:

- kvantitatívne,
- kvalitatívne.

Pri kvantitatívnych zmenách znalosti ostávajú tie isté, mení sa iba ich forma (celkový *vonkajší* rozsah znalostí sa môže meniť, pretože zdrojová a cieľová reprezentácia môžu mať rôznu kompaktnosť – ich *vnútorný* rozsah však ostáva nezmenený). V skutočnosti sa teda nejedná o získavanie znalostí. Naproti tomu pri druhom type zmien dochádza aj k modifikácii obsahu znalostí (ktorá môže byť sprevádzaná ich doplnením alebo redukciou), ktorá vyžaduje získavanie nových znalostí, na základe ktorých sa daná modifikácia realizuje.

Znalosti vyjadrené pomocou nejakého *zdrojového* reprezentačného mechanizmu sú prenášané do *cieľovej* reprezentácie. Tento prenos sa môže realizovať

- priamo,
- prostredníctvom jednej alebo viacerých medzireprezentácií.

Druhý spôsob je s výhodou použiteľný vtedy, ak samotná transformácia znalostí z jednej formy do druhej je súčasne kombinovaná s nejakými zmenami znalostí. Vtedy sa zvolí taká medzireprezentácia, ktorá umožní jednoduchú realizáciu požadovaných zmien znalostí.

## 6.1 Transformácia s modifikáciou znalostí

Jednou z metód použiteľných pre transformáciu znalostí z jedného tvaru do druhého je *subsymbolická modifikácia.* Táto transformačná metóda (zahŕňajúca aj revíziu a prípadnú korekciu znalostí) je zameraná na znalosti reprezentované v symbolickej podobe. Pre svoju činnosť využíva dopredné neurónové siete ako empirický učiaci algoritmus. Základná štruktúra metódy je totožná so štruktúrou znázornenou na obr. 5.4.

Symbolické znalosti, ktoré nemusia byť ani kompletné ani korektné, sú transformované do tvaru neurónovej siete, ktorá má podobu viacvrstvovej doprednej siete. Táto transformácia zo symbolickej do subsymbolickej podoby umožňuje pre modifikáciu znalostí využiť štandardné prostriedky z oblasti neurónových sietí.

Váhy vytvorenej neurónovej siete sú nastavené tak, aby sieť bola v súlade s množinou klasifikovaných príkladov. Toto učenie môže mať podobu štandardného učiaceho algoritmu, napr. algoritmu spätného šírenia chyby alebo ľubovoľnej inej metódy pre optimalizáciu váh dopredných neurónových sietí. Po skončení fázy učenia znalosti, reprezentované neurónovou sieťou, sú konzistentné s trénovacími príkladmi.

Natrénovaná neurónová siete je spätne transformovaná na znalosti v symbolickej podobe. Výsledkom je množina pravidiel ktorá sa môže od pôvodnej množiny pravidiel líšiť počtom jednotlivých pravidiel, tvarom jednotlivých pravidiel a dokonca aj formou reprezentácie.

### Vytvorenie neurónovej siete

Pod vytvorením neurónovej siete sa rozumie nielen definovanie jej topológie ale aj stanovenie počiatočných odhadov váh tak, aby vytvorená sieť odrážala pôvodné pravidlá. Samotná tvorba siete pozostáva z niekoľkých krokov:

- prepísanie pravidiel,
- mapovanie štruktúry,
- odhad váh,

- doplnenie vstupných neurónov,
- doplnenie hrán,
- perturbácia.

Na začiatku je potrebné upraviť pravidlá do formátu, ktorý spriehľadňuje ich hierarchickú štruktúru. Ak existuje skupina pravidiel s rovnakým záverom, tak každé z týchto pravidiel, ktorého predpoklad je tvorený konjunkciou tvrdení, je nahradené dvoma pravidlami. Jedno z týchto nových pravidiel má pôvodný záver a jednoduchý novo vytvorený predpoklad. Druhé má toto novovytvorené tvrdenie ako svoj záver a originálne predpoklady nahrádzaného pravidla. Táto náhrada zabezpečí, že každé tvrdenie, ktoré má funkciu záveru aspoň v jednom pravidle, je zviazané iba s jedným spôsobom kombinovania predpokladov tohto tvrdenia – konjunkciou alebo disjunkciou.

Upravená množina pravidiel je mapovaná na štruktúru neurónovej siete. Topológia siete je vytváraná jednoducho – tvrdeniam nachádzajúcim sa v pravidlách zodpovedajú uzly siete. Ak dve tvrdenia sú navzájom viazané pomocou nejakého pravidla, tak tento vzťah je reprezentovaný hranou pričom hrana vystupuje z neurónu reprezentujúceho tvrdenie v predpoklade pravidla a vstupuje do neurónu zastupujúceho záver pravidla.

Jednotlivé neuróny sú usporiadané do vrstiev (nultá vrstva obsahuje všetky vstupné neuróny). Pri tomto zatrieďovaní číslo vrstvy do ktorej patrí nejaký neurón je dané najdlhšou cestou od daného neurónu k vrstve vstupných neurónov.

Tie tvrdenia, ktoré sa vyskytujú aj v predpokladoch aj v záveroch pravidiel budú tvoriť neuróny umiestnené v skrytých vrstvách. Všetky tvrdenia vyskytujúce sa iba v záverovej časti pravidiel budú umiestnené vo výstupnej vrstve.

Ohodnotenie hrán siete váhami (počiatočný odhad) musí zohľadňovať pôvodné pravidlá. Neurón reprezentujúci nejaké tvrdenie môže získať signifikantnú aktiváciu (aktiváciu blízko hodnote 1) pri vyjadrení doménových znalostí v subsymbolickom tvare iba ak dané tvrdenie môže byť odvodené pomocou pravidiel pri vyjadrení tých istých znalostí v symbolickom tvare. V opačnom prípade daný neurón musí byť neaktívny (s aktiváciou blízko hodnote 0).

Pre zjednodušenie jednotlivým hranám sa priradzujú rovnaké váhy – kladným hranám váha $\omega$ a záporným (reprezentujúcim negáciu tvrdenia v predpoklade) váha $-\omega$. Pre konkrétnu hodnotu váh sa používa napr. $\omega = 4$.

Pre neurón reprezentujúci záver pravidla, ktorého predpoklad je tvorený konjunkciou, sa prah určuje $(P - 1/2)\omega$ kde $P$ je počet kladných hrán vstupujúcich do daného neurónu. Takýto neurón bude signifikantne aktivovaný iba v prípade, že všetky do neho vstupujúce kladné hrany nesú signál blízky hodnote 1 a všetky záporné hrany zase signál blízky 0.

Pre neurón reprezentujúci záver niekoľkých disjunktívnych pravidiel sa zase prah nastaví na hodnotu $(1/2 - N)\omega$, kde $N$ je počet negatívnych hrán vstupujúcich do daného neurónu. Toto nastavenie umožňuje aktiváciu neurónu v prípade, že aspoň jedna kladná hrana nesie signál blízky hodnote 1 alebo aspoň jedna záporná hrana nenesie signál tejto hodnoty.

Keďže pôvodná skupina pravidiel nemusí byť úplná, nemusí zohľadňovať každú vstupnú vlastnosť, ktorá je podstatná pre výsledné rozhodnutie, reprezentované výstupnou vrstvou neurónov. Ak existuje nejaký faktor, dosiaľ nezohľadnený, je pridaný do siete vo forme vstupného neurónu.

Z rovnakého dôvodu je sieť doplnená o všetky chýbajúce hrany medzi neurónmi dvoch susedných vrstiev. To zabezpečí rozšírenie pôvodne daných vzťahov medzi jednotlivými tvrdeniami o nové vzťahy (ktoré sa môžu ukázať ako nesprávne – vtedy v konečnom dôsledku budú eliminované). Všetky tieto nové hrany sa pridávajú do siete s nulovými váhami.

Záverečným krokom vo vytváraní neurónovej siete je pridanie malého náhodného čísla ku každej váhe. Táto perturbácia je príliš malá na to aby zmenila interpretáciu vzťahov zakódovaných v sieti. Zároveň je však dosť veľká na to, aby mohla reprezentovať úmyselné "rozladenie" siete, uľahčujúce rozbeh učiaceho algoritmu.

## Učenie siete podľa príkladov

Siete použité pre reprezentáciu množiny pravidiel sú bežné dopredné siete. Ak modifikované pravidlá vytvárajú hierarchické štruktúry (existujú také tvrdenia, ktoré sa vyskytujú súčasne v predpokladoch niektorých pravidiel aj ako závery iných pravidiel), výsledná neurónová sieť bude mať tvar mnohovrstvovej siete, kde v rámci susedných vrstiev existujúce hrany spájajú ľubovoľný neurón jednej vrstvy s každým neurónom druhej vrstvy. Ak však doménové znalosti nie sú hierarchicky štruktúrované, tak výsledná sieť nebude obsahovať skryté neuróny.

Neuróny týchto sietí budú predstavovať procesné prvky so vstupnou funkciou

$$\sum_j w_{ij} a_j - \theta_i \tag{6.1}$$

a výstupnou funkciou

$$\frac{1}{1 + e^{-sx}} \tag{6.2}$$

pričom $\theta_i$ označuje prah i-teho neurónu, $w_{ij}$ váhy hrán spájajúcich i-ty neurón s jeho vstupmi, a $s$ je parameter ovplyvňujúci strmosť výstupnej funkcie.

Pretože takáto štruktúra predstavuje bežne používaný tvar neurónovej siete, je možné využiť existujúce algoritmy učenia dopredných sietí, napr. algoritmus spätného šírenia chyby.

Neurónová sieť reprezentuje pravidlá bez neurčitosti. Jednotlivé neuróny môžu byť preto alebo maximálne aktívne (majú aktiváciu blízku hodnote 1) alebo neaktívne (aktivácia blízka hodnote 0). Preto parameter $s$ vo výstupnej funkcii musí byť volený tak, aby výstupná funkcia aproximovala skokovú funkciu.

Keďže každá hrana bude niesť signál rovný váhe tejto hrany (alebo žiadny signál), je možné použiť vstupnú funkciu v tvare

$$\sum_{j \in aktiv} w_{ij} - \theta_i \tag{6.3}$$

a teda pri spätnej extrakcii pravidiel je postačujúce uvažovať iba váhy jednotlivých hrán siete.

Pretože každý neurón má zaručený nezáporný výstup (ktorý môže byť vstupom do ďalšieho neurónu), hrany s negatívnou váhou predstavujú negatívne tvrdenia, zatiaľ čo pozitívne vážené hrany reprezentujú tvrdenia v priamom (nenegovanom) tvare.

### Extrakcia pravidiel z neurónovej siete

Poslednou etapou je vyhľadávanie pravidiel v štruktúre trénovanej neurónovej siete. Predpoklady výsledných pravidiel budú mať tvar

M z N tvrdení je pravdivých

Takýto tvar umožňuje pre M=1 reprezentovať disjunktívne pravidlá a pre M=N zase pravidlá v konjunktívnej forme ako špeciálne prípady. Samotný algoritmus extrakcie pravidiel pozostáva z niekoľkých krokov (opakovaných pre každý skrytý a výstupný neurón):

- formovanie skupín hrán,
- úprava váh hrán,
- redukcia hrán,
- optimalizácia hodnoty prahu,
- extrakcia kvantitatívnych pravidiel,
- tvorba kvalitatívnych pravidiel.

Pri tvorbe pravidiel zastupujúcich jeden neurón spolu s vstupnými hranami tohto neurónu sa vychádza z predpokladu, že váhy hrán nemajú jedinečný význam (a teda fakt že jedna váha je o jednu stotinu väčšia ako iná ešte neznamená jej väčšiu dôležitosť) ale že slúžia pre vytváranie ekvivalentných tried predpokladov, v rámci ktorých všetky predpoklady majú rovnakú dôležitosť.

Pre vytváranie týchto ekvivalentných tried je možné použiť štandardné zhlukovacie algoritmy. Jedna z možných metód postupne kombinuje dvojice najsusednejších zhlukov do jedného spoločného zhluku. Na začiatku každý zhluk obsahuje jednu hranu. Zhlukovanie sa zastaví keď vzdialenosť medzi žiadnymi dvoma zhlukmi nie je menšia ako stanovená vzdialenosť (napr. 0.25).

Váhy všetkých hrán v nejakom zhluku sú nastavené na rovnakú hodnotu – priemer váh všetkých hrán v danom zhluku. Tým celková váha zhluku ostáva nezmenená.

Všetky vytvorené zhluky tak zahŕňajú všetky možné hrany medzi dvoma neurónmi susedných vrstiev. Pre odstránenie zbytočných hrán je potrebné eliminovať tie zhluky hrán, ktoré nemajú vplyv na aktiváciu neurónu. Typicky sú to malé zhluky hrán s nízkymi váhami. Samotnú elimináciu je možné realizovať dvoma spôsobmi:

1. algoritmicky

   Hľadá sa taký zhluk hrán, ktorý nemá vplyv na to, či celková vstupná aktivácia prevýši prah alebo nie. Pre toto je potrebné určovať celkovú možnú aktiváciu ktorú môže zhluk spôsobiť. Táto aktivácia je potom porovnávaná s úrovňami ktoré sú dosiahnuteľné vzhľadom k váham vstupných hrán neurónu. Tie zhluky, ktoré neovplyvňujú dosiahnutie prahu, sú odstránené.

2. heuristicky

   Nerobí sa teoretický rozbor užitočnosti každého zhluku ale empiricky sa overuje na dostupných príkladoch či aktivácia neurónu sa zmení, ak hrany v niektorom zhluku sa neberú do úvahy. Ak pre niektorý z príkladov sa získané aktivácie líšia, testovaný zhluk je označený ako nutný.

Po redukcii počtu hrán v sieti (jej orezaní) sa zopakuje fáza učenia. Terajším cieľom už nie je meniť váhy hrán ale optimalizovať hodnotu prahu. Je to nutný krok, pretože odstránenie hrán mohlo spôsobiť zvýšenie chyby siete nad prijateľnú hranicu.

Pre výslednú sieť je možné uskutočniť prepis siete do sústavy pravidiel, vyjadrujúcich kvantitatívne vzťahy v sieti. Tieto pravidlá sú priamym prepisom neurónovej siete a majú tvar

$$\text{AK} \quad prah <\mid zhluk_1 \mid váha(zhluk_1) \quad + \quad \ldots \quad \text{POTOM} \quad \text{záver}$$

kde $\mid X \mid$ označuje počet prvkov množiny X. Uvedený tvar explicitne stanovuje podmienku platnosti tvrdenia reprezentovaného neurónom.

Tieto pravidlá musia byť transformované na tvar neobsahujúci kvantitatívne hodnoty váh a prahu ale iba kombinácie tvrdení v predpoklade, postačujúcich pre dokázanie tvrdenia, asociovaného s kvantitatívnym pravidlom. Pri tomto prepise jedno kvantitatívne pravidlo môže by nahradené viacerými pravidlami.

Pre ilustráciu uvedených postupov uvažujme prípad množiny znalostí v tvare Hornových klauzulí. Nech táto znalostná báza pozostáva zo štyroch pravidiel podľa tab. 6.1.

| | | |
|---|---|---|
| A | :- | B, C. |
| B | :- | not D, E. |
| B | :- | not F, G. |
| C | :- | G, H, I. |

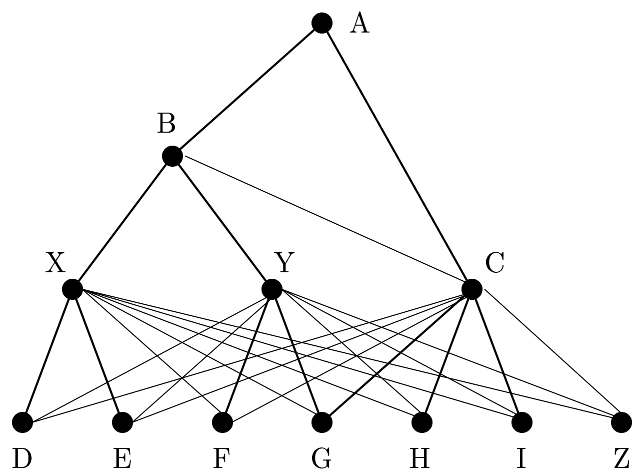
Tabuľka 6.1: Ilustračné príklady

Predpoklady týchto pravidiel sú tvorené konjunkciou tvrdení v priamom alebo negovanom tvare (disjunkcia je reprezentovaná viacerými pravidlami s tým istým záverom. Podmienkou je, aby pravidlá nevytvárali cyklické štruktúry.

Pravidlá vyjadrujú disjunkciu pomocou dvoch pravidiel, z ktorých obe majú predpoklad v tvare konjunkcie. Preto je nutné tieto pravidlá nahradiť dvojicami

| | | | | | |
|---|---|---|---|---|---|
| B | :- | X. | B | :- | Y. |
| X | :- | not D, E. | Y | :- | not F, G. |

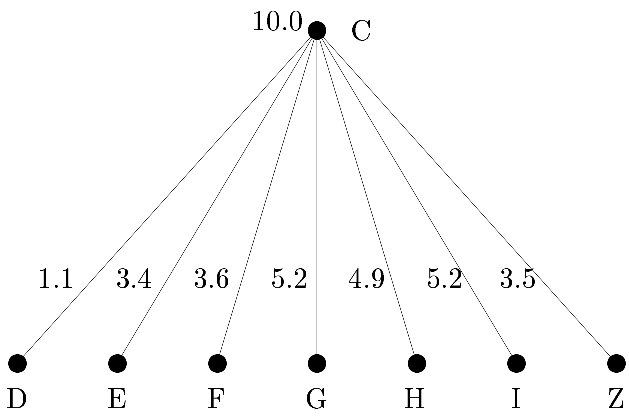
Ak predpokladáme, že existuje ešte jeden faktor $Z$, ktorý môže mať vplyv na rozhodnutie o platnosti $A$, je výsledná topológia siete podľa obr. 6.1 kde hrubšie hrany vyjadrujú pôvodnú množinu pravidiel a tenšie zase dodatočne pridané pravidlá.




Obr. 6.1: Topológia neurónovej siete

Hrany z $D$ do $X$ a z $F$ do $Y$ majú váhu $-\omega \pm r$, ostatné hrubo znázornené hrany majú váhu $\omega \pm r$, a všetky ostatné hrany majú váhu $r$, kde $\pm r$ je malé náhodné číslo. Prah uzlov $X$, $Y$, a $B$ je $\omega/2$, prah uzla $A$ je $3\omega/2$ a uzol $C$ má prah $5\omega/2$.

Predpokladajme že po natrénovaní siete z obr. 6.1 časť siete relevantná pre uzol C má tvar podľa obr. 6.2.




Obr. 6.2: Časť natrénovanej siete

V sieti vznikli tri zhluky hrán {D-C}, {E-C, F-C, Z-C} a {G-C, H-C, I-C}. Váhy hrán v týchto zhlukoch budú spriemernené na hodnoty 1.1, 3.5 a 5.1. Keďže algoritmická analýza ukazuje, že zhluk {D-C} je zbytočný, nebude v ďalšom uvažovaný.

Ak nové trénovanie siete (bez hrany medzi neurónmi D a C) nastavilo prah uzla C na hodnotu 10.1, tak prepis fragmentu siete pre neurón C bude mať tvar pravidla

$$\text{AK} \quad 10.1 < 5.1\, true(G,H,I) + 3.5\, true(E,F,Z) \quad \text{POTOM uzol\_C}$$

kde *true* reprezentuje funkciu, udávajúcu koľko z jej argumentov je splnených (platných).

Toto pravidlo môže byť nahradené pravidlami

| | | |
|---|---|---|
| AK 2 $z$ $(G,H,I)$ | | POTOM uzol_C |
| AK 1 $z$ $(G,H,I)$ | A 2 $z$ $(E,F,Z)$ | POTOM uzol_C |
| AK 3 $z$ $(G,H,I)$ | | POTOM uzol_C |

zahŕňajúcimi pôvodné pravidlo týkajúce sa platnosti tvrdenia reprezentovaného uzlom C.

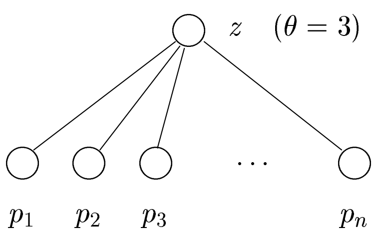
## 6.2 Zmena reprezentácie znalostí

Transformácia znalostí zo zdrojovej reprezentácie do cieľovej podoby môže v závislosti od cieľa transformácie a použitej metódy zachovávať obsah znalostí alebo ho môže meniť. Pri transformácii, pri ktorej nedochádza k zmene obsahu znalostí ale iba k modifikácii formy ich vyjadrenia, nie je potrebné zabezpečovať získavanie nových znalostí.

Príkladom takejto transformácie je práca s nemerateľnými hodnotami. Táto nemerateľnosť môže byť v danej situácii jednoducho neuskutočniteľná alebo spojená s veľkými nákladmi a preto je vhodné pokúsiť sa zaobísť bez nej. V takomto prípade nie je možné využiť všetky znalosti v báze znalostí, ale je potrebné upraviť znalosti tak, aby riešenie bolo možné získať aj pri zohľadnení nedostupnosti niektorých faktov (samozrejme jedným z možných riešení potom môže byť, že bez niektorých údajov nie je možné získať riešenie).

Uvažujme zdrojovú reprezentáciu znalostí pomocou inferenčnej siete pozostávajúcej z prahových uzlov. Riešiť úlohu znamená z hľadiska reprezentácie znalostí pomocou inferenčnej siete zistiť hodnoty listových uzlov a propagovaním týchto hodnôt inferenčnou sieťou odvodiť hodnoty uzlov koreňových. Tieto hodnoty môžu byť napr. informácie o platnosti alebo neplatnosti nejakých faktov. V prípade nemerateľnosti niektorých údajov však toto nie je možné či už úplne alebo z časti bez úpravy znalostí (a prípadne aj inferenčného mechanizmu).

Pre ilustráciu majme fragment znalostí pozostávajúci z jedného prahového uzla podľa obr. 6.3.




Obr. 6.3: Reprezentácia znalostí prahovým uzlom

Hodnota prahu $\theta = 3$ znamená, že záver $z$ bude splnený ak budú splnené aspoň tri z jeho predpokladov. Extrémnymi hodnotami prahu $\theta$ (1 a počet predpokladov) je možné modelovať disjunkciu a konjunkciu predpokladov.

Nech sa ukáže, že predpoklad $p_2$ je nemerateľný a teda nie je možné rozhodnúť o jeho platnosti alebo neplatnosti. Je vhodné upraviť daný fragment znalostí tak, aby sa zohľadnila nemerateľnosť tohto predpokladu. Dva extrémne spôsoby sú:

**pesimistický** — predpokladá, že predpoklad $p_2$ by neplatil ani keby bol merateľný. Teda záver musí byť splnený pomocou iných troch merateľných predpokladov. Predpoklad $p_2$ neprispieva k jeho splneniu.

**optimistický** — nemerateľný predpoklad sa považuje za platný. Pretože teraz $p_2$ prispieva k platnosti záveru, stačí nájsť dva splnené merateľné predpoklady.

Niekde v strede medzi týmito dvoma extrémnymi prístupmi leží tretí (uvažujúci možnosť splnenia aj nesplnenia nemerateľného predpokladu). Ak o splnení alebo nesplnení záveru je možné rozhodnúť bez ohľadu na nemerateľný uzol, tak sa tento nemerateľný predpoklad neuvažuje. Ak však neplatnosť tohto predpokladu vedie k neplatnosti záveru zatiaľ čo jeho platnosť implikuje platnosť záveru, tak táto šanca splnenia záveru je vyjadrená ako tretia logická hodnota MOŽNO. Konkretizácia tejto možnosti (ak to používateľ požaduje) je možná iba pomocou hodnoty nemerateľného predpokladu.

V zmysle tohto kompromisného prístupu je možné fragment znalostí podľa obr. 6.3 nahradiť dvoma pravidlami – pesimistickým a optimistickým. Pesimistické pravidlo má tvar

$$\text{AK } \theta \text{ z } (p_1, p_3, \ldots, p_n) \text{ POTOM } z \text{ (ÁNO)}$$

pričom nezohľadňuje tie predpoklady, ktoré majú hodnotu MOŽNO. Ak pesimistické pravidlo potvrdí záver $z$, tak ten je splnený bez ohľadu na platnosť nemerateľných predpokladov.

Optimistické pravidlo sa využije iba ak jeho pesimistický protějšok nebol úspešný. Má tvar

$$\text{AK } \theta \text{ z } (p_1, \text{MOŽNO}, p_3, \ldots, p_n) \text{ POTOM } z \text{ (MOŽNO)}$$

pričom zohľadňuje všetky predpoklady s hodnotou MOŽNO (či už sú nemerateľné alebo získali túto podmienenú platnosť vďaka nemerateľnosti svojich predpokladov). Jeho splnenie dáva záveru $z$ šancu na platnosť, ale bez jej potvrdenia.

Iným príkladom transformácie nemeniacej obsah znalostí je prevod z reprezentácie využívajúcej kombinovanie disjunkcie a konjunkcie (vychádzajúcej z existencie jednotlivých tvrdení) na reprezentáciu používajúcu v predošlom uvedené prahové vyjadrenie (uvažujúce nie samostatné tvrdenia ale triedy tvrdení, pričom členy tej istej triedy sú významovo ekvivalentné).

# Literatúra


*1. Získavanie znalostí a tvorba znalostných báz*

GUIDA, G. - TASSO, C.: Design and development of knowledge-based system. John Wiley & Sons, Chichester, 1994.

HATALA, M. - PARALIČ, J. - MACH, M. - SABOL, T.: Intelligent development of large-scale knowledge based systems. In: Proc. of the Int. Conference on Intelligent Technologies, Elfa, Herľany, 1996. 21-28.

MACH, M.: Projektovanie znalostných systémov. ES TU, Košice, 1992.

NEWELL, A.: The knowledge level. Artificial Intelligence, vol. 18, 1982, no. 1. 157-224.

OSUGA, S. - SAEKI, J.: Priobretenie znanij. Mir, Moskva, 1990. 179-137.

PAYNE, E.C. - McARTHUR, R.C.: Developing expert systems. John Wiley & Sons, New York, 1990.

STEFIK, M.: Introduction to knowledge systems. Morgan Kaufmann, San Francisco, 1995.

VODÁČEK, L. - ROSICKÝ, A.: Informační management. Pojetí, poslání a aplikace. Management Press, Praha, 1997.

*2. Priame získavanie znalostí od expertov*

BERG-CROSS, G. - PRICE, M.E.: Acquiring and managing knowledge using a conceptual structures approach: introduction and framework. IEEE Trans. on Systems, Man, and Cybernetics, vol. 19, 1989, no. 3. 513-527.

COOKE, N.J.: Varieties of knowledge elicitation techniques. Int. J. Human-Computer Studies, vol. 41, 1994, no. 6. 801-849.

DAVIES, M. - HAKIEL, S.: Knowledge harvesting: a practical guide to interviewing. Expert Systems, vol. 5, 1988, no. 1. 42-50.

FORSYTH, D.E. - BUCHANAN, B.G.: Knowledge acquisition for expert systems: some pitfalls and suggestions. IEEE Trans. on Systems, Man, and Cybernetics, vol. 19, 1989, no. 3. 435-442.

GREENWELL, M.: Knowledge engineering for expert systems. John Wiley & Sons, New York, 1988, 44-63, 92-117.

JOHNSON, L. - JOHNSON N.E.: Knowledge elicitation involving teachback interviewing. In: Knowledge acquisition for expert systems. KIDD, A.L. (eds). Plenum Press, New York, 1987. 91-108.

MEHDI OWRANG, M. - MEDSKER, L.R.: Analysis of schemes for developing rule-based expert systems. In: Expert Systems Applications EXPERSYS 90, IITT Int., Gournay-Sur-Marne, 1990. 287-292.

OLSON, J.R. - RUETER, H.H.: Extracting expertise from expert: methods for knowledge acquisition. Expert Systems, vol. 4, 1987, no. 3. 152-168.

*3. Nepriame získavanie znalostí od expertov*

BOOSE, J.H.: A knowledge acquisition program for expert systems based on personal construct psychology. Int. J. Man-Machine Studies, vol. 23, 1985. 495-525.

BOOSE, J.H.: Expertise transfer for expert system. McGraw-Hill, New York, 1986.

BOOSE, J.H. - BRADSHAW, J.M.: Expertise transfer and complex problems: using AQUINAS as a knowledge acquisition workbench for knowledge-based systems. Int. J. Man-Machine Studies, vol. 26, 1987. 3-28.

GAINES, B.R. - SHAW, M.L.G.: Knowledge acquisition tools based on personal construct psychology. The Knowledge Engineering Review, vol. 8, 1993, no. 1. 49-85.

KITTO, C.M. - BOOSE, J.H.: Heuristics for expertise transfer: an implementation of a dialog manager for knowledge acquisition. Int. J. Man-Machine Studies, vol. 26, 1987. 183-202.

SHAW, M.L.G. - GAINES, B.R.: An interactive knowledge elicitation technique using personal construct technology. In: Knowledge acquisition for expert systems. KIDD, A.L. (eds). Plenum Press, New York, 1987. 109-136.

*4. Induktívne vytváranie bázy znalostí*

ARBAB, B. - MICHIE, D.: Generating rules from examples. In: Proc. of the 9th Int. Joint Conference on Artificial Intelligence, Morgan Kaufmann, Los Angeles, 1985. 631-636.

GRAY, N.A.B.: Capturing knowledge through top-down induction of decision trees. IEEE Expert, vol. 5, 1990, no. 3. 41-50.

GROSS, D.: Induction and ID/3: more powerful than we think. Expert Systems, vol. 5, 1988, no. 4. 348-350.

HART, A.: Role of induction in knowledge elicitation. In: Knowledge acquisition for expert systems. KIDD, A.L. (eds). Plenum Press, New York, 1987. 165-190.

IVÁNEK, J. - FERJENČÍK, J. - STEJSKAL, B. - ŠVENDA, J.: Základy matematické informatiky III. Znalostní inženýrství. VŠE Praha, 1987.

JACKSON, A.H.: Machine learning. Expert Systems, vol. 5, 1988, no. 2. 132-150.

MACH, M. - ANDRÁSSYOVÁ, E.: Genetic algorithms in knowledge base design. In: Proc of the 1st Int. Conference on Genetic Algorithms MENDEL'95, PC-DIR, Brno, 1995. 77-82.

SHAW, M.J. - GENTRY, J.A.: Inductive learning for risk classification. IEEE Expert, vol. 5, 1990, no. 1. 47-53.

WEHENKEL, L. - PAVELLA, M.: Decision trees and transient stability of electric power systems. Automatica, vol. 27, 1991, no. 1. 115-134.

ZDRÁHAL, Z. - BRATKO, I. - SHAPIRO, A.: Recognition of complex patterns using cellular arrays. Computer Journal, vol. 24, 1981, no. 3. 263-270.

*5. Priamy prenos znalostí expertov do znalostných báz*

DAVIS, R. - LENAT, D.B.: Knowledge-based systems in artificial intelligence. McGraw Hill, New York, 1982. 227-490.

MACH, M.: A generic knowledge based problem solving method for parameter design. In: Proc. of the 12th Int. Conference ASRTP'96, BERG TU, Košice, 1996. 455-460.

POPOV, E.V.: Ekspertnye sistemy. Nauka, Moskva, 1987. 212-274.

*6. Vylepšovanie znalostí*

FOX, J. at all: A systematic study of knowledge base refinement in the diagnosis of leukemia. In: Knowledge acquisition for expert systems. KIDD, A.L. (eds). Plenum Press, New York, 1987. 73-90.

HWANG, G.J. - TSENG, S.S.: Eliciting embedded meanings in building expert systems under uncertainty. In: Proc. on Computers and Communications, IEEE Computer Soc., Scottsdale, 1990. 815-821.

LACHER, R.C. - HRUSKA, S.I. - KUNCICKY, D.C.: Back-propagation learning in expert networks. IEEE Trans. on Neural Networks, vol. 3, 1992, no. 1. 62-72.

QUINLAIN, J.R.: Simplifying decision trees. Int. J. Man-Machine Studies, vol. 27, 1987. 221-234.

*7. Transformácia znalostí*

MACHOVÁ, K.: Modifikácia vedomostí pomocou učenia založeného na vysvetľovaní [kandidátska dizertačná práca]. Technická Univerzita, Fakulta elektrotechniky a informatiky, Košice, 1995.

TOWELL, G.G. - SHAVLIK, J.W.: Extracting refined rules from knowledge based neural networks. Machine Learning, vol. 13, 1993. 71-101.

TOWELL, G.G. - SHAVLIK, J.W.: Knowledge-based artificial neural networks. Artificial Intelligence, vol. 70, 1994, no. 1-2. 119-165.

# Register