Teadmussüsteemid - kursus 2000.a. kevadel

Jaak Tepandi
TTÜ Informaatikainstituut

On palju ülesandeid, mida ulatuslikult ja edukalt arvutitel lahendatakse, näiteks rutiinsed arvutused või andmetöötlus. Samal ajal on aga ka piisavalt neid, millega arvutid praegu eriti üldse hakkama ei saa (kuigi inimene neid mõnikord efektiivselt ja pidevalt lahendab, näiteks liiklusolukorra visuaalne analüüs ja selles tegutsemine). Nende kahe äärmuse vahele jääb "hall ala" ülesannetest (näiteks mitmesugused ekspertteadmistel põhinevad otsustusülesanded), mis on halvasti algoritmeeritavad ja ei allu hästi rutiinsele arendustegevusele, kuid on siiski juba praegu arvutitele võimetekohased. Mõnede sedasorti ülesannete lahendamise meetodid on käesoleva väljaande põhiline teema.

Märkus: mõiste "arvutid" ülal tähistas igasuguseid tehislikke vahendeid, millega vaadeldud meetodeid realiseerida saab, sealhulgas infotehnoloogilisi vahendeid: riistvara, tarkvara, kommunikatsioone, keeli, meetodeid jne. Käsitletud meetodeid võib muuta vahenditest sõltumatuks.

Käesolev väljaanne on lühikonspekt TTÜ-s õpetatavale ekspertsüsteemide kursusele, siintoodud mõisted avatakse kursuse käigus pikemalt. Kursuse eesmärk on anda sissejuhatus probleemidesse ja meetoditesse, mis ei mahu tänapäeva tüüpilise infotöötluse raamidesse. Kursuse sihid on:

- anda ülevaade teadmussüsteemide meetoditest,
- tutvuda lähemalt reeglipõhiste süsteemidega,
- saada praktiline kogemus lihtsa ekspertteadmisi kasutava süsteemi realiseerimisel.

Esimeses osas antakse ülevaade kogu teemast. Kursuses on selle osa ülesanne anda algteadmised iseseisvate tööde teemade valikuks ja alustamiseks. Järgmised osad täpsustavad kasutatavaid mõisteid (teadmine, teadmiste kujutamine) ning annavad sissejuhatuse olulisematesse meetoditesse (järeldamine, õppimine, otsing).

1. Ülevaade

Jaotis annab eelteadmised teadmussüsteemidest, sobiva ülesande valikust ja realiseerimise võimalustest.

1.1. Teadmussüsteemi näide: börsi järelevalve

Näitena süsteemi kohta, kus kasutatakse halvasti algoritmeeritavaid ekspertteadmisi, toome börsil The American Stock Exchange kasutatava börsi järelevalve süsteemi MESS. Süsteemi ülesanne on ära tunda siseinfo kasutamise juhtumeid börsil. See on oluline nii investoritele, kes on huvitatud ausast ärist, kui ka börsile endale, kes soovib olla usaldusväärne klientide silmis. Järgnevad süsteemi omadused on teadmussüsteemidele üsna iseloomulikud:

- süsteemil on mitu rolli: otsustamise toetus (sorteerib väga suurest hulgast juhtumitest välja potentsiaalselt kahtlased), juhtumite arutamise stimuleerimine (ka juhul, kui järeldustega ei nõustuta), juhtide treening (juhtumite kirjeldamine ja analüüs),
- otsustusteadmised valdkonnast muutuvad pidevalt, neid tuleb korrigeerida,
- süsteem opereerib ebakindlate teadmistega (tulemus ei ole kindel järeldus, vaid pigem tõeväärtusega hinnatud oletus, mille põhjal jälgiv osakond teeb otsuse),
- tulemuste töötlus põhineb mitme vahendi kombinatsioonil. Nendeks on suurarvuti andmebaasisüsteem (andmete esmane valik), tabelarvutusprogramm (esialgsete kokkuvõtete tegemine), algoritmiline keel (sisendandmete töötlus) ja ekspertsüsteem (otsuste tegemine).

1.2. Mõiste ja iseärasused

Teadmussüsteeme on väga mitmeti iseloomustatud. Tihti korduvad neis iseloomustustes järgmised omadused:

- teadmiste kirjeldamine, uuendamine, kasutamine,
- heuristiliste (kogemuslikkude) teadmiste kasutamine - üldse, erinevate teadmiste liikide kasutamine, eriti sümbolkujul,
- läbipaistev/selgitab oma järeldamist,
- eksperdi tasemel/eksperdi teadmisi nõudev,
- õpib kogemustest, areneb,
- oluline/perspektiivne ülesanne.

Et vahet teha eri liiki süsteemide vahel, võib võrrelda järgmisi ülesandeid ja probleeme. Osutub, et näiliselt sarnase ülesande püstitusega probleemide lahendusmeetodid võivad olla väga erinevad.

- Tunda ära olukord ristmikul
- Tunda ära õige kood
- Leida pangalaenude üldsumma
- Kellele anda pangalaenu?
- Tõlge/vestlus
- Menüüdega dialoog/teksti redigeerimine
- Leida kõik keskharidusega töötajad
- Valida sobiv töötaja
- Diagnoosida mäluseadmeid
- Diagnoosida haigust

Toodud võrdlus aitab välja tuua järgmised teadmussüsteemide probleemide iseärasused.

- Halvasti struktureeritud probleemala, otsest algoritmi pole
- Tihti sümbolandmed
- Andmed ebakindlad/vigased
- Erinevad teadmiste allikad, vasturääkivad tulemused
- Raske hinnata tulemuste õigsust
- Opereerib mitmesugust liiki teadmistega
- Arvuti lahendab (seni?) halvemini kui inimene (“intelligentsust nõudvad”)
- Palju seotud/suhtlevaid mittehomogeenseid struktuure
- Kiiresti muutuvad teadmised/soovid/arusaamad

Toome veel näiteid “intelligentsete” süsteemide rakendustest.

- Arvutite konfigureerimine
- Kiipide kavandamine/tootmine
- Telefoniseadmete/liinide diagnostika
- Õpetamissüsteemid
- Maavarade prognoos
- Reostuse prognoos/kõrvaldamise abinõud
- Riskianalüüs

1.3. Teadmussüsteemide ideed/meetodid

Kuidas realiseerida ülalmainitud omadustega süsteeme? Võimalusi on väga palju, mõned end rohkem õigustanud ideed on toodud allpool.

- Nõrgad meetodid - ei üritata leida parimat lahendust, piisab esimesest sobivast.
- Heuristiline otsing - otsingu juhtimiseks kasutatakse kogemuslikke teadmisi.
- Lahuta tuletusvahendid ja teadmusbaas.
- Sümboltöötlus.
- Mustrite sobitamine (pattern matching).
- Kindlusinfo töötlemine.
- Koostöötavad eksperdid - meetodid arvamuste sobitamiseks ja konsensuse leidmiseks.
- Arutlused mitmel abstraktsioonitasemel.
- On mitmesuguseid teadmussüsteemide liike, näiteks reeglipõhised, freimipõhised ja muud.

Üks loomulikumaid, lihtsamaid ja seetõttu ka laiemalt käsitletud teadmiste kujutamise viise on tuletusreeglid (production rules). Et illustreerida seda formalismi, toome lihtsustatud näite reeglibaasist (realistlikud reeglibaasid sisaldavad sadu ja tuhandeid reegleid).

R1: KUI ei reosta keskkonda
JA hea tasuvus
SIIS rahasta projekt
MUIDU keeldu

R2: KUI uus tehnoloogia
JA investeering tasub < kahe aastaga (prognoos)
JA käivitamine < aasta
SIIS hea tasuvus

R3: KUI säästab raha, tööd jne
JA investeering tasub < kolme aastaga (prognoos)
SIIS hea tasuvus

Reeglipõhiseid süsteeme saab otsustamiseks läbida sihipõhiselt või andmepõhiselt, on ka muid meetodeid. Esimesel juhul alustatakse sihtidest - antud juhul näiteks "rahasta projekt" - ja liigutakse ettepoole eeldusteni, mida nende sihtide kehtivuseks on vaja rahuldada. Teisel juhul antakse ette teatavad andmed - näiteks väites, et "ei reosta keskkonda" on tõene - ja järeldatakse nendest kõik, mis võimalik. Pikemalt on järeldamisest räägitud allpool.

1.4. Arendusest

Milliseid ülesandeid on otstarbekas lahendada teadmussüsteemide vahendite ja meetoditega? Valik sobiva probleemi iseloomustusi on toodud allpool. Mõned neist on rohkem, mõned vähem olulised/kategoorilised väited.

- Kindlat algoritmi pole.
- Ekspertteadmised on vajalikud.
- Andmed mittehomogeensed, ebakindlad, sümboltöötlus.
- Teised süsteemid ei sobi.
- Realiseeritav/tasuv.
- Kitsas ala.
- Ülesande lahendamine eksperdil võtab üle poole tunni, alla poole päeva.

- Ekspert olemas, eksperthinnangud ühtivad, andmed kättesaadavad, vaheetapid eristatavad, korduv, rutiinne.

Süsteemiarenduses võib kasutada kõiki tavalisi vahendeid. Tihti töötatakse prototüüpimise meetodil. Spetsifitseerimiseks võib kasutada infosüsteemide arendusest tuttavaid formalisme ja vahendeid (andmevoogude diagramm, olem-suhte diagramm, CASE-vahendid jne.), aga ka näiteks järgmisi:

- spetsifikatsiooni standard (nt. vastav ANSI/IEEE standard),
- näiteülesanded/stsenaariumid,
- prototüüp,
- mõistete nimekiri, kontekstidiagramm,
- seoste graaf, sealhulgas seosed: on (is-a), on-osa (is-part-of), põhjustab (if-then).

Süsteeme võib realiseerida, kasutades eriotstarbelisi arendusvahendeid (agentide arenduskeskkond, ekspertsüsteemi kest - nt. Exsys; mitmeid sedasorti süsteeme on võimalik kätte saada internetist) või muid süsteemiarenduse/programmeerimise keskkondi (programmeerimiskeeled, tabelarvutuse makrod, andmebaasisüsteemid jne.). Viimasel juhul realiseeritakse teadmussüsteemide meetodid valitud vahendi abil.

Realiseerimisel on oluline kasutada tõepoolest ekspertteadmisi ("kuidas on antud olukorras kõige parem toimida?"; tavaliselt raamatutes antud teadmised on teist liiki) ja arvestada tulevase kasutajaga (näiteks, kas kasutaja saab üldse süsteemi poolt esitatavatest küsimustest aru?). Testimisel oluline kindlaks määrata vajalik töökindluse tase (muidu on väga raske hinnata testimise piisavust) ja esitada tulevase kasutaja laadis küsimusi (ka ekspertsüsteemide puhul kehtib see, et arendaja kipub testima arendusloogikas ja ei leia vigu).

1.5. Ülevaade teadmussüsteemidest: kordamisküsimusi ja ülesandeid

Kordamisküsimusi

- Vajadused ja motiivid teadmussüsteemide (TS) tekkeks ja arenguks
- Teadmussüsteemi probleemide iseärasusi
- Sobivad probleemid
- TS mõiste
- TS iseärasusi
- Näiteid rakendustest
- TS ideid
- Reeglipõhised süsteemid: sihi- ja andmepõhine järeldamine
- Vahendeid TS spetsifitseerimiseks
- Realiseerimisvahendid

Ülesandeid

- Antud ülesanne. Kas see sobib realiseerimiseks TS-na? Miks?

2. Andmed, info, teadmine, ekspertsus

Jaotise siht on osata eristada ja suhtestada pealkirjas nimetatud mõisteid.

2.1. Võimalik tõlgendus: andmed, info, teadmine

Andmete, info ja teadmise mõisted näivad erinevad, aga seda erinevust sõnastada polegi nii lihtne. Pakume ühe võimaluse teadmise määratlemiseks ja ühe eristuse andmete, info ja teadmise jaoks.

Teadmine on sihtide kontekstis korrastatud (esitatud) suhe kolme ilmingu vahel:

- objekt (asi, mille kohta teatakse),
- seda esindav (sellega seotud) märk,
- teadja (agent, subjekt, kes teab; samuti edaspidi järeldamise masin).

Seda tõlgendust saab paremini tajuda kui vaadelda olukordi, kus mingi neist kolmest puudub - teadmine ilma teadjata, objektita või märgita. Samuti, mis juhtub siis, kui keegi neist, nt. agent, muutub, aga teised jäävad samaks?

Mainitud mõisteid võib eristada nii:

- andmed iseloomustavad objekti mingeid atribuute üldiselt aktsepteeritud skaalal (pikkus, kaal, hulk jne),
- info tekib omavahel seostatud andmete põhjal; seda saab mingil eesmärgil kasutada,
- teadmine on seosed andmete vahel, mis on vajalikud otsuse tegemiseks; samuti üldistatud info või kogemus, mille põhjal saab efektiivselt tegutseda,
- toodud mõisted ei ole absoluutsed: mis ühes kontekstis on info, võib teises olla andmed, teadmine jne.

2.2. Kogemuslik teadmine

Inimteadmisi võib klassifitseerida tehnilisteks (kehtivad “kindlasti” - neid saab esitada näiteks valemite või algokeele programmidega), mitteformaliseeritavateks (intuitsioon, üldised teadmised) ja kogemuslikeks (heuristilisteks - kehtivad enamasti, kuid mitte alati).

Veel üks liigitus on antud järgnevas tabelis - erialade näited kahe dimensiooni (võimete ja vabaduse) lõikes.

	Kognitiivsed võimed	Otsustusvõimed	Sotsiaalsed võimed
Tugev loomingulisus	Muusik	Tippjuht	Kirjanik
Analüütilisus	Matemaatik	Finantsist	Sotsiaalteadlane
Valdav protseduursus	Masinakirjutaja	Autojuht	Sotsiaaltöötaja

Lõpuks võib rääkida sellistest teadmiste alaliikidest nagu faktidest, arvutuslikest seostest, protseduuridest, mõistetest, heuristikatest, kujunditest jne.

Toodud liigituste üle võib vaielda, kuid nad annavad mingi lähtepunkti ja aitavad piiritleda käesoleva kursuse rakendusvaldkonda.

2.3. Teadmised ja süsteemid

Alljärgnevas tabelis tähendab “++” ulatuslikku kasutamist, “+” - vähemat või staatilist kasutamist, “?” - enamasti mittekasutamist.

	Programmid	Andmebaasid	Teadmussüsteemid
Faktid/andmed	+	++	+
Arvutuslikud seosed	++	+	+
Protseduurid	++	+	+
Mõisted	+ (andmedef)	+ (AB kirjeldus)	++
Heuristilised teadmised	?	?	++
Kujundid, lõhnad,…	?	?	?

2.4. Andmed, info, teadmine, ekspertsus: kordamisküsimusi ja ülesandeid

Kordamisküsimusi

- Andmed/info/teadmine
- Teadmise liigitusi (min. 2). Kogemuslikud teadmised.
- Programmides, andmebaasides, ES-des kasutatavaid teadmisi

Ülesandeid

- Eristada teadmise komponente mingites konkreetsetes teadmistes
- Antud probleem, mis liiki teadmisi on vaja?
- Antud probleem, mis liiki vahenditega lahendada?

3. Teadmiste esitamine ja järeldamine

Jaotises antakse ülevaade teadmussüsteemides enamkasutatavatest teadmiste esitamise ja järeldamise formalismidest ja meetoditest. Oluline on edaspidi vahet teha keele (märkide) ja keelt töötleva aparaadi (järeldamise masina) vahel, samuti mõelda juurde nende kahe suhe objektiga.

3.1. Teadmiste esitamise keeli

Teadmist võib esitada interpreteeritud märgi(jada)de abil. Need võivad olla antud mitmeti:

- faktidena
- protseduuridena
- seostena: arvutuslikud, heuristilised,…
- mõistetena (ja nende sisestruktuuri abil),
- ja nii edasi.

Teadmiste esitamise keeli ja formalisme:

- loomulik keel, semantilised võrgud
- loogikad
- reeglid, freimid, …
- AB kirjeldused, algoritmilised keeled,
- otsustuspuud, graafid, …
- närvivõrgud,
- …

Sellisid keeli võib liigitada deklaratiivseteks ja protseduurseteks. Deklaratiivsete keelte puhul (Prolog) kirjeldatakse olukord ja küsitakse lahendust (lahenduskäik tuleb ise leida). Protseduursetes keeltes (algoritmilised keeled) esitatakse lahenduskäik koos küsimusega. Esimesel juhul tuleb teha rohkem tööd vastamaks antud küsimusele, seevastu saab vastata paljudele küsimustele. Teisel juhul on vastuse leidmine lihtsam, kuid vastata saab ainult seatud küsimusele.

3.2. Reeglid

Ideid:

- inimtegevuses kasutatakse tihti seda tüüpi teadmisi
- kogemuslikud teadmised, järeldamine, selgitused
- suhteliselt sõltumatu reeglite lisamine/muutmine/kustutamine

- suhteliselt lihtne lisada ebatäpsete andmete töötlust

Lihtsamal juhul:

- kasutatakse tingimuslike seoste kirjeldamiseks
- esitatakse kujul: Kui (tingimus) Siis (tegevus) {Muidu (tegevus}
- järeldamine: sihipõhine ja andmepõhine
- eriteemad: selgitused, kaalud, järeldamise juhtimine

Üldisem vorm - produktsiooni(reegli)d

Sihipõhise järeldamise algoritmi skeem (skeem, kuna mitmed nüansid on ainult markeeritud)

Procedure Siht-andmed (G) ❶

S := reeglid, mis määravad G

if S = Ø

then küsi G

else

while G teadmata❹ **and** S =/ Ø

vali reegel R hulgast S ❷

G' := R eeldus ❸

if G' teadmata **then** rakenda Siht-andmed hulga G' elementidele

if G' tõene **then** rakenda R

end while

end if

end Siht-andmed

Sihipõhise järeldamise juhtimine:

- ❶ sihi valik (sihtide esinemise järjekord, kasutaja antud järjekord, tõenäosemad enne, vms)
- ❷reegli valik (reeglite esinemise järjekord, kasutaja antud järjekord, tõenäosemad enne, vms)
- ❸reegli eelduse valik (eelduste esinemise järjekord, kasutaja antud järjekord, tõenäosemad enne, vms)
- ❹järelduse põhjalikkus (kas piirdutakse esimese politiivse järeldusega või uuritakse edasi)

Andmepõhise järeldamise algoritmi skeem (skeem, kuna mitmed nüansid on ainult markeeritud)

Procedure Andmed-siht

Küsi andmeid

S := rakendatavad reeglid

while S =/ Ø **and** probleem lahendamata

vali reegel R hulgast S

Rakenda R, uuenda andmed

S := rakendatavad reeglid

end while

end Andmed-siht

Kuidas saab juhtida seda järeldamist?:

Selgitused

- Miks{küsitakse seda}? - anda reeglid, mis tingivad küsimuse
- Kuidas {jõuti vastuseni}? - anda faktid ja reeglid, mis viisid vastuseni
- Mis siis kui {anname teised vastused}?
- Reegel?
- Abi

Reeglipõhised süsteemid ja andmebaasid

Mõlemal on oma eelised ja kasutusalad. Andmebaas on sobivaim fikseerunud struktuuriga andmete puhul. Reeglipõhise süsteemi reegel võib katta suure arvu kirjeid, reeglid sobivad esitama järeldamise ahelaid ja ebakindlaid teadmisi.

Reeglid ja otsustuspuud

Puu on ülevaatlikum, seda kasutataksegi mõnes ES-s baasist ülevaate saamiseks, samuti võib seda kasutada/vaadata spetsifikatsioonina. Reeglibaasis on lihtsam kujutada suurt süsteemi, teha muudatusi, kajastada detaile. Puu võib luua dünaamiliselt baasi põhjal.

3.3. Freimid

Kui hakata mõtlema, mis tunnuste põhjal inimene eristab mingit mõistet (näiteks, kuidas ta teeb kindlaks, et mingi asi on “laud”, aga mitte midagi muud), siis selgub, et isegi lihtsate asjade puhul pole seda definitsiooni anda alati sugugi kerge. Freimi all mõisteti esmaselt konstruktsiooni, mis esitas mingi objekti olulisi, seda objekti piiritlevaid ja teistest eristavaid omadusi.

Edaspidi hakati freimidega kujutama objektide ja nende vaheliste seoste kirjeldusi sõltumatult sellest, kas need kirjeldused olid ammendavad või olemuslikud. Selles mõistes on freim teadmiste kirjeldamise vahend, mis võimaldab teha järeldusi, anda selgitusi, muuta teadmisi. Freimi mõiste on üheks eellaseks objekti mõistele OOP-s. OOP on läinud suurema detailsuse ja protseduursuse suunas (mis on aluseks edukale realiseerimisele), freim on säilitanud illustratiivsuse ja seosed, mis võimaldavad teha järeldusi.

Tihti antakse freimi määratlemisel selle nimi, tüüp ja atribuudid (“slotid”, omadused, seosed). Atribuutideks võivad olla: on-seos (is-a, side ülemliik-objektiga), ulatus (range, alamliigid), näited (objektid, mis kuuluvad antud freimi koosseisu), on-osa (is-part-of, side objektiga, mille osaks on antud freim), koosneb, kuidas-leida (protseduur leidmiseks), vaikimisi-väärtus, lisamisel-teha (lisamisel rakendatav protseduur) jne. Ühenimelisi seoseid saab organiseerida seda seost kajastavatesse hierarhiatesse või võrkudesse. Näiteid: klasside hierarhia objekt-orienteeritud keeles põhineb on-seosel; moodulite puu kajastab on-osa seost; reeglid võivad põhineda põhjuslikkuse seosel.

Protseduurideks võivad olla ka reeglibaasid. Järeldamisel püütakse jõuda etteantud sihini (hinnata selle tõesust või väärtust), hinnates freimi kõiki atribuute, rakendades pärimist is-a seose alusel, rakendades sihi- või andmepõhist järeldamist reeglibaasis, täites algoritmilises keeles kirjutatud protseduuri, hinnates mingi tingimuse täitmist vms. Mitmese pärimise puhul, nagu OOP puhulgi, võib tekkida konflikte, kui erinevatelt vanematelt päritakse sama atribuudi erinevaid väärtusi.

3.4. Semantilised võrgud

Algselt arvati, et loomuliku keele masintõlge on lihtne: tõlgi aga sõnad ära (jälgides käändelõppe jms.). Kui selgus, et sellest ei piisa, võeti kasutusele keele süntaks (eristades keele põhilisi objekte nagu alus/öeldis/... ja nende vahelisi seoseid). Osutus, et ka see on ebapiisav ja tuleb jälgida semantilisi/sisulisi seoseid mõistete vahel. Lõpuks, elus on küllalt olukordi, kus tõlge peab põhinema teadmistel konkreetsete tegeliku maailma objektide kohta.

Semantiline võrk kajastab tegevuste ja olukordade tüüpe, mida teades kirjeldatakse loomuliku keele lause sisemine struktuur. See ei kajasta enam niivõrd keele iseärasusi, kuivõrd maailma objekte ja nendevahelisi seoseid. Sellest struktuurist lähtudes saab vastata päringutele, tõlkida teise keelde, teha otsustusi. Tõlke puhul liigutakse siis lähtekeele leksika (sõnade/morfoloogia) tasemelt süntaksi tasemele ja sealt semantilisele või ka edasi konkreetsete objektide tasemele. Saadud nivoolt liigutakse juba sihtkeele konstruktsioonides sihtkeeles antud tekstini tagasi.

3.5. Närvivõrgud

Närvivõrgud on üks teadmiste kujutamise formalismidest. Infotehnoloogia sõnastik (EVS-ISO/IEC 2382-28:1998) määratleb seda järgmiselt: "Kaalutud sidemete kaudu ühendatud elementaarsetest töötluselementidest koosnev võrk, milles iga element tekitab mingi väärtuse, rakendades oma *sisend*väärtustele mingit mittelineaarset funktsiooni, ja edastab selle väärtuse teistele elementidele või esitades ta *väljundina."* Närvivõrgud modelleerivad mingil määral neuronite talitlust närvisüsteemis.

Närvivõrgu elementaarne töötluselement ehk neuron realiseerib siis mingit funktsiooni. Tihti kasutatav funktsioon on sisendväärtuste kaalutud summa võrdlemine läveväärtusega (üldisemalt, rakendatav mittelineaarne funktsioon on harilikult lävefunktsioon). Neuronid ühendatakse võrku, mis koosneb kahest või enamast kihist. Kahe kihi puhul tegemist sisend- ja väljundkihiga, nende vahel võivad olla varjatud kihid.

Närvivõrk on üks järgmisest viiest arvutusmudelist:

- Matemaatiline mudel (arvutatavus antud mat. operatsioonide abil)
- Turingi masin (operatsioonilis-loogikaline)
- Arvuti (mälu, operatsioonid, programmid)
- Rakuautomaadid (massiivne parallelism)
- Bioloogiline mudel (närvivõrgud)

Närvivõrkudel on palju rakendusi, näiteks:

- Aktsiahindade prognoosimine
- Lõhna/värvi tuvastamine
- Allveelaevade avastamine sonari abil
- Käekirja lugemine
- Vigase toodangu diagnoos
- Kohtuvälise lahendi ennustamine

Etteantud närvivõrku on võimalik paljudel juhtudel teisendada ekspertsüsteemiks või vastupidi. Närvivõrkudel ja ekspertsüsteemide võrdlust esitab järgmine tabel:

Närvivõrgud	Ekspertsüsteemid

• "must kast"
• teadmine on suurel määral kaaludes
• struktuur püsiv
• treenitav minimaalse abiga inimese poolt
• sobib:
  • pidevalt muutuva teadmise korral
  • põhjendust pole vaja
  • füüsikalised parameetrid
	• struktuur nähtav, kergem sisust aru saada
	• teadmine on suurel määral struktuuris
	• struktuur muutub
	• selgitab järeldusi
	• nõuab arendamiseks eksperdi abi
	• sobib:
	  • heuristiline teadmine
	  • vaja põhjendada otsuseid
	  • aeglaselt muutuv teadmine

3.6. Teadmiste esitamine ja järeldamine: kordamisküsimusi

Kordamisküsimusi

- Teadmiste esitamise formalisme
- Seos teadmiste esitamise ja järeldamise vahel
- Reeglid:

    - milleks?
    - üldkuju
    - sihi/andmepõhine järeldamine
    - selgitused
    - üldistused
    - järeldamise juhtimine
    - reegli- ja andmebaasid
    - reeglibaasid ja otsustuspuud
    - reeglibaasi spetsifikatsioon
- Freimid:
    - milleks (kaks sihti)
    - üldkuju(d)
    - pärimine
    - järeldamine: freimid, reeglid, protseduurid
    - seoste graafe
    - mitmene pärimine, selle probleeme
- Semantilised võrgud:
    - milleks?
    - süntaks ja semantika
    - semantilise võrgu esitus
    - tõlkimise skeem
- Närvivõrgud:
    - Viis arvutusmudelit
    - Neuron
    - Tüüpiline funktsioon
    - Närvivõrk, selle kihid
    - Rakendusi
    - Närvivõrgud ja ekspertsüsteemid

Ülesandeid

- Antud faktid ja reeglibaas. Andmepõhise järeldamise käik?
- Antud reeglibaas. Mida küsitakse esimesena sihi (andme)põhisel järeldamisel?
- Antud AB ja reegel - mitut kirjet asendab?
- Koostada antud teadmusbaasi seoste graaf
- Reeglite järjestust muudetakse. Milline küsimus esitatakse esimesena?
- Visandada algeline närvivõrk mingi ülesande jaoks

4. "Ebakindlad" teadmised

Tavaline püüdlus andmetöötluses on saada andmed kindlateks ja täpseteks. Enamasti on see õigustatud. Firma ei saa oma laosüsteemi kasutada, kui seal on vigased või puudulikud andmed - või saab siiski? Äkki on hoopis nii, et mingid tuumikandmed peavad olema täpsed, aga on osa andmeid, mille täpsus ei pruugi olla suur või mis võivad hoopis puududa? Lähemal vaatlusel selgubki, et enamik tänapäeva süsteeme lubab mingit ebatäpsust teatud andmetes. Teadmussüsteemides on see muudetud põhimõtteks ja pakutud hulk meetodeid, kuidas "elada koos" ebatäpsete, puudulike, vasturääkivate jne. andmetega.

Vaatleme ebasoovitavaid omadusi andmetes ja nendesse suhtumise viise, ebakindluste esitamise mooduseid ja nende omadusi (sobivus alaga, formalism, efektiivsus), soovitavaid omadusi kindlustegurite töötlusel ja nende omaduste esitust, järeldamist ebakindlate andmetega.

4.1. Ebatäpsed, ebakindlad jne

On mitut sorti omadusi, mida tavaliselt andmetes näha ei soovita:

- Ebatäpsus (maatüki pikkus on 50meetrit täpsusega +-2 meetrit)
- Ebakindlus (ma arvan, et see maksis 50)
- Hägused (see on vana tööriist)
- Vasturääkivad
- Puudulikud
- Liiased

Mitmete nende omaduste kombinatsioonid on sõltumatud ja mittevälistavad, nt. info võib olla ebatäpne ja samal ajal kindel.

Kui minna piisavalt detailseks, on pea iga info ebatäpne. Seepärast võib öelda, et täpsuse hinnang on absoluutse täpsuse ja nõudmiste kombinatsioon. Et parandada täpsuse hinnangut, võib suurendada tegelikku täpsust või vähendada nõudmisi. Ligikaudu sama kehtib mitmete teiste ülalmainitud omaduste puhul.

Iga mainitud omaduse puhul võib rääkida selle kujutamisest (kuidas mõõdame ja kirjeldame omadust) ja töötlusest (kuidas mitmete objektide koos esinemisel omadus levib). Edaspidi vaatame kolme ebakindluse kirjeldamise viisi, aga ainult ühe puhul - töötlust.

4.2. Ebakindluste kirjeldamise võimalusi

Et üldse rääkida ebakindlate andmete (nt., väidete) töötlusest, peame kõigepealt ebakindlust kirjeldama/mõõtma, alles peale seda saame seda töödelda. Vaatame siinkohal kolme küllalt levinud ebakindluse kirjeldamise viisi andmete jaoks.

Sündmuse (väite) *tõenäosus* asub vahemikus nullist üheni, mingi sündmuse ja selle vastandsündmuse tõenäosuste summa on 1. Tõenäosuste eeliseks on selge sisu, ühtne kujutamine, üsna täpse ja põhjaliku matemaatilise aparaadi olemasolu. Puudused: tõenäosus ei kajasta sisuliselt ebakindlust, sündmused on tavaliselt vastastikuses sõltuvuses - seepärast sündmuste tegelik hindamine ja töötlemine teadmussüsteemides ei vasta tihti tõenäosuse formaalsetele eeldustele (matemaatika on hea, aga ei vasta tihti vajadustele).

Väite *kaal* püüab olla ebakindlusmõõt. Tavaliselt eksisteerib üks minimaalne (kindlasti ei, näiteks 0 või -1) ja maksimaalne (kindlasti jah, näiteks 1 või 10 või 100) kaal, teised on nende vahel. Kaalud on tihti normeeritud (maks = 1). Variante: ühe väite ebakindlust iseloomustavad kaks mõõtu. Kaalusüsteeme on palju, üldtunnustatut ei ole, matemaatilised mudelid on vähe arenenud (vastab paremini vajadustele, aga formalismina vilets).

Häguste hulkade puhul on ebakindlusmõõduks mitte üks või kaks suurust, vaid terve hulk või seda iseloomustav funktsioon. Selline funktsioon kirjeldab kindalt mõistet, näiteks “pikk” või “külm”. On mitu tüüpilist ebakindlust iseloomustavat häguse hulga kuju, nt. S-kuju ja “kaabu”. Hägused hulgad vastavad üsna hästi teatavatele praktilistele ebakindlusolukordadele ja on ka hästi formaliseeritavad. Kuna aga mõõdud on keerulised, on nende töötlus teadmussüsteemides tihti raske (vastab vajadustele ja formaliseeritav, aga ebaefektiivne).

On loodud veel mitmeid “mitte päris korralike andmete” kirjeldamise formalisme, aga ka need kannatavad mõne puuduse all samuti nagu ülalmainitud.

Kindlushinnanguid saab anda ka mittenumbrilisel skaalal, lihtsaim nendest: jah/vist/ ei.

Senine praktika on näidanud, et ühte ideaalset formalismi pole, erinevatele valdkondadele sobivad erinevad formalismid.

4.3. Tehted kindlushinnangutega

Oleme oma andmed varustanud mingite kindlushinnangutega, nüüd tuleb neid andmeid kasutada. Kui andmed osalevad järeldusahelates, peame me selleks esmalt oskama teha oma kindlushinnangutega lihtsamaid tehteid. Vaatame kolme enim vajalikku tehet kaalude jaoks: ja (korrutamine, AND), või (liitmine, OR), järeldamine (INFERENCE).

Kuna kindlushinnangute tüüpe ja seega ka tehete konkreetseid kujusid on väga palju, püüame esmalt kindlaks teha, millised peavad olema tehete omadused. Olgu minimaalne kaal tähistatud a, maksimaalne - ü, vahepealsed - tähtedega f, g, h, ... Omadused tehetele lähtuvad rakendusest, näiteks igapäevase elu tavalistest reeglitest. Me võime soovida näiteks, et:

- kui kindlushinnang ei saa olla negatiivne (väljendab ainult positiivset arvamust) ja kui on kaks kindlushinnangut mingi väite kohta, siis summaarne hinnang poleks neist väiksem, ehk: OR(f,g)>=max(f,g)
- väidete koosesinemise hinnang ei väljuks lubatud vahemikust: a<=AND(f,g)<=ü
- kui esimesel juhul arvajad olid rohkem veendunud kui teisel, siis ka nende summaarne kindlus oleks esimesel juhul suurem/võrdne,
- kui üks arvaja ei tea midagi, siis summaarselt jääb kehtima teise arvaja hinnang (OR) ,
- kui kaks väidet peavad kehtima üheaegselt ja teine ei kehti, siis tulemus ei kehti (AND),
- kui üks arvaja on absoluutselt kindel ja teine mitte, siis on ka tulemus absoluutselt kindel (OR),
- kui kaks väidet peavad kehtima üheaegselt ja teine on absoluutselt kindel, siis tulemus on esimese arvamuse kindlusega (AND),
- tulemus ei sõltu arvamuste järjekorrast (või vastupidi, esimene arvamus on õige),
- tulemus ei sõltu arvutamise järjekorrast,
- ja nii edasi.

Järgnevalt toome ühe võimaliku valiku aksioome OR jaoks, AND puhul kehtivad analoogilised:

- a<=OR(f,g)<=ü
- OR(f,g)>=max(f,g)
- OR(a,f)=f
- OR(f, ü) = ü
- OR(f,g)=OR(g,f)
- OR(f’,g)<=OR(f,g), kui f’<=g
- OR(f,OR(g,h))=OR(OR(f,g),h)
- (Jf,g) OR(f,g)>max(f,g)

On leitud, et INFERENCE on oma iseloomult umbes sama nagu AND, seega sobivad sellele AND-i aksioomid. Kõik esitatud aksioomid pole sõltumatud; sama komplekti võib esitada teisel kujul.

Toodud valik pole ka iseenesest (sõltumatult esitusest) midagi absoluutset, vaid sõltub probleemalast. Tegelikkuses võib siis ka kogu aksiomaatika muutuda (mõned aksioomid puududa, lisanduda või teiseneda).

Näiteid valemitest, mida on kasutatud tehetes kindlushinnangutega (nad ei rahulda kõik tingimata kõiki ülaltoodud toodud aksioome):

- AND(f,g)=min(f,g)
- INFERENCE(f,g)=f*g
- OR(f,g)=(f+g)/(1+f*g)
- OR(f,g)=f+g-f*g
- OR(f,g)=(f+g)/2

- OR(f,g)=f (esimene mulje on õige)
- OR(f,g)= **Kui** f=g=a **Siis** a **Muidu** ü (Kui kaks midagi positiivset arvavad, peab väide kokku olema tõsi)
- skaalana antud hinnangu puhul võib operatsioonid määrata vastavate tabelitega

4.4. Järeldamine ebakindlate andmetega

Ebakindlad võivad olla andmed, ebakindel võib olla ka teadmusbaas. Ebakindlust andmetes vaatasime ülal. Reeglibaasis antakse reeglile tihti kindlushinnang (tegur), mis iseloomustab reegli usaldusväärsust. Süsteemi kasutamisel toimub ebakindluste töötlemine siis järgmistes etappides:

1. Olukorras esinev ebakindlus interpreteeritakse andmetega koos sisestatavaks kindlushinnanguks.
2. Reeglite eeldustes esinevad kindlushinnangud kombineeritakse eelduse koguhinnanguks (kasutades tavaliselt AND, mõnikord ka OR)
3. Eelduse koguhinnang ja reegli kindlushinnang kombineeritakse INFERENCE abil
4. Kui mitu reeglit viivad sama järelduseni, kombineeritakse tulemused OR abil
5. Saadud järelduse arvulist koguhinnangut interpreteeritakse tegelikkuses midagi tähendama.

Osa samme võib puududa. Süsteemi sisesed on sammud 2,3,4.

4.5. "Ebakindlad" teadmised: kordamisküsimusi ja ülesandeid

Kordamisküsimusi

- Ebakindluse/mittetäpsuse liike (näited!)
- Esinemine teadmussüsteemis (andmed, teadmusbaas)
- Formaliseerimise võimalusi (3)
- Reeglibaasis: 5/3 üleminekut
- Näiteid AND, OR, INF kohta
- AND, OR, INF soovitavaid omadusi (tõestada antud valemite puhul)
- Analoogiaid, allikaid (küsimustikud kaaludega, hindamissüsteem, arvamuste kombineerimine elus, . . .)

Ülesandeid

- Tõesta AND, OR, INF omadusi
- Antud järeldusskeem, leida tulemuse kaal
- Näide kindlast&ebatäpsest (eba . . .) teadmisest

5. Masinõpe

Masinõppe meetodeid on palju. Võib öelda, et mingi fakti meeldejätmine on samuti õppimine ja selles suhtes on arvutid inimesest kaugel ees. Siiski on õppimine laiemas mõttes - kohanemine muutuvate keskkonnatingimustega ja nende jätkuv kasutamine - arvutile raskesti kättesaadav.

Traditsiooniline masinõpe on tavaliselt suunatud mitmesuguste seoste leidmisele, mitte üksikute faktide meeldejätmisele (see oleks liiga lihtne) või käitumise kujundamisele (see oleks liiga raske). Seosed on tüüpiliselt väljendatud mingis enam või vähem formaalses (teadmiste kirjeldamise) keeles. Seoseid õpitakse, kasutades näiteks induktsiooni, üldistamist, juhuslikke mutatsioone testimisega jne.

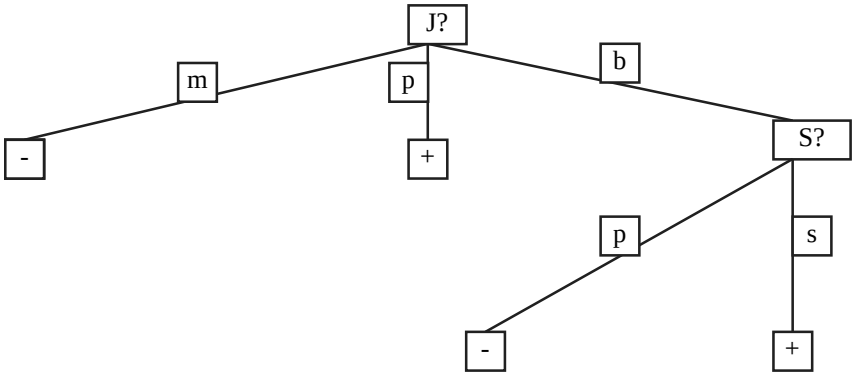
Selles jaotises vaatame konkreetset induktiivse masinõppe algoritmi ID3 ning selle põhjal visandame ka ühe masinõppe üldisema skeemi.

5.1. ID3

Olgu antud otsustustabel T, mille igas reas on andmed ühe objekti kohta. Veergudes on objekti atribuutide väärtused ja sellele vastav objekti klassimäärang. Järgnev näide on Quinlanilt (viit allpool): tabelis on kolm atribuuti, 8 objekti. Objektid jaotatakse kahte klassi.

Nr	P	J	S	Klassimäärang
1	l	b	s	+
2	p	b	p	-
3	p	p	s	+
4	l	m	s	-
5	p	m	s	-
6	p	b	s	+
7	p	m	p	-
8	l	b	p	-

Otsustustabelile võib vastata mitmeid otsustuspuid. Otsustuspuu tippudes on atribuudid, neist väljuvad kaared on tähistatud atribuudi väärtustega. Lehtedes on klassijaotused. Puud mööda liigutakse alates juurest, andes iga tipu puhul ette vastava atribuudi väärtuse antud objekti jaoks. Lõpuks jõutakse leheni, mi s määrab uuritava objekti klassi. Allpool on üks võimalik otsustuspuu ülaltoodud tabelile.



Masinõppe algoritm ID3 (J.R.Quinlan, Learning Efficient Classification Procedures and their Application to Chess End Games. Machine Learning. R. Michalski, J. Carbonell, T. Mitchell, Eds. Morgan Kaufmann Publishers, Los Altos, 1979) loob etteantud otsustustabeli põhjal tippude arvu mõttes optimaalse otsustuspuu. Algoritmi idee on järgmine.

Olgu C tabeli T objektide hulk. Piirdume lihtsuse mõttes kahe klassiga - plussid ja miinused (suurema arvu klasside puhul on käsitlus analoogiline). Kui p^+ ja p^- on vastavalt plusside ja miinuste sagedused hulgas C, siis mingi objekti klassifitseerimiseks hulgast C on vaja $M(C) = - p^+ \log p^+ - p^- \log p^-$ bitti (logaritmid on alusel 2). Selle info annab kogu otsustuspuu tervikuna.

Kui hakkame puud konstrueerima, valime juureks kõige enam infot andva atribuudi. Selleks märkame, et peale mingi tipu A etteandmist jaguneb hulk C osahulkadeks C_i, kus hulga C_i objektidel on atribuudi A väärtus A_i. Peale tippu A jäävat määramatust B(C,A) saab hinnata kui korrutiste

$M(C_i)$*(tõenäosus, et A väärtus on A_i)

summat üle kõikide i väärtuste. Me soovime, et see määramatus oleks minimaalne, mis ongi kriteerium juure valikuks.

Edasi on protseduur rekursiivne. Kui A väärtus on A_i, tuleb luua uus otsustuspuu hulga C_i jaoks. Kui C_i kõik elemendid on samast klassist, saame vastavasse tippu lehe antud klassimääranguga, vastasel juhul kasutame uue juure valikuks hulgale C_i juba tuttavat protseduuri. (Tegelikult algab rekursioon juba tipust A - kui hulga C kõik elemendid on samast klassist, koosneb kogu puu vastava klassinimega märgendatud lehest.)

Formaalselt võib algoritmi ID3 kirjeldada järgmiselt.

Sisend

- Hulk C, mille elementideks on tabeli read. Iga tabeli veeru element saab väärtusi vastava atribuudi määramispiirkonnast. Tabeli viimane veerg klassifitseerib vektori.

Väljund:

- Minimaalse tippude arvuga otsustuspuu hulga C jaoks.

Algoritm:

loo triviaalne otsustuspuu P, mis koosneb ühest märgendamata tipust t;

otsustuspuu (P, t, C).

procedure otsustuspuu (P, t, C)

if C elemendid kuuluvad samasse klassi k

then märgenda P tipp t selle klassitunnusega

else

vali-atribuut (P, t, C, A);

```
   Märgenda tipp t atribuudiga A (olgu väärtused A1…Ak);
   for I=1 to k
      ehita P tipust t harud uutesse tippudesse t1…ti;
      leia C alamhulk Ci, mis vastab atribuudi väärtusele Ai;
      otsustuspuu (P, ti, Ci)
   end for
end otsustuspuu.

procedure vali-atribuut  (P, t, C, A);
  'Vali atribuut A, mis mis vähendab maksimaalselt ebamäärasust tipus t;
  Leia infohulk M(C), mis on vajalik hulga C klassifitseerimiseks;
  korda iga atribuudi A jaoks  (olgu väärtused A1…Ak), mis ei esine teel puu P juurest kuni tipuni t
   for I=1 to k
      leia C alamhulk Ci, mis vastab atribuudi väärtusele Ai;
      Leia M(Ci)
   end for;
   B(C.A) = Σ M(Ci)* ( |Ci| / |C| )
  end korda;
  Leia atribuut A, mille puhul M(C) - B(C,A) on maksimaalne;
end vali-atribuut.
```

5.2. ID3 suure arvu objektide korral

Algoritm:

Vali juhuslik C' hulgast C ("aken")

Korda

Moodusta puu akna jaoks

Leia erandid ülejäänud objektides

Lisa erandid aknasse (variant: viska osa vanu ära)

Kuni erandeid pole

Otsustuspuu genereerimise keerukus on O(objektide-arv * atribuutide-arv * puu-tippude-arv)

Algoritmi analüüs:

- Millal puud pole?
- Mida teha vasturääkivate andmete puhul?
- Kas iga puu annab sama funktsiooni? Mis mõttes sama?
- Kas efektiivseim puu on ka sisult optimaalne?

5.3. Närvivõrgud

Üldine õppimise põhimõte närvivõrkudes on, et lisaks väljundi arvutamisele modifitseeritakse närvivõrku ennast - tavaliselt seoste kaale, harvemini lävifunktsiooni läveväärtust/tüüpi või närvivõrgu struktuuri. Kasutatakse erinevaid närvivõrgu õppimistüüpe. Üks võimalik õppimise idee kajastub Delta reeglis,

Dij= K*(Oi - Ti)*Vj, kus:

- Dij - muudatus neuronist j neuronisse i viiva seose kaalus
- K - õppimise kiirust määrav kordaja
- Oi - oodatav neuroni i väljund
- Ti - tegelik neuroni i väljund
- Vj - neuroni i sisend = neuroni j väljund.

Reegel ütleb siis, et kui neuroni i väljund ei vasta oodatule, muudetakse j-ndast neuronist i-ndasse neuronisse tuleva sisendi kaalu vea, j-nda neuroni väljundi ja etteantud kordaja korrutise võrra. Näiteks, kui tegelik väljund oli liiga väike (Oi > Ti), suurendatakse sisendi kaalu. Kordaja määrab õppimise kiiruse - kui see on väga suur, toimub õppimine kiiresti, kuid võrk võib "üle õppida" ja saada valesid tulemusi; kui K on väike, on õppimine aeglane.

Levinud õppimise meetod mitmetasemelistes närvivõrkudes on tagasilevi (backpropagation). Selle puhul liigutakse kõigepealt sisenditelt väljunditele, leides tulemuse. Võrreldes seda õige tulemusega, saadakse viga. Edasi liigutakse tagurpidi läbi kõikide kihtide, parandades kaalusid Delta reegli abil.

5.4. Üldisem käsitlus

Õppimine sisaldab millegi uue omandamist (meeldejätmist, tuletamist jne) ja kasutamist. Täpsemalt, õppimist on määratletud kui:

- teadmise/oskuse omandamine iseõppimise, juhendamise või kogemuse kaudu
- käitumise muutmine tegevuse, harjutamise või kogemuse kaudu

Mõistete õppimise meetodeid elust (enamasti rakenduvad kombineeritult):

- kirjelduse meeldejätmine ja integreerimine
- hõlmava objekti kitsendamine
- analoogia
- kirjelduse/funktsiooni induktsioon
- avastus

Uus, mida omandatakse, võib olla mitut liiki, nt. andmed, info, teadmine jne. (vt eelmised jaotised). Masinõppes vaadeldakse enamasti mingite seoste õppimist (tuletamist, tekitamist). Seosed on tüüpiliselt väljendatud tekstina mingis formaalses keeles. Näiteks, kõikide otsustuspuude hulka võib vaadelda formaalse keelena (selle grammatika on antud ilmutatud või ilmutamata kujul otsustuspuu kujundamise reeglites). Sellise keele tekst on üks konkreetne otsustuspuu. Matemaatilised funktsioonid esitavad teist funktsioonide klassi, mille üks tekst on konkreetne funktsioon. Ka närvivõrke saab vaadelda formaalse konstruktsioonina, mille üks “tekst” on konkreetne närvivõrk ja nii edasi.

Võimalikke õpitavaid tekste/ funktsioone/ keeli/ formalisme/ mehhanisme masinõppe jaoks:

- konjunktsioon
- otsustuspuu
- üldine normaalkuju
- lausearvutusvalem
- predikaatarvutusvalem
- protseduur
- närvivõrk
- teadmussüsteeem
- matemaatiline avaldis

Õppimise algoritme/põhimõtteid:

- ID3 ja teised sellest perest
- geneetilised algoritmid
- üldistamine predikaatidel
- klasteranalüüs
- närvivõrkude treenimine

Näidetel põhineva (induktiivse) õppimise üldine skeem:

1. Koosta õppe- ja katseandmete kogumid, koosta päring õppimiseks
2. Leia õpitav funktsioon õppeandmete põhjal
3. Katseta katseandmetega
4. Hinda tulemusi. Kui õpitud funktsioon on piisavalt hea, siis väljasta tulemus või kasuta seda. Vastasel juhul uuenda õppeandmed ja mine sammule 2.

5.5. Õppimine: kordamisküsimusi ja ülesandeid

Kordamisküsimusi

- Õppimine:
  - mõiste
  - meetodid
  - funktsioone, keeli, formalisme
  - algoritme/põhimõtteid
- Näidetel põhinev õppimine: üldine skeem
- ID3:
  - meetod
  - palju objekte
  - õppimise seisukohast
  - analüüs
- Lihtne õppimine närvivõrkudes

Ülesandeid

- Antud tabel, koostada otsingupuu
- Koostada lihtne tabel&puu
- Koostada minimaalne puu
- Leida sama tabeli põhjal 2 puud, mis annavad erinevad tulemused vms.
- Puu -> loogika
- Näide õppimisest närvivõrgus
- Antud mõiste definitsioon, mis liiki on (nt., kitsendab hõlmavat mõistet)?

6. Otsing

Kuna juba definitsiooni järgi on teadmussüsteemide puhul tegemist ebakorrapärase, halvasti struktureeritava probleemalaga, on siin millegi leidmiseks või ärategemiseks vaja sageli kasutada otsingualgoritme. Näiteks viib otsingu algoritmini see, kui süsteemi määramispiirkond liigendatakse seisunditeks ja lahenduse otsing teisendatakse tee otsingule lähteseisundist soovitud seisundini. Vaatleme otsingualgoritmide liike, Dijkstra algoritmi, heuristiliste otsingumeetodite põhimõtteid.

6.1. Otsingualgoritmid

Otsingualgoritme klassifitseeritakse mitmeti:

- Rakenduste järgi, kus rakendusteks võivad olla näiteks planeerimine, äratundmine, mängud, otsustamine,
- Probleemipüstituse formalismi järgi ("loogiline tase", millel toimub otsing): loomulik keel, matemaatiline loogika, formaalne keel, reeglid, freimid, semantilised võrgud,
- Millel otsitakse ("füüsiline tase") - tabelil, puul, orienteeritud/orienteerimata graafil, ja/või graafil, kaartel on/ei ole kaalud, automaadil,...,
- Mida otsitakse - lühimat teed tipust tippu, ühest tipust kõikide tippudeni, kõigist kõikideni,
- Selle järgi, kas kasutatakse otsingu heuristikaid,

Otsingu suunamiseks võib anda kogemuslikku (heuristilist) lisainfot. Näiteks võib kaardil soovitada tee otsingul arvestada ilmakaari, oma heuristilised reeglid on paljudes mängudes jne. Ilmselt ei garanteeri see lühimat ega parimat teed, kuid annab - nagu ka heuristiliste teadmiste puhul - hea lähendi paljudel juhtudel. Heuristiline otsing võib vähendada otsinguruumi väga suurel määral, näiteks ühe näite puhul 60000-lt erijuhult mõnekümne erijuhuni.

Ülesande lahendamise etappe otsingu kasutamisega:

- ülesanne tuleb formaliseerida,
- eristada formaliseeritud ülesandest otsingu osa (esitada otsingu terminoloogias),
- otsinguülesanne tuleb lahendada,
- lahendus tuleb tagasi tõlkida formaliseeritud ülesande terminoloogiasse,
- lahendus tuleb tõlgendada algse probleemi terminites.

6.2. Dijkstra algoritm

Dijkstra algoritm:

- näide heast algoritmist orienteeritud kaalutud graafil
- ei kasuta heuristilist infot
- leiab lühima tee ühest tipust kõikide tippudeni (samuti ühest tipust teise)
- keerukus $O(n^2)$, kus n on graafi tippude arv

Sisend

- Kaalutud orienteeritud graaf G=(V,E)
- Kaalud l(v,w), kokkulepe: l(v,v)=0; kui kaart (v,w) pole, siis $l(v,w)= \infty$
- Tipp v_0 hulgast V, millest alates lühimaid teid otsitakse

Väljund:

- Lühima tee pikkused tipust v_0 teistesse tippudesse

Meetod:

S:={ v_0 };

for $v \in V$ **do** D[v]:=l(v_0 ,v);

while S<> V **do**

vali tipp w hulgast V \ S, mille D[w] on minimaalne;

lisa w hulka S;

for $v \in V \setminus S$ **do** D[v] := min(D[v], D[w]+l(w,v))

end while.

6.3. Heuristiline otsing

Heuristilise otsingu puhul kasutatakse lisainfot otsingu suunamiseks. Näiteks, kui otsitakse kaardil lühimat teed punktist A punkti B, siis võib kasutada soovitust "vaata teid suunal A - B" (otsingu keerukus o(n□n) või "vaata ainult suuri teid" (otsingu keerukus o(n)). Mõnede ülesannete puhul võib vajalike operatsioonide arv väheneda tuhandeid kordi. Samal ajal ei pruugi heuristiline otsing leida parimat lahendit (leiab hea).

Seega on otsingu efektiivsemaks muutmiseks vähemalt kaks teed: (1) kasutada paremat algoritmi ja (2) kasutada suunavat lisainfot. Paneme tähele, et nad on sõltumatud; võib kasutada ühte, teist või mõlemat korraga.

6.4. Otsing: kordamisküsimusi ja ülesandeid

Kordamisküsimusi

- Otsingu rakendused: tüübid, näited
- Otsingu meetodite liigitusi
- Otsinguülesande formalisme
- Dijkstra algoritm, rakendused, efektiivsus
- Heuristiline otsing
- H.O. eelised/puudused
- Näiteid heuristikate kohta

Ülesandeid

- Antud probleem, formuleerida otsinguprobleemina
- Koostada otsingu graaf
- Kasutada Dijkstra algoritmi mingi graafi puhul (näidata, kuidas algoritm töötab)
- Kas antud heuristika töötab?

7. Kokkuvõte

Teadmussüsteemid on tekkinud vajadusest luua rakendusi, milledes kasutaja saab ise aktiivselt tegutseda mõistete ja seoste nivool (pärida, luua uusi, muuta, kustutada).

Sellistel süsteemidel on tihti järgmised omadused:

- heuristiline, ebakindel info,
- süsteem peab õppima ja selle tulemusena uuenema,
- parima lahenduse leidmine on liiga kulukas, seepärst piirdutakse hea lahenduse leidmisega.

Kasulikke tehnikaid selliste süsteemide realiseerimisel:

- teadmiste kujutamise keeled, deklaratiivne programmeerimine,
- reeglisüsteemid,
- kaalud,
- (heuristiline) otsing,
- närvivõrgud,
- loomuliku keele töötlus.

Arendus: kuna spetsifikatsiooni on raske luua (spetsifikatsioon on süsteem), siis kasutatakse tihti prototüüpimist koos valideerimisega.

8. Kirjandust

Kirjandust ja viiteid võib otsida WWW-st selliste märksõnade alt nagu “knowledge based systems”; “expert systems”, “artificial intelligence”, “machine learning”, "knowledge representation and reasoning" jne. Ka TTÜ raamatukogus on mitmeid raamatuid, mille pealkirjas on eeltoodud märksõnad.

Teemat võib leida mitut liiki ajakirjadest. On erialaajakirju (nt. IEEE Expert, IEEE Trans. On Knowledge and Data Engineering, Artificial Intelligence), aga vastavaid artikleid avaldatakse perioodiliselt ka muudes ajakirjades (nt. Communications of the ACM)

Konverentsid - on üldiseid sellele teemale pühendatud konverentse (nt. IJCAI, ECAI), erialateemadele pühendatud konverentse (nt. ES testimine, reaalaja ES), sektsioonid muudel konverentsidel.

Näiteid aadressidest ja raamatutest on toodud allpool.

- G. Piatetsky-Shapiro and W.J.Frawley, eds. Knowledge Discovery in Databases, AAAI/MIT Press, Menlo Park, Calif., 1991.
- S. Russell, P. Novig, Artificial Intelligence: A Modern Approach, Prentice Hall, Upper Saddle River, N.J., 1995.
- William H. VerDuin. Better Products Faster: A Practice Guide to Knowledge-Based Systems for Manufacturers", IRWIN Professional Publishing, 1995
- Edwards, Alex and Connell, N.A.D. Expert Systems in Accounting, 1989, Herfordshire, UK: Prentice Hall International (UK) Ltd.
- Rich, Elaine and Knight, Kevin. Artificial Intelligence Second Edition. 1991, New York: McGraw-Hill.
- http://itri.loyola.edu/kb/
- http://www.serviceware.com/Industry/Publications/Wizard/wizardt.htm