Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Alenka Kavčič

Prilagajanje v spletnih izobraževalnih hipermedijih z upoštevanjem nezanesljivosti uporabnikovega znanja

DOKTORSKA DISERTACIJA

Mentor: prof. dr. Saša Divjak

Ljubljana, 2001

‘In *that* direction,’ the Cat said, waving its right paw round, ‘lives a Hatter: and in *that* direction,’ waving the other paw, ‘lives a March Hare. Visit either you like: they’re both mad.’

‘But I don’t want to go among mad people,’ Alice remarked.

‘Oh, you can’t help that,’ said the Cat: ‘we’re all mad here. I’m mad. You’re mad.’

‘How do you know I’m mad?’ said Alice.

‘You must be,’ said the Cat, ‘or you wouldn’t have come here.’

Alice’s Adventures in Wonderland
Lewis Carroll

Zahvala

Iskreno se zahvaljujem mentorju prof. dr. Saši Divjaku za vodstvo in nasvete pri izdelavi disertacije ter za vsestransko podporo v času mojega podiplomskega študija. Zahvaljujem se tudi sodelavcem iz Laboratorija za računalniško grafiko in multimedije za vzpodbudo, podporo ter konstruktivne pripombe ob nastajanju tega dela.

Posebno pa sem hvaležna svojim domačim za potrpežljivost in razumevanje, saj so mi vedno stali ob strani ter me podpirali pri mojem študiju in delu.

Povzetek

Današnja informacijska družba z uvajanjem informacijskih tehnologij na vseh področjih delovanja prinaša nove izzive tudi v izobraževanju. Velike spremembe na tem področju se kažejo predvsem v uporabi učinkovitejših in kakovostnejših oblik poučevanja in učenja, novih možnosti izobraževanja na daljavo, vedno bolj pomembna in iskana pa postaja tudi možnost vseživljenjskega učenja.

Z bliskovito širitvijo interneta in priljubljenostjo svetovnega spleta postajajo vedno bolj zanimivi tudi spletni izobraževalni sistemi. Med njimi v zadnjem času igrajo vidno vlogo predvsem prilagodljivi hipermediji, katerih glavni namen je izboljšanje uporabnosti hipermedijev z vpeljavo inteligentnih tehnik, ki omogočajo poznavanje posameznega uporabnika, prirejanje informacij in njihovo predstavitev po meri uporabnika ter dinamično podporo navigaciji po spletnem materialu. Prav možnost prilagajanja uporabnikovim individualnim potrebam lahko veliko prispeva k izboljšanju procesa poučevanja, saj se je individualno poučevanje izkazalo kot najboljši način poučevanja.

Na področje prilagodljivih hipermedijev sodijo tudi raziskave v okviru pričujoče disertacije. Glavni namen je bil razviti in izdelati prilagodljiv hipermedijski sistem, ki pri modeliranju uporabnikovega znanja vključuje tudi elemente nezanesljivosti znanja. Prilagajanje sistema temelji na klasičnih tehnologijah prilagodljive podpore navigaciji (anotacija in neposredno vodenje), poleg teh pa uvaja tudi tehnologijo prilagodljivega vstavljanja povezav, ki omogoča dinamično izbiro povezav na relevantne enote znotraj domenskega hiperprostora.

Problema modeliranja uporabnikovega znanja z upoštevanjem nezanesljivosti pri opisu znanja smo se sistematično lotili s pregledom obstoječih tehnik za predstavitev in obravnavo nedoločenosti. Na osnovi primerjav posameznih tehnik smo za obravnavo nezanesljivosti znanja uporabnika izbrali teorijo mehkih množic. Tako je znanje uporabnika v modelu predstavljeno s pomočjo pripadnostnih funkcij trem mehkim množicam (nepoznanih, poznanih in naučenih konceptov), posodabljanje modela pa deluje na podlagi mehkih pravil (algoritem za razširjanje vrednosti znanja, s pomočjo katerega iz prikazanega znanja koncepta sklepamo tudi na znanje njegovih predpogojnih konceptov).

Postopek prilagodljivega vstavljanja povezav je svojevrstna kombinacija skrivanja in razvrščanja povezav. Osnovan je na mehkih relacijah med koncepti domene (model domene) in

na mehki predstavitvi uporabnikovega poznavanja teh konceptov (model uporabnika). Algoritem dinamično izbira povezave na enote, ki so najprimernejše za nadaljevanje in najbližje, tako vsebinsko kot težavnostno, trenutni enoti. Tako sestavljen seznam povezav ne vključuje vseh možnih povezav iz enote (skrivanje), povezave pa so v seznamu razvrščene po primernosti (sortiranje).

Predstavljen hipermedijski sistem je služil tudi kot orodje za ovrednotenje prilagodljivega sistema, njegove uporabnosti, všečnosti uporabnikom ter učinkovitosti vgrajenih metod prilagajanja (osredotočili smo se le na vstavljanje povezav in barvno anotacijo). Predvsem smo želeli raziskati vpliv uporabe prilagodljivega sistema na uspešnost učenja. V ta namen smo izdelali testno učno domeno, za katero smo izbrali začetni tečaj o programskem jeziku Java. Delovanje sistema smo nato preverili v realnem okolju na populaciji študentov, začetnikov v tem programskem jeziku. Sistem je bil med uporabniki dobro sprejet in se je izkazal kot učinkovit pripomoček, ki je enostaven za uporabo in pripomore tudi k boljšim učnim rezultatom. Uporabniki, ki so izkoriščali prilagoditvene možnosti sistema, so pri testu znanja v povprečju dosegli boljše rezultate.

Zanimiva je tudi ideja o združevanju informacij pri modeliranju uporabnikovega znanja. Model uporabnika smo nadgradili z možnostmi povezovanja z modeli drugih (so)uporabnikov sistema. Tu sicer ne gre za neposredno komunikacijo med različnimi uporabniki sistema, temveč za posredno primerjavo in sodelovanje samih modelov. Pri primerjavi gre predvsem za nekoliko drugačen pogled na dosežen nivo znanja, saj se znanje posameznega uporabnika ne meri več absolutno, temveč relativno glede na ostale uporabnike sistema. Sodelovanje med uporabniki pa omogoča združevanje uporabnikov v skupine, katerih cilj je s čim manjšo redundanco znanja skupno obvladanje učne domene. Pri tem vsak član skupine predela in se nauči le del domene tečaja, kot skupina pa pokrivajo celotno znanje domene.

Na koncu naj povzamemo še glavne prispevke disertacije, ki jih lahko strnemo v naslednjih pet točk:

- Primerjalna analiza različnih pristopov k obravnavi nedoločenosti pri modeliranju uporabnika. Izbira in uporaba primerne rešitve za modeliranje uporabnikovega znanja z upoštevanjem nezanesljivosti pri določanju stopnje poznavanja konceptov.
- Razvoj postopka prilagodljive podpore navigaciji z vstavljanjem povezav.
- Načrtovanje in implementacija prilagodljivega hipermedijskega sistema, ki uporablja postopek vstavljanja povezav na podlagi mehkega modela uporabnika.

- Ovrednotenje razvitega sistema na testni domeni in ocena učinkovitosti uporabljenih prilagoditvenih tehnologij.
- Uvedba komparativnega in kooperativnega združevanja informacij modelov uporabnikov.

Abstract

Adaptation in Web-Based Educational Hypermedia With Regard to the Uncertainty of User Knowledge

Ph.D. Thesis

In today's information society, information technologies have been introduced in all fields of activity, and have brought new challenges in the field of education. Tremendous changes in this area can be seen especially in the use of better and more efficient teaching and learning methods, new opportunities for distance learning, and the prospects for lifelong learning, which is becoming increasingly important these days.

With the rapid growth of the Internet and the popularity of the World Wide Web, Web-based Educational Systems are becoming ever more attractive. Among these, Adaptive Hypermedia in particular have been playing the key role lately. Their main purpose is to improve the usability of hypermedia through the integration of intelligent techniques, which enable a system to "understand" an individual user, arranging and presenting customized information and dynamic navigation support for Web-based material. This ability to adapt to an individual user's needs can significantly improve the teaching process, since it has been shown that the best method of teaching is individualized teaching.

The research carried out in this dissertation is also based in the field of Adaptive Hypermedia. Its main purpose was to develop and implement an Adaptive Hypermedia System, comprising elements of knowledge uncertainty in the process of user knowledge modelling. Adaptation of the system is based on traditional adaptive navigation support technologies (annotation and direct guidance), and furthermore presents a technology of adaptive link insertion. The latter enables a dynamic selection of links to relevant units in the domain hyperspace.

With regard to the uncertainty of knowledge description, the problem of user knowledge modelling was handled systematically by means of an overview of existing techniques for

representing and managing uncertainty. Based on a comparison of the individual techniques, a fuzzy set theory was chosen for treatment of knowledge uncertainty. Thus, the user's knowledge is represented in the model by membership functions for three fuzzy sets (unknown, known, and learned concepts). Model updating is based on fuzzy rules, which are used for inferring prerequisite concept knowledge on the basis of demonstrated concept knowledge (a knowledge value propagation algorithm).

The method of adaptive link insertion is a unique combination of link hiding and link sorting. It is based on fuzzy relations between domain concepts (a domain model) and a fuzzy representation of the user's knowledge of these concepts (a user model). The algorithm dynamically selects links to units that are most appropriate for learning next, and closest to the current unit in terms of content and level of difficulty. A list of links compiled in this manner does not include all possible links from the current unit (hiding), and organizes the links according to their suitability (sorting).

The hypermedia system presented also served as a tool for evaluating an adaptive system - its usability, user satisfaction, and the efficiency of the adaptive methods implemented (we focused on link insertion and annotation only). Above all, we sought to investigate the influence of adaptive system usage on learning efficiency. Therefore, a teaching domain was developed, namely, an introductory course in the Java programming language. The system was then tested in a real environment on a population of students, all beginners in Java. The system was well received by them and proved to be a valuable, easy-to-use tool that led to better learning results. On average, users who took advantage of adaptive system capabilities scored higher on the final examination.

The idea of merging information when modelling user knowledge is also a very attractive one. Therefore, the user model was upgraded so that it was possible to connect it to models of other system users. In this regard, direct communication between different system users is not of interest, rather only indirect comparison and cooperation among models are important here. Model comparison provides, above all, a different perspective on the level of knowledge acquired, in which the individual user's knowledge is measured not in absolute terms, but relative to that of other users of the same system. Cooperation among users provides an opportunity to create groups of users who have the task of group-mastering the teaching domain with minimum knowledge redundancy. Thus, each member of the group studies and learns only a part of the domain course, while the group as a whole covers all of the domain knowledge.

In conclusion, the main contributions of this dissertation may be summarized:

- Comparative analysis of different approaches for handling uncertainty in user modelling. Selection and application of a suitable solution for user knowledge modelling with regard to uncertainty when determining the level of concept knowledge.
- Development of an adaptive navigation support technology of link insertion.
- Design and implementation of an Adaptive Hypermedia System that uses a link insertion technology based on a fuzzy user model.
- Evaluation of the developed system in the testing domain, and efficiency assessment of the adaptive technologies used.
- Comparative and cooperative information merging of user models.

Kazalo

1. Uvod

Informacijska družba in spremljajoče nove tehnologije so prinesle nove izzive tudi na področju izobraževanja. Velike spremembe, ki so posledica uvajanja informacijskih tehnologij, se kažejo predvsem v uporabi učinkovitejših in kakovostnejših oblik poučevanja in učenja, novih možnosti izobraževanja na daljavo, vedno bolj pomembna in iskana pa postaja tudi možnost vseživljenjskega učenja.

Zadnja leta je elektronsko učenje v pravem razcvetu. Za primer navedimo razmere v Združenih državah Amerike [Ubell 2000]. Po uradnih podatkih in napovedih se bo povpraševanje po elektronskem učenju povečalo s pet odstotkov vseh študentov v visokem šolstvu iz leta 1998 na petnajst odstotkov v letu 2002. V podjetjih, kjer je njegova uporaba še močneje zastopana, pa je že v lanskem letu dosegel 40% vseh načinov izobraževanja, v naslednjih letih pa pričakujejo vsako leto podvojitev tega števila. Elektronsko učenje je torej trend, ki ga ne moremo in ne smemo zanemariti, saj bo v prihodnosti igralo vedno večjo vlogo.

Seveda je tehnologija že od nekdaj ponujala rešitve problema pomanjkanja individualnega poučevanja v razredih. Tu so se računalniki izkazali kot zelo primerni pedagoški pripomočki, saj zaradi svojih zmogljivosti in atraktivnosti zlahka pritegnejo in motivirajo učence. S hitro razširitvijo interneta in popularizacijo svetovnega spleta postaja vedno bolj zanimiva tematika tudi poučevanje preko spleta. Zato so postali zelo popularni hipermedijski izobraževalni sistemi, saj so na hipermedijih temelječi sistemi že po svoji zasnovi zelo primerni za spletno uporabo. Novi standardi dokumentov in programski jeziki (kot so na primer HTML, XML, JavaScript ali Java) so še izdatneje podprli širitev izobraževalnih sistemov v svetovni splet. Podobno kot so k razširitvi interneta veliko prispevali grafični uporabniški vmesniki do interneta in uveljavljanje svetovnega spleta, predstavlja vpeljevanje “inteligence” v splet velik korak k večji vlogi spletnih izobraževalnih sistemov. Tu mislimo predvsem na možnosti poznavanja uporabnika, prirejanje informacij in njihovo predstavitev po meri uporabnika ter dinamično podporo navigaciji po spletnem materialu.

Poglavitne odlike spletnih izobraževalnih sistemov so njihova neodvisnost od platforme, enostavnejša priprava materialov, ki ni vezana na točno določen produkt, enostavnejše in cenejše vzdrževanje, posodabljanje in tudi uporaba sistemov, možnost komunikacije in sodelovanja med uporabniki sistema ter dostopnost na daljavo in s tem možnost uporabe iz katerekoli lokacije (ne samo v šolski učilnici, temveč tudi od doma, iz službe, na potovanjih in

podobno). Vse naštete lastnosti so zelo zaželene tudi pri učenju na daljavo in vseživljenjskem učenju, ki postajata vedno bolj pomembni vrsti izobraževanja v današnji informacijski družbi.

Ena od poti razvoja z novimi tehnologijami podprtih izobraževalnih sistemov je pripeljala tudi do prilagodljivih izobraževalnih hipermedijev, ki svojo veljavo pridobivajo predvsem v zadnjih letih. Prilagodljivi hipermedijski sistemi so plod raziskav s področja hipermedijev in modeliranja uporabnika, njihov splošni namen pa je izboljšanje uporabnosti tradicionalnih hipermedijev. Ti sistemi so zmožni prilagajanja vsakemu posamezniku, ki jih uporablja pri učenju. Prav ta zmožnost prilagajanja veliko prispeva k izboljšanju procesa poučevanja, saj je znano, da je najboljši način poučevanja ravno individualno poučevanje.

Čeprav so se prilagodljivi hipermediji pojavili šele v začetku devetdesetih let dvajsetega stoletja, so zelo hitro rastoče področje. Zanimanje zanje se je bliskovito razširilo, članki na to temo pa so se pojavljali na različnih konferencah, od hipermedijskih do konferenc o modeliranju uporabnika. Leta 1994 so v okviru četrte mednarodne konference o modeliranju uporabnikov UM’94 v Združenih državah Amerike organizirali tudi prvo delavnico na temo prilagodljivih hipermedijev (Workshop on Adaptive Hypertext and Hypermedia). Tej je sledila še vrsta drugih uspešnih delavnic, na katerih so raziskovalci predstavljali svoje delo, izmenjevali izkušnje ter razpravljali o novih tehnologijah in aplikacijah s področja prilagodljivih hipermedijev. Razširjenost tega raziskovalnega področja je pripeljala tudi do prve mednarodne konference s področja prilagodljivih hipermedijev in prilagodljivih spletnih sistemov AH 2000 (International Conference on Adaptive Hypermedia and Adaptive Web-based Systems), ki se je odvijala konec lanskega avgusta v italijanskem Trentu. Zbornik konference je objavila založba Springer Verlag v okviru zbirke Lecture Notes in Computer Science [Brusilovsky 2000]. Naslednja konferenca iz te serije je napovedana za leto 2002.

Na področje prilagodljivih hipermedijev spadajo tudi raziskave v okviru pričujoče disertacije. Glavni namen je bil izdelati prilagodljiv izobraževalni sistem, ki v modelu uporabnika vključuje tudi elemente nezanesljivosti znanja in tako gradi realnejšo podobo znanja uporabnika. Sistem naj bi temeljil na postopku prilagodljivega vstavljanja povezav, ki omogoča dinamično določanje povezav iz trenutno prikazane učne enote na sorodne enote znotraj domenskega hiperprostora. Ta postopek je osnovan na mehkih relacijah med koncepti domene in na uporabnikovem poznavanju teh konceptov. Zato algoritem pri izbiri povezav upošteva tako trdnost relacij v grafu domene kot z neko mero nedoločenosti opisano znanje uporabnika.

1.1 Opis poglavij

Disertacijo sestavlja osem poglavij, skupaj z uvodnim in zaključnim poglavjem, celotno delo pa je logično razdeljeno na tri dele. V prvem, ki zajema poglavja od ena do štiri, smo se osredotočili na opis problema in pregled področja skupaj s trenutnim stanjem v svetu. Drugi del sestavljata peto in sedmo poglavje, kjer je opisan prototip prilagodljivega sistema in predlagane možnosti združevanja informacij iz modelov uporabnikov. Tretji del (šesto poglavje) pa smo namenili obsežnejšemu testiranju sistema, ki zajema tako pripravo testne učne domene in testnih različic sistema kot tudi izvedbo poizkusa in analizo njegovih rezultatov. V nadaljevanju so povzete vsebine posameznih poglavij.

Prvo poglavje je uvodno poglavje. V njem je strnjenih nekaj misli in idej, ki so vodile k raziskavam in nastanku tega dela.

V drugem poglavju smo opisali ozadje in razvoj spletnih izobraževalnih sistemov, od začetkov v sistemih za računalniško podprto poučevanje do današnjih dni. Podrobneje so opisani tudi inteligentni učni sistemi in izobraževalni hipermedijski sistemi, saj so oboji osnova za razvoj prilagodljive hipermedije. Na koncu poglavja se dotaknemo tudi problema navigacije v hiperprostoru, ki ima ključni pomen pri razvoju prilagodljivih hipermedijev.

Tretje poglavje je namenjeno prilagodljivim hipermedijem. Najprej so opisana različna področja njihove uporabe skupaj s primeri prilagodljivih sistemov. Sledi opis splošne strukture in dveh najpomembnejših komponent: modela domene in modela uporabnika. Pregledu prilagoditvenih tehnologij in njihove uporabnosti sledi opis in primerjava obstoječih prilagodljivih sistemov. Tu smo se omejili le na sisteme s področja izobraževanja (teh je tudi največ realiziranih), ki imajo vse lastnosti prilagodljivih spletnih hipermedijev.

V četrtem poglavju smo na splošno obdelali problematiko obravnave nedoločenosti, ki zajema tudi nezanesljivost. Najprej smo poskušali klasificirati različne vrste nedoločenosti, potem pa smo sistematično predelali obstoječe tehnike, ki se uporabljajo za predstavitev in obravnavo posameznih vrst nedoločenosti. Podrobneje smo pregledali teorijo mehkih množic, teorijo mehkih mer ter razne druge mere nedoločenosti. Poglavje smo sklenili s pregledom različnih pristopov k obravnavi nedoločenosti pri modeliranju uporabnika. Večina jih izhaja iz uporabe v umetni inteligenci, predvsem iz ekspertnih sistemov in inteligentnih učnih sistemov.

Peto poglavje je nekoliko obširnejše, saj zajema ključni del naloge – opis sistema, ki smo ga razvili v okviru doktorske disertacije. Začnemo z opisom osnovnih komponent sistema:

modela učne domene in njegove povezave z učnimi enotami ter pristopa k modeliranju uporabnika z uporabo mehkih množic. Sledi opis določanja uporabnikovega poznavanja posameznih domenskih konceptov (predstavitev znanja v modelu uporabnika) ter njihovega posodabljanja na podlagi mehkih pravil (vzdrževanje modela uporabnika). Na osnovi modela domene in modela uporabnika deluje tudi prilagodljivo obnašanje sistema. To temelji na klasičnih tehnologijah prilagodljive podpore navigaciji (anotacija in neposredno vodenje), poleg tega pa uvaja tudi tehnologijo prilagodljivega vstavljanja povezav. Slednja je bila razvita za uporabo v opisanem sistemu in je svojstvena kombinacija skrivanja in razvrščanja povezav. Poglavje zaključimo z opisom prototipa zasnovanega sistema, njegovo implementacijo in primerjavo z drugimi prilagodljivimi izobraževalnimi hipermediji.

Šesto poglavje se ukvarja s testiranjem izdelanega sistema. Najprej opišemo učno domeno, ki smo jo uporabili pri testiranju, temu pa sledi opis treh testnih različic sistema in glavnih razlik med njimi. Po podrobnejšem opisu poteka testiranja in opravljenih meritev obravnavamo tudi dobljene rezultate. Iz njihove analize sledi, da obstaja pomembna razlika med uporabniki, ki so izkoriščali prilagoditvene možnosti sistema, in vsemi ostalimi. Slednji so namreč pri testu dosegli za petino slabše rezultate. Na koncu poglavja so strnjene naše ugotovitve in zaključki ter podane primerjave z rezultati podobnih raziskav.

V sedmem poglavju smo naredili še korak naprej na področju modeliranja uporabnikovega znanja. Predlagali smo možnosti za združevanje informacij modelov uporabnikov, ki smo jih uporabili za primerjanje in sodelovanje med posameznimi uporabniki.

Na koncu smo v osmem poglavju podali nekaj zaključnih misli, navedli možne nadaljnje smernice razvoja in raziskav ter sistematično pregledali prispevke k znanosti tega dela.

2. Izobraževalni sistemi

Ponavadi je učna praksa taka, da se vse učence v skupini poučuje z enako hitrostjo in na enak način. Kadar sta tako tempo učenja kot način poučevanja nespremenljiva, je uspeh vsakega posameznika odvisen od njegovih sposobnosti (nadarjenosti). Če pa se tempo učenja ali način poučevanja spreminja za posamezne učence, je veliko več možnosti, da je pri učenju uspešnih (osvoji učno snov) več učencev. Z zagotovitvijo ustreznih pogojev učenja je mogoče doseči, da več kot devetdeset odstotkov učencev obvlada večino ciljev, ki jih sicer dosežejo le "dobri učenci" [Gagné 1988]. Zato se individualno poučevanje oziroma učenje že dolgo pojmuje kot najprimernejši in najučinkovitejši način pridobivanja novega znanja, saj lahko nenehno prilagaja poučevanje trenutnim potrebam učenca.

Ker pri poučevanju največkrat pride en učitelj na (večjo) skupino učencev, je nerealno pričakovati, da se bo učitelj individualno ukvarjal z vsakim učencem. Tak problem pomaga reševati tehnologija, saj je učenje s pomočjo računalnika lahko povsem individualizirano. Pravzaprav je tehnologija že od nekdaj ponujala rešitev za problem pomanjkanja individualnega poučevanja v razredih. Pri tem pa so se ravno računalniki izkazali kot najprimernejši pedagoški pripomočki, saj tako zaradi svojih zmogljivosti kot zaradi svoje privlačnosti zlahka pritegnejo in motivirajo učence.

2.1 Učne teorije za izobraževalne tehnologije

Izobraževalni teoretiki so že od nekdaj povzdigovali tehnologijo in ji pripisovali, večkrat pretirano, velike možnosti tudi v izobraževanju [Berg 2000]. Vendar pa je izobraževalna tehnologija samo tako uspešna, kot so učinkovite učne metode, ki jih uporablja. Seveda so bile učne teorije že od začetka gonilna sila pri pripravi računalniško podprtega izobraževanja in so zato tudi tesno povezane z razvojem izobraževalnih sistemov.

Prvi sistemi za računalniško podprto poučevanje, ki so poudarjali predvsem urjenje in vajo (angl. drill and practise), so temeljili na *vedenjski* ali *behavioristični* teoriji [Good 1990]. V ospredju te teorije je vedenje oziroma povezanost med vedenjem posameznika in posledicami izbranega vedenja. Ta zveza med vedenjem, dražljaji in odgovori omogoča, da se obnašanje oblikuje z uporabo principa okrepitve (če vedenju takoj sledi okrepitev in če je okrepitev močna

in pogosta, lahko pričakujemo, da se bo vedenje ohranilo). Učenje je tako usmerjano od zunaj po principu okrepitve in večinoma podpira le nižjenivojske kognitivne procese.

Kasneje se je na površje prebila *konstruktivistična* učna teorija [Good 1990] in je postala prevladujoča teorija v sodobnem računalniško podprtem poučevanju. Po konstruktivistični teoriji je učenje predstavljeno kot aktivni proces izgradnje znanja, v katerem ima učenec aktivno vlogo pri oblikovanju novih idej ali konceptov na osnovi predhodnega znanja. Učenje torej temelji na učenčevem kognitivnem delovanju, saj je učenec aktivni udeleženec pri pridobivanju znanja, njegovem prestrukturiranju, obdelavi, odkrivanju in poskusih, kar ima za posledico, da je znanje smiselno, organizirano in trajno. Učenje je notranji proces, na katerega vplivajo učenčevi cilji, predhodno znanje in učni cilji, ter podpira višjenivojske kognitivne procese. Konstruktivizem poudarja predvsem pomoč učencu pri izgradnji njegovega lastnega znanja. Konstruktivistična teorija se v mnogih pogledih dobro prilega novim tehnologijam [Berg 2000] in je tesno povezana s hipermedijskimi učnimi okolji, kjer je učenje nadzorovano s strani učenca.

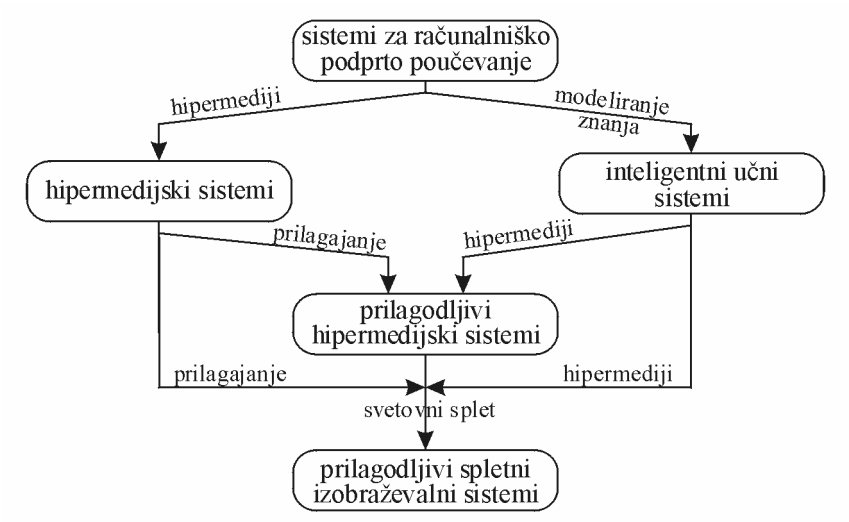
Če na kratko povzamemo, lahko na vrste učenja gledamo iz dveh zornih kotov: *objektivističnega* (vedenjskega) in *konstruktivističnega* (kognitivnega) [Hayes 1998, Guttormsen 2000]. Prvi temelji na učenju z razlago (s podajanjem znanja), zato je tako naučeno znanje bolj eksplicitno in deklarativno. Pri tem je znanje objektivno in obstaja neodvisno od učenca. Po drugi strani pa konstruktivizem predpostavlja, da se ljudje učijo z dejavnostjo in da znanje gradijo. Kot posledica je tako znanje bolj intuitivno, a trdnejše. Kognitivne vrste učenja so bolj povezane z samim mišljenjem in razumevanjem.

2.2 Računalniško podprti izobraževalni sistemi

Računalniška tehnologija se bolj ali manj uspešno uporablja v izobraževanju že več desetletij. Razvoj računalniško podprtih izobraževalnih sistemov je tesno povezan z razvojem strojne in tudi programske opreme, torej z samo zmogljivostjo (predvsem osebnih) računalnikov, razvojem novih tehnologij ter z raziskavami s področij umetne inteligence, človeškega zaznavanja (kognicije) in drugih.

Sistemi za računalniško podprto izobraževanje so se v svojem štiridesetletnem razvoju močno spremenili. Danes se sicer uporablja več vrst sistemov, vendar pa gredo večinoma vse raziskave in spremembe v isto smer – proti svetovnemu spletu. Na sliki 2.1 je prikazan razvoj in

delitev računalniško podprtih izobraževalnih sistemov. Pri tem velja poudariti, da taka delitev na posamezne vrste ni ostra, saj so realni sistemi največkrat križanci med različnimi vrstami.



Slika 2.1

Razvoj računalniško podprtih izobraževalnih sistemov

V naslednjih razdelkih bomo na kratko opisali vsako od naštetih vrst izobraževalnih sistemov.

2.2.1 Sistemi za računalniško podprto poučevanje

Prvi računalniško podprti izobraževalni sistemi so se pojavili že v zgodnjih šestdesetih letih dvajsetega stoletja. To so bili sistemi za *računalniško podprto poučevanje* ali *računalniško podprto učenje* (angl. *computer–assisted instruction* ali *computer–assisted learning*). Ti sistemi so bili namenjeni predvsem dvema različnima rabama: kot elektronske knjige ali kot sredstvo za utrjevanje pridobljenega znanja. Elektronske knjige so posnemale navadne tiskane učbenike, saj so bile le zbirke vnaprej pripravljenega učnega materiala, ki ga je učenec pregledoval zaporedno po vnaprej določeni poti. Sistemi za utrjevanje znanja pa so prikazovali različne probleme in odgovarjali na učenčeve rešitve s pomočjo vnaprej shranjenih odgovorov ter pri reševanju nudili tudi določeno vrsto pomoči. Taki sistemi so temeljili na principu dražljajev in odgovorov vedenjske učne teorije, ki se odlično prilega njihovi uporabi.

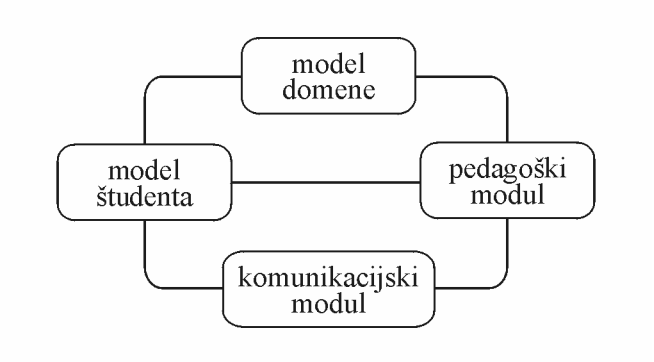
Sistemi za računalniško podprto poučevanje uporabljajo nespremenljivo predstavitev didaktičnega materiala, zato je pot skozi učni material linearna in vedno enaka. Tudi izbira

problema, reakcija ob napačnem odgovoru in izbira učne strategije so vnaprej definirane procedure v sistemu. Zato se taki sistemi obnašajo enako za vse učence, ki sistem uporabljajo, in ne nudijo nikakršne podpore individualiziranemu poučevanju. Prav ta majhna fleksibilnost je največja pomanjkljivost sistemov računalniško podprtega poučevanja.

2.2.2 Inteligentni učni sistemi

Z razvojem sistemov za računalniško podprto poučevanje so se v začetku sedemdesetih let pojavili prvi *inteligentni učni sistemi* (angl. *intelligent tutoring systems*), ki uporabljajo formalizme in tehnike umetne inteligence za predstavitev znanja v sistemu. Inteligentni učni sistemi so pomenili veliko revolucijo v izobraževalnih sistemih, saj so od programiranja odločitev prešli k programiranju znanja [Millán 2000]. Prav ta prehod k modeliranju znanja in učnega procesa je omogočil prilagajanje sistema vsakemu posameznemu učencu, spremenljivo pot skozi učni material ter popolno individualnost učenca.

Osnovna struktura inteligentnih učnih sistemov je modularna in se v skoraj tridesetletnem razvoju sistemov ni spremenila. Sestavljajo jo štiri komponente: model domene, model študenta, pedagoški modul ter vmesnik ali komunikacijski modul. Slika 2.2 prikazuje zgradbo inteligentnih učnih sistemov z vidika omenjenih štirih komponent.



Slika 2.2
Zgradba inteligentnih učnih sistemov

Na poučevanje lahko gledamo tudi z vidika komunikacije znanja [Wenger 1987]. Tako vsaka od navedenih komponent sistema lokalizira eno od štirih vrst znanja, to so znanje domene, znanje študenta, pedagoško znanje in komunikacijsko znanje.

Model domene. Najpomembnejša komponenta sistema je model domene, ki predstavlja znanje sistema o svoji učni domeni. Znanje domene je objekt komunikacije znanja in pojasnjuje, *kaj* se uči. Prav znanje domene je tisto, ki je bilo prvo eksplicitno predstavljeno v sistemu. Model domene je hkrati vir domenskega znanja in standard za ocenjevanje študenta.

Model študenta. Informacije, ki so značilne za vsakega posameznega študenta, hrani model študenta. Je prejemnik komunikacije znanja in pojasnjuje, *koga* se uči oziroma *kdo* je učenec. Ta model vključuje vse vidike študentovega obnašanja in znanja, ki vplivajo na njegovo delo in učenje.

Pedagoški modul. Model načina učenja je predstavljen s pedagoškim modulom, ki vsebuje principe in metode učenja. Pedagoško znanje je veščina komunikacije znanja in razlaga, *kako* učiti snov in učenca. Ta modul skrbi za model učnega procesa, na primer kdaj ponoviti, kdaj predstaviti novo snov, katero snov predstaviti naslednjo in podobno.

Komunikacijski modul. Vmesnik med sistemom in študentom predstavlja komunikacijski modul, ki predpisuje obliko komunikacije znanja. Skrbi za interakcijo sistema s študentom (na primer za prikaz dialogov ali predstavitev na zaslonu) in določa končno obliko akcij pedagoškega modula.

Največja prednost inteligentnih učnih sistemov se kaže v sposobnosti prilagajanja sistema vsakemu posameznemu učencu in vodenju skozi učni material. Vendar pa ravno zaradi sistemsko vodenega poučevanja učenec nima nobenega nadzora in vpliva na učni proces. To je tudi ena največjih kritik inteligentnih učnih sistemov. Poleg tega je razvoj teh sistemov dolgotrajen, uporaba in vzdrževanje pa zelo draga, zato se večina sistemov uporablja le kot prototipi v laboratorijih, kjer so jih razvili, in se nikoli niso imeli priložnosti izkazati pri uporabi v šolah.

2.2.3 Izobraževalni hipermedijski sistemi

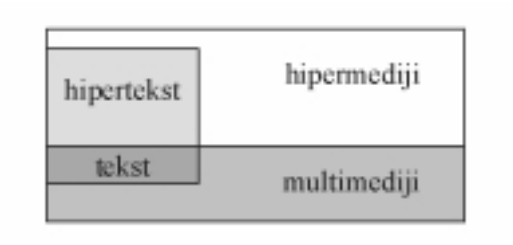
Hipertekst. Hipertekst lahko definiramo kot način ustvarjanja in dostopanja do nelinearnega besedila [Kommers 1996]. Tako besedilo sestavljajo manjši zaključeni odstavki. Ključne besede v besedilu kažejo na druge odstavke ali na skupine besed ali besede v drugih odstavkih. Tako so odstavki povezani s pomenskimi ali miselnimi povezavami. Idejo hiperteksta je prvi uporabil Vannevar Bush [Bush 1945] v povezavi s problemom raziskovalcev pri pregledovanju ogromnih količin objavljenih informacij njihovih kolegov. Namesto kartic z indeksi po omejenem številu ključnih besed je predlagal sistem, kjer bi vsaka beseda v teh

opisih kazala na vse odlomke v celi zbirki dokumentov, v katerih se pojavlja ista beseda. Opisani stroj, imenovan *memex*, bi skrbel za osebno informacijsko bazo in bi shranjeval vse dokumente (knjige, zapise, dnevnike, beležke, sporočila in podobno) posamezne osebe, ki bi bili organizirani po asociacijah. Tak način organizacije je bližji človekovemu razmišljanju, saj tudi človeški možgani delujejo na principu asociacij. Sam izraz hipertekst, ki odraža nelinearno strukturo besedila, je skoval Ted Nelson [Nelson 1967] v šestdesetih letih, ko je delal na ambiciozno zastavljenem projektu *Xanadu*, ki temelji na ideji medsebojno povezanega nelinearnega besedila in drugih medijev. Tu je hipertekst razložen kot nezaporedno pisanje, ki se veji in dovoljuje bralcu različne izbire. To zaporedje kosov besedila povezujejo povezave, ki bralcu omogočajo izbiro različnih poti.

Multimediji. Računalniško podprte aplikacije, ki dovoljujejo uporabniku videti in slišati različne vrste informacij preko enega zaslona z avdio podporo, imenujemo multimediji [Kommers 1996]. Taka kombinacija različnih elektronskih medijev, kot so tekst, slike, zvok, video, animacije in drugi, prinese večjo raznolikost informacij in olajša njihovo razumevanje. Pri tem so mediji mišljeni kot orodja, ki jih uporabljamo za shranjevanje, obdelavo in komunikacijo informacij.

Hipermediji. Izraz hipermediji pomeni nezaporedno in nelinearno predstavitev medsebojno povezanih multimedijskih dokumentov. Zajema tako nelinearno strukturo vozlišč, ki vsebujejo dele multimedijskih dokumentov, kot povezave med temi vozlišči. Povezanost dveh vozlišč kaže tudi na vsebinsko sorodnost obeh vozlišč.

Podobno kot je hipertekst s povezavami nadgrajen tekst, lahko tudi na hipermedije gledamo kot na s povezavami nadgrajene multimedije. Slika 2.3 prikazuje omenjene relacije. Pri tem velja opozoriti, da je navaden tekst le eden od medijev in da ga multimediji na nek način nadgrajujejo.

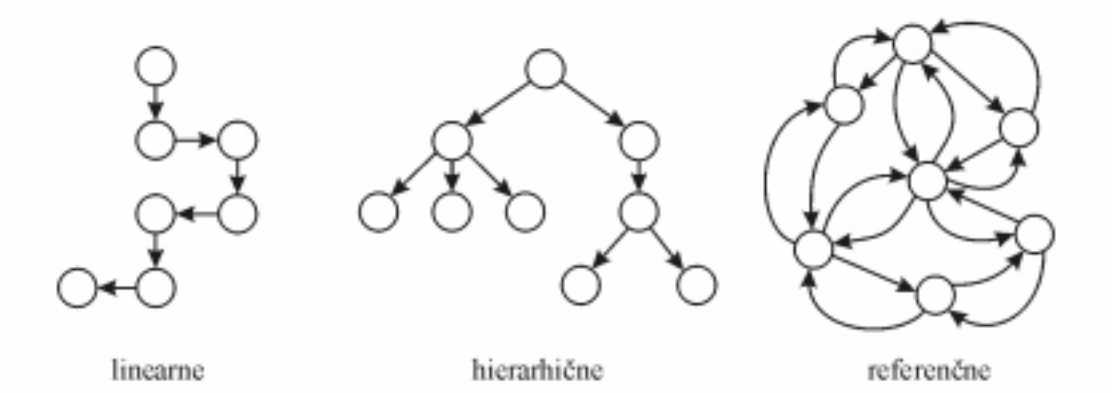


Slika 2.3

Relacije med tekstom, hipertekstom, multimediji in hipermediji

Osnovni gradniki hipermedijev so torej vozlišča in povezave med njimi. Vozlišča so informacijske enote, povezave pa fizično in konceptualno povezujejo te enote [Kommers 1996]. Hipermedijska vozlišča tvorijo tudi neko semantično strukturo, zato ne preseneča podobnost predstavitve hipermedijev s semantičnimi mrežami, poznanimi iz umetne inteligence kot sheme za predstavitev znanja [Angelides 1995]. Semantično mrežo sestavljajo vozlišča, ki so medsebojno povezana z različnimi vrstami asociativnih povezav. Tudi hipermediji nudijo obliko predstavitve informacij, ki je podobna asociativnemu hranjenju znanja pri človeku. Glavna razlika med hipermediji in semantičnimi mrežami pa je v vrsti povezav, saj imajo povezave pri semantičnih mrežah semantični kontekst, pri hipermediji pa so poljubno urejene.

Hipermediji se pojavljajo v različnih oblikah, ki jih lahko preprosto izrazimo z medsebojno povezanostjo vozlišč [Oliver 1995]. Tri najznačilnejše oblike prikazuje slika 2.4. Linearna oblika ima malo povezav, vozlišča pa so povezana v določenem zaporedju. Taka oblika je podobna običajnemu tekstu v knjigah in omogoča le zaporedno sledenje povezavam. Več možnosti pri izbiri poti skozi učni material nudi hierarhična oblika, kjer povezave tvorijo drevesno strukturo. Največ svobode pa daje referenčna oblika, ki je popolnoma nestrukturirano okolje z več povezavami med vozlišči, ki omogoča prosto sprehajanje preko referenčnih povezav.



Slika 2.4
Povezave v hipermedijih

Čeprav se pojem hipermedijev ponavadi povezuje s svetovnim spletom, so se prvi hipermedijski sistemi pojavili že dolgo pred prihodom in razširitvijo interneta. V osemdesetih letih se je kot alternativa inteligentnim učnim sistemom razvila nova veja računalniško podrtih izobraževalnih sistemov, to so *izobraževalni hipermedijski sistemi* (angl. *educational*

hypermedia systems). Prvi hipermedijski sistemi, kot sta na primer HyperCard ali AuthorWare, so uporabljali različne formate zapisa, ki pa niso bili medsebojno prenosljivi. Priprava materiala na enem sistemu je zato zahtevala isti sistem tudi za prikazovanje. Prav tako orodja niso delovala na vseh sistemskih platformah. Kljub temu pa so bili hipermedijski sistemi zelo priljubljeni v izobraževanju. Danes so te težave presežene s skupnim, od platforme neodvisnim standardom za opis hipermedijskih dokumentov, ki je omogočil bliskoviti razširitev uporabe hipermedijev tako v pedagoške kot nepedagoške namene.

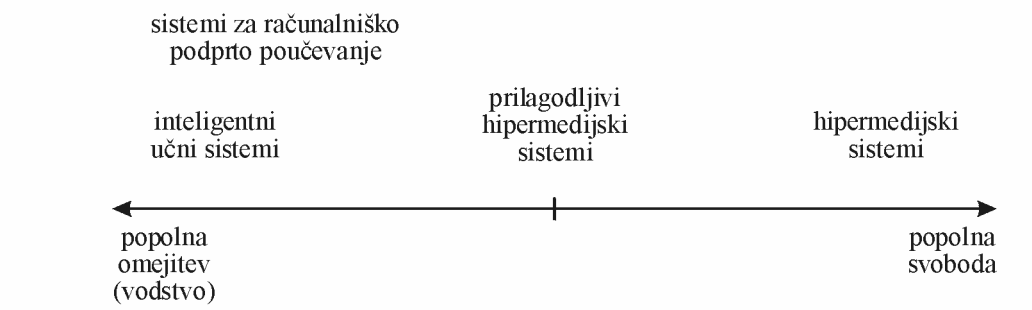
Izobraževalni hipermedijski sistemi omogočajo učencu dostop do ogromne količine učnega materiala, skozi katerega lahko prosto izbira pot. Tako ima učenec popolno kontrolo nad pregledovanjem materiala, seveda znotraj njegovih navigacijskih zmožnosti, in lahko po svoje raziskuje in razume zbirko učnega materiala ter samostojno izlušči pomen. Hipermedijski sistemi so ponavadi pasivna okolja, ki ne razlikujejo med različnimi uporabniki ter njihovim različnim znanjem in sposobnostmi. So statični, neprilagodljivi učni mediji, ki ne učijo, temveč nudijo učencu le dobro okolje za samostojno učenje. Ker vsebujejo le malo strukturnega znanja o vsebini, jih lahko označimo za nepedagoško tehnologijo [Eklund 1995].

Glavne prednosti hipermedijskih sistemov se kažejo v ureditvi in povezovanju informacij ter iz tega izhajajočih možnosti, v dostopu do velikih količin podatkov ter v svobodi pri izbiri poti skozi učni material in s tem tudi popolni kontroli nad učnim procesom. Vendar pa tudi pri hipermedijskih sistemih, podobno kot pri inteligentnih učnih sistemih, njihove največje pomanjkljivosti izhajajo ravno iz njihovih prednosti. Izpostaviti velja predvsem dve glavni pomanjkljivosti, ki sta znani kot problem izgubljenega v hiperprostoru (angl. lost in hyperspace) in problem kognitivne obremenitve (angl. cognitive overhead) [Nielsen 1990, Hammond 1991]. Prvi problem izhaja iz kompleksnosti hipermedijskih sistemov in se kaže v težavah pri iskanju nekoč že najdenih informacij ali v problemih pri določanju položaja in orientaciji v hiperprostoru. Drugi problem pa izvira iz veliko možnosti izbire poti, kompleksnosti domene ter dodatnega bremena navigacije. V najslabšem primeru se lahko zgodi, da zaradi prevelike svobode učenec celo izgubi stik z izobraževalnimi cilji.

2.2.4 Prilagodljivi hipermedijski sistemi

V predhodnih dveh razdelkih smo opisali inteligentne učne sisteme in hipermedijske sisteme, kateri uporabljajo dva popolnoma različna pristopa k računalniško podprtem poučevanju. Na eni strani so inteligentni učni sistemi, ki temeljijo na znanju in prilagajajo zaporedje poučevanja posameznemu učencu. Ti sistemi diktirajo potek učenja s kontrolo

učnega dialoga in strategij. Učenje je zato nadzorovano s strani sistema, ki vodi učenca skozi učni material. Po drugi strani pa hipermedijski sistemi temeljijo na multimedijskih dokumentih, ki so nadgrajeni s povezavami. Pri njih učenje sloni na različnih pripomočkih za navigacijo po domeni učenja, ki puščajo učencu popolno svobodo pri izbiri poti skozi učni material. Ti sistemi nudijo raziskovalno okolje in spodbujajo učenca k samostojnemu odkrivanju konceptov. Obe skrajnosti shematično prikazuje slika 2.5. Inteligentni učni sistemi in klasični sistemi za računalniško podprto poučevanje se nahajajo na strani omejevanja oziroma vodenja uporabnika pri učnem procesu. Nasprotno pa hipermedijski sistemi uporabniku omogočajo veliko svobode pri učnem procesu.



Slika 2.5

Omejevanje uporabnika v računalniško podprtih izobraževalnih sistemih

Na sliki 2.5 se v sredini med obema skrajnostma nahajajo *prilagodljivi hipermedijski sistemi* (angl. *adaptive hypermedia systems*). Ti uporabljajo novejše pristope, ki ubirajo neko srednjo pot in tako združujejo prednosti obeh skrajnih načinov. V osnovi so hipermedijski sistemi, vendar pa klasične hipermedije nadgrajujejo z inteligentnimi tehnikami, ki pomagajo uporabniku pri učenju. Tako nudijo s strani učenca voden učni proces in vso svobodo učenja z raziskovanjem, a se hkrati dinamično prilagajajo učenčevim individualnim potrebam, njegovemu nivoju znanja in učnim ciljem ter tako pomagajo učencu pri sprejemanju novega znanja. Prilagodljivi hipermedijski sistemi združujejo pasivni hipermedijski informacijski model z orodji za aktivno prilagajanje uporabniku. Učenec je torej voden implicitno in ne eksplicitno kot pri inteligentnih učnih sistemih.

Podrobneje so prilagodljivi hipermedijski sistemi, njihova zgradba in primeri uporabe opisani v tretjem poglavju.

2.3 Spletni izobraževalni sistemi

V zadnjih desetih letih, predvsem zaradi bliskovite širitve računalniških omrežij in interneta ter s pojavitvijo svetovnega spleta, so izobraževalni sistemi dobili novo dimenzijo. Spletni izobraževalni sistemi so zelo priljubljeni prav zaradi novih možnosti, ki jih prinašajo v primerjavi s tradicionalnimi sistemi, tako hipermedijskimi kot inteligentnimi učnimi sistemi. Svetovni splet je namreč zelo primeren medij za izobraževalne sisteme, ker deluje na odprtih standardih, ima zelo nizke stroške izdelave in tudi vzdrževanja vsebin, podpira izdelavo dinamične multimedijske vsebine, za njegovo prikazovanje pa so na voljo brezplačni, a precej močni spletni brskalniki [Mann 1999]. Poleg tega ga odlikuje enostavnost uporabe, je neodvisen od platforme, omogoča pa tudi različne možnosti za komunikacijo in sodelovanje ter enostavnejši dostop na daljavo.

Vendar pa svetovni splet v osnovi ni bil zgrajen v izobraževalne namene, zato se kažejo problemi pri kognitivnih in pedagoških vprašanjih spleta. Dajati informacije namreč še ne pomeni izobraževati in imeti dostop do informacij ne pomeni učenja [Fernandez 1997]. Tako se kljub velikim prednostim, ki jih nudi splet, pojavljajo problemi, kot so slaba orientacija, pomanjkanje planiranja in vodstva, preobremenjenost z informacijami ter podobni, ki lahko resno ogrozijo uporabnost spleta kot izobraževalnega orodja.

Pedagoška uporaba spleta se razvija v dveh smereh [Brusilovsky 1998a]. Na eni strani so zaprte zbirke učnega materiala (angl. closed corpus), kjer se spletna tehnologija uporablja zgolj zaradi njenih hipermedijskih zmogljivosti in možnosti dostave materiala na daljavo. Drugo stran pa predstavljajo odprte zbirke (angl. open corpus), kjer se izkoriščajo ogromne količine informacij, ki so na voljo preko interneta, ne glede na njihov osnovni namen (ali so bile objavljene v izobraževalne namene). Čeprav različni materiali v spletu v svoji osnovi niso nujno namenjeni rabi v izobraževanju, to še ne pomeni, da nimajo nobene izobraževalne uporabe.

Svetovni splet je po svoji naravi hipertekstoven (hipermedijski), zato so hipermedijski sistemi zaradi svoje osnovne zgradbe in ideje najprimernejši za spletno uporabo. Zaradi tega tudi ne preseneča, da hipermedijske sisteme, ki so po naravi odprti sistemi, ponavadi samodejno povezujemo s svetovnim spletom. Tudi prilagodljivi hipermedijski sistemi so zelo blizu ideji hiperteksta in primerni za spletno uporabo. Nasprotno pa so inteligentni učni sistemi po naravi zaprti sistemi, zato predstavlja prenos v svetovni splet velik poseg in predelavo sistema in tudi ni primeren za vse inteligentne sisteme.

Kadar gre za uporabo prilagodljivih hipermedijskih sistemov in inteligentnih učnih sistemov v svetovnem spletu, se oboji zbližajo, razlike med njimi so manjše in govorimo lahko o eni sami vrsti sistemov (glej sliko 2.1). Take sisteme bomo imenovali *prilagodljivi spletni izobraževalni sistemi* (angl. *adaptive educational systems on the Web*). Na prilagodljive spletne izobraževalne sisteme sicer lahko gledamo kot na spletno različico inteligentnih učnih sistemov ali prilagodljivih hipermedijskih sistemov, vendar pa ravno spletni kontekst močno vpliva na zasnovo in izvedbo teh sistemov [Brusilovsky 1998d]. Zato jih ponavadi obravnavamo kot posebno vrsto sistemov. Poleg tega je pri spletnem izobraževanju prilagajanje sistema še posebej pomembno, ker sisteme uporablja veliko večje število različnih uporabnikov, ki na sistemih delajo in se učijo sami in nimajo pomoči kolegov ali učitelja, kot pri učenju v razredu [Brusilovsky 1998d].

2.4 Problem navigacije v hiperprostoru

Navigacija v osnovnem pomenu se ukvarja s problemom, kako na učinkovit način doseči določeno točko ali množico točk [Kommers 1996]. V hiperprostoru so to vozlišča oziroma hipermedijske strani. Pri navigaciji si pomagamo z različnimi orientacijskimi znaki, ki naj bi omogočali predvsem tri stvari [Ebersole 1997]: ugotoviti položaj trenutnega vozlišča glede na celotno strukturo hiperprostora, rekonstruirati pot, ki je pripeljala do tega vozlišča, ter razlikovati med različnimi možnimi potmi, ki vodijo iz trenutnega vozlišča.

Največji problem navigacije v hiperprostoru je tako imenovan problem izgubljenega v hiperprostoru (angl. lost in hyperspace), ki se pokaže že pri manjših hipermedijskih dokumentih (torej pri manjšem številu vozlišč) [Nielsen 1990]. Opišemo ga lahko kot kombinacijo dveh problemov [Nielsen 1990]. Prvi je ponovno lociranje že najdene informacije, ko uporabnik poprej najdene strani ne zna ponovno poiskati. Drugi problem pa je slaba orientacija v hiperprostoru, ko uporabnik med njegovim preiskovanjem izgubi občutek, kje v strukturi se nahaja.

Na učenje lahko gledamo tudi kot na navigacijo po zbirki znanja. Vendar se koncept navigacije pri izobraževalnih sistemih rahlo razlikuje od tistega pri informacijskih sistemih, saj je namen sistema ne le zagotoviti, da uporabnik vidi določeno količino ustreznih informacij, temveč tudi, da se jih nauči [Linard 1995].

Uporabniki imajo različne zmožnosti, preference in spretnosti. Te osebne razlike so zelo pomembne in močno vplivajo tudi na navigacijo, ker vsi uporabniki nimajo enakih navigacijskih zmožnosti. Tako na primer slabe prostorske zmožnosti močno omejujejo uporabnika pri navigaciji po velikih datotečnih strukturah [Benyon 1997]. Seveda je z izkušnjami mogoče premagati tudi te pomanjkljivosti, neizkušeni začetniški uporabniki pa pri navigaciji še vedno potrebujejo veliko pomoči sistema. Pri navigaciji v splošnem velja, da manj ko morajo uporabniki paziti, kje se nahajajo in kaj naj naredijo naslednje, bolj se lahko posvetijo vsebini predstavljene strani in s tem izboljšajo učenje [Brickell 1993]. Če zmanjšamo motnje, kot so slaba orientacija in nepoznavanje okolice, lahko izboljšamo razumevanje učnega materiala.

3. Prilagodljivi hipermediji

Prilagodljivi hipermediji pomenijo novo smer raziskav na področju prilagodljivih in na modelu uporabnika temelječih vmesnikov [Brusilovsky 1998a, Eklund 1998b]. Njihov glavni cilj je povečati uporabnost hipermedijev s pomočjo poosebljanja.

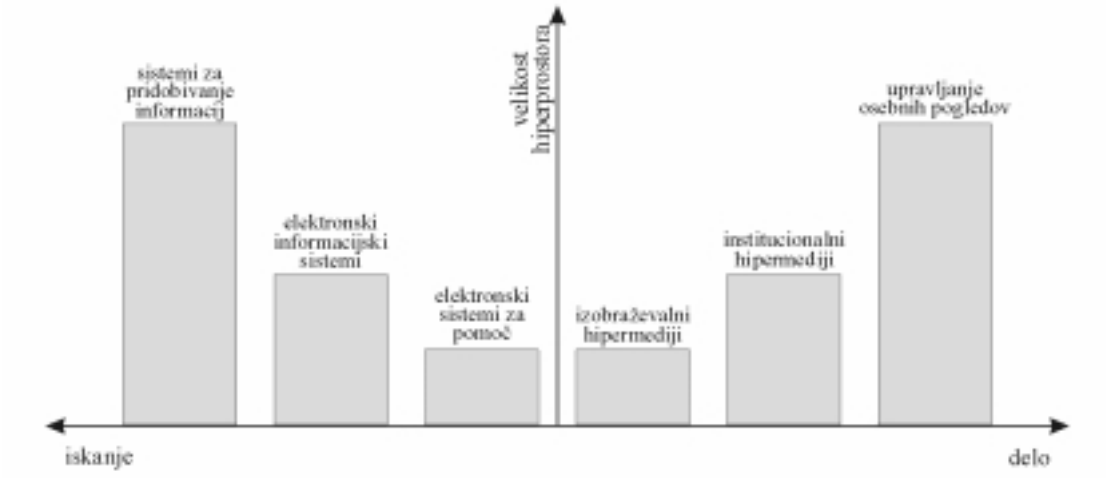
3.1 Področja uporabe prilagodljivih hipermedijev

V splošnem so prilagodljivi hipermediji koristni na vseh področjih, kjer je pričakovati različne uporabnike (z različnimi cilji in znanjem) ali pa je hiperprostor razmeroma velik. Njihovo uporabo lahko zasledimo predvsem na naslednjih šestih področjih [Brusilovsky 1998a]:

- *Hipermedijski sistemi za iskanje in izbiranje informacij* (angl. *information retrieval hypermedia systems*), kot so na primer iskalniki, delujejo nad zelo velikim hiperprostorom in zato sistem sam sproti računa povezave. S tega področja so na primer sistemi Adaptive HyperMan [Mathé 1994], Hyperflex [Kaplan 1998] ali WebWatcher [Armstrong 1995].
- *Elektronski informacijski sistemi* (angl. *on–line information systems*) nudijo referenčni dostop do informacij uporabnikom z različno stopnjo znanja o temi. Primeri takih sistemov so elektronske dokumentacije ali enciklopedije, mednje pa sodijo na primer sistemi Hypadapter [Hohl 1998], MetaDoc [Boyle 1998] in PUSH [Höök 1996].
- *Elektronski sistemi za pomoč* (angl. *on–line help systems*) nudijo na primer kontekstno odvisno pomoč. So zelo podobni elektronskim informacijskim sistemom, le da gre tu za manjši hiperprostor. Mednje bi lahko šteli sistema HyPLAN [Fox 1993] in ORIMUHS [Encarnação 1995].
- *Izobraževalni hipermediji* (angl. *educational hypermedia*) so bolj tradicionalni sistemi in spadajo med najpogosteje raziskovana področja uporabe prilagodljivih hipermedijev. Tudi zato je največ sistemov ravno s področja izobraževalnih hipermedijev. Nekatere od njih bomo podrobneje opisali v razdelku 3.5.

- *Institucionalni informacijski sistemi* (angl. *institutional information systems*) nudijo elektronsko vse informacije, potrebne za podporo delovanju neke institucije, kot je na primer bolnišnica. Take sisteme uporablja veliko ljudi pri svojem vsakdanjem delu, kjer lahko dostopajo le do določenega dela hiperprostora. Ti sistemi so izrazito delovno usmerjeni. Najznačilnejši sistem te skupine je prav gotovo Hynecosum [Vassileva 1996].
- *Sistemi za upravljanje osebnih pogledov v informacijskem prostoru* (angl. *systems for managing personalized views in information spaces*) so tudi izrazito delovno usmerjeni sistemi. Delujejo nad izredno velikim hiperprostorom, kjer mnogi uporabniki potrebujejo pri svojem vsakdanjem delu dostop le do omejene podmnožice tega hiperprostora. Da bi se obvarovali pred zapletenostjo celotnega hiperprostora, si uporabniki določijo oseben pogled na celoten hiperprostor, ki je lahko različen za vsak njihov cilj ali zanimanje. Primer takega sistema je sistem Basar [Thomas 1995].

Vseh šest navedenih področij se medsebojno ne izključuje in tudi prehodi med njimi niso ostro zarisani. Različni sistemi so si tudi v nekem pogledu podobni in rešujejo podobne probleme, pa naj gre za velikost hiperprostora ali pa za usmerjenost sistema. Slika 3.1 prikazuje navedena področja uporabe prilagodljivih hipermedijev, kjer so sosednja področja medsebojno podobna.



Slika 3.1

Povezanost prilagodljivih hipermedijskih sistemov

Hipermedijski sistemi za pridobivanje informacij in elektronski informacijski sistemi so usmerjeni bolj v iskanje, institucionalni informacijski sistemi in sistemi za upravljanje osebnih

pogledov v informacijskem prostoru pa so usmerjeni v delo. Elektronski sistemi za pomoč in izobraževalni hipermediji se nahajajo na sredini. Velikost hiperprostora, prikazana s sivim pravokotnikom, raste od razmeroma majhnega v sredini do ogromnega na obeh koncih.

V disertaciji se bomo osredotočili na uporabo prilagodljivih hipermedijskih sistemov v izobraževalne namene. Pri izobraževalnih hipermedijih imamo sicer razmeroma majhen in omejen hiperprostor, to je učni material za določen predmet, vendar je razpon uporabnikovega poznavanja tega prostora (učne domene) lahko zelo velik. Najpomembnejša lastnost uporabnika je njegovo znanje o učni domeni [Brusilovsky 1998a]. Znanje različnih uporabnikov se lahko zelo razlikuje, znanje posameznega uporabnika pa lahko zelo hitro raste. Tako se lahko ista stran, ki je začetniku popolnoma nerazumljiva, zdi naprednejšemu učencu nekoristna in dolgočasna. Razen tega potrebujejo začetniki izdatnejšo pomoč pri navigaciji skozi učni material, saj o predmetu ne vedo skoraj nič, vse povezave pa vodijo na strani z njim popolnoma novo vsebino. Ker je cilj vsakega učenca, da predela in se nauči ves učni material oziroma večji njegov del, je poglavitni namen prilagajanja predvsem usmerjanje učenca (s podporo pri navigaciji) ter zmanjšanje njegove kognitivne obremenitve.

3.2 Definicija prilagodljivih hipermedijskih sistemov

Za naše potrebe bomo povzeli definicijo prilagodljivih hipermedijev, ki jo je podal eden od pionirjev tega področja Peter Brusilovsky [Brusilovsky 1998a]. Po njej prilagodljivi hipermedijski sistemi zajemajo vse tiste hipertekstovne in hipermedijske sisteme, ki odražajo določene značilnosti uporabnika v modelu uporabnika in ta model uporabljajo pri prilagajanju vizualnih ali funkcionalnih delov sistema danemu uporabniku.

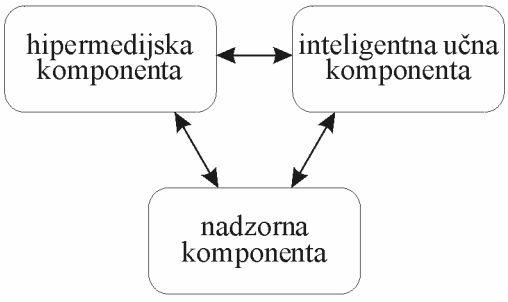
Za prilagodljive hipermedijske sisteme tako štejemo vse tiste sisteme, ki imajo naslednje štiri lastnosti [Eklund 1998b]:

- temeljijo na hipertekstu ali hipermediji,
- vzdržujejo eksplicitni model uporabnika, ki beleži individualne lastnosti uporabnika,
- vključujejo model domene, ki je sestavljen iz množice elementarnih delcev znanja in njihovih medsebojnih relacij v informacijskem prostoru, ter
- so zmožni spreminjati nekatere vizualne ali funkcionalne dele sistema na podlagi informacij iz modela uporabnika.

Prilagodljivi hipermedijski sistemi so torej v osnovi hipermedijski sistemi, ki pa imajo vgrajene določene inteligentne pristope. Njihova glavna ideja je omogočiti uporabniku polno funkcionalnost tradicionalnih hipermedijev z dodatno možnostjo, da se vsebina prilagaja uporabnikovim individualnim potrebam. Različni uporabniki imajo namreč različne cilje in znanje, zanimajo jih različni deli informacij na hipermedijskih straneh, pri navigaciji pa uporabljajo različne povezave. Zato jih tudi sistem različno obravnava. S spreminjanjem prikazanih informacij in povezav se sistem prilagaja ciljem in znanju vsakega posameznega uporabnika. Pomaga mu tudi pri navigaciji z omejevanjem prostora za preiskovanje, predlaganjem najprimernejših povezav ali z dinamičnimi komentarji k povezavam.

3.3 Struktura prilagodljivih hipermedijskih sistemov

Če povzamemo prejšnji razdelek, prilagodljiv hipermedijski sistem temelji na hipermedijih, vključuje (ter seveda tudi vzdržuje) model domene in model uporabnika ter se je zmožen na nek način prilagajati posameznemu uporabniku glede na model uporabnika. Torej gre v osnovi za hipermedije, ki jih nadgrajujejo določene inteligentne tehnike, kar razkriva tudi njihova zgradba. Osnovna zgradba, ki jo prikazuje slika 3.2, zajema tri komponente: hipermedijsko, inteligentno učno ter nadzorno.



Slika 3.2
Zgradba prilagodljivih hipermedijskih sistemov

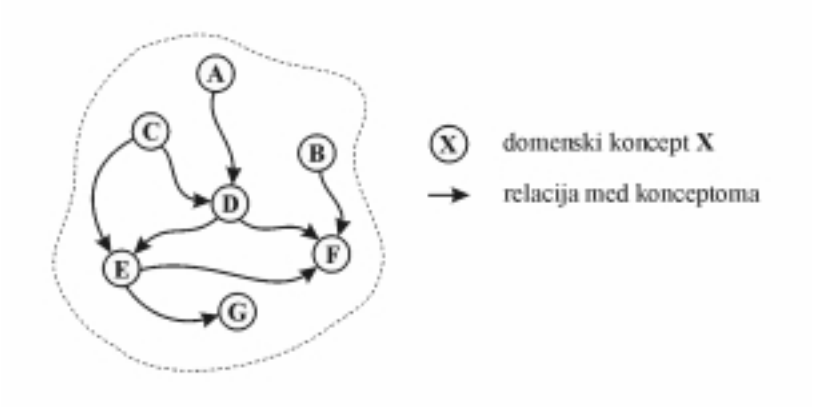
Hipermedijska komponenta skrbi za organizacijo domenskega hiperprostora, njegovo predstavitev in navigacijo v njem. Inteligentna učna komponenta, ki posnema strukturo inteligentnih učnih sistemov, skrbi za vse inteligentne akcije sistema (prilagajanje sistema). Obe pa povezuje nadzorna komponenta, ki usklajuje njuno delovanje.

Da se sistem lahko prilagaja posameznemu uporabniku, mora poznati tako učno domeno kot vsakega uporabnika, njegovo znanje in napredovanje pri učenju. Zato sta za izvedbo uspešnega in uporabnega prilagajanja, poleg modela prilagajanja, ključnega pomena tudi model domene in model uporabnika [Wu 2000]. Prvi omogoča sistemu poznavanje učne domene in s tem, kaj sistem uči, drugi pa poznavanje uporabnika in s tem, koga sistem uči.

3.3.1 Predstavitev znanja domene

Pri modeliranju znanja domene se večina sistemov naslanja na konceptni model domene, kjer je učna domena razgrajena na množico osnovnih konceptov [Brusilovsky 1998a, Eklund 1997]. Tu izraz koncept pomeni temeljni delec znanja v dani domeni učenja.

Najenostavnejša oblika modela domene je množica domenskih konceptov, ki med seboj niso povezani. Več informacij zajema model domene v obliki mreže ali grafa domene, kjer vozlišča ustrezajo konceptom domene, povezave pa odražajo relacije med njimi. Tak graf domene je predstavljen na sliki 3.3. V kompleksnejših modelih domene najdemo več tipov povezav in konceptov, z njimi pa so v model domene vgrajene tudi semantične relacije med koncepti.



Slika 3.3
Konceptni graf domene

Vsak domenski koncept je ponavadi opisan na eni ali več hipermedijskih straneh. Tako vsakemu vozlišču mreže domene ustreza eno ali več vozlišč v hiperprostoru, povezave med vozlišči mreže domene pa tvorijo poti med vozlišči v hiperprostoru. Zato je struktura celotnega hiperprostora podobna pedagoški strukturi znanja domene.

Graf domene je vrsta semantične mreže. Tak model predstavitve znanja domene nam je precej blizu, saj strukturira znanje podobno kot človeški spomin. Zato se mrežni model znanja

uporablja tudi za predstavitev organizacije znanja v človeškem spominu [Woodhead 1991]. Semantične mreže so se razvile iz podobnih modelov v psihologiji pomnjenja in zaznavanja ter spadajo med formalne modele predstavitve znanja, imajo pa tudi psihološke ekvivalente. Semantika igra pomembno vlogo, saj je oddaljenost vozlišč v mreži določena s pomensko podobnostjo vozlišč.

3.3.2 Modeliranje uporabnika

Model uporabnika predstavlja osnovo za prilagajanje sistema posameznemu uporabniku, saj hrani vse potrebne informacije o uporabniku. Brez informacij o uporabnikih sistem ne bi ločil med posameznimi uporabniki in bi vse uporabnike obravnaval enako.

Popoln model uporabnika bi moral vključevati vse lastnosti uporabnika, njegovo obnašanje in znanje, ki lahko vplivajo na uporabnikovo učenje in učinkovitost [Wenger 1987]. Sestaviti tak model je zelo zapletena naloga celo za ljudi, zato se v računalniških sistemih ponavadi uporabljajo močno poenostavljeni modeli.

Pri modeliranju uporabnika so pomembne informacije o uporabniku, ki so vključene v model, način pridobitve teh informacij, njihova predstavitev v sistemu ter postopek oblikovanja in posodabljanja modela.

Na informacije o uporabniku, ki jih zbira sistem, lahko gledamo iz treh vidikov: kot odvisne in neodvisne od domene učenja, kot znanje (uporabnikovo znanje) in vmesnik (uporabnikove preference) ter kot statične in dinamične lastnosti. Statični del modela uporabnika je nespremenljiv skozi učni proces [Jameson 1999] in zajema uporabnikove osebne značilnosti (starost, spol, stopnjo), uporabnikove zmožnosti (predhodno znanje, spretnosti, sposobnosti), uporabnikove preference in podobno. Dinamični del modela uporabnika pa se med učnim procesom spreminja, saj vključuje informacije v zvezi z uporabnikovo interakcijo s sistemom [Jameson 1999]. To zajema uporabnikovo znanje, koncepte in izkušenost, način učenja, motivacijo, trenutne cilje, plane in domneve, izpeljane učne aktivnosti, dosežene cilje in podobno.

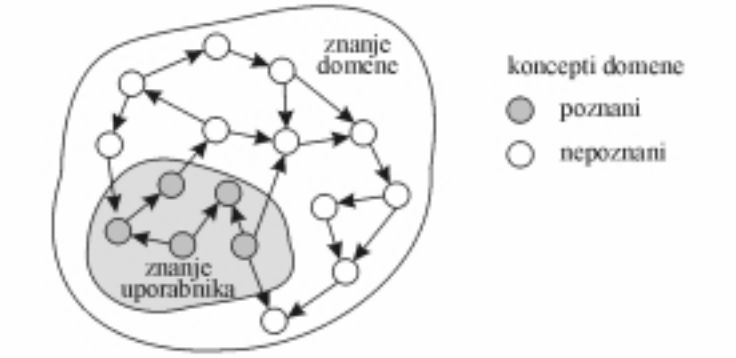
Potrebne informacije o uporabniku lahko pridobimo na več načinov. Statične informacije se zberejo na začetku učnega procesa, najpogosteje s pomočjo vprašalnikov in različnih testov, saj jih mora podati uporabnik sam. Dinamične informacije pa lahko pridobimo z nadzorom uporabnikovih akcij, testi za preverjanje znanja, vajami in nalogami, lahko pa jih neposredno

poda tudi uporabnik. Te informacije se nenehno zbirajo med celotnim učnim procesom in jih sistem uporabi tudi za posodabljanje modela uporabnika.

Za predstavitev informacij o uporabniku in izgradnjo modela uporabnika se uporabljajo različne metode. Te zajemajo [Jameson 1999] metode umetne inteligence in strojnega učenja (Bayesove mreže, učenje pravil, učenje verjetnosti, učenje s primeri, logične metode, hevristične metode), prekrivne metode, stereotipne metode, druge splošne tehnike in principe (razpoznavanje plana), posebej razvite računske postopke ter posebej razvita kvalitativna pravila in postopke. Podobne metode se uporabljajo tudi za oblikovanje in posodabljanje modela uporabnika.

V izobraževalnih sistemih je ena najpomembnejših lastnosti uporabnika njegovo znanje [Brusilovsky 1998a]. Zato se večina modelov uporabnika omeji na modeliranje znanja uporabnika. Ta ponavadi sloni na modelu domene. V nadaljevanju bomo opisali dve enostavni metodi modeliranja uporabnika, ki se pogosto uporabljata v izobraževalnih sistemih. Nekatere kompleksnejše metode so opisane v četrtem poglavju.

Prekrivni model. Znanje uporabnika je najpogosteje predstavljeno s prekrivnim modelom (angl. overlay model), ki temelji na strukturnem modelu domene [Brusilovsky 1998a]. Uporabnikovo znanje je predstavljeno kot podmnožica konceptov domene. Za vsak domenski koncept hrani prekrivni model neko vrednost, ki je ocena uporabnikovega poznavanja tega koncepta [Eklund 1996]. Ta vrednost je lahko dvojiška (poznan ali nepoznan koncept), kakovostna mera (dobro, povprečno, slabo poznavanje koncepta) ali pa količinska mera (verjetnost poznavanja koncepta). Prekrivni modeli so močni in upogljivi, saj lahko neodvisno merijo uporabnikovo poznavanje različne snovi. Slika 3.4 shematično prikazuje prekrivni model preko modela domene.



Slika 3.4

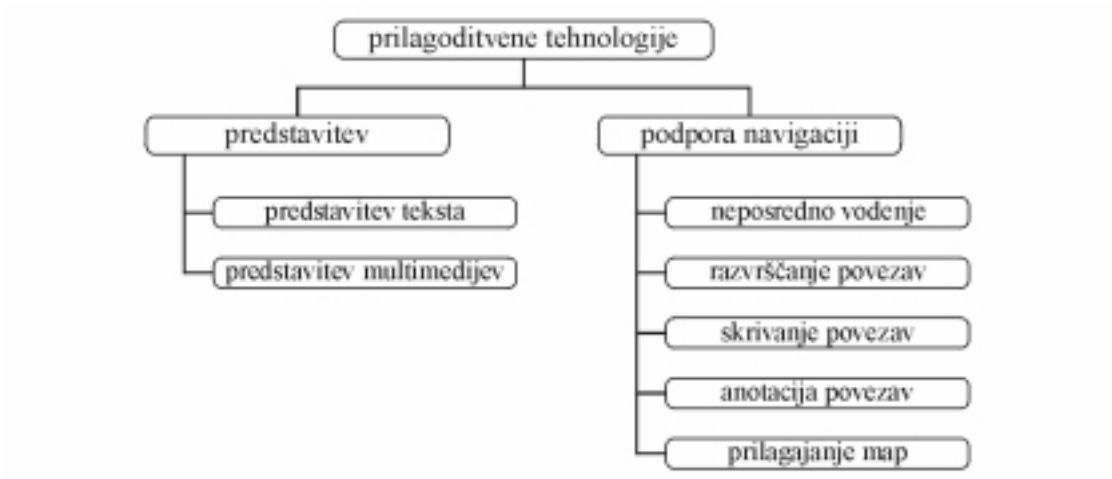
Prekrivni model uporabnika

Stereotipni model. Stereotipni model (angl. stereotype model) za predstavitev uporabnikovega znanja je enostavnejši in splošnejši od prekrivnega modela. Lažje je nastaviti začetno vrednost in vzdrževati model. Vendar pa je zato šibkejši in manj elastičen. Stereotipni model razlikuje med več tipičnimi uporabniki (kot so začetnik, naprednejši uporabnik ali poznavalec), ki imajo vnaprej predpisane vrednosti poznavanja posameznih konceptov [Eklund 1996]. Te vrednosti med seboj niso povsem neodvisne. Vsak uporabnik je dodeljen enemu ali več stereotipom in s tem se uporabniku pripišejo tudi ustrezne vrednosti poznavanja konceptov.

Seveda lahko različne predstavitve uporabnikovega znanja medsebojno kombiniramo [Eklund 1996]. Tako se pogosto uporablja stereotipni model za določitev začetnega stanja modela uporabnikovega znanja, za njegovo posodabljanje pa se uporablja prekrivni model. To zagotavlja natančnejše modeliranje in boljši izkoristek zbranih informacij.

3.4 Prilagoditvene tehnologije

V prvem razdelku tega poglavja smo opisali, kje (na katerih področjih) se prilagodljivi hipermediji izkažejo za uporabne. Prejšnji razdelek med drugim razjasni, katere lastnosti uporabnika se lahko uporabljajo kot vir prilagajanja oziroma katerim lastnostim uporabnika lahko sistem prilagodi svoje obnašanje. V tem razdelku pa se bomo ukvarjali s tem, kaj lahko v sistemu prilagajamo oziroma katere lastnosti sistema so lahko različne za različne uporabnike.



Slika 3.5

Prilagoditvene tehnologije

Ločimo sedem različnih načinov prilagajanja hipermedijev, ki jih lahko razdelimo na dve osnovni skupini, to sta prilagodljiva predstavitev in prilagodljiva podpora navigaciji [Eklund 1996, Brusilovsky 1998a]. Različne načine prilagajanja hipermedijev imenujemo prilagoditvene tehnologije [Brusilovsky 1998a]. Slika 3.5 prikazuje vseh sedem prilagoditvenih tehnologij in njihovo razporeditev v skupini.

Obe skupini tehnologij si med seboj nista nasprotujoči, temveč sta tesno povezani in se lepo dopolnjujeta. V nekaterih primerih pa se celo prekrivata, saj lahko na primer prilagajanje povezav uporabljamo tako za prilagodljivo predstavitev (spreminjanje množice vidnih povezav) kot za prilagodljivo navigacijo (spreminjanje postavitve povezav, da nudimo vodenje) [Brusilovsky 1996a].

Seveda se vse prilagoditvene tehnologije izvajajo na podlagi informacij o posameznem uporabniku, ki so zapisane v modelu uporabnika.

3.4.1 Prilagodljiva predstavitev

Prvo skupino prilagoditvenih tehnologij, ki delujejo na nivoju prilagajanja vsebine hipermedijske strani, imenujemo *prilagodljiva predstavitev* (angl. *adaptive presentation*). Tu se informacija, ki jo vsebuje hipermedijsko vozlišče ali stran, predstavi na prilagodljiv način tako, da se spreminja glede na detajle, razlago, uporabo medijev ali število povezav [Eklund 1996].

Uporabnikom z različnimi modeli uporabnika se tako ista stran prikaže z različno vsebino, saj je sama vsebina strani dinamično sestavljena šele tik pred prikazom. Na ta način se vsebina hipermedijske strani prilagodi trenutni stopnji uporabnikovega znanja, ki je zapisana v modelu uporabnika. Na ta način lahko začetniki, ki so novi v dani učni domeni in si učni material ogledujejo prvič, dobijo več njim primerne dodatne razlage, obremenjujoče podrobnosti pa so jim začasno skrite. Naprednejši uporabniki, ki so že seznanjeni z učno domeno in jim učni material ni več tuj, pa ne dobijo toliko osnovne razlage, temveč bolj zgoščene, podrobne in poglobljene informacije [Brusilovsky 1994].

V hipermedijskih sistemih vsebine strani ne sestavlja le golo besedilo, temveč tudi množica različnih multimedijskih elementov. Zato lahko ločimo *predstavitev teksta* in *predstavitev multimedijev*. Seveda se dinamična predstavitev teksta v realnih sistemih veliko pogosteje uporablja, deloma tudi zaradi svoje enostavnosti in učinkovitosti, in tako spada med najbolj raziskane prilagoditvene tehnologije [Brusilovsky 1998a].

3.4.2 Prilagodljiva podpora navigaciji

Druga skupina prilagoditvenih tehnologij je *prilagodljiva podpora navigaciji* (angl. *adaptive navigation support*) in deluje na nivoju povezav oziroma poti v hiperprostoru. Glavna ideja je pomagati uporabniku najti pot v hiperprostoru, kar naredimo s prilagajanjem načina predstavitve povezav glede na cilje, znanje ali druge značilnosti uporabnika [Brusilovsky 1998a]. Ker je z uporabnikove perspektive učenje s hipermedijskimi sistemi v glavnem le stvar navigacije po zbirki znanja (po učnem materialu) [Linard 1995], so tehnologije za podporo navigaciji še posebej pomembne v izobraževalnih sistemih.

Glede na način prilagajanja povezav ločimo pet različnih tehnologij, ki jih prikazuje tudi slika 3.5.

Neposredno vodenje (angl. *direct guidance*) je najenostavnejša tehnologija. Izhaja iz inteligentnih učnih sistemov, kjer sistem sam neposredno usmeri uporabnika na naslednjo najprimernejšo stran. Največkrat se za to uporablja poseben gumb, imenovan “naprej”. Ponavadi se tehnologija neposrednega vodenja uporablja v kombinaciji z drugimi tehnologijami, ker sama ne nudi nobenih drugih možnosti uporabniku, ki ne želi slediti predlagani povezavi.

Namen tehnologije *prilagodljivega razvrščanja povezav* (angl. *adaptive reordering of links*) je sortiranje povezav z dane strani glede na model uporabnika ali po določenem kriteriju, ki je koristen za uporabnika. Pri tem se najprimernejše oziroma najpomembnejše povezave nahajajo na vrhu. Tehnologija razvrščanja povezav je uporabna le pri kontekstno neodvisnih povezavah, saj kontekstne povezave ne dovoljujejo prerazporejanja. Največji problem te tehnologije je nestabilen vrstni red povezav, ki je lahko še posebej moteč za začetnike.

Prilagodljivo skrivanje povezav (angl. *adaptive hiding of links*) je najpogosteje uporabljana tehnologija podpore navigaciji. Njen namen je omejiti navigacijski prostor s skrivanjem povezav do nepomembnih strani. Med nepomembne štejemo tiste strani, ki niso neposredno povezane z uporabnikovim trenutnim ciljem ali pa prikazujejo material, na katerega uporabnik še ni pripravljen. S skrivanjem povezav sistem ščiti uporabnika pred zapletenostjo neomejenega hiperprostora in hkrati zmanjša njegovo kognitivno obremenitev. Povezave lahko skrivamo na tri načine [De Bra 1998]. Pri čistem skrivanju povezave ostanejo aktivne, a se ne ločijo od ostalega besedila. Vse na ta način skrite povezave lahko uporabimo, vendar moramo točno vedeti, kje se nahajajo. Odstranjevanje povezav pomeni, da so povezave neaktivne in se ne ločijo od ostalega besedila. Tretji način pa je onemogočanje povezav, kjer so povezave sicer

še vedno označene, a so neaktivne in jih zato ne moremo uporabljati. Zadnja možnost je ponavadi precej moteča za uporabnike, zato je večinoma nezaželena.

Ideja *prilagodljive anotacije povezav* (angl. *adaptive annotation of links*) je označevanje povezav s komentarji, ki lahko uporabniku povedo več o trenutnem stanju vozlišč, do katerih vodijo te povezave. Taki komentarji so lahko tekstovni ali pa v obliki različnih ikon, barv, velikosti ali tipa pisave in podobno. V tradicionalnih hipermedijih se uporablja statična anotacija, ki je neodvisna od posameznega uporabnika. Dinamična anotacija, ki nudi prilagodljivo podporo navigaciji, pa se opira na model uporabnika in se dinamično spreminja glede na trenutne informacije iz tega modela. Najpreprostejši način anotacije temelji na zgodovini in označuje povezave do že obiskanih vozlišč. Tak način anotacije, ki ga uporablja tudi večina spletnih brskalnikov, je zelo enostaven, a se je izkazal za zelo uporabnega. Prilagodljiva anotacija se je pokazala za posebej učinkovito v izobraževalnih hipermedijih [Eklund 1998b], saj se lahko uporablja na vseh vrstah povezav, podpira stabilen vrstni red povezav in se hkrati izogne tudi problemom z nepravilnimi mentalnimi mapami.

Prilagajanje map (angl. *map adaptation*) zajema različne načine prilagajanja oblike globalnih in lokalnih hipermedijskih map, ki jih sistem prikaže uporabniku. Tu gre za dinamično spreminjanje oblike ali strukture mape glede na model uporabnika. Najpogosteje se uporablja grafična predstavitev domenskih konceptov ali tako imenovane konceptualne mape [Pilar 1998].

3.4.3 Učinkovitost prilagoditvenih tehnologij

Čeprav je bilo razvitih že precejšnje število prilagodljivih hipermedijskih sistemov [Brusilovsky 1998a], je bilo izpeljanih bolj malo raziskav o uporabnosti teh sistemov in posameznih prilagoditvenih tehnologij [Mann 1999]. Pri vseh obstoječih raziskavah je bilo glavno vodilo ugotoviti, kako različne prilagoditvene tehnologije vplivajo na uporabnikovo razumevanje vsebine in navigacijo v hiperprostoru oziroma kakšna je njihova uporabnost in učinkovitost. V nadaljevanju si bomo pogledali ugotovitve nekaterih študij, ki so proučevale uporabnost posameznih tehnologij.

Tehnologija prilagodljive predstavitve. Najobširnejše ovrednotenje te tehnologije sta opravila Boyle in Encarnacion s sistemom MetaDoc [Boyle 1998]. Izvedla sta več testiranj, kjer sta primerjala navaden hipertekst, hipertekst z možnostjo raztezanja besedila (angl. stretchtext) ter prilagodljiv hipertekst z možnostjo raztezanja besedila. Upoštevala sta tako razumevanje

prebranega besedila kot navigacijo. Rezultati so pokazali, da prilagajanje vsebine z metodo raztezanja besedila občutno izboljša razumevanje prebranega besedila in izboljša uporabnikovo storilnost (zmanjša čas preiskovanja), a ne vpliva na njegovo navigacijo. Podrobnosti o celotni raziskavi in njenih rezultatih najdete v [Boyle 1998].

Tehnologija prilagodljivega razvrščanja povezav. Prvo ocenitev prilagodljive podpore navigaciji z razvrščanjem povezav je opravil Kaplan s sodelavci na sistemu HYPERFLEX [Kaplan 1998]. Gre za dve študiji na manjšem vzorcu uporabnikov, ki sta bili osredotočeni na učinkovitost lociranja informacij. V njih so primerjali prilagajanje glede na trenutno vozlišče, prilagajanje glede na trenutni cilj in obe vrsti prilagajanja skupaj. Rezultati uporabe prilagodljive navigacije so pokazali, da razvrščanje povezav lahko izboljša uporabnikovo storilnost pri iskanju informacij, zmanjša čas iskanja informacije in število pregledanih vozlišč. Podrobnosti o obeh študijah so v [Kaplan 1998].

Tehnologija prilagodljivega skrivanja in anotacije povezav. Ovrednotenje prilagodljive podpore navigaciji s skrivanjem in anotacijo povezav sta izpeljala Brusilovsky in Pesin na sistemu ISIS–Tutor [Brusilovsky 1994, Brusilovsky 1998c]. Primerjala sta čas za izvedbo naloge in število navigacijskih korakov v navadnem neprilagodljivem sistemu, sistemu z anotacijo povezav ter sistemu z anotacijo in skrivanjem povezav. Iz rezultatov je razvidno, da se pri prilagodljivi različici sistema zmanjša število navigacijskih korakov in obiskov istih strani, zmanjša se čas učenja materiala in poveča njegovo razumevanje. Med obema načinoma prilagajanja pa ni opaznejših razlik. Natančnejši podatki o rezultatih raziskave so v [Brusilovsky 1998c].

Tehnologija prilagodljivega skrivanja povezav. Raziskavo uporabnosti prilagodljive navigacije s skrivanjem povezav je izvedel tudi Mann v okviru svoje doktorske disertacije na primeru učenja jezika HTML [Mann 1999]. V njej je ugotovil, da prilagodljivo skrivanje povezav pomaga uporabnikom k boljšemu razumevanju vsebine, večjemu učinku učenja in zato tudi k boljšim rezultatom končnega testa. Ta tehnologija hkrati prepreči kognitivno preobremenjenost in slabo orientacijo v hiperprostoru, ki uporabnika zmoti in zmede. Podrobnosti o raziskavi in rezultatih so objavljene v [Mann 1999]. Vendar pa je pri prilagodljivem skrivanju zelo pomemben tudi način skrivanja povezav, predvsem pri povezavah na začetni indeksni strani. Način, pri katerem so vse povezave vidne, a po potrebi onemogočene (vendar tudi ustrezno označene), se je v nekaterih primerih izkazal za boljšega kot popolno odstranjevanje povezav [Calvi 2000], saj uporabniku omogoča pogled na celotno strukturo povezav in mu s tem olajša izgradnjo ustreznega mentalnega modela. Onemogočanje povezav

prispeva k hitrejšemu iskanju informacij (manjše število skupnih zadetkov) in visoki stopnji ujemanja (zaupanje, sprejemanje in strinjanje uporabnika s predlogi sistema).

Tehnologija prilagodljive anotacije povezav. Raziskavo o učinkovitosti prilagodljive navigacije z anotacijo povezav sta na sistemu InterBook izvedla Eklund in Brusilovsky [Eklund 1998a]. Rezultati so pokazali, da prilagodljiva navigacija lahko izboljša razumevanje tistim uporabnikom, ki nimajo veliko izkušenj z navigacijo v hiperprostoru in ki so pripravljeni sprejeti predloge sistema. Anotacija povezav jim omogoča določeno varnost tudi pri uporabi nezaporednih poti in se zato ne omejujejo le na strogo zaporedno pot skozi material z uporabo gumba “naprej”. Podrobneje so rezultati opisani v [Brusilovsky 1998b].

Tehnologija neposrednega vodenja. Raziskava uporabe prilagodljive navigacije z neposrednim vodenjem je pri projektu CAIN [Lamas 2000] pokazala, da se z uporabo neposrednega vodenja razumevanje lahko poveča skoraj za tretjino, hkrati pa se je tudi zmanjšal čas izvedbe naloge. Vendar pa uporabnikom, ki so vešči v svetovnem spletu ali v domeni učenja, sistem ne nudi nobenih prednosti. Tehnologija neposrednega vodenja je torej primernejša za začetnike, ki še niso dovolj spretni in se ne znajdejo dobro v spletu ali učni domeni.

Uporaba in učinkovitost različnih prilagoditvenih tehnologij sta bili obdelani v raznih študijah. Rezultati se sicer precej razlikujejo, v splošnem pa kažejo na to, da prilagajanje uporabniku lahko izboljša njegovo storilnost, izboljša razumevanje snovi, zmanjša število navigacijskih korakov in obiskov istih strani, zmanjša čas učenja in pomaga pri učenju. Seveda ima vsaka prilagoditvena tehnologija določene prednosti, zato je najučinkovitejša kombinacija različnih tehnologij, ki skupaj pomagajo uporabniku do boljših učnih rezultatov.

3.5 Primeri prilagodljivih izobraževalnih hipermedijev

Prilagodljivi hipermediji se največ uporabljajo v izobraževalne namene [Brusilovsky 1998a], zato je s tega področja tudi največ realiziranih sistemov. Nekateri od prilagodljivih izobraževalnih sistemov se uporabljajo v praksi pri samem pedagoškem procesu [Beaumont 1994, De Bra 1998, Kay 1994, Weber 1997], kjer so se izkazali za učinkovita orodja. Drugi sistemi so bolj eksperimentalne narave in (še) niso zaživeli zunaj laboratorijskega okolja. V nadaljevanju so opisani nekateri tipični prilagodljivi izobraževalni sistemi, ki uporabljajo različne pristope k modeliranju domene in uporabnika ter raznovrstne tehnologije prilagajanja.

3.5.1 ABITS

Sistem ABITS (Agent Based Intelligent Tutoring System) so razvili v okviru projekta InTraSys ESPRIT na italijanski univerzi Universitá degli Studi di Salerno (Capuano in sodelavci, 1999) [Capuano 2000]. Je spletni izobraževalni sistem za različne učne domene, ki uporablja modeliranje uporabnika in samodejno sestavlja učne tečaje.

Ves didaktični material je v sistemu organiziran v obliki več učnih objektov. To so logične enote kot so lekcije, simulacije, navidezni svetovi ali testi. Vsak učni objekt razlaga neko množico domenskih konceptov.

Učna domena je konceptualizirana s konceptnim grafom, ki določa tudi medsebojne relacije med koncepti. Povezave v grafu (relacije med koncepti) so lahko vrste predpogoj, podkoncept ali splošna relacija.

Model uporabnika obsega uporabnikovo kognitivno stanje in njegove učne preference. Kognitivno stanje opisuje dosežene stopnje znanja vsakega koncepta domene in je predstavljeno z nizom mehkih števil, eno število za vsak koncept. Učne preference zajemajo informacije o uporabnikovih zmožnostih dojemanja (vrsta medija, pedagoški pristop, stopnja interaktivnosti, semantična zgoščenost in težavnost), za njihovo oceno primernosti za uporabnika pa se tudi uporabljajo mehka števila. Kognitivno stanje se posodablja na podlagi učnih objektov (prebranih lekcij ali rezultatov testov), učne preference pa glede na uporabnikovo obnašanje v sistemu. Uporaba mehkih števil pri modeliranju uporabnika omogoča obravnavo nezanesljivosti pri ovrednotenju uporabnikovega znanja. Podrobneje je način opisan v razdelku 4.4.2.3.

Tečaj v sistemu sestavlja množica učnih ciljev in učni načrt. Učni cilj je množica ključnih konceptov, ki se jih mora uporabnik naučiti, da uspešno zaključi tečaj. Učni načrt pa je urejen seznam učnih objektov, ki se jih lahko uporabi za prikaz vsega potrebnega znanja za zaključitev tečaja. Prvi določa, katere koncepte se mora uporabnik naučiti, drugi pa, kako se naj jih nauči. Ker je učni načrt odvisen od modela uporabnika, ga sistem samodejno sestavi glede na dani model uporabnika in dane učne cilje.

3.5.2 AHA

Sistem AHA (Adaptive Hypermedia Architecture) [De Bra 1997, De Bra 1998] (in njegovega naslednika AHAM [Wu 1999, Wu 2000]) so razvili na univerzi Technische

Universiteit Eindhoven na Nizozemskem pod vodstvom De Braja (1996). Je prilagodljiv hipermedijski sistem za podajanje poljubnih hipermedijskih tečajev preko svetovnega spleta.

Tečaj sestavlja več hipermedijskih strani, kjer je vsaka stran še dodatno označena s komentarji. Ti opisujejo pogoje za prikaz strani (kateri koncepti morajo biti poznani, kateri koncepti ne smejo biti poznani) in seznam konceptov, ki jih stran razlaga. Iz teh komentarjev sistem zgradi seznam vseh konceptov domene in razbere tudi njihovo medsebojno odvisnost.

Model uporabnika beleži uporabnikovo poznavanje domenskih konceptov. Koncepti so drobno zrnati in jih je razmeroma veliko, saj model uporablja dvojiške vrednosti za opis znanja uporabnika. Poleg tega sistem vzdržuje dnevnik dostopov do posameznih strani, meri čas branja strani (beleži se vstop na stran in izstop s strani) in hrani rezultate vseh testov. Kratki testi so na koncu vsakega poglavja in pomagajo uporabniku pri razumevanju in utrjevanju snovi. Sistem predvideva, da uporabnik pridobi znanje z ogledi (in branjem) hipermedijskih strani in reševanjem testov, zato se po vsaki taki akciji posodobi model uporabnika.

Sistem AHA uporablja tri različne prilagoditvene tehnologije. Najosnovnejše je prilagajanje vsebine hipermedijskih strani, ki omogoča prikaz kontekstno odvisnih informacij v skladu z uporabnikovim napredovanjem v pridobivanju znanja. Skrivanje povezav je glavna tehnologija prilagodljive podpore navigaciji. Skrite so tako povezave na že poznane definicije in strani kot na prezahtevne strani za trenutno stopnjo znanja uporabnika. Za dodatno pomoč pri navigaciji nudi sistem tudi anotacijo povezav, ki poskrbi za spreminjanje barve besedila pri povezavi.

Sistem AHAM (Adaptive Hypermedia Application Model) je referenčni model arhitekture prilagodljivih sistemov, ki je nastal skozi proces ponovnega načrtovanja sistema AHA. Pomembna sprememba se pokaže predvsem v predstavitvi znanja, saj je znanje domene predstavljeno z manjšimi osnovnimi koncepti in s hierarhijo večjih sestavljenih abstraktnih konceptov.

3.5.3 AHM

Sistem AHM, prototipni sistem za pripravo prilagodljivih hipermedijskih učbenikov, je razvila skupina pod vodstvom Pilar da Silve (1997) na univerzi Katholieke Universiteit Leuven v Belgiji [Pilar 1998].

Zgradba sistema temelji na konceptih, katere razlaga množica hipermedijskih dokumentov. Koncepti so povezani z dokumenti in med seboj z obteženimi in pomensko označenimi

povezavami. Uteži povezav med koncepti določajo prag poznavanja koncepta za napredovanje. Vsak dokument ima pripisano tudi določeno težavnostno stopnjo, ki se uporablja za posodabljanje modela uporabnika.

Model uporabnika je zelo enostaven, saj hrani le stopnjo poznavanja posameznega koncepta. Začetne vrednosti so za vse koncepte nastavljene na nič, spremenijo pa se ob obisku dokumenta.

Vsakemu uporabniku je dodeljena množica dokumentov, ki so mu dostopni, in množica konceptov, ki so zanj primerni. Osnovni koncepti nimajo nobenih predpogojev (za razumevanje ne potrebujejo predhodnega znanja drugih konceptov), zato so primerni za uporabnika že takoj na začetku. Ostali koncepti pa so na voljo šele takrat, ko so dovolj dobro poznani vsi njihovi predpogojni koncepti. Vsaki posodobitvi modela uporabnika sledi ponoven izračun množice primernih konceptov in dostopnih dokumentov ter ustrezna posodobitev prikaza vsebine na zaslonu. Tako se povezave do konceptov in dokumentov, ki so na voljo, dinamično prilagajajo posameznemu uporabniku, odvisno od njegovega poznavanja domenskih konceptov.

Glavna metoda prilagajanja sistema je skrivanje povezav. Poleg tega omogoča tudi grafične predstavitve konceptov ali konceptualne mape in sicer dveh statičnih map (osnovni koncepti s pripadajočimi dokumenti in vsi koncepti brez dokumentov) in ene dinamične mape (trenutni koncept s pripadajočimi dokumenti, koncepti, ki vodijo do trenutnega koncepta, in koncepti, ki so iz njega dostopni).

3.5.4 Anatom–Tutor

Anatom–Tutor je hipermedijski inteligentni sistem za učenje anatomije. Razvil ga je Beaumont s sodelavci (1993) na inštitutu Fraunhofer Institut für Biomedizinische Technik v Nemčiji [Beaumont 1994].

Sistem lahko deluje v treh načinih. Brskalni način omogoča prost dostop do baze znanja preko menijev in interaktivnih diagramov ter tako nudi samostojno raziskovalno učenje. Vprašalni način, ki izdatno uporablja model uporabnika, simulira izpit in predstavlja vrsto priprave na izpit. Sistem zastavlja uporabniku vprašanja in ob nepredvidenih (napačnih) odgovorih tudi popravlja uporabnikove napačne predstave. Tretji način je tako imenovan hipernačin, kjer sistem na oseben način vodi uporabnika skozi učni material, poudarja pomembne dele in ilustrira uporabo materiala. Ta način je prilagodljiv, saj se hipertekstovni

sistem uporablja za strukturirano predstavitev znanja domene na način, ki najbolj ustreza stopnji znanja uporabnika.

Znanje učne domene je zajeto v inteligentni bazi znanja in hipertekstovnem modulu. Zajema koncepte, relacije med njimi in zakonitosti domene. Vsi imajo pripisane tudi faktorje gotovosti in če je ta faktor nad določenim pragom (ki je odvisen od stereotipa), se lahko uporabljajo pri sklepanju o domeni.

Model uporabnika uporablja stereotipe, ki se na začetku nastavijo s pomočjo vprašalnika ob prvi prijavi. Z njimi sistem modelira tako uporabnikovo znanje kot prepričanja. Stereotipi so obteženi ter zajemajo pare vrednost/atribut za opis konceptov iz baze znanja domene in produkcijska pravila za opis zakonitosti domene.

Prilagajanje uporabniku se izvaja na nivoju vsebine hipermedijskih strani, ki se spreminja z uporabo raztegljivega besedila. Sistem uporablja tudi neposredno vodenje za podporo navigaciji v hipernačinu.

3.5.5 APHID

Sistem APHID (Applying Patterns to Hypermedia Instructional Design) in njegova izboljšana različica APHID–2 sta plod raziskovanja skupine (Thomson, 2000) s kanadske univerze University of Saskatchewan [Kettel 2000]. Sistem omogoča individualizacijo spleta v izobraževalne namene, saj za vsakega uporabnika zgradi individualno predavanje v obliki hipermedijske aplikacije.

Konceptna mapa sistema zajema seznam vseh domenskih konceptov in relacije med njimi. Vsakemu konceptu pripadajo neki podatkovni elementi (tekst, video, slike in drugi podatki, ki bodo prikazani uporabniku), ki so lahko tipa vprašanje, razlaga, simulacija ali navodila. Sistem potem iz primitivnih podatkovnih elementov in na podlagi konceptne mape tvori prilagodljive hipermedijske učne aplikacije, ki temeljijo na nespremenljivih stereotipih uporabnikov.

Tako zgrajene hipermedijske učne aplikacije se lahko ločijo po izgledu posameznih strani, po izgledu spletnih strani kot celote ter po izbranih konceptih in podatkovnih elementih. Spreminja se lahko tudi vrstni red predavanja, barvna in tekstovna anotacija ter izbira povezav.

Sistem APHID predvideva, da ima uporabnik neko stereotipno obnašanje in ga zato posebej ne modelira. Zato sestavi celotno predavanje naenkrat in ga med uporabo ne spreminja.

Sistem APHID–2 je izpopolnjena različica sistema APHID, saj modelira uporabnikovo znanje, ki se spreminja z napredovanjem skozi hipermedijsko aplikacijo. Zato sistem, kadar aplikacija ne ustreza več danemu uporabniku, sestavi nov preostanek predavanja. Model uporabnika je prekrivni model preko konceptne mape. Hrani stopnjo znanja domenskih konceptov in tudi oceno uporabnikove splošne stopnje znanja. Preference in učni cilji uporabnika so zbrani v množici pravil, katera jih pretvorijo v parametre za sestavo hipermedijske aplikacije. Za posodabljanje in sklepanje o uporabnikovem znanju sistem na podlagi konceptne mape zgradi Bayesovo mrežo zaupanja. Podrobneje je sklepanje o uporabnikovem znanju razloženo v razdelku 4.4.3.3.

3.5.6 C–Book

Sistem C–Book je razvila Kay s sodelavci (1994) na avstralski univerzi University of Sydney [Kay 1994]. Sistem je namenjen učenju programskega jezika C in deluje v svetovnem spletu.

Vse hipermedijske strani, ki prikazujejo učni material, so označene s posebnimi komentarji, ki omogočajo sistemu, da spreminja vsebino strani v odvisnosti od vrednosti spremenljivk iz modela uporabnika. Na ta način sistem prilagaja vsebino prikazanih strani in izbiro povezav s strani.

Za modeliranje uporabnika sistem uporablja stereotipni model skupaj s posameznim modelom. Modelira se tako uporabnikovo predznanje kot njegove preference. Pri prvem je pomembno uporabnikovo poznavanje drugih programskih jezikov (na primer jezikov Pascal ali Basic) in tudi kako dobro jih uporabnik pozna. Preference uporabnika pa določajo način predstavitve informacij o programskem jeziku (abstraktno ali konkretno, zgoščeno ali bolj detajlno in eksplicitno, aktivno učenje ali usmerjeno učenje). Začetne vrednosti modela uporabnika so stereotipne in jih sistem izlušči iz odgovorov na kratek vprašalnik.

3.5.7 DCG

Sistem Dynamic Courseware Generator ali DCG je orodje za pripravo prilagodljivih računalniško podprtih tečajev. Razvili so ga na nemški univerzi Universität der Bundeswehr München pod vodstvom Vassileve (1992) [Vassileva 1995] in ga kasneje (1997) priredili tudi za delovanje v svetovnem spletu [Vassileva 1997]. Sistem omogoča avtomatično pripravo individualnega računalniško podprtega tečaja iz obstoječega učnega materiala glede na

uporabnikov izbrani cilj in njegovo predhodno znanje. Sistem tudi dinamično spreminja tečaj glede na uporabnikov uspeh pri pridobivanju znanja.

Sistem DCG uporablja eksplicitno predstavitev strukture konceptov domene, ki je ločena od učnega materiala. Konceptna struktura je AND/OR graf s koncepti v vozliščih in povezavami, ki ustrezajo možnim relacijam med koncepti. Vsakemu konceptu je pripisana množica učnega materiala, ki ta koncept opisuje. Učni materiali, ki so shranjeni v obliki HTML datotek na strežniku, zajemajo enote s predstavitvami, testi, vajami, razlago, pomočjo ali problemi.

Struktura domene se uporablja tudi za izgradnjo plana tečaja. Za določen ciljni koncept in z upoštevanjem že poznanih konceptov iz modela uporabnika sistem sestavi plan tečaja, to je zaporedje konceptov in relacij, ki se jih mora uporabnik naučiti skozi tečaj. Plan tečaja je pravzaprav podgraf strukture domene, ki povezuje uporabniku poznane koncepte s ciljnim konceptom. Sistem potem poskrbi za izvajanje plana z zaporednim prikazovanjem enot učnega materiala za vsak koncept iz sestavljenega plana. Za ustrezno razvrščanje učnih enot pri posameznem konceptu iz plana poskrbi množica pedagoških pravil. Med izvajanjem plana lahko uporabnik tudi testira svoje znanje konceptov s pomočjo pripravljenih vprašanj. Če se uporabnik ne more zadovoljivo naučiti določenega koncepta iz plana, sistem poskuša zgraditi nov plan, ki tega koncepta ne vključuje.

Model uporabnika temelji na verjetnostnem prekrivnem modelu za modeliranje znanja, poleg tega pa hrani še zgodovino in uporabnikove osebne preference. Prekrivni model se na začetku nastavi na vrednosti, ki jih sistem določi iz rezultatov reševanja uvodnega testa. Posodobi se po vsakem reševanju testov med izvajanjem učnega plana. Verjetnostna ocena poznavanja koncepta se izračuna po hevristični formuli iz števila in težavnosti uspešno rešenih testov, ki se nanašajo na ta koncept.

Prilagajanje uporabniku sistem DCG izvaja preko neposrednega vodenja skozi učni material.

3.5.8 ELM–ART

Sistem ELM–ART [Brusilovsky 1996b, Schwarz 1996], kar je okrajšava za Episodic Learner Model – Adaptive Remote Tutor, je tako kot njegov naslednik ELM–ART II [Weber 1997] nastal na univerzi Universität Trier v Nemčiji (avtorji Brusilovsky, Schwarz in Weber, 1996) in je eden prvih prilagodljivih sistemov v svetovnem spletu. Sistem temelji na

samostojnem inteligentnem učnem sistemu ELM–PE [Weber 1995] in je namenjen učenju programskega jezika Lisp.

Učni material se v sistemu ELM–ART nahaja v svetovnem spletu in je zato v hipermedijski obliki. Celoten tečaj je hierarhično razgrajen na enote, vsaka enota pa ima določen tudi pripadajoč seznam konceptov in njihovih vlog v enoti (koncept je lahko glede na enoto uveden, predstavljen, povzet ali predpogojen).

Domeno učenja sestavljajo koncepti, ki so medsebojno povezani s svojimi predpogojnimi koncepti in tako tvorijo konceptno mrežo. Vsak koncept je opisan v eni ali več učnih enotah in obratno, vsaka enota opisuje enega ali več konceptov.

Modeliranje uporabnika bazira na obteženem prekrivnem modelu in epizodičnem modelu (znanje je predstavljeno z nizom epizod, ki pomenijo učno zgodovino uporabnika). Znanje uporabnika je organizirano po konceptih, ki imajo pripisane vrednosti zaupanja, in se spreminja z obiski hipermedijskih strani in z odgovori na testna vprašanja. Vprašanja imajo določene stopnje zahtevnosti in ob odgovoru na vprašanje se vrednost zaupanja koncepta ustrezno spremeni glede na pravilnost odgovora in zahtevnost vprašanja. Koncept se smatra za poznan, ko njegova vrednost zaupanja preseže določen prag.

V sistemu ELM–ART je prosto preiskovanje učnega materiala podprto s prilagodljivo navigacijo v obliki anotacije (metafora semaforja) in razvrščanja povezav. Sistem uporablja tudi prilagajanje vsebine hipermedijskih strani, podpira programiranje s primeri in ima vgrajeno okolje za reševanje problemov.

Sistem ELM–ART II je novejša različica sistema ELM–ART z več izboljšavami, predvsem pri organizaciji elektronskega učbenika, sklepanju o znanju uporabnika na podlagi testov in anotaciji povezav (več informacij o stanju konceptov).

3.5.9 Hypernet

Sistem Hypernet je hipermedijska lupina, ki povezuje na znanju temelječe predstavitve (semantični hipermediji) z modeli nevronskih mrež ter tako uporablja zanimiv pristop k izobraževalnim hipermedijem. Prototip sistema, katerega avtor je Mullier (1999), so razvili na Leeds Metropolitan University v Angliji [Mullier 1999b].

Učno domeno opisuje semantična mreža, kateri je pripeta tudi hipermedijska vsebina v obliki neodvisnih vozlišč, ki uporabniku nudijo informacije s pedagoško vrednostjo. Sistem

uporablja semantično mrežo za predstavitev znanja domene in za izvajanje procesa sklepanja, ki mu omogoča avtomatično ustvarjanje obteženih povezav. Utež povezave pomeni moč semantične zveze med vozliščema, to je število semantičnih povezav, ki posredno povezujejo obe vozlišči. Na podlagi informacij iz modela uporabnika (katere vrste vozlišč je uporabnik predhodno obiskal) lahko sistem tudi spremeni uteži pri posameznih povezavah. Na ta način se povezave dinamično spreminjajo glede na trenutno stanje danega uporabnika. Sistem omogoča tudi ročno spremembo uteži povezav in s tem večjo naklonjenost določenim področjem semantične hipermedije.

Model uporabnika hrani informacije o uporabniku in njegovi izkušenosti, tako da lahko sistem obtežene povezave prikroji uporabnikovim zanimanjem. Zajema prekrivni model, bazo mehkih pravil ter dve nevronski mreži. Prekrivni model se uporablja za izgradnjo zgodovine obiskov vozlišč v semantični mreži (določa uporabnikova zanimanja). Ena nevronska mreža prepoznava uporabnikove strategije brskanja in jih poveže z nalogami in zmožnostmi uporabnika. Druga pa meri sposobnost uporabnika in ga na podlagi interakcije s sistemom razvrsti v skupine po zmožnostih. Za obe nalogi so nevronske mreže zelo primerne, saj so sposobne prepoznavati vzorce na nenatančnih podatkih, nalogo razvrščanja pa lahko opravijo avtomatično, brez ročne intervencije. Tako lahko model izlušči informacije o uporabniku brez neposrednih dialogov le na podlagi opazovanj njegovega obnašanja. Ta način modeliranja uporabnika si bomo natančneje ogledali v razdelku 4.4.2.4.

Sistem uporablja različne tehnologije prilagodljive podpore navigaciji za prilagajanje posameznemu uporabniku, njegovim ciljem in zanimanjem. Tako omejuje razpoložljivo izbiro povezav manj izkušenim uporabnikom z odstranjevanjem povezav na najmanj pomembne strani (skrivanje povezav), razpoložljive povezave pa so urejene (razvrščanje povezav) in označene s potjo do ciljnega vozlišča v hipermedijski strukturi (anotacija povezav).

3.5.10 HyperTutor

Sistem HyperTutor je razvila skupina raziskovalcev na španski univerzi Universidad del País Vasco pod vodstvom Péreza (1995) [Pérez 1995, Gutiérrez 1996]. Je prilagodljiv izobraževalni hipermedijski sistem, katerega zgradba jasno odraža združitev hipermedijev z inteligentnim poučevanjem.

Sistem sestavljata dva modula. Prvi, hipermedijska komponenta, obvladuje hipermedijsko obnašanje sistema, od organizacije hiperprostora domene, prikaza ustreznih vozlišč in

navigacije do komunikacije z uporabnikom. Drugi modul je učna komponenta, ki poskrbi za inteligentno obnašanje sistema. To zajema tako kontrolo nad učenjem in ocenjevanje kot prilagajanje dostopnosti hiperdokumentov.

Pedagoško domeno, katere znanje predstavlja hiperprostor, sestavljajo koncepti, njihove težavnostne stopnje, relacije med njimi ter različni vidiki učenja konceptov. Vsak koncept ustreza enemu hiperdokumentu. Povezave znotraj hiperdokumenta so obtežene, vrednost uteži pa določa njihovo aktivnost (neaktivnost) glede na model uporabnika. Povezave med hiperdokumenti pa so aktivne v odvisnosti od modela uporabnika in didaktičnih odločitev sistema, ki jih določa množica učnih pravil.

Modeliranje uporabnika v sistemu HyperTutor temelji na stereotipnem modelu, ki ga dopolnjuje s prekrivnim modelom (vsi poznani koncepti) za določanje poznavanja domene. V modelu so shranjeni tudi vse podatki o nalogah in njihovem reševanju. Pridobljeno znanje uporabnika se ovrednoti na podlagi reševanja nalog in obiskov vozlišč. Tako je poznavanje posameznega koncepta ovrednoteno s hevristično oceno (produkt povprečne ocene nalog in odstotka obiskanih vozlišč v hiperdokumentu), koncept pa postane poznan, ko ta ocena preseže polovico.

Sistem HyperTutor uporablja za prilagajanje uporabniku tehnologijo skrivanja povezav. Tako spreminja dostopnost do posameznih vozlišč v hiperprostoru in s tem skriva del informacij, vsebina vozlišč pa ostaja nespremenjena. Poleg tega nudi tudi klasična orodja za navigacijo po hiperprostoru.

3.5.11 InterBook

Na sistem InterBook lahko gledamo kot na nadaljevanje sistema ELM–ART. Razvili so ga na ameriški univerzi Carnegie Mellon University v sodelovanju z nemško univerzo Universität Trier (avtor Brusilovsky s sodelavci, 1996) [Brusilovsky 1998d, Eklund 1997]. Sistem je splošno orodje za pripravo in dostavo prilagodljivih elektronskih učbenikov v svetovnem spletu.

Učni material sestavlja množica elektronskih učbenikov, ki so hierarhično strukturirani v enote. Znanje o tečaju je predstavljeno z indeksiranjem hipermedijskih vozlišč, ki vsebujejo različne enote učnega materiala (predstavitve, teste, primere, probleme), s koncepti modela domene, ki se nanašajo na enoto.

Sistem InterBook uporablja strukturiran model domene, ki ga predstavlja mreža domenskih konceptov. Koncepti so razpoznavni po imenu in ne podpirajo semantičnih relacij med njimi. Vsaki končni enoti je pripet tudi seznam pripadajočih konceptov in njihovih vlog v enoti. Po vlogi se koncepti delijo na izhodne (naučeni v tej enoti) in na predpogojne (morajo biti poznani za razumevanje te enote).

Znanje uporabnika je predstavljeno s prekrivnim modelom, ki se na začetku nastavi na enega od stereotipov, določenih pri registraciji uporabnika. Za vsak domenski koncept hrani prekrivni model neko vrednost, ki je ocena nivoja uporabnikovega poznavanja tega koncepta. Model uporabnika se posodobi, ko uporabnik zaključi neko enoto. Za povečevanje (ali zmanjševanje) stopnje znanja posameznih konceptov se upoštevajo vse akcije uporabnika, kot so obiski strani, reševanje problemov ali odgovori na vprašanja. Model uporabnika vključuje tudi učne cilje, ki so opisani z množico konceptov, ki se jih mora uporabnik naučiti.

Prilagajanje uporabniku v sistemu InterBook temelji na tehnologiji prilagodljive podpore navigaciji. Povezave so anotirane z barvnimi ikonami in tipi pisav, ki razkrivajo status vozlišča, do katerega vodi povezava. Povezave, ki vodijo do predhodno že obiskanih vozlišč, so označene s kljukico. Sistem uporablja tudi neposredno vodenje (gumb “nauči me”) in prilagodljivo razvrščanje povezav.

3.5.12 ISIS–Tutor

Med prve prilagodljive hipermedijske učne sisteme spada sistem ISIS–Tutor, katerega sta razvila Brusilovsky in Pesin (1994) na moskovski državni univerzi v Rusiji [Brusilovsky 1994]. Sistem je namenjen učenju jezika za oblikovanje izpisa v sistemu za pridobivanje informacij CDS/ISIS.

ISIS–Tutor temelji na sistemu ITEM/PG [Brusilovsky 1992], ki omogoča modeliranje domene in uporabnika, vključuje učno komponento ter omogoča uporabniku eksperimentiranje z ukazi jezika. Dodana je hipermedijska komponenta, ki podpira s strani uporabnika vodeno pridobivanje konceptualnega znanja in na osnovi modela uporabnika nudi prilagodljivo podporo navigaciji. Hkrati med delom tudi posodablja model uporabnika tako, da ta odraža rezultate dela in napredovanje uporabnika.

Model domene sestavlja mreža domenskih konceptov, kjer povezave izražajo več vrst relacij med njimi. Vsako vozlišče te mreže ustreza vozlišču v hiperprostoru, povezave pa tvorijo glavne poti med hipermedijskimi vozlišči. Na ta način struktura hiperprostora odraža

tudi pedagoško strukturo znanja domene. Hipermedijske strani se dinamično sestavijo iz materiala, ki je shranjen v bazi znanja.

Prekrivni model uporabnika temelji na konceptih, kateri imajo vsak svoj celoštevilčni števec. Na začetku je model prazen, z uporabo sistema pa se model polni in posodablja vzporedno z naraščanjem uporabnikovega znanja. S pomočjo pragov lahko razdelimo koncepte na različne stopnje poznavanja, od le dveh (znano, neznano) do šestih različnih, odvisno od kompleksnosti prilagajanja. Ti pragovi se za različne koncepte ali uporabnike lahko tudi razlikujejo.

Sistem poleg prilagodljive predstavitve vsebine in neposrednega vodenja podpira tudi barvno anotacijo povezav (svoja barva za vsako od štirih stanj znanja). Sistem omogoča tudi določitev izobraževalnega cilja (to je množica konceptov, ki naj bi se jih uporabnik naučil) in prilagajanje glede na tako določen cilj (skrije vse koncepte, ki ne pripadajo danemu cilju, ali pa izpostavi ciljne koncepte).

3.5.13 KBS Hyperbook

Sistem KBS Hyperbook so razvili na nemški univerzi Universität Hannover (vodja projekta Nejdl, 1999), ime pa je dobil po angleški kratici razvojne skupine – Knowledge Based Systems Group [Henze 1999, Henze 2000a]. Je orodje za sestavo in vzdrževanje odprtih spletnih prilagodljivih izobraževalnih sistemov. Ker sistem deluje odprto, omogoča prepletanje spletnih strani s hipermedijskimi dokumenti sistema.

Sistem povezuje svoje informacijske vire (hipermedijski dokumenti sistema, projekti, primeri, zunanje spletne strani in drugi) z modelom uporabnika preko indeksiranja. Indeksirani koncepti se imenujejo elementi znanja in so medsebojno povezani v graf odvisnosti na podlagi konceptualnega modela tečaja. Elementi znanja se uporabljajo za indeksiranje informacijskih enot (vsak element znanja je povezan z eno informacijsko enoto kot glavni, lahko pa tudi z drugimi, kjer se pojavi, a ni glavni), projektnih enot (tisti elementi znanja, ki jih mora uporabnik poznati, da lahko uspešno zaključi projekt) in za opis obsega uporabnikovih ciljev (množica elementov znanja).

Model uporabnika sestavljajo tri komponente. Prva je vektor znanja, ki s pogojnimi verjetnostmi modelira uporabnikovo poznavanje elementov znanja. Druga komponenta modela uporabnika, model učnih odvisnosti, opisuje vse (predpogojne) odvisnosti med elementi znanja. Tretja komponenta pa je mehanizem sklepanja, ki s pomočjo Bayesove mreže izračuna

prepričanje sistema o uporabnikovem znanju posameznih elementov znanja. Taka predstavitev omogoča tudi obravnavo nezanesljivosti opazovanj sistema. Podrobnejši opis modela je v razdelku 4.4.3.4.

Sistem KBS Hyperbook nudi uporabniku na podlagi modela uporabnika prilagodljivo podporo navigaciji. Tako sistem sestavi bralno zaporedje strani glede na uporabnikove cilje in znanje, predlaga naslednji primerni učni korak v odvisnosti od izbranega cilja in projekta (neposredno vodenje), prikaže izobraževalno stanje povezav z metaforo semaforja (prilagodljiva anotacija) ter svetuje pri izbiri ciljev in projektov. Sistem podpira projektno učenje in eksplicitno ciljno učenje. Uporabnik lahko sam določi svoje učne cilje ali pa zahteva naslednji učni cilj od sistema. Za vsak izbran cilj se sestavi bralno zaporedje strani, ki vsebujejo vse potrebno znanje za dosego tega cilja. Hkrati sistem izbere tudi projekte, ki so primerni za dosego izbranih učnih ciljev, in sestavi indeks do ustreznih informacijskih dokumentov.

3.5.14 TANGOW

Sistem TANGOW (Task–based Adaptive learNer Guidance On the Web) je razvila Carro (1999) s sodelavci na univerzi Universidad Autónoma de Madrid v Španiji [Carro 1999]. Sistem je namenjen razvoju spletnih tečajev in njihovemu poučevanju.

Vsak tečaj je razdeljen na več nalog, ki so osnovne enote učnega procesa. Naloge so lahko enostavne ali sestavljene. Tudi na celoten tečaj lahko gledamo kot na eno samo sestavljeno nalogo. Razčlenitev nalog opisujejo pravila, ki določajo tudi relacije med nalogami in pogoje za aktiviranje pravil. Slednji se spreminjajo v odvisnosti od profila uporabnika, njegovih dosedanjih akcij (predhodno izvedenih nalog) in uporabljane učne strategije.

Informacije o posameznem uporabniku so shranjene v tako imenovanih profilih. Pri prvem vstopu v sistem se z vprašalnikom zberejo osebni podatki uporabnika (starost, jezik in preference) skupaj z načinom poučevanja. S tem se uporabniku določi enega od profilov, ki zajemajo pravila za dostop do nalog in pogoje za pripadnost posameznim profilom. Poleg profilov uporabnikov sistem beleži in shranjuje tudi njihove akcije.

Sistem omogoča prilagodljivo predstavitev vsebine tečaja glede na lastnosti uporabnika. Pri tem sistem spreminja uporabo multimedijskih elementov na strani in njihovo postavitev. Celotni dokumenti, povezani s posamezno nalogo, se sestavijo dinamično z uporabo tistih multimedijskih elementov, ki so najprimernejši za posameznega uporabnika. Kako se posamezna stran sestavi, pa spet določajo pravila.

3.5.15 Primerjava sistemov

Tu bomo na kratko povzeli poglavitne značilnosti opisanih hipermedijskih sistemov glede na značilnosti prilagodljivih hipermedijev (glej razdelka 3.3 in 3.4).

Vsi navedeni sistemi, razen nekoliko starejših sistemov Anatom–Tutor, HyperTutor in ISIS–Tutor, delujejo v svetovnem spletu. Večinoma so zasnovani splošnonamensko, za poljubno domeno učenja, le sistemi Anatom–Tutor, C–Book, ELM–ART in ISIS–Tutor slonijo na vnaprej določeni domeni učenja, ki pogojuje tudi samo izvedbo sistema. Vsi navedeni sistemi so hipermedijski, zato je učni material predstavljen v hipermedijski obliki.

Tudi pristopi k modeliranju domene se v svojem bistvu ne razlikujejo veliko, saj pri vseh temeljijo na konceptnem grafu. Povezave v grafu imajo lahko semantični pomen (kot v sistemu Hypernet), sestavljajo AND/OR graf (sistem DCG), večinoma pa so navadni grafi z označenimi in/ali obteženimi povezavami. Pri tem je izjema le sistem C–Book, ki modela domene ne uporablja, saj prilagajanje ni osnovano na znanju domene.

Ker so navedeni sistemi izobraževalni sistemi, ne preseneča, da večina sistemov pri modeliranju uporabnika poudarja ravno njegovo znanje (ABITS, AHA, AHM, Anatom–Tutor, APHID, DCG, ELM–ART, HyperTutor, InterBook, ISIS–Tutor, KBS Hyperbook). Poleg znanja so pomembni tudi uporabnikovi učni cilji (ABITS, APHID, DCG, InterBook, ISIS–Tutor, KBS Hyperbook), njegove preference (ABITS, APHID, C–Book, DCG), zgodovina dostopov do hipermedijskih strani (AHA, DCG, HyperTutor, InterBook, TANGOW) ter izkušnje (ELM–ART, InterBook). Metode modeliranja uporabnika so različne. Veliko se uporablja prekrivni model preko modela domene, največkrat v kombinaciji z drugimi metodami (stereotipne, verjetnostne, pravila, mehka pravila in podobne). Uporabljeni modeli so povzeti tudi v tabeli 3.1.

Polovica sistemov omogoča prilagajanje vsebine hipermedijskih strani (AHA, Anatom–Tutor, APHID, C–Book, ELM–ART, ISIS–Tutor, TANGOW), vendar z različnimi pristopi in večinoma v kombinaciji z drugimi prilagodljivimi tehnologijami.

Pri prilagodljivi podpori navigaciji se največ uporablja tehnologija anotacije povezav (AHA, APHID, ELM–ART, Hypernet, InterBook, ISIS–Tutor, KBS Hyperbook), neposrednega vodenja preko ciljnega učenja s sestavo individualiziranih tečajev (ABITS, APHID, DCG, KBS Hyperbook) in neposrednega vodenja s predlaganjem naslednje najprimernejše enote (Anatom–Tutor, InterBook, ISIS–Tutor), skrivanja povezav (AHA, AHM, APHID, C–Book, Hypernet, HyperTutor) ter razvrščanja povezav (ELM–ART, Hypernet, InterBook).

Tabela 3.1 – Primerjava opisanih prilagodljivih hipermedijskih sistemov

Sistem	**spletni**	**model uporabnika**	**prilagoditvene tehnologije**
ABITS	ja	prekrivni, mehka števila	sestava individualnega učnega načrta (neposredno vodenje)
AHA	ja	prekrivni	prilagajanje vsebine, skrivanje, anotacija
AHM	ja	prekrivni	skrivanje, konceptualne mape
Anatom–Tutor	ne	stereotipni, pravila	prilagajanje vsebine, neposredno vodenje
APHID	ja	prekrivni, verjetnostni	sestava individualnega tečaja (neposredno vodenje, prilagajanje vsebine, anotacija, skrivanje)
C–Book	ja	stereotipni	prilagajanje vsebine, skrivanje
DCG	ja	prekrivni, verjetnostni	sestava individualnega učnega načrta (neposredno vodenje)
ELM–ART	ja	prekrivni, obtežen	prilagajanje vsebine, anotacija, razvrščanje
Hypernet	ja	prekrivni, mehka pravila, nevronske mreže	skrivanje, razvrščanje, anotacija
HyperTutor	ne	stereotipni, prekrivni	skrivanje
InterBook	ja	stereotipni, prekrivni	anotacija, razvrščanje, neposredno vodenje
ISIS–Tutor	ne	prekrivni	prilagajanje vsebine, anotacija, neposredno vodenje
KBS Hyperbook	ja	prekrivni, verjetnostni	anotacija, neposredno vodenje
TANGOW	ja	stereotipni	prilagajanje vsebine

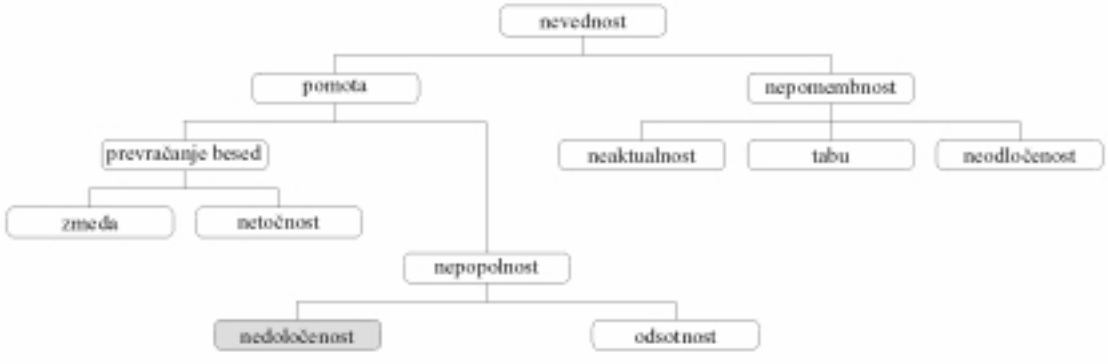
Tabela 3.1 povzema glavne značilnosti štirinajstih opisanih sistemov s poudarkom na možnosti spletne uporabe sistema, načinu modeliranja uporabnika ter uporabljenih prilagoditvenih tehnologijah.

Vsi sistemi so si precej podobni v načinu modeliranja domene, spletni uporabi in namenu, bolj pa se razlikujejo v pristopu k modeliranju uporabnika in uporabljenih prilagoditvenih tehnologijah.

4. Obravnava nedoločenosti

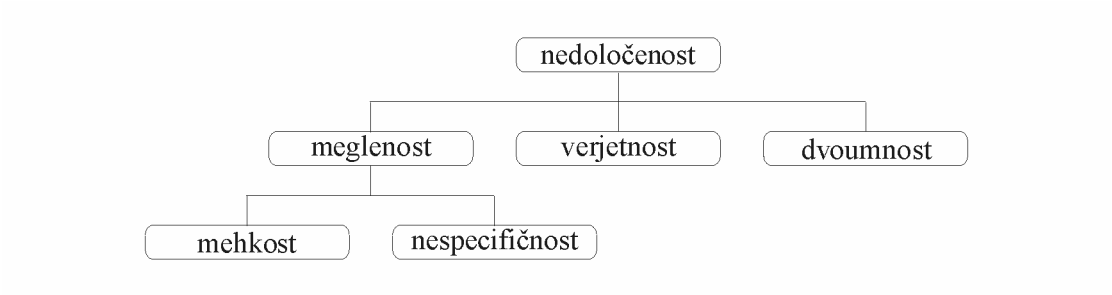
Pojmi, kot so nedoločenost, nejasnost, nerazločnost, dvoumnost, negotovost, nezanesljivost in podobni, opisujejo določeno nesigurnost ali dvom v našem poznavanju sveta. Pojem *nedoločenost* (angl. *uncertainty*) pokriva predvsem dva pojma, to sta *meglenost* (angl. *vagueness*) in *dvoumnost* (angl. *ambiguity*). Prvi se uporablja pri problemu, kako narediti ostre in natančne razlike v svetu. Neka domena je torej meglena, kadar je ne moremo opredeliti z ostrimi mejami (kot primer vzemimo množico starih ljudi, ko kljub poznani točni starosti, človeka ne moremo strogo označiti za starega oziroma ne starega). Po drugi strani pa se pojem dvoumnosti uporablja pri nejasnih relacijah, torej v situacijah, kjer izbira med dvema ali več alternativami ni točno določena. Tu gre za jasno določene (trde) množice s točno določenimi ostrimi mejami, a obstaja negotovost pri določanju pripadnosti elementa taki množici (kot primer vzemimo obtoženca kaznivega dejanja, ki je lahko ali kriv ali nedolžen, negotovost pa izvira iz pomanjkanja trdnih dokazov v povezavi s kaznivim dejanjem) [Klir 1988, Virant 1992].

Splošno tipologijo, ki povezuje različne vrste nedoločenosti (v širšem, bolj splošnem pomenu besede), je predlagal Smithson [Smithson 1989]. Po njej je nedoločenost le ena od oblik *nevednosti* (angl. *ignorance*), kot je razvidno tudi iz shematičnega prikaza na sliki 4.1. Del tipologije, ki se nanaša na nedoločenost, podrobneje prikazuje slika 4.2.



Slika 4.1

Tipologija različnih vrst nevednosti po Smithsonu



Slika 4.2

Del tipologije nevednosti po Smithsonu, nanašajoč se na nedoločenost

Za predstavitev in obravnavo nepopolnih in negotovih informacij so razvili raznovrstne načine in pristope. Različne tehnike, tako numerične kot simbolične, lahko na kratko strnemo v naslednje [Klir 1988, Krause 1993, Smets 1994, Parsons 1998, Isukapalli 1999]:

- *Klasična teorija množic* (angl. *crisp sets theory*). Nedoločenost je izražena z množicami medsebojno izključujočih se alternativ, kjer je zaželena ena sama alternativa. Nedoločenost izhaja iz nespecifičnosti, ki je vsebovana v vsaki množici.
- *Teorija grobih množic* (angl. *rough sets theory*). Groba množica je nenatančna predstavitev klasične trde množice s pomočjo dveh podmnožic, spodnje in zgornje aproksimacije.
- *Teorija mehkih množic* (angl. *fuzzy sets theory*). Koncept mehke množice nam nudi osnovni matematični okvir za obravnavanje meglenosti kot posebne vrste nedoločenosti. Pri mehkih množicah so pogoji za pripadnost elementa množici nejasni in nenatančni. Tu ne gre za izključno pripadnost ali nepripadnost množici, temveč za stopnjo pripadnosti elementa množici.
- *Teorija verjetnosti* (angl. *probability theory*). Nedoločenost je izražena z mero na podmnožicah univerzalne množice alternativ oziroma dogodkov. To je funkcija, ki vsaki podmnožici univerzalne množice priredi vrednost med 0 in 1, odvisno od situacije. Tako vrednost imenujemo verjetnost podmnožice. Teorija verjetnosti je le poseben primer teorije mehkih mer, a velja za najstarejšo in najpogosteje uporabljano. Bayesov model, ki temelji na verjetnostni funkciji, je prvi predlagani model za predstavitev nedoločenosti. Zato je tudi najbolj raziskan in na njegovi osnovi je razvita množica algoritmov, saj je dolgo časa veljal za edino orodje za reševanje problemov nedoločenosti.

- *Teorija možnosti* (angl. *possibility theory*). Možnostna teorija izhaja iz teorije mehkih množic in se tudi uporablja za obravnavo meglenosti. Teorija možnosti je del teorije mehkih mer ter obravnava meri možnosti in potrebnosti.
- *Teorija evidence* (angl. *evidence theory*). Matematična teorija evidence, po avtorjih imenovana tudi *Dempster–Shaferjeva teorija*, je prvi poskus generalizacije Bayesovega verjetnostnega modela. Temelji na funkciji osnovne dodelitve in iz nje izhajajočih mer zaupanja in prepričljivosti. V splošnem pa je tudi ta teorija le del teorije mehkih mer. Teorija evidence je v zadnjih letih dobila več interpretacij, ena od njih je tudi Smetsov *prenosljiv model zaupanja* (angl. *transferable belief model*).
- *Teorija mehkih mer* (angl. *fuzzy measures theory*). Teorija mehkih mer zajema več razredov mer, vsaka od njih pa ima določene posebne lastnosti. Koncept mehke mere uporabljamo pri obravnavi dvoumnosti kot vrste nedoločenosti. Tu so pogoji za pripadnost elementa množici natančno določeni, vendar je informacija o elementu pomanjkljiva in zato ne moremo določiti, ali element zadostuje tem pogojem pripadnosti. Teorija mehkih mer pokriva tudi že omenjene teorije verjetnosti, možnosti in evidence. Navedene tri teorije so najpogosteje uporabljani numerični pristopi k obravnavi nedoločenosti.
- *Faktorji gotovosti* (angl. *certainty factors*). Faktorje gotovosti so veliko uporabljali pri zasnovi ekspertnih sistemov, saj so zelo enostavni in intuitivni za uporabo. Omogočajo formalizacijo hevrističnega sklepanja preko deduktivnih pravil in hkrati nudijo način za merjenje nedoločenosti. Vsakemu pravilu oblike “če evidenca, potem hipoteza” se namreč dodeli neko številčno utež (faktor gotovosti), ki je razlika med stopnjo zaupanja in stopnjo nezaupanja v hipotezo na podlagi evidence.
- *Nemonotone logike* (angl. *non-monotonic logics*). Nemonotone logike so simbolični pristopi k obravnavi nedoločenosti, ki temeljijo na logiki prvega reda. Mednje spadata na primer *modalna logika* (angl. *modal logic*) in *privzeta logika* (angl. *default logic*). Klasična logika je monotona, kar pomeni, da z dodajanjem informacije h kontekstu raste tudi število teoremov, ki so dokazljivi iz tega konteksta. Tako se po dodani informaciji dokazljivost vseh obstoječih teoremov ohranja, s pomočjo dodane informacije pa ponavadi lahko izpeljemo še nekaj dodatnih teoremov (število teoremov se poveča). Pri nemonotonih logikah pa slednje ne velja, torej se po dodani informaciji dokazljivost obstoječih teoremov ne ohranja. Tako lahko pride do primera, da nekaterih teoremov, ki so bili prej dokazljivi, po dodani informaciji ne

moremo več izpeljati iz novega (posodobljenega) konteksta. Zato se lahko število dokazljivih teoremov pri dodajanju informacij tudi zmanjša.

Kot je razvidno tudi iz slike 4.2, se nedoločenost v sistemu lahko kaže v več oblikah. Izbira primernega modela za obravnavo nedoločenosti je odvisna od posameznega primera, ne moremo pa zagotovo trditi, da je določen model najboljši (boljši od drugih modelov) za dani primer [Smets 1994]. Vsak od pristopov je torej dober za neko vrsto nedoločenosti, nekateri primeri pa dajo boljše rezultate ob sočasni uporabi dveh ali več različnih pristopov, torej ob uporabi kombiniranih pristopov [Parsons 1998].

V nadaljevanju bomo podrobneje opisali teorijo mehkih množic in teorijo mehkih mer, skupaj s teorijami verjetnosti, možnosti in evidence.

4.1 Teorija mehkih množic

Z vrsto nedoločenosti, ki smo jo poimenovali meglenost, se ukvarja teorija mehkih množic. Ta temelji na principu mehkosti, kjer ni ostrega prehoda med pripadnostjo in nepripadnostjo množici. Tako je pripadnost množici podana s pripadnostno funkcijo, ki zavzema vrednosti na intervalu [0,1]. Elementi, ki imajo vrednost pripadnostne funkcije 1, so popolnoma prisotni v množici, tisti z vrednostjo 0 popolnoma odsotni, ostali elementi pa množici pripadajo le delno. Pripadnostna ali karakteristična funkcija mehke množice opisuje stopnjo pripadnosti elementa množici, to je stopnjo, s katero element zadovolji posamezne lastnosti, ki označujejo mehko množico [Savšek 1998].

Teorija mehkih množic ima svoje začetke že v dvajsetih letih dvajsetega stoletja v delih Russella na filozofskem področju [Russell 1923]. Vendar pa se je teorija zares prijela šele v šestdesetih letih z deli Zadeha [Zadeh 1965], v katerih je avtor podal matematične temelje teorije mehkih množic in s tem tudi mehke logike. Tako je teorija mehkih množic prešla tudi v širšo, aplikativno uporabo. Zadnja leta se mehka logika uporablja na različnih področjih, na primer pri odločanju, upravljanju, načrtovanju, napovedovanju, optimizaciji, razpoznavanju, krmiljenju, učenju ter na splošno v kompleksnih sistemih [Virant 1992, Klir 1988].

4.1.1 Trde množice in mehke množice

Imamo klasično množico objektov, imenovano univerzalna množica X, katere elemente označimo z x. Klasično podmnožico A univerzalne množice, ki jo imenujemo tudi *trda* ali

normalna množica (angl. *crisp set*), lahko opišemo z navedbo vseh elementov, ki pripadajo dani množici *A* ali pa povemo, da morajo elementi množice zadostovati nekemu pogoju *P*, ki določa množico *A*:

$A = \{x_1, x_2, \ldots x_n\}$ ali

$A = \{x_i : P(x_i)\}, \quad x_i \in X, \; i = 1 \ldots n.$

Pripadnost elementa množici *A* lahko označimo s *pripadnostno funkcijo* μ_A, ki elemente množice *X* preslika v množico {0,1} tako, da velja:

$$\mu_A : X \rightarrow \{0,1\}$$

$$\mu_A(x) = \begin{cases} 1, & x \in A \\ 0, & x \notin A \end{cases}$$

Za vsak element univerzalne množice torej velja, da ali pripada ali pa ne pripada množici *A*. Če pa množico {0,1} razširimo na interval realnih števil [0,1], množico *A* s pripadnostno funkcijo μ_A imenujemo *mehka množica* (angl. *fuzzy set*). Pri tem nam mera $\mu_A : X \rightarrow [0,1]$, ki preslika univerzalno množico *X* na interval realnih števil [0,1], predstavlja *stopnjo pripadnosti* elementa mehki množici *A*. Večja je vrednost μ_A (bližje je vrednosti 1), bolj element pripada množici *A*. Ostre meje med pripadnostjo in nepripadnostjo množici ni; obstaja mehko področje delne pripadnosti, ki pa je seveda odvisno od pripadnostne funkcije μ_A. Pripadnostna funkcija tako določa stopnjo, do katere je lastnost *P* zadovoljena za vsak element univerzalne množice *X*.

Mehko množico lahko popolnoma opišemo s pripadnostno funkcijo, tako da množico *A* popolnoma določa naslednji par:

$A = \{x, \mu_A(x)\}, \quad x \in X.$

Kadar je univerzalna množica *X* končna in $X = \{x_1, x_2, \ldots, x_n\}$, lahko mehko množico *A* izrazimo s pari $\mu_A(x_i)/x_i$, kjer je vsak element x_i opisan s stopnjo pripadnosti $\mu_A(x_i)$ množici *A*. To simbolično zapišemo kot vsoto vseh takih parov:

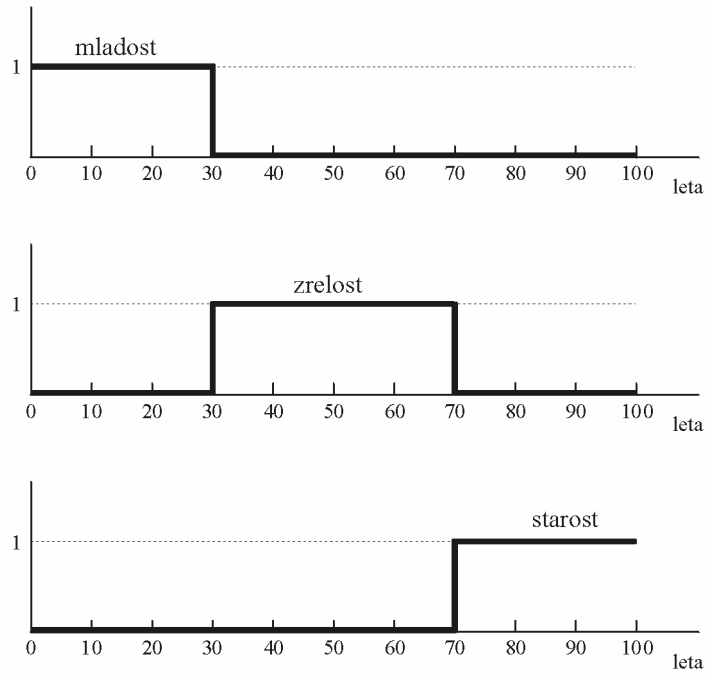
$$A = \{\mu_A(x_1)/x_1, \mu_A(x_2)/x_2, \ldots, \mu_A(x_n)/x_n\}$$

$$= \sum_{i=1}^{n} \mu_A(x_i)/x_i, \quad \text{za} \quad x_i \in X, \; i = 1 \ldots n.$$

Pri neskončni univerzalni množici X pa mehko množico A simbolično zapišemo kot integral parov stopnja pripadnosti in element:

$$A = \int_X \mu_A(x) / x \, .$$

Razliko med trdo in mehko množico najlažje predstavimo s primerom. Recimo, da želimo populacijo ljudi, starih do 100 let, razdeliti na tri množice: mlade, zrele in stare [Virant 1992]. Slika 4.3 prikazuje razdelitev let na tri množice z ostrimi prehodi. Za meje med posameznimi množicami smo postavili 30 in 70 let. Po taki delitvi pripada človek, mlajši od 30 let, množici mladih ljudi, človek, starejši od 70 let pa množici starih ljudi. Vsi ostali ljudje (torej med 30. in 70. letom) pripadajo množici zrelih ljudi.

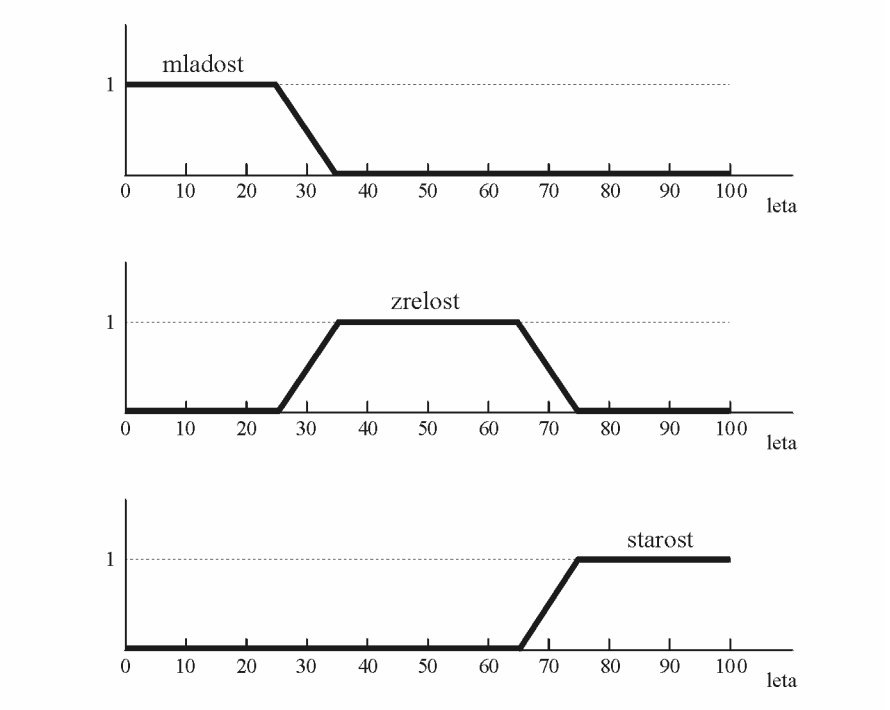


Slika 4.3

Pripadnost trdim množicam z ostrimi mejami

Če uporabljamo trde množice z ostrimi mejami pridemo do dileme, kje postaviti mejo med posameznimi množicami. Če na primer rečemo, kot je to v našem primeru, da je človek mlad do 30. leta, je taka delitev precej neživljenjska, saj le eno leto starejši ljudje (kar predstavlja samo en odstotek veliko spremembo v letih) pripadajo že množici zrelih ljudi. Zelo majhna sprememba v letih torej vodi v veliko spremembo v pripadnostih množicam. Vendar pa v

realnem življenju velja, da hkrati z mladostjo obstaja tudi delna zrelost in obratno, hkrati z zrelostjo obstaja tudi delna mladost. Meja med obema množicama je tako bolj zabrisana in nejasna, obe množici pa se med seboj prekrivata. Posamezen element lahko hkrati pripada obema množicama (v nekaterih primerih tudi več množicam), vsota pripadnostnih funkcij posameznim množicam pa je lahko tudi večja kot ena. Na sliki 4.4 so prikazane pripadnosti trem mehkim množicam, ki ustrezajo našemu primeru. Meje so bolj zabrisane, saj človek med 25. in 35. letom pripada tako množici mladih kot množici zrelih ljudi. Podobno velja za ljudi med 65. in 75. letom, ki pripadajo hkrati množici zrelih in množici starih.



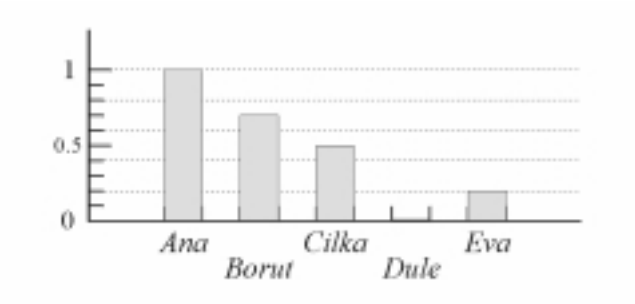
Slika 4.4

Pripadnost mehkim množicam z mehkimi mejami

Kot nadaljevanje primera vzemimo še množico petih ljudi X = {*Ana*, *Borut*, *Cilka*, *Dule*, *Eva*}, ki je naša univerzalna množica. Njihove pripadajoče starosti v letih so *Ana* 24, *Borut* 28, *Cilka* 30, *Dule* 43 in *Eva* 33. Naj bo mehka množica M množica mladih ljudi. Ustrezna pripadnostna funkcija μ_M, ki jo lahko razberemo iz slike 4.4, je definirana na intervalu [0,100] in jo zapišemo kot:

$$\mu_M(x) = \begin{cases} 1, & x \le 25 \\ \dfrac{35-x}{10}, & 25 < x < 35 \\ 0, & x \ge 35 \end{cases}$$

Elemente mehke množice mladih ljudi s pripadajočimi vrednostmi pripadnostne funkcije zapišemo kot *M* = {1.0/*Ana*, 0.7/*Borut*, 0.5/*Cilka*, 0.0/*Dule*, 0.2/*Eva*}. Stopnje pripadnosti množici mladi posameznih elementov prikazuje tudi slika 4.5.



Slika 4.5
Stopnje pripadnosti množici mladih ljudi

Poglejmo si še nekaj definicij, povezanih z mehkimi množicami in njihovimi pripadnostnimi funkcijami. *Območje* (angl. *support*) mehke množice *A*, ki ga zapišemo supp(*A*), imenujemo trdo podmnožico *X*, katere elementi imajo vrednost pripadnostne funkcije množici večjo od nič:

$$\text{supp}(A) = \{x \in X : \mu_A(x) > 0\}.$$

Območje mehke množice je torej množica tistih elementov, ki lahko pripadajo množici. Strožjim pogojem mora zadoščati *jedro* (angl. *kernel*) mehke množice *A*, ki ga sestavljajo le elementi z vrednostmi pripadnostne funkcije enakimi ena:

$$\text{kernel}(A) = \{x \in X : \mu_A(x) = 1\}.$$

Višina (angl. *height*) mehke množice *A*, označena z hgt(*A*), je enaka največji vrednosti pripadnostne funkcije, ki jo doseže nek element množice *X*:

$$\text{hgt}(A) = \sup_{x \in X} \{\mu_A(x)\}.$$

Mehka množica *A* je *normirana* (angl. *normalized*), če velja:

$\exists x \in X : \mu_A(x) = 1$.

Če je mehka množica *A* normirana, je njena višina hgt(*A*) = 1. Vsaki mehki množici *A* lahko določimo tudi njen α - *razrez* (angl. α - *cut*), ki ga označimo z A_α. To je trda podmnožica množice *X*, ki vsebuje vse tiste elemente množice *A*, katerih stopnja pripadnosti je najmanj α:

$A_\alpha = \{x \in X : \mu_A(x) \geq \alpha\}$.

S pomočjo α - razrezov lahko zapišemo tudi območje in jedro množice *A*:

$\text{supp}(A) = \bigcup_{\alpha>0} A_\alpha$ in

$\text{kernel}(A) = A_1$.

Za naš zadnji primer množice mladih ljudi *M* tako velja naslednje: območje množice je supp(*M*) = {*Ana*, *Borut*, *Cilka*, *Eva*}, njena višina je hgt(*M*) = max{1.0, 0.7, 0.5, 0.0, 0.2} = 1 (torej je množica *M* normirana), jedro množice *M* je kernel(*M*) = {*Ana*} = $M_{1.0}$, njeni α - razrezi pa so $M_{0.0}$ = {*Ana*, *Borut*, *Cilka*, *Dule*, *Eva*}, $M_{0.2}$ = {*Ana*, *Borut*, *Cilka*, *Eva*}, $M_{0.5}$ = {*Ana*, *Borut*, *Cilka*}, $M_{0.7}$ = {*Ana*, *Borut*} ter $M_{1.0}$ = {*Ana*}.

4.1.2 Operacije nad mehkimi množicami

Podobno kot pri trdih množicah so tudi nad mehkimi množicami definirane standardne matematične operacije: komplement, presek in unija. Te operacije so opisane s pomočjo pripadnostne funkcije, saj ta popolnoma določa mehko množico. Najprej pa si bomo ogledali še definicije prazne množice, enakosti množic in podmnožice.

Prazna množica. Mehka množica *A* je prazna, če imajo vsi elementi *X* pripadnost množici *A* enako 0:

$A = \emptyset \qquad \forall x \in X : \mu_A(x) = 0$

Enakost množic. Mehki množici *A* in *B* sta enaki, če imata njuni pripadnostni funkciji enaki vrednosti za vse elemente *X*:

$A = B \qquad \forall x \in X : \mu_A(x) = \mu_B(x)$

Podmnožica. Mehka množica *A* je podmnožica mehke množice *B*, če velja $\mu_A \leq \mu_B$ za vse elemente *X*:

$A \subseteq B \qquad \forall x \in X : \mu_A(x) \leq \mu_B(x)$

Komplement množice. Komplement mehke množice *A* ima pripadnostno funkcijo $1-\mu_A$:

$\overline{A}$ $\qquad\qquad \forall x \in X : \mu_{\overline{A}}(x) = 1 - \mu_A(x)$

Unija množic. Če je mehka množica *C* unija mehkih množic *A* in *B*, potem za pripadnostno funkcijo velja $\mu_C = \max(\mu_A, \mu_B)$:

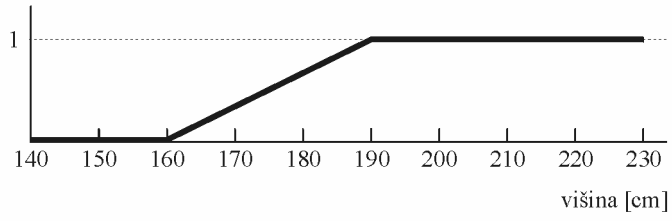
$C = A \cup B$ $\qquad\qquad \forall x \in X : \mu_C(x) = \max(\mu_A(x), \mu_B(x))$

Presek množic. Če je mehka množica *C* presek mehkih množic *A* in *B*, potem za pripadnostno funkcijo velja $\mu_C = \min(\mu_A, \mu_B)$:

$C = A \cap B$ $\qquad\qquad \forall x \in X : \mu_C(x) = \min(\mu_A(x), \mu_B(x))$

Za ilustracijo operacij nad mehkimi množicami se vrnimo k našemu prejšnjemu primeru. Poleg množice *M* mladih ljudi določimo še množico *V* ljudi, ki so visoke postave. Pripadnostna funkcija μ_V je definirana na intervalu [140,230] in pove, v kolikšni meri spada posameznik med ljudi visoke postave glede na njegovo v višino v centimetrih (glej tudi sliko 4.6):

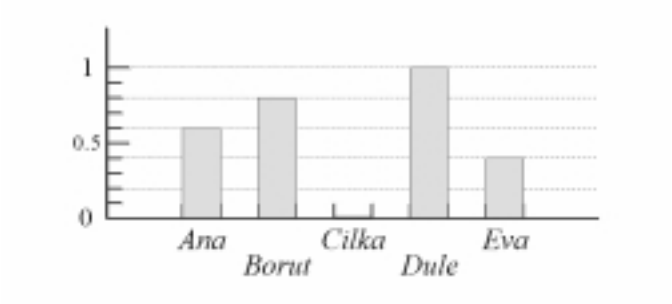
$$\mu_V(x) = \begin{cases} 1, & x > 190 \\ \dfrac{x-160}{30}, & 160 \le x \le 190 \\ 0, & x < 160 \end{cases}$$



Slika 4.6

Pripadnostna funkcija množici "je visoke postave"

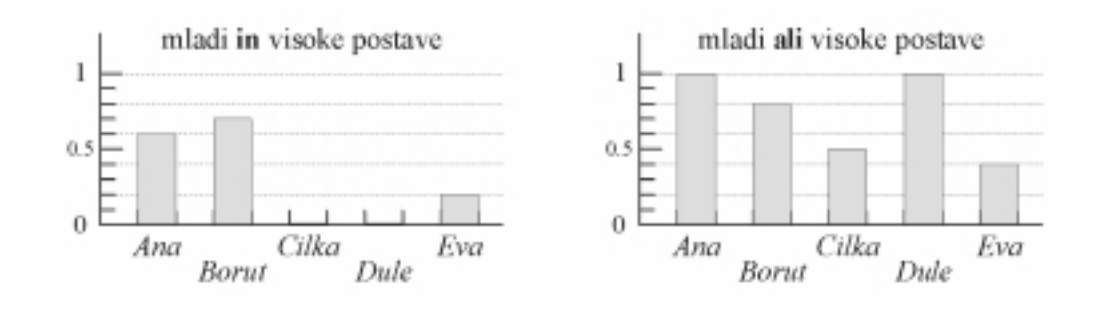
Če so pripadajoče višine *Ana* 178, *Borut* 184, *Cilka* 158, *Dule* 195 in *Eva* 172, potem lahko mehko množico ljudi, ki so visoke postave, zapišemo kot *V* = {0.6/*Ana*, 0.8/*Borut*, 0.0/*Cilka*, 1.0/*Dule*, 0.4/*Eva*}. Stopnje pripadnosti množici *V* prikazuje slika 4.7.



Slika 4.7

Stopnje pripadnosti množici ljudi, ki so visoke postave

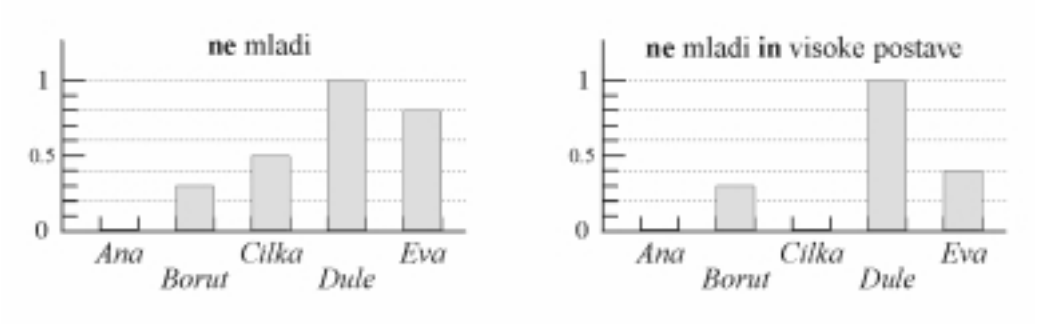
Sedaj lahko poiščemo množico vseh ljudi, ki so mladi in visoke postave, torej presek množic *M* in *V*. Ker je pripadnostna funkcija preseka definirana kot minimum pripadnostnih funkcij obeh množic, dobimo $M \cap V = \{0.6/Ana, 0.7/Borut, 0.0/Cilka, 0.0/Dule, 0.2/Eva\}$. Podobno dobimo množico ljudi, ki so bodisi mladi ali pa visoke postave, $M \cup V = \{1.0/Ana, 0.8/Borut, 0.5/Cilka, 1.0/Dule, 0.4/Eva\}$. Stopnje pripadnosti preseka in unije obeh množic so prikazane na sliki 4.8.



Slika 4.8

Stopnje pripadnosti množicama $M \cap V$ in $M \cup V$

Poiščimo še množico ljudi, ki niso mladi, a so visoke postave. Najprej moramo določiti množico "ne mladih" ljudi, torej $\overline{M} = \{0.0/Ana, 0.3/Borut, 0.5/Cilka, 1.0/Dule, 0.8/Eva\}$. Iskana množica pa je presek množic $\overline{M}$ in *V*, torej $\overline{M} \cap V = \{0.0/Ana, 0.3/Borut, 0.0/Cilka, 1.0/Dule, 0.4/Eva\}$. Slika 4.9 prikazuje ustrezne stopnje pripadnosti množici ne mladih $\overline{M}$ in množici ne mladih, ki so visoke postave, $\overline{M} \cap V$.



Slika 4.9

Stopnje pripadnosti množicama $\overline{M}$ in $\overline{M} \cap V$

Seveda pa zgornje definicije komplementa, unije in preseka nad mehkimi množicami niso edini možni načini posplošenja klasičnih operacij nad trdimi množicami. Poleg te trojice funkcij (1–*x*, *max* in *min*) obstaja tudi veliko splošnejših operacij, čeprav se opisana trojica uporablja zelo pogosto. Te operacije je definiral že Zadeh [Zadeh 1965] v svojih prvih delih s področja mehke logike in se zato imenujejo *originalne operacije*. Različni avtorji so predlagali vrsto drugih razredov funkcij, ki imajo ustrezne aksiomske lastnosti in s tem zadostujejo pogojem, ki jih morajo operacije izpolnjevati. Podrobnosti o tem lahko najdete na primer v [Klir 1988] ali [Virant 1992].

4.1.3 Mehke relacije

Mehke relacije so posplošenje klasičnih relacij nad trdimi množicami na področje mehkih množic. Klasične ali trde relacije nad trdimi množicami predstavljajo prisotnost ali odsotnost povezave (medsebojnega vpliva) med elementi dveh ali več množic. Te povezave lahko posplošimo tako, da dovolimo različne stopnje ali moči povezave med elementi. Pri tako dobljenih mehkih relacijah nam stopnjo povezave določa pripadnostna funkcija relacije.

V splošnem je *n*–razsežna *mehka relacija R* mehka podmnožica kartezičnega produkta $X_1 \times X_2 \times \cdots \times X_n$ s pripadnostno funkcijo $\mu_R(x_1, x_2, \ldots, x_n)$, ki je podana z izrazom:

$$R = \int_{X_1 \times \cdots \times X_n} \mu_R(x_1, \ldots, x_n) / (x_1, \ldots, x_n) .$$

Kadar je *n* enak 2, govorimo o binarni ali dvojiški relaciji. Binarna relacija je torej dvorazsežna relacija, ki povezuje dva elementa. V splošnem je to relacija nad dvema množicama, ki pa sta lahko tudi enaki.

Mehka binarna relacija. Binarna mehka relacija R iz množice X na množico Y je definirana kot mehka podmnožica kartezičnega produkta $X \times Y$, to je zbirka urejenih parov (x,y), kjer je $x \in X$ in $y \in Y$. Relacija R je opisana s pripadnostno funkcijo $\mu_R(x,y)$:

$$R(X,Y) = \{\mu_R(x,y)/(x,y)\}, \qquad R \subseteq X \times Y = \{(x,y) \mid x \in X, y \in Y\}.$$

Binarne relacije določajo pripadnostne funkcije, katerih vrednosti ponavadi zapišemo v obliki tabele ali matrike. Tako dobimo relacijsko matriko, kjer vsak element (m,n) predstavlja vrednost pripadnostne funkcije $\mu_R(x,y)$ za m–to vrednost elementa x in n–to vrednost elementa y.

Predstavimo binarno relacijo še na primeru. Če imamo množici X={*Ana*, *Borut*, *Cilka*} in Y={*Dule*, *Eva*}, lahko relacijo *privlačnost* zapišemo z vrednostmi njenih pripadnostnih funkcij za vsak element kartezičnega produkta *privlačnost* = {0.9/(*Ana*,*Dule*), 0.6/(*Ana*,*Eva*), 0.3/(*Borut*,*Dule*), 0.2/(*Borut*,*Eva*), 0.0/(*Cilka*,*Dule*), 0.8/(*Cilka*,*Eva*)} ali pa z relacijsko matriko:

$$\begin{array}{r} \\ \textit{Ana} \\ \textit{Borut} \\ \textit{Cilka} \end{array} \begin{array}{c} \begin{array}{cc} \textit{Dule} & \textit{Eva} \end{array} \\ \begin{bmatrix} 0.9 & 0.6 \\ 0.3 & 0.2 \\ 0.0 & 0.8 \end{bmatrix} \end{array}$$

Ker je mehka relacija pravzaprav tudi mehka množica, lahko relacije izvajamo tudi nad relacijami in jih tako medsebojno združujemo.

Mehka kompozicija. Mehka kompozicija dveh mehkih relacij $R \subseteq X \times Y$ in $S \subseteq Y \times Z$ je mehka relacija $R \circ S \subseteq X \times Z$, ki ima pripadnostno funkcijo:

$$\mu_{R \circ S}(x,z) = \max_{y \in Y}(\min(\mu_R(x,y), \mu_S(y,z)), \quad x \in X, y \in Y, z \in Z.$$

V našem primeru lahko poleg množic X in Y ter relacije *privlačnost* na $X \times Y$ definiramo še množico Z={*svetlolas*, *temnolas*} in relacijo (*je*/*ima*) *barva las* na $Y \times Z$, kjer relacijo *barva las* = {0.1/(*Dule*,*svetlolas*), 0.6/(*Dule*,*temnolas*), 0.9/(*Eva*,*svetlolas*), 0.5/(*Eva*,*temnolas*)} določa naslednja relacijska matrika:

$$\begin{array}{r} \\ \textit{Dule} \\ \textit{Eva} \end{array} \begin{array}{c} \begin{array}{cc} \textit{svetlolas} & \textit{temnolas} \end{array} \\ \begin{bmatrix} 0.1 & 0.6 \\ 0.9 & 0.5 \end{bmatrix} \end{array}$$

Kompozicijo relacij *privlačnost* (označimo jo s *P*) in *barva las* (označimo jo z *B*), ki je definirana na $X \times Z$, predstavlja spodnja relacijska matrika:

$$P \circ B = \begin{array}{c} \\ Ana \\ Borut \\ Cilka \end{array} \begin{array}{c} \begin{array}{cc} svetlolas & temnolas \end{array} \\ \begin{bmatrix} 0.6 & 0.6 \\ 0.2 & 0.3 \\ 0.8 & 0.5 \end{bmatrix} \end{array}$$

Kot primer izračuna zgornje matrike je naveden izračun vrednosti pripadnostne funkcije za par (*Borut*,*temnolas*):

$$\mu_{P \circ B}(Borut, temnolas) = \max_{y \in \{Dule, Eva\}} [\min(\mu_P(Borut, y), \mu_B(y, temnolas))] =$$

$$\max[\min(\mu_P(Borut, Dule), \mu_B(Dule, temnolas)), \min(\mu_P(Borut, Eva), \mu_B(Eva, temnolas))] =$$

$$\max[\min(0.3,0.6), \min(0.2,0.5)] = \max[0.3,0.2] = 0.3$$

Vidimo, da gre tu za vrsto "množenja matrik", kjer operacija min ustreza produktu (preseku), operacija max pa vsoti (uniji):

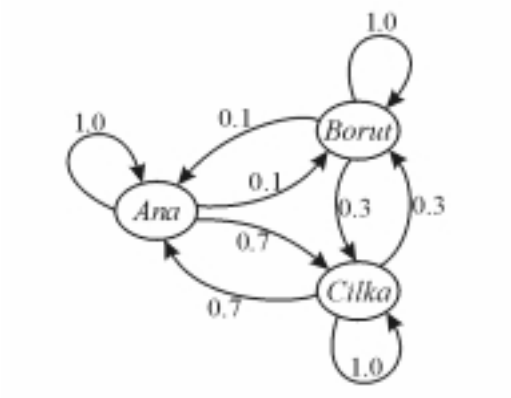
$$\begin{array}{c} \\ Ana \\ Borut \\ Cilka \end{array} \begin{array}{c} \begin{array}{cc} Dule & Eva \end{array} \\ \begin{bmatrix} 0.9 & 0.6 \\ 0.3 & 0.2 \\ 0.0 & 0.8 \end{bmatrix} \end{array} \circ \begin{array}{c} \\ Dule \\ Eva \end{array} \begin{array}{c} \begin{array}{cc} svetlolas & temnolas \end{array} \\ \begin{bmatrix} 0.1 & 0.6 \\ 0.9 & 0.5 \end{bmatrix} \end{array} = \begin{array}{c} \\ Ana \\ Borut \\ Cilka \end{array} \begin{array}{c} \begin{array}{cc} svetlolas & temnolas \end{array} \\ \begin{bmatrix} 0.6 & 0.6 \\ 0.2 & 0.3 \\ 0.8 & 0.5 \end{bmatrix} \end{array}$$

Binarne relacije na eni množici. Pri binarnih relacijah je potrebno izpostaviti še poseben primer, ko sta množici *X* in *Y* enaki, torej relacije med elementi ene same množice *X*. Taka relacija $R(X, X)$ je podmnožica kartezičnega produkta $X \times X$ in jo lahko predstavimo z usmerjenim grafom (vozlišča so elementi množice, povezave relacija med elementi, uteži povezav pa vrednosti pripadnostne funkcije).

Kot primer take relacije vzemimo že znano množico *X*={*Ana*, *Borut*, *Cilka*} ter definirajmo relacijo *je enako velik* = {1.0/(*Ana*,*Ana*), 0.1/(*Ana*,*Borut*), 0.7/(*Ana*,*Cilka*), 0.1/(*Borut*,*Ana*), 1.0/(*Borut*,*Borut*), 0.3/(*Borut*,*Cilka*), 0.7/(*Cilka*,*Ana*), 0.3/(*Cilka*,*Borut*), 1.0/(*Cilka*,*Cilka*)}.

$$\begin{array}{c} \\ Ana \\ Borut \\ Cilka \end{array} \begin{array}{c} \begin{array}{ccc} Ana & Borut & Cilka \end{array} \\ \begin{bmatrix} 1.0 & 0.1 & 0.7 \\ 0.1 & 1.0 & 0.3 \\ 0.7 & 0.3 & 1.0 \end{bmatrix} \end{array}$$

Relacijo *je enako velik* prikazuje zgornja matrika, lahko pa jo prikažemo v obliki usmerjenega grafa (slika 4.10).



Slika 4.10
Usmerjen graf relacije *je enako velik*

Lastnosti binarnih mehkih relacij. Pri binarnih mehkih relacijah lahko zasledimo več zanimivih lastnosti. Najpomembnejše med njimi so simetričnost, refleksivnost in tranzitivnost [Klir 1988].

Relacija $R(X,X)$ je *refleksivna*, če za vse $x \in X$ velja $\mu_R(x,x)=1$. Če to ne velja za nekatere $x \in X$, relacijo R imenujemo *nerefleksivna*, če pa to ne velja za vse $x \in X$, jo imenujemo *antirefleksivna*.

Relacija R je *simetrična*, če za vse $x,y \in X$ velja $\mu_R(x,y)=\mu_R(y,x)$. Če enačba ni izpolnjena za nekatere $x,y \in X$, je relacija *asimetrična*. Kadar pa iz $\mu_R(x,y)>0$ in $\mu_R(y,x)>0$ sledi, da je $x=y$ za vse $x,y \in X$, jo imenujemo *antisimetrična*.

Mehka relacija R je *tranzitivna* (natančneje *max–min tranzitivna*), če za vse pare $(x,z) \in X \times X$ velja $\mu_R(x,z) \geq \max_{y \in X} \min[\mu_R(x,y), \mu_R(y,z)]$. Če za nekatere pare ta neenačba ne velja, je relacija *netranzitivna*. Kadar pa za vse pare $(x,z) \in X \times X$ velja neenačba $\mu_R(x,z) < \max_{y \in X} \min[\mu_R(x,y), \mu_R(y,z)]$, je relacija *antitranzitivna*.

Če se spet vrnemo na naš zadnji primer in na relacijo *je enako velik*, ugotovimo, da je dana relacija refleksivna, simetrična in netranzitivna, saj velja (slika 4.10):

$$\forall x \in X : \mu_{je\ enako\ velik}(x,x)=1,$$

$\forall x, y \in X : \mu_{je\ enako\ velik}(x,y) = \mu_{je\ enako\ velik}(y,x)$ in

$\exists x, y, z \in X : \mu_{je\ enako\ velik}(x,z) < \max_{y \in X} \min[\mu_{je\ enako\ velik}(x,y), \mu_{je\ enako\ velik}(y,z)]$.

Podobno kot je pri klasičnih (trdih) relacijah definirana *ekvivalenčna* relacija, ki množico razdeli na ekvivalenčne razrede, lahko tudi pri mehkih relacijah definiramo relacijo s sorodnimi lastnostmi. Mehko binarno relacijo, ki je refleksivna, simetrična in tranzitivna, imenujemo *podobnostna relacija* (angl. *similarity relation*) [Klir 1988]. Medtem ko ekvivalenčna relacija jasno razdeli po relaciji ekvivalentne elemente v ločene razrede, pa lahko podobnostno relacijo interpretiramo na dva načina. V prvem primeru smatramo, da podobnostna relacija razdeli elemente v trde množice, katerih elementi so si podobni do določene mere. Po drugi strani pa lahko obravnavamo stopnjo podobnosti, ki jo imajo elementi množice *X* do posameznega elementa $x \in X$. Tako lahko za vsak element *x* določimo *podobnostni razred*, ki je mehka množica, njena pripadnostna funkcija pa določa podobnost elementov množice z danim elementom *x*. V nadaljevanju sta opisana oba načina.

Pri dani vrednosti α, ki je realno število na intervalu [0,1], lahko za mehko množico $A \subseteq X$ definiramo α-razrez A_α. To je trda množica, ki vsebuje vse elemente univerzalne množice *X*, ki imajo vrednost pripadnostne funkcije množici *A* večjo ali enako vrednosti α: $A_\alpha = \{x \in X : \mu_A(x) \geq \alpha\}$. Podobno lahko definiramo α-razrez tudi za relacijo *R* in ga označimo z R_α. Tako lahko vsako mehko relacijo *R* zapišemo tudi v obliki $R = \bigcup_\alpha \alpha R_\alpha$, kjer je αR_α mehka relacija, definirana z $\mu_{\alpha R_\alpha}(x,y) = \alpha \mu_{R_\alpha}(x,y)$, za vsak par $(x,y) \in X \times X$. Na ta način lahko mehko relacijo predstavimo kot zaporedje trdih relacij, ki obsega njene α-razreze, pomnožene z vrednostjo α. Kadar je relacija *R* podobnostna relacija, je R_α ekvivalenčna relacija. Tako lahko vzamemo neko podobnostno relacijo *R* in s pomočjo α-razrezov R_α za katerokoli vrednost α tvorimo trde ekvivalenčne relacije, ki predstavljajo prisotnost podobnosti med elementi do stopnje α. Množico *X* tako za vsak α razbijemo na več razredov, za katere velja $\mu_R(x,y) \geq \alpha$ za vsak par elementov (*x*,*y*) iz razreda.

Drugi pristop pa so podobnostni razredi, ki so definirani kot mehke množice. Za dano podobnostno relacijo $R(X,X)$ je podobnostni razred za vsak $x \in X$ definiran kot mehka množica, kjer je pripadnost množici za vsak element $y \in X$ določena z močjo relacije tega elementa do *x*, torej z $\mu_R(x,y)$. Podobnostni razred za element *x* tako predstavlja stopnjo, do katere so vsi ostali elementi množice *X* podobni elementu *x*.

4.1.4 Mehke relacijske strukture

Relacijske strukture sestavljajo osnovni gradniki in povezave med njimi, ki podajajo tudi njihovo zgradbo [Savšek 1998]. Glede na vrsto povezav med gradniki ločimo tri osnovne tipe, to so vrsta, drevo in graf. Slednji je najbolj zahtevna in hkrati tudi najbolj splošna ureditev, saj lahko na drevo in vrsto gledamo kot na posebni različici grafa.

Trdi graf G določa par (V, E), kjer je V (trda) množica vozlišč, $E \subseteq V \times V$ pa (trda) množica povezav med vozlišči, to je urejenih parov (u, v), kjer sta $u, v \in V$. V splošnem je graf G *usmerjen*, saj je $(u, v) \neq (v, u)$. Kadar pa za vsak par elementov $u, v \in V$ velja, če $(u, v) \in E$, potem tudi $(v, u) \in E$, govorimo o *neusmerjenem* grafu. V *obteženem grafu* ima vsaka povezava določeno težo (včasih jo imenujemo tudi dolžina ali kapaciteta povezave), ki predstavlja moč povezave oziroma relacije med vozliščema. Utež povezave w določa utežna funkcija W, ki preslika povezave v realna števila, $W : E \rightarrow \Re$, in tako vsaki povezavi $e_i \in E$ določi vrednost uteži $w_i = W(e_i)$. V obteženem grafu vsako povezavo določa trojica (u, v, w), kjer sta $u, v \in V$ in utež $w \in \Re$.

Dve vozlišči $u, v \in V$ v grafu sta sosednji, če med njima obstaja povezava, torej $(u, v) \in E$. Zaporedje vozlišč $s_0, s_1, s_2, \ldots s_n$, za katera velja, da sta s_k in s_{k+1} sosednji vozlišči za $\forall k \in [0, n-1]$, imenujemo *pot dolžine n*. Vozlišče s_0 je začetno vozlišče poti, vozlišče s_n pa končno. Vsa ostala vozlišča so vmesna vozlišča.

Cikel v grafu je pot, kjer začetno in končno vozlišče sovpadata, pot pa gre skozi vsako vmesno vozlišče le enkrat. Graf brez ciklov imenujemo *aciklični* graf.

Trdi graf opisujeta dve (trdi) množici, množica vozlišč in množica povezav. Kadar pa je katera od obeh množic mehka množica, govorimo o *mehkih grafih*. Tako ločimo tri vrste mehkih grafov: grafi z mehko definiranimi vozlišči in trdimi povezavami, grafi z mehko definiranimi povezavami in trdimi vozlišči ter grafi z mehko definiranimi vozlišči in povezavami [Savšek 1998]. Tem pa lahko dodamo še razne druge variacije mehkosti v grafih, kot so mehka množica trdih grafov, trdi grafi z mehko povezljivostjo (povezava je, a sta začetno in končno vozlišče mehko določeni) ter trdi grafi z mehkimi utežmi [Blue 1997]. Pri tem velja omeniti, da so različne vrste mehkih grafov medsebojno povezane, tako da lahko včasih mehkost enega tipa grafa prevedemo na drug tip grafa (na primer, iz dane mehke množice trdih grafov lahko sestavimo enakovreden mehki graf z mehko množico povezav).

Graf z mehkimi vozlišči. Kadar vozlišča grafa ne obstajajo popolnoma ali pa oznake vozlišča opisuje mehka množica *L*, so posamezna vozlišča mehka vozlišča. Mehki graf *G** je potem par (V^*, E), kjer je *V** mehka množica vozlišč, *E* pa množica povezav nad kartezičnim produktom $V^* \times V^*$, tako da velja:

$$V^* = \{v^*\}$$
$$v^* = \{a_i, \mu_v(a_i)\} \qquad \forall a_i \in L, \quad i = 1, \ldots r$$
$$\mu(a_i) : L \to [0,1]$$

$$\mu(v^*, u^*) : V^* \times V^* \to \{0,1\} \qquad \forall (v^*, u^*) \in V^* \times V^*$$
$$\mu(v^*, u^*) = \begin{cases} 1, & (v^*, u^*) \in E \\ 0, & (v^*, u^*) \notin E \end{cases}$$

Vsako vozlišče je definirano kot mehka množica nad množico oznak vozlišč *L*, povezave med vozlišči pa so trde relacije. Če je sama množica vozlišč mehka, potem je tudi obstoj samih povezav odvisen od obstoja vozlišč, saj povezava ne more obstajati brez začetnega ali končnega vozlišča.

Graf z mehkimi povezavami. Kadar povezave med vozlišči v grafu ne obstajajo popolnoma ali niso popolnoma odsotne, jih lahko opišemo kot mehke povezave. V tem primeru je mehki grafi $G^* = (V, E^*)$, kjer je *V* trda množica vozlišč, *E** pa množica mehkih povezav nad kartezičnim produktom $V \times V$, tako da velja:

$$\mu(v) : V \to \{0,1\} \qquad \forall v \in V$$
$$\mu(v, u) : V \times V \to [0,1] \qquad \forall (v, u) \in V \times V$$

Graf z mehkimi vozlišči in mehkimi povezavami. Mehki grafi $G^* = (V^*, E^*)$, kjer je *V** mehka množica vozlišč, *E** pa množica mehkih povezav nad kartezičnim produktom $V^* \times V^*$, ima mehko definirana tako vozlišča kot povezave. Pri tem velja:

$$V^* = \{v^*\}$$
$$v^* = \{a_i, \mu_v(a_i)\} \qquad \forall a_i \in L, \quad i = 1, \ldots r$$
$$\mu(a_i) : L \to [0,1]$$

$$\mu(v^*, u^*) : V^* \times V^* \to [0,1] \qquad \forall (v^*, u^*) \in V^* \times V^*$$

Tu je vsako vozlišče definirano kot mehka množica nad množico oznak vozlišč *L*, povezave med vozlišči pa so mehke povezave.

4.1.5 Mere mehkosti

Iz teorije mehkih množic je izšla tudi *mera mehkosti* (angl. *measure of fuzziness*), s katero lahko merimo meglenost sistema. Definirana je kot funkcija *f*

$$f : \mathcal{P}_m(X) \to \Re^+,$$

kjer je $\mathcal{P}_m(X)$ množica vseh mehkih podmnožic *X* (mehka potenčna množica). Tako funkcija *f* priredi vrednost *f*(*A*) vsaki mehki podmnožici *A* množice *X*, ta vrednost pa označuje stopnjo mehkosti množice *A*. Poleg tega mora funkcija *f* zadoščati tudi naslednjim trem aksiomom, da jo lahko imenujemo mera mehkosti:

1. *f*(*A*) = 0, če in samo če je *A* trda množica (stopnja mehkosti vseh trdih množic v $\mathcal{P}_m(X)$, in samo njih, je enaka 0),
2. če je $A \prec B$ (*A* je ostrejša od *B*), velja $f(A) \leq f(B)$,
3. *f*(*A*) ima maksimalno vrednost, če in samo če je *A* maksimalno mehka.

Ena takih funkcij, ki ustreza navedenim aksiomom in je tudi najbolj poznana, je definirana na osnovi Shannonove entropije. *Entropija* mehke množice *A* z dopolnilom *CA* (v spodnji formuli kar $\overline{A}$) je:

$$f(A) = H(A) + H(CA) \text{ ali}$$

$$f(A) = -\sum_{x \in X} (\mu_A(x) \log_2 \mu_A(x) + (1 - \mu_A(x)) \log_2 (1 - \mu_A(x))),$$

za katero je relacija ostrine $A \prec B$ definirana kot

$$\forall x \in X: \quad \mu_A(x) \leq \mu_B(x) \quad \text{za} \quad \mu_B(x) \leq \tfrac{1}{2} \quad \text{in}$$

$$\mu_A(x) \geq \mu_B(x) \quad \text{za} \quad \mu_B(x) \geq \tfrac{1}{2}$$

in maksimalna mehkost določena s pripadnostno stopnjo $\frac{1}{2}$ za $\forall x \in X$.

Druga znana mera mehkosti, imenovana tudi *indeks mehkosti* (angl. *index of fuzziness*), je definirana kot metrična razdalja med mehko množico *A* in njej najbližjo trdo množico *C*, za katero je:

$$\mu_C(x) = \begin{cases} 0, & \mu_A(x) \leq \frac{1}{2} \\ 1, & \mu_A(x) > \frac{1}{2} \end{cases}$$

Eden od indeksov mehkosti je *Minkowskijev razred metričnih mehkih razdalj*, ki je definiran z naslednjim izrazom:

$$f_w(A) = \left(\sum_{x \in X} \left| \mu_A(x) - \mu_C(x) \right|^w \right)^{\frac{1}{w}}, \quad \text{kjer je } w \in [1, \infty].$$

Ta zajema tudi *Hammingovo metrično mehko razdaljo*, ki jo dobimo pri vrednosti parametra *w* enaki 1, ter *evklidsko mehko razdaljo*, kadar je vrednost parametra *w* enaka 2. Pri tem sta relacija ostrine in maksimalna mehkost definirani enako kot pri entropiji.

Na bolj naraven način pa lahko stopnjo mehkosti množice izrazimo v obliki pomanjkanja razlik med množico in njenim komplementom, saj ravno to loči mehko množico od trde. Manj ko se množica razlikuje od svojega komplementa, bolj je množica mehka.

Mero mehkosti lahko tako izrazimo kot razliko metričnih razdalj poljubne trde podmnožice *X* (največja možna razdalja v $\mathcal{P}_m(X)$) in mehke množice *A*. Uporabimo lahko kar metrične razdalje Minkowskijevega razreda med množico in njenim komplementom:

$$f_{\bar{A},w}(A) = |X|^{\frac{1}{w}} - \left[\sum_{x \in X} \delta^w_{A,\bar{A}}(x) \right]^{\frac{1}{w}}, \text{ pri}$$

$$\delta_{A,\bar{A}}(x) = \left| \mu_A(x) - \mu_{\bar{A}}(x) \right|,$$

kjer je $\delta_{A,\bar{A}}$ absolutna vrednost razlike pripadnostnih funkcij množice *A* in njenega komplementa $\bar{A}$. Pripadnostna funkcija komplementa je določena z $\mu_{\bar{A}}(x) = 1 - \mu_A(x)$.

V tem primeru maksimalno mehkost določa pripadnostna funkcija enaka $\frac{1}{2}$, relacija ostrine pa je definirana kot

$A \prec B$, če in samo če $\left| \mu_A(x) - \mu_{\bar{A}}(x) \right| \geq \left| \mu_B(x) - \mu_{\bar{B}}(x) \right|$.

Za konec se spet vrnimo na naš prvi primer, kjer smo določili mehko množico mladih ljudi *M* = {1.0/*Ana*, 0.7/*Borut*, 0.5/*Cilka*, 0.0/*Dule*, 0.2/*Eva*}. Zanjo izračunajmo še nekatere mere mehkosti in sicer Hammingovo in evklidsko metrično mehko razdaljo tako do najbližje trde množice kot do njenega komplementa. Pri izračunih metričnih razdalj si pomagamo s spodnjo tabelo (tabela 4.1), kjer imamo zapisane posamezne vrednosti, potrebne pri izračunih. Moč množice *X* ali število vseh elementov univerzalne množice je $|X| = |M_0| = 5$.

Tabela 4.1 – Izračun metričnih mehkih razdalj za množico M

X	$\mu_M(x)$	$\mu_{\overline{M}}(x)$	$\mu_C(x)$	$\delta_{M,\overline{M}}(x)$	$\delta^2_{M,\overline{M}}(x)$	$\delta_{M,C}(x)$	$\delta^2_{M,C}(x)$
Ana	1.0	0.0	1.0	1.0	1.0	0.0	0.0
Borut	0.7	0.3	1.0	0.4	0.16	0.3	0.09
Cilka	0.5	0.5	0.0	0.0	0.0	0.5	0.25
Dule	0.0	1.0	0.0	1.0	1.0	0.0	0.0
Eva	0.2	0.8	0.0	0.6	0.36	0.2	0.04
Hammingova razdalja				2.0		1.0	
Evklidska razdalja					0.65		0.62

V prvem stolpcu tabele so navedeni elementi množice *M*, ustrezne vrednosti pripadnostne funkcije pa so v drugem stolpcu. Sledijo vrednosti pripadnostne funkcije komplementa in najbližje trde množice. Razlike vrednosti pripadnostnih funkcij in njihovi kvadrati so navedeni v zadnjih štirih stolpcih, kjer velja:

$$\delta_{M,\overline{M}}(x) = \left|\mu_M(x) - \mu_{\overline{M}}(x)\right| \quad \text{in} \quad \delta_{M,C}(x) = \left|\mu_M(x) - \mu_C(x)\right|.$$

S pomočjo vrednosti iz tabele lahko izračunamo Hammingovo in evklidsko metrično razdaljo med množico *M* in njej najbližjo trdo množico *C* po formulah:

$$f_1(M) = \sum_{x \in X} \delta_{M,C}(x) \qquad \text{za Hammingovo metrično razdaljo in}$$

$$f_2(M) = \left(\sum_{x \in X} \delta^2_{M,C}(x) \right)^{\frac{1}{2}} \qquad \text{za evklidsko metrično razdaljo.}$$

Ustrezni izračunani metrični razdalji sta tako 1.0 in $\sqrt{0.38} = 0.62$ po vrsti. Podobno izračunamo tudi razlike metričnih razdalj trde in mehke množice, kjer upoštevamo metrične razdalje med mehko množico in njenim komplementom:

$$f_{\overline{M},1}(M) = |X| - \sum_{x \in X} \delta_{M,\overline{M}}(x) \qquad \text{za Hammingovo metrično razdaljo in}$$

$$f_{\overline{M},2}(M) = |X|^{\frac{1}{2}} - \left(\sum_{x \in X} \delta^2_{M,\overline{M}}(x) \right)^{\frac{1}{2}} \qquad \text{za evklidsko metrično razdaljo.}$$

Izračunana mera mehkosti je tako v primeru Hammingove metrične razdalje 5.0 – 3.0 = 2.0, v primeru evklidske pa $\sqrt{5} - \sqrt{2.52} = 0.65$.

4.1.6 Mehka logika in mehko sklepanje

Osnovna predpostavka klasične logike je, da je vsaka prepozicija ali resnična ali neresnična. Ta dvovrednostna logika pa je pod vprašajem že od Aristotelovih časov, saj prepozicija prihodnjih dogodkov, kot je razpravljal Aristotel, ni niti resnična, niti neresnična, ampak je potencialno lahko karkoli. Tako se je razvila *n*–vrednostna logika ($n \geq 2$), kjer se interval [0,1] enakomerno razdeli na *n*–1 delov, imenovanih stopnje resničnosti. Primitivi logike so definirani z naslednjimi enačbami, katerih definicije so povzete po Łukasiewiczu [Klir 1988]:

$\bar{a} = 1 - a$,

$a \wedge b = \min(a, b)$,

$a \vee b = \max(a, b)$ in

$a \Rightarrow b = \min(1, 1 + b - a)$.

Mehka logika je razširitev večvrednostne logike in je v bistvu logika, ki sloni na teoriji mehkih množic. Mehka logika je osnova za približno ali mehko sklepanje, ki izhaja iz netočnih prepozicij in naravnega jezika.

V klasični logiki poznamo izjave in predikate. Prvi so spremenljivke v izjavnem računu, drugi pa v predikatnem računu. Izjavi "*a* je okrogel" in "*a* je zelen" označimo z *x* in *y* ter zapišemo:

x: *a* je okrogel

y: *a* je zelen

Vsaka izjava je lahko resnična ali neresnična. Z izjavnim računom lahko sestavljamo nove izjave v obliki funkcij *f*(*x*,*y*). V okviru predikatnega računa pa je postavljena predikatna funkcija *p*(*x*), kjer je *x* element predikatnega prostora *P*. Predikatna funkcija je izjava o elementih prostora *P* in jo zapišemo:

p(*x*): *x* je okrogel

Pri določenem elementu $x = a$ dobimo $p(a) = 1$, če a je okrogel, sicer $p(a) = 0$. Predikat, kjer je predikatna funkcija $p(x)$ mehka pripadnostna funkcija $\mu(x)$, imenujemo *mehki predikat*. Tako mehke predikate lahko predstavimo z mehkimi množicami. Iz pojma mehkih predikatov je izpeljan tudi pojem *mehke izjave*, ki je osnovni gradnik mehkega sklepanja.

Mehka logika je osnova za *mehko sklepanje*. Pri mehkem sklepanju se uporabljajo predvsem izjave v naravnem jeziku, ki jih težje rešujemo s klasičnim sklepanjem. V tem primeru so izjave *jezikovne spremenljivke*, to so spremenljivke, ki ne zavzemajo številskih vrednosti, marveč jih opisujejo besede in stavki naravnega (ali umetnega) jezika. Primer mehkega sklepanja z jezikovnimi spremenljivkami, ki ga ne moremo obravnavati s klasično predikatno logiko, je naslednji [Klir 1988]:

Stari kovanci so ponavadi redke zbirke.
Redke zbirke so drage.

Stari kovanci so ponavadi dragi.

Za tako sklepanje navadno uporabljamo *mehke predikate* (drag, star, redek), *mehke kvantifikatorje* (ponavadi, veliko, malo, skoraj), *mehke resnične vrednosti* (precej resnično, zelo resnično, bolj ali manj resnično, večinoma napačno) ter druge *mehke modifikatorje* (verjetno, skoraj nemogoče, izredno malo verjetno). V njem nastopajo predvsem jezikovne ali lingvistične spremenljivke, ki ne zavzemajo številskih vrednosti, temveč so opisane z besedami in stavki umetnega ali naravnega jezika.

Mehko sklepanje se naslanja na *mehko implikacijo*. Za mehki množici A in B lahko implikacijo izrazimo lingvistično, kar imenujemo tudi *mehko pravilo*:

if A **then** B oziroma

če A**, potem** B.

Ta trditev pomeni, da dobimo na izhodu množico B povsem zanesljivo samo takrat, kadar je na vhodu množica A. Če imamo na vhodu množico $\overline{A}$, potem na izhodu lahko imamo množico B ali pa tudi ne.

Mehko sklepanje predstavljajo tudi naslednji stavki:

če X je A, potem Y je B
X je A'

Y je B'

kjer sta X in Y jezikovni spremenljivki, A in A' mehki množici iz prostora U ter B in B' mehki množici iz prostora V. Potem mehko pravilo predstavlja mehko relacijo $A \Rightarrow B$ v kartezičnem produktu $U \times V$. Na osnovi kompozicijskega pravila sklepanja lahko B' izračunamo po enačbi:

$$B' = A' \circ (A \Rightarrow B),$$

kjer je $\circ$ operator kompozicije, $\Rightarrow$ pa operator implikacije. Izbira obeh operatorjev je lahko različna, tu bomo uporabili kombinacijo *sup–min* kompozicije in Łukasiewiczove implikacije. Tako velja:

$$\mu_{B'}(v) = \max_{u \in U} \min[\mu_{A'}(u), \mu_{A \Rightarrow B}(u,v)] \quad \text{in}$$

$$\mu_{A \Rightarrow B}(u,v) = \min(1{,}1 + \mu_B(v) - \mu_A(u)).$$

Seveda pa je nabor poznanih implikacij veliko širši, saj zajema več kot petnajst različnih definicij. Podrobnosti o drugih implikacijah lahko najdete v [Virant 1992].

4.2 Teorija mehkih mer

Kot smo že omenili na začetku tega poglavja, lahko v osnovi ločimo dve vrsti nedoločenosti, to sta meglenost in dvoumnost. Meglenost lahko obvladujemo s teorijo mehkih množic, dvoumnost pa obravnava teorija mehkih mer. Pri dvoumnosti gre za trde množice z ostrimi mejami, negotovost pa izhaja iz nepopolne ali pomanjkljive informacije o elementu in posledično tudi iz problema določanja pripadnosti množici. Problem rešujemo tako, da vsaki trdi množici, kateri bi vprašljivi element lahko pripadal, določimo neko vrednost. Ta vrednost določa stopnjo gotovosti, da element pripada dani množici. Tako predstavitev nedoločenosti imenujemo mehka mera.

4.2.1 Mehke mere

Mehkost lahko uvedemo v sistem tudi s pomočjo mehke mere, torej na osnovi trdih množic [Virant 1992]. V prostoru X poiščemo vse možne trde podmnožice in množico vseh takih podmnožic označimo s $\mathcal{P}$ (potenčna množica množice X). Zanima nas, kako pripada element x posameznim podmnožicam v $\mathcal{P}$. Zato vpeljemo *mehko mero* (angl. *fuzzy measure*), ki je:

$$g: \mathcal{P}(X) \rightarrow [0,1].$$

Tako $g(A)$ predstavlja stopnjo zaupanja, da dani element $x \in X$ pripada podmnožici $A \in \mathcal{P}(X)$.

Da lahko funkcijo g označimo za mehko mero, mora ustrezati naslednjim trem aksiomom:

1. $g(\emptyset) = 0$ in $g(X) = 1$,
2. če je $A \subseteq B$ pri $A, B \in \mathcal{P}(X)$, velja $g(A) \leq g(B)$,
3. pri monotonem zaporedju $A_1 \subseteq A_2 \subseteq \ldots$ oziroma $A_1 \supseteq A_2 \supseteq \ldots$, $A_i \in \mathcal{P}(X)$ velja $\lim_{i\to\infty} g(A_i) = g(\lim_{i\to\infty} A_i)$.

Dve pomembni vrsti mehke mere sta *mera zaupanja* (angl. *belief measure*) in *mera prepričljivosti* (angl. *plausibility measure*). Obe meri sta komplementarni, saj lahko eno vedno izpeljemo iz druge. Mera zaupanja *Bel* je funkcija:

$$Bel: \mathcal{P}(X) \rightarrow [0,1],$$

ki ustreza zgoraj navedenim aksiomom za mehke mere in naslednjemu dodatnemu aksiomu:

$$Bel(A_1 \cup A_2 \ldots A_n) \geq \sum_i Bel(A_i) - \sum_{i<j} Bel(A_i \cap A_j) + \ldots (-1)^{n+1} Bel(A_1 \cap A_2 \ldots A_n).$$

Na osnovi prvega aksioma, $Bel(\emptyset) = 0$ in $Bel(X) = 1$, ob upoštevanju $X = A \cup \overline{A}$ in $A \cap \overline{A} = \emptyset$, ter zadnjega aksioma, $Bel(A \cup \overline{A}) \geq Bel(A) + Bel(\overline{A}) - Bel(A \cap \overline{A})$, dobimo naslednjo neenačbo, ki velja za mero zaupanja (mera zaupanja ne izpolnjuje zakona polne verjetnosti):

$$Bel(A) + Bel(\overline{A}) \leq 1.$$

Mero prepričljivosti *Pl*, ki je dualna meri zaupanja, lahko zapišemo s pomočjo mere zaupanja:

$$Pl(A) = 1 - Bel(\overline{A}).$$

Zanjo kot četrti aksiom velja izraz

$$Pl(A_1 \cap A_2 \ldots A_n) \leq \sum_i Pl(A_i) - \sum_{i<j} Pl(A_i \cup A_j) + \ldots (-1)^{n+1} Pl(A_1 \cup A_2 \ldots A_n)$$

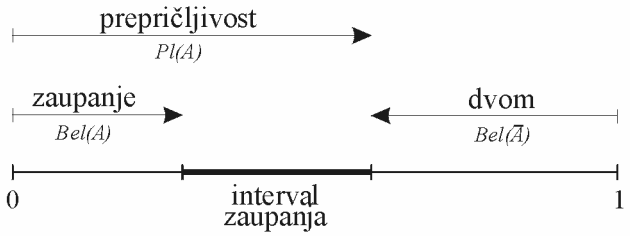
in zato sledi

$$Pl(A) + Pl(\overline{A}) \geq 1.$$

Mera prepričljivosti je vedno večja ali enaka meri zaupanja, kar sledi iz zgornjih neenačb:

$$Pl(A) \geq Bel(A).$$

Interval [*Bel*(*A*), *Pl*(*A*)] imenujemo *interval zaupanja* propozicije *A*. Spodnja meja intervala (*Bel*(*A*)) predstavlja razsežnost, do katere je propozicija *A* sigurno podprta. Zgornja meja intervala (*Pl*(*A*)) pa predstavlja razsežnost, do katere obstoječa evidenca ne uspe zavrniti propozicije *A*. Pri tem interval [1,1] predstavlja popolno sigurnost v propozicijo, interval [0,1] pa popolno nevednost v povezavi s propozicijo. Razmerja med mero zaupanja *Bel*, mero prepričljivosti *Pl* ter intervalom zaupanja so grafično prikazana na sliki 4.11.



Slika 4.11
Grafična predstavitev mer zaupanja in prepričljivosti

Meri zaupanja in prepričljivosti lahko izrazimo na drug način s pomočjo funkcije osnovne dodelitve. Naj bo funkcija *m* definirana kot:

$$m : \mathcal{P}(X) \rightarrow [0,1].$$

Zanjo naj velja:

$m(\emptyset) = 0$ in

$$\sum_{A \in \mathcal{P}(X)} m(A) = 1.$$

m(*A*) je potem stopnja evidence, ki podpira trditev, da določen element množice *X* pripada množici *A*, a ne pripada nobeni posebni podmnožici množice *A*. Funkcijo *m* imenujemo *osnovna dodelitev* (angl. *basic assignment*), z njo pa lahko enolično določimo meri zaupanja in prepričljivosti:

$$Bel(A) = \sum_{B \subseteq A} m(B) \quad \text{in}$$

$$Pl(A) = \sum_{B \cap A \neq \emptyset} m(B)$$

za vse $A \in \mathcal{P}(X)$. Ker je pogoj $B \cap A \neq \emptyset$ šibkejši od pogoja $B \subseteq A$, je očitno, da je $Pl(A) \geq Bel(A)$. Zaupanje v množico *A* je torej vsota vseh osnovnih dodelitev, ki podpirajo komponente množice *A*. Prepričljivost množice *A* pa je vsota vseh osnovnih dodelitev, ki ne podpirajo množice $\overline{A}$.

Vsako množico $A \in \mathcal{P}(X)$, za katero je $m(A) > 0$ (podmnožico *X*, ki ji pripada neničelna stopnja evidence), imenujemo *osrednji element* (angl. *focal element*) za *m*. Kadar je *X* končna množica, lahko *m* opišemo s seznamom osrednjih elementov *A* skupaj s pripadajočimi vrednostmi *m*(*A*). Množico takih osrednjih elementov označimo s $\mathcal{F}$.

Kadar so vsi osrednji elementi osnovne dodelitve množice z enim samim elementom (torej elementarne propozicije), potem za vse osrednje elemente *B* velja, da je $B \subseteq A$, če in samo če je $B \cap A \neq \emptyset$. Zato v takem primeru velja:

$$Bel(A) = Pl(A) = 1 - Bel(\overline{A}) \qquad \text{in}$$

$$Bel(A) + Bel(\overline{A}) = 1 .$$

Tak poseben primer mere zaupanja včasih imenujemo *Bayesova mera zaupanja*, bolj znana pa je pod imenom *mera verjetnosti* (angl. *probability measure*) ali krajše kar *verjetnost*, ki jo poznamo že iz klasične teorije verjetnosti.

Do verjetnosti pridemo tudi tako, da četrti aksiom, kateremu mora ustrezati mera zaupanja, močno omejimo:

$$Bel(A_1 \cup A_2) = Bel(A_1) + Bel(A_2), \quad \text{kadar } A_1 \cap A_2 = \emptyset.$$

Pri vseh množicah *A* iz $\mathcal{P}(X)$ tako velja poenostavitev:

$$Bel(A) = Pl(A) = \sum_{x \in A} m(\{x\}) .$$

Tako vsoto imenujemo verjetnost množice *A* in jo zapišemo kot:

$$P(A) = \sum_{x \in A} p(x) .$$

Funkcija $p(x)$, definirana na elementih množice *X*, se imenuje *funkcija verjetnostne porazdelitve*, ki je v svoji najpreprostejši obliki kar enakomerna porazdelitev:

$$p(x) = \frac{1}{|X|}, \qquad 0 < |X| < \infty .$$

Posebno vrsto mere zaupanja predstavljata tudi komplementarni meri, to sta *mera možnosti* (angl. *possibility measure*) in *mera potrebnosti* (angl. *necessity measure*). Mera možnosti, ki jo označimo s Π, ustreza meri zaupanja in pove, kakšna je možnost, da je nekaj res. Mera potrebnosti, označena z *N*, pa ustreza meri prepričljivosti in pove, kakšna je nemožnost, da nekaj ni res. Obe veljata pri pogoju, da so množice A_i ugnezdene (osrednji elementi iz $\mathcal{F}$ so ugnezdeni in zato nimajo neskladnosti v evidenci). Če torej velja $A_1 \subset A_2 \subset \ldots \subset A_n$, potem za vse $A,B \in \mathcal{P}(X)$ velja:

$$\Pi(A \cup B) = \max[\Pi(A), \Pi(B)] \quad \text{in}$$

$$N(A \cap B) = \min[N(A), N(B)] .$$

Ker sta meri komplementarni, sta povezani z enačbo:

$$N(A) = 1 - \Pi(\overline{A}) .$$

Podobno kot so verjetnostne mere izražene s pomočjo funkcije verjetnostne porazdelitve, se mere možnosti izražajo s *funkcijo možnostne porazdelitve* π:

$$\pi : X \rightarrow [0,1] ,$$

kjer je $\pi(x) = \Pi(\{x\})$. Če je $\pi(x) = 0$, je *x* nemogoč, v primeru $\pi(x) = 1$ pa je *x* popolnoma mogoč.

Vsako mero možnosti Π in mero potrebnosti *N* na $\mathcal{P}(X)$ lahko nedvoumno določimo s funkcijo možnostne porazdelitve π:

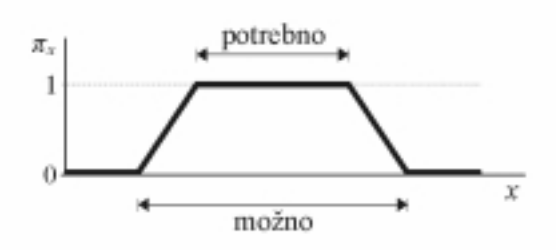
$$\Pi(A) = \sup\{\pi(x) \mid x \in A\} \qquad \text{in} \qquad N(A) = \inf\{1 - \pi(x) \mid x \notin A\} ,$$

kjer je *sup* (max) najmanjša zgornja meja množice *A* in *inf* (min) največja spodnja meja množice *A*.

Če je *A* nek dogodek, njegovo nasprotje pa $\overline{A} = X - A$, lahko zapišemo naslednje lastnosti mer možnosti in potrebnosti [Krause 1993]:

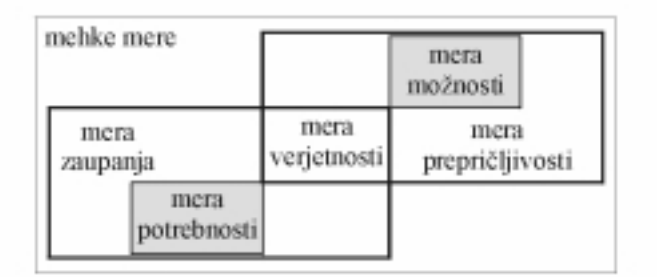
- Ob predpostavki zaprtega sveta mora množica *X* pokrivati vse možnosti in je zato vedno siguren dogodek: $\Pi(X) = N(X) = 1$.
- Ker mora nekaj vedno biti res, pomeni prazna množica protislovje in je zato vedno nemogoč dogodek: $\Pi(\emptyset) = N(\emptyset) = 0$.

- Unija dveh dogodkov je možna, če je katerikoli od obeh dogodkov možen: $\forall A, B \subseteq X: \ \Pi(A \cup B) = \max(\Pi(A), \Pi(B))$.
- Vsaj eden od dveh nasprotujočih si dogodkov je popolnoma možen, lahko pa smatramo tudi oba za možna: $\max(\Pi(A), \Pi(\overline{A})) = 1$.
- Potrebnost preseka dveh dogodkov je omejena na tisto manj potrebnega dogodka: $\forall A, B \subseteq X: \ N(A \cap B) = \min(N(A), N(B))$.
- Možnost dogodka nam pove bolj malo, saj lahko smatramo tako dogodek kot njegovo negacijo za možna. Vendar pa ima potrebnost dogodka veliko večjo moč pri določanju sigurnega dogodka, saj dva nasprotujoča si dogodka ne moreta biti hkrati potrebna: $\min(N(A), N(\overline{A})) = 0$.
- Obe meri sta dualni: $\forall A \subseteq X: \ \Pi(A) = 1 - N(\overline{A})$. Dogodek je torej možen do mere, do katere ne moremo izključiti potrebnosti njegovega nasprotnega dogodka.
- Dogodek postane možen, preden postane potreben (možnost je šibkejši pojem od potrebnosti): $\forall A \subseteq X: \ \Pi(A) \geq N(A)$. Če je dogodek vsaj malo potreben, mora biti popolnoma možen: $\forall A \subseteq X: \ N(A) > 0 \Rightarrow \Pi(A) = 1$. Če do določene mere dvomimo v možnost dogodka, potem dogodek ne more biti vsaj malo potreben: $\forall A \subseteq X: \ \Pi(A) < 1 \Rightarrow N(A) = 0$. Zadnji dve lastnosti prikazuje tudi slika 4.12.



Slika 4.12
Vrednostna povezanost možnosti in potrebnosti

Za zaključek si poglejmo še sistematično razvrstitev opisanih mehkih mer, ki je prikazana na sliki 4.13 [Klir 1988].



Slika 4.13
Glavni razredi mehkih mer in njihove medsebojne relacije

4.2.2 Mehke mere proti mehkim množicam

Na tem mestu naj še enkrat poudarimo razliko med mehkimi množicami in mehkimi merami. Za mehke množice velja, da imajo neostre meje, ter da nek element lahko pripada množici samo delno, torej do neke stopnje. Drugačen problem pa rešujejo mehke mere, ki se uporabljajo za premagovanje negotovosti pri določanju pripadnosti elementa množici (klasifikacija). V takem primeru je sama množica elementov sicer trda množica z jasno določenimi mejami, a so nejasne relacije med elementom in množico. Pri določanju stopnje pripadnosti mehki množici določimo vsakemu elementu univerzalne množice neko vrednost, ki pomeni stopnjo pripadnosti določeni množici z neostrimi mejami (podobnost z nenatančno določenimi lastnostmi). Po drugi strani pa mehka mera vsaki trdi podmnožici univerzalne množice določi neko vrednost, ki pomeni stopnjo evidence ali zaupanja, da določen element pripada tej množici [Klir 1988].

Opisano razliko med mehko množico in verjetnostjo (kot vrsto mehke mere) si poglejmo še na primeru [Brule 1992]. Recimo, da *Ana* pripada množici starih ljudi s stopnjo 0.8, torej $\mu_{stari}(Ana) = 0.8$. To pomeni, da je *Ana* bolj ali manj stara, ne moremo pa reči, da je res (zelo) stara. Če pa je verjetnost, da je *Ana* stara, 0.8 (torej $p(Ana) = 0.8$), to pomeni, da je 80% možnosti, da je *Ana* (zelo) stara, 20% pa, da ni stara. Torej *Ana* je ali pa ni stara (zagotovo ne vemo), možnosti pa so 80:20. Pri verjetnosti se dogodek zgodi ali ne in na to lahko stavimo, pri mehkosti pa ne moremo zagotovo reči, ali se je dogodek zgodil ali ne, določimo lahko le stopnjo, do katere se je dogodek zgodil.

Zanimiv je tudi znan primer močvirske vode, ki ga je Bezdek predstavil v svojem uvodu k prvi številki revije IEEE Transactions on Fuzzy Systems [Bezdek 1993]. Tu imamo univerzalno

množico tekočin in mehko podmnožico pitnih tekočin *L*. Imamo dve steklenici *A* in *B*. Steklenica A nosi oznako $\mu_L(A) = 0.91$, torej je vrednost pripadnostne funkcije množici *L* za element *A* enaka 0.91. Steklenica *B* pa je označena s $p_L(B) = 0.91$, kar pomeni, da je verjetnost, da je element *B* v množici *L* enaka 0.91. Vprašanje je, katero steklenico bi izbrali za pitje. Seveda je najvarnejša izbira steklenica *A*. Ta sicer lahko vsebuje na primer močvirsko vodo, zagotovo pa ne vsebuje strupenih tekočin, kot je na primer solna kislina. Visoka vrednost pripadnostne funkcije množici pitnih tekočin namreč pomeni, da je vsebina steklenice *A* precej podobna popolnoma pitni tekočini (na primer čisti vodi). Po drugi strani pa verjetnost, da je vsebina steklenice *B* pitna, pomeni, da pri daljši seriji poskusov v njej pričakujemo pitno vsebino v 91% primerov, v ostalih 0.09% pa je vsebina neužitna, lahko tudi smrtna. Po drugi strani pa se, če vključimo opazovanje, po pregledu in ugotovitvi vsebin obeh steklenic opisane vrednosti tudi spremenijo. Recimo, da smo ugotovili, da steklenica *A* vsebuje pivo, steklenica *B* pa solno kislino. Medtem ko za steklenico *A* še vedno velja $\mu_L(A) = 0.91$, pa je po opazovanju, ko že poznamo vsebini, verjetnost za steklenico *B* padla iz 0.91 na 0, torej $p_L(B) = 0$.

4.3 Vrste nedoločenosti in njihove mere

Mehke množice (skupaj z merami mehkosti) in mehke mere smo opisali v predhodnih razdelkih. Tu si bomo pogledali še nekatere mere nedoločenosti, ki temeljijo na merjenju informacije.

4.3.1 Klasične mere

Še pred razmahom teorije mehkih množic sta bili postavljeni dve glavni meri za nedoločenost. Prva, ki jo je predlagal Hartley leta že 1928, temelji na klasični teoriji množic, druga, ki jo je uvedel Shannon leta 1948, pa je izražena v obliki teorije verjetnosti [Klir 1988, Virant 1992]. Obe meri se nanašata na dvoumnost in sicer na merjenje nedoločenosti informacij. V izvornem smislu se nedoločenost nanaša na informacijo, zato se tudi omenjeni meri za nedoločenost nanašata nanjo.

Po Hartleyu je mera nedoločenosti, ki jo imenujemo kar *Hartleyeva informacija*, izražena v bitih in predstavlja količino potrebne informacije, da označimo eno izmed *N* alternativ. Pri *N* poskusnih alternativah je enaka:

$I(N) = \log_2 N$.

Shannon je v okvir nedoločenosti vnesel še verjetnost in podal mero nedoločenosti, imenovano *Shannonova entropija*:

$$H(p(x) \mid x \in X) = -\sum_{x \in X} p(x) \log_2 p(x),$$

kjer je $p(x)$ verjetnost pojavitve sporočila x, torej verjetnostna porazdelitev na končni množici X.

4.3.2 Mere disonance, konfuznosti in nespecifičnosti

Mero disonance lahko izpeljemo iz okvira mer zaupanja in prepričljivosti. Kot smo že zapisali v razdelku 4.2.1, lahko mero zaupanja *Bel* in mero prepričljivosti *Pl* izrazimo s pomočjo funkcije osnovnih dodelitev *m*:

$$Bel(A) = \sum_{B \subseteq A} m(B) \qquad \text{in} \qquad Pl(A) = \sum_{B \cap A \neq \emptyset} m(B)$$

za vse $A \in \mathcal{P}(X)$.

Mera disonance (angl. *dissonance measure*) je generalizacija Shannonove entropije v okviru matematične teorije evidence in jo zapišemo kot funkcijo:

$$E : \mathcal{M} \rightarrow [0, \infty)$$

$$E(m) = -\sum_{A \in \mathcal{F}} m(A) \log_2 Pl(A),$$

kjer je $\mathcal{M}$ množica vseh osnovnih dodelitev na potenčni množici z 2^n elementi, n je celo število, $\mathcal{F}$ pa je množica osrednjih elementov.

Meri disonance je komplementarna *mera konfuznosti* (angl. *confusion measure*), ki je definirana s funkcijo:

$$C(m) = -\sum_{A \in \mathcal{F}} m(A) \log_2 Bel(A) .$$

Za obe dualni meri veljajo naslednje neenačbe:

$$C(m) \geq E(m),$$

$$C(m) \geq 0,$$

$E(m) \geq 0$.

Podobno kot meri disonance in konfuznosti lahko v okviru matematične teorije evidence generaliziramo tudi Hartleyevo informacijo. Naj bo:

$U : \mathscr{R} \rightarrow [0, \infty)$,

kjer je $\mathscr{R}$ množica vseh urejenih možnostnih porazdelitev *r*. Pri tem $U(r)$ označuje stopnjo negotovosti, ki je povezana z možnostno porazdelitvijo *r*. Funkcijo *U* imenujemo *mera negotovosti* (angl. *measure of uncertainty* ali *U–uncertainty*) in je za možnostno porazdelitev $r = (\rho_1, \rho_2, \ldots, \rho_n)$, $\rho_{i+1} \leq \rho_i$ in $A_i = \{x \in X : r(x) \geq \rho_i\}$ definirana kot:

$$U(r) = \sum_{i=1}^{n} (\rho_i - \rho_{i+1}) \log_2 |A_i|$$

ali, ker je možnostna porazdelitev *r* urejena in so množice A_i ugnezdene, z upoštevanjem $|A_i| = i$:

$$U(r) = \sum_{i=1}^{n} (\rho_i - \rho_{i+1}) \log_2 i \, .$$

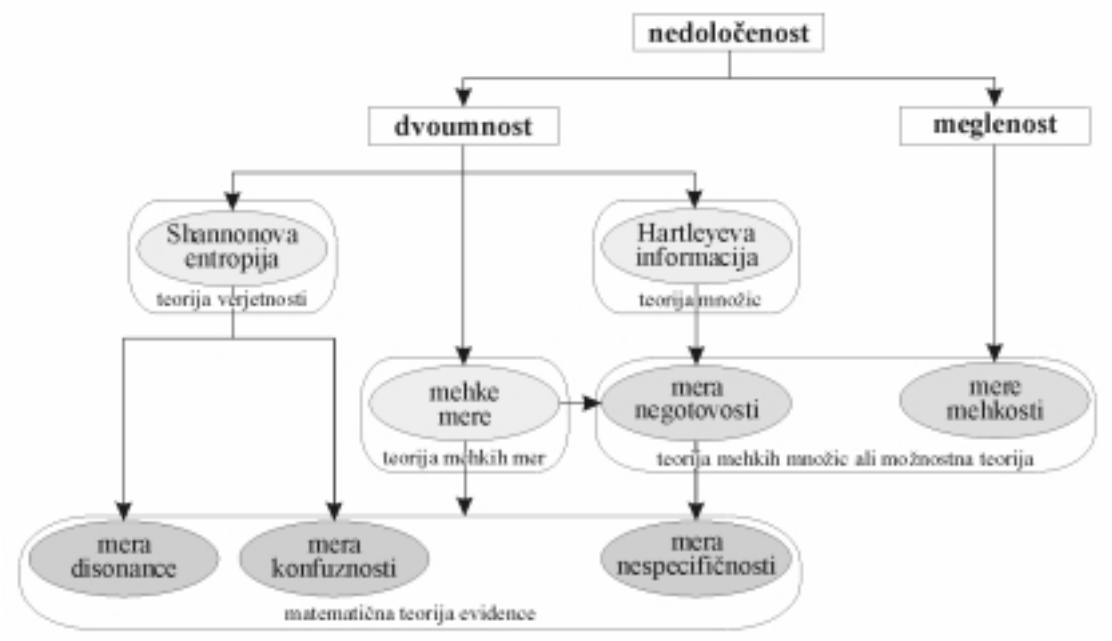
Mero negotovosti lahko generaliziramo in jo izrazimo s pomočjo osnovnih dodelitev *m*. Tako dobljeno funkcijo $V(m)$ imenujemo *mera nespecifičnosti* (angl. *nonspecificity measure*) in jo zapišemo kot funkcijo:

$V : \mathscr{M} \rightarrow [0, \infty)$

$$V(m) = \sum_{A \in \mathscr{F}} m(A) \log_2 |A| \, .$$

4.3.3 Mere nedoločenosti

V prejšnjih razdelkih smo spoznali več vrst nedoločenosti, vsako pa lahko izrazimo z ustrezno mero. Opisane vrste nedoločenosti se nanašajo na različne matematične teorije: trde množice, mehke množice ter teorijo mehkih mer, ki pokriva teorijo verjetnosti, možnostno teorijo in evidenčnostno teorijo. Na sliki 4.14 smo shematično prikazali opisane mere nedoločenosti skupaj z matematičnimi teorijami, iz katerih izhajajo [Klir 1988].



Slika 4.14

Shematičen pregled mer nedoločenosti

Na sliki so matematične teorije, na katere se nanašajo posamezne mere, označene z zaobljenimi pravokotniki. Elipse predstavljajo različne mere, vsaki je pripisano tudi njeno ime, le različne mere mehkosti in mehke mere so zbrane skupaj pod istim imenom. Slika prikazuje tudi relacije med posameznimi merami nedoločenosti. Nekatere mere so se razvile iz drugih mer kot generalizacija, specializacija ali pa so postavljene v okolje druge matematične teorije.

4.4 Obravnava nedoločenosti pri modeliranju uporabnika

Ta razdelek je namenjen pregledu različnih pristopov k obravnavi nedoločenosti pri modeliranju uporabnika. Tu gre predvsem za nezanesljivost in nepopolnost ocene uporabnikovega znanja, zato opisani sistemi upoštevajo faktor nedoločenosti pri razvoju in vzdrževanju modela uporabnika.

Tako izgradnja kot vzdrževanje modela uporabnika temeljita na rezultatih sklepanja sistema o uporabnikovem znanju na podlagi njegovih interakcij s sistemom. Seveda je pri tem prisotno veliko nedoločenosti, nezanesljivosti in negotovosti. Nepravilen uporabnikov odgovor na vprašanje je lahko posledica nerazumevanja koncepta ali pa na primer računske napake. Prav tako je lahko pravilen odgovor le posledica sreče pri izbiri odgovora. Za tako približno

sklepanje, kjer je prisotno veliko negotovosti, so na področju umetne inteligence razvili več tehnik, ki so jih najprej uporabljali v ekspertnih sistemih. Zato ni presenetljivo, da so se prvi poskusi vpeljave nezanesljivosti uporabnikovega znanja v model uporabnika pojavili ravno na področju inteligentnih učnih sistemov. Kljub temu, da so prilagodljivi hipermedijski sistemi podedovali inteligentne tehnike in modeliranje uporabnika od inteligentnih učnih sistemov, pa ti sistemi, razen redkih izjem, pri modeliranju uporabnika ne upoštevajo nezanesljivosti znanja. Največ poskusov vpeljave dejavnika nedoločenosti v prilagodljive hipermedije je bilo narejenih predvsem v zadnjem času, slonijo pa na pravilih (sistem ELM–ART), teoriji mehkih množic (sistema ABITS in Hypernet) ter verjetnostni teoriji na osnovi Bayesovih mrež (sistema APHID in KBS Hyperbook). Rezultati so sicer zanimivi in spodbudni, vendar bi kazalo poskusiti tudi z drugimi pristopi.

V splošnem sistemi, ki modelirajo uporabnika z upoštevanjem nedoločenosti, uporabljajo predvsem štiri različne načine približnega sklepanja [Millán 2000]:

- pravila sklepanja s faktorji gotovosti,
- mehko logiko,
- Bayesove verjetnostne mreže ter
- Dempster–Shaferjevo teorijo evidence.

V nadaljevanju bomo na kratko opisali vsakega od navedenih načinov na primerih sistemov, ki tak način uporabljajo pri modeliranju uporabnika. Obsežnejši pregled različnih rešitev je v [Jameson 1996] skupaj z opisi sistemov, ki te rešitve uporabljajo.

4.4.1 Faktorji gotovosti

Ena prvih tehnik, ki so jo na področju v umetne inteligence razvili za reševanje problema nezanesljivosti znanja, so pravila sklepanja na osnovi faktorjev gotovosti [Bratko 1989, Millán 2000]. Model je bil razvit pri ekspertnem sistemu za diagnostiko infekcijskih bolezni MYCIN [Shortliffe 1976], kjer so dejstvom in pravilom oblike ČE–POTEM dodali še mero zaupanja z intervala od –1 do +1, ki se imenuje faktor gotovosti. Ta izraža stopnjo zaupanja v hipotezo glede na trenutno razpoložljiva dejstva in stopnjo zaupanja v zaključek, do katerega pridemo z uporabo pravila. Faktor gotovosti +1 pomeni, da razpoložljiva dejstva popolnoma potrjujejo hipotezo, faktor –1 pa pomeni, da razpoložljiva dejstva popolnoma potrjujejo negacijo hipoteze. Faktor gotovosti 0 pomeni, da razpoložljiva dejstva ne potrjujejo niti hipoteze niti

njene negacije. Faktorji gotovosti pravil in začetnih dejstev so podani vnaprej (subjektivne ocene ekspertov), nova dejstva pa se računajo na osnovi funkcije min in produkta. Tak način obravnave nedoločenosti je precej enostaven za razumevanje, realizacijo in implementacijo, vendar pa je nemogoče sestaviti skladen model faktorjev gotovosti.

Podobno, s približnim sklepanjem, obravnava negotovost tudi sistem za geološke raziskave rudnih ležišč Prospector [Duda 1979]. Ta uporablja vrsto mehke implikacije (pravila oblike ČE–POTEM), ki sta ji dodana faktor potrebnosti in faktor zadostnosti. Faktor zadostnosti pove, kako zadostno neko dejstvo potrjuje resničnost hipoteze. Faktor potrebnosti pa določa, kako potrebna je resničnost dejstva, da je tudi hipoteza resnična. Pri resničnem dejstvu je torej pri večjem faktorju zadostnosti bolj verjetna tudi resničnost hipoteze. Po drugi strani pa pri neresničnem dejstvu večji faktor potrebnosti pomeni tudi večjo verjetnost za neresničnost hipoteze.

Nekateri sistemi imajo razvito svojo hevristiko za reševanje problema obravnave nezanesljivosti znanja. Eden takih je tudi sistem ELM–ART [Weber 1997, Millán 2000], ki gradi na pravilih (opis sistema je v razdelku 3.5.8). Za merjenje znanja uporabnika, ki je organizirano po konceptih, uporablja vprašanja vrste test. Vsakemu konceptu je pripisana neka vrednost zaupanja in vsako vprašanje ima neko stopnjo zahtevnosti (število med 0.5 in 1.5). Če je odgovor na vprašanje pravilen, se vrednost zaupanja koncepta ustrezno poveča na stopnjo zahtevnosti, ki je določena za dano vprašanje. Če pa je odgovor napačen, se vrednost zmanjša. Kadar vrednost zaupanja preseže neko vnaprej določeno vrednost, se smatra, da uporabnik koncept obvlada.

4.4.2 Mehka logika

Za izražanje nezanesljivosti nudi teorija mehke logike več načinov [Millán 2000]:

- mehke množice,
- jezikovne (mehke) spremenljivke (npr. X = “težavnostna stopnja vprašanja”, vrednosti pa so {*lahko*, *precej lahko*, *precej težko*, *težko*}),
- mehke relacije (npr. relacija R = “težavnostni stopnji vprašanj X in Y sta enaki”) in
- mehko sklepanje.

Nedvomno se prednosti uporabe principov mehke logike za izražanje nezanesljivosti pokažejo predvsem v sistemih, kjer se uporablja sklepanje, ki ga lahko naravno opišemo in

razložimo v obliki nenatančnih konceptov, operatorjev in pravil. Podobno velja tudi za sisteme, kjer je potrebno obdelati nenatančni besedni vhod od uporabnika, saj so za predstavitev takega vhoda zelo primerne pripadnostne funkcije [Jameson 1996]. Mehke pripadnostne kategorije ljudje tudi lažje razumemo, zato je njihova uporaba primerna v sistemih, ki zahtevajo poseganje uporabnika pri dodeljevanju ali tolmačenju vrednosti.

V nadaljevanju si bomo pogledali nekaj primerov uporabe mehke logike za predstavitev nezanesljivosti pri modeliranju uporabnika.

4.4.2.1 KNOME

Sistem UNIX CONSULTANT [Chin 1989], ki v naravnem jeziku svetuje in pomaga pri uporabi operacijskega sistema Unix, izvaja modeliranje uporabnika preko komponente KNOME. Tako med interakcijo z uporabnikom sistem posodablja in vzdržuje model uporabnika, ki ga potem uporablja pri oblikovanju pomoči, predvsem stopnje podrobnosti razlage glede na znanje, ki ga ima uporabnik.

Komponenta KNOME uporablja mehka pravila za sklepanje o uporabnikovi stopnji znanja in za njeno posodabljanje. Verjetnosti poznavanja konceptov in spremembe verjetnosti so predstavljene v obliki jezikovnih spremenljivk z devetimi diskretnimi vrednostmi, katerim ustreza obseg od –4 do +4 (od *napačno* (–4) preko *negotovo* (0) do *pravilno* (+4)). Uporabnik lahko pripada štirim stopnjam strokovnega znanja (od *začetnik* do *izkušen uporabnik*), konceptom pa pripadajo tri težavnostne stopnje (*enostaven*, *srednji*, *zapleten*) in posebna kategorija (*poseben*), ki označuje težjo predvidljivost poznavanja koncepta. Iz štirih stopenj znanja uporabnika in štirih težavnostnih stopenj koncepta lahko sestavimo 16 mehkih pravil ČE–POTEM, ki napovedujejo znanje uporabnika. Primer takega pravila je:

ČE uporabnik ***U*** je *začetnik* IN koncept ***C*** je *zapleten*,
POTEM je *napačno*, da uporabnik ***U*** pozna koncept ***C***.

Poleg omenjenih pravil pa imamo v sistemu še 32 mehkih pravil za ugotavljanje stopnje znanja, ki jo je dosegel uporabnik, na podlagi prikazanega znanja (16 pravil) oziroma pomanjkanja znanja (16 pravil). Primer takega pravila pri pomanjkanju znanja je:

ČE koncept ***C*** je *enostaven* IN uporabnik ***U*** ne pozna koncepta ***C***,
POTEM je *napačno*, da je uporabnik ***U*** *izkušen uporabnik*.

Tako dobljeni *napačno*, kateremu ustreza na lestvici vrednost –4, se prišteje trenutni verjetnosti, da je uporabnik *izkušen uporabnik*.

Stopnja znanja uporabnika je torej določena s četverčkom vrednosti za pripadnost posamezni kategoriji. Začetne mehke vrednosti so določene z vektorjem (*negotovo, nekoliko verjetno, negotovo, negotovo*) oziroma (0,1,0,0) po lestvici ustreznih vrednosti. Med uporabo sistema se te vrednosti spreminjajo glede na akcije uporabnika (po zapisanih mehkih pravilih). Ko sistem dokončno sprejme eno hipotezo o uporabnikovi stopnji znanja (npr. uporabnik je *začetnik*, torej dobi *začetnik* vrednost *pravilno* ali +4), se vse ostale hipoteze zavrnejo (vse ostale stopnje znanja dobijo vrednost *napačno*) in nobena bodoča akcija uporabnika tega ne more več spremeniti (v dani seji). Seveda se pri vsaki uporabi sistema (za vsako novo sejo) model uporabnika zgradi na novo.

4.4.2.2 ML–Modeler in Sherlock II

ML–Modeler [Gürer 1995] je komponenta za modeliranje uporabnika v prilagodljivem sistemu za učenje kemije. Omogoča sklepanje o uporabnikovem znanju na osnovi njegovih akcij. ML–Modeler primerja postopek uporabnikove rešitve problema s postopkom ekspertove rešitve in tako ugotovi, po katerih metodah je posegel uporabnik, da je dosegel trenutno stopnjo znanja. Hkrati tudi sestavi možne hipoteze o napačnih predstavah in napakah, ki jih je naredil uporabnik. Te hipoteze zajemajo tudi napačno uporabo analogije, posploševanja in specializacije.

Za modeliranje uporablja ML–Modeler mrežno predstavitev, ki vključuje abstraktne koncepte in relacije ter strategije reševanja problemov. Znanje je torej zajeto v mreži, ki jo tvorijo postopki, izkušnje in koncepti. Učenje ustreza dodajanju novih struktur v mrežo, posploševanje pa združevanju struktur ter reorganizaciji struktur v mreži. Sistem uporablja isto predstavitev tako za statično ekspertovo znanje kot za dinamično uporabnikovo znanje (prekrivni model).

Jedro modela uporabnika sestavljajo najbolj verjetno znanje in učni mehanizmi, ki jih je prikazal uporabnik. Za izbiro hevristike in konceptov, ki najbolje razlagajo (napovedujejo) uporabnikovo obnašanje, se uporablja mehka logika. Pri tem je nezanesljivost predstavljena s pripadnostjo mehkim množicam.

ML–Modeler uporablja sedem vrednosti (od *sigurno ne* do *sigurno ja*) za mehke verjetnostne porazdelitve, ki so pripete vsakemu konceptu in povezavi v mreži modela uporabnika. Mehko verjetnostno porazdelitev tako zapišemo z vektorjem (p_1, p_2, p_3, p_4, p_5, p_6, p_7), kjer je p_1 verjetnostna masa, ki pripada vrednosti *sigurno ne*, in p_7 verjetnostna masa za vrednost *sigurno ja*. Začetne vrednosti mehkih verjetnostnih porazdelitev

so odvisne od načina dodajanja v model uporabnika. Tako so koncepti in povezave, ki so dodani v mrežo med tolmačenjem uporabnikovih korakov k rešitvi, nagnjeni proti zgornji polovici, na primer (0,0,0,0,10,30,60). Koncepti, ki so dodani kot rezultat možne hipoteze, pa so na začetku nedoločeni, torej s porazdelitvijo $(\frac{1}{7},\frac{1}{7},\frac{1}{7},\frac{1}{7},\frac{1}{7},\frac{1}{7},\frac{1}{7})$. Verjetnostne porazdelitve za koncepte lahko posodabljamo (spreminjamo navzgor ali navzdol) z uporabo pravil, ki premaknejo verjetnostne mase proti bolj verjetnemu (pri spreminjanju navzgor) ali manj verjetnemu (pri spreminjanju navzdol) delu porazdelitve. Pravila za spreminjanje imajo obliko:

spreminjanje navzgor		spreminjanje navzdol
$p_1 = p_1 - p_1 c$		$p_1 = p_1 + p_2 c$
$p_i = p_i - p_i c + p_{i-1} c$	in	$p_i = p_i - p_i c + p_{i+1} c$
$p_7 = p_7 + p_6 c$		$p_7 = p_7 - p_7 c$

kjer je c konstanta, ki nadzoruje hitrost spremembe.

Mehke metode in pravila sklepanja, ki jih uporablja ML–Modeler, so povzete po metodah sistema Sherlock II [Katz 1992]. Tu ima vektor mehkih verjetnostnih porazdelitev p pet vrednosti, od *nobeno znanje* do *popolno znanje*. Pravila posodabljanja vektorja so podana z vektorjem prestavitve $v = (v_1, v_2, v_3, v_4, v_5)$ in odstotkom spremembe c. Vektor v nadzoruje hitrost posodabljanja (hitre ali počasne spremembe). Konstanta c se uporablja za nadzor vzroka spremembe, tako da močni, a redki kazalniki spremenijo vrednosti hitreje, šibki in pogosti pa počasneje. Pravila imajo obliko:

spreminjanje navzgor		spreminjanje navzdol
$p_1 = p_1 - p_1 v_1 c$		$v_1 = 0$
$p_i = p_i - p_i v_i c + p_{i-1} v_{i-1} c$	in	$p_i = p_i - p_i v_i c + p_{i+1} v_{i+1} c$
$v_5 = 0$		$p_5 = p_5 - p_5 v_5 c$

Začetne vrednosti vektorja mehkih verjetnostnih porazdelitev so podane z enakomerno porazdelitvijo (vsaka vrednost je $\frac{1}{5}$), kar izraža nepoznavanje stanja uporabnikovega znanja. Če pa ima sistem kakršnokoli informacijo v povezavi z uporabnikovim znanjem, se začetne vrednosti nastavijo na tiste vrednosti, ki tako informacijo izražajo.

4.4.2.3 ABITS

Sistem ABITS [Capuano 2000] uporablja mehka števila za ovrednotenje uporabnikovega znanja z upoštevanjem nezanesljivosti (celoten sistem opisuje razdelek 3.5.1). V vsakem

trenutku mora biti sistem sposoben za vsakega uporabnika določiti tako njegovo kognitivno stanje kot učne preference. Te informacije zato tvorijo model uporabnika.

Kognitivno stanje uporabnika opisuje stopnje znanja, ki jih doseže uporabnik, pri vsakem konceptu domene. To informacijo predstavlja niz mehkih števil, po eno število za vsak koncept. Preko uporabe učnega materiala, vaj in testov se ta niz dinamično posodablja. Uporaba mehkih števil izhaja iz potrebe obravnavanja nezanesljivosti pri ocenjevanju uporabnika, saj lahko tako ločimo različne vrste ocenjevanja z različnimi stopnjami zanesljivosti. Kadar uporabnik prebere nek učni material, ki opisuje določeno množico konceptov, lahko sistem sklepa, da se je rahlo povečalo uporabnikovo kognitivno stanje pri teh konceptih, vendar z veliko stopnjo nezanesljivosti. Če pa uporabnik pravilno odgovori na neko testno vprašanje, je pri povečanju njegovega kognitivnega stanja v povezavi z vpletenimi koncepti veliko manjša stopnja nezanesljivosti. Stopnjo (ne)zanesljivosti predstavi sistem z ožjimi (širšimi) mehkimi števili (ožja/širša pripadnostna funkcija). Sistem upošteva tudi pozabljanje že naučenega po določenem času, kar modelira s funkcijo pozabljanja, ki jo uporabi nad kognitivnim stanjem. Funkcija ne zmanjša same stopnje znanja, temveč razširi amplitudo mehkih števil, kar pomeni, da so te ocene manj zanesljive.

Druga komponenta modela uporabnika, učne preference, zajema vse informacije o uporabnikovih zmožnostih dojemanja. Te povejo, za katere vrste virov je uporabnik bolj dojemljiv. Tu ločimo vrste medijev, pedagoški pristop, stopnjo interaktivnosti, semantično zgoščenost in težavnost. Vsaka od navedenih petih učnih preferenc ima svoje vrednosti, katerih ocena primernosti za danega uporabnika je tudi določena z mehkim številom. Ocene učnih preferenc se posodabljajo glede na uporabnikovo obnašanje v sistemu.

4.4.2.4 Hypernet

Modeliranje uporabnika v sistemu Hypernet [Mullier 1999b] temelji na nevronskih mrežah, vendar pa njihovo uporabo kombinira tudi z mehkimi množicami in mehkimi pravili. Sistem je opisan v razdelku 3.5.9, tu se bomo osredotočili le na komponento za modeliranje uporabnika.

Model uporabnika hrani informacije o uporabniku, predvsem njegove zmožnosti in zanimanja. Zmožnosti uporabnika se določajo na osnovi njegove uspešnosti pri opravljanju zadanih nalog, zanimanja uporabnika pa sistem ugotovi iz njegovega premikanja po semantični mreži. Tako lahko sistem izlušči informacije o uporabniku brez neposrednih dialogov z njim. Stopnja zmožnosti vpliva na število povezav, ki so prikazane uporabniku. Z njegovim

napredovanjem se povečuje tudi število prikazanih povezav. Uporabnikova zanimanja vplivajo na vrsto prikazanih povezav (katere povezave se prikažejo uporabniku).

Model sestavljajo štiri komponente: razpoznavalec vzorcev brskanja, baza mehkih pravil, nadzornik učenja in zapis o uporabnikovi izkušenosti.

Razpoznavalec vzorcev brskanja prepoznava različne strategije brskanja iz uporabnikovih premikov med uporabo (brskanjem) hipermedijev. Za prepoznavanje uporablja nevronsko mrežo, saj le-te zelo dobro delujejo tudi na nenatančnih podatkih. Razpoznana strategija brskanja, ki je izhod te komponente, je kombinacija več nižjenivojskih brskalnih konstruktov (klin, pot, obroč ali zanka), katerim razpoznana strategija ustreza do neke mere. Zato jo lahko zapišemo z delnimi pripadnostmi več mehkim množicam strategij.

Baza mehkih pravil uporablja tako imenovan nevro-mehki pristop, to je kombinacijo nevronskih mrež in mehkih pravil, za povezavo razpoznanih (mehkih) strategij brskanja z nalogami in zmožnostmi uporabnika. Vsaka razpoznana strategija brskanja ima določene njene pripadnosti posameznim mehkim množicam strategij. Te potem baza mehkih pravil preslika v uporabnikove zmožnosti ob uporabi dane strategije (ki jih določa nadzornik učenja) in v naloge, ki jih je uporabnik izvrševal. Vendar pa neposredna preslikava strategije brskanja v zmožnosti in naloge ni primerna, ker lahko uporabniki za podobne naloge uporabljajo različne strategije brskanja in obratno. Zato je potrebno opisati nek vsesplošni trend preko daljšega časa in več uporabnikov. Na začetku je podanih le nekaj predpostavk, kako so posamezne strategije brskanja povezane z zmožnostjo uporabnika. Med delom pa baza mehkih pravil lahko spremeni interpretacije posameznih strategij brskanja glede na izkušnje, ki jih dobi pri opazovanju uporabnikov. Baza mehkih pravil torej deluje na podoben način kot nevronske mreže in se prilagaja zapletenemu in spreminjajočemu se okolju med samo uporabo sistema. Vendar pa pri tem uporablja mehka pravila, ki jih lahko razumejo tudi ljudje.

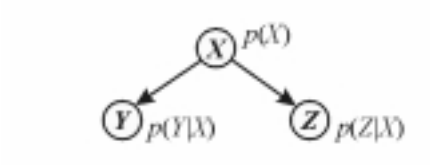
Tudi nadzornik učenja temelji na nevronski mreži. Uporablja se za razvrščanje uporabnika v stopnje zmožnosti na podlagi interakcije s sistemom. Tu gre predvsem za interakcijo s tako imenovanimi učnimi vozlišči, ki od uporabnika zahtevajo odgovore na vprašanja ali pa izvršitev določene naloge, ki zahteva brskanje po hipermedijih. Nadzornik učenja oceni uporabnika po stopnji zmožnosti glede na njegov učinek.

Zapis o uporabnikovi izkušenosti uporablja prekrivni model za izgradnjo zgodovine vozlišč v semantični mreži, ki jih je uporabnik obiskal. Nanj lahko gledamo kot na opis uporabnikovih zanimanj. Ko uporabnik obišče vozlišče, se njegov razred shrani v zapis o

uporabnikovi izkušenosti skupaj s podrazredi do določene globine. Razredu se pripiše neka vrednost, podrazredom pa ustrezno manjša vrednost. Vsakič, ko uporabnik obišče vozlišče, se postopek ponovi, pripisane vrednosti pa se seštevajo. Vsem razredom, ki niso relevantni za trenutno vozlišče, pa se vrednosti odštevajo, kar predstavlja uporabnikovo spreminjanje zanimanja.

4.4.3 Bayesove mreže

Bayesove mreže [Pearl 1988], njihova uporaba in iz nje izhajajoče sklepanje temeljijo na tradicionalni verjetnostni teoriji. Bayesova mreža (ali verjetnostna mreža) je usmerjen acikličen graf, katerega vozlišča predstavljajo naključne spremenljivke, povezave pa ustrezajo relacijam pogojnega vpliva. Relacije med naključnimi spremenljivkami niso nujno vzročne tudi po naravi. Tako v grafu obstaja povezava od *X* do *Y*, če je *Y* odvisen od *X* (glej sliko 4.15). Vozlišče, iz katerega prihaja povezava v dano vozlišče, imenujemo *oče* danega vozlišča (na sliki 4.15 je *X* oče od *Y*). Podobno imenujemo *sin* tisto vozlišče, v katerega vodi povezava iz danega vozlišča (na sliki 4.15 je *Y* sin od *X*).



Slika 4.15
Poenostavljena Bayesova mreža s tremi vozlišči

Na začetku se vsem vozliščem, ki nimajo očetov, pripišejo neke apriorne (začetne) verjetnosti. Vsakemu vozlišču z očeti pripada tudi tabela pogojnih verjetnosti, ki določa količino vpliva njegovih očetov na to vozlišče. Povezava pogojne odvisnosti za vse možne kombinacije vrednosti spremenljivk očetov in sinov določa, kako verjetna je neka vrednost sina pri danih vrednostih očetov (glej vozliščem pripisane verjetnosti na sliki 4.15). Te pogojne verjetnosti lahko dobimo iz empiričnih podatkov, lahko jih oceni strokovnjak domene ali pa temeljijo na splošnejših teorijah o povezanosti spremenljivk določenih tipov.

S pomočjo razširjanja navzdol se izračunajo posteriorne (nove) vrednosti spremenljivk sinov. Pri tem uporablja sistem apriorne verjetnosti in tabele pogojnih verjetnosti.

Bayesove mreže lahko uporabljamo za dve vrsti sklepanja. Napovedno sklepanje je sklepanje iz vzrokov na posledice. Tu se uporablja razširjanje navzdol. Na primeru mreže s slike 4.15 bi za tako sklepanje o verjetnosti *Z* uporabili enačbo:

$$p(Z) = p(Z \mid X)\,p(X)\,.$$

Vrednosti v mreži pa se lahko spreminjajo tudi v obratni smeri, z razširjanjem navzgor. V tem primeru sistem sklepa iz opazovanih učinkov na možne vzroke, kar imenujemo diagnostično sklepanje. Primer takega sklepanja je posteriorna verjetnost *p** za *X* pri opazovanem dejstvu *Z*, ki se lahko izračuna preko Bayesovega teorema [Bayes 1763]:

$$p^*(X) = p(X \mid Z) = \frac{p(X)\,p(Z \mid X)}{p(Z)}$$

Pri sklepanju iz več spremenljivk predpostavimo njihovo pogojno neodvisnost in s tem poenostavimo računanje. Torej za spremenljivke s slike 4.15 velja:

$$p(Y, Z \mid X) = p(Y \mid X)\,p(Z \mid X)\,.$$

Kadar uporabljamo Bayesove mreže za modeliranje uporabnika, lahko spremenljivke predstavljajo različne lastnosti uporabnika, kot so njegovo znanje, dosežena stopnja ali pa njegova sposobnost rešitve problema. Tak pristop omogoča tudi obravnavo nezanesljivosti preko pogojnih verjetnosti vsakega vozlišča pri različnih stanjih njegovih očetov.

Bayesove mreže so uporabne predvsem v sistemih, ki jih sestavljajo komponente z medsebojnim vzročnim vplivom. Nudijo zelo močno osnovo za sklepanje, tako za napovedovanje izhoda v odvisnosti od vzrokov kot za razlaganje opazovanega izhoda v povezavi s spremenljivkami, ki so ga povzročile.

4.4.3.1 OLAE, POLA in Andes

Sistem OLAE [Martin 1995] je orodje za ocenjevanje uporabnikovega znanja. Sistem zbira informacije iz uporabnikovega reševanja fizikalnih problemov (učna domena je uvodna fizika), jih analizira z verjetnostnimi metodami ter tako ugotovi znanje, ki ga uporablja uporabnik. Ker sistem le ocenjuje uporabnikovo znanje (določa, katere dele domene uporabnik pozna) in ne nudi nobene podpore interaktivnemu poučevanju, se sistem odzove šele takrat, ko uporabnik reši problem. Za vsak posamezen problem sistem samodejno sestavi Bayesovo mrežo, ki povezuje znanje (v obliki pravil) z določenimi akcijami (kot so na primer napisane enačbe). Tako zgrajeno mrežo uporabi sistem za opazovanje uporabnikovega obnašanja in

izračun verjetnosti, da uporabnik pozna in uporablja vsako od pravil. Bayesova mreža ima štiri vrste vozlišč (za pravila, za uporabo pravil, za dejstva in za akcije), ki jih povezujejo usmerjene povezave. Različne poti skozi mrežo predstavljajo različne načine reševanja danega problema. Ko uporabnik reši problem, se ustvarijo vozlišča za akcije s pripeto vrednostjo resnično, sistem pa uporabi razširitvene algoritme, ki razširijo to informacijo preko povezav v mreži (posodobijo verjetnosti v posameznih vozliščih) in tako določijo nove verjetnosti za uporabnikovo poznavanje pravil.

Sistem POLA [Conati 1996] je razširitev sistema OLAE, ki izvaja verjetnostno sledenje znanja, na sistem, ki uporablja verjetnostno sklepanje za izvajanje tako sledenja znanja kot sledenja modela. Tako sistem POLA ne ugotavlja le pravil, ki jih uporabnik pozna, temveč tudi pot, ki ji uporabnik sledi pri reševanju problema. To pomeni, da je v primeru, ko obstaja več z uporabnikovimi akcijami skladnih poti k rešitvi problema, sistem zmožen določiti, katera pot je najbolj verjetna, da jo je ubral uporabnik. S tem dobi sistem nove zmožnosti, kot so odgovarjanje na uporabnikova vprašanja, pomoč uporabniku med delom v obliki nasvetov ali izbira pedagoških odločitev. Sistem POLA sestavi prostor rešitev problemov in ga predstavi z grafom rešitev. Graf rešitev je AND/OR graf, ki se samodejno sestavi iz baze znanja produkcijskih pravil in nudi zgoščeno predstavitev vseh razpoložljivih rešitev problema. Iz tega grafa rešitev in iz uporabnikovih akcij se postopoma zgradi Bayesova mreža, ki potem ustvari predvidevanja o rešitvi, ki ji sledi uporabnik. Na koncu reševanja problema poda mreža tudi oceno uporabnikove stopnje znanja, ki je zajeto v rešitvi problema.

Sistem Andes [Gertner 1998] je inteligentni učni sistem za poučevanje Newtonove fizike. Njegova komponenta za modeliranje uporabnika temelji na sistemu POLA. Tako uporablja Bayesovo mrežo za izračun verjetnostne ocene treh vrst informacij: uporabnikovega splošnega znanja o fiziki, uporabnikovega posebnega znanja o trenutnem problemu ter abstraktnih načrtov, ki jim uporabnik lahko sledi pri reševanju problema. Z uporabo takega modela uporabnika lahko sistem sklepa, na katerem delu rešitve uporabnik trenutno dela in kje se mu je morda zataknilo. Zato lahko nudi nasvete in pomoč po meri uporabnikovega znanja in ciljev.

4.4.3.2 HYDRIVE

Sistem HYDRIVE [Mislevy 1995] je inteligentni učni sistem za iskanje in odstranjevanje tehničnih motenj (napak) v letalskem hidravličnem sistemu. Sistem modelira uporabnikovo sposobnost diagnosticiranja napak, sklepanje pri posodabljanju modela uporabnika pa je osnovano na Bayesovi mreži.

Sistem najprej predstavi uporabniku opis problema v delovanju hidravličnega sistema, uporabnikova naloga pa je določiti napako. Pri tem lahko uporablja običajne postopke in tudi priložen tehnični material. Obnašanje uporabnika opazuje sistem in ocenjuje njegovo delovanje, pri tem pa uporablja vse razpoložljive informacije za določanje akcij, ki omogočajo ugotavljanje težave. Sistem oceni kakovost dejanskih diagnostičnih akcij uporabnika in označi njegovo znanje z uporabo splošnih spremenljivk, kot so znanje sistema ter uporabljene strategije in procedure. Znanje uporabnika torej določajo trije deli: poznavanje sistema, poznavanje strategij in poznavanje postopkov rešitve. Vsako od navedenih znanj se nadalje deli na več vozlišč in spremenljivk. Taka uporaba več nivojev znanja omogoča sistemu dovolj natančno diagnozo pri določanju pomanjkljivih delov znanja uporabnika in hkrati dovolj splošno za izbiro osnovnih poučevalnih strategij (npr. na odgovor uporabnika se odzove s predstavitvijo pomoči ali s predlogom neke bolj zapletene situacije).

Sistem HYDRIVE uporablja kombiniran način sklepanja o uporabnikovem znanju. S pomočjo pravil določi strategijo, ki jo uporablja uporabnik, in oceni akcije. Posplošene ugotovitve o uporabnikovi sposobnosti pa temeljijo na verjetnosti (Bayesova mreža).

4.4.3.3 APHID–2

Sistem APHID–2 [Kettel 2000] je izpopolnjena različica sistema APHID (sistem je opisan v razdelku 3.5.5) z dodanim modelom uporabnika, ki prekriva model domene z Bayesovo mrežo zaupanja. Tako sistem APHID–2 gradi individualizirane hipermedijske aplikacije z upoštevanjem modela uporabnika.

Model uporabnika je prekrivni model preko konceptne mape domene, ki zajema vse koncepte domene in njihove medsebojne relacije. Vsak domenski koncept ima pripisane tri atribute: njegovo identifikacijsko številko, uporabnikovo stopnjo znanja koncepta in začetno verjetnost, ki se uporabi v Bayesovi mreži. Poleg tega ima koncept tudi tri elemente: ime, seznam vhodnih konceptov in seznam izhodnih konceptov. Seznam vhodnih konceptov je spisek vseh konceptov, ki v Bayesovi mreži kažejo na dani koncept. Seznam izhodnih konceptov pa je spisek vseh konceptov, na katere v Bayesovi mreži kaže dani koncept. Seznam vhodnih konceptov ima pripisano tudi polno verjetnostno porazdelitev, ki označuje stopnje vpliva predhodnih vozlišč na dano vozlišče. Taka koncepta mapa se potem uporabi za izgradnjo Bayesove mreže zaupanja, ki posodablja model uporabnika in sklepa o njegovem znanju (Bayesovo sklepanje).

Bayesova mreža zaupanja je graf, kjer vozlišča predstavljajo koncepte, povezave pa predpogojne relacije med njimi (graf temelji na strukturi konceptne mape). Če za dva koncepta *A* in *B* velja, da je *A* predpogoj za *B*, potem lahko relacijo med njima opišemo na naslednji način:

p(uporabnik-pozna(*B*)|¬uporabnik-pozna(*A*)) = 0.

To lahko preberemo kot verjetnost, da uporabnik pozna koncept *B*, če vemo, da uporabnik ne pozna koncepta *A*, je nič. Kako uporabnikovo poznavanje koncepta *A* spremeni zaupanje v uporabnikovo poznavanje koncepta *B*, pa določa vrednost naslednje verjetnosti:

p(uporabnik-pozna(*B*)|uporabnik-pozna(*A*)).

Pri tem velja, da je verjetnost večja pri tesnejši povezavi med obema konceptoma. Tako z uporabo Bayesove mreže zaupanja sestavimo model uporabnika, ki ne določa le uporabnikove stopnje poznavanja vsakega koncepta temveč tudi njegovo razumevanje relacij med koncepti.

Poleg stopnje uporabnikovega znanja o vsakem konceptu domene vključuje model uporabnika tudi oceno stopnje uporabnikovega splošnega znanja (*začetnik*, *napredni uporabnik*, *izkušen uporabnik*). Pomembne pa so tudi ostale uporabnikove preference, kot so njegovi učni cilji, priljubljen način učenja, nagnjenost k tekstovni ali vizualni predstavitvi, želja po učenju in podobne.

4.4.3.4 KBS Hyperbook

Sistem KBS Hyperbook [Henze 2000a, Henze 1999] je orodje za modeliranje, organizacijo in vzdrževanje odprtih hipermedijskih sistemov v svetovnem spletu. Sam sistem smo opisali v razdelku 3.5.13, tu bomo podrobneje pogledali le njegovo komponento za modeliranje uporabnika.

Model uporabnika sestavljajo tri komponente: vektor znanja, model učnih odvisnosti in mehanizem sklepanja.

Vektor znanja *KV* modelira uporabnikovo poznavanje elementov znanja domene. Vsaka komponenta vektorja je pogojna verjetnost, ki opisuje oceno sistema, da uporabnik ***U*** pozna element znanja KI_i, $i \in [1, n]$, na osnovi opazovanj *E* uporabnikovega dela s sistemom:

$$KV(\mathbf{U}) = (P(KI_1 \mid E), P(KI_2 \mid E), \ldots P(KI_n \mid E)) .$$

Tudi opazovanje uporabnikovega dela s sistemom je izraženo s pomočjo elementov znanja, saj vsako opazovanje določi tudi oceno uporabnikovega poznavanja danega elementa znanja *KI*, izraženo z verjetnostno porazdelitvijo. Uporabnik ima lahko o elementu znanja *strokovno znanje*, *napredno znanje*, *osnovno znanje* ali *začetniško znanje*.

Model učnih odvisnosti temelji na modelu domene in opisuje vse (predpogojne) odvisnosti med elementi znanja. Model vsebuje vse elemente znanja *KI* in mednje uvede relacijo delne urejenosti (<), ki predstavlja učne odvisnosti med elementi znanja. Tako na primer $KI_1 < KI_2$ pomeni, da se mora uporabnik naučiti KI_1 pred KI_2, ker je razumevanje KI_1 predpogoj za razumevanje KI_2.

Mehanizem sklepanja s pomočjo Bayesove mreže izračuna prepričanje sistema o uporabnikovem poznavanju posameznih elementov znanja. Bayesova mreža, ki se uporablja za sklepanje, pravzaprav implementira model uporabnika. Vozlišča mreže so elementi znanja, odvisnosti med njimi pa so izražene s pogojnimi verjetnostmi med elementi znanja. Po vsakem opravljenem opazovanju uporabnika se novo izračunane vrednosti razširijo preko celotne mreže in tako posodobijo model uporabnika.

Predstavitev modela uporabnika z Bayesovo mrežo omogoča tudi obravnavo nezanesljivosti opazovanj sistema. Za opis uporabnikovega znanja se uporablja vektor štirih verjetnostnih vrednosti, ki opisujejo oceno uporabnikovega razumevanja posamezne enote znanja na stopnji odlično (*strokovni uporabnik*), z nekaj težavami (*napredni uporabnik*), z veliko težavami (*osnovni uporabnik*) ali nepripravljen (*začetnik*).

4.4.4 Dempster–Shaferjeva teorija

Dempster–Shaferjevo teorijo evidence uporablja bolj malo sistemov v primerjavi z razširjenostjo Bayesovih mrež. Teorija evidence je primerna predvsem v primerih nezanesljivih opazovanj uporabnikovih akcij. Od verjetnostne teorije se razlikuje predvsem v treh bistvenih stvareh. Funkcije zaupanja so definirane kot funkcije nad množicami, kar pomeni, da lahko zaupanja pripišemo celi množici propozicij. Pri verjetnosti pa gre vedno za verjetnost ene same propozicije. Zaupanje tudi ne ustreza zakonu o polni verjetnosti, saj je vsota zaupanj v propozicijo in njeno negacijo vedno manjša ali enaka ena (pri verjetnosti je vedno enaka ena). Zadnja bistvena razlika pa je v operaciji za kombiniranje evidence iz različnih neodvisnih virov, ki jo lahko uporabimo v teoriji evidence (tako imenovano Dempsterjevo pravilo kombinacije). Z uporabo take operacije lahko združujemo argumente.

Kot primer uporabe teorije evidence za obravnavo nedoločenosti bomo na kratko opisali sistem PHI [Bauer 1995], ki uporabnikom elektronske pošte nudi inteligentno pomoč. Sistem deluje na prepoznavanju uporabnikovih načrtov. Načrti se delijo na osnovne in abstraktne, slednji se lahko izvedejo preko enega od različnih osnovnih načrtov. Taka hierarhija pomeni, da obstajajo naravne podmnožice načrtov, ki izvajajo isti abstraktni načrt. Tako na primer opazovanje uporabnika daje slutiti, da želi uporabnik shraniti sporočila (abstrakten načrt), a ne kaže na to, ali jih namerava napisati ali posneti (osnovna načrta).

Sistem PHI razpoznava načrte uporabnikov na podlagi evidence, ki izhaja iz opazovanj uporabnikovih akcij. Pri tem uporablja Dempster–Shaferjevo teorijo, tako da obdela obstoječo evidenco nad načrti, ki bi jih uporabnik lahko imel (hipoteza). Za predvidenje uporabnikovih namer uporablja sistem kot evidenco tudi informacije o uporabniku, ki so bile zbrane v prejšnjih uporabah sistema. To deluje zelo dobro, ko je število uporab sistema že večje.

Sistem za vsako hipotezo H računa tri mere: osnovno dodelitev $m(H)$, mero zaupanja $Bel(H)$ in mero prepričljivosti $Pl(H)$. Za vsako hipotezo torej obstajajo tri različne mere, ki pojasnjujejo skladnost hipoteze z razpoložljivo evidenco. Zato je odločanje na podlagi rezultatov analize težje in pogosto potrebujemo še dodatne kriterije [Millán 2000].

4.4.5 Primerjava navedenih pristopov

Kot smo omenili že na začetku tega poglavja, je izbira primernega pristopa k obravnavi nedoločenosti odvisna od posameznega primera, saj je vsak izbran pristop dober za neko vrsto nedoločenosti, pri kateri se potem pokažejo tako njegove prednosti kot tudi slabosti.

Faktorji gotovosti so zelo enostavni za razumevanje in omogočajo tudi preprosto realizacijo in implementacijo, saj zahtevajo le malo izračunavanj. Vendar pa jih lahko uporabljamo le pri pravilih, ki so vsa enake vrste (ali vsa napovedna ali pa vsa diagnostična, ne pa pri kombinaciji obeh) [Krause 1993]. Poleg tega morajo biti vse odvisne evidence združene v eno samo pravilo. Zahtevajo bazo pravil, ki je popolna (glede na nek kriterij pokrivanja), ne vsebuje odvečnih pravil (neuporabnih, podvojenih ali vsebovanih pravil) ali nasprotujočih si pravil. Faktorji gotovosti tudi nimajo povsem trdne teoretične osnove in so neskladni, kar ima lahko za posledico nepredvidljivo obnašanje modela, predvsem v primerih, ki niso bili vnaprej pričakovani [Millán 2000].

Mehka logika nudi možnost procesiranja vhodnih podatkov, ki so podani v verbalno nenatančni obliki. Omogoča tudi naraven opis znanja in sklepanja v obliki nenatančnih

konceptov, operatorjev in pravil. Zahteva relativno malo računanja, a lahko naletimo na težave pri določanju pripadnostnih funkcij in izbiri operatorjev za združevanje pravil [Jameson 1996].

Bayesove mreže se pogosto uporabljajo, saj nudijo trdno teoretično osnovo, so konsistentne in hkrati uporabne pri vseh vrstah problemov. Take mreže so zelo močne v sklepanju (tako diagnostičnem kot napovednem), vendar potrebujejo nadroben model (spremenljivke in relacije, vse pripadajoče znanje mora biti zakodirano v modelu) skupaj z ocenami vseh parametrov (pogojnih in apriornih verjetnosti). Zato je računska kompleksnost razširitvenih algoritmov zelo velika (v splošnem je to NP–težek problem [Cooper 1990]), iz tega pa izhajajo tudi problemi pri implementaciji algoritmov. Vendar pa obstaja množica približnih algoritmov, ki občutno zmanjšajo kompleksnost računanja [Millán 2000].

Dempster–Shaferjeva teorija evidence je bolj splošna kot Bayesova verjetnost, saj zaupanja niso nujno aditivna, lahko jih pripišemo množici hipotez in ne nujno le vsaki posamezni hipotezi ter zato omogoča tudi združevanje evidence glede na hierarhično gnezdene množice hipotez. Teorija evidence je semantično bogatejši formalizem kot verjetnostna teorija, ker omogoča tudi izražanje delnega znanja [Krause 1993]. Vendar pa je odločitev na osnovi rezultatov analize pri teoriji evidence bolj zamotana, ker obstajajo za vsako množico hipotez tri različne mere (osnovna dodelitev, mera zaupanja in mera prepričljivosti), ki razlagajo združljivost hipoteze z razpoložljivo evidenco, in zato potrebujemo še dodatne odločitvene kriterije. Omejuje jo tudi dejstvo, da lahko izvaja samo diagnostično sklepanje, ne pa tudi napovednega, ki je včasih zelo uporabno pri modeliranju uporabnika. Njena prednost pa je, da ne zahteva podanih apriornih zaupanj, saj izhaja le iz opazovane evidence. Zato je kompleksnost natančnega računanja sicer še večja (kombiniranje funkcij zaupanja je v splošnem eksponentno), a se tudi tu uporablja različne približne tehnike, ki uspešno rešujejo ta problem.

Določitev ustreznih začetnih parametrov je skupen problem vseh navedenih pristopov. Sicer je najbolj izrazit pri Bayesovih mrežah, kjer moramo poleg strukture mreže določiti tudi pogojne verjetnosti in apriorne verjetnosti, a je prisoten tudi pri Dempster–Shaferjevi teoriji (osnovne dodelitve), mehki logiki (pripadnostne funkcije) in faktorjih gotovosti.

V našem primeru, pri izobraževalnih hipermedijih, je glavni problem, ki ga želimo rešiti, kako v sistemu predstaviti in modelirati znanje uporabnika in pri tem upoštevati in opisati tudi stopnjo (ne)zanesljivosti ocene uporabnikovega znanja. Pri tem ne gre toliko za določanje pripadnosti uporabnika posameznim kategorijam (problem klasifikacije) kot za predstavitev znanja samega. V praksi se podobni problemi rešujejo v glavnem na osnovi teorije verjetnosti,

torej s pomočjo Bayesovih mrež, saj je ta teorija, morda prav zaradi svoje razširjenosti, precej razvita. Njeni največji pomanjkljivosti sta potreba po začetnih apriornih verjetnostnih vrednostih (tudi če o nečem ne vemo nič, moramo dodeliti neke verjetnosti – ponavadi se uporablja kar enakomerno porazdelitev) ter kompleksni algoritmi razširjanja preko pogojnih verjetnosti. Teorija evidence in teorija možnosti sta splošnejši od teorije verjetnosti in se ukvarjata predvsem z negotovostjo pri določanju pripadnosti posamezni množici, kar je pravzaprav problem klasifikacije. Zato nobena od njiju ni najbolj primerna za reševanje našega problema.

Na osnovi podrobnega pregleda različnih pristopov k reševanju problema predstavitve uporabnikovega znanja z določeno mero nezanesljivosti smo se odločili, da bomo rešitev poiskali v teoriji mehkih množic. Ta je namreč kot nalašč za probleme, kjer negotovost izhaja iz nejasnih oziroma neostrih mej med posameznimi množicami (za opis meglenosti). Tudi uporabnikovo znanje vključuje veliko meglenosti, saj je meja med znanjem in neznanjem zelo neostra. Zato je prikladen opis uporabnikovega poznavanja posameznega koncepta domene s pomočjo mehkih množic. Poleg tega je teorija mehkih množic tudi računsko manj zahtevna.

5. Prototip prilagodljivega sistema

To poglavje opisuje naš pristop k prilagodljivi izobraževalni hipermediji. Najprej bomo opisali način modeliranja učne domene in uporabnika, pri čemer smo upoštevali tudi nezanesljivost pri določanju uporabnikovega znanja, kar smo vgradili v sam model uporabnika. Sledi opis prilagoditvenih zmožnosti hipermedijskega sistema, na koncu pa predstavimo tudi prototip hipermedijskega sistema, ki je bil razvit v okviru te naloge.

Ker gre tu za splošen izobraževalni sistem, ki je razvit neodvisno od same domene učenja, je sistem v bistvu le ogrodje ali lupina, ki omogoča določeno funkcionalnost. Domeno učenja se določi naknadno, pripravljeno učno vsebino pa se vgradi v izdelano lupino sistema.

5.1 Modeliranje učne domene

Učno domeno lahko logično razdelimo na več delov, ki jih imenujemo *koncepti*. Koncept je tako osnovni delec znanja dane domene in predstavlja (večji ali manjši) del domenskega znanja. Skupno število konceptov je odvisno od dane učne domene in od izbrane zrnatosti. Dano domeno učenja namreč lahko predstavimo z večjim številom enostavnih konceptov ali pa z manjšim številom bolj kompleksnih konceptov. Na ta način lahko učno domeno zapišemo kot končno množico $C = \{c_1, c_2, \ldots c_n\}$, katere elementi so posamezni domenski koncepti, moč množice (število elementov v množici) pa je $|C| = n$.

Tako določeni koncepti domene pa medsebojno niso povsem neodvisni. Zato lahko mednje uvedemo relacijo urejenosti, ki predstavlja učne odvisnosti med koncepti. To relacijo bomo poimenovali *predpogojna relacija* in jo označili z $\prec$. Za poljubna koncepta c_1 in c_2 velja, da sta v predpogojni relaciji $c_1 \prec c_2$, kadar se mora uporabnik naučiti koncept c_1 pred konceptom c_2, ker je za razumevanje (poznavanje) koncepta c_2 potrebno razumevanje (poznavanje) koncepta c_1. V tem primeru rečemo, da je koncept c_1 *predpogojni koncept* za koncept c_2. Seveda nek koncept ne more biti predpogoj samemu sebi, kot tudi dva koncepta ne moreta biti predpogoj drug drugemu. Predpogojna relacija je torej antirefleksivna, strogo antisimetrična in tranzitivna, saj velja:

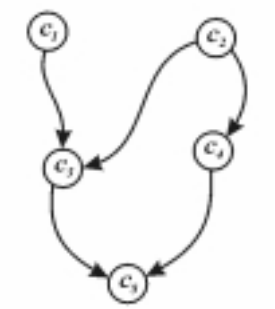
$\forall c_i \in C: \quad c_i \not\prec c_i$,
$\forall c_i, c_j \in C: \quad c_i \prec c_j \Rightarrow c_j \not\prec c_i$ in
$\forall c_i, c_j, c_k \in C: \quad (c_i \prec c_j) \wedge (c_j \prec c_k) \Rightarrow c_i \prec c_k$.

Če imamo koncepte c_1, c_2 in c_3, za katere velja $c_1 \prec c_2$ in $c_2 \prec c_3$, lahko zaradi tranzitivnosti relacije sklepamo tudi na povezanost $c_1 \prec c_3$. Vendar se zadnja relacija nekoliko razlikuje od prvih dveh, saj smo do nje prišli posredno preko koncepta c_2. Zato bomo koncept c_1 imenovali *posredni predpogoj* za koncept c_3, za razliko od koncepta c_2, ki je njegov *neposredni predpogoj*. V nadaljevanju se bo povsod, kjer to ne bo izrecno izraženo, pojem predpogoj nanašal na neposredne predpogoje.

Glede na predpogojno relacijo lahko vse koncepte domene razdelimo na tri skupine. V prvi so koncepti, ki nimajo nobenih predpogojev. Imenovali jih bomo *osnovni koncepti*. Drugo skupino sestavljajo koncepti, ki niso predpogoj nobenemu drugemu konceptu. Te bomo imenovali *končni koncepti*. Tretjo skupino pa sestavljajo vsi ostali koncepti, ki imajo določene predpogoje in so hkrati tudi predpogoji drugim konceptom. Zaradi te njihove lastnosti, da se nahajajo med osnovnimi in končnimi koncepti, jih bomo imenovali *vmesni koncepti*. Za končne koncepte domene lahko rečemo, da so cilj učenja v tečaju. Ko se namreč uporabnik nauči vse končne koncepte, mora poznati tudi vse vmesne in osnovne koncepte (ki so njihovi predpogoji) in s tem pozna tudi celotno domeno učenja.

Koncepte domene skupaj z medsebojnimi relacijami lahko predstavimo z usmerjenim acikličnim grafom, s katerim tudi modeliramo domeno. Tak graf imenujemo *graf domene*. Vozlišča grafa predstavljajo koncepte domene, povezave pa relacije med njimi. Graf lahko zapišemo kot $G = (C, R)$, kjer množica domenskih konceptov C ustreza množici vozlišč grafa, množico povezav grafa pa predpisuje predpogojna relacija $R \subseteq C \times C$, to je množica urejenih parov (c_i, c_j), za katere velja $c_i \prec c_j$ pri $c_i, c_j \in C$. Ker je predpogojna relacija R antirefleksivna, strogo antisimetrična in tranzitivna, je graf domene usmerjen in acikličen.

Primer takega grafa domene prikazuje slika 5.1, kjer je $C = \{c_1, c_2, c_3, c_4, c_5\}$ in $R = \{(c_1, c_3), (c_2, c_3), (c_2, c_4), (c_3, c_5), (c_4, c_5)\}$. Koncepta c_1 in c_2 sta osnovna koncepta (v vozlišče grafa ne pride nobena povezava), koncept c_5 je končni koncept (iz vozlišča grafa ne izhaja nobena povezava), koncepta c_3 in c_4 pa sta vmesna koncepta (vozlišče grafa ima tako vhodne kot izhodne povezave).



Slika 5.1
Primer grafa domene $G = (C, R)$

Podobno kot Gagné v svoji teoriji načrtovanja poučevanja [Gagné 1988] loči dve vrsti predpogojev za različne vrste rezultatov učenja, bomo tudi mi uporabili delitev predpogojnih konceptov na dve vrsti. Vsi predpogojni koncepti sicer omogočajo ali pomagajo pri učenju določenega koncepta, a se lahko razlikujejo po svoji potrebnosti. Tako so *nujni predpogojni koncepti* nekega koncepta bistvenega pomena za razumevanje tega koncepta. Uporabnik se jih lahko nauči prej ali kasneje v tečaju, vendar morajo nujno biti poznani pred učenjem koncepta, katerega predpogoji so. Po drugi strani pa *podporni predpogojni koncepti* nekega koncepta le pomagajo pri učenju tega koncepta ter tako omogočajo njegovo lažje in hitrejše osvajanje.

Po taki delitvi predpogojnih konceptov lahko množico relacij *R* grafa domene razdelimo na dve tuji si podmnožici *E* in *S*, za kateri velja: $R = E \cup S$ in $E \cap S = \emptyset$. Množica *E* je množica vseh *nujnih* (angl. *essential*) predpogojnih relacij, ki povezujejo nujne predpogojne koncepte z danim konceptom. Množica *S* pa je množica vseh *podpornih* (angl. *supportive*) predpogojnih relacij, ki povezujejo vse podporne predpogojne koncepte z danim konceptom.

Ker povezave v grafu domene predstavljajo splošno predpogojno relacijo med dvema konceptoma, bomo tem povezavam dodali še uteži, s pomočjo katerih bomo razlikovali med nujnimi in podpornimi predpogojnimi relacijami. Za uteži v grafu lahko rečemo, da opisujejo jakost relacije, to je trdnost povezave, kar lahko enačimo z nujnostjo ali potrebnostjo predpogoja.

Vsakemu elementu množice *R* torej pripišemo še neko utež *w*, ki določa njegovo jakost. Tako postanejo elementi množice *R* trojice (c_i, c_j, w_{ij}), $i, j \in [1, n]$, kjer je w_{ij} jakost predpogojne relacije $c_i \prec c_j$ (nujnost oziroma potrebnost koncepta c_i kot predpogoja za koncept c_j). Graf domene potem zapišemo s trojico $G = (C, R, W)$, kjer je *C* množica

domenskih konceptov, *W* je utežna funkcija, predpogojna relacija $R \subseteq C \times C$ pa je množica urejenih trojic (c_i, c_j, w_{ij}), za katere velja $c_i \prec c_j$ in $w_{ij} = W((c_i, c_j))$, $c_i, c_j \in C$, kar krajše zapišemo kot $c_i \prec_{w_{ij}} c_j$.

Utežna funkcija *W* (vrednosti uteži w_{ij}) je odvisna od posamezne učne domene in jo določi avtor ob razdelitvi domene na koncepte in določitvi predpogojnih relacij. Mi bomo uporabili funkcijo, ki preslika povezave na interval [0,1], kjer vrednost 0 pomeni podporno predpogojno relacijo, vrednost 1 pa nujno predpogojno relacijo. Ker pa vsi predpogoji niso enako močni (potrebni), je lahko vrednost uteži pri nujnih predpogojnih relacijah tudi manjša od 1. Utežno funkcijo *W* lahko potemtakem zapišemo kot:

$$W((c_i, c_j)) = w_{ij} = \begin{cases} 1, & c_i \prec_{nujni} c_j \\ 0, & c_i \prec_{podporni} c_j \\ x, \quad 0 < x < 1, & c_i \prec_{šibki-nujni} c_j \end{cases}$$

Predpogojne relacije, ki imajo vrednost utežne funkcije w_{ij} med 0 in 1, spadajo med nujne predpogojne relacije med dvema konceptoma, a je povezanost obeh konceptov v takem primeru šibkejša. Na to lahko gledamo tudi kot na delno pripadnost množici nujnih predpogojnih relacij *E*, kjer vrednost uteži w_{ij} določa stopnjo pripadnosti množici: $\mu_E(c_i, c_j) = w_{ij}$. Nujna predpogojna relacija je torej mehka relacija ali, povedano drugače, množica nujnih predpogojnih relacij *E* je mehka množica. Množica podpornih relacij *S* pri tem ostaja trda množica (podporna predpogojna relacija je trda relacija).

Če povzamemo vse skupaj, lahko model domene predstavimo z mehkim grafom domene, ki vključuje dve vrsti povezav: mehko množico nujnih predpogojnih relacij *E* in trdo množico podpornih predpogojnih konceptov *S*. Graf tako zapišemo kot $G = (C, E, S)$, kjer je *C* množica domenskih konceptov, obe relaciji $E \subseteq C \times C$ in $S \subseteq C \times C$ pa imata pri $c_i, c_j \in C$ naslednji pripadnostni funkciji:

$$\mu_E(c_i, c_j) = \begin{cases} w_{ij}, & c_i \prec_{w_{ij}} c_j, \quad w_{ij} > 0 \\ 0, & \text{sicer} \end{cases}$$

in

$$\mu_S(c_i, c_j) = \begin{cases} 1, & c_i \prec_{w_{ij}} c_j, \quad w_{ij} = 0 \\ 0, & \text{sicer} \end{cases}$$

V nadaljevanju bomo za predstavitev domene uporabljali opisani mehki graf.

5.2 Povezava modela domene z učnimi enotami

Posamezen tečaj sestavlja več učnih enot, ki so hierarhično urejene. Vsaka učna enota zajema del učne snovi in opisuje enega ali več domenskih konceptov. Učne enote so shranjene v HTML (kratica iz angl. HyperText Markup Language, de facto standard za objavljanje v spletu) datotekah na poljubnih spletnih strežnikih. Ker je lokacija teh datotek poljubna, lahko v tečaj vstavimo tudi katerekoli zunanje spletne datoteke, ki ustrezajo našim potrebam, pri njihovi uporabi pa ni nobene razlike napram datotekam na lokalnem strežniku. Za vsako učno enoto moramo poznati njeno lokacijo, ki jo enolično določa njen spletni naslov – URL (kratica iz angl. Universal Resource Locator).

Učno enoto lahko sestavlja več delov, kot so na primer uvod, razlaga, povzetek, primer, vaja ali test, odvisno od njenega namena in zgradbe. Zaenkrat bomo v naši aplikaciji uporabljali le razlago in test. Razlaga enote opisuje določene koncepte domene in pojasnjuje z njimi povezano učno snov. Test pa poskrbi za preverjanje razumevanja razlage in poznavanja konceptov s pomočjo vprašanj uporabniku.

Vsebino učne enote lahko opišemo z množico konceptov domene. Pri tem velja, da vsaka učna enota opisuje en glavni koncept, poleg njega pa lahko tudi več stranskih konceptov. Podobno tudi vsak koncept domene opisuje natanko ena učna enota kot glavni koncept. Koncepte domene lahko torej preslikamo v učne enote tako, da vsaki enoti ustreza natanko en koncept. Pri tem je pomembna le vsebina učne enote, ne pa njeno mesto v tečaju ali njena lokacija. Omenjeno preslikavo lahko opišemo s funkcijo mapiranja *M*:

$$M : U \to \mathcal{P}(C) \setminus \{\emptyset\},$$

kjer je *U* množica učnih enot, *C* pa množica domenskih konceptov. Mapirna funkcija *M* priredi vsaki učni enoti $u \in U$ neko neprazno podmnožico konceptov $I \subseteq C$, $I \in \mathcal{P}(C) \setminus \{\emptyset\}$, ki opisuje vsebino enote: $I = M(u)$. To podmnožico konceptov *I* imenujemo tudi indeks učne enote, uporabo funkcije *M* pa indeksiranje učne enote po konceptih domene.

Ker vsaka učna enota opisuje natanko en glavni koncept, lahko določimo še funkcijo mapiranja glavnih konceptov M_m, ki vsaki učni enoti priredi en sam domenski koncept (to je glavni koncept enote):

$$M_m : U \to C.$$

Pri tem za vsako učno enoto $u \in U$ velja, da pri indeksu enote $I = M(u)$ obstaja natanko en tak element $c \in I$, da $c = M_m(u)$. In obratno, če je $c = M_m(u)$ in $I = M(u)$, potem je $c \in I$.

Za prirejanje ustreznih konceptov domene posameznim učnim enotam (indeksiranje) poskrbi avtor učne enote ali avtor tečaja. Ta mora za vsako učno enoto določiti natanko en glavni koncept, ki ga enota opisuje, lahko pa doda tudi več stranskih konceptov, ki se pojavijo v enoti. Poleg tega mora za vsako enoto določiti še njen naslov (ime enote), lokacijo datoteke (URL) in enoto ustrezno umestiti v hierarhijo tečaja.

Za zdaj v naši aplikaciji uporabljamo le indeksiranje po glavnih konceptih enot, kar pa ne vpliva bistveno na splošnost modela. Tako ena učna enota opisuje natanko en koncept, vsak koncept pa je opisan v natanko eni učni enoti. Učne enote in koncepti so torej v relaciji ena proti ena.

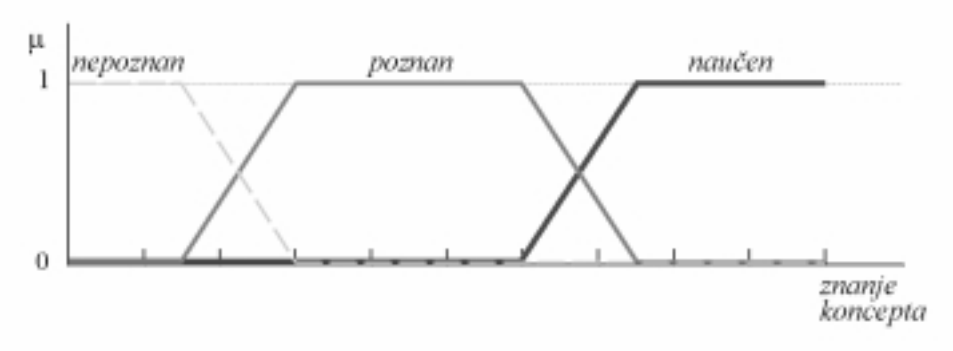
5.3 Modeliranje uporabnikovega znanja

V izobraževalnih sistemih je najpomembnejša lastnost uporabnika prav njegovo znanje. Zato model uporabnika zajema predvsem uporabnikovo poznavanje posameznih konceptov domene, kar lahko preprosto izrazimo s prekrivnim modelom preko modela domene.

Vendar je določanje uporabnikovega poznavanja posameznih konceptov domene precej nezanesljivo, nenatančno in približno, ter vključuje veliko stopnjo nedoločenosti. Tu nedoločenost izhaja iz nejasnih mej med posameznimi razredi (npr. poznan ali nepoznan koncept), torej gre za meglenost, kar lahko rešujemo s pomočjo mehkih množic.

Zato se pri opisu uporabnikovega znanja konceptov naslanjamo predvsem na teorijo mehkih množic, ki je po svoji naravi zelo primerna za reševanje problemov te vrste. Uporabnikovo znanje opišemo z jezikovnimi spremenljivkami, katerih vrednosti so predstavljene z mehkimi množicami. Vsaka od mehkih množic je definirana s svojo pripadnostno funkcijo.

Naj bo množica jezikovnih izrazov *L*={*nepoznan*, *poznan*, *naučen*}, ali v krajšem zapisu $\{l_1, l_2, l_3\}$, zaloga vrednosti jezikovne spremenljivke *znanje koncepta*. Posamezne vrednosti so predstavljene z mehkimi množicami, njihove pripadnostne funkcije pa prikazuje slika 5.2.



Slika 5.2

Pripadnostne funkcije mehkih množic, ki predstavljajo posamezne vrednosti jezikovne spremenljivke *znanje koncepta*

Uporabnikovo poznavanje vsakega koncepta domene lahko opišemo z jezikovno spremenljivko *znanje koncepta*, torej s stopnjami pripadnosti mehkim množicam *nepoznan*, *poznan* in *naučen* (z oznakami l_1, l_2 in l_3 po vrsti):

$$znanje\ koncepta(c_i) = \{\mu_{i,l_1} / l_1, \mu_{i,l_2} / l_2, \mu_{i,l_3} / l_3\}, \quad c_i \in C$$

ali krajše le s posameznimi stopnjami pripadnosti mehkim množicam l_1, l_2 in l_3 po vrsti:

$$znanje\ koncepta(c_i) = (\mu_{i,l_1}, \mu_{i,l_2}, \mu_{i,l_3}).$$

Ker za modeliranje uporabnika uporabljamo prekrivni model preko modela domene, je osnova za model uporabnika graf domene. In ker poznavanje vsakega koncepta opišemo s tremi mehkimi množicami, uporabimo za modeliranje znanja uporabnika mehki graf z mehkimi vozlišči. Model domene opisuje mehki graf z mehkimi povezavami, za potrebe modeliranja uporabnika pa bomo omehčali tudi vozlišča grafa. Ker pri modeliranju uporabnikovega znanja podporne predpogojne relacije niso potrebne (te uporabljamo le pri prilagoditvenih algoritmih), bomo v nadaljevanju upoštevali za osnovo modela uporabnika le graf z mehkimi nujnimi predpogojnimi relacijami iz množice *E*.

Model uporabnika tako predstavlja usmerjen aciklični graf $G = (C, E, L)$, kjer je C množica domenskih konceptov, nujna predpogojna relacija $E \subseteq C \times C$ je mehka relacija, L pa množica oznak, ki je enaka zalogi vrednosti jezikovne spremenljivke *znanje koncepta*. Osnovno množico konceptov C označimo z množico oznak L={ l_1, l_2, l_3}={*nepoznan*, *poznan*, *naučen*}, ki povejo stanje posameznega koncepta. Ker vsakemu konceptu pripada posamezna oznaka le v določeni meri (z določeno stopnjo), je tako označevanje mehko. Zato so vozlišča grafa

definirana kot mehke množice nad množico oznak vozlišč *L*. Stopnje pripadnosti oznak l_j vozlišču c_i zaznamujemo z $\mu_{c_i}(l_j)$. Za graf *G* potemtakem velja:

$$C = \{c_i\}, \quad i = 1 \ldots n$$
$$c_i = \{\mu_{c_i}(l_j) / l_j\}, \quad \forall l_j \in L, \quad j = 1,2,3$$
$$\mu_{c_i} : L \rightarrow [0,1]$$

ter

$$\mu_{c_i}(l_j) = \mu_{i,l_j} \quad \text{za vse} \quad l_j \in L,\ c_i \in C,\ j = 1,2,3 \ \text{ in } \ i = 1 \ldots n.$$

Vsakemu konceptu c_i tako pripišemo ustrezne vrednosti spremenljivke *znanje koncepta*, ki so izražene s stopnjami pripadnosti trem mehkim množicam. Na ta način koncept c_i opišemo (označimo) kot *nepoznan* s stopnjo $\mu_{c_i}(l_1)$, kot *poznan* s stopnjo $\mu_{c_i}(l_2)$ in kot *naučen* s stopnjo $\mu_{c_i}(l_3)$.

5.4 Določanje poznavanja posameznih konceptov

Seveda je eden glavnih problemov pri modeliranju uporabnika določitev primernih vrednosti, ki jih tak model hrani. Iz okolja, torej interakcije uporabnika s sistemom, moramo dobiti kar največ podatkov, katere potem obdelamo in tako izluščimo za nas zanimive podatke. Tu vedno prihaja do dileme, kako pridobiti kakovostne podatke brez nepotrebnega obremenjevanja uporabnika. Pri izobraževalnih hipermedijih je ta problem nekoliko lažji, saj lahko uporabljamo teste za preverjanje znanja, ki so naravni del izobraževalnih aplikacij.

Testov se bomo posluževali tudi v naši aplikaciji kot glavni vir informacij o uporabnikovem pridobljenem znanju. Poleg tega pa bomo upoštevali tudi oglede posameznih učnih enot, ki pa so manj merodajen indikator uporabnikovega znanja.

Kot smo opisali v prejšnjem razdelku, določa uporabnikovo poznavanje vsakega koncepta domene jezikovna spremenljivka *znanje koncepta*. Njene vrednosti so predstavljene s tremi mehkimi množicami *nepoznan*, *poznan* in *naučen*. Označimo množice s C_U, C_K in C_L po vrsti. Pripadnostne funkcije vsem trem množicam prikazuje slika 5.2, iz katere lahko razberemo, da je presek množic C_U in C_L prazen, torej $C_U \cap C_L = \emptyset$. Vidimo tudi, da se množice delno prekrivajo in da je vsota vrednosti vseh treh pripadnostnih funkcij vedno enaka ena. Če za nek koncept $c \in C$ opišemo *znanje koncepta* s trojico (μ_L, μ_K, μ_U), potem velja:

$\mu_U + \mu_K + \mu_L = 1,$
$\mu_U > 0 \Rightarrow \mu_L = 0$ in
$\mu_L > 0 \Rightarrow \mu_U = 0.$

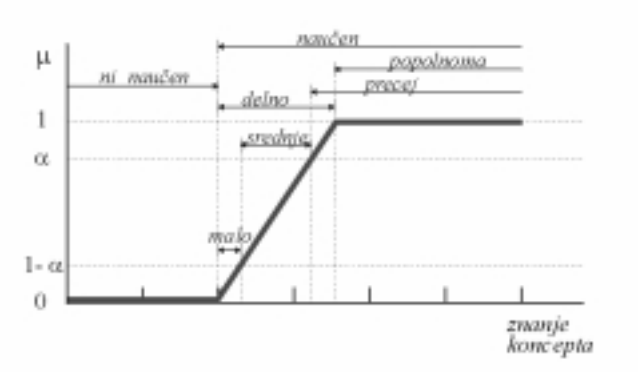
Zgornje tri lastnosti vrednosti znanja koncepta bomo upoštevali tudi pri spreminjanju teh vrednosti. Ker želimo ohraniti skladnost vseh treh vrednosti, vedno spreminjamo le eno komponento naenkrat, ostali dve komponenti pa izračunamo s pomočjo zgornjih enačb. Če je novo izračunana vrednost komponente enaka λ, $0 \le \lambda \le 1$, imata preostali dve komponenti vrednosti $1-\lambda$ in 0. Spreminjamo vedno le najbolj relevantno komponento, ponavadi je to stopnja naučenosti ali stopnja poznavanja koncepta.

Tabela 5.1 – Oznake znanja koncepta in njihov pomen

oznaka znanja koncepta *c*	vrednost μ	pomen
nepoznan	$\mu_U > 0$	$c \in C_U$
popolnoma nepoznan	$\mu_U = 1$	$c \in \text{kernel}(C_U)$
delno nepoznan	$0 < \mu_U < 1$	$c \in C_U$ in $c \notin \text{kernel}(C_U)$
precej nepoznan	$\mu_U \ge \alpha$	$c \in C_{U\alpha}$
*srednje nepoznan**	$1-\alpha \le \mu_U < \alpha$	$c \in C_{U(1-\alpha)}$ in $c \notin C_{U\alpha}$
*malo nepoznan**	$0 < \mu_U < 1-\alpha$	$c \in C_U$ in $c \notin C_{U(1-\alpha)}$
poznan	$\mu_K > 0$	$c \in C_K$
popolnoma poznan	$\mu_K = 1$	$c \in \text{kernel}(C_K)$
delno poznan	$0 < \mu_K < 1$	$c \in C_K$ in $c \notin \text{kernel}(C_K)$
precej poznan	$\mu_K \ge \alpha$	$c \in C_{K\alpha}$
*srednje poznan**	$1-\alpha \le \mu_K < \alpha$	$c \in C_{K(1-\alpha)}$ in $c \notin C_{K\alpha}$
*malo poznan**	$0 < \mu_K < 1-\alpha$	$c \in C_K$ in $c \notin C_{K(1-\alpha)}$
naučen	$\mu_L > 0$	$c \in C_L$
popolnoma naučen	$\mu_L = 1$	$c \in \text{kernel}(C_L)$
delno naučen	$0 < \mu_L < 1$	$c \in C_L$ in $c \notin \text{kernel}(C_L)$
precej naučen	$\mu_L \ge \alpha$	$c \in C_{L\alpha}$
*srednje naučen**	$1-\alpha \le \mu_L < \alpha$	$c \in C_{L(1-\alpha)}$ in $c \notin C_{L\alpha}$
*malo naučen**	$0 < \mu_L < 1-\alpha$	$c \in C_L$ in $c \notin C_{L(1-\alpha)}$

Zalogo vrednosti jezikovne spremenljivke *znanje koncepta* pa lahko še malce razširimo, kar nam omogoča natančnejši opis znanja koncepta, ki je bližji človeškim ocenam znanja. Zato bomo uporabili še nekaj dodatnih kvantifikatorjev. Kadar je μ_K, vrednost pripadnostne funkcije množici C_K, večja od nič, za koncept *c* pravimo, da je *poznan*. Če je vrednost enaka ena, je *popolnoma poznan*, če je manjša od ena, pa *delno poznan*. V primeru, ko vrednost doseže določen prag α (vrednost je večja ali enaka α, $0.5 < \alpha < 1$, v našem primeru smo prag postavili na 0.8), je koncept *precej poznan*, kadar pa ne doseže niti praga $1-\alpha$ (vrednost je manjša od $1-\alpha$), je *malo poznan*. Sicer je koncept *srednje poznan*. Podobno velja tudi za *nepoznane* in *naučene* koncepte.

Vse oznake skupaj so povzete v tabeli 5.1. Različnih oznak je veliko in se delno tudi prekrivajo. Vendar tu ne bomo uporabljali vseh oznak, ker jih ne potrebujemo in nekatere tudi niso preveč smiselne za uporabo. Podane so le zaradi popolnosti in so zato označene z zvezdico (*). Za lažje razumevanje so razmerja med posameznimi oznakami ilustrirana tudi na spodnji sliki (slika 5.3). Prikazane so le oznake za naučen koncept, za nepoznan ali poznan koncept veljajo podobni odnosi.



Slika 5.3
Razmerja med stopnjami naučenosti koncepta

Model učne domene določa poleg množice domenskih konceptov tudi nujno predpogojno relacijo nad temi koncepti. Tako za koncepta c_i in c_j, ki ju povezuje nujna predpogojna relacija $c_i \prec_{w_{ij}} c_j$, $w_{ij} > 0$, velja, da je za razumevanje koncepta c_j potrebno poznavanje koncepta c_i. Vendar pa velja tudi obraten vpliv, saj znanje koncepta c_j v neki meri določa tudi znanje koncepta c_i. Rečemo lahko, da znanje enega koncepta vpliva na znanje drugega, hkrati pa iz znanja drugega koncepta lahko sklepamo na znanje prvega. Če je na primer koncept c_j

popolnoma naučen, koncept c_i, ki je nujni predpogojni koncept koncepta c_j ($c_i \prec_{w_{ij}} c_j$), pa velja za še *popolnoma nepoznanega*, potem lahko sklepamo, da je uporabnikovo znanje koncepta c_i vseeno nekoliko večje, kot je zapisano v modelu uporabnika, saj uporabnik brez poznavanja koncepta c_i (predpogoja) ne bi mogel osvojiti koncepta c_j. Zato lahko v primeru nujno potrebnega predpogoja ($w_{ij} = 1$) sklepamo tudi na popolno naučenost predpogojnega koncepta c_i. V primeru šibkejšega predpogoja ($0 < w_{ij} < 1$), pa je znanje predpogojnega koncepta sorazmerno šibkejše, torej je le *delno naučen* (w_{ij} – kratno znanje koncepta c_j). Tudi ta razmerja bomo upoštevali pri posodabljanju modela uporabnika.

V nadaljevanju bomo najprej opisali postopek določanja znanja posameznih konceptov, ki ga uporabimo na začetku, ob prvi uporabi sistema. Sledi opis različnih načinov spreminjanja vrednosti znanja posameznega koncepta. Pri tem uporabljamo različne pripadnostne funkcije mehkim množicam, ki pa so vse postavljene subjektivno (predvsem izkustveno). Te funkcije so sicer postavljene splošno in so neodvisne od domene učenja, vendar pa jih lahko po potrebi ustrezno spremenimo (oziroma spreminjamo določene parametre), da bolje odražajo stanje v konkretnih primerih.

5.4.1 Določitev začetnih vrednosti

Za vsakega novega uporabnika sistema najprej predpostavimo, da je povsem nov v učni domeni in torej nima nikakršnega znanja o dani domeni učenja. Zato lahko vse koncepte opišemo kot uporabniku neznane. Spremenljivka *znanje koncepta* ima pri vseh konceptih enako vrednost, vsak koncept domene je *popolnoma nepoznan*. Znanje koncepta lahko zapišemo tudi z vrednostmi pripadnostnih funkcij posameznim množicam, torej s trojico (1,0,0).

Vsekakor pa taka predpostavka ne more ustrezati realnim situacijam. Zato smo se odločili, da uporabnik pred začetkom dela s sistemom reši krajši začetni uvrstitveni test, na podlagi katerega lahko dobimo vsaj približno začetno sliko njegovega znanja. To omogoča tudi realnejše opazovanje uporabnikovega napredka skozi delo s sistemom in relativno oceno pridobljenega znanja (primerjava z rezultati končnega testa).

Na podlagi rezultatov tega uvodnega testa, tu ga bomo imenovali *predtest*, lahko določimo začetne (vstopne) vrednosti poznavanja posameznih konceptov. Ovrednotenje rezultatov predtesta je zelo preprosto. Če uporabnik pravilno odgovori na vprašanje v povezavi s konceptom *c*, se koncept *c* smatra kot *popolnoma naučen* in se mu pripišejo vrednosti (0,0,1). V primeru napačnega odgovora ali odgovora “ne vem”, ki je v predtestu tudi eden od možnih

odgovorov, pa ostane koncept *popolnoma nepoznan* oziroma (1,0,0). Seveda lahko uporabnik na vsako vprašanje odgovarja le enkrat.

V kratkem predtestu ne moremo preveriti uporabnikovega poznavanja vseh konceptov domene, saj jih je ponavadi preveč, da bi testirali poznavanje vsakega koncepta posebej. Zato razdelimo vprašanja predtesta v tri skupine. Najprej preverimo poznavanje vseh osnovnih konceptov s pomočjo vprašanj iz prve skupine (vsako vprašanje obravnava en osnovni koncept). Če uporabnik na vsa vprašanja odgovori pravilno (to pomeni, da so vsi osnovni koncepti že poznani), lahko nadaljujemo s preverjanjem poznavanja več skupin vmesnih in končnih konceptov (druga skupina vprašanj, kjer eno vprašanje lahko obsega več konceptov). Če dobimo tudi tu same pravilne odgovore, preverimo še poznavanje vseh končnih konceptov (tretja skupina vprašanj). Če so tudi na ta vprašanja vsi odgovori pravilni, potem uporabnik že pozna učno domeno, ki jo pokriva tečaj.

Vseeno pa so tako dobljene vrednosti znanja domenskih konceptov le prvi približek uporabnikovega znanja. To se potem spreminja in izpopolnjuje skozi uporabo sistema.

5.4.2 Posodabljanje vrednosti

Med samo uporabo sistema se uporabnikovo poznavanje posameznih domenskih konceptov spreminja, saj je to glavni namen izobraževalnih sistemov. Zato moramo model uporabnika sproti posodabljati, da odraža spremembe v uporabnikovem znanju. Do največjih sprememb lahko pride, kadar uporabnik dostopi do nove učne enote (prvi ogled enote), ko si neko učno enoto ponovno ogleda (ponoven ogled enote) ali pa kadar odgovarja na testna vprašanja, ki razkrijejo njegovo razumevanje posameznih konceptov. Ob vseh treh akcijah uporabnika bomo zato posodobili model uporabnika na podlagi vnaprej določenih pravil. V modelu zapisano poznavanje konceptov se na ta način spremeni, kar vpliva tudi na prilagajanje sistema uporabniku.

Pri posodabljanju vrednosti smo ubrali optimističen pristop, zato spreminjamo vrednosti modela uporabnika le v primeru prikazanega znanja, ne pa tudi ob pomanjkanju znanja. Tako napačen odgovor na vprašanje ne vpliva na trenutne stopnje znanja koncepta.

Sledi opis mehanizmov za povečanje znanja uporabnika v primeru ogleda učne enote, v primeru odgovarjanja na testna vprašanja pri dani učni enoti in spremembe vrednosti drugih konceptov na podlagi novih vrednosti danega koncepta (razširjanje vrednosti).

5.4.2.1 Ogled učne enote

Sklepanje o uporabnikovem znanju na osnovi ogleda učne enote je zelo nehvaležno, saj sam ogled strani ne daje nikakršnega jamstva, da uporabnik učno snov tudi zares preštudira, jo razume in si jo zapomni. Pri tem lahko močno varirajo tudi čas ogleda strani, intenzivnost dela (branja strani, reševanja problema in podobno) ter osredotočenost uporabnika.

Kratek čas lahko pomeni, da je enota prelahka (uporabnik že vse pozna), pretežka (nič ne razume) ali pa na splošno nezanimiva za uporabnika. Podobno lahko dolg čas pomeni, da je enota zelo zanimiva, da je pretežka ali pa da uporabnik trenutno sploh ne dela s sistemom.

Ogled učne enote poveča stopnjo pripadnosti množici poznanih konceptov, a le v primeru, če je koncept še (vsaj *malo*) *nepoznan*. Seveda se ob ogledu enote stopnja pripadnosti množici *poznan* ne zmanjša, lahko pa se poveča, saj se poveča poznavanje koncepta. Pri tem smo upoštevali, da je razumevanje vsebine učne enote lažje, če so poznani ali naučeni tudi podporni predpogojni koncepti, zato se v takem primeru vrednost poznavanja koncepta bolj poveča. Podobno velja tudi za nujne predpogojne koncepte, le da so ti resnično potrebni za razumevanje vsebine trenutne enote in zato njihovo nepoznavanje vpliva na manjši doprinos – vrednost poznavanja koncepta se poveča manj. Vse skupaj lahko zapišemo z naslednjim pravilom, katero določa znanje koncepta po ogledu strani, ki ta koncept opisuje:

> Spremenljivka *znanje koncepta* dobi najprej vrednost *popolnoma poznan*. Če so kateri nujni predpogojni koncepti danega koncepta še nepoznani, se ta vrednost zmanjša za povprečno vrednost vseh z močjo predpogojne relacije oslabljenih vrednosti *nepoznan* nujnih predpogojnih konceptov danega koncepta. Če pa kateri njegovih podpornih predpogojnih konceptov niso več nepoznani, se vrednost poveča za povprečno vrednost vseh vrednosti *popolnoma poznan* ali *naučen* podpornih predpogojnih konceptov danega koncepta, korigirano s faktorjem β ($0 < \beta < 1$). Kadar je tako dobljena nova vrednost *poznan* manjša od prvotne, se ohrani prvotna vrednost.

To pravilo lahko zapišemo tudi drugače. Naj bo *u* učna enota, ki opisuje koncept c_i, kar lahko zapišemo kot $M_m(u) = c_i$. Po ogledu učne enote *u* se znanje koncepta c_i spremeni le v primeru, če so njegove sedanje vrednosti $(\mu_{i1}, \mu_{i2}, 0)$, $\mu_{i1} > 0$. Če ima koncept c_i p nujnih predpogojnih konceptov c'_k, $c'_k \prec_{w_{ki}} c_i$, $w_{ki} > 0$, $k = 1 \ldots p$, z vrednostmi $(\mu_{k1}, \mu_{k2}, \mu_{k3})$ in r podpornih predpogojnih konceptov c''_j, $c''_j \prec_0 c_i$, $j = 1 \ldots r$, z vrednostmi $(\mu_{j1}, \mu_{j2}, \mu_{j3})$,

potem lahko nove vrednosti koncepta c_i zapišemo s trojico $(\mu_{i1}', \mu_{i2}', 0)$, katere komponenti dobimo na podlagi naslednjih formul:

$$\mu_{i2}' = \max[\mu_{i2}, \min(1, 1 - a + b)] \quad \text{in}$$

$$\mu_{i1}' = 1 - \mu_{i2}',$$

kjer spremenljivki *a* in *b* določata naslednji enačbi:

$$a = \frac{1}{p}\sum_{k=1}^{p} \mu_{k1} w_{ki}, \quad \text{kjer je } w_{ki} > 0 \text{ za vse } c_k' \prec_{w_{ki}} c_i,\ k = 1 \ldots p, \text{ ter}$$

$$b = \beta \cdot (1 - \frac{1}{r}\sum_{j=1}^{r} \mu_{j1}), \quad \text{kjer je } c_j'' \prec_0 c_i,\ j = 1 \ldots r \quad \text{in} \quad \beta = \tfrac{1}{3}.$$

Spremenljivka *a* določa zmanjšanje vrednosti pripadnostne funkcije v primeru nepoznavanja nujnih predpogojnih konceptov, spremenljivka *b* pa povečanje njene vrednosti v primeru poznavanja podpornih predpogojnih konceptov. Za korekcijski faktor smo v našem primeru uporabili $\beta = \frac{1}{3}$. Če koncept c_i nima nujnih predpogojnih konceptov, je spremenljivka $a = 0$ (v takem primeru poznavanje predpogojev ne vpliva na poznavanje koncepta). Kadar pa koncept c_i nima podpornih predpogojnih konceptov, je spremenljivka $b = 0$.

5.4.2.2 Odgovori na vprašanja

Vsaka učna enota, ki razlaga nek glavni koncept, ima tudi pripadajoča vprašanja, s pomočjo katerih se preverja razumevanje tega koncepta. Vseh vprašanj je *q* (lahko je le eno samo ali pa jih je več), vsako vprašanje pa ima *m* odgovorov. Na primer, če gre za vprašanje z več izbirami (angl. multiple choice question), je ponujenih *m* različnih odgovorov, od katerih je le en pravilen. Če uporabnik ne odgovori pravilno v prvem poskusu (njegov prvi odgovor ni pravilen), lahko poskusi znova, a se tako ponavljanje upošteva pri oceni znanja koncepta.

S testi preverjamo le naučenost koncepta, zato se po uspešno rešenem testu spremeni vrednost pripadnosti naučenim konceptom, ne glede na predhodno poznavanje oziroma nepoznavanje koncepta. Temu ustrezno se spremenijo tudi stopnje pripadnosti množicama nepoznanih in poznanih konceptov. Če test ni bil uspešno opravljen, se vrednosti ne spremenijo.

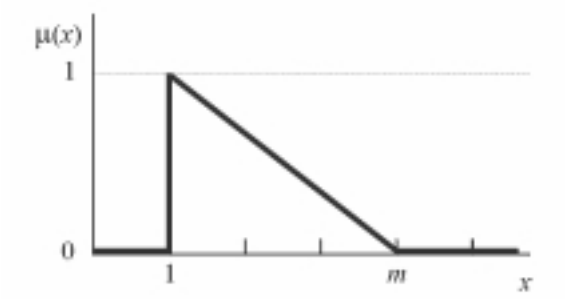
Tako pravilo, ki določa uporabnikovo znanje koncepta po reševanju testov o danem konceptu, se glasi:

Spremenljivka *znanje koncepta* dobi vrednost *naučen*, ki je enaka povprečni vrednosti vseh vrednosti *naučen*, določenih na podlagi odgovorov na vprašanja, ki preverjajo znanje danega koncepta. Če je tako dobljena nova vrednost *naučen* manjša od prvotne, se ohrani prvotna vrednost.

Za vsako posamezno vprašanje, pri katerem je *m* število vseh odgovorov na vprašanje, *x* pa poskus, v katerem je bil odgovor pravilen, določimo pripadnostno funkcijo množici naučenih konceptov, ki je enaka:

$$\mu(x) = \frac{m - x}{m - 1}$$

Konstanta *m* lahko zavzame vrednosti večje ali enake 2 (ponavadi uporabljamo vrednost 4), spremenljivka *x* pa je $1 \le x \le m$. Funkcija v primeru pravilnega prvega odgovora vrne vrednost 1, v primeru pravilnega zadnjega možnega odgovora (*m*-tega) vrne nič, sicer pa neko vmesno vrednost (funkcija linearno pada). Pripadnostna funkcija je grafično ponazorjena na sliki 5.4, kjer smo za *m* vzeli vrednost 4.



Slika 5.4
Pripadnost množici naučenih konceptov v odvisnosti od števila odgovorov

Naj test *t*, ki pripada učni enoti *u*, preverja poznavanje koncepta *c*, torej je $M_m(u) = c$. Če ima koncept *c* stopnje pripadnosti posameznim množicam (μ_1, μ_2, μ_3) in je na podlagi uspešno opravljenega testa ugotovljena nova stopnja pripadnosti množici naučenih konceptov μ_3', $\mu_3' > 0$, se vrednosti znanja koncepta *c* spremenijo na (0, 1– μ_3', μ_3'). Če je bil koncept že prej delno naučen ($\mu_3 > 0$), to upoštevamo tudi pri računanju novih stopenj pripadnosti. Vrednost μ_3' izračunamo na podlagi naslednje enačbe:

$$\mu_3' = \max[\mu_3, a],$$

kjer je *a* povprečje stopenj pripadnosti preko vseh *q* vprašanj:

$$a = \frac{1}{q}\sum_{k=1}^{q}\mu_k(x_k).$$

5.4.2.3 Razširjanje vrednosti

V Bayesovih mrežah lahko s pomočjo Bayesovega pravila revidiramo začetne verjetnosti z uporabo pogojnih verjetnostnih porazdelitev, ko je na razpolago nova informacija. Podobno revidiranje vrednosti znanja koncepta bomo uporabili tudi v našem grafu modela uporabnika, le da bo temeljilo na pripadnostih funkcijah mehkim množicam in predpogojnih relacijah med koncepti.

Pri vsakem ogledu strani ali uspešnem reševanju testa v povezavi z nekim konceptom se spremeni tudi njegovo poznavanje (vrednosti stopenj pripadnosti). Iz poznavanja koncepta pa lahko v določeni meri sklepamo tudi na poznavanje njegovih nujnih predpogojnih konceptov, zato se ob vsaki spremembi (posodobljenju) vrednosti sproži tudi mehanizem sklepanja za vse (posredne) predpogojne koncepte. Tako sklepanje temelji na mehkih pravilih, ki upoštevajo razširjanje znanja, to so vrednosti pri poznanih in naučenih konceptih (druga in tretja vrednost v trojici, ki predstavlja znanje koncepta). Pri tem se stopnja pripadnosti naučenim konceptom lahko le poveča, stopnja pripadnosti nepoznanim konceptom pa le zmanjša, seveda pa lahko obe ostaneta tudi nespremenjeni. Stopnja pripadnosti poznanim konceptom se lahko poveča (zmanjša) na račun zmanjšanja (povečanja) stopnje pripadnosti nepoznanim (naučenim) konceptom. Tako vsi koncepti ohranjajo (ali povečajo) svoje največje vrednosti znanja.

Pri opisu znanja koncepta bomo uporabili vrednosti *popolnoma naučen* in *delno naučen*, kadar je se je uporabnik dani koncept že naučil, *popolnoma nepoznan* in *delno nepoznan* v drugi skrajnosti, ko je koncept uporabniku nepoznan, ter vmesno stopnjo *popolnoma poznan*, ko ni ne prvo ne drugo. Z upoštevanjem teh petih vrednosti in dveh konceptov (dani koncept c_j in njegov nujni predpogoj c_i) lahko sestavimo 25 različnih pravil, ki določajo novo vrednost znanja predpogojnega koncepta. Nekatera pravila lahko nadalje združimo in tako število pravil, na katerih temelji sklepanje o znanju konceptov, skrčimo na naslednjih šest:

1. Če koncept c_i ni *nepoznan* (je *naučen* ali *popolnoma poznan*) in koncept c_j ni *naučen*, ohrani koncept c_i svoje vrednosti *naučen* oziroma *poznan*.

2. Če je koncept c_i *naučen* in je koncept c_j *naučen*, dobi koncept c_i vrednost *naučen*, ki je večja od obeh vrednosti: *naučen* koncept c_i ali z močjo nujne predpogojne relacije oslabljena vrednost *naučen* koncepta c_j.
3. Če koncept c_i ni *naučen* in je koncept c_j *naučen*, postane vrednost *naučen* pri konceptu c_i z močjo nujne predpogojne relacije oslabljena vrednost *naučen* koncepta c_j.
4. Če je koncept c_i *popolnoma nepoznan* in je koncept c_j *poznan*, a ni *naučen*, postane vrednost *poznan* pri konceptu c_i z močjo nujne predpogojne relacije oslabljena vrednost *poznan* koncepta c_j.
5. Če je koncept c_i *nepoznan* in je koncept c_j *popolnoma nepoznan*, ostane koncept c_i *nepoznan*.
6. Če je koncept c_i *delno nepoznan* in je koncept c_j *poznan*, a ni *naučen*, dobi koncept c_i vrednost *poznan*, ki je večja od obeh vrednosti: *delno poznan* koncept c_i ali z močjo nujne predpogojne relacije oslabljena vrednost *poznan* koncepta c_j.

Tudi ta pravila lahko zapišemo z uporabo petih enačb, kjer pravili 4 in 6 združimo v eno. Naj bo znanje koncepta c_j enako $(\mu_{j1}, \mu_{j2}, \mu_{j3})$ in znanje koncepta c_i enako $(\mu_{i1}, \mu_{i2}, \mu_{i3})$, kjer je koncept c_i nujni predpogojni koncept koncepta c_j z utežjo w_{ij} ($c_i \prec_{w_{ij}} c_j$). Potem lahko nove vrednosti znanja koncepta c_i zapišemo kot $(\mu_{i1}', \mu_{i2}', \mu_{i3}')$. Izračunamo jih lahko na podlagi enačb iz tabele 5.2, ki prikazuje spreminjanje posameznih stopenj pripadnosti v odvisnosti od prvotnih vrednosti znanja konceptov c_i in c_j.

Tabela 5.2 – Izračun novih vrednosti znanja koncepta

koncept c_i	koncept c_j	nove vrednosti znanja koncepta c_i
$(0, \mu_{i2}, \mu_{i3})$	$(\mu_{j1}, \mu_{j2}, 0)$	$(0, \mu_{i2}, \mu_{i3})$
$(0, \mu_{i2}, \mu_{i3})$	$(0, \mu_{j2}, \mu_{j3})$, $\mu_{j3} > 0$	$(0, 1 - \max[\mu_{i3}, \mu_{j3} \cdot w_{ij}], \max[\mu_{i3}, \mu_{j3} \cdot w_{ij}])$
$(\mu_{i1}, \mu_{i2}, 0)$	$(0, \mu_{j2}, \mu_{j3})$, $\mu_{j3} > 0$	$(0, 1 - \mu_{j3} \cdot w_{ij}, \mu_{j3} \cdot w_{ij})$
$(\mu_{i1}, \mu_{i2}, 0)$	$(\mu_{j1}, \mu_{j2}, 0)$, $\mu_{j2} > 0$	$(1 - \max[\mu_{i2}, \mu_{j2} \cdot w_{ij}], \max[\mu_{i2}, \mu_{j2} \cdot w_{ij}], 0)$
$(\mu_{i1}, \mu_{i2}, 0)$	$(1, 0, 0)$	$(\mu_{i1}, \mu_{i2}, 0)$

Do sedaj smo opisali, kako se ob spremembi vrednosti znanja koncepta c_j spremenijo tudi vrednosti znanja vseh njegovih nujnih predpogojnih konceptov c_i. Vendar pa te spremembe vrednosti znanja c_i nadalje vplivajo na druge koncepte, ki so nujni predpogoji koncepta c_i in tako naprej. Sprememba vrednosti znanja določenega koncepta torej vpliva na vse neposredne in posredne predpogojne koncepte tega koncepta. Ob spremembi znanja enega koncepta se sproži neke vrste verižna reakcija, ki steče po posrednih predpogojih tega koncepta vse do osnovnih konceptov, kateri nimajo nobenih predpogojev. Zato se algoritem za razširjanje vrednosti koncepta ne omejuje le na neposredne predpogoje tega koncepta, temveč mora zaobsegati tudi vse njegove posredne predpogoje.

Graf domene, na katerem temelji model uporabnika, je acikličen in usmerjen. Pri razširjanju vrednosti konceptov nas zanima njegov podgraf, ki zajema izbrani koncept c_j in vse njegove neposredne in posredne predpogojne koncepte. Ta podgraf je prav tako acikličen in usmerjen, ni pa nujno v obliki drevesa, čeprav ima en koren (koncept c_j) in več listov (vsi osnovni koncepti v podgrafu), saj je lahko bolj povezan. Algoritem razširjanja mora prečesati cel podgraf in v vsakemu vozlišču posodobiti vrednosti znanja pripadajočega koncepta. Pri posodabljanju vedno pripisujemo vozliščnim konceptom največje možne vrednosti, torej se znanje koncepta lahko le povečuje. Za razširjanje vrednosti znanja konceptov uporabimo naslednji algoritem:

Algoritem: Razširjanje vrednosti konceptov;

Vhod: graf modela uporabnika, koncept c_j;
Izhod: posodobljen graf modela uporabnika;

```
Telo:  {
    if (cj je osnovni koncept)
        return;
    za cj sestavi seznam nujni_predpogoji = [ ci : ci ≺wij cj, wij > 0, i = 1…p ];
    for ( i = 1;  i <= p;  i++ )  {
        za ci s seznama nujni_predpogoji izračunaj nove vrednosti po tabeli 5.2;
        rekurzivno kliči Razširjanje vrednosti konceptov za ci;
    }
}
```

Opisani algoritem pregleduje graf modela uporabnika (graf domene) tako, da išče vse možne poti od koncepta c_j do vseh ostalih konceptov v podgrafu (pri tem upoštevamo nasprotno usmerjenost predpogojnih relacij). Algoritem uporablja razširjanje v globino, začne pri izbranem konceptu in se premika v globino do osnovnih konceptov, ki nimajo predpogojev. Ker

v grafu ni ciklov, se postopek ustavi. Po vsaki od poti spreminja (posodablja) vrednosti znanja koncepta v vsakem vozlišču na poti. Do nekega vozlišča v grafu lahko vodi tudi več poti, a vedno izbere največjo vrednost za znanje koncepta. To vrednost potem tudi posreduje naprej do drugih vozlišč.

V najbolj splošnem primeru algoritem obišče vsako vozlišče podgrafa največ tolikokrat, kolikor je vseh vozlišč podgrafa. Teh pa je manj ali enako kot vozlišč grafa domene, torej največ n. Iz tega sledi, da je v splošnem kompleksnost algoritma $O(n^2)$. Kadar pa ima dani podgraf obliko drevesa (manjša povezanost), je kompleksnost manjša, le $O(n)$.

5.5 Prilagoditvene zmožnosti sistema

Prilagajanje v sistemu temelji na modelu domene, ki opisuje koncepte domene in njihove medsebojne relacije, in na modelu uporabnika, ki opisuje uporabnikovo poznavanje posameznih domenskih konceptov. Prilagodljivi hipermediji uporabljajo dva načina prilagajanja, to sta predstavitev in podpora navigaciji. Pri prilagajanju vsebine strani ima vsaka hipermedijska stran dodane določene komentarje, ki jih sistem prepozna in obdela in ki določajo pravilen prikaz vsebine strani. Druga možnost je, da sistem dinamično sestavlja strani iz manjših odstavkov, ki se nahajajo v datotekah v strežniku ali pa v bazi znanja [Brusilovsky 1998a]. Če pa so hipermedijske strani že vnaprej pripravljene in ne vključujejo posebnih metapodatkov za prikaz strani, prilagajanje vsebine strani ni možno. Naš sistem smo zasnovali tako, da učni material sestavljajo katerekoli spletne strani, ki so lahko tudi že prej pripravljeni deli drugih tečajev. Tako lahko učni material obravnavamo kot del svetovnega spleta. Takšna zasnova torej ne omogoča prilagajanja vsebine strani, zato smo se osredotočili predvsem na prilagodljivo pomoč pri navigaciji skozi učni material (prilagodljiva podpora navigaciji).

Sistem sicer omogoča uporabniku popolnoma prosto navigacijo skozi učni material, a ta ni najbolj zaželena, ker uporabniku ponavadi ne nudi najprimernejšega zaporedja učnih enot (optimalne poti skozi učni material). Zato je priporočljiva uporaba navigacijskih pripomočkov, ki omogočajo bolj selektivno izbiro poti skozi material in podpirajo za uporabnika ustreznejšo navigacijo. Sistem ponuja več načinov navigacije skozi učni material: linearno navigacijo (neposredno vodenje, naprej in nazaj), hierarhično navigacijo (preko kazala vsebine), relacijsko navigacijo (na podlagi predpogojne povezanosti konceptov preko vstavljenih povezav) ali navigacijo po konceptih (preko indeksa konceptov).

Učne enote praviloma ne vključujejo povezav na druge enote. Torej so povezave ločene od vsebine strani in jih zato tudi prikažemo ločeno, dodajamo jih posebej in po potrebi. Same povezave na druge enote določimo iz grafa domene na podlagi predpogojne povezanosti konceptov. Zato se navigacija po učnem materialu (v domenskem prostoru) izvaja posredno preko navigacije v mreži domenskih konceptov.

V nadaljevanju bomo opisali tri uporabljene tehnologije prilagodljive podpore navigaciji (anotacija povezav, prilagodljivo vstavljanje povezav in neposredno vodenje) ter uporabo kazala vsebine in indeksa kot pripomočkov za navigacijo skozi učni material.

5.5.1 Stanje učne enote

Vsaki učni enoti lahko določimo njeno izobraževalno stanje, ki temelji na uporabnikovem znanju glavnega koncepta te enote. Tako je lahko učna enota *u*, ki opisuje koncept *c* (torej je $M_m(u) = c$), v enem od naslednjih petih stanj:

- *Naučena.* Koncept *c* ima vrednost *precej naučen*. Ko je enota enkrat v tem stanju, se njena vsebina smatra za osvojeno. Uporabnik sicer lahko dostopa do take enote, vendar od tega ni pričakovati večje koristi.
- *Poznana.* Koncept *c* je že *precej poznan* ali celo *delno naučen*, zato ogled enote ne prinese nič novega, vendar pa enota še ne spada med naučene enote. Izbira takih enot je priporočljiva le kot osvežitev znanja in kot odskočna deska za reševanje testa na dano temo.
- *Pripravljena na učenje.* Koncept *c* je še *nepoznan*, vsi njegovi nujni predpogojni koncepti pa so že *precej naučeni*, vendar enota še ni poznana. Enote v tem stanju so najprimernejše za ogled in nadaljevanje učenja, saj prinašajo novo znanje, ki ga je uporabnik že pripravljen sprejeti, ker ima dovolj predznanja za njihovo razumevanje.
- *Pogojno pripravljena na učenje.* Koncept *c* je še *nepoznan*, vsi njegovi nujni predpogojni koncepti pa so ali *precej poznani* ali *naučeni*. Obstaja pa vsaj en predpogojni koncept, ki še ni *precej naučen* (sicer bi ta enota spadala med enote, pripravljene na učenje). Sama enota tudi še ni poznana. Tudi te enote so primerne za ogled in nadaljevanje učenja, a zahtevano uporabnikovo predznanje zanje še ni povsem potrjeno.

- *Nepripravljena na učenje*. Koncept *c* je še *nepoznan*, prav tako pa so še *nepoznani* nekateri od njegovih nujnih predpogojnih konceptov. Ogled take enote ni priporočljiv, saj uporabnik še nima dovolj predznanja za njeno razumevanje.

Posamezna izobraževalna stanja enot se med seboj izključujejo in vsaka enota je naenkrat v natanko enem od navedenih stanj. Stanje učne enote se lahko spremeni, kadar glavni koncept enote ali njegovi nujni predpogojni koncepti spremenijo vrednosti znanja koncepta. Stanje enote se lahko spreminja, dokler enota ne doseže stanja naučene enote. Za izračunavanje izobraževalnega stanja učne enote uporabimo naslednji algoritem:

Algoritem: Določanje izobraževalnega stanja učne enote;

Vhod: graf modela uporabnika, učna enota *u*;
Izhod: izobraževalno stanje učne enote *u*;

```
Telo: {
    stanje = "nepripravljena na učenje";
    α = 0.8;
    c = M_m(u);
    vrednosti c so (μ_L, μ_K, μ_U);
    if ( μ_L ≥ α )
        stanje = "naučena";
    else if ( ( μ_K ≥ α ) || ( 0 < μ_L < α ) )
        stanje = "poznana";
    else {    // μ_U > 0
        za c sestavi seznam nujni_predpogoji = [ c_i : c_i ≺_wi c, w_i > 0, i = 1…p ];
        stanje = "pripravljena na učenje";
        for ( i = 1; i <= p; i++ ) {
            vrednosti c_i so (μ_iL, μ_iK, μ_iU);
            if ( ( μ_iK ≥ α ) || ( 0 < μ_iL < α ) ) {
                stanje = "pogojno pripravljena na učenje";
            }
            if ( μ_iU > 1 – α ) {
                stanje = "nepripravljena na učenje";
                break;
            }
        }
    }
    return stanje;
}
```

Algoritem pregleda uporabnikovo znanje glavnega koncepta učne enote in po potrebi tudi njegovih nujnih predpogojnih konceptov ter na podlagi vrednosti znanja koncepta določi

izobraževalno stanje učne enote. Ker je vseh konceptov, ki jih algoritem preveri, največ n, je kompleksnost algoritma linearna – $O(n)$.

5.5.2 Prilagodljiva anotacija

Anotacija povezav lahko zelo pomaga uporabniku pri navigaciji skozi učni material, saj z označevanjem posameznih povezav podaja uporabniku dodatno informacijo o teh povezavah oziroma o tem, kam vodijo. V našem sistemu bomo uporabili dve vrsti anotacije, označili bomo obiskane učne enote in različna izobraževalna stanja učnih enot.

Pri vseh enotah, ki jih je uporabnik že obiskal, bomo v kazalu vsebine za imenom enote dodali še kljukico, ki označuje opravljen ogled enote. Pri tem velja opozoriti, da obiskana enota še ne pomeni nujno tudi že poznane (ali celo naučene) enote, saj je slednje odvisno od znanja glavnega koncepta enote in tudi vseh njegovih predpogojnih konceptov. Pa tudi obratno ne velja, saj poznana enota ni bila nujno tudi obiskana (uporabnik lahko glavni koncept enote pozna že na začetku tečaja ali pa na njegovo poznavanje sklepa sistem na podlagi znanja ostalih konceptov domene). Je pa pomembno, da so obiskane enote posebej označene, ker to pomaga uporabniku pri pregledu nad učno snovjo in daje dodatno informacijo o sami učni enoti.

Za označevanje izobraževalnega stanja učnih enot bomo uporabili barvno anotacijo tako, da je vsako stanje enote označeno z drugo barvo. Izbira barv temelji na metafori semaforja in je naslednja:

- *Črna.* S črno so označene naučene enote, ki se na ta način zlijejo z ostalim besedilom in ne izstopajo. Te enote za uporabnika niso posebej pomembne, saj se iz njihove vsebine ne more naučiti nič novega.
- *Modra.* Modra barva označuje že poznane enote, katerih koncepti pa še niso naučeni, zato oznake takih enot le malo izstopajo in so umirjene barve. Ta barva uporabnika le opozarja, da enota še ni dokončno obdelana, saj manjka še uspešno opravljen test.
- *Zelena.* Učne enote, ki so pripravljena za učenje in so zato tudi trenutno najprimernejše za ogled, so označene z zeleno. Zelena barva predstavlja prosto pot (kot na semaforju) in privablja uporabnika, da ji sledi.
- *Oranžna.* Oranžna barva še vedno vabi, a hkrati tudi opozarja (semafor). Uporabljena je za označevanje enot, ki so pogojno pripravljene za učenje. Te sicer predstavljajo

nov izziv (pridobitev novega znanja) za uporabnika, vendar pa istočasno kaže tudi na delno pomanjkljivo naučenost predpogojev te enote.

- *Rdeča.* Za učenje še nepripravljene enote so označene z rdečo. To je barva prepovedi, ki zapira pot (kot rdeča luč na semaforju) in ne vzpodbuja nadaljevanja v tej smeri. Obisk takih enot ni priporočljiv, saj uporabnik še nima dovolj predznanja za razumevanje njihove vsebine.

Povezave na učne enote so barvno označene povsod, kjer se pojavljajo: v kazalu vsebine, indeksu in v seznamih vstavljenih povezav.

5.5.3 Prilagodljivo vstavljanje povezav

Glavna tehnologija prilagodljive podpore navigaciji, ki jo bomo uporabljali v našem sistemu, je prilagodljivo vstavljanje povezav [Kavčič 2000a]. To je sicer neke vrste kombinacija prilagodljivega skrivanja in prilagodljivega razvrščanja povezav, vendar zaradi posebnega pristopa k povezanosti učnih enot ne moremo govoriti ne o prvem ne o drugem.

Kot smo že omenili, so povezave na druge učne enote ločene od vsebine učne enote. Ker same enote ne vključujejo povezav (med enotami ni neposrednih povezav), se te za vsako učno enoto posebej dinamično dodajo v ločenem okvirju pod vsebino enote (dinamično povezovanje enot). Na katere učne enote bodo na ta način dodane povezave, je odvisno od trenutno obiskane učne enote, izobraževalnega stanja posameznih učnih enot, nivoja uporabnikovega znanja domenskih konceptov in predpogojne povezanosti domenskih konceptov. Za vsako učno enoto tako določimo ustrezne povezave na druge enote, ki jih priporočamo uporabniku.

Izbira povezav se izvede s pomočjo algoritma ALI (kratica iz angl. Adaptive Link Insertion). Ta algoritem izbira le nehierarhične povezave na enote, hierarhija učnih enot je dostopna preko kazala vsebine. Algoritem ALI določi tri skupine povezav na enote, ki se na nek način navezujejo na trenutno enoto.

V prvi skupini so povezave na naslednje primerne enote za učenje. To so enote, ki jih sistem priporoča uporabniku, ker so smiselno nadaljevanje trenutne učne enote in so hkrati primerne glede na uporabnikovo stopnjo znanja. Te povezave so na voljo le pri enotah, ki so že naučene ali poznane.

Druga skupina vsebuje povezave na enote, katere zajemajo znanje, ki je potrebno za lažje razumevanje izbrane enote, a še ni osvojeno. Uporabnik naj bi predelal vse učne enote s

predlaganega seznama, preden se loti trenutno izbrane enote. Te povezave so na voljo le pri enotah, ki še niso pripravljene na učenje. Če pa je enota poznana ali (pogojno) pripravljena na učenje, se v tej skupini pojavijo povezave na še neosvojeno znanje, ki lahko pomaga k boljšemu razumevanju dane enote.

Tretjo skupino povezav tvorijo povezave na celotno že osvojeno predznanje, ki je potrebno za razumevanje izbrane enote. To so povezave za osvežitev znanja (preprečevanje pozabljanja), hkrati pa pomagajo k boljši orientaciji in lažji umestitvi izbrane enote v uporabnikov kognitivni zemljevid učne domene.

Algoritem ALI, ki sestavi opisane tri sezname povezav iz dane enote, je naslednji:

Algoritem: Prilagodljivo vstavljanje povezav;

Vhod: graf domene, model uporabnika, trenutna učna enota *u*;
Izhod: trije seznami povezav na ustrezne učne enote;

```
Telo: {
    stanje = izobraževalno stanje učne enote u;
    predpogojne_enote = Poišči predpogojne enote dane enote u;
    naslednje_enote = [];
    potrebne_enote = [];
    if ( stanje != “nepripravljena na učenje” ) {
        if ( stanje != “naučena”)
            potrebne_enote = Poišči pomožne enote dane enote u;
        if ( (stanje = = “naučena”) || (stanje = = “poznana”) )
            naslednje_enote = Poišči naslednje enote dane enote u;
    }
    else
        potrebne_enote = Poišči potrebne enote dane enote u;
    return naslednje_enote, potrebne_enote, predpogojne_enote;
}
```

Zaradi preglednosti smo za iskanje posameznih seznamov povezav na učne enote uporabili štiri ločene podprograme. Prvo skupino povezav sestavimo v seznam “naslednje_enote”, ki zajema vse primerne enote za nadaljevanje. To pomeni, da poiščemo vse enote, ki imajo glavni koncept trenutne enote za nujni predpogojni koncept svojega glavnega koncepta. Če je tako najdena enota nepripravljena na učenje, jo zamenjamo z njenimi nujnimi predpogojnimi enotami, ki pa tudi ne smejo biti nepripravljene na učenje (sicer jih ne vključimo v seznam). Seznamu dodamo še vse enote, ki imajo glavni koncept trenutne enote za podporni predpogojni koncept svojega glavnega koncepta in niso nepripravljene na učenje. Seveda iz seznama izločimo vse že naučene enote. Za vse končne enote (njihov glavni koncept je končni koncept) je tako

sestavljen seznam naslednjih enot prazen, kar pomeni, da smo dosegli enega od ciljev (končni koncept je namreč poznan ali naučen). Če izbrana učna enota ni poznana ali naučena, je seznam naslednjih enot prazen.

Algoritem: Poišči naslednje enote;

Vhod: graf domene, model uporabnika, trenutna učna enota u;
Izhod: seznam povezav na ustrezne učne enote;

```
Telo: {
    c = M_m(u);
    naslednje_enote = [];
    for ( vse c_i, c ≺_wi c_i, c_i = M_m(u_i) ) {
        if ( (enota u_i je "nepripravljena na učenje") && (w_i > 0) ) {
            for ( vse c_j, c_j ≺_wj c_i, w_j > 0, c_j = M_m(u_j), u_j != u ) {
                if ( (enota u_j ni "naučena") && (enota u_j ni "nepripravljena na učenje") )
                    naslednje_enote = naslednje_enote + [ u_j ];
            }
        }
        if ( (enota u_i ni "naučena") && (enota u_i ni "nepripravljena na učenje") ) {
            naslednje_enote = naslednje_enote + [ u_i ];
        }
    }
    izloči duplikate iz seznama naslednje_enote;
    sortiraj seznam naslednje_enote po padajoči vrednosti w ;
    return naslednje_enote;
}
```

Druga skupina povezav, seznam "potrebne_enote", je odvisna od izobraževalnega stanja izbrane enote. Če je enota naučena, je seznam prazen. Pri enoti, nepripravljeni na učenje, vsebuje seznam povezave na vse njene nujne predpogojne enote (to so enote, katerih glavni koncept je nujni predpogojni koncept glavnega koncepta dane enote), ki so (pogojno) pripravljene na učenje (algoritem "Poišči potrebne enote"). Seznam sestavimo tako, da najprej poiščemo vse nujne predpogojne enote, če pa je najdena enota nepripravljena na učenje, dodamo v seznam tudi njene nujne predpogojne enote (in postopek po potrebi ponavljamo do osnovnih enot). Slednje so dani učni enoti le posredne predpogojne enote, kar tudi posebej označimo. Iz seznama izločimo tudi vse enote, ki so naučene ali poznane. Pri vseh ostalih stanjih enote pa so v seznamu "potrebne_enote" povezave na vse podporne predpogojne enote, ki so pripravljene na učenje ali pa le pogojno pripravljene na učenje (algoritem "Poišči pomožne enote").

Algoritem: Poišči potrebne enote;

Vhod: graf domene, model uporabnika, trenutna učna enota u;
Izhod: seznam povezav na ustrezne učne enote;

```
Telo: {
    c = M_m(u);
    potrebne_enote = [];
    for ( vse c_i , c_i ≺_wi c, w_i > 0 , c_i = M_m(u_i) ) {
        if ( enota u_i ni "naučena" ali "poznana" )
            potrebne_enote = potrebne_enote + [ u_i ];
    }
    sortiraj seznam potrebne_enote po padajoči vrednosti w_i ;
    for ( vse u_i iz seznama potrebne_enote ) {
        if ( enota u_i je "nepripravljena na učenje") {
            temp = [];
            for ( vse c_j, c_j ≺_wj c_i, w_j > 0, c_j = M_m(u_j) ) {
                if ( enota u_j ni "naučena" ali "poznana" )
                    temp = temp + [ u_j ];
            }
            za enoto u_i v seznamu potrebne_enote dodaj vse enote iz seznama temp;
        }
    }
    izloči duplikate iz seznama potrebne_enote;
    return potrebne_enote;
}
```

Algoritem: Poišči pomožne enote;

Vhod: graf domene, model uporabnika, trenutna učna enota u;
Izhod: seznam povezav na ustrezne učne enote;

```
Telo: {
    c = M_m(u);
    potrebne_enote = [];
    for ( vse c_i, c_i ≺_0 c, c_i = M_m(u_i) ) {
        if ( enota u_i je "pripravljena na učenje" ali "pogojno pripravljena na učenje" )
            potrebne_enote = potrebne_enote + [ u_i ];
    }
    return potrebne_enote;
}
```

V seznam "predpogojne_enote", ki predstavlja tretjo skupino povezav, so vključene povezave na vse že naučene ali poznane neposredne predpogojne enote (tako nujne kot podporne). Ta seznam se vedno sestavi ne glede na izobraževalni status izbrane učne enote.

Algoritem: Poišči predpogojne enote;

Vhod: graf domene, model uporabnika, trenutna učna enota u;
Izhod: seznam povezav na ustrezne učne enote;

Telo: {
$c = M_m(u)$;
predpogojne_enote = [];
for (vse c_i, $c_i \prec_{wi} c$, $c_i = M_m(u_i)$) {
if (enota u_i je "naučena" ali "poznana")
predpogojne_enote = predpogojne_enote + [u_i];
}
sortiraj seznam predpogojne_enote po padajoči vrednosti w_i ;
return predpogojne_enote;
}

Vse vstavljene povezave, ki so pomensko enakovredne, so razvrščene po moči povezave, to je po padajoči vrednosti *w*, saj so si enote pri močneje povezanih konceptih vsebinsko bližje.

Opisano tehnologijo prilagodljivega vstavljanja povezav lahko štejemo med vrste prilagodljive podpore navigaciji, saj deluje na nivoju povezav do različnih hipermedijskih strani. Ta tehnologija se sicer zgleduje tako po tehnologiji skrivanja povezav kot po tehnologiji razvrščanja povezav, a se hkrati od obeh tudi precej razlikuje. Pri skrivanju povezav gre namreč za to, da so povezave na strani sicer prisotne, a jih po potrebi onemogočimo ali pa jih sploh ne prikažemo. Take povezave so lahko popolnoma odstranjene, kjer odstranimo tako oznako povezave (besedilo) kot samo povezavo. Lahko pa povezava ostane na svojem mestu, a je neoznačena, ali pa je označena povezava onemogočena. Nasprotno pa pri vstavljanju povezav sama stran praviloma sploh ne vsebuje povezav na druge strani, saj so te ločene od vsebine strani. Zato tudi ne moremo govoriti o njihovem skrivanju. V tem primeru povezave dodamo naknadno, katere povezave dodamo posamezni strani, pa je odvisno od same strani in trenutnega uporabnikovega znanja domenskih konceptov in se dinamično spreminja. Vse dodane povezave so prave povezave, torej oznaka povezave (besedilo), za katero stoji omogočena povezava (kar pri skrivanju ne velja vedno). Po drugi strani pa pri vstavljanju povezav ne gre niti za razvrščanje povezav, čeprav so vstavljene povezave urejene po nekih zakonitostih, ki pa niso nujno najprimernejša povezava. Pri tem razvrstimo le vstavljene povezave, ki se spreminjajo glede na model uporabnika, ne pa vseh povezav iz dane strani, kot pri prilagodljivem razvrščanju.

5.5.4 Neposredno vodenje

Sistem omogoča linearno navigacijo preko tehnologije neposrednega vodenja. Ta tehnologija je sicer povzeta po inteligentnih učnih sistemih in uporabniku ne dopušča pretirane svobode, a je včasih lahko zelo koristna pomoč sistema predvsem začetnim, manj izkušenim uporabnikom. V prilagodljivih hipermedijih je priporočljiva njena uporaba predvsem v kombinaciji z drugimi tehnologijami prilagodljive podpore navigaciji.

Neposredno vodenje dinamično izbere naslednji primeren učni korak za uporabnika. Po njem lahko uporabnik poseže, kadar želi več vodstva pri svojem učenju, kar pomeni bolj vodeno poučevanje (s strani sistema). V takem primeru mu sistem predlaga naslednjo učno enoto za branje ali pa reševanje primernega testa.

Podobno kot prilagodljivo vstavljanje povezav tudi neposredno vodenje temelji na izobraževalnem stanju učne enote. Tako je izbira naslednje enote odvisna od stanja trenutne enote, njenih predpogojnih enot in enot, katerim je predpogoj. Algoritem je sicer podoben algoritmu za iskanje naslednje učne enote pri vstavljanju povezav (enote iz prvega seznama), le da je nekoliko razširjen. Če ne najdemo nobene ustrezne enote, ki je v predpogojni relaciji s trenutno enoto, izberemo naslednjo enoto preko hierarhije učnih enot v tečaju. Algoritem mora namreč vedno vrniti neko enoto, razen v primeru, ko so naučene že vse enote tečaja (takrat zaključimo s tečajem).

Poleg možnosti izbire naslednjega koraka omogoča sistem tudi ponavljanje oziroma pregled nad že narejenimi koraki. Sistem si namreč zapomni tudi vso zgodovino korakov od začetka seje, to je od zadnje prijave v sistem. Po tem seznamu predhodno obiskanih enot se uporabnik lahko ciklično sprehaja.

5.5.5 Kazalo vsebine in indeks

Kazalo vsebine nudi uporabniku možnost hierarhične navigacije po učnem materialu. Vse učne enote namreč tvorijo hierarhično strukturo poglavij, podpoglavij in razdelkov (kot v klasičnih učbenikih), ki se odraža tudi v kazalu vsebine. Naslovi posameznih učnih enot, ki jih vsebuje kazalo, so povezave na ustrezne enote in so tudi barvno anotirane glede na izobraževalno stanje enote. Poleg tega so posebej označene tudi trenutna učna enota (siv pravokotnik za ozadje) in vse že obdelane učne enote (s kljukico).

Za razliko od kazala vsebine, ki prikazuje hierarhijo učnih enot, se indeks osredotoči na koncepte domene in s tem omogoča na konceptih temelječo navigacijo. Domenski koncepti v indeksu sestavljajo linearen seznam, ki je urejen po abecedi (po imenih konceptov). Vsakemu konceptu indeksa je pripisano tudi ime učne enote, ki ima ta koncept za glavni koncept enote, ter v oklepaju tudi izobraževalno stanje te enote. Imena enot so hkrati tudi povezave na ustrezno učno gradivo (hipermedijsko stran enote) in so barvno anotirane. V kolikor je uporabnik neko enoto označil za obdelano, se to v indeksu odraža s kljukico ob imenu enote.

5.6 Opis sistema *ALICE*

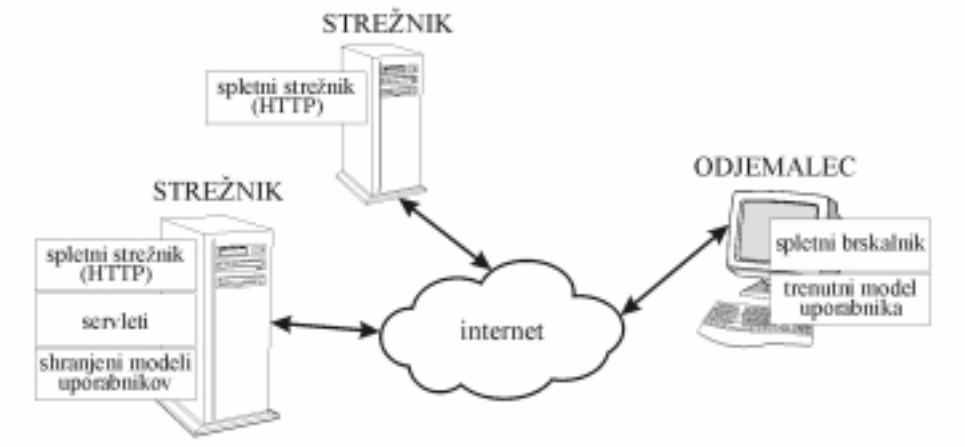
Opisani prototip prilagodljivega sistema smo realizirali z učnim sistemom, katerega smo poimenovali *ALICE* (iz angl. Adaptive Link Insertion in Concept–based Educational system). V tem razdelku bomo opisali sistem, njegov izgled in delovanje, pri opisu pa bomo uporabljali učno domeno, ki smo jo uporabili tudi za njegovo testiranje. Podrobneje je uporabljena učna domena opisana v naslednjem poglavju.

Sistem *ALICE* je spletni sistem, sestavljen iz dveh delov: strežniške in odjemalske aplikacije. Je popoln spletni sistem in na odjemalčevi strani za delovanje ne zahteva drugega kot spletni brskalnik (zaenkrat je prototip podprt le v brskalniku Microsoft Internet Explorer). Strežniški del spletne aplikacije je implementiran kot javanski servlet (servleti so apleti, ki se izvajajo na strežniku). Zato mora na strežniški strani poleg spletnega strežnika delovati tudi strežnik, ki omogoča izvajanje servletov (prototip sistema deluje na strežniku JSWDK, JavaServer Web Development Kit). Pri izdelavi sistema smo uporabljali, poleg opisnega jezika HTML za sestavo spletnih strani, tudi skriptni jezik JavaScript in programski jezik Java.

Strežniški del aplikacije sestavljajo trije servleti. Prvi preverja vpisano uporabniško ime in geslo ter dovoli ali zavrne dostop do sistema. Drugi servlet je namenjen obdelavi rezultatov predtesta in določitvi začetnega znanja uporabnika. Ta se izvede le pri prvi prijavi uporabnika v sistem. Tretji servlet pa shrani vrednosti modela uporabnika po končanem delu (izhodu iz sistema), ki se kasneje uporabijo pri ponovni prijavi istega uporabnika v sistem.

Glavnina aplikacije se izvaja na odjemalčevi strani, kjer poteka tudi celotno računanje, posodabljanje in vzdrževanje modela uporabnika. Strežnik se uporablja le na začetku dela za preverjanje identitete uporabnika in na koncu dela za shranjevanje rezultatov dela. Med samim delom s sistemom pa nudi spletni strežnik (oziroma različni spletni strežniki) le ustrezne spletne

strani. Tak način dela razbremeni delovanje strežnika, saj izkoristi odjemalčeve vire tudi za vodenje modela uporabnika, s tem pa hkrati zmanjša promet po mreži. Na sliki 5.5 je poenostavljen prikaz opisane dejavnosti.



Slika 5.5
Vlogi strežnika in odjemalca v sistemu *ALICE*

Pri opisu samega sistema bomo začeli kar na začetku. Ko v spletnem brskalniku odpremo uvodno spletno stran [Kavčič 2001], dobimo kratek opis vsebine tečaja, ki se izvaja s pomočjo sistema, in začetna navodila. Vsak uporabnik se mora prijaviti v sistem, saj vsakemu pripada svoj model uporabnika, zato se morajo posamezni uporabniki med seboj razlikovati. Uporabniško ime in geslo lahko izbere uporabnik sam.

Slika 5.6 prikazuje uvodno stran, ki poleg kratkih navodil nudi tudi prijavo v sistem (uporabnik vpiše svoje ime in geslo). V primeru na sliki (ter v vseh nadaljnjih primerih) je prikazan tečaj *Uvod v programski jezik Java*, s pomočjo katerega smo sistem tudi testirali. Za sam opis sistema in njegovo delovanje pa vsebina tečaja ni bistvena.

Slika 5.6

Uvodna stran sistema *ALICE* s prijavo v sistem

Ob prvi prijavi v sistem se uporabniku najprej prikaže začetni uvrstitveni test (predtest), sicer pa se takoj odpre glavno okno sistema. To je razdeljeno na tri dele (trije okvirji), kot prikazuje slika 5.7.

Slika 5.7

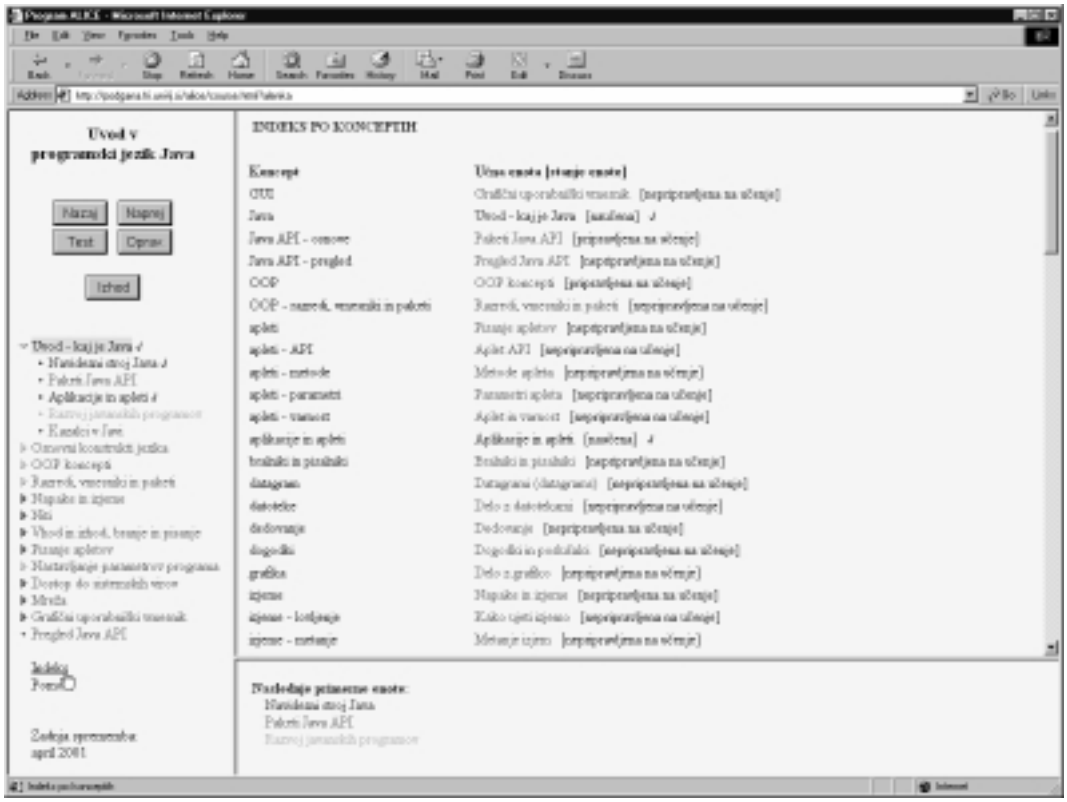
Glavno okno sistema *ALICE*

Prvi okvir (skrajno levo) zajema naslov tečaja, pet navigacijskih gumbov, kazalo vsebine tečaja ter povezavi na indeks in pomoč. V drugem okvirju (desno spodaj) se prikazujejo povezave na druge učne enote, ki so povezane s trenutno prikazano enoto. Te povezave določimo s pomočjo tehnologije prilagodljivega vstavljanja povezav in so razdeljene v tri skupine (povezave na naslednje učne enote, povezave na potrebno predznanje ter povezave na že osvojeno predznanje). Največji del glavnega okna (zgornji desni okvir) pa uporabimo za prikazovanje vsebine posameznih učnih enot tečaja.

Kazalo vsebine tečaja hierarhično strukturira poglavja, podpoglavja in lekcije tečaja. Vse enote so izpisane s svojim imenom, pred katerim stoji ikona. Enote, ki vsebujejo druge podenote, so označene s trikotnikom, tiste brez podenot pa s krogom. S klikom na omenjen trikotnik lahko uporabnik po potrebi prikaže oziroma skrije podenote. Barvo ikone določajo izobraževalna stanja enote, pred katero stoji ikona, in tudi vseh njenih podenot. V kazalu vsebine so posebej označene (s kljukico) tudi vse enote, ki so že obdelane (prebrane), ter trenutno izbrana in prikazana enota (na sivem ozadju).

Povezave v kazalu vsebine, v skupinah povezav na sorodne enote in v indeksu so barvno anotirane. Barvo povezave določa uporabnikovo poznavanje snovi tečaja, ki se sproti tudi spreminja. Pomen posameznih barv pri označevanju povezav, ki smo ga opisali že v razdelku 5.5.2, temelji na izobraževalnem stanju enote, ki stoji za povezavo. V primeru s slike 5.7 je trenutno prikazana enota *Kazalci v Javi* še nepripravljena za branje (učenje), saj še ni osvojeno vse potrebno predznanje (enoti *Razredi, vmesniki in paketi* ter *Spremenljivke*), čeprav je predpogojna enota *Uvod – kaj je Java* že naučena (z uspešno rešenim testom). Na še neosvojeno predznanje opozarjajo tudi povezave na enote, ki razlagajo potrebno predznanje.

Indeks po konceptih tečaja (glej sliko 5.8) je pravzaprav po abecedi urejen seznam vseh konceptov tečaja, ki vključuje tudi povezave na ustrezne učne enote, s pripisanim izobraževalnim stanjem. Tudi tu so posebej označene že obdelane enote.



Slika 5.8

Indeks po konceptih domene

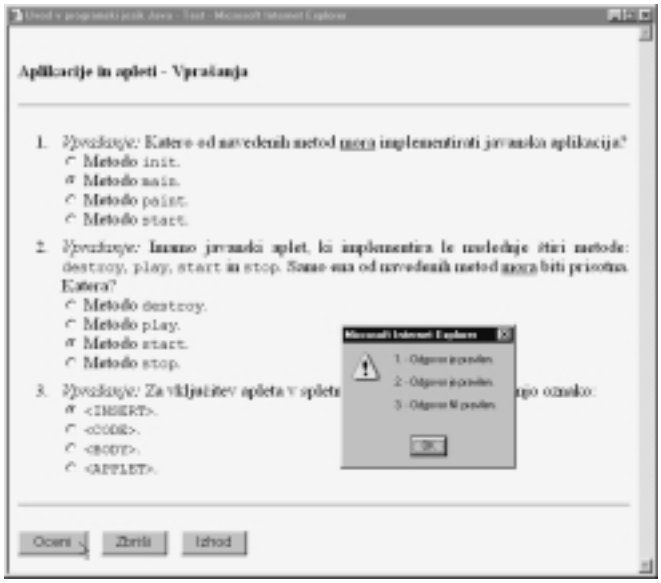
Pomoč v sistemu je omejena le na osnovna navodila za uporabo sistema ter razlago pomena posameznih gumbov in barvnih oznak.

Linearno navigacijo skozi učno domeno omogočata gumb *Naprej* in gumb *Nazaj*. Prvi implementira neposredno vodenje sistema (razdelek 5.5.4) in prikaže naslednjo (za učenje najprimernejšo) učno enoto. Drugi pa omogoča pregled zgodovine obiskanih enot, saj uporabnika vrne na predhodno obiskano učno enoto.

Gumb *Oprav.* je namenjen označevanju predelanih (prebranih) enot. Vsakič, ko uporabnik prebere vsebino neke učne enote, lahko s klikom na ta gumb označi, da je to enoto že predelal. Gumb je potreben zato, da ločimo med naključno (pomotoma) izbrano in prikazano enoto in enoto, ki jo uporabnik tudi zares prebere in predela.

Do sedaj smo govorili predvsem o učnih enotah, ki pojasnjujejo nek koncept (razlaga). Večino teh enot pa dopolnjujejo tudi testi, ki preverjajo poznavanje njihove vsebine. Pri vsaki izbrani učni enoti (razlagi) lahko pridemo do ustreznega testa s klikom na gumb *Test*. Tega uporabimo, ko želimo preveriti poznavanje trenutno prikazane enote. Prikaže se novo okno z vprašanji za dano učno enoto.

Na sliki 5.9 je primer testnih vprašanj za učno enoto *Aplikacije in apleti*. Vsa vprašanja zahtevajo izbiro (enega) pravilnega odgovora izmed več možnih. Ko uporabnik izbere odgovore, mu sistem odgovori s podatki o pravilnosti odgovorov, pri nepravilnih odgovorih pa lahko nudi tudi nasvet, ki pomaga razumeti, zakaj je odgovor napačen (namig k pravilni rešitvi). Izbiro in oceno odgovorov lahko uporabnik ponavlja toliko časa, dokler ne izbere vseh pravilnih odgovorov ali pa prekine testiranje (v tem primeru se rezultati ne upoštevajo).



Slika 5.9

Vprašanja za preverjanje znanja – ocena pravilnosti odgovorov

Ko uporabnik na vsa vprašanja odgovori pravilno, mu sistem prikaže tudi njegovo uspešnost pri odgovarjanju, kjer se upoštevajo tudi vsi zgrešeni odgovori (slika 5.10). Tako izračunana uspešnost je enaka stopnji pripadnosti množici naučenih konceptov, izraženi v odstotkih (podrobnosti so v razdelku 5.4.2.2).

Slika 5.10

Vprašanja za preverjanje znanja – skupna uspešnost pri odgovorih

Učne enote za učenje izbira uporabnik sam. Na voljo ima več načinov: enoto lahko izbere v kazalu vsebine (hierarhična navigacija), med povezavami na sorodne učne enote (relacijska navigacija), z gumboma *Naprej* in *Nazaj* (linearna navigacija) ali pa preko indeksa konceptov domene (na konceptih temelječa navigacija).

5.7 Primerjava z drugimi prilagodljivimi sistemi

Sistem *ALICE* ima veliko skupnega z drugimi prilagodljivimi izobraževalnimi sistemi, hkrati pa se v nekaterih pogledih od njih tudi močno razlikuje. Primerjava je najbolj nazorna, če naš sistem umestimo med druge sisteme, ki smo jih opisali v razdelku 3.5 in njihove glavne značilnosti povzeli v tabeli 3.1. Podobno kot večina opisanih sistemov je tudi sistem *ALICE* popoln spletni sistem, ki je zasnovan za poljubno domeno učenja. Tudi v našem sistemu temelji model domene na konceptnem grafu z označenimi povezavami, saj med koncepte uvedemo dve vrsti relacij – nujne in podporne predpogojne relacije. Tukaj pa se pokaže že prva večja razlika.

Nujne predpogojne relacije so namreč definirane kot mehke relacije, zato je tudi graf domene mehka relacijska struktura.

Še večje razlike pa so v pristopu k modeliranju uporabnika. Ta sicer temelji na prekrivnem modelu preko modela domene in zajema uporabnikovo znanje, vendar se pri tem opira na mehke množice in mehka pravila. Znanje uporabnika določajo pripadnostne funkcije trem mehkim množicam: množici naučenih, poznanih in nepoznanih konceptov. S tako predstavljenim znanjem uporabnika tudi "računamo" (nad njim izvajamo operacije), saj pri posodabljanju modela uporabnika upoštevamo tako koncept, katerega vrednost se je neposredno spremenila, kot tudi njegov vpliv na druge koncepte, ki so z njim v predpogojni relaciji. V ta namen smo razvili mehanizem sklepanja, ki deluje na podlagi mehkih pravil za razširjanje vrednosti znanja konceptov. Princip je podoben kot pri Bayesovih mrežah, le da se vrednosti razširjajo na podlagi mehkih pravil preko grafa predpogojno povezanih konceptov. Takega pristopa k modeliranju uporabnika nismo zasledili pri drugih prilagodljivih izobraževalnih sistemih. Podoben način sicer uporablja sistem Hypernet, a le v toliko, da si pri modeliranju pomaga z bazo pravil, ki uporablja kombinacijo delovanja nevronskih mrež z mehkimi pravili.

Sistem *ALICE* se prilagaja uporabniku z nudenjem podpore navigaciji. Uporabljene tehnologije zajemajo neposredno vodenje, kazalo vsebine in indeks, barvno anotacijo povezav ter vstavljanje povezav. Vse naštete so klasične tehnologije prilagodljivih hipermedijev, ki se jih poslužuje tudi veliko izobraževalnih sistemov. Izjema je le zadnja, to je vstavljanje povezav, ki je neke vrste kombinacija skrivanja in razvrščanja povezav in temelji na povezavah, ločenih od vsebine strani. K vsaki strani se posebej dodajo ustrezne skupine povezav, odvisno od uporabnikovega poznavanja domenskih konceptov. Tehnologiji vstavljanja povezav je še najbliže način, ki ga uporablja sistem Hypernet. Ta povezave prikazuje ločeno od samega besedila vozlišča, razdeli pa jih na neposredne povezave, hierarhične povezave in povezave na sorodni material. Tudi sistem MetaLinks [Murray 2000b] ima povezave ločene od vsebine strani in jih dodaja glede na uporabnikovo pot skozi sistem. Dodane povezave kažejo na očeta, brate in sinove v hierarhiji strani. Sistem MetaLinks, ki ne uporablja modela uporabnika, ni pravi prilagodljiv sistem, zato ga tudi nismo posebej omenjali. Za razliko od omenjenih dveh sistemov se skupine vstavljenih povezav v sistemu *ALICE* ne razlikujejo le po tipu povezav, temveč tudi po njihovi namembnosti.

6. Testiranje sistema

V prejšnjem poglavju opisan prototip prilagodljivega sistema smo tudi ovrednotili na določeni testni učni domeni. Zanimalo nas je predvsem splošno mnenje uporabnikov o sistemu in njegovi uporabnosti, poleg tega pa smo želeli preveriti tudi vpliv uporabe prilagodljivega sistema na uspešnost učenja. Pri ocenjevanju smo se osredotočili na učinkovitost prilagodljivega vstavljanja povezav in barvne anotacije povezav.

V ta namen smo pripravili testno učno domeno (začetni tečaj o programskem jeziku Java), ki nam je omogočila testiranje sistema v realnem okolju. Nato smo prototip prilagodljivega sistema uporabili v praksi na skupini študentov, začetnikov v programskem jeziku Java.

6.1 Testna učna domena

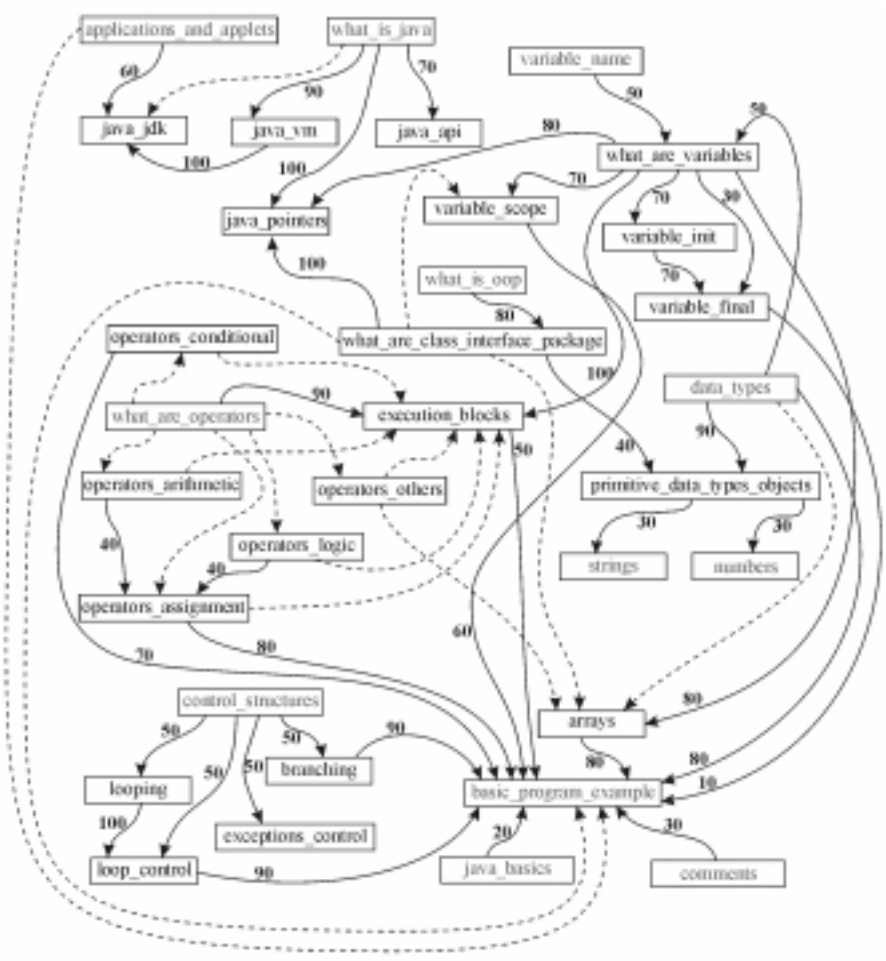
Sistem *ALICE* predstavlja le ogrodje prilagodljivega hipermedijskega sistema, kateremu moramo dodati določeno vsebino – učno domeno tečaja. Domeno tečaja lahko poljubno izberemo, pri tem pa moramo določiti naslednje: množico domenskih konceptov, njihove medsebojne relacije, učne enote z razlago, ki pripadajo posameznim konceptom, in po potrebi tudi testna vprašanja, hierarhično strukturo učnih enot ter predtest za začetno določitev znanja uporabnika.

Problema izdelave vsebine tečaja se lahko lotimo na dva načina. V prvem primeru najprej določimo učno domeno, jo razgradimo na temeljne enote – koncepte ter določimo medsebojne predpogojne relacije med koncepti (skupaj z utežmi). Potem za vsak koncept sestavimo eno učno enoto, ki ta koncept opisuje in razlaga, poleg nje pa sestavimo tudi eno ali več vprašanj, ki preverjajo poznavanje tega koncepta. Učne enote smiselno uredimo v hierarhično strukturo tečaja (ki pa je neodvisna od predpogojne povezanosti konceptov) in na koncu vse skupaj združimo z našim sistemom.

Drug način pa izhaja iz že obstoječega materiala za tečaj (učnih enot), na podlagi katerega določimo množico konceptov, ki so ključni v dani domeni in ki pokrivajo celotno domeno, ter njihove predpogojne relacije, skupaj z utežmi. Pripravljenim učnim enotam po potrebi dodamo vprašanja za preverjanje znanja, ki ga te enote razlagajo, ter vsakemu konceptu priredimo eno

učno enoto, ki opisuje predvsem ta koncept. Nazadnje določimo še hierarhijo učnih enot in vse skupaj združimo s sistemom.

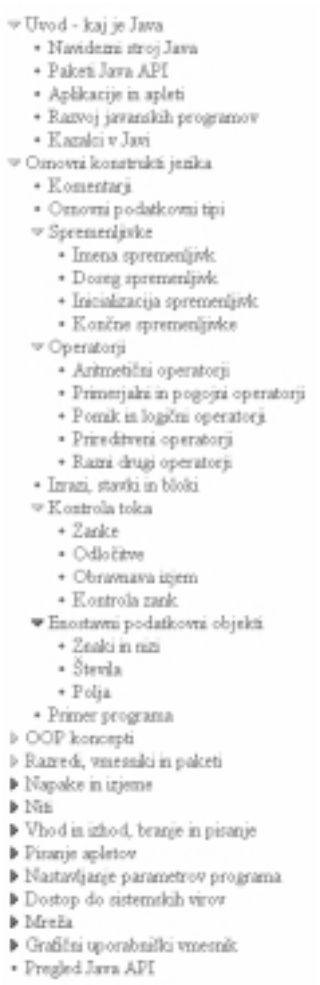
Pristop, ki izhaja iz že obstoječega učnega materiala, je krajši in hitrejši način izdelave tečaja, saj izkorišča že pripravljen spletni material. Tudi pri pripravi naše testne domene smo uporabili tak pristop. Za učno domeno smo izbrali začetni tečaj o programskem jeziku Java, učni materiali pa se naslanjajo na dva učbenika o jeziku Java [Marolt 2000, Campione 2000], ki sta na voljo tudi v elektronski obliki v svetovnem spletu. Za naše potrebe smo prvi učbenik tudi malce priredili, drugega pa, zaradi upoštevanja avtorskih pravic, nismo spreminjali. Na podlagi spletnih materialov obeh učbenikov smo določili množico domenskih konceptov ter sestavili testna vprašanja. Slednja smo uporabili tudi pri sestavi predtesta.



Slika 6.1

Del konceptov (koncepti prvega in drugega poglavja) domene tečaja *Uvod v programski jezik Java* in njihove predpogojne relacije z utežmi

Našo domeno učenja tako sestavlja 97 konceptov, od katerih je 9 konceptov osnovnih in so brez predpogojev. Del učne domene (33 konceptov) prikazuje slika 6.1, kjer so osnovni koncepti označeni z zeleno, končni koncepti pa z rdečo barvo. Nujne predpogojne relacije med koncepti ponazarja puščica s pripisano utežjo (izraženo v odstotkih), podporne predpogojne relacije pa so označene s črtkano povezavo.



Slika 6.2

Kazalo vsebine tečaja *Uvod v programski jezik Java*

Celoten tečaj *Uvod v programski jezik Java* sestoji iz 13 poglavij, ki so razdeljena na podpoglavja in lekcije. Hierarhijo tečaja prikazuje slika 6.2, ki je (črno–bel) posnetek kazala vsebine s prikazanimi vsemi podenotami prvih dveh poglavij. Vseh učnih enot je skupaj toliko, kot je domenskih konceptov (97), saj vsak koncept opisuje natanko ena enota. V slovenščini je pripravljenih 48 enot, ki se nahajajo na istem spletnem strežniku kot sistem. Ostale enote (49)

pa so v angleščini, saj so del neodvisnih materialov o jeziku Java z drugih spletnih strežnikov. Za 44 učnih enot smo pripravili tudi pripadajoča testna vprašanja.

6.2 Testne različice sistema

Za testiranje smo pripravili tri različice sistema, ki smo jih označili z A, B in C. Pri vseh različicah je vsebina tečaja (učne enote) ista, prav tako pa so na razpolago ista testna vprašanja pri enotah. Tudi izgled vseh treh različic je podoben, razen tistih podrobnosti, ki izhajajo iz razlik med njimi. Gumbi *Test*, *Oprav.* in *Nazaj* so obdržali isto funkcionalnost v vseh treh različicah.

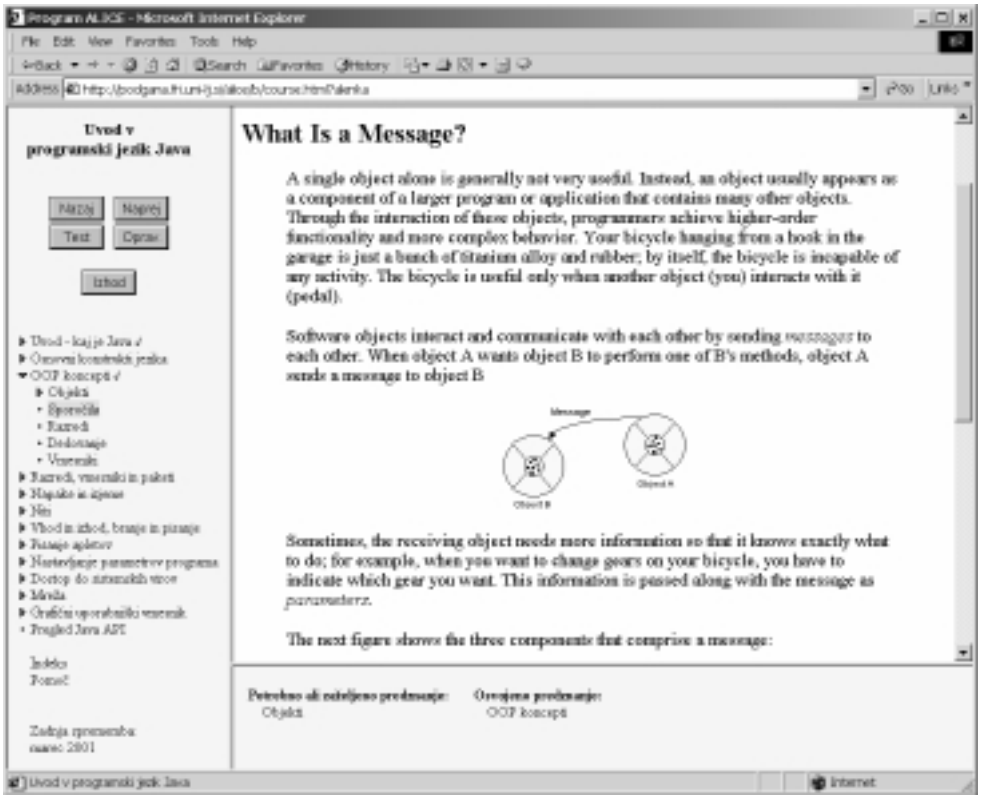
Slika 6.3

Glavno okno sistema, različica A

Različica A, katere glavno okno sistema prikazuje slika 6.3, je poln prilagodljiv sistem, ki omogoča vso funkcionalnost opisanega sistema *ALICE*. Vključuje barvno anotacijo povezav na enote, povezave na sorodne učne enote ter gumb *Naprej*, ki prikaže naslednjo najprimernejšo

enoto za učenje. Prilagajanje sistema deluje na podlagi modela uporabnika, ki temelji na mehkih množicah in mehkih pravilih sklepanja o uporabnikovem znanju. Uporabnik si lahko pri navigaciji po materialu pomaga s kazalom vsebine, s povezavami na sorodne učne enote, z gumboma *Naprej* in *Nazaj* ali pa z indeksom konceptov domene.

Tudi druga različica sistema (B) je zelo podobna različici A. Edina razlika je le v barvnem označevanju povezav, saj so v različici B vse povezave in ikone črne barve, ne glede na njihovo izobraževalno stanje. Različica B je torej nekoliko okrnjen prilagodljiv sistem, ki ne omogoča barvne anotacije povezav (izobraževalno stanje učnih enot se neposredno prikaže le v indeksu). Okno te različice sistema prikazuje slika 6.4, kjer je vsebina prikazane lekcije *Sporočila* iz tujih spletnih virov.

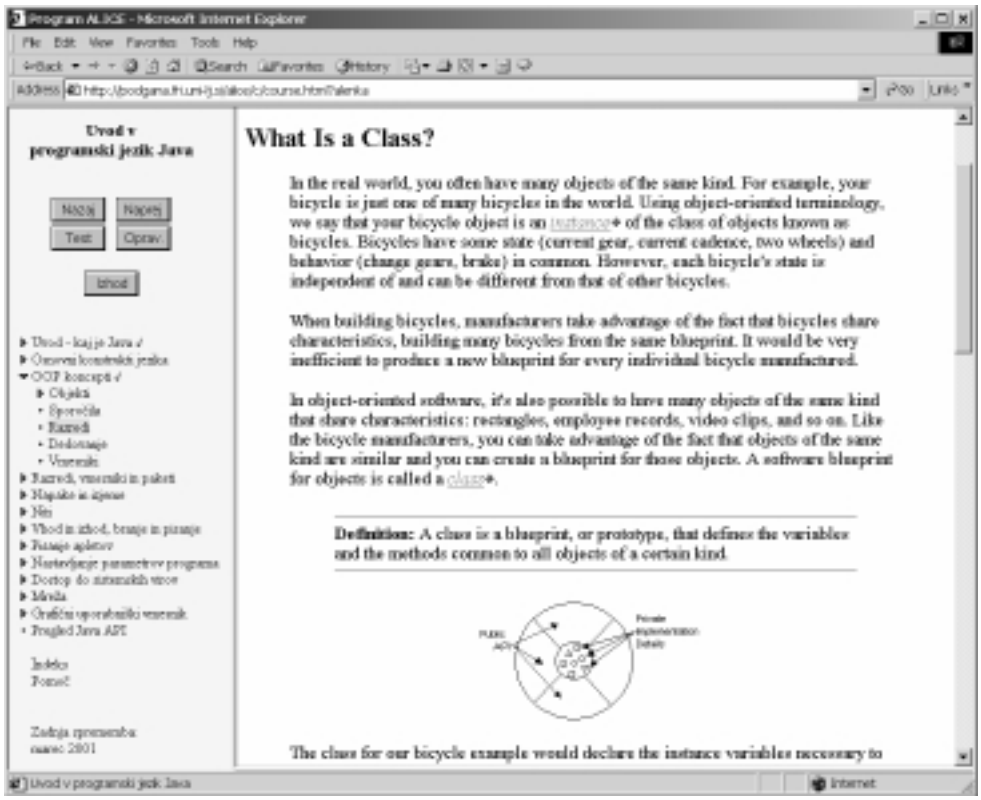


Slika 6.4

Glavno okno sistema, različica B

Tretja različica sistema (različica C) pa predstavlja popolnoma neprilagodljiv sistem. Ta sistem ne omogoča barvne anotacije povezav (vse povezave in ikone so črne), vstavljanja povezav (ni okvirja s povezavami na sorodne učne enote) ali neposrednega vodenja. Gumb *Naprej* sicer ostaja, a vedno prikaže naslednjo enoto v hierarhiji učnih enot. Ker je ta različica

neprilagodljiva, se obnaša enako pri vseh uporabnikih (ne glede na njihovo poznavanje domene). Zato tudi beleženje uporabnikovega poznavanja posameznih enot in modeliranje uporabnikovega znanja ni smiselno. V pomoč uporabniku pri navigaciji skozi material so kazalo vsebine, gumba *Naprej* in *Nazaj* ter indeks konceptov domene (brez označenega izobraževalnega stanja enote, ki ga v tej različici ne določamo). Primer glavnega okna različice C prikazuje slika 6.5.



Slika 6.5

Glavno okno sistema, različica C

Če povzamemo, za testiranje smo pripravili tri različice sistema. Različica A je poln prilagodljiv sistem, različica B je prilagodljiv sistem brez barvne anotacije povezav, različica C pa popolnoma neprilagodljiv sistem.

6.3 Potek testiranja in meritve

Z uporabo treh testnih različic sistema smo izvedli poizkus, s katerim smo želeli ugotoviti, ali način učenja (uporaba posamezne različice sistema) vpliva na dosežen rezultat pri zaključnem testu. Zanima nas torej, ali ima eksperimentalni faktor "uporaba različice sistema" kakšen učinek na povprečno vrednost dosežka pri zaključnem testu. Poleg tega smo s pomočjo vprašalnika poskušali zbrati tudi splošne ocene sistema in njegove uporabnosti po mnenju uporabnikov.

V raziskavi, katero smo izvedli konec marca 2001, je sodelovalo 84 študentov prvega letnika Fakultete za računalništvo in informatiko, ki se v okviru predmeta Programiranje II učijo programska jezika C in Java. Vsi študenti so se poizkusa udeležili prostovoljno, doseženi rezultati pri opravljenih testih pa nikakor niso vplivali na uspeh pri predmetu. Poleg tega je bilo testiranje anonimno, vsak udeleženec se je identificiral le s posebej za to določenim uporabniškim imenom.

Poizkus je potekal v računalniški učilnici na osebnih računalnikih, učni sistemi pa so bili dosegljivi preko spleta. Vse tri različice sistemov so se nahajale na istem spletnem strežniku in so uporabljale isti učni material. Zaradi prostorske stiske (učilnica sprejme le do 30 uporabnikov) smo poizkus izvedli v štirih delih. Razen časovne razlike med njimi ni bilo drugih razlik, v vsakem pa smo preizkušali vse tri različice sistema.

Vse udeležence poizkusa smo na slepo razdelili na tri bolj ali manj enakomerno zastopane skupine, vsaki skupini pa smo naključno dodelili eno od različic sistema A, B ali C. Tako je različico A uporabljalo 29 udeležencev, različico B 27 udeležencev (eksperimentalni skupini), različico C pa je uporabljalo 28 udeležencev (kontrolna skupina).

Vsak udeleženec je dobil list z osnovnimi navodili, na katerem je bila označena različica sistema in zapisano uporabniško ime, pod katerim naj bi uporabljal sistem. Navodila so bila kratka in jasna: kako začeti delo s sistemom, na katerem naslovu se sistem nahaja, katere lekcije naj bi predelali, koliko časa je na voljo ter kje se nahajata zaključni test in vprašalnik, ki naj bi ju rešili na koncu. Vsi udeleženci so najprej reševali uvodni test, ki je služil za začetno postavitev modela uporabnika, njegove rezultate pa smo uporabili tudi za primerjavo začetnega znanja udeležencev. Potem so samostojno uporabljali sistem z navodilom, da naj predelajo lekcije prvih dveh poglavij, ki skupaj zajemajo 31 konceptov. Na koncu so reševali še zaključni test, kateri je glavni vir podatkov našega eksperimenta. Celoten poizkus so zaključili z izpolnjevanjem vprašalnika, v katerem so med drugim navedli svoje predhodno poznavanje

programskih jezikov, izkušnje s spletnimi brskalniki ter ocenili izgled in delovanje sistema. Vse podrobnosti v povezavi z uporabo učnega sistema so dobili na uvodnih spletnih straneh sistema. Navodila so se nekoliko razlikovala, saj so bila odvisna od same različice sistema. Celoten eksperiment je trajal približno dve uri.

Uvodni test in zaključni test sta bila ista za vse tri skupine. Tudi vprašalnik je bil v osnovi enak, razlikoval se je le pri vprašanjih, ki so bila neposredno vezana na posamezno različico sistema. Za vsakega uporabnika sistema smo odgovore uvodnega testa in zaključnega testa (po posameznih vprašanjih) skupaj z doseženim rezultatom shranili ločeno v dveh datotekah na strežniku. Tudi vse odgovore na vprašalnik smo shranili na strežniku. Poleg tega pa smo v posebnem dnevniku beležili vse akcije uporabnika med delom s sistemom, ki je k zbirki podatkov o uporabniku na strežniku prispeval še eno datoteko. Tako smo za vsakega uporabnika dobili štiri datoteke s pomembnimi podatki za analizo.

V poizkusu je sodelovalo skupaj 84 študentov, vendar smo štiri rezultate (dva iz skupine A ter po enega iz skupin B in C) zaradi nepopolnosti podatkov zavrnili (nismo dobili rezultatov vseh testov). Na koncu smo tako dobili 80 popolnih rezultatov poizkusa, po 27 v skupinah A in B, 26 pa v skupini C.

6.4 Rezultati testiranja

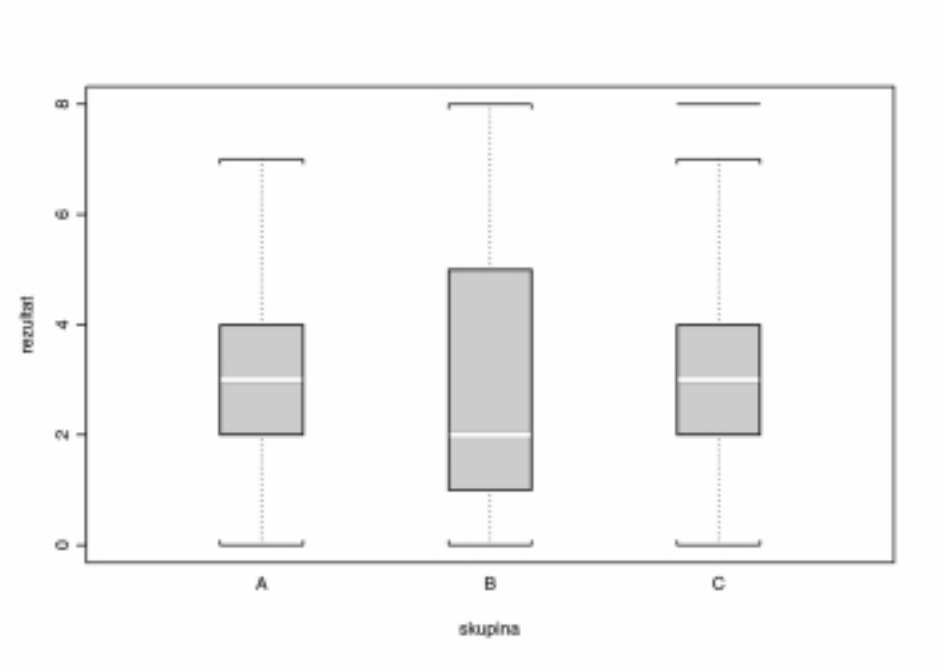
V izvedenem poskusu smo dobili rezultate uvodnih testov, zaključnih testov ter izpolnjene vprašalnike 80 udeležencev, od katerih jih je bilo po 27 v skupinah A in B ter 26 v skupini C. Vsaka skupina določa uporabo druge različice sistema, ki v našem primeru pomeni obdelavo.

Naj še enkrat povzamemo potek poskusa. Na 80 študentih smo preizkusili tri učne postopke (uporaba določene različice učnega sistema). Študente (poskusne enote) smo na slepo razdelili na 3 skupine, vsako skupino pa smo podvrgli enemu izmed učnih postopkov (obdelav). Po končanem postopku učenja preverimo znanje študentov in dobimo rezultate zaključnega testa (podatke).

V tem razdelku bomo analizirali dobljene podatke, ocenili učinke obdelav in predstavili prve ugotovitve.

6.4.1 Analiza rezultatov uvodnega testa

Glede na to, da so bili študenti, ki so sodelovali v poizkusu, naključno razdeljeni v tri skupine, pri rezultatih uvodnega testa (predtesta) ni bilo pričakovati bistvenih razlik med posameznimi skupinami. Podrobnejša analiza doseženih rezultatov pri testu je naše domneve podprla.



Slika 6.6
Rezultati uvodnega testa po skupinah

Veliko o rezultatih uvodnega testa nam pove slika 6.6, na kateri je prikazana primerjava povprečij doseženih točk v posameznih skupinah. Vsak test je prinesel največ 10 točk. Iz slike je razvidno, da je razpon doseženih točk (variacijski razmak) precej velik, saj znaša v skupini A od 0 do 7 točk, v skupinah B in C pa od 0 do 8 točk. Mediane so 3, 2 in 3 za skupine A, B in C po vrsti. Povprečno število doseženih točk je v skupini A 3.00, v skupini B 2.81 ter v skupini C 3.11. Standardni odkloni pa so v skupini A 1.90, v skupini B 2.39 ter v skupini C 1.93. Ta osnovna statistika na rezultatih uvodnega testa je povzeta tudi v tabeli 6.1, kjer so obdelane vse posamezne skupine in vsi rezultati skupaj.

Tabela 6.1 – Rezultati uvodnega testa po skupinah

skupina	število udeležencev	povprečje (od 10)	mediana	variacijski razmak	standardni odklon	std. povpr. napak
A	27	3.00	3	0 – 7	1.90	0.3659
B	27	2.81	2	0 – 8	2.39	0.4593
C	26	3.11	3	0 – 8	1.93	0.3776
skupaj	80	2.96	3	0 – 8	2.06	0.2306

Slika 6.6 nam že na pogled da slutiti, da med rezultati v posameznih skupinah ni velikih razlik. To je potrdila tudi analiza variance, ki je povzeta v tabeli 6.2 [Frigon 1997].

Tabela 6.2 – Analiza variance rezultatov uvodnega testa po skupinah

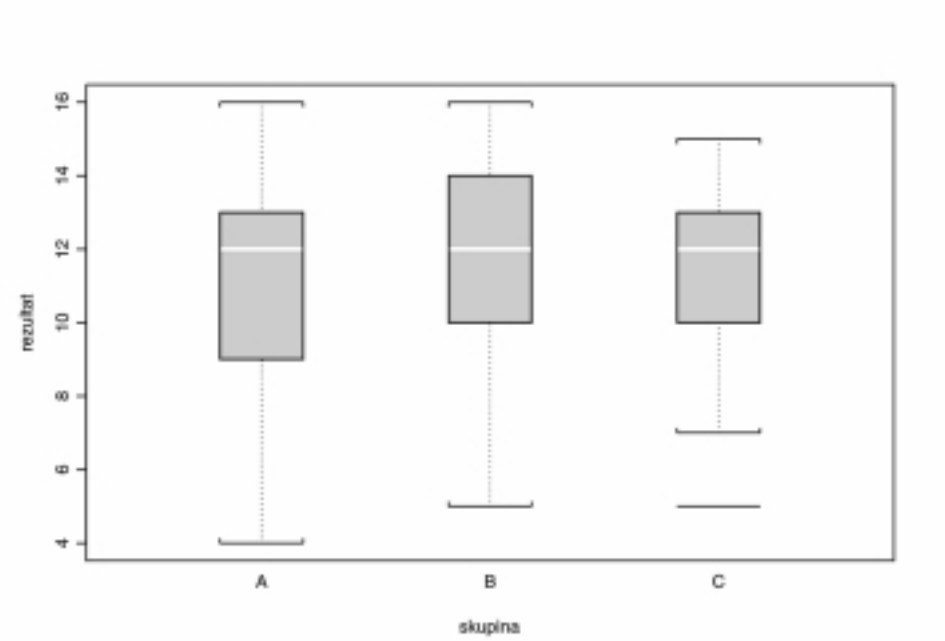
vir razpršenosti	vsota kvadratov	prostostne stopnje	povprečja kvadratov	F	p	prispevek v odstotkih
med skup.	1.2221	2	0.611040	0.1405621	0.8690921	0.36
v skupinah	334.7279	77	4.347116			99.64
skupaj	335.95	79				

Ker je izračunano razmerje povprečij kvadratov odklonov med skupinami in v skupinah, označimo ga z F, manjše od ena, je hipoteza, da ni razlike v učinkih različnih obdelav, pravilna. To pomeni, da razpršenost zaradi pripadnosti skupini ni pomembna (signifikantna), saj k skupni razpršenosti podatkov prispeva le dobro tretjino odstotka. Pripadnost skupini A, B ali C torej nima učinka na dosežen rezultat uvodnega testa. Dobljene razlike v povprečnih vrednostih rezultatov niso večje od tistih, ki jih lahko dobimo naključno pri istem številu udeležencev (vse razlike lahko torej pripišemo naključnemu odstopanju podatkov).

Morda bi bilo tu zanimivo pogledati še nekatere podatke, ki smo jih zbrali s pomočjo vprašalnika. Večina študentov (skoraj 87%) ima že precej izkušenj s spletnimi brskalniki, vendar pa imajo zelo malo izkušenj s programskima jezikoma C in C++ (okoli 88% ima z njima le malo ali nič izkušenj). Prav vsi udeleženci pa so začetniki v programskem jeziku Java (naši testni učni domeni).

6.4.2 Analiza rezultatov zaključnega testa

Pri izvedbi poskusa smo pričakovali, da bo sistem, s pomočjo katerega so uporabniki pridobivali znanje, vplival tudi na dosežen rezultat pri zaključnem testu. To pomeni, da bi se morali rezultati, doseženi pri zaključnem testu, precej razlikovati po posameznih skupinah.



Slika 6.7
Rezultati zaključnega testa po skupinah

Podobno kot pri uvodnem testu smo tudi rezultate zaključnega testa najprej primerjali s pomočjo splošnih mer. Slika 6.7 prikazuje splošne značilnosti rezultatov v posameznih skupinah. Tudi tu je razpon doseženih točk precej velik, saj znaša v skupini A od 4 do 16 točk, v skupini B od 5 do 16 točk ter v skupini C od 5 do 15 točk, od skupno 17 možnih točk. Mediane so 12 in so enake v vseh skupinah. Povprečno število doseženih točk v skupini A je 11.22 (66% vseh možnih točk), nekoliko boljši pa sta skupina B z 11.81 (69%) in skupina C z 11.31 (67%). Ustrezni standardni odkloni od povprečja so 2.89 v skupini A, 2.63 v skupini B in 2.28 v skupini C. Opisano osnovno statistiko na rezultatih zaključnega testa povzemamo v tabeli 6.3.

Tabela 6.3 – Rezultati zaključnega testa po skupinah

skupina	število udeležencev	povprečje (od 17)	mediana	variacijski razmak	standardni odklon	std. povpr. napak
A	27	11.22	12	4 – 16	2.89	0.5556
B	27	11.81	12	5 – 16	2.63	0.5065
C	26	11.31	12	5 – 15	2.28	0.4464
skupaj	80	11.45	12	4 – 16	2.59	0.2901

Čeprav smo pričakovali precejšnje razlike v rezultatih med posameznimi skupinami, nam slika 6.7 razkriva ravno obratno – rezultati so med skupinami precej podobni. Analiza variance, katere rezultati so povzeti v tabeli 6.4, nam to tudi potrdi. Tudi tukaj dobimo vrednost razmerja povprečij kvadratov odklonov F manjšo od ena, kar potrjuje pravilnost hipoteze o enakih odzivih vseh obdelav.

Tabela 6.4 – Analiza variance rezultatov zaključnega testa po skupinah

vir razpršenosti	vsota kvadratov	prostostne stopnje	povprečja kvadratov	F	p	prispevek v odstotkih
med skup.	5.5208	2	2.760399	0.4038744	0.6691341	1.04
v skupinah	526.2792	77	6.834795			98.96
skupaj	531.8	79				

Razpršenost zaradi pripadnosti skupini prispeva okoli en odstotek skupne razpršenosti podatkov, zato ni pomembna. Pripadnost skupini A, B ali C torej nima učinka na dosežen rezultat zaključnega testa.

Seveda so nas dobljeni rezultati malce presenetili, saj smo pričakovali popolnoma drugačno sliko, katero so nakazovali tudi rezultati drugih raziskav [Mann 1999]. Podrobnejša analiza podatkov, pridobljenih v poizkusu, pa je pokazala, kje je prišlo do napake v razmišljanju in s tem do odstopanja rezultatov od pričakovanega. Različne rezultate v posameznih skupinah smo namreč pričakovali zaradi uporabe različnih učnih sistemov in s tem povezanih različnih načinov učenja. Predpostavljali smo, da uporaba A in B različic sistema že sama po sebi vodi tudi k uporabi prilagoditvenih zmožnosti sistema. Podrobnejši pregled dnevnikov, kjer so zapisani uporabnikovi koraki pri uporabi sistema, pa je pokazal, da v večini primerov sploh ni bilo razlik med uporabo sistemov različic A, B in C. Približno polovica uporabnikov različic A in B namreč ni izrabljala prilagodljivih zmožnosti sistema in je sistem

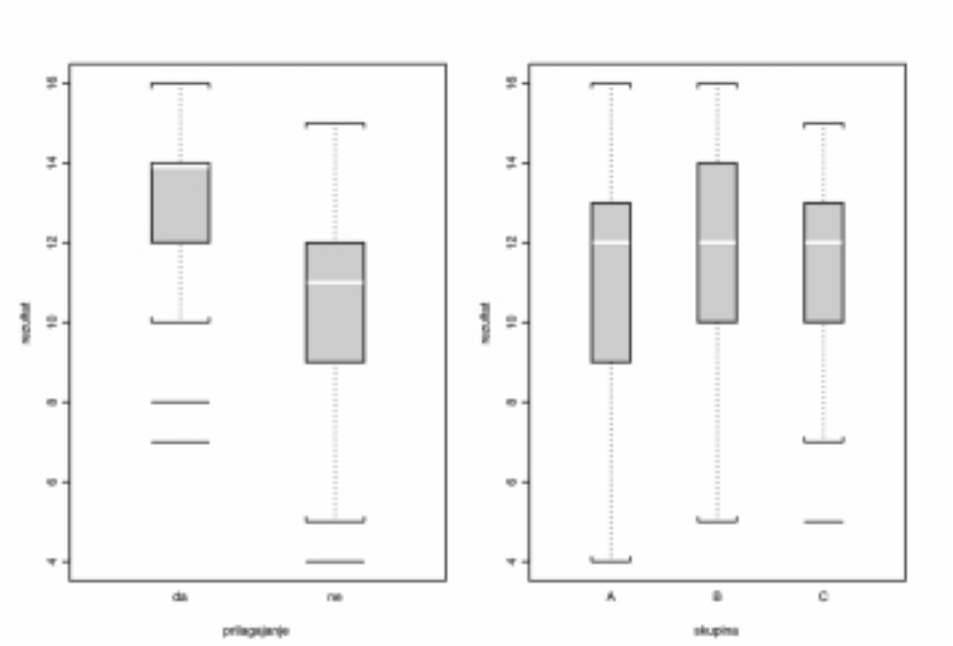
uporabljala povsem enako, kot če bi bil neprilagodljiv (naslednjo enoto so vedno izbirali neposredno iz kazala vsebine, predelovali pa so vse enote po vrsti od prve naprej). V takih primerih pa je lahko prilagodljiva podpora navigaciji prej moteča za uporabnika, saj mu ne pomaga pri pregledovanju materiala, kakor si ga je zamislil uporabnik (linearno od začetka do konca), zato uporabnika lahko tudi zmede.

Če pogledamo rezultate zaključnega testa z vidika uporabe prilagoditvenih zmožnosti sistema, dobimo povsem drugačno sliko. Povprečni vrednosti rezultatov se namreč kar precej razlikujeta, razlika (2.2 točke) je veliko večja kot pa razlike med skupinami (0.59, 0.50 in 0.09 točke). Osnovne statistične obdelave rezultatov zaključnega testa glede na uporabo prilagoditvenih zmožnosti sistema smo strnili v tabeli 6.5.

Tabela 6.5 – Rezultati zaključnega testa glede na uporabo prilagoditvenih zmožnosti sistema

prilagajanje	število udeležencev	povprečje (od 17)	mediana	variacijski razmak	standardni odklon	std. povpr. napak
ja	25	12.96	14	7 – 16	2.28	0.4564
ne	55	10.76	11	4 – 15	2.45	0.3303
skupaj	80	11.45	12	4 – 16	2.59	0.2901

Opisana razmerja prikazuje slika 6.8, kjer smo rezultate zaključnega testa prikazali ločeno, po uporabi prilagajanja in po skupinah.



Slika 6.8

Rezultati zaključnega testa po uporabi prilagoditvenih zmožnosti sistema in skupinah

Seveda pa je najbolj zanimivo vprašanje, ali so razlike, ki izhajajo iz povprečnih vrednosti dovolj pomembne, da jih lahko pripišemo obdelavam, v našem primeru uporabi določene različice sistema in uporabi njegovih prilagoditvenih zmožnosti. Odgovor lahko dobimo z analizo variance. Tokrat uporabimo dvosmerno analizo, ker imamo dve različni obdelavi: po skupinah in po uporabi prilagajanja. Rezultat dvosmerne analize variance povzema tabela 6.6.

Tabela 6.6 – Dvosmerna analiza variance rezultatov zaključnega testa po uporabi prilagoditvenih zmožnosti sistema in skupinah

vir razpršenosti	vsota kvadratov	prostostne stopnje	povprečja kvadratov	F	p	prispevek v odstotkih
prilagajanje	82.9127	1	82.91273	14.54251	0.0002792	15.59
skupina	21.0694	2	10.53470	1.84774	0.1646954	3.96
interakcija	0.2126	1	0.21256	0.03728	0.8474121	0.04
v skupinah	427.6053	75	5.70140			80.41
skupaj	531.8	79				

Iz tabeliranih vrednosti porazdelitve F [Hays 1994] dobimo, da je za zgornjih 5% točk ($\alpha = 0.05$), pri prostostnih stopnjah 75 (gledamo najbližjo večjo vrednost, to je 120) in 1,

vrednost F enaka 3.92, kaj je precej manj od izračunane vrednosti 14.54 (tabela 6.6). Podobno dobimo iz statističnih tabel vrednost za F pri prostostnih stopnjah 75 in 2, ki je enaka 3.07, kar pa presega izračunano vrednost 1.85 iz tabele 6.6. Tako v prvem primeru, ki določa vpliv uporabe prilagajanja, dobljena vrednost F močno presega kritično vrednost, zato hipotezo o enakih učinkih obdelav zavrnemo z verjetnostjo napake prve vrste 0.05. V drugem primeru, pri vplivu pripadnosti skupini, pa ničelne hipoteze o enakih učinkih obdelav ne moremo zavrniti. Podobno velja tudi za interakcijo obeh obdelav ($F < 0$), ki skoraj nič ne vpliva na celotno razpršenost podatkov. Tako smo s pomočjo analize variance pokazali, da uporaba prilagajanja vpliva na dosežen rezultat zaključnega testa.

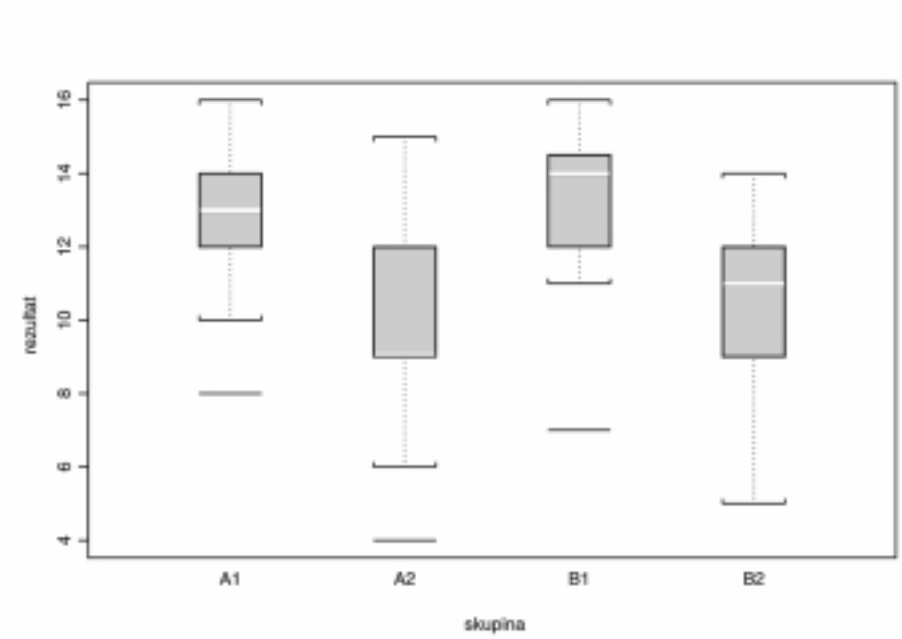
Ker je bil naš namen oceniti uporabnost prilagodljivega sistema, oziroma ugotoviti razlike med učenjem s pomočjo prilagodljivega sistema v primerjavi z neprilagodljivim sistemom, smo se odločili, da dobljene rezultate skupin A in B razdelimo na dva dela: v prvem so rezultati uporabnikov, ki so izkoriščali tudi prilagoditvene zmožnosti sistema, v drugem pa rezultati uporabnikov, ki teh zmožnosti niso izkoriščali. Tako dobljene skupine smo označili z A1 (uporabljali prilagajanje) in A2 (niso uporabljali prilagajanja) oziroma B1 in B2. Rezultate uvodnega testa in zaključnega testa smo ponovno analizirali z upoštevanjem štirih različnih skupin (A1, A2, B1 in B2). Dobljene vrednosti analize so opisane v naslednjem razdelku.

6.4.3 Ponovna analiza rezultatov zaključnega testa

Tokrat smo analizirali rezultate zaključnega testa glede na uporabo prilagoditvenih zmožnosti sistema. Uporabniki prilagodljivih sistemov, ki so izkoriščali prilagoditvene zmožnosti sistema, so zajeti v skupinah A1 in B1, ostali pa v skupinah A2 in B2, odvisno od različice sistema, ki so jo uporabljali pri poizkusu. Uporabnikov različice C tu nismo upoštevali, saj je bila to le kontrolna skupina z neprilagodljivim sistemom.

Analiza variance pri rezultatih uvodnega testa tudi v tem primeru ni pokazala bistvenih razlik med posameznimi skupinami. Nasprotno pa je analiza variance pri rezultatih zaključnega testa nakazala pomembne razlike med skupinami uporabnikov A1, A2, B1 in B2.

Že grafični prikaz povprečnih vrednosti posameznih štirih skupin (slika 6.9) pokaže razlike med povprečji skupin A1 in B1, ki so višja kot pri skupinah A2 in B2 (prav tako so večje tudi mediane). Standardni odkloni od povprečja so podobni v vseh štirih skupinah. Osnovne statistične mere centralne tendence in razpršenosti nad rezultati zaključnega testa v vseh štirih skupinah so strnjene v tabeli 6.7.



Slika 6.9

Rezultati zaključnega testa po skupinah A1, A2, B1 in B2

Tabela 6.7 – Rezultati zaključnega testa po skupinah A1, A2, B1 in B2

skupina	število udeležencev	povprečje (od 17)	mediana	variacijski razmak	standardni odklon	std. povpr. napak
A1	13	12.69	13	8 – 16	2.14	0.5925
A2	14	9.86	9	4 – 15	2.88	0.7693
B1	12	13.25	14	7 – 16	2.49	0.7191
B2	15	10.67	11	5 – 14	2.19	0.5662
skupaj	54	11.52	12	4 – 16	2.75	0.3745

Pri analizi variance po skupinah A1, A2, B1 in B2 (tabela 6.8) dobimo F, to je razmerje povprečij kvadratov odklonov med skupinami in v skupinah, enak 5.78, kritična vrednost pri $\alpha = 0.05$ ter prostostnih stopnjah 50 (gledamo 60) in 3 pa je enaka 2.76. Ker izračunana vrednost F presega kritično vrednost, lahko z verjetnostjo 0.05 napake prve vrste zavrnemo hipotezo o enakosti povprečij populacij. To pomeni, da med navedenimi štirimi skupinami obstajajo pomembne razlike in da ima skupina (različica sistema in uporaba njegovih prilagoditvenih zmožnosti) določen učinek na rezultat zaključnega testa.

Tabela 6.8 – Analiza variance po skupinah A1, A2, B1 in B2

vir razpršenosti	vsota kvadratov	prostostne stopnje	povprečja kvadratov	F	p	prispevek v odstotkih
med skup.	103.4146	3	34.47154	5.782519	0.0017860	25.76
v skupinah	298.0668	50	5.96134			74.24
skupaj	401.4814	53				

S pomočjo analize variance smo torej potrdili učinke skupin, sedaj pa moramo poiskati tudi vir teh učinkov ter razložiti njihov pomen. Zaenkrat vemo le to, da med navedenimi štirimi skupinami obstajajo pomembne razlike, ne vemo pa, kje te razlike so in kako velike so.

Zato podatke podrobneje analiziramo še s primerjavo povprečij. Uporabili bomo post–hoc primerjavo vseh možnih parov povprečij. Ker imamo le štiri skupine, nam primerjava vseh možnih parov povprečij posameznih skupin prinese šest primerjav. Te so zajete v tabeli 6.9, kjer je za vsak par primerjanih skupin določena ocena vrednosti primerjave, ocena standardne napake primerjave ter meji (spodnja in zgornja) intervala zaupanja. Tudi tu smo predpostavili verjetnost napake prve vrste $\alpha = 0.05$ na posamezno primerjavo, torej je dobljeni interval zaupanja 95% za dano linearno kombinacijo skupin.

Tabela 6.9 – Primerjave parov povprečij skupin A1, A2, B1, B2

primerjava skupin	ocenjene vrednosti	standardna napaka	spodnja meja	zgornja meja	0 je izven intervala
A1 – A2	2.840	0.940	0.946	4.72	ja
A1 – B1	– 0.558	0.977	– 2.520	1.41	ne
A1 – B2	2.030	0.925	0.167	3.88	ja
A2 – B1	– 3.390	0.961	– 5.320	– 1.46	ja
A2 – B2	– 0.810	0.907	– 2.630	1.01	ne
B1 – B2	2.580	0.946	0.684	4.48	ja

Če testiramo hipotezo, da sta povprečji skupin enaki (vrednost primerjave je enaka nič), mora interval zaupanja zajemati tudi našo hipotetično vrednost. V primeru, ko interval zaupanja ne pokriva vrednosti nič, lahko hipotezo o enakosti povprečij zavrnemo pod stopnjo α, ustrezna primerjava pa kaže signifikantno razliko na stopnji α med povprečji obeh skupin.

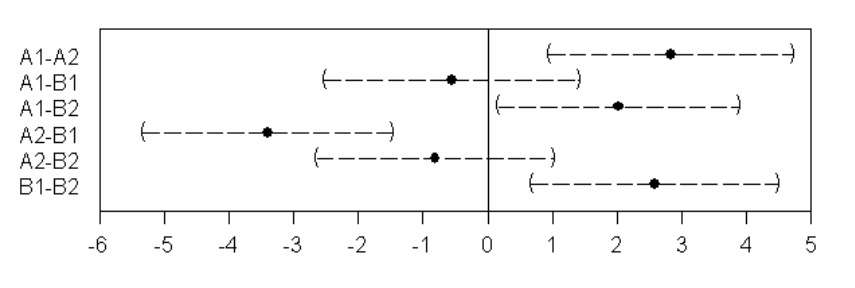
Interval zaupanja določajo ocenjena vrednost primerjave skupin, ocenjena standardna napaka te primerjave in izračunana kritična točka [Hays 1994]:

(interval) = (ocenjena vrednost) ± (kritična točka) · (standardna napaka).

Kritično točko izračunamo v odvisnosti od uporabljene metode in stopnje zaupanja. Tu smo uporabili Fisherjevo LSD metodo in stopnjo zaupanja 0.95 ($\alpha = 0.05$). Tako dobljena kritična točka je enaka 2.0086.

Primerjave skupin, pri katerih pade število 0 izven intervala zaupanja, so v tabeli 6.9 označene v zadnjem stolpcu. Zanje velja, da je razlika v povprečjih med skupinami pomembna nad stopnjo $\alpha = 0.05$.

Podatke iz tabele 6.9 grafično prikazuje slika 6.10, kjer so s piko označene ocenjene vrednosti primerjave (razlike povprečij dveh skupin), okoli njih je s črtkano črto zarisan interval zaupanja, polna navpična črta pa označuje ničlo.



Slika 6.10

Primerjava parov povprečij skupin ob individualnih 95% intervalih zaupanja
z uporabo Fisherjeve LSD metode na rezultatih zaključnega testa
po skupinah A1, A2, B1 in B2

Kakšni so torej učinki posameznih skupin na dosežen rezultat? Odgovore najlažje razberemo kar iz slike 6.10. Razlike med skupinami A1 in A2, A1 in B2, B1 in A2 ter B1 in B2 so statistično pomembne. Med skupinami A1 in B1 ter A2 in B2 pa statistično ni razlik.

To pomeni, da uporaba prilagajanja vodi v znatno boljše rezultate v primerjavi z neuporabo prilagajanja (v povprečju je razlika 2.71 točke ali 16% vseh možnih točk), ne glede na uporabljeno različico sistema. Pri uporabi (oziroma neuporabi) prilagajanja med različicama A in B ni razlik.

6.5 Ugotovitve in zaključki

Rezultati poizkusa so pokazali, da obstaja pomembna razlika v rezultatih zaključnega testa med uporabniki, ki so izkoriščali prilagoditvene možnosti učnega sistema, in tistimi, ki so sistem uporabljali kot navaden neprilagodljiv sistem. Slednji so v povprečju na testu dosegali za petino slabše rezultate. Med skupino uporabnikov, kateri je sistem omogočal tudi barvno anotacijo povezav, in skupino s sistemom brez prilagodljive anotacije povezav ni bilo opaznejših razlik.

V našem eksperimentu smo kot glavni način prilagajanja uporabljali tehnologijo prilagodljivega vstavljanja povezav. Ta se je izkazala za učinkovito pomoč navigaciji, čeprav dobljeni rezultati niso tako spodbudni kot rezultati nekaterih drugih raziskav. V svoji raziskavi o vplivu tehnologije prilagodljivega skrivanja povezav na rezultat testa, ki jo je izvedel Mann [Mann 1999], je razlika med eksperimentalno skupino, ki je uporabljala skrivanje povezav, in kontrolno skupino znašala v povprečju kar 27%. Izsledkov drugih raziskav, ki bi povezovale uporabo tehnologij prilagodljive podpore navigaciji z rezultati kontrolnega testa, pa žal zaenkrat še ni na voljo.

Tabela 6.10 – Povzetek odgovorov iz vprašalnika (po skupinah)

Sistem *ALICE*	**A**	**B**	**C**	**skupaj**
za privajanje sistemu so porabili manj kot 10 minut (% vseh)	96	81	85	87
kako radi so delali s sistemom (ocena 1 do 5)	4.3	3.8	3.5	3.9
sistem bi uporabljali še naprej (% vseh)	96	92	65	85
up. vmesnik, navigacija in celoten sistem so dovolj enostavni (% vseh)	100	99	100	99
ustreza jim samostojna izbira lekcij (% vseh)	77	73	62	71
ustreza jim določanje zaporedja lekcij s strani sistema (% vseh)	65	58	73	65
splošna ocena izgleda sistema (ocena 1 do 5)	4.3	3.9	4.1	4.1
splošna ocena uporabnosti sistema (ocena 1 do 5)	4.4	4.0	4.2	4.2
ocena uporabnosti gumba *Test* (ocena 1 do 5)	4.7	4.5	4.7	4.6
ocena uporabnosti barvne anotacije povezav (ocena 1 do 5)	4.4	–	–	4.4
ocena uporabnosti povezav na sorodne enote (ocena 1 do 5)	3.5	3.6	–	3.6
uporabljali so povezave na sorodne enote (% vseh)	58	54	–	56

Poleg samih meritev rezultatov testov pa nas je ob izvedbi poizkusa zanimalo tudi splošno mnenje uporabnikov o sistemu in njegovi uporabnosti. Te podatke smo zbrali s pomočjo

vprašalnika, v katerem so uporabniki podali svoje subjektivno mnenje o sistemu, s katerim so delali. V tabeli 6.10 smo povzeli nekaj odgovorov na glavna vprašanja.

Iz podanih odgovorov je razvidno, da so uporabniški vmesnik, navigacija v sistemu in sistem na splošno zadosti enostavni za uporabo. To potrjuje tudi dejstvo, da je več kot polovica uporabnikov (55%) porabila manj kot pet minut za privajanje na sistem (87% pa manj kot deset minut). Večini uporabnikov je bilo učenje s pomočjo sistema zanimivo in bi ga uporabljali tudi vnaprej (85%). Pri tem gre poudariti, da je odstotek tistih, ki bi sistem radi uporabljali tudi v bodoče, v skupinah A in B neprimerno večji (96% in 92%) kot v skupini C (65%). Iz tega lahko sklepamo, da imajo uporabniki raje (prilagodljivi) različici A in B. Svoje veselje pri delu s sistemom so v povprečju ocenili s 3.9 (tudi tu je bila ocena skupine C nekoliko nižja).

Splošni oceni izgleda (4.1) in uporabnosti (4.2) sistema nakazujeta, da je prototip sistema zasnovan dobro, kljub temu pa bo z nadaljnjim delom potrebno še marsikaj izboljšati. Vprašalnik je zajemal tudi pohvale in kritike ter morebitne predloge za izboljšanje sistema, katere bomo v prihodnje z veseljem upoštevali.

Zanimiv je tudi odstotek tistih uporabnikov, ki jim ustreza samostojna izbira lekcij, in tistih, ki jim ustreza vodenje sistema. Prvih je v skupinah A in B 75%, v skupini C pa le 62%. Nasprotno pa je drugih več v skupini C (73%), v skupinah A in B pa le slabih 62%. To lahko povežemo z dejstvom, da neprilagodljiv sistem, ki ne nudi toliko podpore samostojni navigaciji po učnem materialu, bolj podpira pasivno udeležbo uporabnika pri izbiri lekcij. Zato je učenje z neprilagodljivim sistemom lažje, kadar je vodeno s strani sistema. Podobno pa prilagodljivi sistemi spodbujajo uporabnika k aktivnejšemu raziskovanju učnega materiala, saj mu pri tem nudijo tudi več podpore. V tem primeru je enostavneje in tudi zabavneje samostojno odločati o izbiri lekcij, saj ima uporabnik več kontrole pri učenju in večji pregled nad materialom.

Večina uporabnikov sistema je bila navdušena nad možnostjo sprotnega preverjanja znanja, kar se odraža v oceni gumba *Test*, ki znaša povprečno kar 4.6. Tudi uporabnost barvne anotacije povezav je ocenjena zelo dobro (4.4), čeprav rezultati zaključnega testa ne kažejo posebnega vpliva barvne anotacije na kvaliteto učenja. Seveda gre tu lahko za bolj posredne učinke, ki jih ne moremo določiti v tako kratkem času, saj barvna anotacija nedvomno pomaga pri orientaciji uporabnika v sistemu in nudi uporabniku dodatne informacije.

Stopnja uporabe povezav na sorodne enote (56%) se približno ujema s tisto, ki smo jo razbrali iz dnevnikov uporabe sistemov. Tu je sicer nekoliko višja, a razlika lahko izhaja iz dejstva, da so seznami povezav na sorodne enote lahko služili tudi samo za orientacijo, brez

neposredne uporabe samih povezav. Ocena uporabnosti povezav na sorodne enote je relativno nizka (3.6), kar pa je logična posledica nizkega odstotka njihove uporabe (študentom, ki niso uporabljali povezav na sorodne enote, so se te povezave zdele manj uporabne, zato so njihovo uporabo tudi slabše ocenili).

S pomočjo vprašalnika dobljeni rezultati so vezani konkretno na našo izvedbo prilagodljivega spletnega sistema in jih zato ni smiselno primerjati z rezultati podobnih raziskav (kot je na primer [Mitrovic 2000]), čeprav so izsledki dokaj podobni. Sicer pa so številke iz tabele 6.10 že same po sebi precej zgovorne in kažejo na uspešno predstavitev sistema.

Iz navedenega lahko zaključimo, da je tehnologija prilagodljivega vstavljanja povezav koristna in pozitivno učinkuje na uporabnikovo pridobivanje znanja. Tudi kombinacija z barvno anotacijo povezav je bila med uporabniki dobro sprejeta, čeprav ni neposredno vplivala na rezultate zaključnega testa. Naše ugotovitve potrjujejo tudi izsledke drugih raziskav o uporabnosti prilagoditvenih tehnologij. Te kažejo, da skrivanje povezav izboljša razumevanje vsebine in poveča učinek učenja [Mann 1999], anotacija pa tudi izboljša razumevanje, a le tistim, ki so jo pripravljeni sprejeti [Brusilovsky 1998b].

V poizkusu smo merili rezultate po manj kot dvournem delu s sistemom. To je zelo malo časa za popolno prilagoditev in udomačenost v sistemu. Oceniti bi bilo potrebno tudi delo na daljši rok, kjer se uporabnik bolj navadi na sistem in začne izkoriščati več njegovih zmožnosti. Seveda bi bila zanimiva primerjava v tej raziskavi dobljenih rezultatov z rezultati po daljšem delu s sistemom.

7. Združevanje informacij modelov uporabnikov

V tem poglavju bomo modeliranje uporabnika v prilagodljivem sistemu nadgradili z možnostmi združevanja in povezovanja modelov. Tak pristop uvaja neke vrste komunikacijo med (so)uporabniki sistema oziroma le med njihovimi modeli znanja, ki smo jo uporabili za posredno primerjanje in sodelovanje med posameznimi uporabniki.

Primerjava modelov uporabnikov omogoča nekoliko drugačen pogled na dosežen nivo znanja posameznika, saj poznavanja konceptov ne meri le absolutno, temveč tudi relativno glede na ostale uporabnike sistema.

Po drugi strani pa smo možnost sodelovanja med uporabniki izkoristili za neke vrste skupinsko učenje. To omogoča združevanje uporabnikov v skupino, katere cilj je osvojiti celotno znanje učne domene. Pri tem vsak član skupine predela le njemu določen del domene tečaja, vsi skupaj pa kot skupina pokrivajo celotno vsebino tečaja.

7.1 Primerjanje modelov uporabnikov

V nekem časovnem obdobju (kot je na primer en semester) obiskuje spletni tečaj več uporabnikov sočasno. Vsakemu od njih pripada en model uporabnika, kjer se beleži njegovo napredovanje v tečaju oziroma osvojeno znanje domene. Zato imamo za isti tečaj celo množico različnih modelov uporabnikov, ki pa odražajo določene skupne značilnosti uporabnikov. Te se kažejo predvsem v ocenah znanja in številu predelanih učnih enot. S podrobnejšo analizo vseh modelov bi lahko ugotovili, kako uporabniki napredujejo v tečaju, kje so njihove šibke točke, na katerih je potrebno izboljšati in utrditi znanje, katere enote oziroma katera snov, vprašanje ali koncept prinaša največ težav, kateri deli tečaja so enostavni za večino uporabnikov in podobno.

V ta namen vpeljemo pojem *globalnega modela* uporabnikov, ki združuje informacije o znanju vseh uporabnikov, ki obiskujejo določen tečaj. Iz njega lahko izpeljemo tudi model povprečnega uporabnika. Modeli posameznih uporabnikov so zapisani v datotekah na strežniku. Podobno se tudi globalni model uporabnikov hrani na strežniku, le da se tam tudi vzdržuje in sproti posodablja, to je ob vsaki spremembi vrednosti kateregakoli od posameznih modelov

uporabnikov (slednji se sicer računajo in posodabljajo na strani odjemalca in se na strežniku zabeležijo le ob koncu seje).

S takim primerjanjem modelov uporabnikov lahko pridobimo nove informacije, ki niso zanimive le za pripravljavce in sestavljavce tečaja (učitelje), temveč tudi za uporabnike (študente). Pripravljavcem tečaja lahko globalni model nudi smernice za nadaljnje spremembe in izboljšave vsebine tečaja. Iz njega lahko razberejo, kateri deli snovi povzročajo težave večini uporabnikov (morda je preveč kompleksna ali neprimerno predstavljena), ter ustrezno ukrepajo s spremembo ali dopolnitvijo te snovi (podrobnejša razlaga ali več primerov). Prepoznajo lahko tudi dele snovi, ki je lažje razumljiva vsem uporabnikom, in to ustrezno upoštevajo pri sestavi novih učnih enot ali vprašanj za preverjanje znanja. Podobno pa so informacije iz globalnega modela koristne tudi za uporabnike, ki se na ta način lahko primerjajo z ostalimi uporabniki sistema. Svoje znanje lahko vzporedijo z znanjem najboljšega ali z znanjem povprečnega uporabnika ter določijo, kje se nahajajo (po doseženih rezultatih pri odgovorih na vprašanja za preverjanje znanja, po številu osvojenih konceptov ali po porabljenem času) med vsemi ostalimi uporabniki. Poleg tega lahko sistem vsakemu uporabniku izpostavi učne enote oz. koncepte, ki jih v primerjavi z drugimi uporabniki (še) ni dovolj temeljito predelal ali primerno osvojil.

Pri sestavi globalnega modela pridemo do problema, kako združiti posamezne modele uporabnikov in poznavanje katerih konceptov upoštevati pri njegovi izgradnji. Seveda moramo upoštevati vse modele uporabnikov, vendar so v njih shranjeni podatki lahko zelo različni. Ker uporabniki sami določajo vrstni red izbire učnih enot in se učijo s poljubnim tempom, pri primerjavi ne moremo upoštevati vseh konceptov, saj vsi modeli niso enaki v tem, katere koncepte je uporabnik že predelal in osvojil. Zato bi morali v primerjavi upoštevati le tiste koncepte, ki jih je posamezen uporabnik že obravnaval in se jih učil, ali pa veljajo za precej naučene (iz predtesta ali preko razširjanja vrednosti znanja). Prvi so najbolje prepoznavni po tem, da je uporabnik že odgovarjal na testna vprašanja v povezavi z njimi. Zanima nas seveda samo stopnja naučenosti teh konceptov, torej vrednost pripadnostne funkcije množici naučenih konceptov C_L, ki smo jo označili z μ_L. Pogoj, da je vrednost μ_L večja od nič, ne bi zadostoval, ker se ta vrednost lahko poveča tudi na podlagi razširjanja vrednosti znanja, a ne doseže stopnje precej naučenega koncepta. Drugi pomembni vrednosti, ki ju mora hraniti globalni model, pa sta skupno število predelanih oziroma naučenih konceptov in za to porabljen čas. Ti dve omogočata realnejšo primerjavo uporabnikov, saj ni vse v kvaliteti učenja, včasih je pomembna tudi kvantiteta učenja.

Znanje uporabnika je v modelu predstavljeno z mehkimi množicami. Za vsak koncept domene so podane stopnje pripadnosti trem mehkim množicam: nepoznanih, poznanih in naučenih konceptov. Tako za *j*–tega uporabnika, ki ga lahko označimo tudi uporabnik *j*, zapišemo njegovo znanje koncepta c_i kot:

znanje koncepta $(c_{i,j}) = (\mu_{iU,j}, \mu_{iK,j}, \mu_{iL,j})$.

Sedaj nas zanima le stopnja pripadnosti množici naučenih konceptov, to je $\mu_{iL,j}$, predvsem njena največja vrednost in njena povprečna vrednost preko vseh uporabnikov sistema. Definirajmo trdo množico T_j, katere elementi so koncepti domene, pri katerih je *j*–ti uporabnik že reševal test:

$$T_j = \{c_i \in C: \text{ uporabnik } j \text{ reševal test za } c_i\}.$$

Potem lahko z U_i označimo množico vseh uporabnikov, ki so reševali test pri konceptu c_i ali pa ta koncept že velja za precej naučenega:

$$U_i = \{j : c_i \in T_j \text{ ali } c_i \in C_{L\alpha,j}\}.$$

Moč množice U_i naj bo enaka $|U_i| = m_i$, torej je m_i število uporabnikov, pri katerih je koncept c_i že precej naučen ali pa so že reševali test pri tem konceptu. Potem lahko iz stopenj pripadnosti množicam $C_{L,j}$ izračunamo za koncept c_i povprečno vrednost stopnje pripadnosti množici naučenih konceptov preko vseh uporabnikov:

$$\overline{\mu}_{iL} = \frac{1}{m_i} \sum_{j \in U_i} \mu_{iL,j}$$

in največjo vrednost preko vseh uporabnikov:

$$\mu_{iL}^{\max} = \max_{j \in U_i}(\mu_{iL,j}),$$

kjer je *j* uporabnik iz množice U_i.

Izračunana vrednost $\mu_{iL}^{\max}$ naj bi bila enaka 1, saj to pomeni, da je vsaj en uporabnik popolnoma osvojil dani koncept c_i. Če je vrednost $\mu_{iL}^{\max}$ manjša od 1, je lahko to indikator, da je pri tem konceptu nekaj narobe, saj ga nobeden od uporabnikov ni uspel popolnoma osvojiti. Morda je snov slabo podana ali pa so vprašanja neprimerna in pretežka glede na vsebino lekcije. Podobno je tudi nizko povprečje $\overline{\mu}_{iL}$ znak, da bi bilo potrebno preveriti stanje pri ustrezni enoti.

Kadar je $\mu_{iL,j}$ enak 1, to pomeni, da velja koncept c_i pri *j*–tem uporabniku za popolnoma naučenega v absolutnem smislu, torej je uporabnikovo znanje o tem konceptu enako znanju eksperta. Vendar pa ne moremo vedno stremeti za takim absolutno popolnim znanjem, saj od povprečnih uporabnikov učnega sistema (študentov, učencev) ne moremo niti pričakovati niti zahtevati, da vsi dosežejo to stopnjo pri vseh konceptih. Zato je bolj realno primerjava dosežene stopnje znanja z drugimi uporabniki sistema, torej s povprečno stopnjo znanja vseh uporabnikov.

Pri pregledu in primerjavi znanja posameznega uporabnika je zanimiva tudi stopnja nedoločenosti pri opisu njegovega znanja. To lahko izrazimo z indeksom mehkosti mehke množice vseh naučenih konceptov. Eden od možnih indeksov, ki jih lahko uporabimo v ta namen, je razlika razdalj med trdo množico vseh konceptov *C* in mehko množico naučenih konceptov $C_{L,j}$, kjer za razdalje upoštevamo Hammingove mehke metrične razdalje med množico in njenim komplementom. Indeks mehkosti zapišemo z enačbo:

$$f_1(C_{L,j}) = |C| - \sum_{c_i \in C} \left| \mu_{iL,j} - \mu_{iN,j} \right|,$$

kjer je $\mu_{iN,j}$ vrednost pripadnostne funkcije množici $C_{N,j}$, ki je komplement množice naučenih konceptov $C_{L,j}$, $|C|$ pa je število vseh konceptov domene. Če izrazimo vrednost pripadnostne funkcije komplementa množice s pripadnostno funkcijo množici, dobimo:

$$f_1(C_{L,j}) = |C| - \sum_{c_i \in C} \left| 2\mu_{iL,j} - 1 \right|.$$

Če je množica $C_{L,j}$ trda množica, je indeks mehkosti f_1 enak 0. Če pa je $C_{L,j}$ popolnoma mehka množica (vsi elementi imajo pripadnost množici enako 0.5), je indeks enak moči množice *C* ($f_1 = |C|$), to je številu vseh konceptov domene. Torej velja, da večji ko je indeks, manjša je razlika med množico in njenim komplementom ter s tem večja nezanesljivost pri oceni znanja.

7.2 Sodelovanje modelov uporabnikov

Povsem drugačen pristop k združevanju informacij modelov pa prinaša način sodelovanja uporabnikov, ki ga bomo opisali v tem razdelku. Tu več uporabnikov sodeluje tako, da skupaj (kot skupina) predelajo določeno snov tečaja, pri tem pa se vsak posameznik nauči le njemu določen del te snovi.

Tak način skupinskega učenja je zanimiv predvsem v podjetjih, kjer pričakujejo (in zahtevajo) od svojih zaposlenih poznavanje določenega področja. Pri tem ni toliko važno, kaj zna kdo od posameznikov, temveč da celoten tim kot skupina pokriva želeno področje, vsak posameznik pa podrobneje obvlada le del tega področja (in sploh ni potrebno, da vsi vedo vse). Seveda imajo vsi posamezniki neko splošno, temeljno znanje tega področja, specialno znanje pa je razdeljeno. Taka delitev je še posebej pomembna v današnji družbi, ki je preobložena z različnimi informacijami in kjer neprestano uvajanje vedno novejših tehnologij zahteva veliko prožnosti in novega znanja.

Pri sodelovanju modelov uporabnikov gre torej za to, da na čim lažji način in zato tudi v čim krajšem času skupina uporabnikov predela in se nauči celotno domeno tečaja. Seveda je potrebno pri tem doseči tudi čim manjšo redundanco znanja v skupini. Skupno (prekrivajoče se) znanje uporabnikov mora biti zatorej kar najmanjše, znanje vseh posameznikov skupaj pa mora pokrivati celotno domeno.

Imamo torej skupino uporabnikov, ki morajo skupaj predelati učno snov tečaja. Takih skupin je sicer lahko tudi več, a se bomo sedaj omejili na eno samo. Vsi uporabniki v skupini uporabljajo sistem individualno in neodvisno od drugih uporabnikov iste skupine. Razlika je le ta, da ima vsak uporabnik nalogo predelati le nek njemu določen del učne domene. Ta del se dinamično izbere v odvisnosti od uporabnikovega znanja, njegovih preferenc ter znanja drugih uporabnikov iz skupine. V ta namen uvedemo *skupinski model* uporabnikov, ki je model celotne skupine (ne smemo ga zamenjevati z globalnim modelom iz predhodnega razdelka).

Pri učnih enotah, ki so določene posameznemu uporabniku, prilagodljiv sistem deluje normalno, vključno z barvno anotacijo in predlaganjem povezav na sorodne enote. Pri vseh ostalih učnih enotah, ki ne spadajo med izbrane enote danega uporabnika, pa se sistem obnaša nekoliko drugače. V kazalu vsebine in indeksu povezave na te enote sicer obstajajo, a so označene sivo, kar pomeni, da za uporabnika niso toliko zanimive. Povezave na sorodne učne enote ne vključujejo (in ne prikazujejo) teh enot, gumb *Naprej* pa tudi ne. Vse take enote, ki niso med izbranimi enotami uporabnika, so uporabniku sicer še vedno na voljo, a so posebej označene.

Tu je eden glavnih problemov, kako vsakemu uporabniku izbrati učne enote, ki jih mora predelati. Pri tem se moramo držati naslednjih zahtev:

- Vse učne enote morajo biti čimbolj enakomerno razporejene med uporabnike.
- Vsi uporabniki skupaj morajo pokrivati vse enote tečaja.

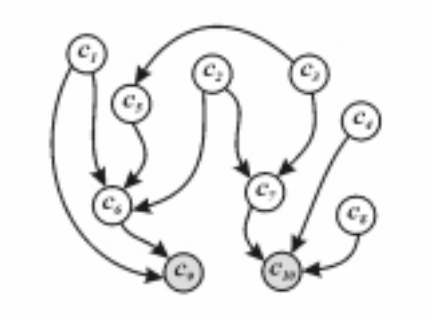
- Enot, ki so dodeljene več kot enemu uporabniku iz skupine, mora biti čimmanj (le toliko, kolikor je nujno potrebno zaradi predpogojnih relacij).
- Vsak uporabnik naj dobi tiste enote, ki so mu najbližje (katerih koncepte je najbolj poznal na predtestu, katerih snov se najraje uči in podobno).

Pri rešitvi tega problema izhajamo iz grafa domene. Za vse končne koncepte v grafu domene velja, da za poznavanje vseh domenskih konceptov, zaradi predpogojnih relacij med koncepti, zadostuje poznavanje le končnih konceptov domene. Če torej skupina obvlada vse končne enote (in s tem končne koncepte), obvlada tudi celotno domeno. Ponavadi je število vseh končnih konceptov veliko večje od števila uporabnikov v skupini (smiselno je sestaviti manjše skupine uporabnikov, tudi zaradi kasnejše lažje uporabe pridobljenega znanja; večje skupine lahko sestavimo le pri zelo velikih učnih domenah). Zato je naša naloga, da vse končne koncepte enakomerno razporedimo med uporabnike v skupini, pri tem pa seveda upoštevamo omejitve zgornjih štirih postavk.

Vsak končni koncept ima ponavadi tudi svoje predpogojne koncepte. Zato v grafu domene poiščemo vse podgrafe, ki zajemajo natanko en končni koncept in vse njegove nujne predpogojne koncepte, tako neposredne kot vse posredne do osnovnih konceptov. Število takih podgrafov je enako številu končnih konceptov domene. Če se mora uporabnik naučiti določen končni koncept, se mora prej naučiti tudi vse druge koncepte iz podgrafa, ki vsebuje ta končni koncept. Seveda se lahko posamezni podgrafi tudi delno prekrivajo (glej sliko 7.2).

Pri izbiri in dodeljevanju podgrafov uporabnikom iz skupine pazimo, da tiste podgrafe, ki imajo veliko skupnih konceptov, dodelimo istemu uporabniku. Pri tem upoštevamo tudi že pridobljeno znanje uporabnika in mu zato dodelimo podgraf s kar največ koncepti, ki jih že pozna (na primer po ocenah znanja iz predtesta). Hkrati pa moramo paziti, da je število konceptov, ki se jih morajo uporabniki naučiti in seveda še niso naučeni, dovolj enakomerno porazdeljeno med uporabniki.

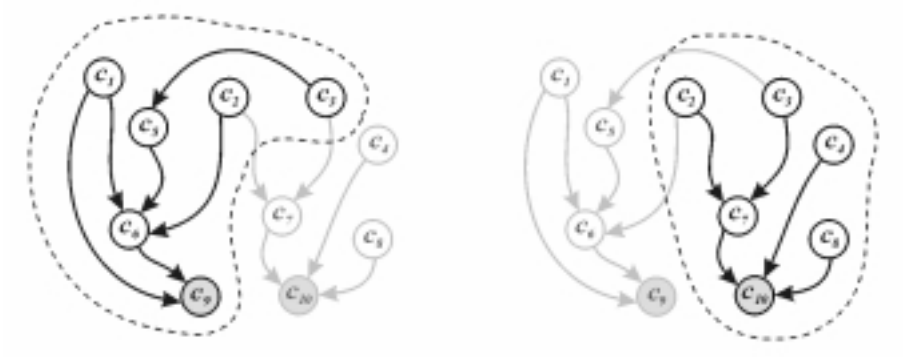
Sodelovanje in grupiranje uporabnikov si oglejmo še na poenostavljenem primeru. Imamo graf domene, ki ga prikazuje slika 7.1 in zajema deset konceptov, od katerih sta dva koncepta končna (na sliki sta označena sivo). Tu bomo koncepte označili kar s številkami od ena do deset. Celotna domena obsega množico konceptov $C = \{c_1, c_2, c_3, c_4, c_5, c_6, c_7, c_8, c_9, c_{10}\}$, kjer sta c_9 in c_{10} končna koncepta (slika 7.1). Mehke povezave v grafu domene, ki predstavljajo nujne predpogojne relacije med koncepti, nimajo pripisanih stopenj pripadnosti, ker le–te niso pomembne pri ponazoritvi našega primera.



Slika 7.1
Graf domene (za primer)

Skupino uporabnikov sestavljata dva uporabnika. Ker sta v domeni le dva končna koncepta, bo vsakemu uporabniku dodeljen natanko en končni koncept. Kateri bo to, je odvisno tudi od prikazanega znanja obeh uporabnikov na predtestu.

Graf domene razdelimo na dva podgrafa (na sliki 7.2), ki se delno prekrivata. Prvi podgraf zajema množico konceptov $C_1 = \{c_1, c_2, c_3, c_5, c_6, c_9\}$, drugi podgraf pa pokriva množico konceptov $C_2 = \{c_2, c_3, c_4, c_7, c_8, c_{10}\}$. Unija obeh množic je enaka množici vseh konceptov domene $C_1 \cup C_2 = C$, njun presek pa zajema koncepta c_2 in c_3, torej $C_1 \cap C_2 = \{c_2, c_3\}$.



Slika 7.2
Razdelitev grafa domene na dva podgrafa (za primer)

Recimo, da je prvi uporabnik na predtestu pokazal poznavanje konceptov c_1 in c_3, drugi uporabnik pa pozna le koncept c_3. Potem bi sicer lahko drugemu uporabniku dodelili katerokoli množico konceptov C_1 ali C_2, za prvega pa je primernejša množica C_1, iz katere že pozna dva koncepta. Zato opravimo razdelitev tako, da prvemu uporabniku dodelimo prvi podgraf ali množico konceptov C_1, drugemu pa drugega oziroma množico konceptov C_2.

Razdelitev podgrafov med uporabnike in napredovanje vsakega posameznika iz skupine nadzoruje vpeljan skupinski model uporabnikov. Ta model tudi spremlja vse uporabnike pri učenju in po potrebi intervenira z novo razdelitvijo učnih enot, torej ima bolj aktivno vlogo. Med delom lahko sistem po potrebi revidira dodeljene učne enote in jih ponovno razdeli med uporabnike, upoštevajoč zadnje prikazano znanje posameznih uporabnikov. Pri razdeljevanju učnih enot lahko vključimo in upoštevamo tudi uporabnikove preference in njegovo splošno zanimanje za določeno področje.

Iz navedenega primera vidimo, da lahko problem razdelitve grafa domene med uporabnike skupine prenesemo na problem razdelitve ustreznih podmnožic množice domenskih konceptov *C*. Množico konceptov, ki jo sestavljajo en končni koncept c_{fi} in vsi njegovi neposredni in posredni predpogojni koncepti, označimo s C_{fi}:

$$C_{fi} = \{c_{fi}\} \cup \{c_j \in C : c_j \prec^* c_{fi}\},$$

kjer operator $\prec^*$ pomeni (posredno) nujno predpogojno relacijo med koncepti. Potem lahko množico vseh domenskih konceptov *C* zapišemo kot unijo množic C_{fi}:

$$C = \bigcup_i C_{fi}.$$

Vsak od uporabnikov mora osvojiti vse koncepte ene ali več množic C_{fi}. Unijo teh množic, ki so dodeljene uporabniku *m*, označimo s C_m. To je množica vseh konceptov, ki jih mora predelati uporabnik *m*:

$$C_m = \bigcup_k C_{fk}.$$

Za določeno skupino uporabnikov moramo torej poiskati tako razdelitev množic C_{fi} med posamezne uporabnike, da je skupno število konceptov, ki se jih nauči več kot en uporabnik, čim manjše. Poiskati moramo množice C_m, pri katerih velja, da je presek vseh množic C_m minimalen, najbolje pa prazen ($\bigcap_m C_m = \text{minimalen}$), da je unija vseh množic enaka množici vseh konceptov domene $\bigcup_m C_m = C$ ter da so moči posameznih množic $|C_m|$ približno enake pri vseh uporabnikih (za vsak *m*).

8. Zaključek

V tem poglavju smo zbrali glavne ideje disertacije ter nekaj zaključnih misli. Podane so tudi smernice za nadaljnje delo in razvoj. Na koncu smo povzeli glavne prispevke tega dela.

8.1 Pregled narejenega in rezultatov

V okviru doktorske disertacije smo razvili in realizirali prilagodljiv hipermedijski sistem, ki je namenjen področju izobraževanja preko svetovnega spleta. Prilagodljivi hipermediji so zelo primerni za uporabo v izobraževanju, predvsem pri učenju na daljavo. Omogočajo tudi bolj individualno poučevanje, saj se znajo prilagajati vsakemu posameznemu uporabniku.

V tezi smo se posvetili predvsem problemu modeliranja uporabnikovega znanja z upoštevanjem nezanesljivosti pri opisu znanja. Model uporabnika smo zasnovali na mehkih množicah in mehkih pravilih. Znanje uporabnika je v modelu predstavljeno s pomočjo pripadnostnih funkcij trem mehkim množicam (nepoznanih, poznanih in naučenih konceptov). Iz prikazanega znanja koncepta pa lahko sklepamo tudi na znanje njegovih predpogojnih konceptov. To nam omogoča algoritem za razširjanje vrednosti znanja, ki temelji na množici mehkih pravil.

Tako zastavljen model uporabnika uporabimo skupaj z modelom domene pri dinamičnem določanju možnih povezav iz dane učne enote – tehnologiji prilagodljivega vstavljanja povezav. Za vsako prikazano enoto sistem sproti zgradi seznam povezav na sorodne enote znotraj domenskega hiperprostora. Tak seznam temelji na mehkih relacijah med koncepti domene (informacije o sorodnosti in vsebinski bližini enot) in na uporabnikovem poznavanju teh konceptov (primernost posameznih enot). Hipermedijski material učnih enot zato ne vsebuje neposrednih notranjih povezav, ampak so te posredno podane preko strukture grafa domene in so tudi prikazane ločeno.

Delovanje v okviru disertacije izdelanega prilagodljivega sistema smo tudi testirali v realnem okolju na populaciji 84 študentov. Na ta način smo lažje ovrednotili njegovo uporabnost, dopadljivost uporabnikom ter učinkovitost vgrajenih metod. Sistem se je izkazal za uspešnega, saj pripomore k večjemu učinku učenja, in enostavnega za uporabo.

Po naši raziskavi so se uporabniki, ki so izkoriščali prilagoditvene možnosti sistema, na zaključnem testu bolje odrezali kot ostali. Razlika ni zanemarljiva, saj je prva skupina dosegla za približno četrtino boljše rezultate od druge. Ta razlika gre predvsem na račun prilagodljivega vstavljanja povezav, saj med skupinama z barvno anotacijo povezav in brez nje ni bilo večjih razlik. Kljub temu pa je anotacija povezav za uporabnike zanimiva in uporabna, kot lahko razberemo iz odgovorov na vprašalnik, ki so ga izpolnili udeleženci. Iz njih tudi sledi, da sta obe prilagodljivi različici med uporabniki veliko bolj priljubljeni kot neprilagodljiva različica.

Na tem mestu naj opozorimo še na probleme, ki smo jih zasledili pri našem delu. Na prvega smo naleteli že pri izbiri metode za obravnavo nezanesljivosti znanja. Vsaka od opisanih metod ima namreč določene prednosti in slabosti, tako da njen izbor, čeravno je v veliki meri odvisen od samega problema, zahteva tudi določen kompromis. Pri mehkih množicah je tako problematična predvsem določitev ustreznih pripadnostnih funkcij, ki ponavadi temeljijo na subjektivni oceni in izkušnjah razvijalcev sistema. Tudi izbira primernih mehkih pravil ni enostavna in je večinoma zelo subjektivna. Vendar pa so mehka pravila blizu naravnemu opisu znanja in sklepanja, kar močno poenostavi njihovo sestavljanje in izboljša razumevanje.

Druga večja težava pa še vedno ostaja priprava tečajev. Tu ne gre le za same hipermedijske vsebine učnih enot (zanje lahko nenazadnje uporabimo že pripravljene strani), temveč tudi za njihovo hierarhično ureditev in pripravo modela učne domene. Slednje zajema izbor konceptov domene, vpeljavo predpogojne relacije med koncepti ter indeksiranje učnih enot po pripravljenih konceptih. Vse navedene naloge opravijo pripravljavci tečaja ročno, saj zahtevajo dobro poznavanje domene učenja. Zaenkrat zanje še ni videti boljših rešitev, a bi bile zanimive raziskave tudi v tej smeri.

8.2 Možnosti nadaljnjega dela

Nadaljnje raziskave bi lahko potekale v več smereh. Zajemajo ideje za izboljšanje sistema, ki so se porodile že med njegovim razvojem, ter nadaljnje raziskave s področja prilagodljivih hipermedijev, ki se neposredno navezujejo oziroma so nadaljevanje v tem delu začetih raziskav.

V vrsto manjših izboljšav predstavljenega prilagodljivega sistema, ki so usmerjene predvsem v dopolnitev njegovega delovanja, sodi tudi opustitev gumba *Oprav.*, ki ga nadomestimo z merjenjem časa ogleda posamezne lekcije ob danem predvidenem času za ogled T (kjer lahko T predstavimo kot mehko število s trikotno pripadnostno funkcijo). Če si

uporabnik stran ogleduje čas *t* in če je *t* dovolj blizu *T*, potem je ogled zelo donosen (poznavanje koncepta se močno poveča). Bolj ko *t* odstopa od idealne vrednosti *T*, manj je donosen (prekratek ali predolg čas ogleda lahko pomeni težave). Pri tem bi lahko upoštevali tudi tako imenovan čas brezdelnosti (angl. idle time), to je čas brez interakcije uporabnika s sistemom (uporabnik ne premika miške, ne pritiska na tipke ali kako drugače sodeluje s sistemom). Če je ta čas brezdelnosti predolg, lahko začasno zaustavimo merjenje časa ogleda strani, ker predpostavimo, da uporabnik ni prisoten (če že ne fizično, pa vsaj umsko).

Druga možna izboljšava bi bila vključitev ideje pozabljanja v model uporabnika. Za vsak koncept bi si lahko zapomnili, kdaj je bil nazadnje obravnavan in uporabljen. Če trenutni koncepti niso povezani z nekim konceptom, ki ga je uporabnik obravnaval že dolgo nazaj, se ta koncept “malo pozabi”, kar pomeni, da se vrednost pripadnostne funkcije množici naučenih konceptov zmanjša.

Prilagajanje sistema bi lahko izboljšali tudi z uvedbo prilagodljive predstavitve strani, saj so raziskave pokazale, da se ta tehnologija lepo dopolnjuje s podporo navigaciji. Vsako stran bi lahko razdelili na več sklopov, kot so povzetek, uvod, opis, več vrst primerov, simulacija in podobni. Odvisno od preferenc in znanja uporabnika bi lahko prikazali vse skupaj ali pa samo posamezne kategorije.

Model uporabnika bi lahko razširili tudi na druge informacije in se ne bi zanašali le na prikaz uporabnikovega znanja. Poleg tega bi lahko preizkusili še druge načine vpeljave nedoločenosti v model, saj tudi v teoriji mehkih množic obstaja še nekaj zanimivih načinov (na primer mehka števila).

Vzpostavitev baze testnih vprašanj bi popestrila predvsem delo uporabnikov. Obstoječi sistem pri izbrani lekciji prikazuje vedno ista vprašanja za preverjanje znanja. To bi lahko razširili na naključno izbiro vprašanj na določeno temo iz zgrajene baze vprašanj (izbiranje na podlagi modela uporabnika). Pri tem bi lahko izbrali tudi kompleksnejša vprašanja, ki bi poleg trenutnega koncepta, katerega znanje preverjamo, zajemala tudi že preverjeno znanje (naučeni koncepti). Prilagodljivi izbiri vprašanj bi lahko dodali tudi prilagodljive nasvete v primeru napačnega odgovora.

Med nekoliko zahtevnejše naloge pa spada razvoj razširitve sistema oziroma njegova dopolnitev v orodje za pripravo tečajev (angl. authoring tool), s katerim bi lahko uspešno delali tudi tisti, ki nimajo izkušenj v programiranju ali poznavanju modeliranja uporabnika (na primer učitelji z drugih področij). Tako orodje bi uporabljali kot pripomoček pri sestavi tečajev, za

opis učnega materiala in njegovo integracijo s prilagodljivim sistemom. Pri tem bi bila zelo zanimiva (a tudi zahtevna) naloga avtomatizirati indeksiranje učnih enot po domenskih konceptih. Tu bi se lahko oprli predvsem na tehnike, ki so poznane s področja iskanja in izbiranja informacij (angl. information retrieval).

V disertaciji smo nakazali tudi načine združevanja informacij modelov uporabnikov, kar prinaša zelo zanimive možnosti nadaljnjega razvoja. Ideje združevanja informacij, tako primerjave kot sodelovanja modelov uporabnikov, bi bilo nadalje potrebno razviti do podrobnosti in jih implementirati v okviru obstoječega sistema.

8.3 Pregled prispevkov k znanosti

Glavni znanstveni prispevki disertacije so naslednji:

- **Primerjalna analiza različnih pristopov k obravnavi nedoločenosti pri modeliranju uporabnika. Izbira in uporaba primerne rešitve za modeliranje uporabnikovega znanja z upoštevanjem nezanesljivosti pri določanju stopnje poznavanja konceptov.** Sistematično smo pregledali različne možnosti in načine obravnave nedoločenosti in njihove aplikacije pri modeliranju uporabnika. Kot rezultat navedenih primerjav smo za naše potrebe pri modeliranju nezanesljivosti znanja uporabnika izbrali pristop z mehko logiko. Znanje uporabnika smo opisali s pomočjo mehkih množic, posodabljanje vrednosti znanja pa temelji na mehkih pravilih.
- **Razvoj postopka prilagodljive podpore navigaciji z vstavljanjem povezav.** Vpeljali smo novo tehnologijo prilagodljive podpore navigaciji z vstavljanjem povezav na vse relevantne enote glede na trenutno učno enoto. Tak postopek je kombinacija skrivanja in sortiranja povezav, saj na podlagi grafa domene in modela uporabnika dinamično izbira povezave na enote, ki so najprimernejše za nadaljevanje in najbližje (tako vsebinsko kot težavnostno) trenutni enoti. Tako sestavljen seznam povezav ne vključuje vseh možnih povezav iz enote (skrivanje), povezave pa so v seznamu razvrščene po primernosti (sortiranje). Postopek se v veliki meri naslanja na mehko predstavitev uporabnikovega poznavanja konceptov in njihove medsebojne odvisnosti.

- **Načrtovanje in implementacija prilagodljivega hipermedijskega sistema, ki uporablja postopek vstavljanja povezav na podlagi mehkega modela uporabnika.** Izdelali smo prototip hipermedijskega sistema za učenje, ki implementira zgoraj navedeni postopek prilagodljivega vstavljanja povezav za podporo pri navigaciji in modelira uporabnikovo znanje s pomočjo mehkih množic in mehkih pravil.
- **Ovrednotenje razvitega sistema na testni domeni in ocena učinkovitosti uporabljenih prilagoditvenih tehnologij.** Izdelan prototip hipermedijskega sistema smo testirali in ovrednotili na testni učni domeni. Prototip sistema je služil tudi kot orodje za oceno učinkovitosti tehnologije prilagodljivega vstavljanja povezav. Dobljene rezultate smo primerjali z izsledki drugih raziskav s tega področja. Delovanje sistema smo preverili v realnem okolju pri učenju jezika Java, kjer se je izkazal kot učinkovit pripomoček in enostaven za uporabo. Uporabniki, ki so izkoriščali prilagoditvene možnosti sistema, so pri kontrolnem testu znanja dosegali v povprečju za okoli četrtino boljše rezultate.
- **Uvedba komparativnega in kooperativnega združevanja informacij modelov uporabnikov.** Pregledali smo možnosti združevanja modelov uporabnikov, predvsem v smislu primerjanja in sodelovanja med njimi. To sta sicer dve različni nalogi, ki pa se po potrebi ob sočasni uporabi tudi dopolnjujeta. Oba načina sta noviteti in ju pri prilagodljivih hipermedijih še nismo zasledili. Pri primerjavi modelov gre predvsem za nekoliko drugačen pogled na dosežen nivo znanja. Znanje posameznega uporabnika se ne meri več absolutno (primerjava s popolnim poznavanjem vseh domenskih konceptov oziroma z ekspertovim znanjem), temveč relativno glede na ostale (so)uporabnike sistema. Tak način omogoča bolj življenjsko določanje poznavanja domene in zastavljanje učnih ciljev ter hkrati nudi možnost razpoznavanja uporabnikom težje razumljivih učnih enot. Drugačen način pa predstavlja združevanje uporabnikov v skupine, katerih cilj je “skupinsko” poznavanje učne domene. Vsak član skupine obvlada določen del domenskega znanja, vsi skupaj pa pokrivajo celotno področje. Seveda želimo pri tem doseči čim manjšo redundanco znanja in tako optimizirati čas učenja.

9. Literatura

[AH 2000]

Adaptive Hypertext and Hypermedia Home Page, 2000.
(http://wwwis.win.tue.nl/ah/)

[Angelides 1995]

M. C. Angelides, "Developing hybrid intelligent tutoring and hypertext systems", *The New Review of Hypermedia and Multimedia: Applications and research*, Volume 1, 1995. pp. 67-106.

[Armstrong 1995]

R. Armstrong, D. Freitag, T. Joachims in T. Mitchell, "WebWatcher: A learning Apprentice for the World Wide Web", *AAAI Spring Symposium on Information Gathering from Distributed, Heterogeneous Environments*, Stanford, CA, USA, 1995.

[Bauer 1995]

M. A. Bauer, "A Dempster–Shafer Approach to Modeling User Preferences for Plan Recognition", *User Modeling and User–Adapted Interaction*, Volume 5, Number 3/4, 1995. pp. 317-348.

[Bayes 1763]

Rev. T. Bayes, "An Essay towards solving a Problem in the Doctrine of Chances", *Philosophical Transactions of the Royal Society*, Volume 53, 1763. pp. 370-418.

[Beaumont 1994]

I. Beaumont, "User Modeling in the Interactive Anatomy Tutoring System ANATOM–TUTOR", *User Modeling and User–Adapted Interaction*, Volume 4, Number 1, 1994. pp. 21-45.

[Beck 1996]

J. Beck, M. Stern in E. Haugsjaa, "Applications of AI in Education", *ACM Crossroads*, 1996.

[Benyon 1997]

D. Benyon in K. Höök, "Navigation in Information Spaces: supporting the individual", *Proceedings of Interact'97*, Sydney, Australia, Chapman & Hall, 1997.

[Berg 2000]

G. A. Berg, "Towards a Learning Theory for Educational Technology", *WebNet Journal: Internet Technologies, Applications & Issues*, Volume 2, Number 1, January–March 2000. pp. 5-6.

[Bezdek 1993]

J. Bezdek, "Fuzzy Models – What Are They, and Why?", *IEEE Transactions on Fuzzy Systems*, Volume 1, Number 1, February 1993. pp. 1-6.

[Blue 1997]

M. Blue, B. Bush in J. Puckett, "Applications of Fuzzy Logic to Graph Theory," *Report LA-UR-96-4792*, Los Alamos National Laboratory, 1997.

[Boyle 1998]

C. Boyle in A. O. Encarnacion, "Metadoc: An Adaptive Hypertext Reading System", *Adaptive Hypertext and Hypermedia*, Kluwer Academic Publishers, 1998. pp. 71-89.

[Bratko 1989]

I. Bratko, *Prolog in umetna inteligenca*, Ljubljana: Društvo matematikov, fizikov in astronomov SR Slovenije: Zveza organizacij za tehnično kulturo Slovenije, 1989.

[Brickell 1993]

G. Brickell, "Navigation and learning style", *Australian Journal of Educational Technology*, Volume 9, Number 2,1993. pp. 103-114.

[Brule 1992]

J. F. Brule, *Fuzzy Systems – A Tutorial*, 1992.
(http://www.csu.edu.au/complex_systems/fuzzy.html)

[Brusilovsky 2001]

P. Brusilovsky in P. Miller, "Course Delivery Systems for the Virtual University", *Access to Knowledge: New Information Technologies and the Emergence of the Virtual University*, Amsterdam: Elsevier Science and International Association of Universities, 2001. pp. 167-206.

[Brusilovsky 2000]

P. Brusilovsky, O. Stock in C. Strapparava (eds.), *Adaptive Hypermedia and Adaptive Web-Based Systems, Proceedings of the International Conference AH 2000, Trento, Italy, August 2000, Lecture Notes in Computer Science 1892*, Springer Verlag, 2000.

[Brusilovsky 1998a]

P. Brusilovsky, "Methods and Techniques of Adaptive Hypermedia", *Adaptive Hypertext and Hypermedia*, Kluwer Academic Publishers, 1998. pp. 1-43.

[Brusilovsky 1998b]

P. Brusilovsky in J. Eklund, "A Study of User Model Based Link Annotation in Educational Hypermedia", *Journal of Universal Computer Science*, Volume 4, Issue 4, 1998. pp. 429-448.

[Brusilovsky 1998c]

P. Brusilovsky in L. Pesin, "Adaptive Navigation Support in Educational Hypermedia: An Evaluation of the ISIS–Tutor", *Journal of Computing and Information Technology – CIT*, Volume 6, Number 1, 1998. pp. 27-38.

[Brusilovsky 1998d]

P. Brusilovsky, "Adaptive Educational Systems on the World–Wide Web: A Review of Available Technologies", *Proceedings of Workshop WWW–Based Tutoring at 4th International Conference on Intelligent Tutoring Systems ITS'98*, San Antonio, USA, August 1998.

[Brusilovsky 1996a]

P. Brusilovsky, "Adaptive hypermedia: an attempt to analyze and generalize", *Multimedia, Hypermedia, and Virtual Reality*, Springer Verlag, Berlin, 1996.

[Brusilovsky 1996b]

P. Brusilovsky, E. Schwarz in G. Weber, "ELM–ART: An Intelligent Tutoring System on World Wide Web", *Proceedings of the Third International Conference on Intelligent Tutoring Systems ITS'96*, Montreal, Canada, 1996. pp. 261-269.

[Brusilovsky 1994]

P. Brusilovsky in L. Pesin, "ISIS–Tutor: An Intelligent Learning Environment for CDS/ISIS Users", *Proceedings of the Interdisciplinary Workshop on Complex Learning in Computer Environments CLCE'94*, Joensuu, Finland, 1994.

[Brusilovsky 1992]

P. Brusilovsky, "Intelligent Tutor, Environment and Manual for Introductory Programming", *Educational and Training Technology International*, Volume 29, Number 1, 1992. pp. 26-34.

[Bush 1945]

V. Bush, "As We May Think", *The Atlantic Monthly*, Volume 176, Number 1, July 1945. pp. 101-108.

[Calvi 2000]

L. Calvi, "Formative Evaluation of Adaptive CALLware: A Case Study", *Adaptive Hypermedia and Adaptive Web-Based Systems, Proceedings of the International Conference AH 2000, Trento, Italy, August 2000, Lecture Notes in Computer Science 1892*, Springer Verlag, 2000. pp. 276-279.

[Campione 2000]

M. Campione, K. Walrath in A. Huml, *The Java Tutorial: A Short Course on the Basics*, Third Edition, Addison-Wesley, 2000. (http://java.sun.com/docs/books/tutorial/)

[Capuano 2000]

N. Capuano, M. Marsella in S. Salerno, "ABITS: An Agent Based Intelligent Tutoring System for Distance Learning", *Proceedings of the International Workshop on Adaptive and Intelligent Web-based Educational Systems held in Conjunction with ITS 2000 Montreal, Canada. Osnabrück: Technical Report of the Institute for Semantic Information Processing*, 2000. pp. 17-28.

[Carro 1999]

R. M. Carro, E. Pulido in P. Rodríguez, "TANGOW: Un Sistema de Enseñanza Adaptiva a través de Internet", *Congreso Nacional de Informática Educativa, CONIED'99, Proceedings*, Puertollano, Ciudad Real, November 1999.

[Carver 1999a]

C. A. Carver, R. A. Howard in W. D. Lane, "Enhancing Student Learning Through Hypermedia Courseware and Incorporation of Student Learning Styles", *IEEE Transactions on Education*, Volume 42, Number 1, 1999. pp. 33-38.

[Carver 1999b]

C. A. Carver, J. M. D. Hill in U. W. Pooch, "Third Generation Adaptive Hypermedia Systems", *AACE WebNet Conference*, Honolulu, HI, USA, October 1999.

[Catledge 1995]

L. D. Catledge in J. E. Pitkow, "Characterizing Browsing Strategies in the World Wide Web", *Computer Networks and ISDN Systems 27*, Elsevier Science, 1995.

[Chen 1997]

S. Y. Chen in N. J. Ford, "Towards Adaptive Information Systems: Individual Differences and Hypermedia", *Information Research: an electronic journal*, Volume 3, Number 2, September 1997. (http://www.shef.ac.uk/~is/publications/infres/paper37.html)

[Chin 1989]

D. N. Chin, "KNOME: Modeling what the user knows in UC", *User Models in Dialog Systems*, Springer Verlag, New York, 1989. pp. 74-107.

[Clibbon 1995]

K. Clibbon, "Conceptually Adapted Hypertext For Learning", *Proceedings of CHI'95*, Denver: ACM, 1995. pp. 224-225.

[Conati 1996]

C. Conati in K. VanLehn, "POLA: A student modeling framework for probabilistic on–line assessment of problem solving performance", *Proceedings of UM'96, Fifth International Conference on User Modeling*, Kailua–Kona, HI, USA, 1996.

[Cooper 1990]

G. F. Cooper, "The Computational Complexity of Probabilistic Inference Using Bayesian Belief Networks", *Artificial Intelligence*, Volume 42, 1990. pp. 393-405.

[De Bra 1999]

P. De Bra, P. Brusilovsky in G. J. Houben, "Adaptive Hypermedia: From Systems to Framework", *ACM Computing Surveys*, Volume 31, Number 4es, December 1999.

[De Bra 1998]

P. De Bra in L. Calvi, "AHA: a Generic Adaptive Hypermedia System", *Proceedings of the 2nd Workshop on Adaptive Hypertext and Hypermedia HYPERTEXT'98*, Pittsburgh, USA, 1998.

[De Bra 1997]

P. De Bra, "Teaching Hypertext and Hypermedia through the Web", *International Journal of Educational Telecommunications*, Volume 3, Number 2/3, 1997.

[Deitel 2000]

H. M. Deitel, P. J. Deitel in T. R. Nieto, *Internet and World Wide Web, How to program*, Prentice Hall, 2000.

[Dix 1993]

A. Dix, J. Finlay, G. Aboed in R. Beale, *Human–computer Interaction*, Prentice Hall, 1993.

[Duchastel 1992]

P. Duchastel, "Towards Methodologies for Building Knowledge–Based Instructional Systems", *Instructional Science*, Volume 20, Number 5/6, 1992.

[Duda 1979]

R. O. Duda, J. Gaschnig in P. E. Hart, "Model Design in the Prospector Consultant System for Mineral Exploration", *Expert Systems in the Micro Electronic Age*, Edinburgh University Press, Edinburgh, 1979. pp. 153-167.

[Ebersole 1997]

S. Ebersole, “Cognitive Issues in the Design and Deployment of Interactive Hypermedia: Implications for Authoring WWW Sites”, *Interpersonal Computing and Technology: An Electronic Journal for the 21st Century*, Volume 5, Number 1/2, 1997. pp. 19-36.

[Eklund 1999]

J. Eklund, J. Sawers in R. Zeiliger, “NESTOR Navigator: A Tool For The Collaborative Construction Of Knowledge Through Constructive Navigation”, *Proceedings of Ausweb’99, The Fifth Australian World Wide Web Conference*, Southern Cross University Press, Lismore, 1999.

[Eklund 1998a]

J. Eklund in P. Brusilovsky, “The Value of Adaptivity in Hypermedia Learning Environments: A short Review of Empirical Evidence”, *The 2nd Workshop on Adaptive Hypertext and Hypermedia held in conjunction with HYPERTEXT '98: The Ninth ACM Conference on Hypertext and Hypermedia*, Pittsburgh, USA, 1998.

[Eklund 1998b]

J. Eklund in P. Brusilovsky, “Individualising Interaction in Web–based Instructional Systems in Higher Education”, *AUC Academic Conference 98*, University of Melbourne, Melbourne, Australia, 1998.

[Eklund 1997]

J. Eklund, P. Brusilovsky in E. Schwarz, “Adaptive Textbooks on the World Wide Web”, *Proceedings of AUSWEB’97, The Third Australian Conference on the World Wide Web*, Queensland, Australia, 1997.

[Eklund 1996]

J. Eklund in R. Zeiliger, “Navigating the Web: Possibilities and Practicalities for Adaptive Navigational Support”, *Proceedings of Ausweb’96, The Second Australian World Wide Web Conference*, Southern Cross University Press, 1996.

[Eklund 1995]

J. Eklund, “Cognitive models for structuring hypermedia and implications for learning from the world-wide web”, *Innovation and Diversity – The World Wide Web in Australia, AusWeb’95 – Proceedings of the First Australian World Wide Web Conference*, Lismore, Australia, Norsearch Publishing, 1995.

[Encarnação 1995]

L. M. Encarnação, “Adaptivity in Graphical User Interfaces: An Experimental Framework”, *Computers & Graphics*, Volume 19, Number 6, 1995. pp. 873-884.

[Espinoza 1996]

F. Espinoza in K. Höök, "A WWW Interface to an Adaptive Hypermedia System", *Practical Applications of Intelligent Agents and Multi–Agents: PAAM'96, The First International Conference and Exhibition*, London, UK, 1996.

[Fernandez 1997]

B. Fernandez–Manjon, A. Fernandez–Valmayor, "Improving World Wide Web educational uses promoting hypertext and standard general markup language content–based features", *Education and Information Technologies*, Volume 2, 1997. pp. 193-206.

[Fox 1993]

T. Fox, G. Grunst in K. J. Quast, "HyPlan – A Context-Sensitive Hypermedia Help System", *Report No. 743: Arbeitspapiere der GMD*, GMD, Germany, 1993.

[Frigon 1997]

N. L. Frigon in D. Mathews, *Practical guide to experimental design*, John Wiley & Sons, New York, 1997.

[Fulantelli 2000]

G. Fulantelli, R. Rizzo, M. Arrigo in R. Corrao, "An Adaptive Open Hypermedia System on the Web", *Adaptive Hypermedia and Adaptive Web-Based Systems, Proceedings of the International Conference AH 2000, Trento, Italy, August 2000, Lecture Notes in Computer Science 1892*, Springer Verlag, 2000. pp. 305-310.

[Gagné 1988]

R. M. Gagné, *Principles of Instructional Design*, Third Edition, Holt, Rinehart and Winston, New York, 1988.

[Gams 1998]

M. Gams, *Informacijska družba*, Ljubljana: Institut Jožef Stefan: DZS, 1998.

[Gedge 1997]

R. Gedge, "Navigating in Hypermedia – Interfaces and Individual Differences", *Proceedings of SITE'97, Eighth International Conference of the Society for Information Technology and Teacher Education (SITE)*, Orlando, Florida, USA, 1997.

[Gertner 1998]

A. S. Gertner, C. Conati in K. VanLehn, "Procedural help in Andes: Generating hints using Bayesian network student model", *Proceedings of the 15th National Conference on Artificial Intelligence*, Madison, Wisconsin, USA, 1998.

[Gilbert 1999]

J. E. Gilbert in C. Y. Han, "Adapting instruction in search of 'a significant difference'", *Journal of Network and Computer Applications*, Volume 22, 1999.

[Goldman 2001]

S. R. Goldman, "Professional Development in a Digital Age: Issues and Challenges for Standards-Based Reform", *Interactive Educational Multimedia*, Number 2, March 2001. pp. 19-46.

[Good 1990]

T. L. Good in J. E. Brophy, *Educational psychology: A realistic approach*, 4th Edition, White Plains, NY: Longman, 1990.

[Greening 1998]

T. Greening, "WWW support of student learning: A case study", *Australian Journal of Educational Technology*, Volume 14, Number 1, 1998. pp. 49-59.

[Gürer 1995]

D. W. Gürer, M. desJardins in M. Schlager, "Representing a Student's Learning States and Transitions", *AAAI Spring Symposium on Representing Mental States and Mechanisms, Stanford, CA, USA, AAAI technical report, SRI International*, 1995.

[Gutiérrez 1996]

J. Gutiérrez, T. A. Pérez, I. Usandizaga in P. Lopistéguy, "HyperTutor: Adapting Hypermedia Systems to the User", *Proceedings of the 5th International Conference on User Modeling UM'96*, Kailua-Kona, Hawaii, USA, 1996.

[Guttormsen 2000]

S. Guttormsen Schär in H. Krueger, "Using New Learning Technologies with Multimedia", *IEEE MultiMedia*, Volume 7, Number 3, July – September 2000. pp. 40-51.

[Halpern 1997]

J. Y. Halpern, "A Logical Approach to Reasoning about Uncertainty: A Tutorial", *Discourse, Interaction, and Communication*, Kluwer, 1997.

[Hammond 1991]

N. Hammond, "Teaching with hypermedia: problems and prospects", *Hypermedia/Hypertext: An Object–oriented Databases*, Chapman & Hall, 1991. pp. 107-124.

[Hayes 1998]

N. Hayes in S. Orrell, *Psihologija*, druga izdaja, Ljubljana: Zavod Republike Slovenije za šolstvo, 1998.

[Hays 1994]

W. L. Hays, *Statistics*, 5th edition, Harcourt Brace College Publishers, Fort Worth, 1994.

[Hedberg 1993]

J. G. Hedberg, B. Harper in C. Brown, "Reducing cognitive load in multimedia navigation", *Australian Journal of Educational Technology*, Volume 9, Number 2, 1993. pp. 157-181.

[Henze 2000a]

N. Henze, *Adaptive Hyperbooks: Adaptation for Project–Based Learning Resources*, Ph.D. Dissertation, Universität Hannover, 2000.

[Henze 2000b]

N. Henze in W. Nejdl, "Extendible Adaptive Hypermedia Courseware: Integrating Different Courses and Web Material", *Adaptive Hypermedia and Adaptive Web-Based Systems, Proceedings of the International Conference AH 2000, Trento, Italy, August 2000, Lecture Notes in Computer Science 1892*, Springer Verlag, 2000. pp. 109-120.

[Henze 1999]

N. Henze in W. Nejdl, "Bayesian Modeling for Adaptive Hypermedia Systems", *ABIS'99, 7. GI-Workshop Adaptivität und Benutzermodellierung in interaktiven Softwaresystemen*, Otto-von-Guericke-Universität, Magdeburg, Deutschland, 1999.

[Hockemeyer 1998]

C. Hockemeyer, T. Held in D. Albert, "RATH: A Relational Adaptive Tutoring Hypertext WWW–Environment Based on Knowledge Space Theory", *Computer Aided Learning and Instruction in Science and Engineering, Proceedings of the Fourth International Conference CALISCE'98*, Göteborg, Sweden, June 1998.

[Hohl 1998]

H. Hohl, H. D. Böcker in R. Gunzenhäuser, "Hypadapter: An Adaptive Hypertext System for Exploratory Learning and Programming", *Adaptive Hypertext and Hypermedia*, Kluwer Academic Publishers, 1998. pp. 117-142.

[Höök 1996]

K. Höök, J. Karlgren, A. Waern, N. Dahlbäck, C. G. Jansson, K. Karlgren in B. Lemaire, "A Glass Box Approach to Adaptive Hypermedia", *User Modeling and User–Adapted Interaction*, Volume 6, Number 2/3, 1996. pp. 157-184.

[Horton 2000]

W. Horton, *Designing Web–Based Training, How to teach anyone anything anywhere anytime*, Wiley, 2000.

[Hothi 2000]

J. Hothi, W. Hall in T. Sly, "A Study Comparing the Use of Shaded Text and Adaptive Navigational Support in Adaptive Hypermedia", *Adaptive Hypermedia and Adaptive Web-Based Systems, Proceedings of the International Conference AH 2000, Trento, Italy, August 2000, Lecture Notes in Computer Science 1892*, Springer Verlag, 2000. pp. 335-342.

[Hothi 1998]

J. Hothi in W. Hall, "An Evaluation of Adapted Hypermedia Techniques Using Static User Modelling", *Proceedings of the 2nd Workshop on Adaptive Hypertext and Hypermedia HYPERTEXT'98*, Pittsburgh, USA, 1998.

[Hübscher 2000]

R. Hübscher, "Logically Optimal Curriculum Sequences for Adaptive Hypermedia Systems", *Adaptive Hypermedia and Adaptive Web-Based Systems, Proceedings of the International Conference AH 2000, Trento, Italy, August 2000, Lecture Notes in Computer Science 1892*, Springer Verlag, 2000. pp. 121-132.

[Isukapalli 1999]

S. S. Isukapalli, *Uncertainty Analysis of Transport-Transformation Models*, Ph.D. Dissertation, The State University of New Jersey, New Brunswick, New Jersey, 1999.

[Jameson 1999]

A. Jameson, "User–Adaptive Systems: An Integrative Overview", *Tutorial presented at UM'99, 7th International Conference on User Modeling*, Banff, Canada, 1999.

[Jameson 1996]

A. Jameson, "Numerical Uncertainty Management in User and Student Modelling: An Overview of Systems and Issues", *User Modeling and User–Adapted Interaction*, 5, 1996. pp. 193-251.

[Jamison 1998]

K. D. Jamison, *Modeling Uncertainty Using Probabilistic Based Possibility Theory With Applications To Optimization*, Ph.D. Dissertation, University of Colorado, Denver, 1998.

[JavaScript 1999]

Client-Side JavaScript Reference, version 1.3, 1999.
(http://developer.netscape.com/docs/manuals/js/client/jsref/index.htm)

[JavaScript 1998]

Core JavaScript Reference, version 1.4, 1998.
(http://developer.netscape.com/docs/manuals/js/core/jsref/index.htm)

[Joolingen 1999]

W. van Joolingen, "Cognitive tools for discovery learning", *International Journal of Artificial Intelligence in Education*, 10, 1999. pp. 385-397.

[Kaplan 1998]

C. Kaplan, J. Fenwick in J. Chen, "Adaptive Hypertext Navigation Based On User Goals and Context", *Adaptive Hypertext and Hypermedia*, Kluwer Academic Publishers, 1998. pp. 45-69.

[Katz 1992]

S. Katz, A. Lesgold, G. Eggan in M. Gordin, "Modelling the Student in Sherlock II", *Journal of Artificial Intelligence in Education*, Volume 3, 1992. pp. 495-518.

[Kavčič 2001]

Prilagodljiv hipermedijski izobraževalni sistem ALICE, 2001.
(http://podgana.fri.uni-lj.si/alice/)

[Kavčič 2000a]

A. Kavčič, "Tehnika prilagodljivega vstavljanja povezav v izobraževalnih hipermedijskih sistemih", *Zbornik devete Elektrotehniške in računalniške konference ERK 2000*, Portorož, Slovenija, 2000.

[Kavčič 2000b]

A. Kavčič, "The Role of User Models in Adaptive Hypermedia Systems", *Proceedings of the 10th Mediterranean Electrotechnical Conference MEleCon 2000*, Lemesos, Ciper, 2000.

[Kavčič 1999a]

A. Kavčič, "Adaptation Techniques in Adaptive Hypermedia Systems", *Proceedings of the 22nd International Convention MIPRO'99, Conference on Multimedia and Hypermedia Systems*, Volume 1, Opatija, Hrvaška, 1999.

[Kavčič 1999b]

A. Kavčič, "Adaptive Hypermedia Learning System on the Web", *EUROMEDIA'99, 4th EUROMEDIA Conference 1999*, München, Nemčija, 1999.

[Kavčič 1998]

A. Kavčič, *Prilagodljivi hipermedijski učni sistemi*, Magistrska naloga, Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, 1998.

[Kay 1994]

J. Kay in R. J. Kummerfeld, "An Individualised Course for the C Programming Language", *Proceedings of the Second World Wide Web Conference'94: Mosaic and the Web*, Elsevier, 1994.

[Kettel 2000]

L. Kettel, J. Thomson in J. Greer, "Generating Individualized Hypermedia Applications", *Proceedings of the International Workshop on Adaptive and Intelligent Web-based Educational Systems held in Conjunction with ITS 2000 Montreal, Canada. Osnabrück: Technical Report of the Institute for Semantic Information Processing*, 2000. pp. 37-49.

[Klir 1988]

G. J. Klir in T. A. Folger, *Fuzzy Sets, Uncertainty, and Information*, Prentice–Hall International, 1988.

[Kommers 1996]

P. A. M. Kommers, S. Grabinger in J. C. Dunlap, *Hypermedia Learning Environments: Instructional Design and Integration*, Lawrence Erlbaum Associates, 1996.

[Krause 1993]

P. Krause in D. Clark, *Representing Uncertain Knowledge: An Artificial Intelligence Approach*, Intellect Books, 1993.

[Lamas 2000]

D. R. Lamas, J. Jerrams-Smith in D. Heathcote, "Using Directed World Wide Web Navigation Guidance: An empirical investigation", *Proceedings of Ed–Media 2000*, Montreal, Canada, 2000.

[La Passardiere 1992]

B. de La Passardiere in A. Dufresne, "Adaptive Navigational Tools for Educational Hypermedia", *Computer Assisted Learning*, Berlin, New York: Springer Verlag, 1992. pp. 555-567.

[Lapointe 2000]

S. Lapointe in B. Bobée, "Revision of possibility distributions: A Bayesian inference pattern", *Fuzzy Sets and Systems*, Volume 116, Number 1-3, 2000. pp. 119-140.

[Lee 2000]

J. Lee, K. F. R. Liu in W. Chiang, "A possibilistic–logic–based approach to integrating imprecise and uncertain information", *Fuzzy Sets and Systems*, Volume 113, Number 1-3, 2000. pp. 309-322.

[Linard 1995]

M. Linard in R. Zeiliger, "Designing a Navigational Support for an Educational Software", *Lecture Notes in Computer Science 1015, 5th International Conference EWHCI'95, Moscow, Russia, Selected Papers*, Springer, Berlin, 1995.

[Mann 1999]

M. D. Mann, *Using the Adaptive Navigation Support Technique of Link Hiding in an Educational Hypermedia System: an Experimental Study*, Ph. D. Dissertation, Oklahoma State University, 1999.

[Marolt 2000]

M. Marolt, M. Privošnik in A. Kavčič, *Uvod v programski jezik Java*, učno gradivo, Ljubljana, Fakulteta za računalništvo in informatiko, 2000.
(http://colos1.fri.uni-lj.si/~sis/COMPUTING/Java/JAVA_TECAJ/index.htm)

[Martin 1995]

J. Martin in K. VanLehn, "Student assessment using Bayesian nets", *International Journal of Human Computer Studies*, Volume 42, Number 6, 1995. pp. 575-591.

[Martínez 1998]

R. Martínez–Béjar, H. Shiraz in P. Compton, "Using Ripple Down Rules–based Systems for Acquiring Fuzzy Domain Knowledge", *Proceedings of KAW'98, Eleventh Workshop on Knowledge Acquisition, Modeling and Management*, Banff, Alberta, Canada, 1998.

[Mathé 1994]

N. Mathé in J. Chen, "A User–Centered Approach to Adaptive Hypertext based on an Information Relevance Model", *Proceedings of the Fourth International Conference on User Modeling UM'94*, Hyannis, USA, August 1994. pp. 107-114.

[Mayo 2001]

M. Mayo in A. Mitrovic, "Optimising ITS Behaviour with Bayesian Networks and Decision Theory", *International Journal of Artificial Intelligence in Education*, Volume 12, 2001, to appear.

[McCormack 1998]

C. McCormack in D. Jones, *Building a Web-Based Education System*, Wiley Computer Publishing, 1998.

[McDonald 1999]

S. McDonald, “Spatial Versus Conceptual Maps as Learning Tools in Hypertext”, *Journal of Educational Multimedia and Hypermedia*, Volume 8, Number 1, 1999.

[McLoughlin 1999]

C. McLoughlin, “The implications of the research literature on learning styles for the design of instructional material”, *Australian Journal of Educational Technology*, Volume 15, Number 3, 1999. pp. 222-241.

[Millán 2000]

E. Millán Valldeperas, *Sistema bayesiano para modelado del alumno*, Tesis doctoral, Universidad de Málaga, 2000.

[Mislevy 1995]

R. J. Mislevy in D. H. Gitomer, “The Role of Probability–Based Inference in an Intelligent Tutoring System”, *User Modeling and User–Adapted Interaction*, Volume 5, Number 3/4, 1995. pp. 253-282.

[Mitrovic 2000]

A. Mitrovic in K. Hausler, “Porting SQL–Tutor to the Web”, *Proceedings of the International Workshop on Adaptive and Intelligent Web-based Educational Systems held in Conjunction with ITS 2000 Montreal, Canada. Osnabrück: Technical Report of the Institute for Semantic Information Processing*, 2000. pp. 50-60.

[Mullier 1999a]

D. J. Mullier, D. J. Hobbs in D. J. Moore, “A Hybrid Semantic/Connectionist Approach to Adaptivity in Educational Hypermedia Systems”, *Proceedings of ED-MEDIA’99*, Seattle, USA, 1999.

[Mullier 1999b]

D. Mullier, *The Application of Neutral Network and Fuzzy Logic Techniques to Educational Hypermedia*, Ph.D. Dissertation, Leeds Metropolitan University, 1999.

[Murray 2000a]

T. Murray, C. Condit, J. Piemonte, T. Shen in S. Khan, “Evaluating the Need for Intelligence in an Adaptive Hypermedia System”, *Proceedings of ITS’2000*, Montreal, Canada, 2000.

[Murray 2000b]

T. Murray, T. Shen, J. Piemonte, C. Condit in J. Thibedeau, “Adaptivity in the MetaLinks Hyper–Book Authoring Framework”, *Proceedings of the International Workshop on Adaptive and Intelligent Web-based Educational Systems held in Conjunction with ITS 2000 Montreal, Canada.*

Osnabrück: Technical Report of the Institute for Semantic Information Processing, 2000. pp. 61-72.

[Murray 1998]

T. Murray, C. Condit in E. Haugsjaa, "MetaLinks: A Preliminary Framework for Concept-Based Adaptive Hypermedia", *Workshop Proceedings for WWW-Based Tutoring in ITS'98 Conference*, San Antonio, 1998.

[Nelson 1967]

T. H. Nelson, "Getting it out of our system", *Information retrieval: A critical review*, Washington DC: Thompson Books, 1967.

[Newton 1997]

P. Newton, "HIPPO: Incorporating Hypertext using Fuzzy Components", *Position paper, Incorporating Hypertext Functionality Into Software Systems III Workshop held in conjunction with the ACM Hypertext'97 Conference*, Southampton, UK, 1997.

[Nielsen 1990]

J. Nielsen, *Hypertext and Hypermedia*, Academic Press, San Diego, 1990.

[Nykänen 1998]

O. Nykänen in M. Ala–Rantala, "A design for a hypermedia–based learning environment", *Education and Information Technologies*, Volume 3, 1998. pp. 277-290.

[Oliver 1995]

R. Oliver in J. Herrington, "Developing effective hypermedia instructional materials", Australian Journal of Educational Technology, Volume 11, Number 2, 1995. pp. 8-22.

[Papanikolaou 2000]

K. A. Papanikolaou, G. D. Magoulas in M. Grigoriadou, "A Connectionist Approach for Supporting Personalized Learning in a Web–Based Learning Environment", *Adaptive Hypermedia and Adaptive Web-Based Systems, Proceedings of the International Conference AH 2000, Trento, Italy, August 2000, Lecture Notes in Computer Science 1892*, Springer Verlag, 2000. pp. 189-201.

[Parsons 1998]

S. Parsons in A. Hunter, "A Review of Uncertainty Handling Formalisms", *Applications of Uncertainty Formalisms, Lecture Notes in Artificial Intelligence 1455*, Springer, 1998.

[Pavešić 1981]

N. Pavešić in F. Mihelič, *Statistične metode*, Fakulteta za elektrotehniko, Ljubljana, 1981.

[Pearl 1988]

J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, San Mateo, California: Morgan Kaufmann, 1988.

[Pérez 1995]

T. A. Pérez, J. Gutiérrez in P. Lopistéguy, "An Adaptive Hypermedia System", *Artificial Intelligence in Education AIED'95*, Charlottesville, USA, 1995.

[Pilar 1998]

D. Pilar da Silva, R. Van Durm, E. Duval in H. Olivié, "Concepts and documents for adaptive educational hypermedia: a model and a prototype", *Proceedings of the 2nd Workshop on Adaptive Hypertext and Hypermedia HYPERTEXT'98*, Pittsburgh, USA, 1998.

[Razaz 1997]

M. Razaz, J. R. King, A. Duke in A. Rogers, "Application of Fuzzy Logic to Multimedia Technology", *Fuzzy Logic and Applications ISFL'97*, ICSC Academic Press, 1997.

[Reye 1996]

J. Reye, "A Belief Net Backbone for Student Modelling", *Intelligent Tutoring Systems, Proceedings of the Third International Conference ITS'96, Montreal, Canada, June 1996, Lecture notes in computer science 1086*, Springer, 1996. pp. 596-604.

[Russell 1923]

B. Russell, "Vagueness", *The Australasian Journal of Psychology and Philosophy*, 1, June 1923. pp. 84-92.

[Sandri 1994]

S. Sandri in G. Bittencourt, "Possibilistic Semantic Nets", *Advances in Intelligent Computing – IPMU'94, Selected Papers of the 5th International Conference on Information Processing and Management of Uncertainty in Knowledge–Based Systems, Paris, France July 1994, Lecture Notes in Computer Science 945*, Springer, 1994. pp. 318-325.

[Sanrach 2000]

C. Sanrach in M. Grandbastien, "ECSAIWeb: A Web–Based Authoring System to Create Adaptive Learning Systems", *Adaptive Hypermedia and Adaptive Web-Based Systems, Proceedings of the International Conference AH 2000, Trento, Italy, August 2000, Lecture Notes in Computer Science 1892*, Springer Verlag, 2000. pp. 214-226.

[Savšek 1998]

T. Savšek, *Odločanje na podlagi primerjave mehkih dreves*, Doktorska disertacija, Fakulteta za elektrotehniko, Univerza v Ljubljani, 1998.

[Schwarz 1996]

E. Schwarz, P. Brusilovsky in G. Weber, "World–Wide Intelligent Textbooks", *Proceedings of ED–TELECOM'96, World Conference on Educational Telecommunications*, Boston, 1996.

[SDK 2000]

Java 2 SDK, Standard Edition Documentation, Version 1.3, 2000.
(http://java.sun.com/j2se/1.3/docs/index.html)

[Shortliffe 1976]

E. H. Shortliffe, *Computer–Based Medical Consultations: MYCIN*, Elsevier/North Holland, New York, 1976.

[Smets 1994]

P. Smets, "Non Standard Probabilistic and Non Probabilistic Representations of Uncertainty", *Advances in Intelligent Computing – IPMU'94, Selected Papers of the 5th International Conference on Information Processing and Management of Uncertainty in Knowledge–Based Systems, Paris, France July 1994, Lecture Notes in Computer Science 945*, Springer, 1994. pp. 13-38.

[Smets 1988]

P. Smets, "Belief Functions versus Probability Functions", *Uncertainty and Intelligent Systems: Proceedings of the Second International Conference on Information Processing and Management of Uncertainty in Knowledge–Based Systems IPMU'88, Urbino, Italy, July 1988, Lecture Notes in Computer Science 313*, Springer Verlag, 1988. pp. 17-24.

[Smithson 1989]

M. Smithson, *Ignorance and uncertainty: emerging paradigms*, Springer Verlag, 1989.

[Specht 2000]

M. Specht, "ACE – Adaptive Courseware Environment", *Adaptive Hypermedia and Adaptive Web-Based Systems, Proceedings of the International Conference AH 2000, Trento, Italy, August 2000, Lecture Notes in Computer Science 1892*, Springer Verlag, 2000. pp. 380-383.

[Stern 2000]

M. K. Stern in B. P. Woolf, "Adaptive Content in an Online Lecture Systems", *Adaptive Hypermedia and Adaptive Web-Based Systems, Proceedings of the International Conference AH*

2000, Trento, Italy, August 2000, Lecture Notes in Computer Science 1892, Springer Verlag, 2000. pp. 227-238.

[Škrjanc 1996]

I. Škrjanc, *Mehko adaptivno vodenje*, Doktorska disertacija, Fakulteta za elektrotehniko, Univerza v Ljubljani, 1996.

[Thomas 1995]

C. G. Thomas, "Basar: A Framework for Integrating Agents in the World Wide Web", *IEEE Computer*, Volume 28, Number 5, 1995. pp. 84-86.

[Thomas 1998]

M. Thomas, "Enhancing Student Learning Outcomes through Hypermedia–Based Learning: Has Theory and Research Been Applied Effectively?", *Unpublished paper presented at the annual conference of the Australian Association for Research in Education, AARE*, Adelaide, Australia, 1998. (http://chomsky.arts.adelaide.edu.au/environmental/research/mthomas/papers/9803/thomas9803.htm)

[Ubell 2000]

R. Ubell, "Engineers turn to e–learning", *IEEE Spectrum*, Volume 37, Number 10, October 2000.

[Vassileva 1997]

J. Vassileva, "Dynamic Courseware Generation", *Journal of Computing and Information Technology – CIT*, Volume 5, Number 2, Jun 1997. pp. 87-102.

[Vassileva 1996]

J. Vassileva, "A Task–Centered Approach for User Modeling in a Hypermedia Office Documentation System", *User Modeling and User–Adapted Interaction*, Volume 6, Number 2/3, 1996. pp. 185-223.

[Vassileva 1995]

J. Vassileva, "Dynamic Courseware Generation: at the Cross Point of CAL, ITS and Authoring", *Proceedings of International Conference on Computers in Education ICCE'95*, Singapore, December 1995. pp. 290-297.

[Vassileva 1994]

J. Vassileva, "A Practical Architecture for User Modeling in a Hypermedia–Based Information System", *Proceedings of the 4th International Conference on User Modeling*, Cape Cod, MA, USA, August 1994. pp. 115-120.

[Veerman 1999]

N. Veerman, “Adaptive Hypermedia: Techniques and Purposes”, 1999. (http://members.tripod.lycos.nl/nveerman/hypermedia.html)

[Virant 2000]

J. Virant, *Design Considerations of Time in Fuzzy Systems*, Kluwer Academic Publishers, 2000.

[Virant 1998]

J. Virant, *Čas v mehkih sistemih*, Didakta, Radovljica, 1998.

[Virant 1992]

J. Virant, *Uporaba mehke logike v sodobnih sistemih: fuzzy logika kot nova možnost za načrtovanje in postavljanje sistemov*, Didakta, Radovljica, 1992.

[Weber 1997]

G. Weber in M. Specht, “User Modeling and Adaptive Navigation Support in WWW–based Tutoring Systems”, *Proceedings of the 6th International Conference on User Modeling*, Sardinia, Italy, 1997. pp. 289-300.

[Weber 1995]

G. Weber in A. Möllenberg, “ELM Programming Environment: A Tutoring System for LISP Beginners”, *Cognition and Computer Programming*, Norwood, NJ: Ablex Publishing Corporation, 1995.

[Wenger 1987]

E. Wenger, *Artificial Intelligence and Tutoring Systems: Computational and Cognitive Approaches to the Communication of Knowledge*, Morgan Kaufmann Publishers, Inc., Los Altos, 1987.

[Williams 2000]

J. Williams, N. Steele in H. Robinson, “Modelling Non-Numeric Linguistic Variables”, *Soft Computing Techniques and Applications, International Conference on Recent Advances in Soft Computing, RASC 2000, Leicester, England, June 2000, Advances in Soft Computing Series*, Springer, 2000.

[Wittig 2000]

F. Wittig in A. Jameson, “Exploiting Qualitative Knowledge in the Learning of Conditional Probabilities of Bayesian Networks”, *Uncertainty in Artificial Intelligence: Proceedings of the Sixteenth Conference*, San Francisco, Morgan Kaufmann, 2000.

[Woodhead 1991]

N. Woodhead, *Hypertext & Hypermedia: Theory and Applications*, Addison–Wesley Publishing Company, 1991.

[Wu 2000]

H. Wu, P. De Bra, A. Aerts in G. J. Houben, "Adaptation Control in Adaptive Hypermedia Systems", *Adaptive Hypermedia and Adaptive Web-Based Systems, Proceedings of the International Conference AH 2000, Trento, Italy, August 2000, Lecture Notes in Computer Science 1892*, Springer Verlag, 2000. pp. 250-259.

[Wu 1999]

H. Wu, G. J. Houben in P. De Bra, "User Modeling in Adaptive Hypermedia Applications", *Proceedings of the Interdisciplinaire Conferentie Informatiewetenschap*, Amsterdam, 1999. pp. 10-21.

[Yaghlane 1999]

B. B. Yaghlane in K. Mellouli, "Updating Directed Belief Networks", *Symbolic and Quantitative Approaches to Reasoning and Uncertainty, European Conference, ECSQARU'99, London, July 1999, Proceedings, Lecture Notes in Artificial Intelligence 1638*, Springer, 1999. pp. 43-54.

[Zadeh 1999]

L. A. Zadeh, "Some Reflections on the Relationship Between AI and Fuzzy Logic (FL) – A Heretical View", *Fuzzy Logic in Artificial Intelligence, IJCAI'97 Workshop, Selected and Invited Papers, Nagoya, Japan, August 1997, Lecture Notes in Artificial Intelligence 1566*, Springer, 1999. pp. 1-8.

[Zadeh 1965]

L. A. Zadeh, "Fuzzy Sets", *Information and Control*, Volume 8, 1965.

[Zeiliger 1999]

R. Zeiliger, "Implementing a Constructivist Approach to Web Navigation Support", *Proceedings of the ED-MEDIA'99 Conference, AACE*, Seattle, USA, 1999.

[Zeiliger 1998]

R. Zeiliger, "Supporting Constructive Navigation of Web Space", *Workshop on Personalised and Social Navigation in Information Space*, Stockholm, Sweden, 1998.

Izjava o avtorstvu disertacije

Podpisana Alenka Kavčič izjavljam, da sem doktorsko disertacijo izdelala samostojno pod vodstvom mentorja prof. dr. Saše Divjaka. Izkazano pomoč drugih sodelavcev sem v celoti navedla v zahvali.

Alenka Kavčič