TREĆA MATEMATIČKA KONFERENCIJA REPUBLIKE SRPSKE
Trebinje, 7. i 8. juni 2013. godine

# Poboljšanje nastavnog plana optimizovanjem broja pokrivenih tematskih oblasti po semestrima

Dragan Matić
Univerzitet u Banja Luci, Prirodno matematički fakultet
matic.dragan@gmail.com

Vladimi Filipović
Univerzitet u Beogradu, Matematički fakultet
vladaf@matf.bg.ac.rs

Jozef Kratica
Matematički institut Beograd
jkratica@mi.sanu.ac.rs

Stručni rad

**Apstrakt**

U ovom radu se ispituju mogućnosti unapređenja nastavnih planova matematičkih i računarskih kurseva, primjenom alata kombinatorne optimizacije. Analizirani pristup je zasnovan na metodičkom stavu da osnove (i ne samo osnove) svake tematske cjeline treba periodično obrađivati u što više različitih konteksta, kao i na rastućem nivou složenosti. Ako je cilj organizovati lekcije u dva vremenska perioda, tako da što veći broj tematskih cjelina bude obrađivan u oba perioda, pokazuje se da odgovarajući matematički model posmatranog problema odgovara postavci poznatog kombinatornog problema dijeljenja skupa (Set Splitting Problem). U radu je na primjeru kursa iz Uvoda u računarstvo predložena organizacija datog kursa prema opisanom pristupu. Navedene su i opisane neke metode rješavanja odgovarajućeg matematičkog problema.

## 1 Uvod

Informatizacijom čitavog društva i povećanjem količine dostupnih informacija, obrazovanje se posljednjih godina susreće sa novim izazovom: da li velika količina obrazovnog materijala koja je dostupna studentima zaista olakšava učenje i podiže nivo studentovog znanja? Usljed nedovoljno dobro organizovanih nastavnih planova, velike količine dostupnih informacija, neadekvatnih udžbenika i metoda podučavanja, studentima je u doba intenzivne primjene naprednih metoda učenja često otežan obrazovni proces, što je suprotno od željenog cilja. Naglašavanjem da se, umjesto na istraživanju pitanja i podsticanju kritične misli, studentima nudi učenje odgovora, ulazi se u problem da studenti nisu podstaknuti da rade i uče zajedno, dijele ideje i informacije, te proširuju intelektualne sposobnosti.

Čest je slučaj da ambiciozni planovi i programi na matematičkim fakultetima uzrokuju nedostatak nastavnog kadra i lošu iskorištenost resursa, što sa jedne strane otežava nastavni proces, a sa druge generiše nepotrebne troškove. U nesređenim planovima i programima, neke teme se obrađuju do manje potrebnih detalja, dok su neke druge, iste ili čak i veće važnosti, preskočene, ili dostupne manjim grupama studenata. Dodatno, svaka tema u nauci, matematici ili tehnologiji koja se uči ili pominje samo u jednoj lekciji, najvjerovatnije neće biti dovoljno shvaćena i naučena od strane studenta. Da bi se koncept koji se prezentuje suštinski shvatio, studentima se on treba prezentovati periodično, u različitim kontekstima i u rastućem nivou složenosti [1].

Uvođenje jednosemestralnih predmeta dovelo je do "podjele" ranijih planova i programa na dva ili više dijelova. Te podjele su u najvećem broju slučajeva podrazumijevale da se lekcije redom uključuju u prvi, odnosno drugi semestar. Sa druge strane, podjela plana i programa kursa na dva dijela može biti zasnovana i na drugačijim principima. Na primjer, jedan mogući pristup zasnovan je na konceptu da se, prije nego što se materijal podijeli na dva dijela, prvobitno odrede težine pojedinačnih lekcija, a da se podjela vodi principom da dijelovi budu ujednačene težine. Kako bi se održao kontinuitet u predavanju lekcija, uvodi se dodatni uslov, koji zahtijeva povezanost lekcija unutar svakog dijela. Principi podjele kursa na ovaj način prikazani su u [2]. Da bi se napravio matematički model koji odgovara ovom pristupu, lekcije kursa se posmatraju kao čvorovi grafa, dok se veze između lekcija (grane grafa) ostvaruju po nekoliko kriterijuma, kao što su sličnost, analogija, uopštenje ili uslovljenost. U [2] je pokazano da podjela lekcija kursa u dvije cjeline uz zahtjeve da svaka cjelina bude povezana i da težine te dvije cjeline budu u što većoj mjeri slične, odgovara matematičkom NP teškom problemu pronalaženja maksimalno balansirane povezane particije u grafu.

U ovom radu se ispituje drugačiji pristup podjele lekcija u jednog kursa u dva dijela, koji podrazumijeva da se što više koncepata (tematskih cjelina) pominje i obrađuje u oba dijela. Za dati kurs se, dakle, identifikuju tematske cjeline koje se obrađuju unutar njega, a podjela lekcija na dva dijela je vođena pristupom da što više tematskih cjelina bude "pokriveno" barem jednom lekcijom u svakom dijelu.

Pokazaće se da određivanje što bolje moguće raspodjele lekcija po opisanom principu odgovara poznatom optimizacionom problemu maksimalnog dijeljenja skupa (eng. Maximum Set Splitting problem - MSSP). Time se otvara mogućnost za rješavanje opisanog problema u nastavi već razvijenim metodama za rješavanje MSSP.

## 2 Polazne pretpostavke i definicija problema

Da bismo definisali problem i napravili matematički model za njega, polazimo od sljedećih pretpostavki. Materijal za neki kurs se prirodno dijeli po lekcijama. Pretpostavljamo da u datom kursu postoji više tematskih cjelina koje se u nekoj mjeri pokrivaju datim kursom. Jedna lekcija može da spada u različite tem-

atske cjeline, a svaka tematska cjelina sadrži više lekcija. Sa metodičkog aspekta, postavljamo pretpostavku da bi bilo korisno napraviti podjelu lekcija u dvije cjeline (na primjer u zimski i ljetnji semestar), ali tako da što više tematskih cjelina bude prisutno u oba semestra. Time se omogućava da elementi svake tematske cjeline budu ponovljeni i obrađivani, kako je već i zahtijevano, u različitim kontekstima i nivoima složenosti.

Da bismo konstruisali matematičku formulaciju problema, najprije lekcije $L_1, L_2, ..., L_m$ datog kursa posmatrajmo kao elemente skupa $S$. Tematske cjeline (oblasti) identifikujemo preko pripadajućih lekcija, tj. tematske cjeline posmatramo kao podskupove skupa lekcija. Neka je $C = \{S_1, S_2, ..., S_n\}$ kolekcija tematskih cjelina, tj. podskupova od $S$. Problem pronalaženja podjele (particije) skupa lekcija $S$ na dva semestra, tako da je što više temastkih cjelina pokriveno u oba semestra je zapravo pronalaženje particije $(P_1, P_2)$ skupa lekcija, tako da je broj podskupova iz kolekcije $C$ koji imaju neprazan presjek i sa $P_1$ i sa $P_2$ najveći mogući.

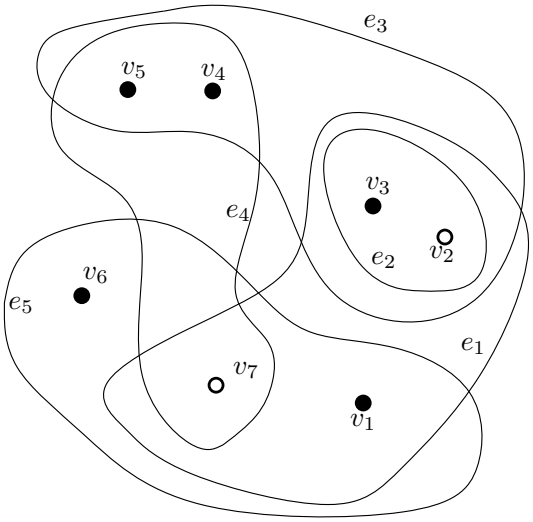
Matematički model problema iz prethodnog pasusa, u kome su lekcije predstavljene elementima skupa $S$, a tematske cjeline kolekcijom $C$, podskupova skupa $S$, doslovce odgovara definiciji MSSP. MSSP je NP težak problem, što je istražujući neke invarijante hipergrafova, prvi dokazao L. Lovász u [3]. Hipergraf je uopštenje grafa, gdje za dati skup čvorova $V$, sa jednom granom, može biti incidentan proizvoljan broj čvorova iz $V$ (ovakva grana naziva se hipergrana).

Problem pronalaženja minimalnog broja boja koje su potrebne da se oboje čvorovi grafa na taj način da nijedna grana nema sve čvorove obojene samo jednom bojom se pokazao podjednako teškim kao i određivanje hromatskog broja grafa. Poseban slučaj je bojenje čvorova u tačno dvije boje, a optimizaciona varijanta ovog problema se naziva "maksimalno 2 - bojenje hipergrafa" (eng. Max Hypergraph 2-Coloring - MH2C). Preciznije, 2 - bojenje hipergrafa je određivanje da li se svi čvorovi grafa mogu obojiti sa dvije broje (na primjer u bijelu i crnu), tako da svaka hipergrana ima najmanje po jedan čvor svake boje. Problem maksimalnog 2-bojenja hipergrafa je maksimizovanje broja hipergrana koje sadrže najmanje po jedan bijeli i jedan crni čvor.

Utvrđeno je da su MSSP i MH2C isti problemi. Zaista, za dati skup čvorova $V$ u hipergrafu, svaka hipergrana definiše jedan podskup od $V$, koji sadrži elemente incidentne sa hipergranom. Tada je skup svih hipergrana zapravo jedna kolekcija $C$ podskupova od $V$. Tako, određivanje 2-bojenja skupa čvorova kojim se maksimizuje broj hipergrana koje sadrže čvorove u obje boje je isti zadatak kao i pronalaženje particije skupa $V$, takve da svaki podskup iz $C$ sadrži čvorove iz obje particije.

**Primer 2.1.** Neka je $S = \{1, 2, 3, 4, 5, 6, 7\}$ i $C = \{S_1, S_2, S_3, S_4, S_5\}$, $S_1 = \{1, 2, 3, 7\}$, $S_2 = \{2, 3\}$, $S_3 = \{2, 3, 4, 5\}$, $S_4 = \{4, 5, 7\}$, $S_5 = \{1, 6, 7\}$. Na slici 1 je prikazan hipergraf, sa čvorovima $\{v_1, v_2, ..., v_7\}$ koji odgovaraju elementima skupa $S$ i hipergranama koje odgovaraju elementima iz $C$. Čvorovi $v_2$ i $v_7$ su obojeni bijelom bojom, a ostali crnom. Lako se vidi da svaka hipergrana ima barem po

jedan čvor obojen u svaku boju. Tako je particija $(P_1, P_2) = (\{2, 7\}, \{1, 3, 4, 5, 6\})$ optimalna, jer svaki podskup sadrži bar po jedan element u svakoj particiji.




Slika 1: Primjer maksimalnog 2 - bojenja hipergrafa

## 3 Analiza upotrebe opisanog pristupa

Da bi se ilustrovao opisani pristup podjele kursa na dva dijela po principu MSSP, odabran je kurs iz Uvoda u računarstvo koji se predaje na prvoj godini studija na Studijskom programu Matematika i informatika Prirodno matematičkog fakulteta u Banjaluci. Ciljevi ovog kursa su:

(a) savladavanje osnovnih znanja iz arhitekture računara;

(b) savladavanje osnovnih znanja iz osnova programiranja, kroz programski jezik Python.

Ukupan fond časova predavanja je 90 (15 sedmica po 3 časa sedmično, u dva semestra).

Da bi se napravio odgovarajući matematički model za dati kurs na koji može da se primijeni neka tehnika za rješavanje MSSP, potrebno je da se urade tri zadatka:

(a) identifikacija lekcija;

(b) identifikacija tematskih cjelina;

(c) smještanje lekcija u odgovarajuće tematske cjeline.

## 3.1 Identifikacija lekcija

Na osnovu plana i programa, dostupnog i na internet stranicama Studijskog programa za matematiku i informatiku, (http://www.matinf.pmfbl.org), vidi se da je sadržaj čitavog kursa raspoređen u dva semestra od po 15 sedmica. S obzirom na fond od 3 časa sedmično, u zavisnosti od obima lekcija i težine gradiva, u svakoj sedmici se najčešće obrađuju po dvije ili tri lekcije. Na taj način, identifikovano je ukupno **76** lekcija, koje u matematičkom modelu predstavljaju elemente skupa koji se particioniše.

Spisak svih lekcija, prikazan je u tabeli 1. Zbog uštede prostora i bolje preglednosti, nazivi nekih lekcija su skraćeni. Po zvaničnom akreditovanom programu, čitav kurs iz Uvoda u računarstvo je podijeljen i organizovan u dva jednosemestralna kursa, Uvod u računarstvo 1 i Uvod u računarstvo 2. Trenutna podjela lekcija po semestrima odgovara redoslijedu lekcija u tabeli 1: prvih 39 lekcija su lekcije iz zimskog, dok su lekcije od rednog broja 40 do 76 iz ljetnjeg semestra. S obzirom na činjenicu da su lekcije već podijeljene po semestrima, cilj istraživanja prikazan u ovom odjeljku je dvojak:

(a) da se ustanovi koliko trenutna podjela kursa odgovara preporuci da što više tematskih cjelina budu prisutne u oba semestra;

(b) da se predloži preraspodjela lekcija, u odnosu na postojeću, kako bi se postigla bolja podjela, na osnovu formiranog matematičkog modela datog MSSP.

Tabela 1 – Spisak lekcija

| R.b. | Naziv lekcije |
|---|---|
| 1 | Hardver i softver. CPU i glavna memorija. Input/Output jedinice. Kategorije softvera. |
| 2 | Analogno i digitalno. Digitalno predstavljanje podataka. Binarni brojevi. |
| 3 | Specifikacija računara. Memorija. Čuvanje podataka u memoriji. Vrste i kapacitet memorije. |
| 4 | Struktura CPU. Instrukcioni ciklus. |
| 5 | Mreže računara, načini povezivanja. Internet. TCP/IP. Domeni. WWW. HTML |
| 6 | Funkcija računarskog sistema. Struktura računarskog sistema. Organizacija i arhitektura računarskog sistema. Fon Nojmanova mašina. |
| 7 | Azbuka i kodovi. Brojčani sistemi. Prevođenje cijelih brojeva. Prevođenje razlomljenog dijela. Hijerarhija podataka u računaru. Zapis znakovnih podataka u računaru. |
| 8 | Sabiranje i oduzimanje u binarnom i heksadecimalnom sistemu. Prevođenje iz heksadecimalnog u dekadni sistem i obratno. |
| 9 | Zapis neoznačenih i označenih brojeva, nepotpuni i potpuni komplement. |
| 10 | Konverzija između zapisa različitih dužina, promjena znaka i sabiranje i oduzimanje neoznačenih brojeva, sabiranje i oduzimanje brojeva u nepotpunom i potpunom komplementu. |
| 11 | Množenje neoznačenih cijelih brojeva, množenje cijelih brojeva u nepotpunom i potpunom komplementu, dijeljenje cijelih brojeva. |
| 12 | Decimalna aritmetika: promjena znaka, sabiranje i oduzimanje. Realni brojevi u nepokretnom i pokretnom zarezu. Zapis sa heksadecimalnom osnovom. |
| 13 | IEEE standard 754. Sabiranje i oduzimanje u nepokretnom i poketnom zarezu. Množenje i dijeljenje u nepokretnom i pokretnom zarezu. |
| 14 | Zapis teksta, slike i zvuka. |
| 15 | Definicija programa, programiranja i računarskih nauka. |
| 16 | Instalacija i opis radnog okruženja za programski jezik Python. Proces formiranja programa u radnom okruženju. Izrazi u Python-u. |
| 17 | Aritmetički i logički izrazi. Tipovi u Python-u. Tipovi int i float. Prioritet operatora. |
| 18 | Varijable i operator dodjele vrijednosti. Složeni operatori i operator dodjele vrijednosti. |

Tabela 1 – Spisak lekcija - nastavak

| R.b. | Naziv lekcije |
|---|---|
| 19 | Greške i mehanizmi otkrivanja grešaka. Pojam funkcije i osnovne osobine funkcija. Lokalne i globalne varijable. |
| 20 | Ugrađene funkcije (eng. Build-in) jezika Python. Stilovi pri pisanju programa koji sadrže izraze i operator dodjele. Komentari u programima. |
| 21 | Stringovi i operacije nad stringovima. Escape sekvence. Stringovi u više linija. Komanda print(). Formatiranje izlaza. Korisnički ulaz i komanda raw_input(). |
| 22 | Moduli i primjeri modula. Importovanje modula. Definisanje vlastitih modula. Šta se dešava prilikom importovanja modula. Programiranje help-a u Python-u. |
| 23 | Objekti i metode u Python-u. |
| 24 | Rad sa slikama. Pikseli i boje. |
| 25 | Liste. Liste i indeksi. Modifikovanje listi. Ugrađene funkcije za rad sa lstama. Obrada članova liste. Izdvajanje dijelova liste (eng. slicing). Inverz liste i palindromi. Pseudonimi (aliasi). Metode liste. Ugniježdene liste. Razne vrste nizova. Datoteke kao liste (predstavljanje datoteke listama). Argumenti komandne linije. |
| 26 | Komande izbora (grananja) u Python-u. Bulova logika. Bulovi operatori. Relacioni operatori. Primjena Bulovih operatora na int, float i string. |
| 27 | Komanda if. Ugnježdene komande grananja, operator dodjele i uslovi grananja. |
| 28 | Komande ponavljanja u Python-u. Komanda for. Funkcija range(). Funkcija enumerate(). |
| 29 | Ugniježdene for petlje. Primjer obilaženja dvodimenzionalne matrice. Komanda while. Komande break i continue. Stilovi pri programiranju komandi grananja i petlji. |
| 30 | Datoteke. Otvaranje datoteka sa računara za čitanje, dodavanje ili pisanje. Otvaranje datoteka sa Interneta. Rad sa datotekom koja ima jedan slog u liniji. Slogovi sa više polja u liniji. Slogovi sa pozicioniranim poljima u liniji. Slogovi sa više linija. Look ahead pri čitanju datoteka. |
| 31 | Pisanje u datoteke. |
| 32 | Skupovi u Python-u. Rječnici. Invertovanje rječnika. Algoritmi. Euklid-ov, Wirth-ov i školski algoritam za NZD. |
| 33 | Rekurzija. Rekurzivni i iterativni algoritam za računanje faktorijala i Fibonacci-jevih brojeva. |
| 34 | Algoritmi pretraživanja. Poređenje programa po dužini vremena izvođenja. Pretraživanje i sortiranje. |
| 35 | Osnovno linearno pretraživanje i pretraživanje sa sentinel-om. Mjerenje vremena linearnog pretraživanja. Binarno pretraživanje. |
| 36 | Sort mjehurićima. Sort izborom. Sort umetanjem. Quick sort. Merge sort. Poređenje vremena rada algoritama sortiranja. |
| 37 | Konstrukcije. Dodatne osobine funkcija. |
| 38 | Izuzeci. Testiranje programa. Debagiranje programa. |
| 39 | Uzorci (eng. Patterns) |
| 40 | Logičke osnove obrade podataka: Bulova algebra, Pun sistem funkcija, SDNF i SKNF, Metode minimizacije logičkih funkcija, Logički elementi za obradu podataka, Kombinacione mreže, Sekvencijalne mreže, Primjeri. |
| 41 | Struktura savremenog računarskog sistema: Fon Nojmanova mašina, Sistem prekida, Brzina obrade podataka |
| 42 | Procesori: Organizacija centralnog procesora, Registri, Tehnologije izrade mikroprocesora, Tranzistori, Tehnologije izrade brzih čipova, CISC i RISC arhitektura mikroprocesora, Osobine RISC procesora. |
| 43 | Mašinske instrukcije: Karakteristike mašinskih instrukcija, Formati i tipovi instrukcija, Broj adresa u instrukciji. Načini adresiranja u mašinskim instrukcijama |
| 44 | Mašinski i asemblerski jezici, Pozivi podprograma, Smještanje podataka u memoriji. |
| 45 | Memorije i ulazno/izlazni podsistem: Unutrašnja memorija, Karakteristike, Keš memorija, Spoljašnje memorije, Magnetni i optički mediji, |
| 46 | U/I moduli, Tehnike izvršavanja U/I operacija, U/I uređaji i njihove karakteristike |
| 47 | Otkrivanje i korekcija grešaka: Metode za otkrivanje i metode za otkrivanje i korekciju grešaka |
| 48 | Pretraživanje: Nalaženje minimuma u listi, Prolaženje kroz listu, Mjerenje vremena pretraživanja, |
| 49 | Linearno pretraživanje: while i for verzije linearnog pretraživanja, Sentinel, Tajming linearnog pretraživanja. |
| 50 | Binarno pretraživanje: Složenost algoritama binarnog pretraživanja, ugrađena funkcija binarnog pretraživanja |
| 51 | Algoritmi, Program |
| 52 | Složenost sortova u najboljem i najgorem slučaju: Sort mjehurićima (eng. Bubble sort), Sort umetanjem (eng. Insert sort), Sort izborom (eng. Choice sort). |

Tabela 1 – Spisak lekcija - nastavak

| R.b. | Naziv lekcije |
|---|---|
| 53 | Sort spajanjem (eng. Merge sort), Brzi sort (eng. Quick sort) |
| 54 | Definnicije funkcija: Podrazumijevani (eng. default) parametri, Liste parametara, Parametri sa nazivima. |
| 55 | Izuzeci: tray i except, except objekata. Izuzeci: Funkcije i izuzeci, raise izuzetak, Still izuzetka. |
| 56 | Testiranje: Funkcionalni test, unit test, black-box test, glass-box test, Nezavisnost, Ograničenja, Testom vođen razvoj programa. Debagiranje. |
| 57 | Obrasci dizajniranja: Fiksne vrijednosti, Steperi i kaunteri, Najpoželjniji nosilac, Zadnji nosilac, |
| 58 | Kontejner, Akumulator, Temporalna varijabla, Zastavica. Klase: Atributi, Metode, Konstruktori. |
| 59 | Specijalne metode: init ; str ; repr ; add ; sub Metode dir i help. |
| 60 | Inkapsulacija u objektno orijentisanom programiranju (eng. Encapsulation) |
| 61 | Polimorfizam i nasljeđivanje: ad hock polimorfizam, Parametarski polimorfizam, Hijerarhijski polimorfizam. |
| 62 | Nasljeđivanje: Primjeri class Atom i class Molecule. |
| 63 | GUI: Osnovni pojmovi, Event-driven programiranje. Modul Tkinter: Widgets: Button, Canvas, Checkbutton, Entry, Frame, Label, Listbox, Menu, Message, Menubutton, Text, TopLevel. |
| 64 | Osnovne GUI konstrukcije: Frame, Entry, Laki i teški procesi. Mutable variables: intVar, StringVar, BooleanVar, DoubleVar. |
| 65 | Modeli, pogledi, kontroleri: Primjer. Korištenje Lambda: Primjer. |
| 66 | Stil pisanja GUI: Font, Color, Layout. Widget-i: Text, Checkbutton, Menu. |
| 67 | Objektno-orijentisani GUI: Primjer. |
| 68 | Baze podataka: Arhitektura baza podataka , sqlite3, MySQLdb, SQLAlchemy. sqlite3 funkcije: connect(), cursor(), execute(), commit(). |
| 69 | SQL tipovi podataka: NULL, INTEGER, REAL, TEXT, BLOB. Tabele u SQL: CREATE TABLE TableName(ColumnName Type, ColumnName Type, ...), INSERT INTO VALUES. |
| 70 | Nalaženje podataka u Bazama podataka: SELECT FROM, fetchone(), fetchall(), SELECT FROM ORDER BY. Upiti sa uslovima: SELECT FROM WHERE, =; ! =; >; <;>=;<=. |
| 71 | Apdejtovanje i brisanje u bazama podataka: UPDATE SET WHERE, DELETE FROM WHERE, DROP TABLE. |
| 72 | Transakcije u Bazama podataka. NULL kao zamjena za nepostojeće podatke. Korištenje JOIN za kombinovanje tabela(INNER JOIN). |
| 73 | Ključevi i ograničenja u Bazama podataka. Napredne tehnike Baza podataka: Agregacija, Grupisanje, SelfJoins, Ugniježdeni upiti. |
| 74 | Mrežni moduli : Socket. Klijent/Server par: Primjer. Mrežni moduli urllib i urllib2: Otvaranje udaljenih datoteka, povraćaj udaljenih datoteka. |
| 75 | Opis funkcija nekih mrežnih modula 1: asynchat, asyncore, cgi, Cookie, cookielib, email, ftplib,gopherlib, httplib, imaplib. |
| 76 | Opis funkcija nekih mrežnih modula 2: mailbox, mailcap, mhlib, nntplib, poplib, robotparser, impleXMLRPCServer, smtpd, smtplib, telnetlib, urlparse, xmlrpclib. |

## 3.2 Identifikacija tematskih cjelina

Određivanje tematskih cjelina koje pokrivaju čitavu oblast računarskih nauka predstavlja zahtjevan posao. U današnje vrijeme, računarske nauke su podložne raznim uticajima i s obzirom na ogromnu količinu naučnog, obrazovnog, komercijalnog, zabavnog i drugog dostupnog informatičkog sadržaja, srećemo prilično heterogene klasifikacije i ontologije naučnih računarskih oblasti.

Kako bi se osigurao objektivan pristup, sa jedne strane, i kompletnost i koreknost sa druge, kao glavna referenca za određivanje tematskih cjelina u ovom radu korišteni su resursi objavljeni od strane udruženja *Association for Computing Machinery* (*ACM*), jednog od najpoznatijih i najvećih svjetskih obrazovnih i naučnih udruženja iz oblasti računarskih nauka. Jedan od zadataka ove organizacije je kontinuirana isporuka materijala koji sadrži preporuke za kreiranje

Tabela 2: Spisak oblasti znanja

| Oblast | Oblast (eng.) | skr. | ob. | iz. |
|---|---|---|---|---|
| Diskretne strukture | Discrete Structures | DS | 6 | 0 |
| Osnove programiranja | Programming Fundamentals | PF | 8 | 0 |
| Algoritmi i kompleksnost | Algorithms and Complexity | AL | 5 | 6 |
| Arhitektura i organizacija | Architecture and Organization | AR | 6 | 4 |
| Operativni sistemi | Operating Systems | OS | 6 | 8 |
| Mrežno računarstvo | Net-Centric Computing | NC | 3 | 6 |
| Programski jezici | Programming Languages | PL | 6 | 5 |
| Interakcija čovjeka i računara | Human-Computer Interaction | HC | 2 | 8 |
| Grafika i vizuelno računanje | Graphics and Visual Computing | GV | 2 | 11 |
| Inteligentni sistemi | Intelligent Systems | IS | 3 | 8 |
| Upravljanje informacijama | Information Management | IM | 3 | 12 |
| Društveni i profesion. aspekti | Social and Professional Issues | SP | 7 | 4 |
| Softverski inženjering | Software Engineering | SE | 8 | 6 |
| Nauka o izračunavanju | Computational Science | CN | 0 | 3 |

planova i programa računarskih kurseva, kao i čitavih studijskih programa, kako bi se uspješnije pratili nagli napredak i brze promjene u oblasti računarstva.

U ovom radu se citira posljednji izvještaj [4] iz 2008. godine, u kome je ACM u saradnji sa odjelom za obrazovanje IEEE udruženja (*IEEE Computer Society Education Activities Board*) predstavio novi Vodič za formiranje planova i programa iz oblasti kompjuterskih nauka, zasnovan na prethodnom izvještaju iz 2001. godine i novim saznanjima koja su stečena u međuvremenu. Ovaj izvještaj pruža validnu osnovu za kreiranje planova studijskih programa iz oblasti računarskih nauka, kao i rasporeda pojedinačnih kurseva i koristi se u mnogim visokoškolskim ustanovama širom svijeta. Kao osnovni doprinos prikazan u ovom izvještaju, identifikovano je tzv. "tijelo znanja" (eng. body of knowledge), koje sadrži sve relevantne računarske oblasti, podijeljene u 14 "oblasti znanja" (eng. knowledge areas). U tabeli 2 prikazane su te oblasti, sa nazivima na našem i engleskom jeziku, skraćenim nazivom na engleskom jeziku, zajedno sa brojem ključnih, odnosno izbornih oblasti.

Svaka od ovih oblasti znanja podijeljena je na tematske cjeline, označene kao ključne ili izborne. Ukupno je predloženo 146 tematskih cjelina, od kojih su 65 ključne, a 81 izborne. U zavisnosti od toga koliki se značaj posvećuje posmatranoj cjelini, u izvještaju [4] je predložen i broj časova koje treba posvetiti ključnim tematskim cjelinama. Prirodno, fundamentalnim oblastima je data veća važnost, tj. predložen je veći broj ključnih tematskih cjelina i veći predložen broj časova. To su, prije svih, oblasti: diskretne strukture, osnove programiranja, algoritmi i kompleksnost, arhitektura i organizacija, operativni sistemi, programski jezici, softverski inženjering, te društveni i profesionalni aspekti. Za ostale oblasti znanja predložen je manji broj sati i manji broj ključnih oblasti. Time se kreatorima planova i programa omogućava veća fleksibilnost i prilagođavanje specifičnim potrebama.

Tabela 3: Spisak tematskih cjelina i pripadajućih lekcija

| R.b. | Naziv tematske cjeline | Oblast (eng.) | Spisak lekcija |
|---|---|---|---|
| 1 | Funkcije, relacije i skupovi | DS | 18,19,32,33,61,37 |
| 2 | Osnovna logika | DS | 17,26,40,70 |
| 3 | Tehnike dokazivanja | DS | 17,19,33,40 |
| 4 | Osnove prebrojavanja | DS | 7,9,17,24,40,48 |
| 5 | Grafovi i stabla | DS | 33,35,50,74 |
| 6 | Diskretna vjerovatnoća | DS | 36,52,53 |
| 7 | Osnove konstrukcija | PF | 2,17,18,20,27,28,29,30,54 |
| 8 | Algoritamsko rješavanje problema | PF | 15,32,33,34,35,38,47,51,56 |
| 9 | Strukture podataka | PF | 17,21,25 |
| 10 | Rekurzija | PF | 33,35,37,50 |
| 11 | Programiranje zasnovano na događajima | PF | 38,55,63 |
| 12 | Objektno orijentisana paradigma | PF | 23,58,60,61,62,67 |
| 13 | Sigurno programiranje | PF | 19,47,55 |
| 14 | Osnovna analiza algoritama | AL | 32,36,48,52,53 |
| 15 | Algoritamske strategije | AL | 20, 32,34,35,39,50 |
| 16 | Osnovni algoritmi | AL | 32,35,36,48,49,50 |
| 17 | Digitalna logika i reprezentacija podataka | AR | 2,3,7,8,9,10,11,12,13,14 |
| 18 | Računarska arhitektura i organizacija | AR | 1,3,4,6,40,41,42,43,44 |
| 19 | Interfejsi i U/I strategije | AR | 1,30,46 |
| 20 | Arhitektura memorije | AR | 1,3,44,45 |
| 21 | Funkcionalna organizacija | AR | 1,3,4,42,43 |
| 22 | Uređaji | AR | 2,14,24 |
| 23 | Osnovni pregled operativnih sistema | OS | 1,45,46 |
| 24 | Mrežne komunikacije | NC | 5,74,75,76 |
| 25 | Pregled programskih jezika | PL | 15,16,58,59,60,61,62 |
| 26 | Osnove prevođenja jezika | PL | 15,16,19,20,68,69 |
| 27 | Deklaracije i tipovi | PL | 17,18,20,58 |
| 28 | Mehanizmi apstrakcije | PL | 22,57,58,65 |
| 29 | Objektno orijentisano programiranje | PL | 16,58,60,61,62,63,64,65,67 |
| 30 | Osnove interakcije čovjek-računar | HC | 16,63,66 |
| 31 | Izgradnja GUI interfejsa | HC | 63,64,65,66,67 |
| 32 | Evaluacija korisnički orijentisanih sistema | HC | 47,56,57 |
| 33 | Dizajn GUI | HC | 65,66 |
| 34 | Osnove tehnike grafike i vizuelnog računanja | GV | 14,24 |
| 35 | Informacioni modeli | IM | 60,68 |
| 36 | Jezici upita | IM | 68,69,70,71,72,73 |
| 37 | Istorija računarstva | SP | 2,15,41 |

## 3.3 Smještanje lekcija u tematske cjeline

Predloženo "tijelo znanja" pokriva čitavu oblast računarskih nauka, dok je za potrebe istraživanja i grupisanja lekcija kursa Uvod u računarstvo u odgovarajuće tematske cjeline broj tematskih oblasti redukovan u značajnoj mjeri. Pored uvodnih fundamentalnih tematskih oblasti, uvrštene su uvodne tematske oblasti koje se odnose na informacione sisteme, interakciju čovjeka i računara, računarske mreže i slično. Preostale oblasti, koje se odnose na napredne kurseve iz računarstva, su izostavljene.

Informacije o razmatranim tematskim cjelinama, kao i pregled odgovarajućih lekcija prikazane su u tabeli 3. Prve tri kolone sadrže redni broj, naziv tematske cjeline i skraćeni naziv oblasti znanja kojoj tematska cjelina pripada, dok je u krajnjoj desnoj koloni naveden spisak rednih brojeva pripadajućih lekcija. Redni brojevi lekcija odgovaraju rednim brojevima iz tabele 1. Kao što se vidi iz

Tabela 4: Raspored broja lekcija po tematskim cjelinama

| Broj lekcija | Broj tematskih cjelina sa datim brojem lekcija |
|---|---|
| 2 | 3 |
| 3 | 10 |
| 4 | 8 |
| 5 | 3 |
| 6 | 7 |
| 7 | 1 |
| 8 | 0 |
| 9 | 4 |
| 10 | 1 |

tabele 3, lekcije su svrstavane samo u one tematske cjeline za koje se pouzdano može smatrati da se problematika tematske cjeline obrađuje unutar lekcije, a ne samo usputno pominje. Primjećujemo da su sve fundamentalne tematske cjeline pokrivene većim brojem lekcija, dok se neke druge obrađuju samo u manjem broju lekcija. U tabeli 4 sistematizovana je raspodjela lekcija po tematskim cjelinama. Vidimo da je za 21 tematsku cjelinu broj lekcija u kojima se one obrađuju manji ili jednak od 4, za 11 tematskih cjelina važi da se obrađuju u 5, 6 ili 7 lekcija, dok se 5 tematskih cjelina obrađuje u 9 i više lekcija.

### 3.3.1 Analiza trenutne podjele lekcija i prijedlog poboljšanja

Analizirano je koliko trenutna podjela lekcija odražava polaznu preporuku da što više tematskih cjelina bude obrađeno u oba semestra. Ponovimo da se po trenutnoj raspodjeli, (tabela 1), lekcije sa rednim brojevima 1-39 obrađuju u zimskom semestru, a lekcije 40-76 u ljetnjem. Prema rasporedu lekcija po tematskim cjelinama prikazanom u tabeli 3, od ukupno 37 tematskih cjelina koje su razmatrane, za 30 važi da se one obrađuju u bar po jednoj lekciji u svakom semestru, dok za 7 tematskih cjelina taj uslov ne važi. Primijetimo da algoritam kojim se određuje broj podijeljenih tematskih cjelina odgovara algoritmu verifikacije rješenja odgovarajućeg MSSP. U prve dvije kolone tabele 5 su prikazani redni brojevi i imena tematskih cjelina za koje uslov podijeljenosti ne važi. U narednoj koloni je prikazan skraćeni engleski naziv odgovarajuće oblasti znanja, a u krajnjoj desnoj koloni informacija u kom semestru je cjelina prisutna.

Kao što vidimo iz tabele 5, od sedam "nepodijeljenih cjelina", po jedna se odnosi na arhitekturu računara, te grafiku i vizuelno računanje, dok se tri odnose na interakciju čovjeka i računara i dvije na upravljanje informacijama. Primjećujemo takođe da se oblastima koje se odnose na interakciju čovjeka i računara i upravljanju informacijama pažnja posvećuje u ljetnjem semestru, dok u zimskom semestru ovoj problematici nije posvećena značajnija pažnja. Stoga bi pravac

Tabela 5: Spisak cjelina za koje ne važi traženi uslov

| R.b. | Tematska cjelina | Oblast (eng.) | Prisutna samo u |
|---|---|---|---|
| 22 | Uređaji | AR | zimskom |
| 31 | Izgradnja GUI interfejsa | HC | ljetnjem |
| 32 | Evaluacija korisnički orijentisanih sistema | HC | ljetnjem |
| 33 | Dizajn GUI | HC | ljetnjem |
| 34 | Osnove tehnike grafike i vizuelnog računanja | GV | zimskom |
| 35 | Informacioni modeli | IM | ljetnjem |
| 36 | Jezici upita | IM | ljetnjem |

Tabela 6: Sistematizacija predloženih rješenja

| Postupak | Efekat na tematske cjeline | Posljedica na ukupno rješenje |
|---|---|---|
| Prebacivanje lekcije 24 u ljetnji semestar | Tematske cjeline 22 i 34 će biti podijeljene | Smanjenje broja nepodijeljenih lekcija za 2 |
| Prebacivanje lekcije 56 u ljetnji semestar | Tematska cjelina 32 će biti podijeljena | Smanjenje broja nepodijeljenih lekcija za 1 |
| Prebacivanje lekcije 65 u zimski semestar | Tematske cjeline 31 i 33 će biti podijeljene | Smanjenje broja nepodijeljenih lekcija za 2 |
| Prebacivanje lekcije 68 u zimski semestar | Tematske cjeline 35 i 36 će biti podijeljene | Smanjenje broja nepodijeljenih lekcija za 2 |

unapređenja Plana i programa kursa mogao da se zasniva na uključivanju po jedne lekcije iz grafičkog korisničkog okruženja, te baza podataka u zimskom semestru. Pažljivom analizom trenutnog rasporeda lekcija, može se primijetiti da bi se prebacivanjem polazne lekcije u vezi sa bazama podataka i jedne lekcije u vezi sa interakcijom čovjeka i računara u zimski semestar, mogao riješiti problem "nepodijeljenosti" cjelina 35 i 36, odnosno 31 i 33, a da se ne naruše drugi metodički aspekti, kao što su kontituitet ili zavisnost lekcija.

Takođe, može se zaključiti da bi se razmjenom lekcija 24 i 56 ("Rad sa slikama. Pikseli i boje." odnosno "Testiranje: Funkcionalni test, *unit test*, *black-box test*, *glass-box test*, Nezavisnost, Ograničenja, Testom vođen razvoj programa. Debagiranje.") postigla situacija da su cjeline 22, 32 i 34 podijeljene, a podijeljenost preostalih tematskih cjelina se ne bi narušila.

Sistematizacija predloženih unapređenja prikazana je u tabeli 6.

Ovom preraspodjelom lekcija dobija se stanje da su sve tematske cjeline podi-

jeljene.

# 4 Pregled razvijenih metoda za rješavanje MSSP

U ovom odjeljku dat je pregled metoda rješavanje MSSP problema koje se sreću u literaturi. Pošto je problem NP težak, ne postoji deterministički algoritam koji rješava opšti slučaj problema u polinomnom vremenu. Stoga, egzaktne metode uspijevaju da riješe samo probleme manjih dimenzija, jer vrijeme izvršenja algoritma eksponencijalno raste porastom dimenzije. Za probleme većih dimenzija primjenjuju se heurističke metode. Opisane su dvije formulacije cjelobrojnog programiranja (kvadratna i linearna formulacija), te tri metaheurističke metode, zasnovane na genetskom algoritmu, elektromagnetizmu i metodi promjenljivih okolina.

Prije nego što se navedu metode za rješavanje MSSP, uvedimo sljedeću notaciju. Neka je $S$ konačan skup kardinalnosti $m = |S|$ i neka je $C = \{S_1, S_2, ..., S_n\}$ kolekcija podskupova od $S$. Bez gubljenja opštosti, možemo pretpostaviti da je $S = \{1, 2, ..., m\}$. Neka je $(P_1, P_2)$ particija od $S$. Za skup $S_j \in C$ kažemo da je "podijeljen", ako $S_j$ ima neprazan presjek i sa $P_1$ i sa $P_2$.

## 4.1 Kvadratna cjelobrojna formulacija

O ovom pododjeljku prikazana je kvadratna cjelobrojna formulacija iz [5].

Kvadratna cjelobrojna formulacija (1)-(4) uvodi dva skupa promjenljivih:

$$y_i = \begin{cases} 1, & i \in P_1 \\ -1, & i \in P_2 \end{cases}, \quad \text{za } i = 1, ..., m,$$

i

$$z_j = \begin{cases} 1, & S_j \text{ je podijeljen} \\ 0, & S_j \text{ nije podijeljen} \end{cases}, \quad \text{za } j = 1, ..., n.$$

QIP se definiše kao

$$\max \sum_{j=1}^{n} z_j \tag{1}$$

uz ograničenja

$$\frac{1}{|S_j| - 1} \sum_{\substack{i_1, i_2 \in S_j \\ i_1 \neq i_2}} \frac{1 - y_{i_1} \cdot y_{i_2}}{2} \geq z_j, \quad \text{za sve } S_j \in C \tag{2}$$

$$z_j \in \{0, 1\}, \quad \text{za } j = 1, ..., n \tag{3}$$

$$y_i \in \{-1, 1\}, \quad \text{za } i = 1, ..., m \tag{4}$$

## 4.2 Linearna cjelobrojna formulacija

Ovaj pododjeljak sadrži opis linearne cjelobrojne formulacije, prikazane u [6].

Za ILP uvodimo parametre

$$s_{ij} = \begin{cases} 1 & i \in S_j \\ 0 & i \notin S_j \end{cases} \qquad \text{za } i = 1, ..., m,\ j = 1, ..., n,$$

koji označavaju da li element $i$ iz $S$ pripada podskupu $S_j$. Promjenljive se definišu kao

$$x_i = \begin{cases} 1 & i \in P_1 \\ 0 & i \in P_2 \end{cases}$$

$$y_j = \begin{cases} 1 & S_j \text{ je podijeljen} \\ 0 & S_j \text{ nije podijeljen} \end{cases}$$

ILP model za MSSP se formuliše kao

$$\max \sum_{j=1}^{n} y_j \tag{5}$$

uz ograničenja

$$y_j \leq \sum_{i=1}^{m} s_{ij} \cdot x_i, \qquad \text{za sve } j = 1, ..., n \tag{6}$$

$$y_j + \sum_{i=1}^{m} s_{ij} \cdot x_i \leq |S_j| \ , \qquad \text{za sve} j = 1, ..., n \tag{7}$$

$$y_j \in \{0, 1\} \ , \qquad \text{za sve } j = 1, ..., n \tag{8}$$

$$x_i \in \{0, 1\} \ , \qquad \text{za sve } i = 1, ..., m \tag{9}$$

Lako se vidi da ILP formulacija raspolaže sa $m+n$ binarnih promjenljivih i ukupno $2 \cdot n$ ograničenja.

### 4.2.1 Genetski algoritam za rješavanje MSSP

Genetski algoritam (GA) za rješavanje MSSP prikazan je zajedno sa cjelobrojnom linearnom formulacijom u [6].

GA spadaju u klasu evolucijskih algoritama, čije ponašanje potiče iz procesa evolucije koji se odvija u prirodi. U osnovi se podrazumijeva rad sa populacijom jedinki, gdje svaka jedinka predstavlja potencijalno rješenje, a svaka populacija je podskup ukupnog prostora pretraživanja. Populacija se u iterativnom postupku mijenja tako što se stare jedinke mijenjaju novim, potencijalno boljim.

GA za rješavanje MSSP koristi binarno kodiranje, dok je genetski kôd one dužine koliko ima elemenata skupa. Vrijednost 1 na $i$-tom mjestu genetskog kôda označava da odgovarajući element $i$ pripada skupu $P_1$, dok vrijednost 0 označava

da dati element pripada skupu $P_2$. Za datu jedinku, funkcija cilja ima vrijednost broja podijeljenih skupova kolekcije $C$. Fitnes jedinke $f_{ind}$ se računa skaliranjem vrijednosti funkcije cilja svih jedinki (kojih u populaciji ima $N_{pop}$ u interval [0,1]. Tako najbolja jedinka $ind_{max}$ ima fitnes jednak 1, dok je vrijednost fitnesa najmanje jedinke ($ind_{min}$) jednak nuli. Preciznije,

$$f_{ind} = \frac{obj_{ind} - obj_{ind_{min}}}{obj_{ind_{max}} - obj_{ind_{min}}}$$

GA implementira standardni genetski operator jednopozicionog ukrštanja, dok za operator selekcije koristi unaprijeđenu turnirsku selekciju, koja omogućava da prosječna veličina turnira bude proizvoljan realan (racionalan) broj. Standardni operator mutacije je unaprijeđen konceptom prepoznavanja tzv. "smrznutih gena", tj. pozicija u genetskom kodu na kojima sve jedinke populacije imaju istu vrijednost. Pojava smrznutih gena uzrokuje drastično smanjenje prostora pretrage, što generalno povećava vjerovatnoću za preuranjenu konvergenciju, dok operatorima selekcije i ukrštanja "smrznuti geni" ne mogu promijeniti vrijednost. U slučaju kodiranja GA za MSSP "smrznuti gen" ukazuje da je dati element (koji odgovara tom genu) u svim jedinkama populacije dodijeljen istom skupu ($P_1$ ili $P_2$). Mehanizam kojim se ova pojava eliminiše i koji je implementiran u dati GA, zasnovan je na značajnom povećanju vjerovatnoće za mutaciju "smrznutih gena". Stoga se za "smrznute gene" osnovni nivo mutacije množi odgovarajućim brojem, faktorom mutacije za smrznute gene.

### 4.3 Elektromagnetizam za rješavanje MSSP

Elektromagnetizam (EM) za rješavanje MSSP prikazan je u [7]. EM je metaheuristička metoda koja koristi mehanizam privlačenja - odbijanja čestica, kako bi se realizovalo kretanje i eventualno dostizanje optimalnog rješenja. Svaka tačka (čestica) se smatra potencijalnim rješenjem, kome je dodijeljen neki nivo naelektrisanja. Bolje čestice imaju jaču silu privlačenja drugih čestica. Tačna vrijednost uticaja jedne čestice na drugu definisana je Kulonovim zakonom. To znači da je sila između dvije tačke proporcionalna proizvodu naelektrisanja i obrnuto proporcionalna rastojanju između njih. Drugim riječima, tačke sa višim naelektrisanjem će više uticati na kretanje drugih čestica. Pored toga, najbolja EM tačka će ostati nepromijenjena. Naelektrisanja tačaka se vezuju za vrijednost funkcije cilja, koja je predmet optimizacije.

EM za rješavanje MSSP koristi hibridni pristup. Kombinuje se osnovni mehanizam "pomjeranja čestica", zasnovan na principu privlačanja i odbijanja čestica, zajedno sa specifično osmišljenom tehnikom skaliranja, koja usmjerava čitav EM u regione pretraživanja za koje se pretpostavljada sadrže kvalitetna rješenja.

Ako je $m$ ukupan broj elemenata skupa, svaka tačka $p_i$ EM algoritma se kodira nizom od $m$ elemenata, gdje su elementi iz intervala $[0, 1]$. Da bi se za svaku tačku $p_j$ odredila funkcija cilja, prvo se odgovarajuća particija $(P_1, P_2)$ odredi zaokruživanjem na sljedeći način: ako je $i$-ta koordinata tačke $p_j$ veća ili jednaka

0.5, tada se element $i$ pridružuje $P_1$, a u protivnom $P_2$. Uvedimo promjenljivu $y$ definisanu na sljedeći način:

$$y_i = \begin{cases} 1, & p_{ji} \geq 0.5 \\ 0, & p_{ji} < 0.5 \end{cases}$$

Ako uvedemo i promjenljivu $z$ definisanu na sljedeći način:

$$z_j = \begin{cases} 1, & S_j \text{ je podijeljen} \\ 0, & S_j \text{ nije podijeljen} \end{cases}$$

tada se vrijednost funkcije cilja definiše kao

$$obj(P_1, P_2) = \sum_{j=1}^{n} z_j. \tag{10}$$

Brzo lokalno pretraživanje dodatno unapređuje cjelokupnu pretragu, omogućavajući pronalaženje boljeg rješenja u nekoj okolini trenutnog. Zasnovano je na zamjeni komponenti parova elemenata skupa i koristi strategiju "primjene prvog poboljšanja", tj. strategiju koja podrazumijeva da se otkriveno unapređenje odmah primjenjuje na trenutno rješenje.

Mehanizam privlačenja-odbijanja omogućava kretanje EM tačaka po prostoru pretrage, tako što se pretraga usmjerava ka potencijalno boljim oblastima, gdje se nalaze tačke sa boljim vrijednostima funkcije cilja. U ovom procesu pomjeranja, tačke EM se posmatraju kao naelektrisane čestice. Vrijednost naelektrisanja je u direktnoj vezi sa vrijednošću funkcije cilja i računa se pomoću formule (11).

$$q_i = exp\left( -N_{pop} \frac{f(p_{best}) - f(p^i)}{\sum_{k=1}^{N_{pop}} f(p_{best}) - f(p^k)} \right), \ i = 1, ..., N_{pop} \tag{11}$$

gdje je $N_{pop}$ ukupan broj EM tačaka. Sila između parova tačaka se računa po sličnom principu kao u teoriji elektromagnetizma za naelektrisane čestice. Sila između dvije čestice je obrnuto propocionalna udaljenosti čestica, a direktno proizvodu njihovih naelektrisanja. Tačke sa boljim vrijednostima funkcije cilja privlače druge tačke, dok one sa lošijim odbijaju ostale. Vrijednost sile privlačenja/odbijanja prikazana je formulom (12):

$$F_i = \begin{cases} \sum_{j=1, j\neq i}^{m} (p_j - p_i) \frac{q_i \times q_j}{||p_j - p_i||^2}; \ f(p_j) > f(p_i) \\ \sum_{j=1, j\neq i}^{m} (p_i - p_j) \frac{q_i \times q_j}{||p_j - p_i||^2}; \ f(p_j) \leq f(p_i) \end{cases} \tag{12}$$

Nakon što se izračunaju sile privlačenja/odbijanja, sve EM tačke osim najbolje se kreću u odgovarajućim smjerovima. Najbolja tačka se ne pomjera. U EM algoritmu, pomjeranjem tačaka se upravlja unutar procedure Pomjeri. Pravac i smjer vektora u kom pravcu se tačka $p_i$ pomjera je određen silom $F_i$. Sve sile se najprije normalizuju ($F_i = \frac{F_i}{\|F_i\|}$). Kretanje tačaka je dalje definisano pomoću formule 13, gdje je $\beta$ slučajan broj iz intervala $[0, 1]$.

$$p_i^k = \begin{cases} p_i^k + \beta \frac{F_i^k}{||F_i||}(1 - p_i^k), & F_i^k > 0 \\ p_i^k + \beta \frac{F_i^k}{||F_i||} \cdot p_i^k, & F_i^k < 0 \end{cases} \tag{13}$$

### 4.4 Metoda promjenljivih okolina za rješavanje MSSP

Metoda promjenljivih okolina (eng. Variable Neighborhood Search - VNS) je metaheuristička metoda koja podrazumijeva sistematsku promjenu okolina, kako bi se izbjegle situacije kada algoritam "upada" u suboptimalna rješenja. Efikasnost VNS algoritma je zasnovana na razmatranju da u mnogim praktičnim optimizacionim problemima postoji veza između lokalnih minimuma. Stoga metod promjenljivih okolina i ne prati unaprijed zadatu putanju, već "skače" na potencijalno bolja rješenja koja se nalaze u nekoj okolini trenutno najboljeg. Na ovaj se način očuvavaju dobre osobine trenutnog rješenja (ako su neke promjenljive već dostigle optimalne vrijednosti), dok se u novim okolinama pokušavaju poboljšati preostale promjenljive. Ovaj sistem je dodatno ojačan sistemom lokalnog pretraživanja, kojim se prelazi iz trenutnog rješenja u najkvalitetnije rješenje u nekoj njegovoj okolini. Metoda promjenljivih okolina za rješavanje MSSP prikazan je u [8].

Kodiranje rješenja je slično kao i kodiranje jedinki GA opisanog u [6]. Rješenje je predstavljeno binarnim nizom dužine koliko ima elemenata skupa. Vrijednost 1 na $i$-tom mjestu niza označava da odgovarajući element $i$ pripada skupu $P_1$, dok vrijednost 0 označava da dati element pripada skupu $P_2$. Polazno rješenje se bira na slučajan način.

Za dato rješenje, funkcija cilja ima vrijednost broja podijeljenih skupova kolekcije $C$. Na osnovu kodarješenja, lako se određuje kojoj komponenti pripada koji element. Nakon toga se, slično kao u EM algoritmu, za datu particiju $(P_1, P_2)$, i za svaki element (podskup) $S_j \in C$, određuje vrijednost $z_j$

$$z_j = \begin{cases} 1, & S_j \text{ je podijeljen} \\ 0, & S_j \text{ nije podijeljen} \end{cases}$$

a zatim i funkcija cilja

$$obj(P_1, P_2) = \sum_{j=1}^{n} z_j. \tag{14}$$

Određivanje novih okolina, kao i predlaganje novih potencijalno boljih rješenja iz tih okolina se odvija u proceduri razmrdavanja (eng. shaking). U okviru date

procedure razmrdavanja, sistem okolina se formira na sljedeći način. Za definisanje $k$-te okoline, u algoritmu se na slučajan način bira nekih $k$ elemenata iz $S$. Svakom izabranom elementu mijenja se pripadajuća komponenta, tj. svi izabrani elementi koji pripadaju $P_1$ se prebacuju u $P_2$ i obrnuto. Formalno, $k$-ta okolina vektora (rješenja) $x$ se može zapisati kao $N_k(x) = \{x' : \{i_1, i_2, ..., i_k\} \subset \{1, 2, ..., |S|\}\ x'_{i_j} = 1 - x_{i_j}\}$. U algoritmu se tako unutar procedure razmrdavanja za svaki cio broj $k$, $k \in [k_{min}, k_{max}]$ predlaže novo rješenje $x'$. ($x' \in N_k(x)$), na koje se dalje poziva lokalno pretraživanje.

U svakoj iteraciji lokalne pretrage algoritam pokušava da popravi rješenje tako što mijenja pripadajuće komponente za parove elemenata iz $S$. Na primjer, ako $a \in P_1$ i $b \in P_2$, tada nakon razmjene imamo $a \in P_2$ i $b \in P_1$. U slučaju da je tako dobijeno novo rješenje $x''$ bolje od polaznog rješenja $x'$ iz date okoline tačke $x$, tada $x'$ postaje jednako $x''$. Lokalna pretraga prestaje sa radom nakon prvog takvog poboljšanja. U drugom slučaju, ako nije došlo do poboljšanja ($x''$ nije bolje od $x'$), lokalna pretraga nastavlja sa sljedećim parom elemenata. Po završetku lokalne pretrage, porede se vrijednosti funkcije cilja za $x$ i $x'$ i u zavisnosti koje je bolje, ono se smatra trenutno najboljim rješenjem. U slučaju da su vrijednosti funkcije cilja za $x$ i $x'$ jednake, tada algoritam prelazi u rješenje $x'$ sa vjerovatnoćom 0.4, a u protivnom ostaje u rješenju $x$.

Eksperimentalni rezultati koji prikazuju kvalitet i upotrebljivost opisanih metoda za rješavanje MSSP pokazuju da date metode uspijevaju da pronađu kvalitetna rješenja u razumnom vremenu. To ukazuje na činjenicu da se opisane metode mogu koristiti i za rješavanje konkretnih problema koji se odnose na podjelu kurseva po opisanom kriterijumu.

# 5 Zaključak

U ovom radu je prikazana mogućnost unapređenja planova i programa primjenom poznatog matematičkog problema maskimalne podjele skupa. Ako razmatramo metodološki opravdan pristup podjele kursa na dva dijela (na primjer na dva semestra), na način da što više tematskih cjelina bude prisutno u oba dijela, pokazuje se da je matematički model koji odgovara ovom pristupu identičan postavci poznatog matematičkog problema MSSP. Kao ilustraciju upotrebe MSSP u rješavanju problema u nastavi, prikazana je podjela kursa Uvod u računarstvo i analizirana trenutno važeća podjela. Analizom se došlo do saznanja da trenutna podjela kursa u značajnoj mjeri "prati" propisani pristup, dok se minornim razmjenama lekcija između semestara, taj raspored može učiniti i optimalnim. Ova činjenica ukazuje na pretpostavku da su autori Plana i programa, motivišući se ovim ili nekim sličnim pristupom, u značajnoj mjeri vodili računa da se fundamentalne cjeline razmatraju u oba semestra, kako bi se očuvao kontinuitet u izučavanju.

Pošto je odgovarajući matematički problem NP težak, egzaktne metode se mogu koristiti samo pri rješavanju problema raspoređivanja lekcija u semestre

koji su manjih dimenzija, dok se za probleme većih dimenzija koriste približne metode. S obzirom na tu činjenicu, prikazani pregled i opis nekih egzaktnih i nekih približnih metoda za rješavanje ovog problema koje mogu pomoći odgovornim osobama prilikom kreiranja planova i programa.

## Literatura


[1] Rutherford, F.J. and Ahlgren, A. "Science for All Americans". Oxford University Press, USA. 1991.

[2] D. Matić and M. Božić. "Maximally Balanced Connected Partition Problem in Graphs: Application in Education", The Teaching of Mathematics, 15(2), p. 121–132, 2012.

[3] L. Lovász. "Coverings and colorings of hypergraphs", Proc. 4th Southeastern Conf. on Comb. Springer Berlin Heidelberg. p. 3–12, 1973.

[4] "Computer Science Curriculum 2008: An Interim Revision of CS 2001 Report from the Interim Review Task Force". 2008. http://www.acm.org/education/curricula/ComputerScience2008.pdf

[5] G. Andersson and L. Engebretsen. "Better approximation algorithms for Set Splitting and Not-All-Equal Sat", Information Processing Letters, 65(6), p. 305 – 311, 1998.

[6] B. Lazović, M. Marić,V. Filipović, A. Savić. "An integer linear programming formulation and genetic algorithm for the maximum set splitting problem", Publications de l'Institut Mathematique, 92(106) p. 25–34, 2012.

[7] J. Kratica. "An Electromagnetism-like method for the maximum set splitting problem", Yugoslav Journal of Operations Research, 23(1), p. 31–41, 2013.

[8] Matić, D. "A Variable Neighborhood Search Approach for Solving the Maximum Set Splitting Problem", Serdica Journal of Computing, 6(4) p. 369–384, 2012.