UDC: 519.852:519.83 INFO M: str. 4-10

PROGRAMIRANJE OGRANIČENJA

CONSTRAINT PROGRAMMING[1]

Mirko Vujošević
Fakultet organizacionih nauka Univerziteta u Beogradu, mirkov@fon.bg.ac.rs

REZIME: Programiranje ograničenja (PO) je oblast koja se bavi izučavanjem i rešavanjem upravljačkih problema koji su predstavljeni sistemom ograničenja nad promenljivima odluke. Osnovni cilj je razviti veštine modeliranja realnih problema i njihovog predstavljanja sistemom ograničenja, a zatim pronaći metode i kodirati algoritme kojima se nalaze vrednosti upravljačkih promenljivih koje zadovoljavaju sva ograničenja. Po prirodi problema koji se razmatraju i još više po metodologiji koja se koristi za rešavanje problema, programiranje ograničenja i matematičko programiranje su veoma bliske oblasti. Suštinska razlika je u tome što se u zadacima PO ne traži, odnosno ne mora da se traži rešenje koje je najbolje prema izabranom kriterijumu. Razlog je u činjenici da se u nekim realnim problemima veoma teško dolazi do rešenja koje bi zadovoljilo sva ograničenja. Tada je sa praktične tačke gledišta sasvim zadovoljavajuće naći bilo koje rešenje koje odgovara postavljenim zahtevima. U ovom radu se daje pregled osnovnih pristupa rešavanju problema PO.
KLJUČNE REČI: ograničenje, programiranje, logika, veštačka inteligencija, optimizacija
ABSTRACT: The scope of this survey paper is restricted to basic notions, concepts and fundamental approaches to solving constraint programming problems. The accent is put on constraint satisfaction problem. Three main methodological approaches, generate and test, consistency techniques, and constraint propagation, are briefly described as well as the use of constraint graph. Improvements schemes, global constraints and over-constrained problems are also discussed. At the end, some contemporary research and technological challenges are given.
KEY WORDS: constraint, programming, logic, artificial intelligence, optimization

[1] Ovo je verzija rada saopštenog na XXXIX SYMOPIS, Tara 25-29. septembar 2012. Rad je rezultat istraživanja na projektima 33044 i 35045 koje finansira Ministarstvo prosvete i nauke Republike Srbije

1. UVOD

Programiranje ograničenja je savremeni pristup rešavanju teških kombinatornih problema. Opšti problem PO predstavlja se sistemom ograničenja nad upravljačkim (nepoznatim) promenljivima. Zadatak je naći dopustivo rešenje, odnosno odrediti vrednosti promenljivih koje zadovoljavaju sva postavljena ograničenja, ili dokazati da ono ne postoji [1,2,3].

Sa aspekta naučnog istraživanja, PO je multidisciplinarna oblast u kojoj se kombinuju metode i tehnike iz računarskih nauka, veštačke inteligencije, operacionih istraživanja, baza podataka, teorije grafova i logičkog programiranja. Sa aspekta prakse i primena, PO je softverska tehnologija za deklarativno opisivanje i efektivno rešavanje velikih, prvenstveno kombinatornih problema. Osnovna ideja u PO je da korisnik najpre postavi svoj problem pomoću ograničenja, a zatim pronađe rešenje korišćenjem nekog opštenamenskog *rešavača ograničenja* (solvera). Određujući pojmovi u oblasti PO su ograničenje i programiranje.

2. OGRANIČENJE

Ograničenja prirodno, razumljivo i jasno formulišu međusobne zavisnosti u fizičkom svetu i njegovoj matematičkoj apstrakciji. Ograničenjima se predstavljaju veze koje moraju da postoje između vrednosti promenljivih koje su određene svojim domenima. Ograničenje predstavlja važnu informaciju o upravljačkim promenljivima jer svojim postojanjem smanjuje broj mogućih vrednosti koje one mogu uzeti. Ograničenje je, po pravilu, samo delimična informacija o vrednostima promenljivih; u suprotnom, potpuna informacija bi značila određivanje tačne vrednosti za promenljivu.

Bitno je uočiti sledeće važne osobine ograničenja:

- Ograničenja pojedinačno specificiraju samo delimične, parcijalne, informacije, tj. ograničenje nije jednoznačno određivanje vrednosti promenljive; (za neke klase problema specifikacije izvesnih promenljivih nazivaju se ograničenjima jer se samo po njima razlikuju konkretni primerci);
- Ograničenja su neusmerena; ograničenje nad dve promenljive, recimo X i Y , može biti korišćeno da se izvede zaključak o ograničenjima na vrednosti promenljive X ako su poznata ograničenja na vrednosti promenljive Y i obratno;
- Ograničenja su deklarativna; ograničenja ukazuju koja relacija mora da bude ostvarena bez specificiranja računarske procedure koja sprovodi tu relaciju:
- Ograničenja su aditivna; redosled postavljanja ograničenja nije bitan, bitno je da na kraju važe sva ograničenja, odnosno da se ostvari konjunkcija ograničenja;
- Ograničenja su retko međusobno nezavisna, obično međusobno dele promenljive.

3. PROGRAMIRANJE

U sintagmi PO, termin programiranje se odnosi prevashodno na tehniku računarskog programiranja, sa značenjem

u duhu ostalih paradigmi programiranja kao što su objektno-orijentisano programiranje, funkcionalno programiranje, konkurentno programiranje, strukturno programiranje itd.

S druge strane, termin programiranje u sintagmi *matematičko programiranje* potiče iz naslova čuvenog Dancigovog rada „*Programiranje u linearnoj strukturi*" [4] i ima značenje bliže pojmu planiranje u srpskom jeziku. Programiranje u tom kontekstu odgovara procesu nalaženja (optimalnog) plana za izvršavanje određenog posla.

Računarski program pomoću koga se rešava problem definisan ograničenjima zove se rešavač ili solver ograničenja. Rešavač ograničenja može da se posmatra kao procedura koja iterativno transformiše problem ograničenja u jednostavnije ekvivalentne probleme da bi se na kraju došlo do odgovora na pitanje da li postoji rešenje koje zadovoljava ograničenja i ako postoji, koje je. U ovom kontekstu se smatra da su dva problema ekvivalentna ako imaju istovetna rešenja.

Softverski sistem za PO se razvija na bazi dvoslojne arhitekture koja integriše komponentu ograničenja i komponentu programiranja [2,3,6]. Prvi nivo omogućava korisnicima da postave ograničenja za programske promenljive. Komponenta ograničenja se obično naziva *skladište ograničenja*, po analogiji sa memorijskim skladištem u tradicionalnim programskim jezicima. Drugi nivo arhitekture omogućava korisnicima da napišu računarski program koji određuje kako promenljive mogu biti modifikovane u procesu traženja vrednosti koje bi zadovoljile ograničenja.

PO se naziva i *logičko programiranje ograničenja*. Logičko programiranje je deklarativni, relacioni stil programiranja baziran na logici prvog reda. Analiza nastanka i razvoja PO pokazuje da su istraživanja koja su prethodila PO bila u okvirima logičkog programiranja koje će se kasnije posmatrati kao posebna vrsta PO. Osnovna ideja koja stoji iza PO i deklarativnog programiranja uopšte, jeste da korisnik navodi šta treba da bude rešeno, što se odnosi na ograničenja, umesto toga kako treba da bude rešeno, što se odnosi na programiranje. Metode deklaracije ograničenja i neki algoritmi za njihovo rešavanje integrisani su u standardne imperativne jezike kao što su C++ i Java.

4. MODELIRANJE PROBLEMA OGRANIČENJA

Osnovni okviri u kojima se pristupa rešavanju problema PO i problema matematičkog programiranja (optimizacije) u suštini su isti i poklapaju se sa metodološkim pristupom operacionih istraživanja [7]. Ukratko, u centru je matematički model i eksperimentisanje na njemu pomoću računara. Razmatrani realni problem treba najpre detaljno opisati i na osnovu opisa razviti matematički model pogodan za predstavljanje i eksperimentisanje na računaru. Radi rešavanja postavljenih problema treba proučiti i razviti efikasne algoritme koji su prilagođeni osobinama problema. Zatim je potrebno programirati, odnosno treba kodirati i implementirati na računaru prethodno razvijene matematičke modele i odgovarajuće algoritme. Na kraju se rade eksperimenti na modelu korišćenjem razvijenog programa.

5. PROBLEM ZADOVOLJENJA OGRANIČENJA (CSP)

Prema tipu problema, a sledstveno tome i prema pristupima rešavanju, razlikuju se dva pravca istraživanja u oblasti PO i to: zadovoljenje ograničenja i rešavanje ograničenja.

Problem *zadovoljenja ograničenja* (*Constraint Satisfaction Problem – CSP*) je određen:

- Konačnim skupom promenljivih,
- Funkcijom koja preslikava svaku promenljivu na neki konačan domen,
- Konačnim skupom ograničenja.

Rešenje ovog problema je dodela, pridruživanje, svakoj promenljivoj vrednosti iz njenog domena tako da su sva ograničenja zadovoljena. Zadatak je pronaći jedno ili sva takva rešenja ili utvrditi da su ograničenja protivurečna.

Problem *rešavanja ograničenja* se razlikuje u odnosu na problem zadovoljenja ograničenja po tome što u njemu postoje promenljive čiji domeni nisu konačni. Pored toga, pojedinačna ograničenja mogu da budu složenija, npr. nelinearne jednačine tako da se, pored metoda kombinacija i pretraživanja, ili umesto njih, koriste algebarske i metode numeričke analize. Postoje pristupi da se beskonačni domeni upravljačkih promenljivih diskretizuju i ograniče i na taj način se problem rešavanja ograničenja prevede u problem zadovoljenja ograničenja. Dalja razmatranja u ovom tekstu odnose se isključivo na probleme zadovoljenja ograničenja.

6. POSTAVKA PROBLEMA ZADOVOLJENJA OGRANIČENJA

Formulaciji problema zadovoljenja ograničenja je istorijski prethodila postavka problema zadovoljivosti ili problema bulovske zadovoljivosti, poznatog u literaturi po skraćenici SAT (*Satisfiability Problem*). Radi se o problemu matematičke logike u kojem se posmatraju logičke promenljive i logička formula u konjuktivnoj normalnoj formi. Problem je odgovoriti na pitanje da li je formula zadovoljiva, tj. da li se promenljivima u posmatranoj logičkoj formuli mogu dodeliti vrednosti tako da formula dobije vrednost TAČNO. Ako je to moguće, onda je formula zadovoljiva. U suprotnom, ako takvo dodeljivanje ne postoji, onda je funkcija izražena tom formulom identična sa NETAČNO za sve vrednosti koje mogu biti dodeljene promenljivima, odnosno, formula je nezadovoljiva. Zanimljivo je da je složenost ovog problema takva da je on *NP*-kompletan i da je i koncept *NP*-kompletnosti definisan upravo pri njegovom proučavanju.

Problem zadovoljenja ograničenja (CSP) se definiše na sledeći način. Dato je:

- skup promenljivih $X=\{x_1,...,x_n\}$,
- za svaku promenljivu x_j, $j=1,...,n$, konačni skup D_j njenih mogućih vrednosti, tj. domen promenljive x_j ,

- skup ograničenja $C=\{C_1,...,C_m\}$ koji sužava skup vrednosti koje promenljive mogu istovremeno da uzmu; svako ograničenje C_i definisano je na podskupu k promenljivih $var(C_i)=\{x_{i1},...,x_{ik}\}$ skupa X i podskupu $rel(C_i)$ Dekartovog proizvoda $D_{i1}\times...\times D_{ik}$; ograničenje označava kombinaciju dozvoljenih vrednosti promenljivih $\{x_{i1},...,x_{ik}\}$;

Može se postaviti neki od sledećih zadataka:

- Pronaći jedno, bilo koje rešenje koje zadovoljava ograničenja,
- Pronaći sva rešenja koja zadovoljavaju ograničenja,
- Pronaći optimalno ili što bolje rešenje pri čemu je data kriterijumska funkcija definisana pomoću nekih ili svih promenljivih.

Poslednji zadatak je, u stvari, klasični optimizacioni problem postavljen kao problem zadovoljenja ograničenja pri čemu se definicija optimalnog rešenja koristi kao jedno od ograničenja u problemu.

Treba zapaziti da vrednosti promenljivih, prema opštoj postavci, ne moraju da budu u skupu celih brojeva (iako se najčešće matematički modeli razvijaju na toj osnovi), pa čak ne moraju biti ni numerički. Dakle, u opštem slučaju, promenljive mogu biti:

- Bulove (binarne), celobrojne, simboličke, skup elemenata i podskupovi skupova.

Ograničenja mogu biti:

- Matematička, disjunktivna, relaciona, eksplicitna ili posebne vrste prema tipu zadatka (kao i kombinacija navedenih tipova).

Operatori u ograničenjima mogu biti:

- $=, <, >, \leq, \geq, \neq, \subset, \subseteq, \supset, \supseteq, \in, \cup, \cap, \wedge, \vee, \Rightarrow, \Leftrightarrow$.

Dopustivo rešenje problema CSP je :

- Dodela svakoj promenljivoj neke vrednosti iz njenog domena tako da su sva ograničenja u problemu zadovoljena.

Ako se ograničenje odnosi na jednu promenljivu, naziva se unarno. Kad ograničenje obuhvata dve promenljive, naziva se binarno. Ograničenje može biti *n*-arno kada uključuje n promenljivih.

7. GRAF OGRANIČENJA

CSP se može pogodno predstaviti grafom ograničenja ako su sva ograničenja unarna i/ili binarna. Čvorovi grafa predstavljaju promenljive problema i pridružuju im se domeni vrednosti promenljivih, unarna ograničenja. Grane ili lukovi grafa predstavljaju relacije koje postoje između promenljivih, binarna ograničenja.

Dokazano je da se proizvoljni CSP moze transformisati u ekvivalentni CSP sa unarnim i binarnim ograničenjima. Međutim, ova transformacija često postaje računski veoma zahtevna i neopravdano skupa.

8. Osnovni pristup rešavanju problema CSP

Kao što je navedeno, problem CSP ima sledeće dve bitne osobine:

a) Postoje ograničenja između promenljivih koja sužavaju domene svake od njih a time i skup potencijalnih rešenja;
b) Skup potencijalnih rešenja, prostor pretraživanja, je konačan.

Na osnovu ovih osobina razvijeni su sledeći pristupi rešavanju problema:

a) Pristup proveravanja saglasnosti ograničenja za tekuće vrednosti promenljivih; u ovom pristupu se korišćenjem ograničenja postupno sužava skup potencijalnih rešenja i, ako je moguće, pronalazi traženo rešenje;
b) Pristup pretraživanja skupa mogućih rešenja;
c) Pristup naizmeničnog korišćenja prethodna dva pristupa koji je, u stvari, opšti pristup rešavanju problema PO i obično se naziva pristup prostiranja ograničenja.

Pored navedena tri opšta pristupa, za posebne klase problema razvijaju se i primenjuju i drugi pristupi prilagođeni specifičnim karakteristikama problema.

9. GENERIŠI I TESTIRAJ

Sve metode pretraživanja počivaju na principu generiši i testiraj (GT), s tim što osnovni algoritam ispituje kompletan prostor vrednosti promenljivih. Ideja osnovnog algoritma tipa GT je jednostavna: generišu se konkretne vrednosti upravljačkih promenljivih po pravilima koja obezbeđuju ispitivanje svakog elementa u prostoru potencijalnih rešenja, a onda proveravaju ograničenja; ako generisane vrednosti zadovoljavaju sva ograničenja, rešenje je pronađeno; u suprotnom generišu se sledeće vrednosti promenljivih.

Očigledno je da osnovni algoritam potpunog nabrajanja koji proizilazi iz ideje GT nije efikasan. Njegova neefikasnost je posledica neinformisanosti generatora o šansi da generisano nagađanje bude i konačno rešenje problema. To, po pravilu, uzrokuje relativno kasno otkrivanje nesaglasnosti u vrednostima promenljivih. Otklanjanjem ovog uzroka u što je moguće većem obimu povećava se efikasnost postupka pretraživanja.

U okviru PO razvijeno je više opštih algoritama za sistematsko pretraživanje koji su značajno efikasniji od potpunog nabrajanja. Skoro svi se oslanjaju na metodu vraćanja po tragu (*backtracking*) i pristup pretraživanja po dubini (*depth first principle*). U cilju povećanja efikasnosti i ovde se metoda vraćanja po tragu modifikuje na osnovu specifičnih karakteristika razmatranog problema.

10. VRAĆANJE PO TRAGU

Metoda vraćanja po tragu za rešavanje CSP je konstruktivna metoda u kojoj se polazi od vrednosti za jednu promenljivu, zatim se specificiraju vrednosti za dve promenljive, onda za tri i tako nastavlja sve dok je moguće jer se posle svake specifikacije proverava lokalna saglasnost, odnosno sagla-

snost vrednosti razmatranih promenljivih. Saglasne vrednosti razmatranih promenljivih zovu se parcijalno rešenje. Ono se pokušava proširivati ka konačnom rešenju koje bi trebalo da obuhvati sve promenljive.

Metoda vraćanja po tragu je spoj faze generisanja i faze testiranja. Promenljive su označene sekvencijalno i čim se odrede vrednosti za neke promenljive, proveravaju se ograničenja koja među njima postoje. Ako parcijalno rešenje prekrši bilo koje ograničenje, po tragu se vraća na najbližu prethodno razmatranu promenljivu koja još pruža mogućnost izbora. Kad god se ustanovi da parcijalno rešenje ne zadovoljava neko ograničenje, iz daljeg razmatranja se eliminiše deo ukupnog prostora pretraživanja. Dakle, vraćanje po tragu je efikasnije od osnovnog algoritma GT ali je njegova složenost za većinu problema još uvek eksponencijalna funkcija dimenzije problema.

Postoje tri glavna nedostatka standardnog (hronološkog) algoritma vraćanja po tragu:

- Ponavljanje iste greške;
- Redundantnost u proveravanju saglasnosti, tj. konfliktne vrednosti promenljivih se ne pamte;
- Kasno otkrivanje nesaglasnosti tj. nesaglasnost se ne otkriva pre nego što se stvarno pojavi.

Uopšte, za poboljšanje efikasnosti algoritama zasnovanih na pristupu generiši i testiraj u principu postoje dva pravca:

- Generator vrednosti je obavešten i pametan; takav generator, na primer, generiše vrednosti promenljivih na način koji obezbeđuje da verovatnoća pojavljivanja konflikta u fazi testiranja bude minimalna. Ovo je osnovna ideja algoritama zasnovanih na lokalnom pretraživanju.
- Generator je spojen sa testerom; na primer, vrednosti se testiraju na osnovu ograničenja čim promenljive dobiju određene vrednosti; ovo je pristup koji se koristi u metodi vraćanja po tragu.

11. TEHNIKE PROVERE SAGLASNOSTI (KONZISTENCIJE)

Jedan od osnovnih pristupa za rešavanje CSP-a baziran je na uklanjanju nesaglasnih vrednosti iz domena promenljivih na osnovu postojećih ograničenja. Ove metode, koje su u suštini determinističke, zovu se tehnike provere saglasnosti ili tehnike konzistencije i one su prvi put predstavljene pri rešavanju problema označavanja prostora (Valcov algoritam) [8].

Primena tehnika konzistencije najčešće ne garantuje rešavanje problema i zato se kaže da one nisu potpune. To je razlog zbog koga se retko koriste same već se, kao što je napomenuto, kombinuju sa metodama pretraživanja.

Razlikuju se sledeće tri osnovne tehnike konzistencije:

- Konzistencija čvora NC (*Node Consistency*),
- Konzistencija luka (grane) AC (*Arc Consistency*),
- Konzistencija puta PC (*Path Consistency*).

Najjednostavnija je tehnika konzistencije čvora. Njome se iz domena promenljive uklanjaju vrednosti koje nisu saglasne sa razmatranim ograničenjima nad odnosnom promenljivom i proverava da li je domen promenljive prazan. Ako domen neke promenljive postane prazan, onda nije ispunjen uslov konzistencije čvora i to je znak da posmatrano potencijalno rešenje problema nije dopustivo.

Najšire korišćena tehnika konzistencije se zasniva na proveri saglasnosti vrednosti parova promenljivih između kojih postoji ograničenje i naziva se tehnika konzistencije luka, lučna konzistencija ili konzistencija para.

Ako između dve promenljive V_i i V_j u grafu ograničenja postoji luk (V_i,V_j), to znači da su one povezane ograničenjem. Treba proveriti da li za tekuće domene promenljivih V_i i V_j važi to ograničenje. Ako važi, onda je ispunjena saglasnost tog luka (para promenljivih). Ako za svaki luk važi ograničenje, onda je zadovoljena lučna konzistencija (konzistencija parova).

Primenom tehnike konzistencije luka iterativno se uklanjaju vrednosti iz domena promenljivih koje su nesaglasne sa postojećim binarnim ograničenjima.

Kaže se da je put konzistentan ako su sve grane koje pripadaju putu konzistentne, odnosno, ako su sva binarna ograničenja na putu zadovoljena. Primenom tehnika konzistencije puta u jednoj iteraciji može se ukloniti više nesaglasnih rešenja, odnosno, prostor pretraživanja može se više suziti nego u jednoj iteraciji primene tehnike konzistencije luka.

12. PROSTIRANJE OGRANIČENJA

Iako metode pretraživanja teorijski garantuju rešavanje problema CSP, kao što to garantuju i neke tehnike provere saglasnosti, ni jedna od njih se, po pravilu, ne koristi zasebno i nezavisno. Uobičajeni pristup rešavanju problema CSP je kombinacija navedenih pristupa.

Metoda vraćanja po tragu, kao osnovna metoda implicitne enumeracije, odlikuje se integracijom generatora vrednosti promenljivih i testera koji proverava saglasnost generisanih vrednosti čim se one odrede. U cilju prevazilaženja ili bar ublažavanja nedostataka, razvijeno je više različitih unapređenja, takozvanih shema. Pri tome se razlikuju dva osnovna pristupa koja služe i za klasifikaciju shema. Jednu klasu čine sheme provere saglasnosti koje se zasnivaju na „pogledu unazad" (*look backward*) [9], a drugu one koje se zasnivaju na „pogledu unapred" (*look forward*) [10]. U ovim shemama se u jednoj iteraciji konstrukcije rešenja traži vrednost promenljive tako da nova dodela bude saglasna sa ranije dodeljenim vrednostima promenljivih.

U shemama sa pogledom unazad vrše se provere saglasnosti promenljivih čije su vrednosti već određene. Standardni algoritam vraćanja po tragu je najjednostavniji predstavnik ove sheme. Naprednije sheme su: provera unazad (*back checking*) [11], skakanje unazad (*back jumping*) i obeležavanje unazad (*back marking*) [9].

Za sve sheme sa pogledom unazad važi da se nesaglasnost kasno otkriva, tj. da se problemi nesaglasnosti rešavaju tek posle njihovog otkrivanja. Tim pristupom se ne sprečava

buduće pojavljivanje nesaglasnosti. Sheme sa pogledom unapred se koriste da bi se izbegle moguće buduće nesaglasnosti ili barem smanjila verovatnoća njihovog pojavljivanja.

Pristup u kome se u svakoj iteraciji konstrukcije rešenja proveravaju saglasnosti svih parova promenljivih naziva se „potpuni pogled unapred" (*full look ahead*) ili održavanje saglasnosti parova (*maintaining arc consistency*). U nekim slučajevim je algoritam potpunog pogleda unapred manje efikasan od hronološkog algoritma vraćanja po tragu. Algoritam proveravanja unapred (*forward checking*) je najjednostavniji primer strategije pogleda unapred. U njemu se proverava saglasnost parova (lučna konzistencija) između promenljivih kojima su dodeljene vrednosti (instancirane promenljive) i promenljivih kojima još nisu dodeljene vrednosti (neinstancirane promenljive). Na primer, kada se tekućoj promenljivoj dodeli neka vrednost, sve vrednosti iz domena „budućih" promenljivih koje nisu saglasne sa ovim dodeljivanjem se (privremeno) uklanjaju iz domena. Time se postiže da za svaku buduću (neinstanciranu) promenljivu postoji bar jedna vrednost u njenom domenu koja je saglasna sa vrednostima instanciranih promenljivih; u suprotnom, ako bi domen neke buduće promenljive postao prazan skup, treba se odmah vratiti po tragu. Proveravanje unapred zahteva više računarskih operacija od hronološkog algoritma vraćanja po tragu ali je i pored toga skoro uvek od njega efikasnije.

13. OSTALI PRISTUPI ZA POVEĆANJE EFIKASNOSTI PRETRAŽIVANJA

U prethodnom tekstu su opisane varijacije osnovnog opšteg pristupa rešavanju problema CSP koje se zasnivaju na kombinaciji pretraživanja i proveravanja saglasnosti parcijalnog rešenja sa postojećim ograničenjima. Ovde će se navesti nekoliko heurističkih pristupa povećanju efikasnosti postupka.

U svim algoritmima pretraživanja se na početku zadaje redosled promenljivih, kao i redosled vrednosti koje će se dodeljivati promenljivima u postupku procesa konstrukcije rešenja. Pokazano je da ove odluke često značajno utiču na efikasnost pretraživanja.

Određivanje redosleda ili uređenje promenljivih može biti statičko i dinamičko. U statičkom uređenju redosled se utvrđuje unapred, pre započinjanja konstrukcije rešenja. U dinamičkom uređenju se redosled promenljivih proverava i menja u toku procesa konstrukcije rešenja. Algoritmi za uređenje promenljivih su, po pravilu, heuristički.

Često korišćena heuristika za uređenje je da se na prva mesta stave promenljive za koje postoje veći izgledi da će se najranije pokazati da su dodeljene vrednosti nesaglasne. Ovo pravilo se naziva „najpre neuspeh". To, u opštem slučaju, daje prednost promenljivima sa malim domenom, tj. promenljivima čiji domeni imaju mali broj različitih vrednosti. Objašnjenje za primenu ove heuristike je jednostavno: što se pre otkrije nesaglasnost, veća je šansa da će deo ukupnog skupa pretraživanja koji se eliminiše na osnovu otkrivene nesaglasnosti biti veći. U slučaju kada je broj vrednosti u domenima više promenljivih isti, može se primeniti heuristično pravilo da se najpre ispituju promenljive koje se pojavljuju u više ograničenja.

Delimično eliminisanje potrebe za vraćanjem po tragu je drugi mogući pristup povećanju efikasnosti metode pretraživanja. Naime, u nekim problemima se može otkriti da redosled nekoliko promenljivih otklanja mogućnost da se na tom delu desi vraćanje po tragu.

Određivanje redosleda po kome će se ispitivati vrednosti izvesne promenljive takođe može da ima veliki uticaj na efikasnost algoritma pretraživanja. Najčešće korišćeno heuristično pravilo u ovom slučaju je „najpre uspeh", tj. izabrati vrednost promenljive za koju se smatra da ima najviše izgleda da ne izazove nesaglasnost u proverama koje će uslediti.

14. OPTIMIZACIJA OGRANIČENJA

Problemi zadovoljenja ograničenja razlikuju se od problema optimizacije po tome što matematički modeli ovih drugih, pored ograničenja, sadrže i kriterijumsku funkciju koja određuje kavalitet dopustivog rešenja. Da bi se našlo optimalno rešenje, treba najpre rešiti problem CSP, a onda iz skupa svih dopustivih rešenja problema CSP izabrati najbolje.

Podsetimo da se egzaktni algoritmi, koji garantuju dobijanje optimalnog rešenja, zasnivaju na opštem principu implicitne enumeracije i da je od njih metoda grananja i ograničavanja jedna od najviše korišćenih. Pored egzaktnih, koriste se i približni algoritmi zasnovani na metaheuristikama i heurističkim pristupima karakterističnim za konkretne klase problema.

15. GLOBALNA OGRANIČENJA

Globalno ograničenje je termin koji se koristi da opiše složeno ograničenje koje kao svoj određujući parametar ima izvestan broj promenljivih. Globalno ograničenje zamenjuje veliki broj jednostavnih ograničenja koja se često pojavljuju u realnom primeru. Uobičajeno se kaže da se globalno ograničenje odnosi na sistem kao celinu, odnosno na sve promenljive sistema koje su njime obuhvaćene. Na taj način se značajno olakšava modeliranje problema. Globalna ograničenja se mogu grupisati prema oblastima primene u kojima se najčešće koriste.

Kao ilustraciju ovde se navodi samo par globalnih ograničenja. Često korišćeno globalno ograničenje, koje skoro da ne zavisi od oblasti primene, jeste alldifferent(v), gde je v skup promenljivih koje moraju biti međusobno različite. U rešavanju problema planiranja i raspoređivanja resursa potrebnih za pripremu i realizaciju proizvodnje postoji veliki broj globalnih ograničenja [12,13,14]. Jedno tipično ograničenje je cumulative([S1,...,Sn], [D1,...,Dn],[R1,...,Rn],L), gde su [S1,...,Sn], [D1,...,Dn] i [R1,...,Rn], neprazni nizovi domena nekih promenljivih, a L neki prirodan broj. Sa Si su obeležena vremena početka odgovarajućih aktivnosti, Di su njihova trajanja, a Ri količina potrebnih resursa za njihovu realizaciju. Broj L predstavlja raspoloživu ukupnu količinu resursa. Ovo

ograničenje obezbeđuje da u svakom trenutku izvršavanja aktivnosti, broj angažovanih resursa ne bude veći od zadatog broja L.

Za određeno globalno ograničenje razvija se posebni algoritam koji bi trebalo da o njemu „brine" na najbolji način. Na taj način, modeliranje pomoću globalnih ograničenja, u odnosu na ono koje se oslanja na algoritme opšte namene za prostiranje ograničenja, postaje efikasnije i sa stanovišta analitičara i sa stanovišta efikasnosti izvršavanja. Sve su to razlozi na osnovu kojih se smatra da je uvođenje globalnih ograničenja najvažniji i najuspešniji rezultat u razvoju tehnologije korišćenja solvera i sistema zaključivanja.

Mnogo globalnih ograničenja je razvijeno i programirano u savremenim objektno-orijentisanim programskim jezicima. Njihov broj je reda nekoliko stotina pa i više od hiljadu. I pored toga, globalna ograničenja su još uvek interesantna oblast za teorijska i tehnološka istraživanja kao i obavezan uslov uspešne praktične primene.

16. PREOGRANIČENI PROBLEMI

U praksi se javljaju problemi sa velikim brojem ograničenja za koje ne postoji ni jedno dopustivo rešenje. Takvi problemi se nazivaju preograničeni (*over-constrained*), protivurečni ili kontradiktorni.

Za rad sa problemima za koje je veoma teško utvrditi da li uopšte postoji dopustivo rešenje, jer su možda preograničeni, koristi se nekoliko pristupa od kojih su najpoznatija sledeća dva: delimično (parcijalno) zadovoljenje ograničenja [15] i hijerarhija ograničenja [16]. U oba pristupa na početku treba razdvojiti obavezujuća, tvrda ograničenja, koja se moraju zadovoljiti i koja su posledica prirode problema, od poželjnih, mekih ograničenja, koja su izraz želja vlasnika problema.

17. ISTRAŽIVAČKI I TEHNOLOŠKI IZAZOVI

Podsticaji za dalja istraživanja u oblasti PO proizilaze iz mogućnosti njegove primene kao i iz već dokazanih uspešnih rezultata u rešavanju praktičnih problema. Teoretičarima i proizvođačima softvera neprekidno se postavljaju sve strožiji zahtevi koji se, u grubom, mogu svesti na sledeća dva:

- Učiniti dostupnim već postignute teorijske rezultate što većem broju potencijalnih krajnjih korisnika,
- Omogućiti efikasno rešavanje što većeg broja realnih problema.

Prvi od navedenih zahteva je, pored ostalog, vezan za rešavanje pitanja interfejsa čoveka i računara [17,18]. Treba omogućiti predstavljanje realnih problema odlučivanja na jednostavan i krajnjem korisniku razumljiv način. S obzirom da se često pojavljuju ograničenja koje radi rešavanja na računaru treba predstaviti složenim matematičkim obrascima, jasno je da će ovo pitanje dugo biti otvoreno. U vezi sa drugim zahtevom treba uočiti protivurečnost između želje da se problemi rešavaju u realnom vremenu, što nekad znači zahtev za dobijanje rešenja u vremenu reda sekunde ili minute, i potencijalne osobine razmatranog problema da pripada klasi NP-kompletnih problema. Na osnovu navedenog se može zaključiti da će izazovi u vezi sa rešavanjem ovih pitanja trajati dosta dugo i teško je zamisliti da će u doglednoj budućnosti biti sasvim prevaziđeni.

Važno je uočiti i nespornu činjenicu da je već došlo do prirodnog delimičnog spajanja oblasti PO i kombinatorne optimizacije. Ovom integracijom su se u svakoj od oblasti postigla izvesna unapređenja.

Koncept globalnih ograničenja pruža velike mogućnosti sa aspekta planera, odnosno krajnjeg korisnika. Savremeni interfejsi se prave na osnovu zahteva da krajnji korisnik treba da se pri planiranju služi jednostavnim metodama modeliranja dok su algoritmi rešavanja problema implementirani u solverima. Na taj način se izbegava zahtev da planer bude stručnjak u programiranju ili ekspert za algoritme. To se postiglo razvojem programskih biblioteka koje omogućavaju primenu globalnih ograničenja u objektno-orijentisanom okruženju.

Popularnost programskog jezika Java dovela je do pojave velikog broja biblioteka koje omogućavaju kombinovanje prednosti ovog okruženja sa prednostima korišćenja metoda PO.

Proučavanje interakcija raznih metoda rešavanja ograničenja je jedno od najizazovnijih problema istraživačima i programerima. U ovoj oblasti su već postignuti značajni rezultati razvojem hibridnih algoritama koji kombinuju brojne tehnike rešavanja ograničenja.

Sledeće interesantno područje proučavanja je saradnja, kolaboracija, solvera. Kombinacije tehnika zadovoljenja ograničenja sa tradicionalnim metodama operacionih istraživanja, kao što je celobrojno programiranje, su dodatni izazov trenutnih i budućih istraživanja. Poslednji, ali ne i najmanji izazov je paralelizam i konkurentno rešavanje ograničenja. U ovim sistemima se kao veoma obećavajuća pojavljuje tehnologija više agenata (*multi-agent technology*).

Kako je i dalje efikasnost postojećih algoritama nepredvidiva za najveći broj konkretnih problema klase NP, intuicija i iskustvo istraživača ostaju najvažniji faktor odluke kada i kako koristiti ograničenja.

Veliki problem u primeni PO je stabilnost razmatranog modela ograničenja. Naime, čak i male promene u modelu ili u podacima mogu voditi ka dramatičnim promenama u performansama korišćenih algoritama. Odeđivanje opsega vrednosti parametara koji obezbeđuju stabilno (vremenski efikasno) rešavanje problema ostaje kao otvoreno pitanje.

Posebna klasa problema je izbor prave tehnike zadovoljenja ograničenja za probleme specifične strukture. Ponekad je potpuno nabrajanje ili hronološka metoda vraćanja po tragu, efikasnije od mnogo skupljeg algoritma prostiranja ograničenja.

Najveći deo vremena u praktičnim primenama metoda programiranja ograničenja troši se na projektovanju planova koje potom treba dosledno izvršavati. U praksi se, po pravilu, u toku realizacije dešavaju nepredviđene okolnosti tako da se neminovno mora odstupiti od prvobitnog plana. Modelujući jezici PO omogućavaju lako dodavanje novih ograničenja ili izmenu postojećih. Međutim, to ne garantuje rešavanje novog

problema u realnom vremenu, pogotovo ako se traži optimalno rešenje. Zato je neophodno napraviti analizu razmene, odnosno potražiti kompromis između zahteva za optimalnost i zahteva za efikasnost rešavanja. To za posledicu može da ima prihvatanje rešenja za koje se ne može tvrditi da je najbolje moguće ali je zato dobijeno u realnom vremenu. U tom smislu, specifičan je problem upravljanja troškovima dobijanja rešenja, pogotovo kada je teško poboljšati početno rešenje jer i mala poboljšanja mogu da oduzmu mnogo više vremena od dobijanja početnog rešenja.

LITERATURA

[1] F. Rossi, P. van Beek, T. Walsh, editors. *Handbook of Constraint Programming*. Elsevier Science, Amsterdam, 2006.

[2] K. Apt. *Principles of Constraint Programming*. Cambridge University Press, New York, 2009.

[3] R. Bartak, M.A. Salido, F. Rossi. New trends on constraint satisfaction, planning, and scheduling: A survey. *The Knowledge Engineering Review*, 25(3):247279, 2010.

[4] G. Dantzig. Programming in a linear structure. *Econometrica*, 17:73, 74,1949.

[5] K. Marriot, P. Stuckey. *Programming with Constraints: An Introduction*. MIT Press, Cambridge, 1998.

[6] R. Bartak, D. Toropila. Solving sequential planning problems via constraint satisfaction. *Fundamenta Informaticae*, 99(2):125-145, 2010.

[7] S. Krčevinac, M. Čangalović, V. Kovačević-Vujčić, M. Martić, M. Vujošević. *Operaciona istraživanja*. FON, Beograd, 2004.

[8] D.L. Walz. Understanding line drawings of scenes with shadows. In P. H. Winston, editor, *Psychology of Computer Vision*, pages 19-92, New York, 1975. McGraw-Hill.

[9] R.M. Haralick G.L. Eliot. Increasing tree search efficiency for constraint satisfaction problems. *Articial Intelligence*, (14):263314, 1980.

[10] B. Nadel. Tree search and arc consistency in constraint satisfaction algorithms. In *IEEE Symposium on Logic Programming*, Boston, 1985. IEEEl.

[11] J. Gashing. Performance measurement and analysis of certain search algorithms. Carnegie-Mellon University - CMU-CS-79-124, 79(124), 1979.

[12] M. Strak M. Vujošević. Planiranje resursa primenom globalnih ograničenja u programiranju ograničenja. *Zbornik radova SPIN* 07, V *skup privrednika i naučnika*, pages 207-211, Beograd, 2007. FON.

[13] M. Strak M. Vujošević. Analiza modela poslovnih procesa pomocu logičkog programiranja sa ograničenjima. *Zbornik radova SPIN* 08, VI *skup privrednika i naučnika*, pages 300-304, Beograd, 2008. FON.

[14] M. Strak, M. Vujošević, T. Ognjanović. Pregled realizacija solvera ograničenja u Java programskom okruženju. *Zbornik radova YUINFO*, Kopaonik 9-12.03.2008, Beograd, 2008. FON, CD.

[15] E.C. Freuder R.J. Wallace. Partial constraint satisfaction. *Artifficial Intelligence*, 58(1):21-70, 1992.

[16] A. Borning. The programming language aspects of thinglab, a constraint-oriented simulation laboratory. *ACM Transactions on Programming Languages and Systems*, 3(4):252-387, 1981.

[17] M. Abu Gaben, S. Krčevinac, M. Vujošević. Integracija operacionih istraživanja i sajber infrastrukture. *Info M*, (24):53-58, 2007.

[18] M. Abu Gaben, S. Krčevinac, M. Vujošević. Modelujući sistemi u optimizaciji. *Istraživanja i projektovanja za privredu* (*IIPP*), (18):27-46, 2007.

Prof. dr Mirko Vujošević,
Univerzitet u Beogradu, Fakultet organizacionih nauka; član Akademije inženjerskih nauka Srbije,
mirkov@fon.bg.ac.rs ,
Oblasti interesovanja: operaciona istraživanja, metode optimizacije, višekriterijumska optimizacija, celobrojno programiranje, programiranje ograničenja, meko računarstvo, pouzdanost i upravljanje rizikom

info m

UPUTSTVO ZA PRIPREMU RADA

1. Tekst pripremiti kao Word dokument, A4, u kodnom rasporedu 1250 latinica ili 1251 ćirilica, na srpskom jeziku, bez slika. Preporučen obim – oko 10 strana, single prored, font 11.
2. Naslov, abstakt (100-250 reči) i ključne reči (3-10) dati na srpskom i engleskom jeziku.
3. Jedino formatiranje teksta je normal, bold, italic i bolditalic, VELIKA i mala slova (tekst se naknadno prelama).
4. Mesta gde treba ubaciti slike, naglasiti u tekstu (Slika1...)
5. Slike pripremiti odvojeno, VAN teksta, imenovati ih kao u tekstu, radi identifikacije, u sledećim formatima: rasterske slike: jpg, tif, psd, u rezoluciji 300 dpi 1:1 (fotografije, ekranski prikazi i sl.), vektorske slike – cdr, ai, fh,eps (šeme i grafikoni).
6. Autor(i) treba da obavezno priloži svoju fotografiju (jpg oko 50 Kb), navede instituciju u kojoj radi, kontakt i 2-4 oblasti kojima se bavi.
7. Maksimalni broj autora po jednom radu je 5.

Redakcija časopisa Info M