Shema dva posmatrana literala u DPLL SAT rešavačima

Seminarski rad u okviru kursa
Automatsko rezonovanje
Matematički fakultet

Staša Đorđević 1007/2025

24. februar 2026.

Sažetak

TODO sazetak na kraju - rezime rada, max 100 reci

1 Uvod

Automatsko rezonovanje je oblast računarske nauke koja se bavi razvojem metoda i algoritama za automatsko izvođenje zaključaka iz formalnih logičkih sistema. Jedan od centralnih problema u ovoj oblasti je problem zadovoljivosti iskazne logike (SAT problem), koji se sastoji u određivanju da li postoji interpretacija kojom se formula iskazne logike može učiniti istinitom. SAT problem je fundamentalni NP-kompletan problem i predstavlja osnovu za mnoge primene u verifikaciji softvera, veštačkoj inteligenciji i optimizaciji.

Jedna od najefikasnijih metoda za rešavanje SAT problema je DPLL algoritam (Davis–Putnam–Logemann–Loveland), koji predstavlja unapređenje klasičnog DP (Davis–Putnam) postupka. DPLL algoritam kombinuje sistematsko pretraživanje sa jednostrukim dedukcijama na osnovu jedinica (unit propagation) i detekcijom konflikata, čime omogućava efikasno ispitivanje velikih formula. Ključni deo DPLL algoritma predstavlja izbor literala i način na koji se donose odluke tokom pretraživanja, jer od toga zavisi brzina pronalaženja rešenja ili dokaza da rešenje ne postoji.

U okviru ovog rada fokus je stavljen na analizu i implementaciju *sheme dva posmatrana literala* unutar DPLL SAT rešavača. Ova shema predstavlja strategiju odabira literala, koja istovremeno razmatra dva kandidata za sledeći izbor, čime se povećava verovatnoća bržeg pronalaženja rešenja i smanjuje broj potrebnih grananja pretraživanja. Razumevanje i pravilna implementacija ove sheme može značajno poboljšati performanse DPLL algoritma, posebno na velikim i složenim SAT instancama.

Ostatak ovog rada organizovan je na sledeći način. U poglavlju 2 uvodimo osnovne definicije i notaciju iz iskazne logike, kao i formalni opis SAT problema. Poglavlje 3 sadrži detaljan opis DPLL algoritma i sheme dva posmatrana literala. Poglavlje 4 opisuje implementaciju ove metode u izabranom programskom jeziku Python, uključujući ključne klase i funkcije. Na kraju, poglavlje 5 daje osvrt na rezultate rada i diskusiju o efikasnosti primenjene strategije.

2 Osnove

U ovom poglavlju uvodimo osnovne pojmove i notaciju iz iskazne logike koji će biti korišćeni u ostatku rada.

2.1 Sintaksa iskazne logike

Iskazna logika se bazira na *atomskim iskazima*, koji predstavljaju osnovne logičke promenljive, i na *logičkim veznicima* koji povezuju iskaze u složenije formule.

Formule iskazne logike definišu se rekurzivno:

- Iskazna slova i logičke konstante su iskazne formule.
- Ako su F i G formule, onda su i (F), $\neg F$ (negacija), $F \wedge G$ (konjunkcija), $F \vee G$ (disjunkcija), $F \rightarrow G$ (implikacija) i $F \leftrightarrow G$ (ekvivalencija) formule.

2.2 Semantika iskazne logike

Semantika iskaznih formula definiše se pomoću pojma valuacije i interpretacije. Valuacija je funkcija $v : P \rightarrow \{0, 1\}$ koja svakom atomu iz skupa iskaznih slova P dodeljuje logičku vrednost.

Svaka (potpuna) valuacija v indukuje funkciju $I_v : F_P \rightarrow \{0, 1\}$ na skupu svih iskaznih formula nad P (u oznaci F_P), koju nazivamo interpretacija. Ova funkcija svakoj formuli pridružuje istinitosnu vrednost i definiše se rekurzivno na sledeći način:

- $I_v(p) = 1$ akko je $v(p) = 1$;
- $I_v(\top) = 1$, $I_v(\bot) = 0$;
- $I_v(\neg F) = 1$ akko je $I_v(F) = 0$;
- $I_v(F_1 \wedge F_2) = 1$ akko je $I_v(F_1) = 1$ i $I_v(F_2) = 1$;
- $I_v(F_1 \vee F_2) = 1$ akko je $I_v(F_1) = 1$ ili $I_v(F_2) = 1$;
- $I_v(F_1 \Rightarrow F_2) = 1$ akko je $I_v(F_1) = 0$ ili $I_v(F_2) = 1$;
- $I_v(F_1 \Leftrightarrow F_2) = 1$ akko je $I_v(F_1) = I_v(F_2)$.

2.3 Zadovoljivost i tautologija

Formula F je *zadovoljiva* ako postoji barem jedna interpretacija I takva da je $I(F) = \text{True}$. Formula je *tautologija* ako je $I(F) = \text{True}$ za sve moguće interpretacije I. S druge strane, formula je *nezadovoljiva* ako ne postoji nijedna interpretacija koja je čini istinitom.

2.4 SAT problem

Problem zadovoljivosti iskazne logike (SAT problem) je zadatak određivanja da li je data formula F zadovoljiva. SAT problem je NP-kompletan i predstavlja osnovni problem u oblasti automatskog rezonovanja.

U praksi, često se koriste formule u *konjunktivnoj normalnoj formi* (CNF), gde je formula predstavljena kao konjunkcija klauza, a svaka klauza je disjunkcija literala. Literal je ili atomski iskaz ili njegova negacija.

3 Opis metode

U ovom poglavlju biće opisan DPLL algoritam za rešavanje SAT problema, zajedno sa ključnim mehanizmima koji utiču na njegovu efikasnost. Posebna pažnja biće posvećena jediničnoj propagaciji (engl. unit propagate) i šemi dva posmatrana literala, koja predstavlja standardnu optimizaciju u savremenim SAT rešavačima.

3.1 DPLL algoritam

DPLL algoritam predstavlja rekurzivnu proceduru pretrage za određivanje zadovoljivosti formule u konjunktivnoj normalnoj formi (KNF). Za razliku od originalne DP procedure, DPLL ne eliminiše promenljive primenom rezolucije, već konstruiše parcijalnu valuaciju i sistematski ispituje moguće dodele vrednosti promenljivama.

Neka je F formula u KNF obliku, a M parcijalna valuacija. Algoritam se može opisati sledećim koracima:

1. Ako postoji klauza $C \in F$ koja je netačna u parcijalnoj valuaciji M, vraća se UNSAT.
2. Ako su sve promenljive iz F dodeljene i nijedna klauza nije netačna, vraća se SAT.
3. Ako postoji jedinična klauza, primenjuje se jedinična propagacija.
4. Ako postoji čist literal, dodeljuje mu se odgovarajuća vrednost.
5. U suprotnom, bira se nedefinisani literal i algoritam se rekurzivno poziva nad dve proširene valuacije: $M \cup \{l\}$ i $M \cup \{\neg l\}$.

3.2 Jedinična propagacija (Unit Propagation)

Klasični DPLL. U naivnom DPLL algoritmu koji transformiše formulu, jedinična klauza je klauza sa tačno jednim literalom. Dodeljivanjem vrednosti koja zadovoljava taj literal, sve klauze koje ga sadrže se uklanjaju, a iz preostalih klauza se uklanja njegova negacija. Postupak se ponavlja dok postoje jedinične klauze.

DPLL sa parcijalnom valuacijom. U implementacijama zasnovanim na parcijalnoj valuaciji M, klauza je jedinična ako su svi literali netačni, osim jednog koji je nedefinisan. Umesto transformacije formule, valuacija se proširuje sa tim literalom, a status klauza se ažurira u odnosu na novu dodelu.

Shema dva posmatrana literala. Da bi se izbegao obilazak svih klauza, koristi se shema dva posmatrana literala. U svakoj klauzi dva literala se aktivno prate. Ako jedan postane netačan, pokušava se zamena drugim literalom. Ako zamena nije moguća:

- klauza je konfliktna ako je i drugi literal netačan,
- klauza je jedinična ako je drugi literal nedefinisan.

Ova optimizacija značajno smanjuje broj provera potrebnih za detekciju konflikta i jediničnih klauza.

3.3 Eliminacija čistih literala (Pure Literal Rule)

Klasični DPLL. Literal l je čist ako njegova negacija $\neg l$ ne postoji ni u jednoj klauzi formule. Dodeljivanjem vrednosti koja čini l istinitim, sve klauze koje ga sadrže se uklanjaju, što pojednostavljuje formulu.

DPLL sa parcijalnom valuacijom. Proverava se da li postoji literal l koji još nije dodeljen i čija negacija nije prisutna u nezadovoljenim klauzama. Ako postoji, valuacija se proširuje sa l, a klauze koje ga sadrže se automatski smatraju zadovoljenim. U modernim rešavačima zasnovanim na CDCL paradigmi, eliminacija čistih literala se često izostavlja jer doprinos efikasnosti postaje manji u poređenju sa jediničnom propagacijom i učenjem klauza.

3.4 Split (Grananje)

Kada više nije moguće primeniti jediničnu propagaciju niti eliminaciju čistih literala, bira se nedefinisani literal l i razmatraju se dve mogućnosti: $M \cup \{l\}$ i $M \cup \{\neg l\}$. Ovaj korak uvodi rekurzivnu pretragu prostora svih mogućih valuacija i ključan je za sistematsko ispitivanje formule. Efikasnost algoritma u velikoj meri zavisi od strategije izbora literala za split.

3.5 Shema dva posmatrana literala

Naivna implementacija DPLL algoritma zahteva proveru svih klauza nakon svake izmene valuacije. Shema dva posmatrana literala uvodi optimizaciju propagacije jediničnih klauza i detekcije konflikata. Svaka klauza aktivno prati dva literala, a klauza se proverava samo ako jedan od njih postane netačan. Ako zamena nije moguća, detektuje se konflikt ili klauza postaje jedinična.

3.6 Efekat optimizacije

Primena sheme dva posmatrana literala dovodi do značajnog poboljšanja performansi. Jedinična propagacija se realizuje uz znatno manji broj provera, a amortizovana (??) složenost postupka postaje linearna u odnosu na broj pojavljivanja literala. Ova tehnika je standardna komponenta modernih SAT rešavača i ključni je element njihove praktične efikasnosti. [3]

4 Implementacija

U ovom poglavlju treba dati detalje implementacije. Treba navesti u kom programskom jeziku je metoda implementirana, kako je kôd organizovan, koje su ključne funkcije/klase/metode i koja je njihova uloga.

Takodje je bitno dati upustvo za prevodjenje i pokretanje projekta, kao i navesti softver koji je potreban za prevodjenje i pokretanje programa (prevodilac, alati, dodatne bibiloteke i sl.).

5 Zaključak

Literatura

[1] Prezentacija sa kursa *Automatsko rezonovanje*, Milan Banković, Matematički fakultet, dostupno na: `https://poincare.matf.bg.ac.rs/~milan.bankovic/preuzimanje/ar/ar-iskazna-logika.pdf`

[2] Filip Marić, *Flexible Implementation of SAT Solvers*, dostupno na: `https://poincare.matf.bg.ac.rs/~filip/phd/sat-flexible-implementation.pdf`

[3] Himanshu Jain i Edmund M. Clarke, *Efficient SAT Solving for Non-Clausal Formulas Using DPLL, Graphs, and Watched Cuts*, u: *Proceedings of the 46th Annual Design Automation Conference (DAC '09)*, stranice 563–568, ACM, 2009. DOI: `https://doi.org/10.1145/1629911.1630057`, dostupno na: `https://dl.acm.org/doi/epdf/10.1145/1629911.1630057`

[4] Biere, Armin, Marijn Heule, and Hans van Maaren, eds. Handbook of satisfiability. Vol. 185. IOS press, 2009.

[5] Ganzinger, Harald, et al. *DPLL (T): Fast decision procedures.* Computer Aided Verification: 16th International Conference, CAV 2004, Boston, MA, USA, July 13-17, 2004. Proceedings 16. Springer Berlin Heidelberg, 2004.

[6] https://dl.acm.org/doi/epdf/10.1145/1629911.1630057

[7] Nieuwenhuis, Robert, Albert Oliveras, and Cesare Tinelli. *Solving SAT and SAT modulo theories: From an abstract Davis–Putnam–Logemann–Loveland procedure to DPLL (T).* Journal of the ACM (JACM). 2006.